Центральноукраїнський національний технічний університет
Механіко-технологічний факультет
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2024 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти

на тему

“Дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу”

КбПЗ_2024

Виконав здобувач вищої освіти
ІІ курсу, групи КН-23М
ОПП «Комп’ютерні науки»
спеціальності 122 «Комп’ютерні науки»
_____ Лагута І.А.
«____» _____ 2024 р.

Керівник проекту
кандидат технічних наук
_____ Буравченко К.О.
«____» _____ 2024 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет
Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Рівень вищої освіти *магістр*
Галузь знань _12 _____*"Інформаційні технології"*_____
Спеціальність 122 *"Комп'ютерні науки"*
Освітньо-професійна (освітньо-наукова) програма _*"Комп'ютерні науки"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2024 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Лагуті Ігорю Анатолійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу*

2. Керівник роботи *Буравченко Костянтин Олегович, канд. техн. наук*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 18-13 від 07.08.2024 року

3. Строк подання студентом роботи до захисту *2.12.2024 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу*

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

1. Призначення та область використання.	*6. Наукова новизна.*
2.Перегляд аналогічних існуючих систем.	*7. Маркетингове та економічне обґрунтування ІТ-проєкту.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки.*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

КОПІЯ 2024

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Доренська А.О.	05.10.2024	14.11.2024
Охорона праці	Марченко К.М., к.т.н., доцент	06.10.2024	16.11.2024

7. Дата видачі завдання « 6 » вересня 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2024 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2024 р.	
3.	Розробка моделі компонента	20.10.2024 р.	
4.	Розробка структур даних	25.10.2024 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2024 р.	
6.	Програмування алгоритмів	10.11.2024 р.	
7.	Розрахунок економічної ефективності	13.11.2024 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2024 р.	
9.	Оформлення ПЗ	17.11.2024 р.	
10.	Попередній захист роботи	2.12.2024 р.	

КОПЗ_2024

Дата видачі завдання
« 6 » вересня 2024 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2024 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Лагута І.А. Дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу. 122 Комп'ютерні науки. Центральноукраїнський національний технічний університет. Кропивницький. 2024.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Метою розробки є дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Об'єктом дослідження є процес архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Предметом дослідження є методи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Методи дослідження базуються на методах хмарних технологій, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ з ОС Windows 10/11.

Програму розроблено в середовищі Visual C++.

Ключові слова: Комп'ютерні науки, інформаційні та комп'ютерні системи, віртуальний офіс

КОПЗ-2024

ABSTRACT

Laguta I.A. Research and software implementation of the system of architectural solutions of information and computer systems of the virtual office. 122 Computer Science. Central Ukrainian National Technical University. Kropyvnytskyi. 2024.

In this graduation thesis for the second (master's) level of higher education, software was developed, which is intended for the system of architectural solutions of information and computer systems of the virtual office.

The goal of the development is the research and software implementation of the system of architectural solutions for information and computer systems of the virtual office.

The object of research is the process of architectural decision of information and computer systems of a virtual office.

The subject of the study is the methods of architectural solution of information and computer systems of the virtual office.

Research methods are based on methods of cloud technologies, methods of mathematical statistics, methods of software development.

The result of the work is the software implementation of the architectural solution system of the information and computer systems of the virtual office.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

A convenient user interface has been developed. Instructions for working with software tools are provided.

The program can be used on a PC with Windows OS 10/11.

The program was developed in the Visual C++ environment.

Keywords: Computer science, information and computer systems, virtual office

КОПЗ-2024

ЗМІСТ

КОПІЯ 2024

КбПЗ_2024

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АТМ	–	Автоматизована телефонна мережа
ІТ	–	Інформаційні технології
ПЗ	–	Програмне забезпечення

КбПЗ_2024

ВСТУП

Актуальність теми. Хмарна модель швидко завойовує популярність у різних сферах бізнесу. Компанії використовують сервіси систем електронної пошти, обміну повідомленнями, телефонії, відео-конференц-зв'язку й бізнес-додатків, зберігають у ЦОД провайдерів документи й резервні копії, розгортають у них додатки й навіть IT-інфраструктури. В останні два роки темпи розвитку українського ринку хмарних сервісів були досить високими. Поки він займає лише близько 1% усього вітчизняного ринку IT, однак протягом найближчих 2-3 років його середньорічний ріст може скласти майже 30% – і це на тлі стагнації або спаду в більшості інших сегментів IT.

З недавнього часу набирає популярність послуга «віртуальний офіс»: для ведення бізнесу з мінімальними початковими капіталовкладеннями можна організувати повноцінний «офіс у хмарі» і вибрати оптимальний набір послуг. Використання ресурсів хмари дозволяє не тільки зменшити інвестиційні витрати на придбання програмного забезпечення, серверного або телекомунікаційного встаткування, але навіть повністю відмовитися від власного IT-підрозділу й витрат по його змісту. Віртуальний офіс дає можливість скоротити вкладення в розробку, самостійно здійснювати впровадження й підтримку необхідних для ведення бізнесу додатків. Поряд з істотною економією важливими перевагами хмарних сервісів перед «локальними» рішеннями є швидке підключення послуг і можливість відмовитися від них без зволікання й втрат.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

КПЗ_2024

– Огляд існуючих систем архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

– Дослідження системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

– Програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Об’єктом дослідження є процес архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Предметом дослідження є методи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Методи дослідження базуються на методах хмарних технологій, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

– Розроблено вітчизняний продукт архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

КПЗ – 2024

Робота апробована на LVII Науково-технічній конференції здобувачів вищої освіти LV науково-технічної конференції «Наука в ЦНТУ: основні досягнення та перспективи розвитку» (2024 р.), основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №15.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

КБПЗ_2024

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Як правило, додатки хмарного віртуального офісу прості у використанні й інтуїтивно зрозумілі, підтримують різні засоби спільної роботи, дозволяють задіяти в бізнес-процесах персональні пристрої, а для роботи з ними не потрібно попереднього навчання. Хмарне середовище дає можливість інтегрувати інформацію з різних джерел даних і зробити її доступної для конкретних співробітників. Колективна робота з інформацією незалежно від відстані – сама коштовна якість віртуального офісу.

Провайдери пропонують підприємствам різноманітні хмарні додатки й сервіси, включаючи системи CRM і керування складом, бухгалтерські додатки, системи керування проектами, завданнями й продажами, конструктори Web-сайтів і інтегровану з офісними додатками хмарну телефонію. Такий набір сервісів допомагає автоматизувати бізнес-процеси, підвищити конкурентоспроможність компанії й масштабувати IT-сервіси в міру її росту. Замовники можуть повністю перенести свою IT-інфраструктуру в захищену хмару (IaaS), налаштувати резервне копіювання (BaaS), орендувати програмне забезпечення (Office 365 і ін.) по моделі SaaS, задіяти для розробки ПЗ віртуальні ресурси (PaaS) і т.д.

Віртуальний офіс (cloud office and collaboration productivity applications) стрімко набирає популярність по усьому світі. Згідно із прогнозом WinterGreen Research, в 2025 році обсяг цього ринку може досягти 21,6 млрд доларів. В Україні публічні хмари найбільш затребувані підприємствами роздрібної й оптової торгівлі, де велика частка SMB, організаціями зі сфери фінансових послуг, телекомунікаційними компаніями й ЗМІ.

КОПЗ_2024

1.2 Область застосування

Найбільш доступною хмарною послугою для корпоративних клієнтів залишаються віртуальні сервери – вони присутні в списку пропозицій дев'яти з кожних десяти провайдерів (92,3%). Разом з тим по популярності в замовників віртуальні сервери займають тільки другу позицію: цією послугою користуються лише ледве більше третини учасників опитування – 37,3%. Самий же «споживаний» сервіс (до нього прибігають 53,6% замовників) – електронну пошту – пропонують близько 58% провайдерів. І по поширеності ця послуга навіть не попадає в першу трійку пропозицій.

Другу позицію по поширеності серед пропозицій займають сервіси безпеки. За останні три роки ситуація в даній області помітно змінилася в результаті того, що спектр надаваних клієнтам послуг ІБ постійно розширюється й, судячи із планів замовників, попит на них буде рости. Третє місце ділять віртуальні робочі місця (VDI) і резервне копіювання в хмару (BaaS), що займають схожі позиції (4- 5-е місця) серед переваг клієнтів. Наступний за ними сервісу електронної пошти лише деяким уступають віртуальні АТМ, бухгалтерські системи й офісні додатки. Найменші частки в CRM і систем керування проектами, однак останні займають провідну позицію в планах провайдерів.

Спектр послуг провайдерів не обмежується перерахованими сервісами. Деякі компанії відносять до числа своїх хмарних пропозицій контактний центр, системи електронного документообігу, керування цифровим контентом (Digital Signage), відео^-конференц-зв'язок, корпоративний портал, Service Desk, системи для автоматизації логістичної діяльності й бізнес-аналітики, хоча вони не завжди попадають під визначення хмарних. Крім цього, згадуються сервіси PaaS – платформа розробки додатків (віртуальні сервери й ОС в оренду), резервна площадка в хмарі (DRaaS), сервіси Azure Backup, віртуальний ЦОД, послуги центра керування ІБ (Security Operations Center, SOC), хостинг PCI DSS, ІСПДН у

КОПЗ–2024

хмарі (ФЗ 152). Однак, якщо виключити різні види DRaaS, такі пропозиції одиничні.

Нерідко хмарні сервіси супроводжуються супутніми послугами, до числа яких ставляться міграція прикладних систем у хмару й реалізація різних сценаріїв переходу в хмарне середовище, автоматизація діяльності служби експлуатації IT, адміністрування й побудова приватних хмар (Dedicated Private Cloud), хмарні сховища, БД із хмари, Web-хостинг, пул ресурсів для організації власного віртуального центра обробки даних, сервіс для організації резервної інфраструктури на випадок відмови основних серверів клієнта (за рамками DRaaS), оренда віртуальних потужностей в українських ЦОД і ін. Багато подібних послуг підвищують цінність пропозиції провайдеру (Value Added Services, VAD) і його привабливість для клієнта, дозволяють замовникам вирішити ряд актуальних проблем.

Повідомляючи про свої наміри відносно розвитку й розширення спектра хмарних сервісів, провайдери найчастіше згадують системи керування проектами, що цілком відповідає очікуванням клієнтів. Частки інших сервісів приблизно однакові. Віртуальні ж сервери займають одне з останніх місць у планах виводу на ринок нових послуг, що говорить про насиченість даного сегмента. Далеко не всі провайдери мають намір реалізувати такі сервіси, як аварійне відновлення (DraaS) і ВКЗ. Примітно також, що лише деякі провайдери не збираються розширювати спектр сервісів. Тим часом, якщо судити по відповідях замовників, найбільшого росту попиту можна чекати на сервіси резервного копіювання в хмару (BaaS), системи керування проектами, офісні додатки, віртуальні робочі місця й сервіси безпеки.

Пропоновані багатьма провайдерами сервіси безпеки й BaaS відповідають скоріше потенційним, чим реальним потребам. Можна сказати, що в цьому напрямку компанії діють із випередженням. Офісні додатки й віртуальні робочі місця провайдери теж не залишили без уваги у своїх планах. Однак попит на віртуальні АТМ, CRM і бухгалтерські системи може виявитися переоціненим,

КПЗ-2024

хоча в хмарної телефонії є серйозний потенціал росту, оскільки у зв'язку із кризою організації всі частіше міняють офіси, щоб оптимізувати витрати на їхній зміст.

Резонно розраховуючи на затребуваність систем керування проектами, провайдери недооцінюють потенціал росту традиційних сервісів – електронної пошти й віртуальних серверів (або просто не відносять їх до категорії хмарних), а також офісних додатків і хмарного резервного копіювання.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

КбПЗ_2024

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Assembla.com – безкоштовний віртуальний офіс для команди розроблювачів

Не зовсім стартап, компанія Assembla працює на ринку з 2006 року, надаючи як безкоштовний пакет, так і розширюючи функціональність за невелику плату. Початкова реєстрація й робота цілком безкоштовні, і повірте, для організації невеликого веб-офісу для розподіленої команди розроблювачів, які працюють над одним або декількома проектами, вам цілком вистачить тільки безкоштовного аккаунта. В основному, він обмежений обсягом місця під зберігання файлів (допускається як просто зберігання файлів у проекті, так і системи контролю версій – SVN, Git і Mercurial) – усього 200 Мб на проект, а за додаткову плату – більше. Всі інші функції й модулі – безкоштовні, платні лише преміум сервіси, начебто бекапа на Amazon S3, додатковий рівень безпеки (використовуючи SSL), а також 5 Гб місця. Єдине, що із платного цікаво й необхідно – місце, щоб у більших проектах можна було зберігати всю історію версій, адже в безкоштовному самі старі ревізії коду віддаляються, тільки-но ви досягніть межі дискового простору. Платна версія коштує досить дешево – 150$ у рік, а необхідно це тільки для власника проекту, іншим учасникам досить простих аккаунтів. До речі, для відкритих і студентських проектів вони нададуть всі умови безкоштовно, варто повідомити їм і описати проект.

Проекти, над якими ви працюєте, називаються Spaces, і являють собою набір сервісів у єдиному інтерфейсі й об'єднаної функціональність (наприклад, загальна реєстрація й т.п., чого саме дуже складно домогтися, розгортаючи все це

Копія 2024

самостійно). Далі ми розглянемо надавані сервіси, з яких, як із цеглинок, ви можете будувати свій віртуальний офіс залежно від потреб проекту.

До речі, самі spaces можуть бути як приватній і доступними тільки вашій команді, так і відкритими для всіх. Так, для приклада, можна гнучко управляти роботою над яким-небудь відкритим проектом.

Основні сервіси

Wiki дозволяє спільно вести документацію й редагувати документи, однак я б не став рекомендувати її для серйозної роботи – на жаль, це саме той модуль, що краще ставити самому на власному сайті, а вхідний у пакет недотягає до відкритих аналогів типу DocuWiki або MediaWiki, однак для базового й примітивного використання для документації підійде. Плюсом є те, що стартова сторінка проекту може бути оформлена як вики, а звідти вже будуть посилання на більше зручні інструменти. Однак, якщо wiki вам треба тільки для ведення внутрішньої документації, а всі розроблювачі відмінно підковані, то ви можете використовувати убудовану функціональність від Trac, хоча, звичайно, на мій погляд, зручності менше, і доступ до цієї частини буде завжди через додаткову авторизацію.

Для спілкування між членами команди є модуль Messages, що щось начебто веб-форуму або дошки оголошень, дозволяє задавати загальні питання всім учасникам проекту й далі обговорювати кожний з них. Основний плюс – те, що вся переписка йде в єдиному інтерфейсі й доступна всім учасникам, крім потім перекопування особистої переписки або логів ICQ.

Який же проект без учасників, тому модуль Team настільки важливий – тут ви можете як власник проекту запрошувати інших учасників (по e-mail, якщо вони не мають аккаунта в assembla.com, їм буде запропоновано зареєструватися), а також управляти правами доступу інших членів команди.

Модуль File дозволяє зберігати додаткові файли, пов'язані із проектом, наприклад, документацію або інші файли, при цьому є атрибути web 2.0 начебто розмітки тегами. Для графічних файлів їсти прев'ю (однак для графіки найкраще

КПЗ_2024

використовувати модуль Images), інші забезпечуються позначка-інформацією й доступні для завантаження.

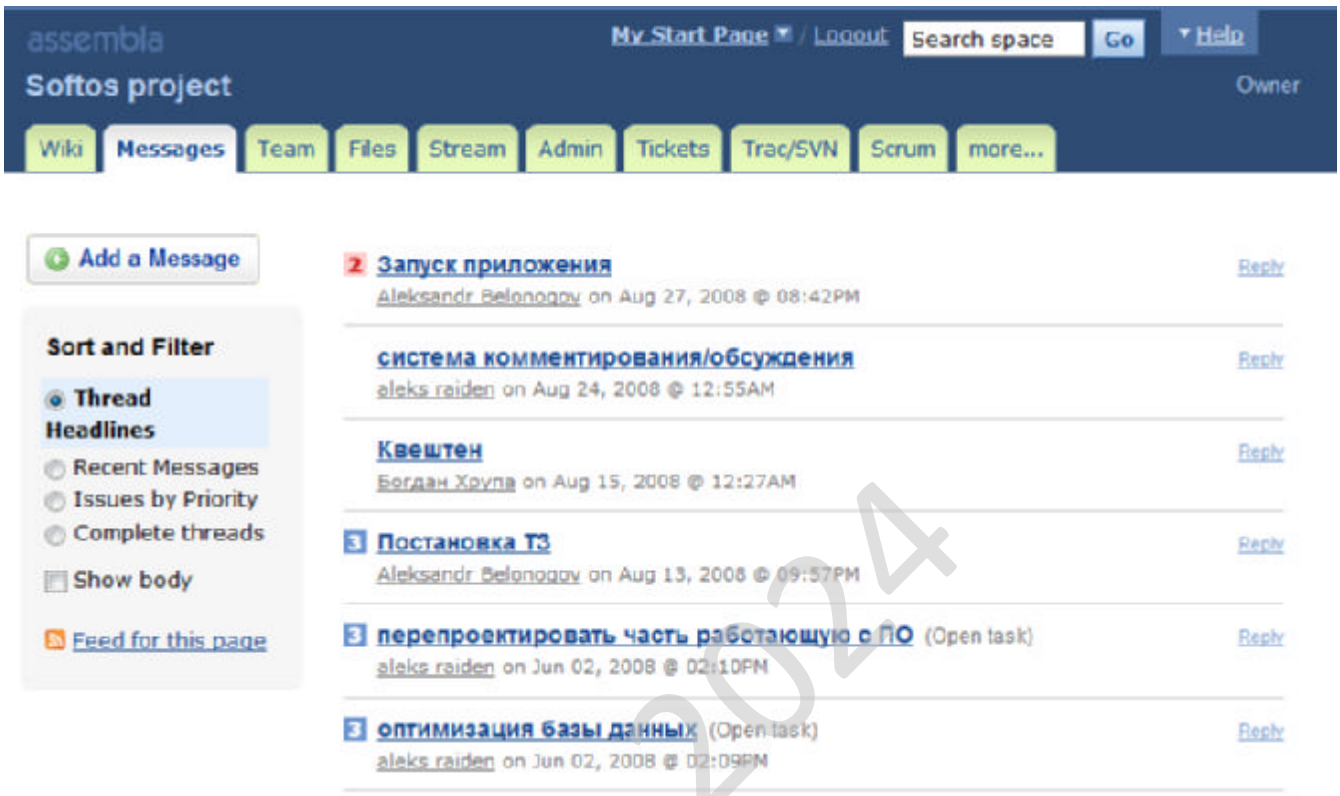

Рисунок 2.1 – Інтерфейс користувача Assembla.com

Основний модуль, з яким прийде працювати, особливо якщо ви менеджер, це Stream – свого роду центральна панель, куди стікаються повідомлення про всі зміни в проекті. Тут відразу видно всю активність – і нові файли, коментарі, хронологія коммітів у систему контролю версій, редагування wiki або зміни в складі команди. При цьому ви можете гнучко набудовувати відображення й указувати тільки ті події, що вам цікаві й у бажаний проміжок часу. Тут також настроюється оповіщення по e-mail – вам не обов'язково тримати відкритим сторінку сервісу увесь час, як тільки щось відбудеться, ви будете сповіщені поштою (звичайно, якщо настроїли собі оповіщення). Або ви можете використовувати модуль Dashboard – він трохи відрізняється по зовнішньому

КБПЗ_2024

вигляді, але зрівняємо по функціональності. І звичайно ж всі зміни доступні по RSS-стрічці.

Сервіси для розробки

От ми й підійшли до самого цікавого. Для розроблювачів Assembla пропонує на вибір кілька систем контролю вихідного коду – традиційний Subversion, а також сучасні Git або Mercurial (докладніше). І все це інтегровано із системою Trac. Ви також можете використовувати й власну систему тікетів – модуль **Tickets**, мені він здався більше зручним і навіть гарним, у порівнянні зі стандартним для Trac, однак це на аматора. Тим більше, перемкнеться можна в будь-який момент. Якщо у вас є ще й власний SVN-репозитарій, ви можете підключити його й працювати як з ним, так і використовувати обидві системи одночасно (убудовану сервісу й власну). Репозитарій можна переглядати як онлайн, через Trac, так і використовувати будь-які інші інструменти (рекомендуються для різних ОС наведені прямо на сторінці).

Для керування й контролю над роботою корисний модуль Scrum, що дозволяє створювати звіти про те, що було зроблено й над чим кожний учасник буде ще працювати. Разом із цим модулем корисний і убудований чат (Chat), що служить для безпосереднього онлайн спілкування між учасниками на сайті.

Додатково, відслідковувати хід робіт, можна через модуль Milestones, що одержує інформацію від модуля тікетів і репозитарія коду, формуючи картину готовності проекту до призначених ключових крапок. Без цього модуля складно представити роботу над більш-менш складним і важливим проектом.

Маленьким відкриттям, навіть після використання сервісу в декількох проектах, став модуль Images. Для проектів, де використовуються графічні елементи, замість файлового модуля найкраще використовувати саме Images – це не тільки зберігання зображень, але ще й оригінальний Flash-редактор, що дозволяє малювати й обговорювати завантажені зображення просто на сторінці. Особливо зручно це на етапі тестування – скріншот можна завантажити й відразу на ньому відзначити проблемні ділянки, тут же прокоментувавши його.

КПЗ_2024

З додаткових модулів, які не обов'язкові, але можливо вам знадобляться, я б відзначив систему Mephisto (керування блогом і сайтом для проекту), Staffing (система публікацій вакансій на порталі assembla.com і керування найманням учасників, така міні-рекрутингова система). Якщо ви використовуєте керування часом для учасників, додайте в проект модуль Time для розподілу й контролю часу на кожне завдання й кожного співробітника.

Ну, а для самих серйозних розроблювачів, або просто для специфічних умов використання сервісу й вимог по інтеграції його у вашу інфраструктуру або проект, є добре документований REST API, а також можливості по інтеграції з іншими системами, наприклад, використовуючи XMPP-сервер Assembla ви можете спілкуватися й бачити всю свою команду просто у вашім jabber-клієнті.

Для користувачів комерційною версією, крім додаткових можливостей, доступні й розширені можливості по керуванню своєю кар'єрою. Загальнодоступний модуль Talents дозволяє підшукати собі в проект кваліфікованого співробітника, або ж указати свої можливості й знання, щоб вас могли знайти роботодавці. А розділ Jobs дозволить публікувати вакансії у своїх проектах і наймати тимчасових (а може й постійних) співробітників. Таким чином, Assembla виступає в ролі аутсорсингової і рекрутингової платформи, що дуже органічно вписується в її сервіси.

У підсумку, для того, щоб зареєструвати новий проект і налаштувати його для комфортної роботи невеликої команди розроблювачів і менеджера, вам досить вибрати йому ім'я й налаштувати необхідні модулі, просто вибираючи їх зі списку, конкретні налаштування будуть створені автоматично. Якщо брати звичайний веб-проект і невелику команду (3 – 5 чоловік), то по моєму досвіді, досить наступного набору модулів: Wiki, Files, Messages, Team, Stream, Trac/SVN, Tickets (мені просто він більше подобатися, чим убудований в Trac), Chat, Images і Milestones. У принципі, частина з них стандартна й доступна відразу, інші необхідно додати з панелі Admin – Tools.

КПЗ-2024

Налаштування такого веб-офісу, разом із запрошенням всіх учасників команди, у мене зайняла всього 15 мінут часу, так сказати, «не відходячи від браузера» і не вникаючи в тонкості конфігурування SVN-A або інших додатків. В умовах невеликих команд і простих проектів таке рішення здатне істотно прискорити початок роботи, не жадаючи від компанії або колективу нічого, крім наявності браузера, інтернету й базових знань своєї роботи.

Tekmi

Останнім часом сильно шумить віртуальна АТМ Tekmi, що надає досить потужний функціонал, привабливі ціни на свої послуги й телефонні номери у великих містах. Спробуємо розібратися в сервісі.

Рисунок 2.2 – Інтерфейс користувача Tekmi

Судячи з публічних даних, Tekmi є внутрішнім проектом великого дистрибьютера ПЗ Softline. Запустившись в 2011 році, Tekmi відразу був визнаний кращим стартапом року.

Потім РВК увійшов до складу засновників і інвестував у проект ні багато ні мало 1 мільйон доларів. На притягнуті засоби Tekmi запустив продукти Віртуальний офіс і Онлайн-консультант для сайту й почав масштабування своїх продажів. Судячи з динаміки відгуків про Tekmi, проект більш ніж живий.

Віртуальна АТМ Tekmi або Віртуальний офіс Tekmi

Tekmi створив гарну віртуальну АТМ, але збагатив її додатковим функціоналом об'єднаних комунікацій (unfied communications), нині настільки популярних на заході: поштою, чатом і файловим сховищем.

Благо, Tekmi надає безкоштовний тріальний період. Це великий плюс, що можна із сайту взяти й спробувати продукт, навіть не звертаючись до менеджера.

Заповнюємо реєстраційні дані.

Біжить модний «прогрес бар», ілюструючи процес створення послуги. Видно, що хлопці постаралися при проектуванні проекту й зробили все зручно. З 2011 року ці екрани не мінялися, тому що й так всі добре.

Для тестування вхідних дзвінків мені надали 7 телефонних номерів. З одним нюансом: при дзвінку на них потрібно набрати додатковий номер, закріплений за моєї АТМ. І тоді дзвінок пройде по настроєному мною маршруту.

Реалізували дзвінок прямо із браузера. Мені не потрібно й телефон реєструвати. Просто вбив номер і подзвонив. Я вбив свій мобільний для тесту – всі добре. Але не обійшлося без жменьки дьогтю: в Google Chrome звонилка відмовилася працювати. В FireFox усе вийшло.

Ну а далі мені на робочому столі доступний весь інший функціонал: гнучке налаштування переадресацій, голосова пошта, автосекретар, запис дзвінків.

У розділі налаштувань я можу більш гнучко налаштувати розподіл дзвінків: створити групи, короткі номери, голосове меню.

Віртуальний офіс Tekmi

Але віртуальна АТМ – лише одна з функцій Tekmi. По-перше, це чат. Усе колеги видні в ньому й можна листуватися прямо в інтерфейсі.

КПЗ-2024

По-друге, з Tekmi не потрібний поштовий сервіс. Прив'язка домену безкоштовна. Можна підключитися по IMAP.

По-третє, контактна книга, хоча їй складно здивувати навіть віртуальні АТМ.

По-четверте, файлове сховище. Команда Tekmi розраховує, що клієнти будуть зберігати файли в хмарі Tekmi, а не в Dropbox або інших сервісах. Що ж, досить сміло.

Онлайн-консультант Tekmi

Tekmi також запустив окремий проект « Онлайн-консультант» по підтримці клієнтів прямо на сайті, про яке ми вже писали. У цілому робота консультанта дуже зручна, але коштує пристойних грошей. Онлайн-консультант також дозволяє дзвонити із сайту й замовляти зворотний дзвінок.

У цілому продукт від Tekmi сподобався й заслуговує гарного відкликання. Будемо сподіватися, що продукт працює так само стабільно, як і виглядає. Рекомендуємо взяти демо й спробувати.

"Київстар"

З 1 квітня 2014 року в «Київстар» формат роботи «Віртуальний офіс» з пілотного режиму перейшов у стандартну HR-практику компанії.

«Віртуальний офіс» дозволяє співробітникам працювати один день у тиждень віддалено – поза офісом. Тепер такий режим роботи доступний представникам практично всіх підрозділів компанії.

Впровадження нового формату роботи «Віртуальний офіс» в «Київстар» почалося в 2012 році. В основу лягла ідея співробітника, опублікована на онлайн-платформі підтримки інновацій «Створи Свій Київстар». В 2012 році відбулися дві фази проекту – для IT-підрозділу й технічної дирекції центрального офісу «Київстар». Тоді можливість віддаленої роботи одержали більше 200 співробітників. Ґрунтуючись на позитивних відкликаннях керівників і учасників проекту, менеджмент компанії ухвалив рішення щодо його продовженні.

КПЗ_2024

На третьому етапі більше 250 співробітників центрального офісу «Київстар» одержали можливість прийняти участь в ініціативі й зробити свій робочий графік більше зручним. У результаті – 100% співробітників і 88% керівників, які взяли участь у проекті «Віртуальний офіс», оцінили його позитивний вплив на ефективність роботи. Наприкінці березня 2014 року Виконавчий комітет «Київстар» ухвалив рішення щодо завершенні пілотной фази й введенні формату роботи «Віртуальний офіс» у стандартну HR-практику компанії.

По статистиці учасниками проекту стали: співробітники технічного й IT-відділів – 60%, співробітники комерційних відділів – 25%, представники інших підрозділів – 15%.

Примітно, що самі затребувані дні серед співробітників для роботи віддалено – п'ятниця (40%), понеділок (23%) і середовище (22%). Вівторок і четвер – найменш популярні дні (5% і 11% відповідно).

«В умовах високої динаміки й інтенсивного навантаження особливо важливо піклуватися про рівень комфорту умов праці. Кілька років назад в «Київстар» впровадили можливість для співробітників самостійно вибирати час початку трудового дня. Висока затребуваність цього інструмента показала, наскільки важливо кожному з нас мати можливість гнучко управляти робочим часом. «Віртуальний офіс» – це наступний крок у цьому напрямку. И це незважаючи на те, що режим «Віртуальний офіс» у масовому використанні – непроста для управління практика. Високий рівень довіри «Київстар» до співробітників дозволяє нам бути впевненими в успіху й ефективності цього інструмента. Для нас «Віртуальний офіс» – це ще одна можливість підтримати співробітників, які забезпечують високі результати для компанії.

Результати пілотного проекту «Віртуальний офіс» показують, що ефективність роботи співробітників, що сполучають офісний і віддалений режими, підвищується. Робота «будинку» дозволяє більш ретельно сконцентруватися на завданні, скоротити строки й підвищити якість її виконання.

КПЗ – 2024

Такий режим роботи дозволяє більш гнучко будувати індивідуальний графік, витримуючи правильний баланс між роботою й особистим життям. При цьому забезпечується безперебійне протікання бізнес-процесів.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Для реалізації програми мною була використана мова програмування Visual C++. У зв'язку з тим, що сьогодні рівень складності програмного забезпечення дуже високий, розробка додатків Windows з використанням тільки якої-небудь мови програмування значно утрудняється. Програміст повинен затратити масу часу на рішення стандартних завдань по створенню багатовіконного інтерфейсу. Реалізація технології зв'язування й вбудовування об'єктів – OLE – зажадає від програміста ще більш складної роботи. Щоб полегшити роботу програміста практично всі сучасні компілятори з мови C++ містять спеціальні бібліотеки класів. Такі бібліотеки містять у собі практично весь програмний інтерфейс Windows і дозволяють користуватися при програмуванні засобами більш високого рівня, чим звичайні виклики функцій. За рахунок цього значно спрощується розробка додатків, що мають складний інтерфейс користувача, полегшується підтримка технології OLE і взаємодія з базами даних. Сучасні інтегровані засоби розробки додатків Windows дозволяють автоматизувати процес створення додатка. Для цього використовуються генератори додатків. Програміст відповідає на питання генератора додатків і визначає властивості додатка – чи підтримує воно багатовіконний режим, технологію OLE, тривимірні органи керування, довідкову систему. Генератор додатків, створить додаток, що відповідає вимогам, і надасть вихідні тексти. Користуючись їм як шаблоном, програміст зможе швидко розробляти свої додатки. Подібні засоби автоматизованого створення додатків включені в компілятор Microsoft Visual C++ і називаються MFC AppWizard. Заповнивши

КОПЗ_2024

кілька діалогових панелей, можна вказати характеристики додатка й одержати його тексти, постачені великими коментарями. MFC AppWizard дозволяє створювати одновіконні й багатовіконні додатки, а також додатки, що не мають головного вікна, – замість нього використовується діалогова панель. Можна також включити підтримку технології OLE, баз даних, довідкової системи. Звичайно, MFC AppWizard не всесильний. Прикладну частину додатка програмістові прийдеться розробляти самостійно. Вихідний текст додатка, створений MFC AppWizard, стане тільки основою, до якої потрібно підключити інше. Але працюючий шаблон додатка – це вже половина всієї роботи. Вихідні тексти додатків, автоматично отриманих від MFC AppWizard, можуть становити сотні рядків тексту. Набір його вручну був би дуже стомлюючий. Потрібно відзначити, що MFC AppWizard створює тексти додатків тільки з використанням бібліотеки класів MFC (Microsoft Foundation Class library). Тому тільки вивчивши мову C++ і бібліотеку MFC, можна користуватися засобами автоматизованої розробки й створювати свої додатки в найкоротший термін. Як уже згадувався, MFC – це базовий набір (бібліотека) класів, написаних мовою C++ і призначених для спрощення й прискорення процесу програмування для Windows. Бібліотека містить багаторівневу ієрархію класів, що нараховує близько 200 членів. Вони дають можливість створювати Windows-додатки на базі об'єктно-орієнтованого підходу. З погляду програміста, MFC являє собою каркас, на основі якого можна писати програми для Windows. Бібліотека MFC розроблялася для спрощення завдань, що стоять перед програмістом. Як відомо, традиційний метод програмування під Windows вимагає написання досить довгих і складних програм, що мають ряд специфічних особливостей. Зокрема, для створення тільки каркаса програми таким методом знадобиться близько 75 рядків коду. У міру ж збільшення складності програми її код може досягати воістину неймовірних розмірів. Однак та ж сама програма, написана з використанням MFC, буде приблизно в три рази менше, оскільки більшість приватних деталей приховано від програміста.

КОПЗ-2024

Одною з основних переваг роботи з MFC є можливість багаторазового використання того самого коду. В зв'язку з тим, що бібліотека містить багато елементів, загальних для всіх Windows-додатків, немає необхідності щораз писати їх заново. Замість цього їх можна просто успадковувати (говорячи мовою об'єктно-орієнтованного програмування). Крім того, інтерфейс, забезпечуваний бібліотекою, практично незалежний від конкретних деталей, його що реалізують. Тому програми, написані на основі MFC, можуть бути легко адаптовані до нових версій Windows (на відміну від більшості програм, написаних звичайними методами). Ще однією істотною перевагою MFC є спрощення взаємодії із прикладним програмним інтерфейсом (API) Windows. Будь-який додаток взаємодіє з Windows через API, що містить кілька сотень функцій. Значний розмір API утрудняє спроби зрозуміти й вивчити його цілком. Найчастіше навіть складно простежити, як окремі частини API зв'язані один з одним! Але оскільки бібліотека MFC поєднує (шляхом інкапсуляції) функції API у логічно організовану безліч класів, інтерфейсом стає значно легше управляти.

Оскільки MFC являє собою набір класів, написаних мовою C++, тому програми, написані з використанням MFC, повинна бути в той же час програмами на C++. Для цього необхідно володіти відповідними знаннями. Для початку необхідно вміти створювати власні класи, розуміти принципи спадкування й уміти перевизначати віртуальні функції. Хоча програми, що використовують бібліотеку MFC, звичайно не містять занадто специфічних елементів з арсеналу C++, для їхнього написання проте потрібні солідні знання в даній області.

КОПЗ_2024

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

КПЗ – 2024

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Як показують результати дослідження, самими популярними хмарними сервісами залишаються електронна пошта й віртуальні сервери. Це очікуваний результат, адже дані сервіси одержали досить широке поширення ще в «дохмарну» епоху, та й сьогодні по формальних ознаках (включаючи гнучкість виділення й звільнення ресурсів, оплату за фактично споживані ресурси) далеко не завжди є дійсно хмарними, навіть якщо позиціонуються провайдерами такими. Проте користувачі цих послуг нерідко неправомірно відносять їх до хмарних сервісів, що дає завищену оцінку по даних позиціях.

Досить широке поширення одержали офісні додатки – Microsoft Office 365, Google Apps і ін. Їх застосовують уже більше 20% респондентів. А в малих компаніях такий сервіс займає друге по популярності місце, хоча організації із цього сегмента не завжди користуються корпоративними версіями Office і відповідними послугами провайдеру. Звідси розбіжність із даними про пропозиції провайдерів, у яких ця послуга займає поки далеко не перше місце в переліку їхніх сервісів.

Сучасні офісні додатки – це цілий набір хмарних сервісів, які значно спрощують більшість робочих процесів. Таке хмарне рішення забезпечує повсюдний доступ до пошти й документів, контактам і календарям, дозволяє створювати сайти для зовнішнього й внутрішнього користування, а також спілкуватися з колегами й партнерами через аудіо- і відеозв'язок. Завдяки розмаїтості сервісів вдається вишиковувати ефективні комунікації між співробітниками й/або партнерами.

Таке рішення гарне вписується в концепцію BYOD і найчастіше виявляється оптимальним для організації віддаленої роботи. Знайомий і простий

КОПЗ_2024

інтерфейс офісних додатків зрозумілий і звичний користувачам. Доступ до пошти й документів можна одержати звідки завгодно й коли завгодно. Нерідко до того ж провайдери включають у пакет послуг доступ в Інтернет для мобільної роботи з різних пристроїв. Замовник здобуває необхідні IT-рішення, при цьому йому не треба купувати дорогі ліцензії й він платить тільки за реальне використання послуг. У поточних економічних умовах така модель, виразно, буде набирати популярність.

Четверту позицію серед затребуваних замовниками сервісів розділяють резервне копіювання в хмару (Backup as a Service, BaaS) і системи керування проектами. Популярність BaaS можна пояснити широким поширенням хмарних сховищ, уже добре знайомих користувачам персональних пристроїв і ПК, а також оснащенням функціями копіювання в хмару багатьох мережних СХД. У системах керування проектами, у свою чергу, найбільше яскраво й повно втілена ідея віртуального офісу. Звичайно вони містять у собі модулі CRM, однак системи керування взаємодією з партнерами й замовниками ми виділили в окрему категорію.

Віртуальні робочі місця поки використовують 14,4% респондентів, на хмарні сервіси безпеки, такі як очищення/фільтрація трафіку, антивірус/антиспам і ін., покладаються у своїй роботі 12,7%. І це – незважаючи на досить широкий спектр сервісів безпеки, пропонованих багатьма провайдерами, і присутність VDI у трійці найпоширеніших пропозицій. Попит у цих сегментах відстає від пропозиції.

Зненацька мало респондентів задіють у своїх бізнес-процесах віртуальні АТМ (хмарну телефонію). Тим часом віртуальна АТМ цілком здатна замінити офісну АТМ із самим широким набором можливостей, а інтеграція функцій телефонії з бізнес-додатками перетворює її в коштовне комунікаційне рішення, завдяки якому діяльність компаній різного профілю і їхня взаємодія із клієнтами стають набагато ефективніше.

КПЗ – 2024

Провайдери хмарної телефонії нерідко пропонують ВАТМ у комплекті з CRM, інтегрують ці продукти, доповнюють віртуальні АТМ функціями центра обробки викликів. Провідні постачальники таких рішень заявляють про високі темпи продажів (50% щорічно) і швидкому росту клієнтської бази. Однак, як показав наше опитування, значного подальшого росту в даному сегменті може не трапитися: хмарну телефонію збираються застосовувати не так вуж багато нових замовників. По інтересі серед замовників вона уступає всім перерахованим хмарним сервісам.

У своїх планах респонденти демонструють сталість: як і раніше пріоритет віддається електронній пошті й віртуальним серверам, однак значного росту попиту на ці види сервісів очікувати не слід. Високий потенціал росту – у сервісів резервного копіювання в хмару, систем керування проектами, офісних додатків, віртуальних робочих місць і сервісів безпеки. Тим часом для успішного розвитку хмарного ринку необхідно перебороти побоювання потенційних замовників, пов'язані з безпекою й надійністю сервісів, а також оптимізувати їхня вартість.

Що ж заважає поширенню хмарних сервісів в Україні? Насамперед, як і в усьому світі, існують побоювання, пов'язані з інформаційною безпекою. У опитуванні даний фактор відзначили більше 66% респондентів. Регулярно публикуємі історії злому сервісів і витоку конфіденційних даних у великих світових провайдерів лише підсилюють ці побоювання. Сумніви можуть викликатися й непевністю в стабільності бізнесу провайдеру, навіть якщо він досить давно є присутнім на ринку

На відміну від закордонних опитувань, у нашій країні друге місце займає висока вартість сервісів, що нерідко зводить нанівець одне з основних їхніх переваг, про які люблять говорити провайдери, – скорочення витрат. Цей фактор виділили понад 44% замовників. Недостатня надійність, зазначена в 30% відповідей, замикає трійку головних негативних факторів. Серед інших причин – обмежена функціональність сервісів, відсутність підходящих пропозицій і зручного інтерфейсу для керування послугами.

КПЗ-2024

У числі перешкод на шляху переходу до сервісів публічних хмар респонденти називають і такі: ризик невідповідності вимогам регуляторів в області забезпечення конфіденційності інформації, законодавству відносно держустанов і корпоративних політиків інформаційної безпеки, схоронності банківської таємниці й персональних даних; недостатнє фінансування й недостача часу для розробки стратегії використання хмарних технологій і розгортання хмарних сервісів; неможливість адаптації сервісів під власні потреби й відсутність стандартизованого програмного інтерфейсу керування; неможливість одержання повного контролю над всіма даними, включаючи резервні копії; недостатня пропускна здатність і надійність каналів зв'язку (ця проблема особливо актуальна для регіонів).

3.2 Розробка структурної схеми

Отже, одним з головних перешкод на шляху використання хмарних сервісів компанії-замовники вважають проблеми в сфері ІБ. Тим часом хмарні провайдери не тільки приділяють підвищену увагу інформаційної безпеки своєї інфраструктури, але й пропонують сервіси ІБ для захисту ІТ-інфраструктури клієнтів. Причому частка таких сервісів виявилася зненацька високою. Це говорить про те, що провайдери, як правило, мають достатні можливості й кваліфікацією, щоб не тільки забезпечити безпека власних продуктів, але й запропонувати різні рішення для захисту даних клієнтів, організації безпечного доступу й зберігання інформації.

Досить широкий і спектр пропонованих сервісів безпеки. Найпоширеніші з них (у порядку убування) – міжмережний екран, шифрування каналу доступу (VPN) і захист від DDoS, фільтрація трафіку, блокування шкідливого контента, антивірус, гарантоване знищення даних клієнта після завершення їхнього життєвого циклу, шифрування даних замовника й боротьба зі спамом.

КБПЗ-2024

Побоювання користувачів викликає також надійність пропонованих їм сервісів. Тим часом, як показав опитування, переважна більшість провайдерів передбачають висновок угоди SLA, найчастіше обов'язкове. Як правило, для підвищення рівня надійності сервісів використовуються кілька рознесених площадок (ЦОД) і катастрофостійкі конфігурації IT-систем. Однак катастрофостійкість забезпечують лише не набагато більше 40% респондентів (докладна інформація представлена в повній версії звіту). Втім, далеко не всі хмарні сервіси мають потребу в такому високому рівні надійності. У поточній економічній ситуації критичним стає ще один фактор – вартість послуг.

Як відзначалася, майже половина замовників уважають високу вартість хмарних сервісів одним з основних перешкод у розвитку й поширенні хмарної моделі. На жаль, у нинішній кризовій ситуації лише деякі провайдери планують знизити вартість послуг. Інші збираються підвищити тарифи або вже проіндексували їх з урахуванням валютного курсу.

Український ринок хмарних сервісів усе ще демонструє ріст і далекий від насичення. Важливими факторами його розвитку залишаються поява нових гравців, особливо великих телекомунікаційних компаній, оптимізація вартості хмарних сервісів, поліпшення їхньої якості, надійності й функціональності. Очікується, що в умовах скорочення IT-бюджетів бажання заощадити й перевести капітальні витрати в операційні підсилить інтерес компаній до хмарних рішень.

Крім відзначених вище перешкод, серйозним бар'єром є небажання переходити на хмарну модель через необхідність здійснення інфраструктурних і організаційних змін. До того ж витрати на міграцію в хмару, особливо для великих компаній, можуть виявитися досить істотними. Та й не завжди робота в хмарі обходиться дешевше. При цьому провайдер повинен надати угоду про рівень обслуговування. Споживачам хмарних сервісів варто уважно вивчити умови SLA, щоб знати, хто й за що відповідає в різних ситуаціях. Угоди повинні бути детально викладеними й зрозумілими.

КПЗ-2024

У контексті віртуального офісу особливу цінність здобувають пакети послуг: по-перше, такі комплексні сервіси найчастіше є інтегрованими, що дає синергетичний ефект (наприклад, офісні додатки доповнюються функціями хмарної телефонії, а віртуальна АТМ – функціями CRM), а в других, замовник може одержати їх з дисконтом. Провайдерам варто формувати свої пакетні пропозиції, виходячи із трьох факторів: уже представлених на ринку пакетів послуг; синергетичного ефекту від інтеграції додатків і сервісів; існуючих потреб і планів наявних і потенційних замовників. Вагомим плюсом будуть і додаткові послуги (VAD), такі як міграція в хмарне середовище, побудову приватної хмари на площадці замовника або провайдеру, а також різні послуги, пов'язані з розгортанням і експлуатацією гібридних хмар.

Переважна більшість опитаних замовників планують розширення спектра використовуваних хмарних сервісів. Тому можливість інтеграції нових сервісів із уже затребуваними, зручна схема оплати при розширенні послуг і комплексні пропозиції стануть додатковими факторами, що підвищують привабливість розроблених провайдерами рішень.

До факторів росту ринку хмарних сервісів можна віднести також збільшення числа компаній, що прагнуть надати широкі можливості своїм мобільним/віддаленим співробітникам, а також зацікавлених у тім, щоб їхній бізнес став більше гнучким і легше адаптувався до нових умов. Усе більше підприємств і організацій стануть використовувати хмарні сервіси як заміну локальним рішенням. Провайдерам прийде адаптувати свої пропозиції відповідно до мінливих вимог клієнтів, а також освоювати інші моделі взаємодії із замовниками в рамках реалізації інфраструктурних і інтеграційних проектів.

При правильному підході до формування лінійки сервісів, їхньому доповненні коштовними для замовника послугами, забезпеченні необхідного рівня безпеки й надійності при прийнятній ціновій політиці віртуальний офіс дозволить малим і середнім компаніям динамічно розвиватися, підвищувати продуктивність, адаптувати сервіси відповідно до потреб і можливостями своєї компанії, вибрати оптимальний тарифний план і набір додаткових опцій.

КОПІЯ_2024

На рисунку 3.1 зображена структурна схема розробленої системи. Вона розроблена на основі проведеного дослідження існуючих еталонних моделей, за результатами яких за основу для синтезу еталонної моделі середовища відкритої системи обміну даними була прийнята модель, описана в міжнародному стандарті ISO/IEC TR 14252:1996 (E), ANSI/IEEE Std 1003/ 0-1995 Information technology – Guide to the POSIX Open System Environment (OSE). На основі даної моделі була розроблена еталонна модель середовища системи обміну даними, що складається із трьох технологічних рівнів:

– зовнішнього середовища;

– програмного забезпечення середнього рівня (посередника);

– користувальницьких додатків.

На рисунку 3.1 зображена структурна схема системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Розглянемо більш докладно елементи структурної схеми системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Система системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу, побудована за допомогою предметно-орієнтованого інструмента, має багаторівневу архітектуру. Архітектура виступає гарантом доступності, надійності й безпеці системи, що дозволяє системі системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу охопити всіх комп'ютеризованих співробітників і підвищити ефективність роботи організації в цілому.

Основними структурними елементами архітектури є:

– Предметно-орієнтований інструмент розробки – середовище виконання коду, що реалізує інтерфейс служб і користувальницьких додатків (у тому числі сторонньої розробки) для доступу до системи. Зокрема, сервер веб-доступу системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу, реалізований на платформі ASP.NET, використовує

КОПЗ_2024

предметно-орієнтований інструмент розробки для реалізації всіх функцій системи, які стають доступні користувачам через веб-браузер.

– Служби перетворення документів. Перетворення документів в інші формати, добування з документів корисної інформації

– Служби введення документів. Масове введення документів у системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу з різних джерел (сканери, БФП, файлова система, факси, електронна пошта й т.д.).

– Набір засобів інтеграції. Легка інтеграція з ERP-системами: двостороння синхронізація довідників, включення об'єктів системи в workflow, робота з документами з ERP-системи. Workflow – це повна або часткова автоматизація бізнес-процесу, при якій документи, інформація або завдання передаються від одного учасника (бізнес-процесу) до іншого для виконання дій відповідно до набору керівних правил.

– Сервер веб-доступу до системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу. Робота з електронними документами, завданнями й завданнями через веб-браузер.

– Розширення для SharePoint. Набір готових веб-частин і інтеграційних механізмів, що забезпечують доступ до даних системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу з порталу на базі Microsoft SharePoint.

– Клієнти системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу – додатки для кінцевих користувачів, інструментарій розробки, утиліти адміністрування системи. Клієнтом може бути як Windows-додаток, що використовує для доступу до системи предметно-орієнтований інструмент розробки, так і веб-браузер.

– Керуючі служби системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу – служби, що забезпечують керування системою. Наприклад, служба workflow управляє роботою завдань системи

КОПІЯ 2024

архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, а сервіси зберігання системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу відповідають за файлові сховища документів. Всі керуючі служби можуть бути встановлені як на один комп'ютер, так і на різні – з метою розподілу навантаження.

– Сервер реплікації. Створення ієрархічної системи розподілених систем, що обмінюються даними в режимі off-line; налаштовується состав даних, що, реплікуються.

– Служба файлових сховищ. Керування довгостроковим і архівним зберіганням великого обсягу документів, у тому числі медіаданих; настроювання політик переміщення даних по сховищах.

– СУБД – сховище даних і метаданих системи. Одним з важливих компонентів системи, що зберігаються в СУБД, є прикладна розробка системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, що визначає функціональність предметних модулів системи, замовлених, а також розроблених партнерами системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу рішень.

– Файлові сховища – архіви більших або рідко використовуваних документів, які ефективніше тримати за межами СУБД; управляються власними службами.

Архітектура системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, будучи частиною інформаційної інфраструктури організації, демонструє характеристики, важливі для будь-якої корпоративної системи:

– Відкритість. Основа системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу – платформа предметно-орієнтованого інструмента розробки – підтримує технології Microsoft COM і .NET. Вона містить готові інструменти інтеграції з корпоративними додатками, у тому числі набір функцій для обробки XML-документів. Корпоративні стандарти й відкрита

КбПЗ-2024

структура даних дозволяють легко інтегрувати системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу в інформаційну інфраструктуру організації.

– Розширюваність. Як правило, у кожній організації висувають унікальні вимоги до побудови електронного документообігу й рішенню завдань взаємодії. Об'єктна модель і предметно-орієнтований інструмент розробки дозволяють створювати власні й змінювати існуючі об'єкти для рішення специфічних завдань. Оскільки ядром системи є COM-сервер, що управляють функції системи можна використовувати в будь-яких сторонніх додатках.

– Масштабованість. Виділення декількох рівнів архітектури дозволяє підвищувати продуктивність системи не тільки за допомогою нарощування потужності апаратних засобів, але й завдяки розподілу служб по різних серверах. Механізм реплікації предметно-орієнтованого інструмента розробки дозволяє побудувати територіально розподілену систему, мінімізуючи як вимоги до пропускної здатності каналів зв'язку за рахунок обсягу переданих даних між серверами, так і технічні вимоги до вторинних серверів. Виділення як SQL-серверних, так і файлових сховищ документів дозволяє гнучко управляти розподілом навантаження на сервера організації при доступі до документів.

– Надійність. Архітектура системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу підтримує транзакційну модель, що гарантує цілісність дані системи протягом всіх стадій їхнього життєвого циклу. Керовані SQL– і файлові сховища документів дозволяють організувати надійне зберігання документів.

– Безпека. Для кожного об'єкта системи може бути задано, які користувачі або групи мають право виконувати з ним певні дії. Конфіденційні електронні документи й завдання можуть бути зашифровані безпосередньо в системі будь-яким CryptoAPI-сумісним криптопровайдером (у тому числі сертифікованим ДСТЗІ СБУ), що гарантує захист навіть від осіб, що мають необмежений доступ до даних. Протоколювання всіх дій користувача дозволить відновити історію

КОПЗ-2024

роботи з об'єктами системи у випадку порушення режиму безпеки. Забезпечується високий захист від несанкціонованого доступу до сховищ документів всіх типів.

Таким чином, архітектура системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу розроблена з урахуванням максимального використання всіх переваг сучасних технологій, платформ і предметно-орієнтованого підходу до побудови інформаційних систем керування.

На базі системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу розробляється широкий набір бізнес-рішень, націлених на побудову ефективної системи керування бізнесом.

Бізнес-рішення – це програмно-консалтинговий продукт, орієнтований на досягнення заданого ефекту при рішенні певного бізнесу-завдання. Відмінні риси бізнес-рішення:

– Фокус на бізнес-завданнях. Метою бізнес-рішення є задоволення потреб конкретних бізнес-замовників і рішення конкретного бізнесу-завдання.

– Вимірюваний бізнес-ефект. Для кожного бізнес-рішення визначається перелік показників ефективності, які дозволяють виміряти ефект від його впровадження в організації.

– Методики й бізнес-консалтинг. Бізнес-рішення крім технічного рішення включає також методичні матеріали, технологію впровадження, послуги бізнесу-консалтингу.

Бізнес-рішення дають замовникам нові можливості в реалізації проектів, пов'язаних з електронним документообігом, керуванням бізнес-процесами й взаємодією. Завдяки пропрацьованості бізнес-рішень і орієнтації їх на конкретні бізнес-завдання, впровадження проходить із мінімальними ризиками.

Проекти реалізуються в цілому швидше й більш результативно. При цьому різні бізнес-рішення побудовані на одній основі – системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу. Це дозволяє

КПЗ-2024

забезпечити їхнє послідовне впровадження в організації й будувати з окремих бізнес-рішень цілісну інформаційну систему.

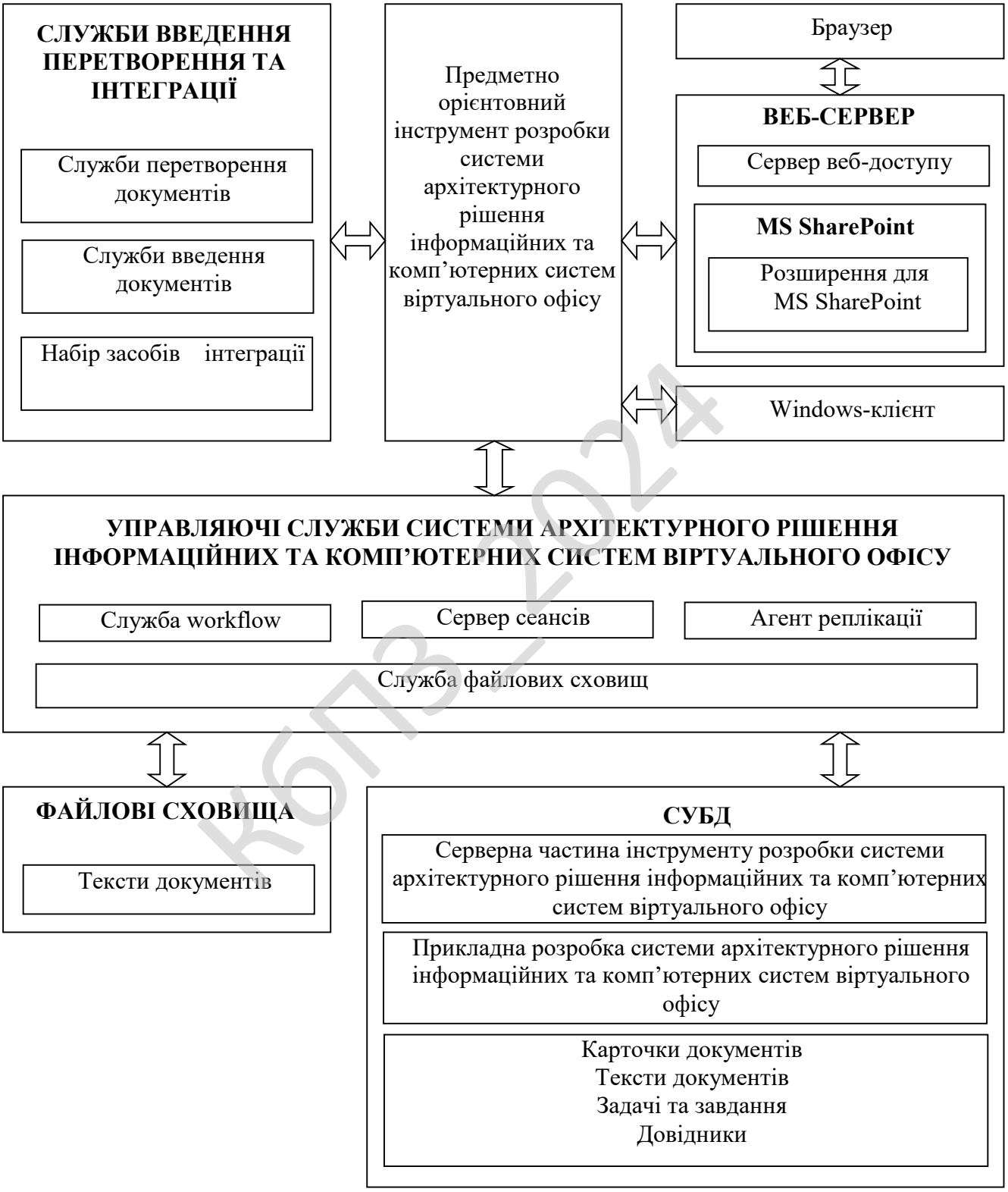

Рисунок 3.1 – Структурна схема системи

КОПІЯ 2024

3.3 Розробка функціональної схеми

На основі структурної схеми моделі середовища системи обміну даними була розроблена функціональна схема моделі середовища відкритої системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, при цьому визначені наступні об'єкти стандартизації:

– служби засобів проектування;

– служби засобів управління змістом;

– служби засобів моделювання процесів;

– служби засобів моделювання даних;

– формати файлових об'єктів;

– служби засобів забезпечення безпеки;

– інтерфейс користувача;

– служби засобів управління даними;

– служби засобів управління системою;

– служби засобів управління сховищем;

– служба комунікацій.

Отримані дані дозволили синтезувати функціональну схему відкритої системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу (рисунок 3.2).

Дана схема використана при розробці функціонального стандарту середовища відкритої системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу. Призначенням даного стандарту є:

– Забезпечення мобільності (переміщуваності) системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

– Забезпечення інтероперабельності системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

– Забезпечення масштабованості системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

КПЗ_2024

– Забезпечення адаптуємості системи.

Визначено, що область застосування функціонального стандарту системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу установлює загальні положення по створенню й експлуатації відкритої системи управління документообігом на основі окремих модулів, що входять у єдину інформаційну систему організації.

Положення функціонального стандарту підлягають застосуванню при рішенні завдань:

– створення, модернізації системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу;

– організації доступу користувачів до ресурсів системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу;

– інтеграції обчислювальних і інформаційних ресурсів із системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Для заповнення профілю в роботі була розроблена методика вибору стандартів на основі експерименту.

Результатом розробки функціонального стандарту є набір вимог до елементів відкритої системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу, розроблений відповідності з ДСТ ІСО/МЕК 10000-1-99 «Інформаційна технологія. Основи й таксономія міжнародних функціональних стандартів. Частина 1. Загальні положення й основи документування» у вигляді функціонального стандарту – Профілю середовища відкритої системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

КОПЗ_2024

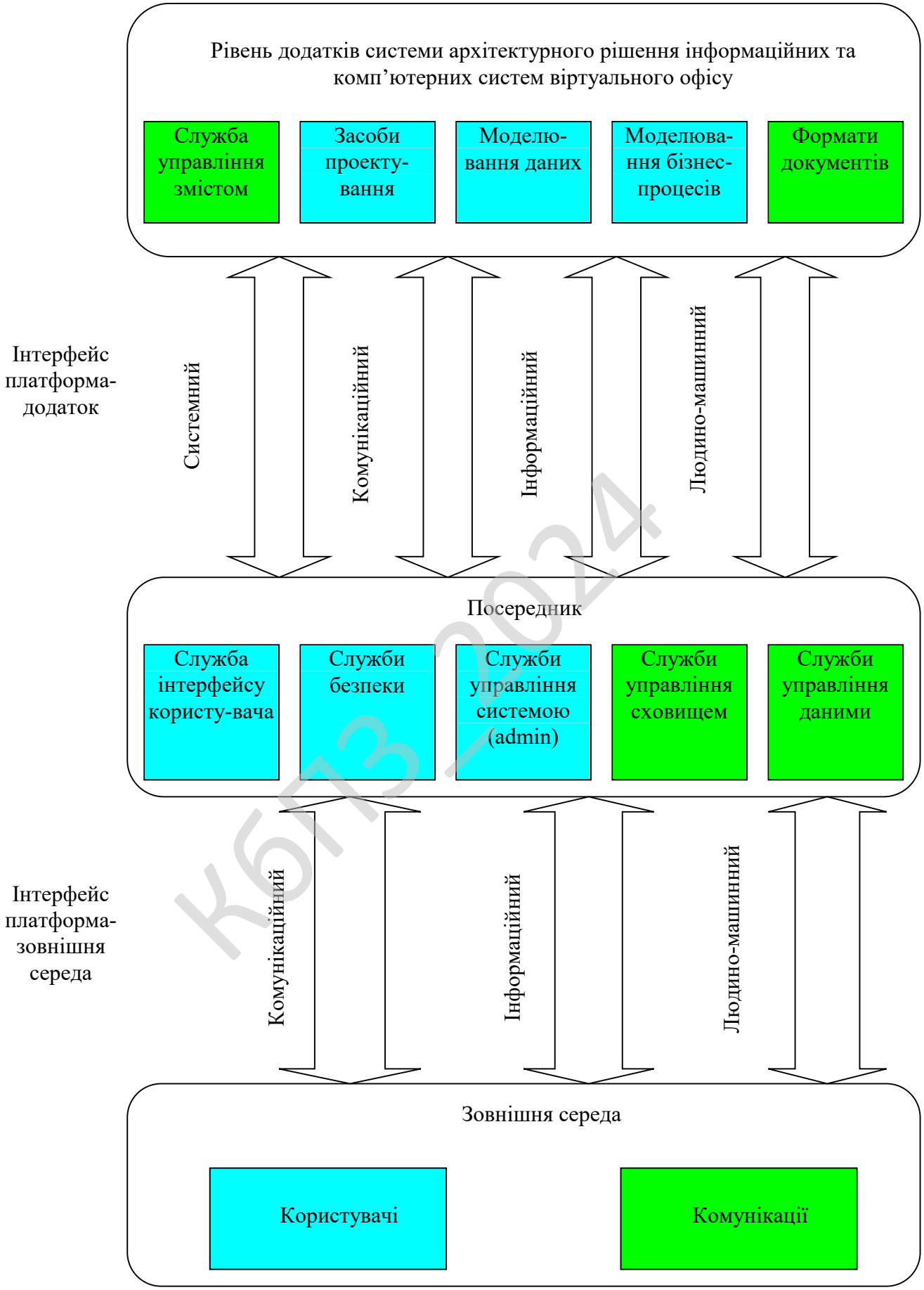

Рисунок 3.2 – Функціональна схема системи

КБПЗ-2024

Проведемо дослідження особливості функціонування елементів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу і експериментальним дослідженням з визначення ефективного формату файлових об'єктів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, як засобу підвищення техніко-економічних характеристик системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Виходячи з вимог до функціонала систем управління й аналізу існуючих системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, у системах управління виділені типові елементи (ТЕ):

– ТЕ «обробки інформації».

– ТЕ «передачі інформації».

– ТЕ «сполучна лінія».

– ТЕ «масив зберігання інформації».

– ТЕ «точка діалогу».

Дані типові елементи служать для опису інформаційної системи й інформаційних потоків у будь-якій системі, що неможливо зробити через описані вище служби еталонної моделі середовища відкритої системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Таким чином показано, що через подібний базис типових елементів може бути представлена кожна, як завгодно складна інформаційна система.

Для проведення функціональної стандартизації ТЕ здійснемо перехід від ТЕ до служб системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Для перевірки експериментальним шляхом залежності ефективності роботи типових елементів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу від форматів використовуваних файлових об'єктів і вибору стандартів для відповідного розділу профілю розроблені:

КОПІЯ 2024

– вимірювальна установка для виміру часу мережної затримки;

– програма статистичного аналізу файлових об'єктів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу;

– методики проведення вимірів часу мережної затримки й розмірів файлів у сховище системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Визначено співвідношення розмірів файлів залежно від типу інформації у файлах і їхніх розмірах. Визначено формати, у яких розмір файлу виходить мінімальним (таблиця 3.1). Визначено, як саме впливає формат файлових об'єктів (ФФО) на роботу служб системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу. На рисунку 3.3 , на моделі системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу показані служби, на роботу яких впливає формат файлових об'єктів. Ці служби виділені жовтим кольором.

Результати досліджень дозволили визначити залежність розмірів файлових об'єктів від типу їхнього вмісту й залежність часу мережної затримки, що підтвердило, що швидкодія системи залежить від розміру файлових об'єктів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

Також дослідження дозволило підтвердити залежність ефективності роботи системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу від ФФО й визначити ефективний формат файлових об'єктів системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу, що дозволяє мінімізувати необхідні для роботи системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу комунікаційні й обчислювальні ресурси, що дозволило гармонізувати стандарти файлових об'єктів у функціональному стандарті (профілі) середовища

КОПЗ_2024

відкритої системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу і підвищити ефективність роботи ТЕ.

Таким чином показано, що залежно від структури документів є ефективні можливості по поліпшенню роботи системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу залежно від сфери її застосування.

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

3.4 Розробка діаграми процесів

Діаграма процесів розробленої системи зображена на рисунку 3.3. При детальному її розгляді можна побачити як саме проходе взаємодія у розробленій системі.

Використовується модель проектування, графічне представлення «потоків» даних в інформаційній системі.

Діаграма взаємодії процесів використовується для візуалізації процесів обробки даних (структурне проектування).

Для розробника вважається звичним спочатку креслити діаграму взаємодії процесів даних рівня контексту, завдяки чому буде показано взаємодію системи.

Ця діаграма в подальшому підлягає уточненню шляхом деталізації процесів та потоків даних з метою показати систему що розробляється.

Діаграми потоків даних містять чотири типи елементів:

– Процеси які являють собою трансформацію даних в рамках описуваної системи.

– Сховища даних (репозиторії).

– Зовнішні по відношенню до системи сутності.

– Потоки даних між елементами трьох попередніх типів.

КПЗ-2024

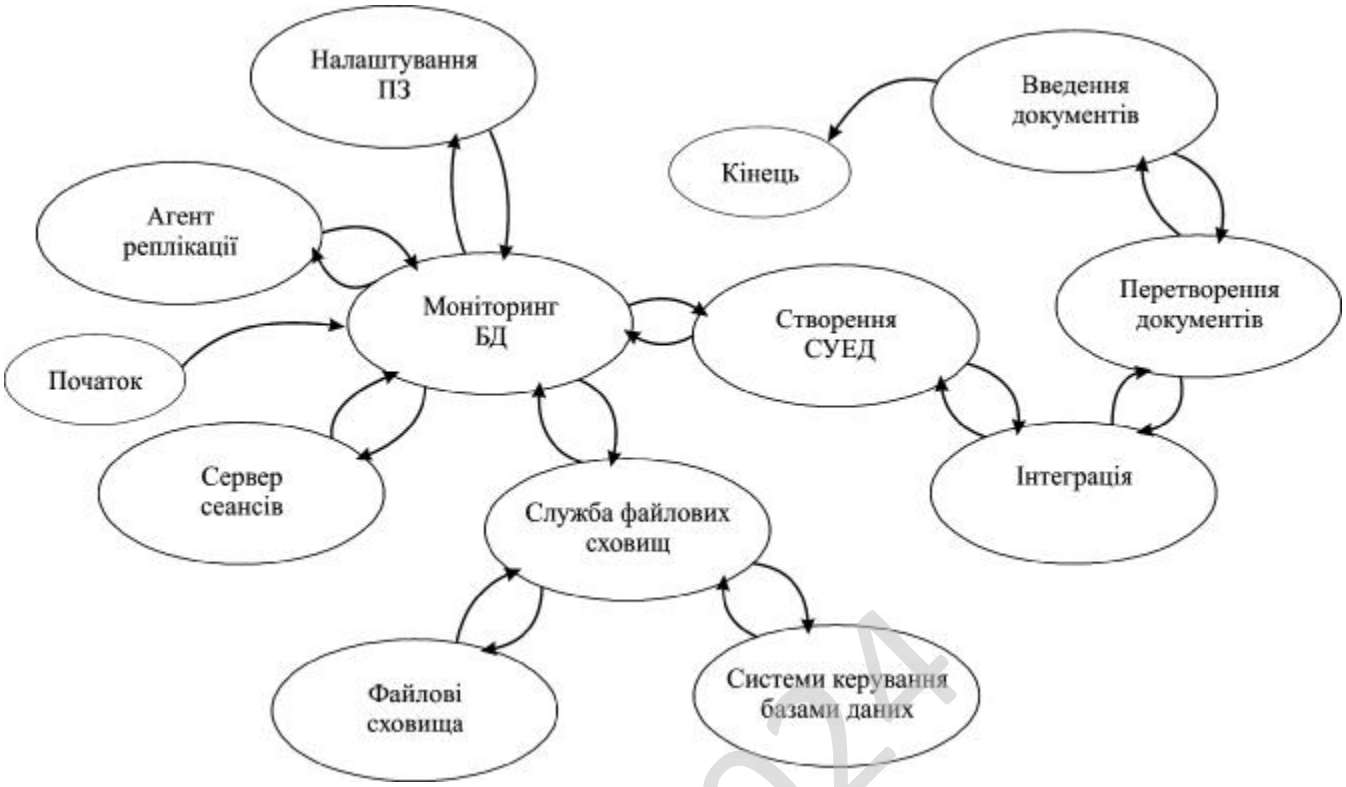

Рисунок 3.3 – Діаграма взаємодії процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

КОПЗ_2024

4 РЕАЛІЗАЦІЯ ПРОЕКТУ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ПРАВИЛЬНІСТЬ ПРОЕКТНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

Під час роботи над магістерською дипломною роботою було створено блок-схеми. Перед їх розглядом необхідно провести роз’яснення який саме тип блок-схем використовується.

Блок-схема це представлення задачі для її аналізу або розв'язування за допомогою спеціальних символів (геометричних образів), які позначають такі елементи, як операції, потік, дані тощо. Блок вхідних та вихідних даних прийнято позначати паралелограмом, блок обчислень (обробки) даних – прямокутником, блок прийняття рішень – ромбом, еліпсом – початок та кінець алгоритму.

У інформаційних технологіях функціональна схема складається з функціональних блоків, які являють собою конструктивно відособлені частини (елементи або пристрої) автоматичних систем, які виконують певні функції. Функціональні блоки на схемі позначають прямокутниками, всередині яких надписують їх найменування відповідно до функцій, що виконуються. Зв'язки між функціональними блоками (внутрішні впливи) позначаються лініями зі стрілками, які вказують напрям впливів.

Функціональні схеми можуть виконуватися в укрупненому і розгорненому вигляді. У першому випадку на схемі зображають найважливіші блоки системи і зв'язки між ними.

У другому варіанті схема відображається більш детально, що полегшує її читання та ілюструє принцип роботи.

Основні елементи схем алгоритму це термінатор, процес, рішення, зумовлений процес (підпрограма), дані та з'єднувач.

КПЗ-2024

Термінатор це елемент відображає вхід із зовнішнього середовища або вихід з неї (найчастіше застосування – початок і кінець програми). Всередині фігури записується відповідна дія.

Процес це виконання однієї або кількох операцій, обробка даних будь-якого виду (зміна значення даних, форми подання, розташування). Всередині фігури записують безпосередньо самі операції.

Рішення це показує рішення або функцію перемикального типу з одним входом і двома або більше альтернативними виходами, з яких тільки один може бути обраний після обчислення умов, визначених всередині цього елементу. Вхід в елемент позначається лінією, що входить зазвичай у верхню вершину елементу. Якщо виходів два чи три то зазвичай кожен вихід позначається лінією, що виходить з решти вершин (бічних і нижній). Якщо виходів більше трьох, то їх слід показувати однією лінією, що виходить з вершини (частіше нижній) елемента, яка потім розгалужується. Відповідні результати обчислень можуть записуватися поруч з лініями, що відображають ці шляхи.

Зумовлений процес (підпрограма) це символ відображає виконання процесу, що складається з однієї або кількох операцій, що визначені в іншому місці програми (у підпрограмі, модулі). Всередині символу записується назва процесу і передані в нього дані.

Дані це перетворення у форму, придатну для обробки (введення) або відображення результатів обробки (виведення). Цей символ не визначає носія даних (для вказівки типу носія даних використовуються специфічні символи).

З'єднувач це символ відображає вихід в частину схеми і вхід з іншої частини цієї схеми. Використовується для обриву лінії та продовження її в іншому місці (приклад: поділ блок-схеми, що не поміщається на листі). Відповідні сполучні символи повинні мати одне (при тому унікальне) позначення.

Первинною стадією без якої не відбувається розробка програмного забезпечення це звичайно розробка блок-схем. На рисунку 4.1 зображена основна блок-схема програми, на рисунку 4.2 зображено роботу підпрограми.

КПЗ_2024

З якої видно що робота основної програми складається з початкових етапів ініціалізації ПЗ, перевірки наявності ресурсів системи, блоку початку основного циклу з чеканням запиту від користувача в якому відбувається виклик підпрограми та останньої стадії – перевірка поточного стану з завершенням роботи розробленого ПЗ.

При роботі підпрограми виконується основний функціонал системи з циклічними послідовностями, перевіркою поточного стану та поверненням в основну програму прапорів стану виконання.

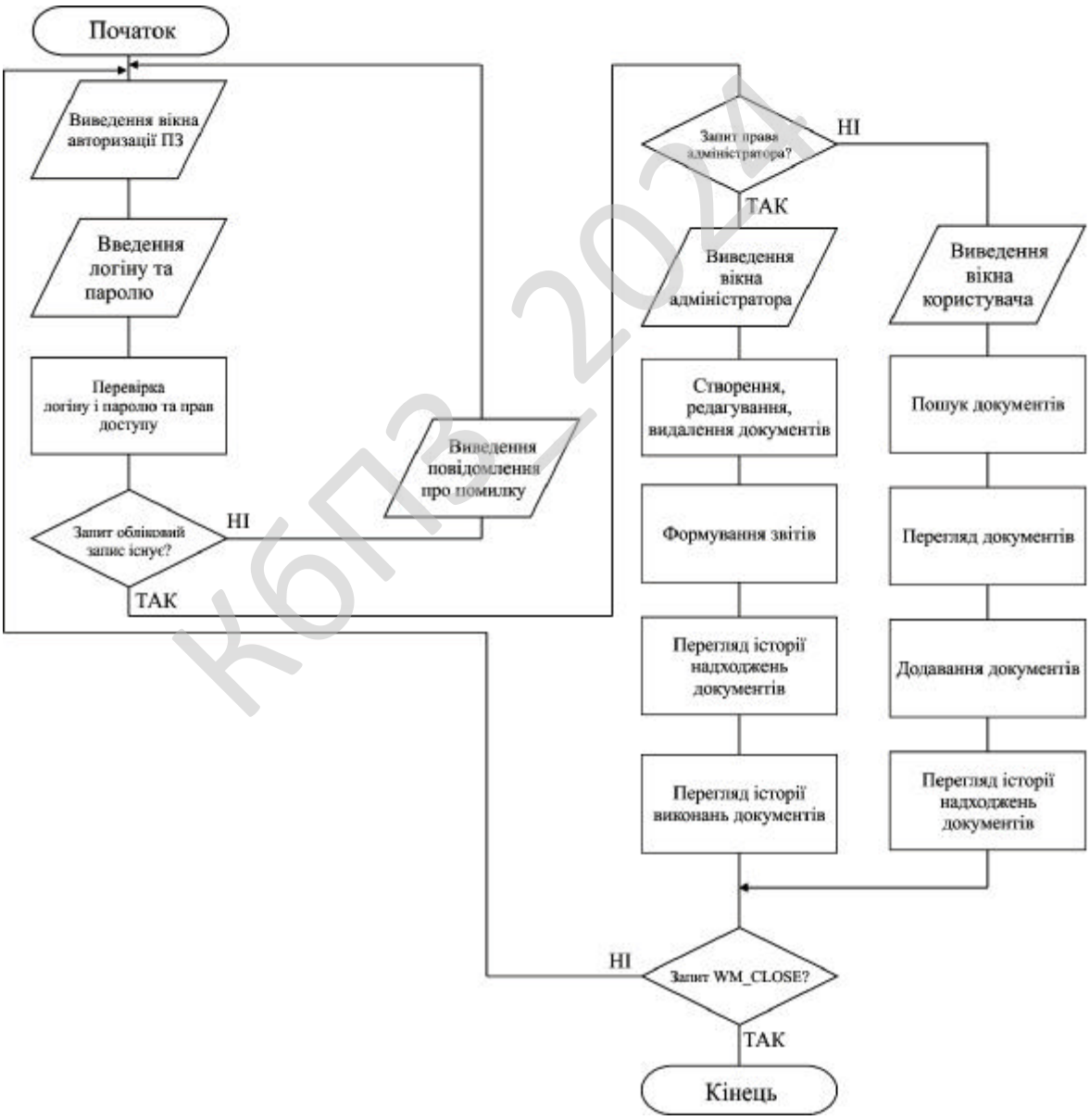

Рисунок 4.1 – Блок-схема основної програми

КПІ 2021

Також при розробці магістерської дипломної роботи було використано наступні підходи UML:

– діаграма діяльності (діаграми поведінки типу);

– діаграма прецедентів (діаграми поведінки типу);

– діаграма класів;

– діаграма компонент;

– діаграма об'єктів;

– діаграма розгортання.

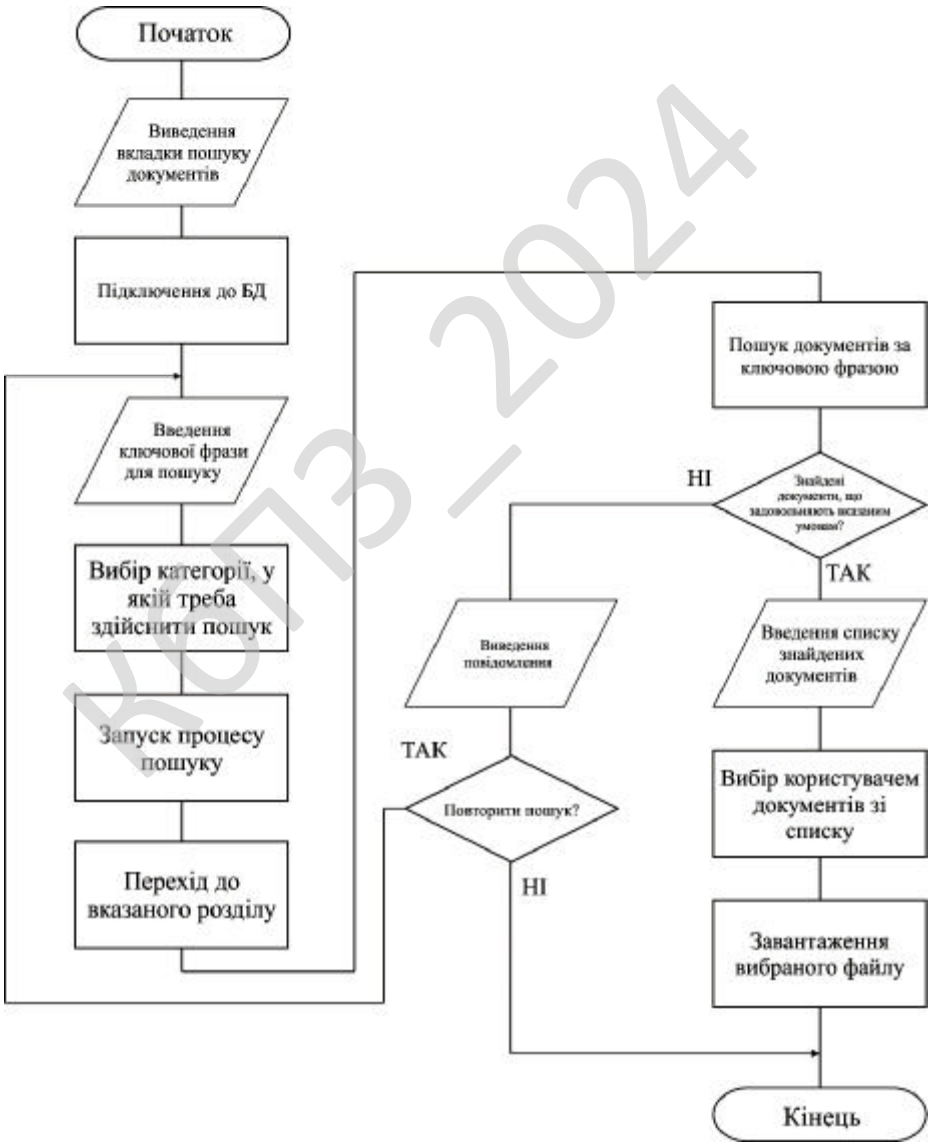

Рисунок 4.2 – Блок-схема роботи підпрограми

КОПЗ_2024

Діаграма діяльності. Це візуальне представлення графу діяльностей. Граф діяльностей є різновидом графу станів скінченного автомату, вершинами якого є певні дії, а переходи відбуваються по завершенню дій.Дія є фундаментальною одиницею визначення поведінки в специфікації. Дія отримує множину вхідних сигналів, та перетворює їх на множину вихідних сигналів.

Одна із цих множин, або обидві водночас, можуть бути порожніми. Виконання дії відповідає виконанню окремої дії. Подібно до цього, виконання діяльності є виконанням окремої діяльності, буквально, включно із виконанням тих дій, що містяться в діяльності. Кожна дія в діяльності може виконуватись один, два, або більше разів під час одного виконання діяльності. Щонайменше, дії мають отримувати дані, перетворювати їх та тестувати, деякі дії можуть вимагати певної послідовності.

Специфікація діяльності (на вищих рівнях сумісності) може дозволяти виконання декількох (логічних) потоків, та існування механізмів синхронізації для гарантування виконання дій у правильному порядку.

Діаграма прецедентів це діаграма, на якій зображено відношення між акторами та прецедентами в системі. Також, перекладається якдіаграма варіантів використання.

Діаграма прецедентів є графом, що складається з множини акторів, прецедентів (варіантів використання) обмежених границею системи (прямокутник), асоціацій між акторами та прецедентами, відношень серед прецедентів, та відношень узагальнення між акторами. Діаграми прецедентів відображають елементи моделі варіантів використання.

Суть даної діаграми полягає в наступному: проектована система представляється у вигляді безлічі сутностей чи акторів, що взаємодіють із системою за допомогою так званих варіантів використання. Варіант використання (use case) використовують для описання послуг, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який виконує система при діалозі з актором.

КОПЗ-2024

При цьому нічого не говориться про те, яким чином буде реалізована взаємодія акторів із системою.

У мові UML є кілька стандартних видів відношень між акторами і варіантами використання:

– асоціації (association relationship);

– включення (include relationship);

– розширення (extend relationship);

– узагальнення (generalization relationship).

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (include) у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов.

Діаграма класів це статичне представлення структури моделі. Відображає статичні (декларативні) елементи, такі як: класи, типи даних, їх зміст та відношення.

Діаграма класів, також, може містити позначення для пакетів та може містити позначення для вкладених пакетів. Також, діаграма класів може містити позначення деяких елементів поведінки, однак їх динаміка розкривається в інших типах діаграм.

КПЗ – 2024

Діаграма класів (class diagram) служить для представлення статичної структури моделі системи в термінології класів об'єктно-орієнтованого програмування. На цій діаграмі показують класи, інтерфейси, об'єкти й кооперації, а також їхні відносини.

В UML існують наступні типи зв'язків які використовуються у діаграмі класів: Асоціації; Агрегація; Композиція.

Асоціації це якщо між двома класами визначена асоціація, то можна переміщатися від об'єктів одного класу до об'єктів іншого. Цілком припустимі випадки, коли обидва кінці асоціації відносяться до одного і того ж класу. Це означає, що з об'єктом деякого класу дозволено зв'язати інші об'єкти з того ж класу. Асоціація, що зв'язує два класи, називається бінарної. Можна, хоча це рідко буває необхідним, створювати асоціації, що зв'язують відразу кілька класів. Графічно асоціація зображується у вигляді лінії, що з'єднує клас сам з собою або з іншими класами.

Асоціації може бути присвоєно ім'я, яке описує природу відносини. Зазвичай ім'я асоціації не вказується, якщо тільки ви не хочете явно задати для неї рольові імена або у вашій моделі настільки багато асоціацій, що виникає необхідність посилатися на них і відрізняти один від одного. Ім'я буде особливо корисним, якщо між одними і тими ж класами існує кілька різних асоціацій.

Клас, що бере участь в асоціації, грає в ній деяку роль. По суті, це "обличчя", яким клас, що знаходиться на одній стороні асоціації, звернений до класу з іншого її боку. Можна явно позначити роль, яку клас грає в асоціації.

Часто при моделюванні буває важливо вказати, скільки об'єктів може бути пов'язано допомогою одного примірника асоціації. Це число називається кратністю (Multiplicity) ролі асоціації та записується або як вираз, значенням якого є діапазон значень, або в явному вигляді.

Вказуючи кратність на одному кінці асоціації, ви тим самим говорите, що на цьому кінці саме стільки об'єктів повинно відповідати кожному об'єкту на протилежному кінці. Кратність можна задати рівною одиниці (1), можна вказати

КПЗ-2024

діапазон: "нуль або одиниця" (0..1), "багато" (0 .. *), "одиниця або більше" (1 .. *). Дозволяється також вказувати певне число (наприклад, 3). За допомогою списку можна задати і більш складні кратності, наприклад 0. . 1, 3..4, 6 .. *, що означає "будь-яке число об'єктів, крім 2 і 5".

Агрегація це проста асоціація між двома класами відображає структурний відношення між рівноправними сутностями, коли обидва класу знаходяться на одному концептуальному рівні і ні один не є більш важливим, ніж інший. Але іноді доводиться моделювати відношення типу «частина/ціле», в якому один з класів має більш високий ранг (ціле) і складається з декількох менших за рангом (частин).

Ставлення такого типу називають агрегацією; воно зараховане до відносин типу «має» (з урахуванням того, що об'єкт-ціле має кілька об'єктів-частин). Агрегація є окремим випадком асоціації і зображується у вигляді простої асоціації з незафарбованим ромбом з боку «цілого». Графічно агрегація представляється порожнім ромбом на блоці класу, і лінією, яка від цього ромба до міститься класу.

Композиція це більш суворий варіант агрегації. Відома також як агрегація за значенням.

Композиція має жорстку залежність часу існування екземплярів класу контейнера та примірників містяться класів. Якщо контейнер буде знищений, то весь його вміст буде також знищено. Графічно представляється як і агрегація, але з зафарбовані ромбиком.

Діаграма компонент в UML це діаграма, на якій відображаються компоненти, залежності та зв'язки між ними.

Діаграма компонент відображає залежності між компонентами програмного забезпечення, включаючи компоненти вихідних кодів, бінарні компоненти, та компоненти, що можуть виконуватись.

Модуль програмного забезпечення може бути представлено в якості компоненти. Деякі компоненти існують під час компіляції, деякі – під час компонування, а деякі під час роботи програми.

КОПЗ_2024

Діаграма компонент відображає лише структурні характеристики, для відображення окремих екземплярів компонент слід використовувати діаграму розгортування.

Компоненти об'єднуються разом використовуючи структурні зв'язки (assembly connector) щоб об'єднати інтерфейси двох компонент. Це ілюструє зв'язок типу «клієнт-сервер».

Структурна взаємодія – «зв'язок двох компонент, який передбачає, що один з них надає послуги, потрібні іншому компоненту».

При використанні діаграми компонент щоб показати внутрішню структуру компонента, клієнтські та серверні інтерфейси можуть утворювати пряме з'єднання з внутрішніми. Таке з'єднання називається з'єднанням делегації.

Діаграма об'єктів в UML це діаграма, що відображає об'єкти та їх зв'язки в певний момент часу. Діаграма об'єктів може розглядатись як окремий випадок діаграми класів, на якій можуть бути представлені як класи, так і екземпляри (об'єкти) класів. Схожою за змістом є діаграма взаємодії (collaboration diagram).

Діаграми об'єктів не мають власної нотації. Оскільки діаграми класів можуть відображати об'єкти, то діаграма класів, на якій відображено лише об'єкти, та не відображено класи, може вважатись діаграмою об'єктів.

Діаграма об'єктів відображає об'єкти та зв'язки в певний момент роботи програми. Об'єкти можуть містити інформацію про власні значення а не про описання. Для відображення загальних шаблонів об'єктів та зв'язків, що можуть багаторазово створюватись під час роботи програми, слід використовувати діаграму взаємодії, яка може відображати характеристики об'єктів та зв'язків. Екземпляр діаграми взаємодії створює діаграму об'єктів.

Діаграма об'єктів не відображає еволюцію системи під час роботи. Натомість, слід використовувати діаграми взаємодії з повідомленнями, або діаграми послідовності.

Діаграма розгортання (deployment diagram) це діаграма в UML, на якій відображаються обчислювальні вузли під час роботи програми, компоненти, та

КОПЗ-2024

об'єкти, що виконуються на цих вузлах. Компоненти відповідають представленню робочих екземплярів одиниць коду. Компоненти, що не мають представлення під час роботи програми на таких діаграмах не відображаються; натомість, їх можна відобразити на діаграмах компонент. Діаграма розгортання відображає робочі екземпляри компонент, а діаграма компонент, натомість, відображає зв'язки між типами компонент.

Приведемо частину програмного коду, яка реалізує функції головного вікна.

```
/*****************************************************/
// DESCRIPTION:    Конструктор класу
/*****************************************************/
CWorker::CWorker(void)
{
  listDoc = gcnew ArrayList();
  logInput = gcnew CLogger();
  logOutput = gcnew CLogger();
}
/*****************************************************/
// NAME:       GetIndexByISBNHash
// DESCRIPTION:    Функція одержання індексу документа в загальному списку по
вхідному номеру документа
// INPUT:      ddHashValue – значення хеша вхідного номера документа
/*****************************************************/
Int32 CWorker::GetIndexByISBNHash(Int32 HashValue)
{
// Цикл пошуку документа із заданим вхідним номером документа
  for(Int32 i = 0; i < listDoc->Count; i++)
  {
// Хеш-значення вхідного номера документа заданого документа збігається з
// хеш-значенням вхідного номера документа знайденого документа
     if(HashValue == ((CDoc^)listDoc[i])->GetISBNHash())
     {
// Документ знайдений, повернути індекс
          return i;
     }
  }
// Документ не знайдений, повернути -1
  return -1;
}
```

КОПЗ_2024

```
/*****************************************************/
// DESCRIPTION:    Функція додавання нового документа
/*****************************************************/
Boolean CWorker::AddDoc(CDoc^ SourceDoc)
{
// Документи з таким вхідним номером документа не існує?
  if(this->GetIndexByISBNHash(SourceDoc->GetISBNHash()) == -1)
  {
// Додати заданній документ у загальний список
     listDoc->Add(SourceDoc);
// Записати подія в лог
     logInput->WriteEvent(
          "Новий документ '" + ((CDoc^)SourceDoc)->GetName() +
          "', Автор '" + ((CDoc^)SourceDoc)->GetAuthor() +
          "', Вхідний номер документа " + ((CDoc^)SourceDoc)->GetISBN());
// Функція відробила успішно
     return true;
  }
  else
  {
// Документ із таким же вхідним номером документа існує, повернути помилку
     return false;
  }
}
/*****************************************************/
// DESCRIPTION:    Функція видалення документів із системи
/*****************************************************/
Int32 CWorker::RemoveDoc(Int32 ddISBNHash)
{
// Знайти документ по вхідному номеру документа
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);
// Документ із таким вхідним номером документа існує?
  if(ddIndex != -1)
  {
// Перевірити, що жодного документа немає в користувача
     if(((CDoc^)listDoc[ddIndex])->GetTotalNumber() ==
          ((CDoc^)listDoc[ddIndex])->GetFreeNumber())
     {
// Записати подія в лог
     logOutput->WriteEvent(
     "Виконаний документ '" + ((CDoc^)listDoc[ddIndex])->GetName() +
     "', Автор '" + ((CDoc^)listDoc[ddIndex])->GetAuthor() +
```

КОПЗ_2024

```
     "', Вхідний номер документа " + ((CDoc^)listDoc[ddIndex])->GetISBN());
// Видалити знайдений документ зі списку
          listDoc->RemoveAt(ddIndex);
// Функція відробила успішно
          return 0;
     }
     else
     {
// Повернути помилку, не всі документи перебувають у системі
     return 1;
     }
  }
  else
  {
// Повернути помилку, документ із таким вхідним номером документа не був знайдений
  return 2;
  }
}
/*****************************************************/
// DESCRIPTION:    Функція видачі документів користувачеві
/*****************************************************/
Boolean CWorker::GiveDoc(ArrayList^ listReader, Int32 ddISBNHash)
{
// Визначити індекс по вхідному номеру документа
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);
// Зменшити кількість вільних документів
  if((ddIndex != -1) && ((CDoc^)listDoc[ddIndex])->DecFreeNumber())
  {
// Додати документ у список користувача
     listReader->Add(listDoc[ddIndex]);
// Функція відробила успішно
     return true;
  }
  else
  {
// Такого документа не є в наявності
     return false;
  }
}
/*****************************************************/
// DESCRIPTION:    Функція прийняття документів від користувача назад у систему
/*****************************************************/
```

КОПІЯ_2024

```
void CWorker::TakeDoc(ArrayList^ listReader, Int32 ddISBNHash)
{
// Визначити індекс по вхідному номеру документа (документ існує,
// тому що його вхідний номер документа був повідомлений користувачеві)
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);
// Видалити документ зі списку користувача
// listReader->RemoveAt(ddIndex);
// Збільшити кількість вільних документів
  ((CDoc^)listDoc[ddIndex])->IncFreeNumber();
}
/*****************************************************/
// DESCRIPTION:    Функція завантаження документів системи з файлу
/*****************************************************/
void CWorker::LoadDocList()
{
// Перевірити існування файлу
  if(!File::Exists(DOCS_FILE))
  {
// Файл не знайдений, закінчити виконання функції
     return;
  }
// Відкрити файл
  StreamReader^ srDoc = gcnew StreamReader(DOCS_FILE);
// Продовжувати, поки не буде знайдений кінець файлу
  while(srDoc->EndOfStream == false)
  {
// Створити новий об'єкт "документ"
     CDoc^ NewDoc = gcnew CDoc();
// Завантажити з файлу й установити параметри документа
     NewDoc->SetName(srDoc->ReadLine());
     NewDoc->SetAuthor(srDoc->ReadLine());
     NewDoc->SetISBN(srDoc->ReadLine());
     NewDoc->SetTheme(srDoc->ReadLine());
     NewDoc->SetPages(Convert::ToInt32(srDoc->ReadLine()));
     NewDoc->SetTotalNumber(Convert::ToInt32(srDoc->ReadLine()));
     NewDoc->SetFreeNumber(Convert::ToInt32(srDoc->ReadLine()));
// Включити документ у загальний список документів
     listDoc->Add(NewDoc);
}
// Закрити файл
  srDoc->Close();
}
```

КОПЗ_2024

```
/*****************************************************/
// DESCRIPTION:    Функція збереження документів системи у файл
/*****************************************************/
void CWorker::SaveDocList()
{
// Перевірити існування файлу
  if(File::Exists(DOCS_FILE))
  {
// Видалити файл
     File::Delete(DOCS_FILE);
  }
// Створити й відкрити файл
  StreamWriter^ swDoc = gcnew StreamWriter(DOCS_FILE);
// Цикл запису по одному документу
  for(Int32 i = 0; i < listDoc->Count; i++)
  {
// Одержати параметри документа й записати їх у файл
     swDoc->WriteLine(((CDoc^)listDoc[i])->GetName());
     swDoc->WriteLine(((CDoc^)listDoc[i])->GetAuthor());
     swDoc->WriteLine(((CDoc^)listDoc[i])->GetISBN());
     swDoc->WriteLine(((CDoc^)listDoc[i])->GetTheme());
     swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])->GetPages()));
     swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])->GetTotalNumber()));
     swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])->GetFreeNumber()));
  }
// Закрити файл
  swDoc->Close();
}
/*****************************************************/
// DESCRIPTION:    Функція пошуку документа по заданих параметрах
/*****************************************************/
ArrayList^ CWorker::FindDoc(String^ strFindValue)
{
// Список для результату
  ArrayList^ listResult = gcnew ArrayList();
// Шукане значення без обліку регістра
  String^ strValue = strFindValue->ToLower();
// Задана порожній рядок?
  if(String::IsNullOrEmpty(strValue->Trim()))
{
//Вивести весь список
     for(Int32 i = 0; i < listDoc->Count; i++)
```

КбПЗ_2024

```
{
// Додати документ у результуючий список
     listResult->Add(listDoc[i]);
  }}
  else
  {
// Вибрати з умовою
     for(Int32 i = 0; i < listDoc->Count; i++)
     {
// Пошук рядка в кожному полі без обліку регістра
if(((((CDoc^)listDoc[i])->GetName())->ToLower())->IndexOf(strValue) != -1 ||
  ((((CDoc^)listDoc[i])->GetAuthor())->ToLower())->IndexOf(strValue) != -1 ||
  ((((CDoc^)listDoc[i])->GetISBN())->ToLower())->IndexOf(strValue) != -1 ||
  ((((CDoc^)listDoc[i])->GetTheme())->ToLower())->IndexOf(strValue) != -1 ||
  Convert::ToString((((CDoc^)listDoc[i])->GetPages()))->IndexOf(strValue) != -1)
{
// Додати документ у результуючий список
              listResult->Add(listDoc[i]);
          }
     }
  }
// Повернути список збігів
  return listResult;
}
/*****************************************************/
// DESCRIPTION:    Функція перегляду інформації про заданий документ
/*****************************************************/
CDoc^ CWorker::ViewDoc(Int32 ddISBNHash)
{
// Визначити індекс по вхідному номеру документа (документ існує,
// тому що його вхідний номер документа був повідомлений користувачеві)
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);
  // Повернути елемент "документ" із загальне списку
  return (CDoc^)listDoc[ddIndex];
}
/*****************************************************/
// DESCRIPTION:    Функція зчитування історії з файлу
/*****************************************************/
void CWorker::ReadLogs()
{
// Файл із історією надходжень існує?
  if(File::Exists(LOG_INPUT))
```

КОПЗ_2024

```
  {
// Зчитування історії надходжень
     logInput->strLog = File::ReadAllText(LOG_INPUT);
  }
  else
  {
     logInput->strLog = "";
  }
// Файл із історією списань існує?
  if(File::Exists(LOG_INPUT))
  {
// Зчитування історії списань
     logOutput->strLog = File::ReadAllText(LOG_OUTPUT);
  }
  else
{    logOutput->strLog = ""; }}
/*****************************************************/
// DESCRIPTION:    Функція збереження історії у файл
/*****************************************************/
void CWorker::WriteLogs()
{
  // Запис історії
  File::WriteAllText(LOG_INPUT, logInput->strLog);
  File::WriteAllText(LOG_OUTPUT, logOutput->strLog);
}
/*****************************************************/
// DESCRIPTION:    Функція очищення історії
/*****************************************************/
void CWorker::ClearLogs()
{ logInput->strLog = "";
  logOutput->strLog = "";}
/*****************************************************/
// DESCRIPTION:    Функція перегляду історії надходжень
/*****************************************************/
String^ CWorker::ViewInputLog()
{ return logInput->strLog;}
/*****************************************************/
// DESCRIPTION:    Функція перегляду історії списань
/*****************************************************/
String^ CWorker::ViewOutputLog()
{ return logOutput->strLog;}
```

КОПЗ_2024

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм Madryga. Алгоритм Madryga складається із двох вкладених циклів. Зовнішній цикл повторюється вісім разів (для гарантії надійності число циклів можна збільшити) і полягає в застосуванні внутрішнього циклу до відкритого тексту. Внутрішній цикл перетворює відкритий текст у шифртекст і виконується однократно над кожним 8-бітовим блоком (байтом) відкритого тексту. Таким чином, весь відкритий текст послідовно вісім разів обробляється алгоритмом.

Ітерація внутрішнього циклу оперує з 3-байтовим вікном даних, називаним робочим кадром (рисунок 4.3).

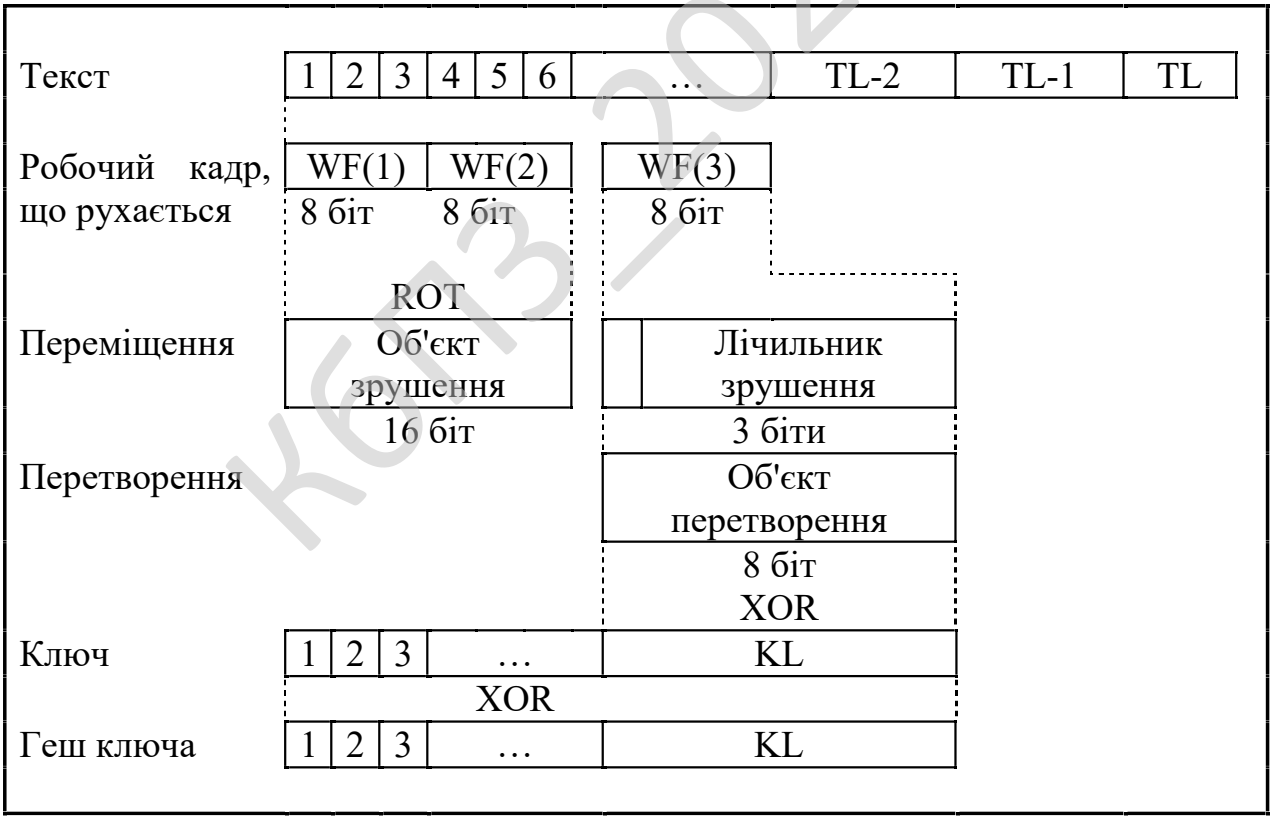

Рисунок 4.3 – Одна ітерація алгоритму Madryga

КБПЗ_2024

Це вікно зрушується на 1 байт за ітерацію. (При роботі з останніми 2 байтами дані покладаються циклічно замкнутими). Перші два байти робочого кадру циклічно зрушуються на змінне число позицій, а для останнього байта виконується операція XOR з декількома бітами ключа. У міру переміщення робочого кадру всі байти послідовно циклічно зрушуються й піддаються операції XOR із частинами ключа. Послідовні циклічні зрушення перемішують результати попередніх операцій XOR і циклічного зрушення, причому на циклічне зрушення впливають результати XOR. Завдяки цьому процес у цілому оборотний.

Оскільки кожний байт даних впливає на два байти ліворуч і на один байт праворуч від себе, після восьми проходів кожний байт шифртексту залежить від 16 байтів ліворуч і 8 байтів праворуч.

При шифруванні кожна ітерація внутрішнього циклу встановлює робочий кадр на передостанній байт відкритого тексту й циклічно переміщає його до третього з кінця байту відкритого тексту. Спочатку весь ключ піддається операції XOR з випадковою константою й потім циклічно зрушується вправо на 3 біти (ключ і дані рухаються в різних напрямках, щоб мінімізувати надлишкові операції з бітами ключа). Молодші три біти молодшого байта робочого кадру зберігаються, вони визначають циклічне зрушення інших двох байтів. Далі конкатенація двох старших байтом циклічно зрушується вліво на змінне число біт (від 0 до 7). Потім над молодшим байтом робочого кадру виконується операція XOR з молодшим байтом ключа. Нарешті робочий кадр зміщається вправо на один байт і весь процес повторюється.

Випадкова константа призначена для перетворення ключа в псевдовипадкову послідовність. Довжина константи повинна бути рівній довжині ключа. При обміні даними абоненти повинні користуватися однієї й тією же константою. Для 64-бітового ключа Madryga рекомендує константу 0x0fle2d3c4b5a6978.

КПЗ-2024

При розшифруванні процес повторюється у зворотному порядку. У кожній ітерації внутрішнього циклу робочий кадр установлюється на байт, третій ліворуч від останнього байта шифртексту, і циклічно зрушується у зворотному напрямку до байта, розташованого на 2 байти уліво відносно останнього байта шифртексту. 2 байти шифртексту в процесі циклічно зрушуються вправо, а ключ – уліво. Після циклічних зрушень виконується операція XOR.

КБПЗ_2024

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

На рисунку 5.1 зображено інтерфейс програмного забезпечення, розробленого у результаті виконання магістерської дипломної роботи. Обчислення виконується через консольний інструмент з подальшою передачею результатів до інтерфейсу. Розроблене програмне забезпечення системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу складається з наступних функціональних блоків:

– Навігаційне меню: Файл; Довідка.

– Функції представлені у графічному вигляді.

– Вікна з вкладками: Пошук; Документи; Історія.

– Вікно виведення результату роботи системи.

– Навігаційного меню яке виводиться на екран натисканням правої клавіші маніпулятора миші на якому знаходиться переходи на основні вікна розробленого ПЗ.

– Функціональних кнопок ПЗ.

КБПЗ_2024

Рисунок 5.1 – Головне вікно розробленого ПЗ

Для перегляду короткої довідки про програму слід натиснути на основному вікні кнопку авторського права, після чого на екрані з’явиться вікно показане на рисунку 5.2.

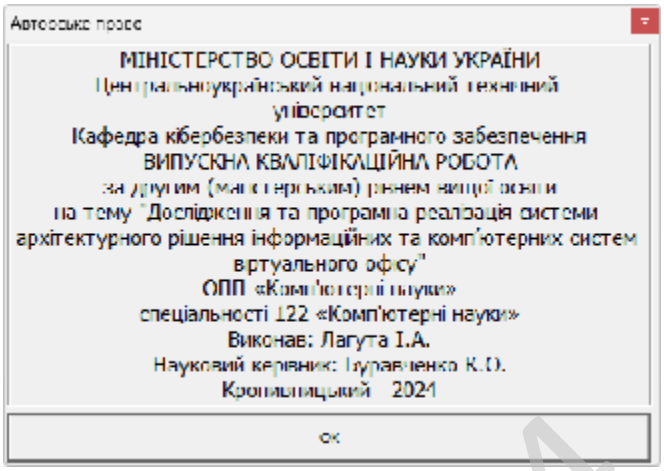

Рисунок 5.2 – Вікно розробника ПЗ

Під час роботи над програмою було проведено тестування програмного забезпечення, тобто технічне дослідження, призначене для виявлення інформації про якість продукту відносно контексту, в якому воно має використовуватись.

Тестування включає як процес пошуку помилок або інших дефектів, так і випробування програмних складових з метою їх оцінки.

Проводилась оцінка:

– відповідності поставленим вимогам;

– правильна відповідь для усіх можливих вхідних даних;

– виконання функцій за прийнятний час;

– практичність;

– сумісність з ОС та стороннім ПЗ.

Оскільки число можливих тестів для програмних компонент практично нескінченне, тому стратегія тестування полягала в тому, щоб провести всі можливі тести з урахуванням наявного часу та ресурсів.

КПЗ_2024

Як результат ПЗ тестувалось стандартним виконанням програми з метою виявлення помилок або інших дефектів.

Проводилось тестування форматом білої скриньки та чорної скриньки.

Тестування форматом білої скриньки засноване на аналізі керуючої структури програми. Програма вважається повністю перевіреною, якщо проведено вичерпне тестування маршрутів (шляхів) її графа управління.

У цьому випадку формуються тестові варіанти, в яких:

– Гарантується перевірка всіх незалежних маршрутів програми.

– Знаходяться гілки True, False для всіх логічних рішень.

– Виконуються всі цикли (у межах їхніх кордонів та діапазонів).

– Аналізується правильність внутрішніх структур даних.

Недоліки тестування "білої скриньки":

– Кількість незалежних маршрутів може бути дуже велика.

– Повне тестування маршрутів не гарантує відповідності програми вихідним вимогам до неї.

– У програмі можуть бути пропущені деякі маршрути.

– Не можна виявити помилки, поява яких залежить від даних.

Переваги тестування "білої скриньки" пов'язані з тим, що принцип «білої скриньки» дозволяє врахувати особливості програмних помилок:

– Кількість помилок мінімально в «центрі» і максимально на «периферії» програми.

– Попередні припущення про ймовірність потоку керування або даних у програмі часто бувають некоректними. У результаті типовим може стати маршрут, модель обчислень за яким опрацьована слабо.

– При записі алгоритму програмного забезпечення у вигляді тексту на мові програмування можливе внесення типових помилок трансляції (синтаксичних та семантичних).

– Деякі результати в програмі залежать не від вихідних даних, а від внутрішніх станів програми.

КБПЗ–2024

Проводилось тестування чорної скриньки.

Основне місце програми тестів «чорної скриньки» – інтерфейс ПЗ. Відомі: функції програми. Досліджується: робота кожної функції на всій області визначення.

Ці тести демонструють:

– Як виконуються функції програми.

– Як приймаються вихідні дані.

– Як виробляються результати.

– Як зберігається цілісність зовнішньої інформації.

При тестуванні «чорної скриньки» розглядаються системні характеристики програм, ігнорується їхня внутрішня логічна структура. Вичерпне тестування, як правило, неможливе.

Наприклад, якщо в програмі 10 вхідних величин і кожна приймає по 10 значень, то кількість тестових варіантів становитиме 10^10. Тестування «чорної скриньки» не реагує на багато особливостей програмних помилок.

Тестування «чорної скриньки» (функціональне тестування) дозволяє отримати комбінації вхідних даних, які забезпечують повну перевірку всіх функціональних вимог до програми.

Програмний виріб тут розглядається як «чорна скринька», чию поведінку можна визначити тільки дослідженням його входів та відповідних виходів. При такому підході бажано мати:

– Набір, утворений такими вхідними даними, які призводять до аномалій у поведінці програми (назвемо його ІТс).

– Набір, утворений такими вхідними даними, які демонструють дефекти програми (назвемо його ОТ).

Будь-який спосіб тестування «чорної скриньки» повинен:

– Виявити такі вхідні дані, які з високою ймовірністю належать набору ІТс;

– Сформулювати такі очікувані результати, які з високою імовірністю є елементами набору ОТ.

КОПЗ-2024

Принцип «чорної скриньки» не альтернативний принципу «білої скриньки». Скоріше це доповнює підхід, який виявляє інший клас помилок.

Тестування «чорної скриньки» забезпечує пошук наступних категорій помилок:

– Некоректних чи відсутніх функцій;

– Помилок інтерфейсу;

– Помилок у зовнішніх структурах даних або в доступі до зовнішньої бази даних;

– Помилок характеристик (необхідна ємність пам'яті і т.д.);

– Помилок ініціалізації та завершення.

Обрано умови розповсюдження – Shareware.

Під умовно-безплатним програмним забезпеченням можна розуміти спосіб або метод розповсюдження комерційного ПЗ на ринку (тобто на шляху до кінцевого користувача), при якому випробувачеві пропонується обмежена за можливостями (не повнофункціональна або демонстраційна версія), терміном дії (тріал версія) або версія з вбудованим набридливим нагадуванням про необхідність оплати використання програми. В угоді про використання (ліцензії для кінцевого користувача, EULA) також може бути обумовлена заборона на комерційне або професійне (не тестове) її використання. Основний принцип умовно-безплатного ПЗ – «спробуй, перш ніж купити» (try before you buy). ПЗ що поширюється як умовно-безплатний, надається користувачам безоплатно. Звичайно користувач платить тільки за час завантаження файлів через Інтернет або за носій (CD диск, флешку, ключ). Протягом певного терміну, що становить зазвичай тридцять днів, він може користуватися програмою, тестувати її, освоювати її можливості. Якщо після закінчення цього терміну користувач вирішить продовжити використання ПЗ, він зобов'язаний купити його (зареєструватися), заплативши авторові певну суму. В іншому випадку користувач повинен припинити використання ПЗ та видалити його зі свого комп'ютера.

КПЗ–2024

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Метою розробки є дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Об'єктом дослідження є процес архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Предметом дослідження є методи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Методи дослідження базуються на методах хмарних технологій, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

– Розроблено вітчизняний продукт архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу, який має більш широкі можливості, на відміну від існуючих аналогів.

КОПЗ-2024

7 МАРКЕТИНГОВЕ ТА ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ ІТ-ПРОЄКТУ

7.1 Визначення цільової аудиторії кінцевого готового продукту

Результати дослідження та програмної реалізації системи архітектурного рішення для інформаційних та комп'ютерних систем віртуального офісу можуть бути корисні для широкої аудиторії, котра може стати потенційними клієнтами (рисунок 7.1).

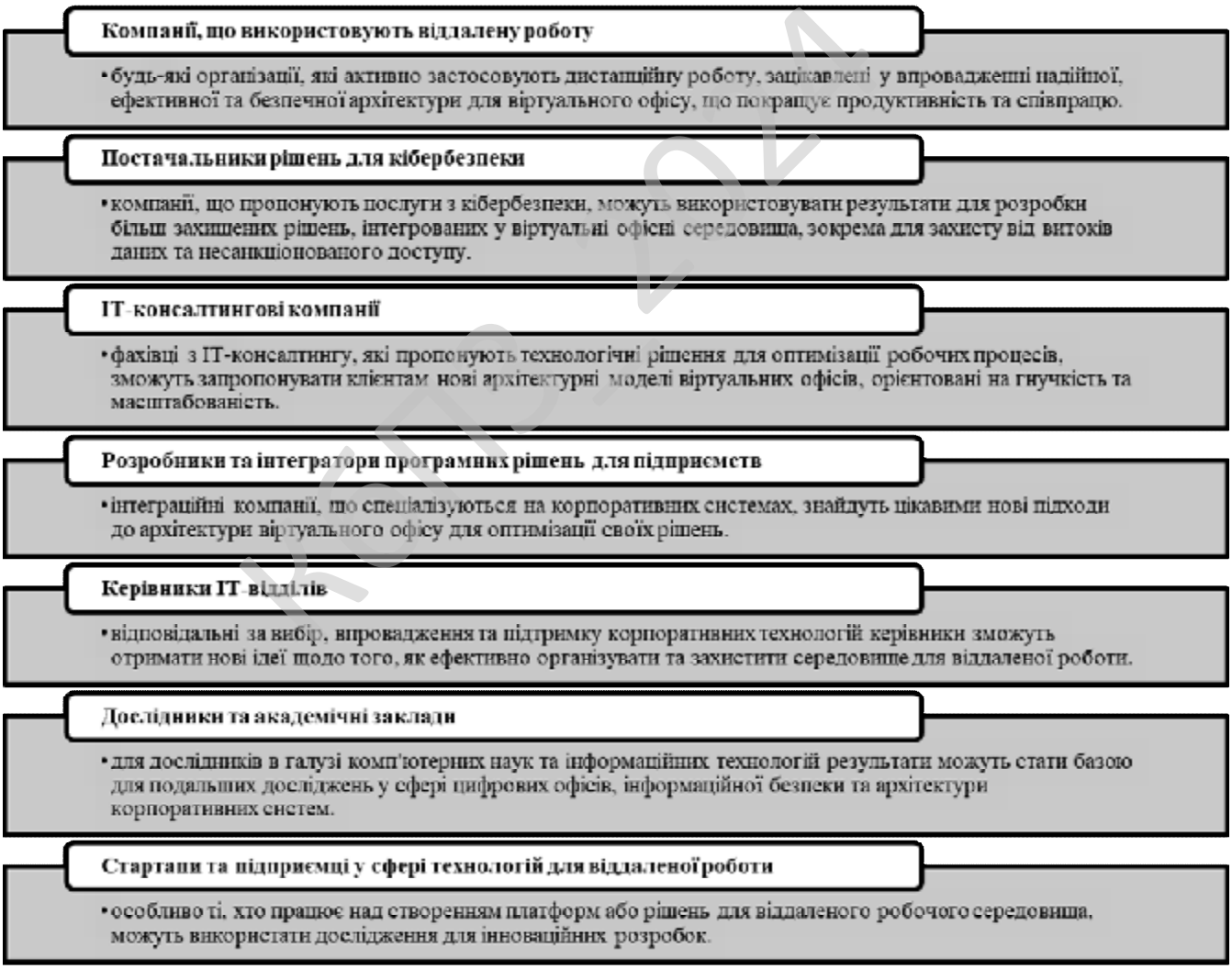

Рисунок 7.1 – Цільова аудиторія

Кожна з цих груп зацікавлена у покращенні функціоналу, безпеки, доступності та адаптивності віртуальних робочих просторів, що робить результати дослідження актуальними для широкого кола учасників.

7.2 Оцінка привабливості шляхом застосування методів експертних оцінок

Оцінку привабливості програмної реалізації системи архітектурного рішення для інформаційних та комп'ютерних систем віртуального офісу можна виконати за допомогою методу експертних оцінок. Нижче наведено приклад такого підходу.

Для об'єктивного експертного оцінювання визначаємо ключові критерії, за якими оцінюватиметься система. Прикладами критеріїв можуть бути:

Рисунок 7.2 – Критерії експертної оцінки

Збираємо команду з різнопрофільних фахівців, що володіють відповідними знаннями для оцінки таких систем. До експертної групи ввійдуть: IT-архітектори, фахівці з інформаційної безпеки, системні адміністратори, керівники проєктів у сфері IT, представники користувачів. Далі проводимо збір експертних оцінок. Кожен експерт оцінює систему за критеріями за шкалою від 1 до 5: 1 – дуже погано, 2 – погано, 3 – задовільно, 4 – добре, 5 – відмінно.

КПІ-2024

Таблиця 7.1 – Зведені результати експертних оцінок

Критерій	Експерт 1	Експерт 2	Експерт 3	Середня оцінка
Функціональність	5	4	4	4.33
Безпека	4	5	4	4.33
Продуктивність	4	4	5	4.33
Масштабованість	5	4	5	4.67
Зручність використання	3	4	3	3.33

Важливість кожного критерію може відрізнятися, тому задаються вагові коефіцієнти для кожного з них: функціональність: 25%, безпека: 20%, продуктивність: 20%, масштабованість: 15%, зручність використання: 10%, вартість: 10%.

Загальна оцінка обчислюється як зважене середнє значення: (0.25*4.33)+(0.20*4.33)+(0.20*4.33)+(0.15*4.67)+(0.10*3.33)+(0.10*3.33). Загальна оцінка становить 4.18. Отримана загальна оцінка є високою (вище 4), це свідчить про високу привабливість системи для впровадження. Нижчі значення вказують на необхідність доопрацювання певних аспектів або пошук альтернативних рішень. Метод експертних оцінок дозволяє отримати об'єктивну оцінку з точки зору різних спеціалістів, що підвищує точність ухвалення рішення щодо впровадження системи віртуального офісу.

КОПЗ-2024

7.3 Вибір методу оцінки вартості ПЗ

Для оцінки вартості програмної реалізації системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу найкраще підійдуть два описані на рисунку 7.3 методи.

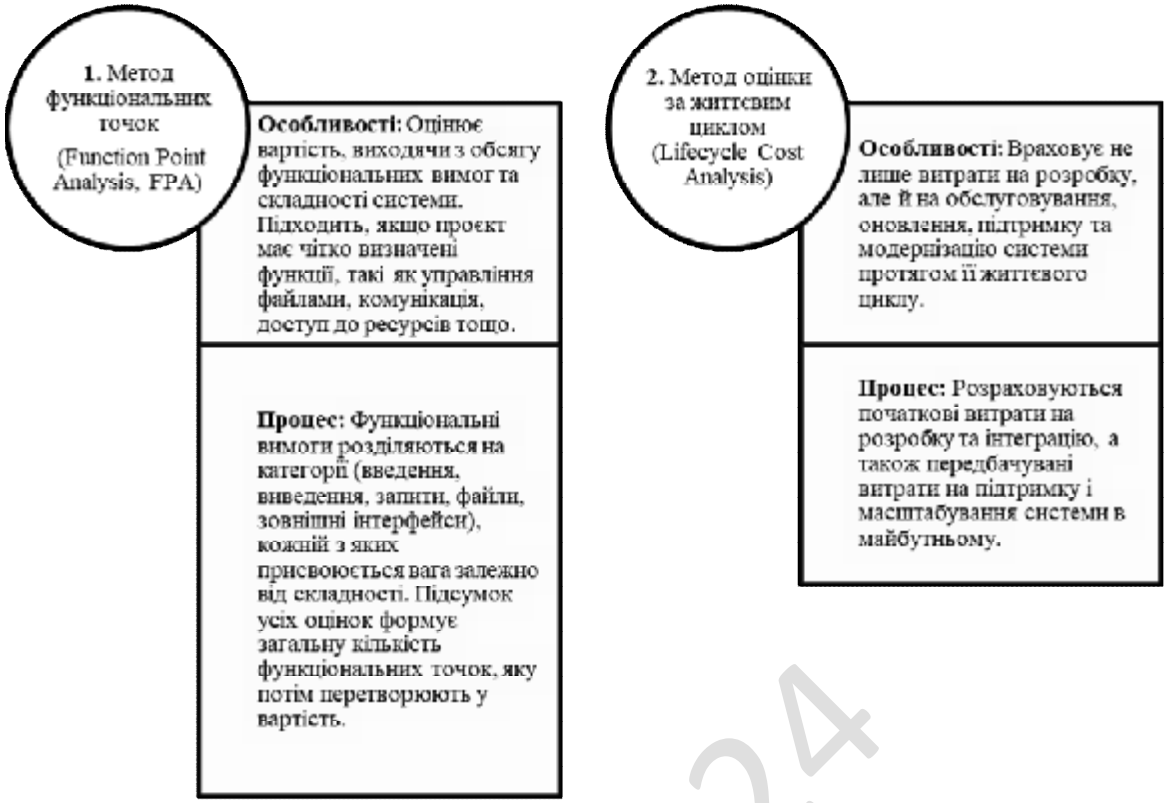

Рисунок 7.3 – Оптимальні методи оцінки вартості ПЗ

Оскільки проєкт включає комплексне архітектурне рішення, оптимальним може бути поєднання методу функціональних точок (FPA) для оцінки функціональності та методу життєвого циклу (Lifecycle Cost Analysis), щоб передбачити майбутні витрати на обслуговування та підтримку. Це дозволить отримати точнішу оцінку вартості впровадження системи та забезпечити її довгострокову економічну ефективність.

7.4 Розрахунок економічної ефективності від впровадження реалізованого ПЗ як фактору його привабливості

Оцінка економічної ефективності від впровадження системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу може базуватися на таких показниках, як підвищення продуктивності, зменшення витрат на обслуговування інфраструктури, скорочення часу на виконання операцій та інше.

КОПІЯ_2024

Орієнтовні розрахунки наведено в таблиці 7.2.

Таблиця 7.2 – Основні показники впровадження проєкту

Вхідні дані: Витрати на впровадження системи – $50,000. Зменшення витрат на підтримку інфраструктури – 20% річних. Поточні витрати на підтримку: $30,000 на рік. Зменшення витрат на комунікацію і поїздки – 15%. Поточні витрати на комунікацію і відрядження: $20,000 на рік. Підвищення продуктивності працівників – на 10%. Середня продуктивність на 1 працівника: $50,000 на рік. Кількість працівників, яких торкнеться нововведення: 20 осіб.
Економія на підтримці інфраструктури: 30,000×0.20=6,000
Економія на комунікаціях і поїздках: 20,000×0.15=3,000 доларів на рік
Підвищення продуктивності: 50,000×0.10=5,000 доларів додаткової продуктивності на одного працівника в рік
Загальна економія та додатковий дохід від впровадження: 6,000+3,000+100,000=109,000 доларів на рік
Період окупності: 50,000109,000≈0.46 року

Впровадження системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу забезпечить щорічну економію та додатковий дохід у розмірі $109,000, а період окупності складе близько 6 місяців. Це свідчить про високу економічну ефективність, оскільки вже за пів року інвестиції повністю окупляться, а підприємство почне отримувати чисту економію.

7.5 Пропозиція алгоритму просування проєкту розробки ПЗ

Алгоритм просування проєкту програмної реалізації системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу має виглядати наступним чином (рисунок 7.5).

КбПЗ-2024

Аналіз ринку та цільової аудиторії

- Дослідити ринок віртуальних офісних систем, визначити конкурентів і їхні сильні та слабкі сторони.
- Окреслити цільові групи: малі та середні компанії, віддалені команди, організації, що прагнуть оптимізувати витрати.
- Зрозуміти потреби користувачів у функціональності, продуктивності та безпеці.

Розробка позиціонування продукту

- Визначити ключові переваги системи, які вирізняють її серед інших: наприклад, підвищена продуктивність, автоматизація процесів, гнучкість масштабування.
- Сформулювати унікальну торговельну пропозицію (УТП), яка чітко передасть цінність для клієнта.

Планування маркетингової стратегії

- **Цифровий маркетинг:** просування через SEO, контекстну рекламу, соціальні мережі, таргетовану рекламу.
- **Контент-маркетинг:** розробка статей, кейсів, інфографік про переваги використання віртуального офісу.
- **Вебінари та онлайн-демонстрації:** організація презентацій для потенційних клієнтів для ознайомлення з функціональністю системи.

Створення інформативних матеріалів

- Розробити детальні презентації, відео-огляди, документи з технічними характеристиками, що розкривають можливості і вигоди системи.
- Підготувати White Papers, які описують, як саме система допомагає у вирішенні ключових проблем бізнесу.

Пошук стратегічних партнерів

- Встановити партнерські відносини з компаніями, що працюють у сфері IT-консалтингу та кібербезпеки.
- Залучити партнерів для спільного просування на ринок через розширені канали збуту.

Тестування та демонстраційний доступ

- Запропонувати потенційним клієнтам безкоштовний демонстраційний доступ на обмежений період для ознайомлення з функціоналом системи.
- Організувати спеціальні пропозиції та знижки для нових клієнтів за підписку на річне обслуговування.

Відгуки та кейс-стаді

- Після впровадження отримати відгуки та рекомендації від перших користувачів, що допоможе підвищити довіру до продукту.
- Опублікувати кейс-стаді з клієнтами, які досягли конкретних результатів завдяки використанню системи.

Аналіз результатів і оптимізація стратегії

- Збирати дані про ефективність каналів просування, зокрема показники конверсії та повернення інвестицій (ROI).
- За необхідності, коригувати маркетингову та продажну стратегію для покращення показників залучення клієнтів.

КПЗ_2024

Рисунок 7.5 – Етапи просування проєкту

1. Оптимізація каналів збуту

- **Цифрові канали**: Використовуйте онлайн-маркетинг для досягнення глобальної аудиторії. Оптимізуйте SEO на сайті проєкту, ведіть активний блог із контентом, що привертає професійну аудиторію, та запускайте контекстну рекламу. Також розгляньте соціальні мережі, зокрема LinkedIn та Facebook, для реклами та залучення цільової аудиторії.
- **Партнерські програми**: Співпрацюйте з IT-компаніями, постачальниками послуг хмарного зберігання даних та кібербезпеки для спільного продажу продукту. Такі партнери можуть додати цінності вашій пропозиції та розширити клієнтську базу.
- **Посередники та дистриб'ютори**: Використовуйте посередників у сфері інформаційних технологій, які вже мають базу клієнтів і контакти в галузі. Вони можуть допомогти швидко розширити ринок збуту та зменшити витрати на просування.
- **Інтеграція з іншими продуктами**: Створіть можливості інтеграції системи з популярними інструментами для управління проєктами, системами CRM, поштовими клієнтами та платформами для спільної роботи. Це підвищить цінність продукту та розширить ринок збуту.

2. Оптимізація шляхів реалізації

- **Створення SaaS-моделі**: Впровадження SaaS (Software as a Service) дозволяє швидко масштабувати продукт та залучити більше користувачів. Це забезпечить постійний дохід, оскільки клієнти зможуть підписатися на щомісячне або річне обслуговування.
- **Фокус на масштабованість та адаптивність**: Розробіть різні тарифні плани, що відповідають потребам різних клієнтів — від малих компаній до великих корпоративних клієнтів. Це дозволить обрати відповідний варіант та знизить бар'єр для нових користувачів.
- **Демонстраційний доступ**: Надання безкоштовного тестового періоду або демонстраційної версії дозволить клієнтам ознайомитися з функціоналом системи, що може значно підвищити конверсію.
- **Персоналізований підхід до клієнтів**: Для великих компаній забезпечте персоналізований підхід із гнучкими умовами обслуговування, виділеною технічною підтримкою та можливістю налаштувань під специфічні потреби бізнесу. Це сприяє зростанню лояльності великих клієнтів.
- **Мобільний доступ та мультиплатформенність**: Створіть можливість доступу до системи через мобільні додатки. Забезпечте сумісність з різними операційними системами (Windows, macOS, Linux), що підвищить зручність використання і розширить ринок.

3. Оптимізація продажів і підтримки

- **Автоматизація процесу продажу**: Використовуйте CRM-систему для управління лідами, автоматизації процесів продажу та підтримки клієнтів. Це допоможе систематизувати роботу з клієнтами та зменшити витрати на продажі.
- **Технічна підтримка**: Забезпечте якісну технічну підтримку, що швидко вирішуватиме запити користувачів, включаючи сервісні пакети із розширеною підтримкою для великих клієнтів.
- **Регулярне оновлення і розвиток функціоналу**: Відстежуйте зворотний зв'язок від клієнтів, регулярно оновлюйте продукт, додаючи нові функції та покращення, що відповідатимуть мінливим потребам ринку.

Рисунок 7.6 – Оптимізація каналів збуту та шляхів реалізації

Цей алгоритм передбачає поетапне планування і реалізацію стратегії просування, яка включає глибокий аналіз ринку, продумане позиціонування продукту, розвиток контент-маркетингу, залучення партнерів і постійне вдосконалення методів просування.

7.6 Оптимізація каналів збуту та шляхів реалізації ПЗ

Для оптимізації каналів збуту та шляхів реалізації проєкту програмної реалізації системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу варто процес розділити на три паралельних масштабних проєкти: оптимізацію каналів збуту, оптимізацію шляхів реалізації та оптимізацію продажів та підтримки (рисунок 7.6).

Запропоновані методи оптимізації каналів збуту та шляхів реалізації допоможуть досягти ширшої аудиторії, покращити користувацький досвід, створити стабільний дохід і збільшити конкурентоспроможність проєкту.

7.7 Визначення ключових факторів успіху конкретного проєкту

Для успішної реалізації проєкту програмної системи архітектурного рішення для інформаційних та комп'ютерних систем віртуального офісу важливими є фактори, приведені на рисунку 7.7.

Рисунок 7.7 – Ключові фактори успіху проєкту

Ці ключові фактори забезпечують стійкість, ефективність та конкурентні переваги проєкту, що сприяє його успіху на ринку і дозволяє залучити більше клієнтів.

ВОПЗ - 2024

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Вимога щодо впровадження заходів з охорони праці передбачається, зокрема, статтею 13 Закона України «Про охорону праці». У відповідності з цим законом, кожна компанія, в рамках якої реалізуються трудові відносини, зобов'язана вжити всіх необхідних заходів з охорони праці та розробити відповідні документи. Правильний підхід до організації охорони праці на виробництві дає працівникам почуття стабільності, захищеності їх прав та інтересів, уваги з боку керівництва. Налагоджена система охорони праці також знижує плинність кадрів, що позитивно впливає на стабільність роботи підприємств.

Аналізуючи умови працівників ІТ-сфери, на перший погляд, може здатися, що працівники сфери інформаційних технологій не схильні до ризиків на виробництві, та якщо більш глибоко розглянути умови і специфіку праці фахівців сфері ІТ-індустрії, можна виявити ряд факторів які будуть мати негативний сплив на стан охорони праці, так і на самого ІТ-фахівця. Сюди можна віднести як невідповідність освітлення, так і високий рівень шуму, що негативно позначатимуться як на емоційному так і на фізичному стані фахівця, призводитимуть до зниження ефективності праці та виробничих травм. Також, важливим моментом охорони праці ІТ-фахівця є врахування його психологічних можливостей (швидкість реакції, особливості пам'яті та уваги, емоційний стан, тощо).

Для того, щоб забезпечити ефективну роботу ІТ-фахівця, потрібно враховувати та максимально компенсувати такі негативні фактори як: надмірне нервово-емоційне навантаження, довготривалі статичні перевантаження, обмежена рухова активність. Всі ці чинники призводить до різноманітних

КПЗ-2024

відхилень у стані здоров'я, зокрема до перевтоми, зниження фізичної та розумової працездатності, неврозів, захворювань серцево-судинної системи тощо.

Метою даного розділу є огляд конкретних умов праці спеціаліста у сфері ІТ-індустрії. Завданнями для даного розділу є: аналіз умов праці на робочому місці фахівця ІТ-індустрії, розробка конкретних рекомендацій щодо покращення умов праці фахівців ІТ-індустрії, огляд пожежної безпеки на ІТ-підприємстві та розрахунок системи загального штучного освітлення виробничого приміщення де працюють ІТ – фахівці.

8.2 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	3
Довжина	4,6
Висота	3

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,9
Об'єм, V	$м^3$	не менше 20.0	20,7

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин).

КПЗ_2024

У зазначеному приміщенні працюють двоє людей. За даними, які наведено у табл. 8.1, та табл. 8.2, можна зробити висновок, що площа та об'єм приміщення у розрахунку на одно робоче місце програміста не відповідають нормативним вимогам ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [2], але відповідають нормативним вимогам Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»). Таким чином можна зробити висновок, що санітарно-гігієнічні умови праці на робочому місці програміста відповідають вимогам.

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, виконувана в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря в приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, так і розглянутого приміщення. У

КОПЗ-2024

приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	22-24	40-55	0,12
Тепла	23-25	50-70	0,1	24-25	50-65	0,9

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер Xerox WorkCentre 3025BI (3025VBI), електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

З 2019 року діють Державні будівельні норми україни "Природне і штучне освітлення" – ДБН В.2.5-28:2018 [1], у яких прописані вимоги до використання всіх освітлювальних приладів, у т.ч. світлодіодних.

Працю працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5-28:2018 [1], можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при

КОПІЯ 2024

фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 Лк. [1], Крім того все поле зору повинне бути освітлено достатньо рівномірно – ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.3 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значеннь;

Таким чином можна припустити, що основною причиною можнивого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов’язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання.

КПЗ – 2024

Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

Регулярна наочне знайомство персоналу із шляхами для евакуації людей із приміщення відповідно до плану евакуації, забезпечення розподільних щитів спеціальними розетками з заземлюючими контактами; організація заземлення всіх приладів і пристроїв, які працюють при напрузі вище 36 В.

Так як при ураженні електричним струмом у людини може статися фібриляція шлуночків серця, в організації бажано мати дефібрилятор і підготовлений персонал для роботи з ним.

8.4 Пожежна безпека

Вимоги до пожежної безпеки на підприємстві неухильно повинен дотримуватися кожен співробітник, а організаційна складова при цьому покладається на посадових осіб за відповідним рішенням керівництва і прописується в посадових інструкціях і положеннях по структурним підрозділам.

Зокрема, вказуються конкретні території, ділянки, зони, об’єкти, цілі будівлі і їх частини, поверхи, на яких відповідального співробітника повинне проводити такі організаційні роботи.

Відповідальні особи зобов’язуються розробити, впровадити та підтримувати в певному інструкцією і положенням на ввірених їм об’єктах протипожежний режим і інструкції відповідно до вимог, викладених в нормативних актах.

Передбачено також створення підрозділу добровільної пожежної охорони та пожежно-рятувальної команди в його складі.

Встановлений режим включає порядки з описом місць спеціального призначення та правила їх користування та утримання, наприклад:

– евакуаційних шляхів;

– так званих «курилок»;

КБПЗ–2024

– місць складування продукції та сировини;

– стоянки транспорту.

Також встановлюється порядок роботи та технічного обслуговування:

– вентиляційного устаткування;

– засобів пожежогасіння і захисту від загорянь;

– нагрівальних приладів;

– електрообладнання.

Розробляються і впроваджуються правила роботи з відкритим вогнем і горючими матеріалами. Створюються графіки проходження інструктажів з пожежної безпеки співробітників, а також порядок і терміни перевірок знань пожежно-технічного мінімуму, в тому числі, тих працівників, які відповідальні за цю ділянку роботи на підприємстві. При цьому можуть передбачатися внутрішні лекції, семінари, тренінги та практичні заняття на підприємстві, а також зовнішні – на базі спеціалізованих навчальних центрів з професійними викладачами.

Важливою складовою протипожежного режиму на будь-якому об’єкті є розробка і впровадження порядку дій при виникненні пожежі. Неодмінно має бути план евакуації, описано, як повинні відключатися електроустановки, що і в якій послідовності необхідно робити співробітникам.

Відповідно, для кожного об’єкта, кожного приміщення (крім коридорів, санвузлів, басейнів і подібних приміщень), окремих видів робіт складаються інструкції, за якими повинен працювати персонал, залучений на певних ділянках і в виконанні окремих видів робіт. За інструкціями проводиться навчання (інструктаж) персоналу з подальшим контролем знань.

Детально про те, як розробити протипожежний режим, прописати порядки та інструкції, пояснюють на тематичних курсах і семінарах [2].

КОПІЯ 2024

8.5 Розрахункова частина

Система освітлення робочого місця користувача ПК має відповідати наступним вимогам (рис. 8.1).

Рисунок 8.1 – Вимоги до системи освітлення робочого місця користувача ПК

Проведемо розрахунок штучного освітлення за методом коефіцієнта використання світлового потоку для приміщення ширина якого складає 6,4 м, довжина – 8,3 м, висота – 3,2 м. У зазначеному приміщенні працює 8 людей. Для того, щоб визначити потрібну кількість світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

КПІ-2024

$$F=E\cdot S\cdot K\cdot Z/n,$$

де: F – світловий потік, що розраховується, Лм;

E – нормована мінімальна освітленість, Лк; *E = 300 Лк;*

S – площа освітлюваного приміщення.

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку *K = 1,5);*

Z – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1.1... 1.2, в нашому випадку *Z = 1,1*);

n – коефіцієнт використання світлового потоку, (відношення світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{стін.}$) і стелі ($\rho_{стелі}$), значення коефіцієнтів дорівнюють:

$$\rho_{стін} = 50\% \text{ і } \rho_{стелі} = 50\%.$$

Обчислимо індекс приміщення за формулою:

$$i=S/(h\cdot (A+B)),$$

де: *S* – площа приміщення, *S* = 53,1 *м*^{2}*;*

h – розрахункова висота підвісу, *h* = 3,2 *м. (співпадає з в*исотою стелі, оскільки лампи освітлення закріплюються на стелі)*;*

A – ширина приміщення, *A = 6,4 м;*

B – довжина приміщення, *B* = 8,3 *м.*

Підставимо всі значення у формулу та визначемо індекса приміщення: i=1,1.

Знаючи індекс приміщення, за знаходимо *n = 0,46 (з табличних даних коефіцієнтів використання світлового потоку (n) світильників з відповідним типом лампам) [8].* Підставимо всі значення у формулу, визначемо світловий потік: F=57161,7 Лм.

КПЗ-2024

Для розрахунку дудемо використовувати стельові *світлодіодні панелі* Призма-72 6400К, світловий потік яких $F_л$ = 7200 *Лм.*

Число ламп визначається по формулі:

$$N=F/F_л$$

де: F – світловий потік,

$F_л$ – світловий потік однієї лампи.

Підставимо всі значення у формулу та визначемо індекса приміщення:

$$N= 57161,7/ 7200=7,9 \text{ шт.}$$

Приймаємо необхідну кількість *світлодіодних світильників* 8 шт.

Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва вцілому.

З цих міркувань було здійснено аналіз умов праці, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров'я програміста. Розроблено заходи з охорони праці.

КбПЗ-2024

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

– Досліджена система архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

– На основі отриманих результатів досліджень створена програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

КПЗ-2024

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Visual C++. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows 10/11.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм Madryga.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

Проведено маркетингове та економічне обґрунтування ІТ-проєкту, що дозволило визначити ключові фактори успіху даного проекту.

КБПЗ_2024

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Лагута І.А. Дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу // Збірник праць молодих науковців ЦНТУ. – Вип. 14. – Кропивницький: ЦНТУ, 2024.

2. Ramon Nastase «Computer Networking: The Beginner's guide for Mastering Computer Networking, the Internet and the OSI Model». 2018. – 186 p.

3. Russ White & Ethan Banks «Computer Networking Problems and Solutions: An Innovative Approach to Building Resilient, Modern Networks». 2017. – 832 p.

4. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». *CEUR Workshop Proceedings*, 2023, 3628, pp. 106-115.

5. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». *Advanced Information Systems*, 2023, 7(2), pp. 49-56.

6. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». *CEUR Workshop Proceedings,* Volume 3530, 2023, pp. 256-265.

7. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». *Lecture Notes on Data Engineering and Communications Technologies*, 2023, 178, pp. 208–223.

8. Smirnova, T., Gnatyuk, S., Yudin, O., Sydorenko, V., Polozhentsev, A., «The Model for Calculating the Quantitative Criteria for Assessing the Security Level of Information and Telecommunication Systems». CEUR Workshop Proceedings Volume 3156, 2022, Pages 390-399.

КОПЗ_2024

9. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». *Communications in Computer and Information Science*, 2021, vol 1486. Springer, Cham. pp 169-184.

10. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». *CEUR Workshop Proceedings*. Volume 2740, 2020, Pages 102-114.

11. Smirnov O., Alimseitova Zh., Adranova A., Akhmetov B., Lakhno V., Zhilkishbayeva G. «Models and algorithms for ensuring functional stability and cybersecurity of virtual cloud resources». *Journal of theoretical and applied information technology* Vol.98. No 21, 2020, P. 3334-3346.

12. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». *2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Ukraine, Kyiv, May 14-18. 2020. P. 172-177.

13. Smirnov O., Kuznetsov A., Pushkar'ov A., Serhiienko R., Babenko V., Kuznetsova T., «Representation of Cascade Codes in the Frequency Domain». In: Radivilova T., Ageyev D., Kryvinska N. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 48. Springer, Cham. 2021. pp 557-587.

14. Smirnov, O., Markovets, O. Vovk, N., Turchyn, Y., «Model of informational support for social network administrators' content creation». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 125-136.

15. Smirnov, O., Drieieva, H., Drieiev, O., Polishchuk, Y., Brzhanov, R., Aleksander, M. «Method of fractal traffic generation by a model of generator on the graph». *CEUR Workshop Proceedings* Volume 2616, 2020, Pages 366-379.

16. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated

КОПЗ-2024

with the Markov chains», *CEUR Workshop Proceedings* Volume 2608, 2020, Pages 633-645.

17. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». *International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019*; Odessa; Ukraine; 9-13 September 2019. P.22-28.

18. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.

19. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.

20. Smirnov, O., Odarchenko, R., Abakumova, A., Usik, P., Kundyz, M., «QoE optimization technique for media delivery in 5G networks». *2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T),* Kyiv, Ukraine, 8 – 11 October 2019. P.597-601.

21. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». *CEUR Workshop Proceedings*, Vol 2588, P. 90-106, 2019.

22. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», *2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019*/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.

23. Smirnov, O., Kuznetsov, A., Kiian, A., Zamula, A., Rudenko, S., Hryhorenko, V., «Variance Analysis of Networks Traffic for Intrusion Detection in

КОПЗ – 2024

Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 353-358.

24. Smirnov, O., Kuznetsov, A., Kavun, S., Babenko, B., Nakisko, O., Kuznetsova, K., «Malware Correlation Monitoring in Computer Networks of Promising Smart Grids», *2019 IEEE 6th International Conference On Energy Smart Systems (2019 IEEE ESS)*, Kyiv, Ukraine April 17-19, 2019 P. 347-352.

25. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Pastukhov, M., Kuznetsova, K., Prokopovych-Tkachenko, D., «Discrete Signals with Special Correlation Properties», *CEUR Workshop Proceedings* Volume 2353, *CEUR Workshop Proceedings* 2019, Pages 618-629.

26. Smirnov A.A., Kuzne tsov A.A., Danilenko D.A., Berezovsky A., «The statistical analysis of a network traffic for the intrusion detection and prevention systems», *Telecommunications and Radio Engineering*. – Volume 74, Issue 1. – Begel House Inc. – 2015. – P. 61-78.

27. Батрак О., Смірнова Т., Гнатюк В., Одарченко Р., Смірнов О. «Дослідження показників ефективності функціонування та перспектив розвитку систем IP-телефонії». *Підводні технології,* 2024, № 13, с. 28-35.

28. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». *Сучасні інформаційні системи,* 2023, том 7, № 2, С. 49-56.

29. Смірнова Т.В., Гнатюк С.О., Сидоренко В.М., Юдін О.Ю., Сидоренко С.Ю., «Модель визначення критичності галузевих інформаційно-телекомунікаційних систем». *Проблеми інформатизації та управління,* № 2(70). 2022. С. 28-37.

30. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Смірнов С.А., Поліщук Л.І., «Дослідження стійкості до диференціального криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного

КБПЗ-2024

захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку,* 2022, № 3(69). С. 93-98.

31. Смірнов О.А., Смірнова Т.В., Якименко Н.М., Поліщук Л.І., Смірнов С.А. «Дослідження статистичної стійкості та швидкісних характеристик запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Вісник Хмельницького національного університету. Серія: «Технічні науки»,* № 2 (307). С. 46-52. 2022.

32. Смірнов О.А., Смірнова Т.В., Константинова Л.В., Смірнов С.А., Якименко Н.М., «Дослідження стійкості до лінійного криптоаналізу запропонованої функції гешування удосконаленого модуля криптографічного захисту в інформаційно-комунікаційних системах» *Системи управління, навігації та зв'язку*, 2022, № 1(67). С. 84-89.

33. Смірнов О.А., Смірнова Т.В., Буравченко К.О., Кравченко С.С., Горбов В.О., «Хмарна система підтримки прийняття рішень технологічного процесу відновлення поверхонь конструкцій і деталей машин». *Сучасні інформаційні системи*. 2021. Т. 5, № 4. С. 79-95

34. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» *Вісник Черкаського державного технологічного університету. Технічні науки.* №4. С. 103-110. 2020.

35. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», *Кібербезпека: освіта, наука, техніка*. № 3(7). С. 43-62. 2020.

36. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В., Поліщук Л.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2020. – 294 с.

КОПЗ_2024

37. О.А. Смірнов, П.С. Усік, «Дослідження перспектив використання технологічних рішень в мережах 5G» у *Кібербезпека та інформаційні технології: монографія.* – Х. : ТОВ «ДІСА ПЛЮС», 2020.С. 122-135.

38. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». *Центральноукраїнський науковий вісник. Технічні науки.* № 2(33). с. 161-172, 2019.

39. Смірнов О.А., Конопліцька-Слободенюк О.К., Смірнов С.А., Буравченко К.О., Смірнова Т.В. Поліщук Л.І. Проектування комп'ютерних систем та мереж. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2019. – 264 с.

40. Smirnov, O., Kuznetsov, A., Kuznetsova., K. Synthesis of Discrete Signals with Improved Correlation Properties. Монографія: In.: ISCI'2019: Information Security in Critical Infrastructures. Collective monograph. Edited by Ivan D. Gorbenko and Alexandr A. Kuznetsov, ASC Academic Publishing, USA, 2019, pp. 281-299. – ISBN: 978-0-9989826-8-7 (Hardback), ISBN: 978-0-9989826-9-4 (Ebook).

41. Смірнов О.А., Дрєєва Г.М. Метод генерування фрактального трафіку за допомогою моделі генератора на графі. Монографія: Інформаційна безпека та інформаційні технології : монографія / за заг. ред. В. С. Пономаренка. – Х. : Вид. Рожко С.Г. 2019. С. 123-139

42. Дрєєва Г.М., Смірнов О.А., Дрєєв О.М. Метод генерування фрактальноподібної числової послідовності на основі скінченного автомату для моделювання трафіку у мережі. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 173-183, 2019.

43. Смірнова Т.В., Солових Є.К., Смірнов О.А., Дрєєв О.М. Побудова хмарних інформаційних технологій оптимізації технологічного процесу відновлення та зміцнення поверхонь деталей. Центральноукраїнський науковий вісник. Технічні науки. № 1(32). с. 184-194, 2019.

КОПЗ-2024

44. Смірнов О.А., Смірнов С.А., Поліщук Л.І., Смірнова Т.В., Конопліцька-Слободенюк О.К. Метод формування антивірусного захисту даних з використанням безпечної маршрутизації метаданих. Кібербезпека: освіта, наука, техніка. – Том 3 № 3. – Київ: КУ ім. Бориса Грінченка. – 2019. – С. 63-87.

45. Смірнов О.А., Гнатюк С.О., Кавун С.В., Терейковський І.А., Жмурко Т.О., Смірнов С.А., Коваленко А.С. Основи безпеки в комп'ютерних мережах. Навчальний посібник – Кропивницький: вид. Лисенко В.Ф. 2018. – 177 с.

46. Смірнов О.А., Котелянець В.В. Стійкі до колізій стохастичні моделі функціонування безпроводових сенсорних мереж. Вісник інженерної академії України, №3, с. 145-152, 2018

47. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Алгоритми формування безлічі маршрутів передачі метаданих у антивірусні хмарні системи. Збірник наукових праць "Системи обробки інформації". – Випуск 5 (142). – Х.: ХУПС – 2016. – С. 148-152.

48. Смірнов О.А., Смірнов С.А. Дідик А.К., Дрєєв О.М. Моделі системи нейромережевих експертів безпечної маршрутизації у хмарних антивірусних системах. Збірник наукових праць "Системи обробки інформації". – Випуск 3 (140). – Х.: ХУПС – 2016. – С. 36-39.

49. Смірнов О.А., Смірнов С.А., Дідик А.К., Дрєєв А.М. Спосіб контролю ліній зв'язку телекомунікаційної системи антивірусу. Збірник наукових праць Харківського університету Повітряних Сил. Випуск 2 (47). – Харків: ХУПС. – 2016. – С. 121-127.

50. Смірнов О.А., Смірнов С.А., Дідик А.К. Метод безпечної маршрутизації метаданих у хмарні антивірусні системи. Системи озброєння та військова техніка. – Випуск 2 (46) – Х.: ХУПС – 2016. – С. 146-149.

51. Смірнов О.А., Кавун С.В., Доренський О.П., Вялкова В.І. Інформаційна безпека в комп'ютерних мережах. Навчальний посібник – Кіровоград: РВЛ КНТУ, 2016. – 151 с.

КОПЗ – 2024

Додаток А
(обов’язковий)

Технічне завдання

Зміст

КПЗ–2024

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 18-13 від 07.08.2024 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

КБПЗ_2024

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи архітектурного рішення інформаційних та комп'ютерних систем віртуального офісу;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

КОПЗ_2024

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ, працювати в ОС Windows 10/11 і з сумісними з цією платформою пристроями і прикладним програмним забезпечення.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows 10/11.

5.8.1 Обладнання

Комп’ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14” 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Visual C++.

КбПЗ – 2024

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати маркетингове та економічне обґрунтування ІТ-проєкту з урахуванням цін на 3 вересня 2024 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинна бути розглянута пожежна безпека.

КбПЗ_2024

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 94 аркуша.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

10.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Маркетингове та економічне обґрунтування ІТ-проєкту.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 02.12.2024 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 16.12.2024 р.

КПЗ – 2024

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

_____ Буравченко К.О.

КбПЗ_2024

Дослідження та програмна реалізація системи архітектурного рішення інформаційних та комп’ютерних систем віртуального офісу

Лістинг програми

Код документу 12

Носій: CD/DVD-диск / USB-флеш-накопичувач

Загальна кількість аркушів: 71

Літера: РП

Кропивницький – 2024 року

Файл Worker.cpp - вікно адміністратора

```
  /*****************************************************************************/
#include "StdAfx.h"
#include "Worker.h"
#include "Doc.h"
#include "Logger.h"
/*****************************************************************************/
using namespace System;
using namespace System::IO;
using namespace System::Collections;
/*****************************************************************************/
#define DOCS_FILE "DATA.LMB"
#define LOG_INPUT "INPUT.LML"
#define LOG_OUTPUT      "OUTPUT.LML"
/*****************************************************************************/
// NAME:          CWorker
// DESCRIPTION:   Конструктор класу
// INPUT:         N/D
/*****************************************************************************/
CWorker::CWorker(void)
{
  listDoc = gcnew ArrayList();
  logInput = gcnew CLogger();
  logOutput = gcnew CLogger();
}
/*****************************************************************************/
// NAME:          GetIndexByISBNHash
// DESCRIPTION:   Функція одержання індексу документа в загальному списку по
вхідному номеру документа
// INPUT:         ddHashValue - значення хеша вхідного номера документа
/*****************************************************************************/
Int32 CWorker::GetIndexByISBNHash(Int32 HashValue)
{
  // Цикл пошуку документа із заданим вхідним номером документа
  for(Int32 i = 0; i < listDoc->Count; i++)
  {
      // Хеш-значення вхідного номера документа заданого документа збігається з
      // хеш-     значенням вхідного номера документа знайденого документа?
      if(HashValue == ((CDoc^)listDoc[i])->GetISBNHash())
      {
            // Документ знайдений, повернути індекс
            return i;
      }
  }

  // Документ не знайдений, повернути -1
  return -1;
}
/*****************************************************************************/
// NAME:          AddDoc
// DESCRIPTION:   Функція додавання нового документа
// INPUT:         SourceDoc - об'єкт "документ" для включення в список
// OUTPUT:        TRUE - документ успішно доданий
//                FALSE - такий документ уже існує в системі
/*****************************************************************************/
Boolean CWorker::AddDoc(CDoc^ SourceDoc)
{
  // Документи з таким вхідним номером документа не існує?
  if(this->GetIndexByISBNHash(SourceDoc->GetISBNHash()) == -1)
  {
      // Додати заданній документ у загальний список
      listDoc->Add(SourceDoc);
      // Записати подія в лог
      logInput->WriteEvent(
            "Новий документ '" + ((CDoc^)SourceDoc)->GetName() +
            "', Автор '" + ((CDoc^)SourceDoc)->GetAuthor() +
```

КоТЭ-2024

```
            "', Вхідний номер документа " + ((CDoc^)SourceDoc)->GetISBN());
      // Функція відробила успішно
      return true;
  }
  else
  {
      // Документ із таким же вхідним номером документа існує, повернути помилку
      return false;
  }
}
/*****************************************************************************/
// NAME:          RemoveDoc
// DESCRIPTION:   Функція видалення документів із системи
// INPUT:         ddISBNHash - значення хеша вхідного номера документа
// OUTPUT:              0 - видалення зроблене
//                      1 - видалення неможливо, не всі екземпляри перебувають у
системі
//                      2 - документ із заданим вхідним номером документа не
знайдений
/*****************************************************************************/
Int32 CWorker::RemoveDoc(Int32 ddISBNHash)
{
  // Знайти документ по вхідному номеру документа
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);

  // Документ із таким вхідним номером документа існує?
  if(ddIndex != -1)
  {
      // Перевірити, що жодного документа немає в користувача
      if(((CDoc^)listDoc[ddIndex])->GetTotalNumber() ==
            ((CDoc^)listDoc[ddIndex])->GetFreeNumber())
      {

            // Записати подія в лог
            logOutput->WriteEvent(
                  "Виконаний документ '" + ((CDoc^)listDoc[ddIndex])->GetName()
+
                  "', Автор '" + ((CDoc^)listDoc[ddIndex])->GetAuthor() +
                  "', Вхідний номер документа " + ((CDoc^)listDoc[ddIndex])-
>GetISBN());
            // Видалити знайдений документ зі списку
            listDoc->RemoveAt(ddIndex);
            // Функція відробила успішно
            return 0;
      }
      else
      {
            // Повернути помилку, не всі документи перебувають у системі
            return 1;
      }
  }
  else
  {
      // Повернути помилку, документ із таким вхідним номером документа не був
знайдений
      return 2;
  }
}
/*****************************************************************************/
// NAME:          GiveDoc
// DESCRIPTION:   Функція видачі документів користувачеві
// INPUT:         listDoc - список документів користувача (для виконання
додавання)
//                      ddISBNHash - хеш вхідного номера документа, що потрібно
одержати
// OUTPUT:        TRUE - операція пройшла успішно
//                      FALSE - у наявності немає жодного екземпляра заданого
документа
/*****************************************************************************/
```

КПЗ 2024

```
Boolean CWorker::GiveDoc(ArrayList^ listReader, Int32 ddISBNHash)
{
  // Визначити індекс по вхідному номеру документа
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);

  // Зменшити кількість вільних документів
  if((ddIndex != -1) && ((CDoc^)listDoc[ddIndex])->DecFreeNumber())
  {
      // Додати документ у список користувача
      listReader->Add(listDoc[ddIndex]);
      // Функція відробила успішно
      return true;
  }
  else
  {
      // Такого документа не є в наявності
      return false;
  }
}
/*************************************************************************/
// NAME:          TakeDoc
// DESCRIPTION:   Функція прийняття документів від користувача назад у систему
// INPUT:         // listDoc - список документів користувача (для виконання
видалення)
//                      ddISBNHash - хеш вхідного номера документа, що потрібно
повернути
/*************************************************************************/
void CWorker::TakeDoc(ArrayList^ listReader, Int32 ddISBNHash)
{
  // Визначити індекс по вхідному номеру документа (документ існує, тому що його
вхідний номер документа був повідомлений користувачеві)
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);

  // Видалити документ зі списку користувача
  // listReader->RemoveAt(ddIndex);

  // Збільшити кількість вільних документів
  ((CDoc^)listDoc[ddIndex])->IncFreeNumber();
}
/*************************************************************************/
// NAME:          LoadDocList
// DESCRIPTION:   Функція завантаження документів системи з файлу
// INPUT:         N/D
/*************************************************************************/
void CWorker::LoadDocList()
{
  // Перевірити існування файлу
  if(!File::Exists(DOCS_FILE))
  {
      // Файл не знайдений, закінчити виконання функції
      return;
  }

  // Відкрити файл
  StreamReader^ srDoc = gcnew StreamReader(DOCS_FILE);

  // Продовжувати, поки не буде знайдений кінець файлу
  while(srDoc->EndOfStream == false)
  {
      // Створити новий об'єкт "документ"
      CDoc^ NewDoc = gcnew CDoc();

      // Завантажити з файлу й установити параметри документа
      NewDoc->SetName(srDoc->ReadLine());
      NewDoc->SetAuthor(srDoc->ReadLine());
      NewDoc->SetISBN(srDoc->ReadLine());
      NewDoc->SetTheme(srDoc->ReadLine());
      NewDoc->SetPages(Convert::ToInt32(srDoc->ReadLine()));
      NewDoc->SetTotalNumber(Convert::ToInt32(srDoc->ReadLine()));
```

КОПІЮ 2024

```
      NewDoc->SetFreeNumber(Convert::ToInt32(srDoc->ReadLine()));

      // Включити документ у загальний список документів
      listDoc->Add(NewDoc);
  }

  // Закрити файл
  srDoc->Close();
}
/*************************************************************************/
// NAME:          SaveDocList
// DESCRIPTION:   Функція збереження документів системи у файл
// INPUT:         N/D
/*************************************************************************/
void CWorker::SaveDocList()
{
  // Перевірити існування файлу
  if(File::Exists(DOCS_FILE))
  {
      // Видалити файл
      File::Delete(DOCS_FILE);
  }

  // Створити й відкрити файл
  StreamWriter^ swDoc = gcnew StreamWriter(DOCS_FILE);

  // Цикл запису по одному документу
  for(Int32 i = 0; i < listDoc->Count; i++)
  {
      // Одержати параметри документа й записати їх у файл
      swDoc->WriteLine(((CDoc^)listDoc[i])->GetName());
      swDoc->WriteLine(((CDoc^)listDoc[i])->GetAuthor());
      swDoc->WriteLine(((CDoc^)listDoc[i])->GetISBN());
      swDoc->WriteLine(((CDoc^)listDoc[i])->GetTheme());
      swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])->GetPages()));
      swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])-
>GetTotalNumber()));
      swDoc->WriteLine(Convert::ToString(((CDoc^)listDoc[i])->GetFreeNumber()));
  }

  // Закрити файл
  swDoc->Close();
}
/*************************************************************************/
// NAME:          FindDoc
// DESCRIPTION:   Функція пошуку документа по заданих параметрах
// INPUT:         strFindDoc - рядок для пошуку
/*************************************************************************/
ArrayList^ CWorker::FindDoc(String^ strFindValue)
{
  // Список для результату
  ArrayList^ listResult = gcnew ArrayList();

  // Шукане значення без обліку регістра
  String^ strValue = strFindValue->ToLower();

  // Задана порожній рядок?
  if(String::IsNullOrEmpty(strValue->Trim()))
  {
      //Вивести весь список
      for(Int32 i = 0; i < listDoc->Count; i++)
      {
            // Додати документ у результуючий список
            listResult->Add(listDoc[i]);
      }
  }
  else
  {
      // Вибрати з умовою
```

КПЗ_2024

```
      for(Int32 i = 0; i < listDoc->Count; i++)
      {
            // Пошук рядка в кожному полі без обліку регістра
            if(((((CDoc^)listDoc[i])->GetName())->ToLower())->IndexOf(strValue)
!= -1 ||
               ((((CDoc^)listDoc[i])->GetAuthor())->ToLower())-
>IndexOf(strValue) != -1 ||
               ((((CDoc^)listDoc[i])->GetISBN())->ToLower())->IndexOf(strValue)
!= -1 ||
               ((((CDoc^)listDoc[i])->GetTheme())->ToLower())->IndexOf(strValue)
!= -1 ||
              Convert::ToString((((CDoc^)listDoc[i])->GetPages()))-
>IndexOf(strValue) != -1)
            {
                  // Додати документ у результуючий список
                  listResult->Add(listDoc[i]);
            }
      }
  }

  // Повернути список збігів
  return listResult;
}
/*************************************************************************/
// NAME:          ViewDoc
// DESCRIPTION:   Функція перегляду інформації про заданий документ
// INPUT:         ddISBNHash - хеш вхідного номера документа, інформацію про яку
треба переглянути
/*************************************************************************/
CDoc^ CWorker::ViewDoc(Int32 ddISBNHash)
{
  // Визначити індекс по вхідному номеру документа (документ існує, тому що його
вхідний номер документа був повідомлений користувачеві)
  Int32 ddIndex = this->GetIndexByISBNHash(ddISBNHash);

  // Повернути елемент "документ" із загальне списку
  return (CDoc^)listDoc[ddIndex];
}
/*************************************************************************/
// NAME:          ReadLogs
// DESCRIPTION:   Функція зчитування історії з файлу
// INPUT:         N/D
/*************************************************************************/
void CWorker::ReadLogs()
{
  // Файл із історією надходжень існує?
  if(File::Exists(LOG_INPUT))
  {
      // Зчитування історії надходжень
      logInput->strLog = File::ReadAllText(LOG_INPUT);
  }
  else
  {
      logInput->strLog = "";
  }

  // Файл із історією списань існує?
  if(File::Exists(LOG_INPUT))
  {
      // Зчитування історії списань
      logOutput->strLog = File::ReadAllText(LOG_OUTPUT);
  }
  else
  {
      logOutput->strLog = "";
  }
}
/*************************************************************************/
// NAME:          WriteLogs
```

КОПЗ – 2024

```
// DESCRIPTION:   Функція збереження історії у файл
// INPUT:         N/D
/*****************************************************************************/
void CWorker::WriteLogs()
{
  // Запис історії
  File::WriteAllText(LOG_INPUT, logInput->strLog);
  File::WriteAllText(LOG_OUTPUT, logOutput->strLog);
}
/*****************************************************************************/
// NAME:          ClearLogs
// DESCRIPTION:   Функція очищення історії
// INPUT:         N/D
/*****************************************************************************/
void CWorker::ClearLogs()
{
  logInput->strLog = "";
  logOutput->strLog = "";
}
/*****************************************************************************/
// NAME:          ViewInputLog
// DESCRIPTION:   Функція перегляду історії надходжень
// INPUT:         N/D
/*****************************************************************************/
String^ CWorker::ViewInputLog()
{
  return logInput->strLog;
}
/*****************************************************************************/
// NAME:          ViewOutputLog
// DESCRIPTION:   Функція перегляду історії списань
// INPUT:         N/D
/*****************************************************************************/
String^ CWorker::ViewOutputLog()
{
  return logOutput->strLog;
}
/*****************************************************************************/
```

КбПЗ_2024

Файл Worker.resx - xml опис вікна адміністратора

```
<?xml version="1.0" encoding=" utf-8"?>
<root>
  <xsd:schema id="root" xmlns="" xmlns:xsd=" " xmlns:msdata="urn: schemas-
microsoft-com: xml-msdata">
    <xsd:import namespace=" " />
    <xsd:element name="root" msdata:IsDataSet="true">
      <xsd:complexType>
        <xsd:choice maxOccurs="unbounded">
          <xsd:element name="metadata">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0" />
              </xsd:sequence>
              <xsd:attribute name="name" use="required" type="xsd:string" />
              <xsd:attribute name="type" type="xsd:string" />
              <xsd:attribute name="mimetype" type="xsd:string" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="assembly">
            <xsd:complexType>
              <xsd:attribute name="alias" type="xsd:string" />
              <xsd:attribute name="name" type="xsd:string" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="data">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
                <xsd:element name="comment" type="xsd:string" minOccurs="0"
msdata:Ordinal="2" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required"
msdata:Ordinal="1" />
              <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
              <xsd:attribute name="mimetype" type="xsd:string"
msdata:Ordinal="4" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="resheader">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required" />
            </xsd:complexType>
          </xsd:element>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
  <resheader name="resmimetype">
    <value>text/ microsoft-resx</value>
  </resheader>
  <resheader name="version">
    <value>2.0</value>
  </resheader>
  <resheader name="reader">
    <value>System.Resources.ResXResourceReader, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <resheader name="writer">
```

КоПЗ_2024

```
    <value>System.Resources.ResXResourceWriter, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <metadata name="menuStrip.TrayLocation" type="System.Drawing.Point,
System.Drawing, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a">
    <value>230, 17</value>
  </metadata>
  <metadata name="statusStrip.TrayLocation" type="System.Drawing.Point,
System.Drawing, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a">
    <value>332, 17</value>
  </metadata>
  <metadata name="groupInfo.Locked" type="System.Boolean, mscorlib,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089">
    <value>True</value>
  </metadata>
  <metadata name="groupInfo.Locked" type="System.Boolean, mscorlib,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089">
    <value>True</value>
  </metadata>
</root>
```

КбПЗ_2024

Файл Worker.h - бібліотека для файлу frmAbout.cpp

```
    #pragma once
/*************************************************************************/
#include "Worker.h"
#include "Doc.h"
#include "Counter.h"
#include "frmAbout.h"
/*************************************************************************/
#define PAGES_LESS                          "менше"
#define PAGES_GREATER                       "більше"
#define PAGES_EQUAL                         "дорівнює"
/*************************************************************************/
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;
/*************************************************************************/
namespace UsrsManager
{
    public ref class frmWorker : public System::Windows::Forms::Form
    {
/*************************************************************************/
    private:
      CWorker^ Worker;              // Об'єкт "Адміністратор"

      Boolean IsCreateClicked;      // Прапор натискання по кнопці "Створити"
      Boolean IsEditClicked;        // Прапор натискання по кнопці "Редагувати"

      ArrayList^ listView;          // Поточний список документів для
відображення

      Int32 ddCurrentIndex;         // Індекс поточного відображуваного елемента
/*************************************************************************/
private: System::Windows::Forms::Button^  btnClearLogs;
/*************************************************************************/
public:     frmWorker(void)
      {
            // Ініціалізація компонентів форми
            InitializeComponent();

            // Кнопки "Створити" і "Редагувати" не минулого натиснуті жодного
разу
            IsCreateClicked = false;
            IsEditClicked = false;

            // Створити список відображуваних документів
            listView = gcnew ArrayList();

            // Поточний індекс документа для відображення не визначений
            ddCurrentIndex = -1;
      }
/*************************************************************************/
protected: ~frmWorker()
      {
            if (components)
            {
                  delete components;
            }
      }
/*************************************************************************/
    private: System::Windows::Forms::ListBox^  lstPagesDocList;
    private: System::Windows::Forms::NumericUpDown^  nmrPagesNumber;
    private: System::Windows::Forms::ComboBox^  cmbPagesDirect;
    private: System::Windows::Forms::ListBox^  lstThemeDocList;
```

КбТЗ - 2024

```
    private: System::Windows::Forms::ListBox^  lstAuthorDocList;
    private: System::Windows::Forms::CheckBox^  chkPagesFlag;
    private: System::Windows::Forms::MenuStrip^  menuStrip;
    private: System::Windows::Forms::ToolStripMenuItem^  menuFile;
    private: System::Windows::Forms::ToolStripMenuItem^  menuFileSave;
    private: System::Windows::Forms::ToolStripMenuItem^  menuFileExit;
    private: System::Windows::Forms::ToolStripMenuItem^  menuHelp;

    private: System::Windows::Forms::ToolStripMenuItem^  menuHelpAbout;
    private: System::Windows::Forms::GroupBox^  groupBox2;
    private: System::Windows::Forms::TextBox^  txtThemeFilter;
    private: System::Windows::Forms::TextBox^  txtAuthorFilter;
    private: System::Windows::Forms::Label^  lblFreeInstanceNumber;
    private: System::Windows::Forms::Label^  lblTotalInstanceNumber;
    private: System::Windows::Forms::CheckBox^  chkThemeDocFlag;
    private: System::Windows::Forms::CheckBox^  chkAuthorDocFlag;
    private: System::Windows::Forms::CheckBox^  chkFreeInstanceFlag;
    private: System::Windows::Forms::CheckBox^  chkTotalInstaceFlag;
    private: System::Windows::Forms::NumericUpDown^  nmrFreeNumber;
    private: System::Windows::Forms::Button^  btnRefresh;
    private: System::Windows::Forms::NumericUpDown^  nmrTotalNumber;
    private: System::Windows::Forms::TabPage^  tabHistory;
    private: System::Windows::Forms::GroupBox^  groupOutput;
    private: System::Windows::Forms::TextBox^  txtOutputHistory;
    private: System::Windows::Forms::GroupBox^  groupInput;
    private: System::Windows::Forms::TextBox^  txtInputHistory;
    private: System::Windows::Forms::NumericUpDown^  nmrPageNumber;
    private: System::Windows::Forms::TextBox^  txtTheme;
    private: System::Windows::Forms::TextBox^  txtISBN;
    private: System::Windows::Forms::TextBox^  txtAuthor;
    private: System::Windows::Forms::TextBox^  txtDocName;
    private: System::Windows::Forms::TabPage^  tabReports;
    private: System::Windows::Forms::Label^  lblFreeNumber;
    private: System::Windows::Forms::Label^  lblTotalNumber;
    private: System::Windows::Forms::Label^  lblPagesNumber;
    private: System::Windows::Forms::Label^  lblTheme;
    private: System::Windows::Forms::StatusStrip^  statusStrip;
    private: System::Windows::Forms::ToolStripStatusLabel^  lblStatus;
    private: System::Windows::Forms::TextBox^  txtFindString;
    private: System::Windows::Forms::TabControl^  tabControl;
    private: System::Windows::Forms::TabPage^  tabDocs;
    private: System::Windows::Forms::Button^  btnEdit;
    private: System::Windows::Forms::Button^  btnDelete;
    private: System::Windows::Forms::Button^  btnCreate;
    private: System::Windows::Forms::GroupBox^  groupInfo;
    private: System::Windows::Forms::Label^  lblISBN;
    private: System::Windows::Forms::Label^  lblAuthor;
    private: System::Windows::Forms::Label^  lblDocName;
    private: System::Windows::Forms::GroupBox^  groupFinding;
    private: System::Windows::Forms::Button^  btnFind;
    private: System::Windows::Forms::Button^  btnFirst;
    private: System::Windows::Forms::Button^  btnPrev;
    private: System::Windows::Forms::Button^  btnNext;
    private: System::Windows::Forms::Button^  btnLast;
    private: System::Windows::Forms::TextBox^  txtPosition;
    private: System::ComponentModel::Container ^components;
/*********************************************************************/
#pragma region Windows Form Designer generated code
      /// <summary>
      /// Необхідний метод для підтримки Розроблювача - не модифікувати
      /// зміст цього методу з кодовим редактором.
      /// </summary>
      void InitializeComponent(void)
      {
            this->lstPagesDocList = (gcnew System::Windows::Forms::ListBox());
            this->nmrPagesNumber = (gcnew
System::Windows::Forms::NumericUpDown());
            this->cmbPagesDirect = (gcnew System::Windows::Forms::ComboBox());
            this->lstThemeDocList = (gcnew System::Windows::Forms::ListBox());
```

КОПІЯ 2024

```
            this->lstAuthorDocList = (gcnew System::Windows::Forms::ListBox());
            this->chkPagesFlag = (gcnew System::Windows::Forms::CheckBox());
            this->menuStrip = (gcnew System::Windows::Forms::MenuStrip());
            this->menuFile = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuFileSave = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuFileExit = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuHelp = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuHelpAbout = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->groupBox2 = (gcnew System::Windows::Forms::GroupBox());
            this->txtThemeFilter = (gcnew System::Windows::Forms::TextBox());
            this->txtAuthorFilter = (gcnew System::Windows::Forms::TextBox());
            this->lblFreeInstanceNumber = (gcnew
System::Windows::Forms::Label());
            this->lblTotalInstanceNumber = (gcnew
System::Windows::Forms::Label());
            this->chkThemeDocFlag = (gcnew System::Windows::Forms::CheckBox());
            this->chkAuthorDocFlag = (gcnew System::Windows::Forms::CheckBox());
            this->chkFreeInstanceFlag = (gcnew
System::Windows::Forms::CheckBox());
            this->chkTotalInstaceFlag = (gcnew
System::Windows::Forms::CheckBox());
            this->nmrFreeNumber = (gcnew
System::Windows::Forms::NumericUpDown());
            this->btnRefresh = (gcnew System::Windows::Forms::Button());
            this->nmrTotalNumber = (gcnew
System::Windows::Forms::NumericUpDown());
            this->tabHistory = (gcnew System::Windows::Forms::TabPage());
            this->btnClearLogs = (gcnew System::Windows::Forms::Button());
            this->groupOutput = (gcnew System::Windows::Forms::GroupBox());
            this->txtOutputHistory = (gcnew System::Windows::Forms::TextBox());
            this->groupInput = (gcnew System::Windows::Forms::GroupBox());
            this->txtInputHistory = (gcnew System::Windows::Forms::TextBox());
            this->nmrPageNumber = (gcnew
System::Windows::Forms::NumericUpDown());
            this->txtTheme = (gcnew System::Windows::Forms::TextBox());
            this->txtISBN = (gcnew System::Windows::Forms::TextBox());
            this->txtAuthor = (gcnew System::Windows::Forms::TextBox());
            this->txtDocName = (gcnew System::Windows::Forms::TextBox());
            this->tabReports = (gcnew System::Windows::Forms::TabPage());
            this->lblFreeNumber = (gcnew System::Windows::Forms::Label());
            this->lblTotalNumber = (gcnew System::Windows::Forms::Label());
            this->lblPagesNumber = (gcnew System::Windows::Forms::Label());
            this->lblTheme = (gcnew System::Windows::Forms::Label());
            this->statusStrip = (gcnew System::Windows::Forms::StatusStrip());
            this->lblStatus = (gcnew
System::Windows::Forms::ToolStripStatusLabel());
            this->txtFindString = (gcnew System::Windows::Forms::TextBox());
            this->tabControl = (gcnew System::Windows::Forms::TabControl());
            this->tabDocs = (gcnew System::Windows::Forms::TabPage());
            this->btnEdit = (gcnew System::Windows::Forms::Button());
            this->btnDelete = (gcnew System::Windows::Forms::Button());
            this->btnCreate = (gcnew System::Windows::Forms::Button());
            this->groupInfo = (gcnew System::Windows::Forms::GroupBox());
            this->txtPosition = (gcnew System::Windows::Forms::TextBox());
            this->btnFirst = (gcnew System::Windows::Forms::Button());
            this->btnPrev = (gcnew System::Windows::Forms::Button());
            this->btnNext = (gcnew System::Windows::Forms::Button());
            this->btnLast = (gcnew System::Windows::Forms::Button());
            this->lblISBN = (gcnew System::Windows::Forms::Label());
            this->lblAuthor = (gcnew System::Windows::Forms::Label());
            this->lblDocName = (gcnew System::Windows::Forms::Label());
            this->groupFinding = (gcnew System::Windows::Forms::GroupBox());
            this->btnFind = (gcnew System::Windows::Forms::Button());
```

KCTS 2024

```
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrPagesNumber))->BeginInit();
            this->menuStrip->SuspendLayout();
            this->groupBox 2-2->SuspendLayout();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrFreeNumber))->BeginInit();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrTotalNumber))->BeginInit();
            this->tabHistory->SuspendLayout();
            this->groupOutput->SuspendLayout();
            this->groupInput->SuspendLayout();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrPageNumber))->BeginInit();
            this->tabReports->SuspendLayout();
            this->statusStrip->SuspendLayout();
            this->tabControl->SuspendLayout();
            this->tabDocs->SuspendLayout();
            this->groupInfo->SuspendLayout();
            this->groupFinding->SuspendLayout();
            this->SuspendLayout();
            //
            // lstPagesDocList
            //
            this->lstPagesDocList->FormattingEnabled = true;
            this->lstPagesDocList->HorizontalScrollbar = true;
            this->lstPagesDocList->Location = System::Drawing::Point(478, 117);
            this->lstPagesDocList->Name = L"lstPagesDocList";
            this->lstPagesDocList->Size = System::Drawing::Size(223, 199);
            this->lstPagesDocList->TabIndex = 13;
            //
            // nmrPagesNumber
            //
            this->nmrPagesNumber->Location = System::Drawing::Point(583, 89);
            this->nmrPagesNumber->Maximum = System::Decimal(gcnew cli::array<
System::Int32 >(4) {65535, 0, 0, 0});
            this->nmrPagesNumber->Name = L"nmrPagesNumber";
            this->nmrPagesNumber->Size = System::Drawing::Size(64, 20);
            this->nmrPagesNumber->TabIndex = 6;
            //
            // cmbPagesDirect
            //
            this->cmbPagesDirect->FormattingEnabled = true;
            this->cmbPagesDirect->Location = System::Drawing::Point(479, 89);
            this->cmbPagesDirect->Name = L"cmbPagesDirect";
            this->cmbPagesDirect->Size = System::Drawing::Size(89, 21);
            this->cmbPagesDirect->TabIndex = 5;
            //
            // lstThemeDocList
            //
            this->lstThemeDocList->FormattingEnabled = true;
            this->lstThemeDocList->HorizontalScrollbar = true;
            this->lstThemeDocList->Location = System::Drawing::Point(242, 117);
            this->lstThemeDocList->Name = L"lstThemeDocList";
            this->lstThemeDocList->Size = System::Drawing::Size(223, 199);
            this->lstThemeDocList->TabIndex = 9;
            //
            // lstAuthorDocList
            //
            this->lstAuthorDocList->FormattingEnabled = true;
            this->lstAuthorDocList->HorizontalScrollbar = true;
            this->lstAuthorDocList->Location = System::Drawing::Point(6, 117);
            this->lstAuthorDocList->Name = L"lstAuthorDocList";
            this->lstAuthorDocList->Size = System::Drawing::Size(223, 199);
            this->lstAuthorDocList->TabIndex = 8;
            //
            // chkPagesFlag
            //
            this->chkPagesFlag->AutoSize = true;
            this->chkPagesFlag->Location = System::Drawing::Point(479, 68);
```

КСПЭ-2024

```
            this->chkPagesFlag->Name = L"chkPagesFlag";
            this->chkPagesFlag->Size = System::Drawing::Size(75, 17);
            this->chkPagesFlag->TabIndex = 4;
            this->chkPagesFlag->Text = L"Сторінок:";
            this->chkPagesFlag->UseVisualStyleBackColor = true;
            //
            // menuStrip
            //
            this->menuStrip->Items->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(2) {this->menuFile, this->menuHelp});
            this->menuStrip->Location = System::Drawing::Point(0, 0);
            this->menuStrip->Name = L"menuStrip";
            this->menuStrip->Size = System::Drawing::Size(722, 24);
            this->menuStrip->TabIndex = 9;
            this->menuStrip->Text = L"menuStrip1";
            //
            // menuFile
            //
            this->menuFile->DropDownItems->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(2) {this->menuFileSave,
                this->menuFileExit});
            this->menuFile->Name = L"menuFile";
            this->menuFile->Size = System::Drawing::Size(45, 20);
            this->menuFile->Text = L"Файл";
            //
            // menuFileSave
            //
            this->menuFileSave->Name = L"menuFileSave";
            this->menuFileSave->Size = System::Drawing::Size(152, 22);
            this->menuFileSave->Text = L"Зберегти";
            this->menuFileSave->Click += gcnew System::EventHandler(this,
&frmWorker::menuFileSave_Click);
            //
            // menuFileExit
            //
            this->menuFileExit->Name = L"menuFileExit";
            this->menuFileExit->Size = System::Drawing::Size(152, 22);
            this->menuFileExit->Text = L"Вихід";
            this->menuFileExit->Click += gcnew System::EventHandler(this,
&frmWorker::menuFileExit_Click);
            //
            // menuHelp
            //
            this->menuHelp->DropDownItems->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(1) {this->menuHelpAbout});
            this->menuHelp->Name = L"menuHelp";
            this->menuHelp->Size = System::Drawing::Size(60, 20);
            this->menuHelp->Text = L"Довідка";
            //
            // menuHelpAbout
            //
            this->menuHelpAbout->Name = L"menuHelpAbout";
            this->menuHelpAbout->Size = System::Drawing::Size(166, 22);
            this->menuHelpAbout->Text = L"Про програму...";
            this->menuHelpAbout->Click += gcnew System::EventHandler(this,
&frmWorker::menuHelpAbout_Click);
            //
            // groupBox2
            //
            this->groupBox 2-2->Controls->Add(this->lstPagesDocList);
            this->groupBox 2-2->Controls->Add(this->nmrPagesNumber);
            this->groupBox 2-2->Controls->Add(this->cmbPagesDirect);
            this->groupBox 2-2->Controls->Add(this->chkPagesFlag);
            this->groupBox 2-2->Controls->Add(this->lstThemeDocList);
            this->groupBox 2-2->Controls->Add(this->lstAuthorDocList);
            this->groupBox 2-2->Controls->Add(this->txtThemeFilter);
            this->groupBox 2-2->Controls->Add(this->txtAuthorFilter);
            this->groupBox 2-2->Controls->Add(this->lblFreeInstanceNumber);
            this->groupBox 2-2->Controls->Add(this->lblTotalInstanceNumber);
```

Копія 2024

```
            this->groupBox 2-2->Controls->Add(this->chkThemeDocFlag);
            this->groupBox 2-2->Controls->Add(this->chkAuthorDocFlag);
            this->groupBox 2-2->Controls->Add(this->chkFreeInstanceFlag);
            this->groupBox 2-2->Controls->Add(this->chkTotalInstaceFlag);
            this->groupBox 2-2->Location = System::Drawing::Point(8, 6);
            this->groupBox 2-2->Name = L"groupBox2";
            this->groupBox 2-2->Size = System::Drawing::Size(712, 325);
            this->groupBox 2-2->TabIndex = 0;
            this->groupBox 2-2->TabStop = false;
            //
            // txtThemeFilter
            //
            this->txtThemeFilter->Location = System::Drawing::Point(242, 90);
            this->txtThemeFilter->Name = L"txtThemeFilter";
            this->txtThemeFilter->Size = System::Drawing::Size(223, 20);
            this->txtThemeFilter->TabIndex = 7;
            //
            // txtAuthorFilter
            //
            this->txtAuthorFilter->Location = System::Drawing::Point(6, 90);
            this->txtAuthorFilter->Name = L"txtAuthorFilter";
            this->txtAuthorFilter->Size = System::Drawing::Size(223, 20);
            this->txtAuthorFilter->TabIndex = 6;
            //
            // lblFreeInstanceNumber
            //
            this->lblFreeInstanceNumber->AutoSize = true;
            this->lblFreeInstanceNumber->Location = System::Drawing::Point(476,
42);
            this->lblFreeInstanceNumber->Name = L"lblFreeInstanceNumber";
            this->lblFreeInstanceNumber->Size = System::Drawing::Size(26, 13);
            this->lblFreeInstanceNumber->TabIndex = 5;
            this->lblFreeInstanceNumber->Text = L"Н/Д";
            //
            // lblTotalInstanceNumber
            //
            this->lblTotalInstanceNumber->AutoSize = true;
            this->lblTotalInstanceNumber->Location = System::Drawing::Point(476,
19);
            this->lblTotalInstanceNumber->Name = L"lblTotalInstanceNumber";
            this->lblTotalInstanceNumber->Size = System::Drawing::Size(26, 13);
            this->lblTotalInstanceNumber->TabIndex = 4;
            this->lblTotalInstanceNumber->Text = L"Н/Д";
            //
            // chkThemeDocFlag
            //
            this->chkThemeDocFlag->AutoSize = true;
            this->chkThemeDocFlag->Location = System::Drawing::Point(240, 67);
            this->chkThemeDocFlag->Name = L"chkThemeDocFlag";
            this->chkThemeDocFlag->Size = System::Drawing::Size(176, 17);
            this->chkThemeDocFlag->TabIndex = 3;
            this->chkThemeDocFlag->Text = L"Список документів за темою:";
            this->chkThemeDocFlag->UseVisualStyleBackColor = true;
            //
            // chkAuthorDocFlag
            //
            this->chkAuthorDocFlag->AutoSize = true;
            this->chkAuthorDocFlag->Location = System::Drawing::Point(6, 67);
            this->chkAuthorDocFlag->Name = L"chkAuthorDocFlag";
            this->chkAuthorDocFlag->Size = System::Drawing::Size(165, 17);
            this->chkAuthorDocFlag->TabIndex = 2;
            this->chkAuthorDocFlag->Text = L"Список документів автора:";
            this->chkAuthorDocFlag->UseVisualStyleBackColor = true;
            //
            // chkFreeInstanceFlag
            //
            this->chkFreeInstanceFlag->AutoSize = true;
            this->chkFreeInstanceFlag->Location = System::Drawing::Point(6, 42);
            this->chkFreeInstanceFlag->Name = L"chkFreeInstanceFlag";
```

КПВ-2024

```
            this->chkFreeInstanceFlag->Size = System::Drawing::Size(208, 17);
            this->chkFreeInstanceFlag->TabIndex = 1;
            this->chkFreeInstanceFlag->Text = L"Кількість екземплярів на
виконання";
            this->chkFreeInstanceFlag->UseVisualStyleBackColor = true;
            //
            // chkTotalInstaceFlag
            //
            this->chkTotalInstaceFlag->AutoSize = true;
            this->chkTotalInstaceFlag->Location = System::Drawing::Point(6, 19);
            this->chkTotalInstaceFlag->Name = L"chkTotalInstaceFlag";
            this->chkTotalInstaceFlag->Size = System::Drawing::Size(293, 17);
            this->chkTotalInstaceFlag->TabIndex = 0;
            this->chkTotalInstaceFlag->Text = L"Загальна кількість екземплярів у
відібраному списку";
            this->chkTotalInstaceFlag->UseVisualStyleBackColor = true;
            //
            // nmrFreeNumber
            //
            this->nmrFreeNumber->Location = System::Drawing::Point(372, 196);
            this->nmrFreeNumber->Maximum = System::Decimal(gcnew cli::array<
System::Int32 >(4) {65535, 0, 0, 0});
            this->nmrFreeNumber->Name = L"nmrFreeNumber";
            this->nmrFreeNumber->ReadOnly = true;
            this->nmrFreeNumber->Size = System::Drawing::Size(80, 20);
            this->nmrFreeNumber->TabIndex = 8;
            //
            // btnRefresh
            //
            this->btnRefresh->Location = System::Drawing::Point(611, 337);
            this->btnRefresh->Name = L"btnRefresh";
            this->btnRefresh->Size = System::Drawing::Size(99, 23);
            this->btnRefresh->TabIndex = 7;
            this->btnRefresh->Text = L"Зберегти";
            this->btnRefresh->UseVisualStyleBackColor = true;
            this->btnRefresh->Click += gcnew System::EventHandler(this,
&frmWorker::btnRefresh_Click);
            //
            // nmrTotalNumber
            //
            this->nmrTotalNumber->Location = System::Drawing::Point(139, 196);
            this->nmrTotalNumber->Maximum = System::Decimal(gcnew cli::array<
System::Int32 >(4) {65535, 0, 0, 0});
            this->nmrTotalNumber->Name = L"nmrTotalNumber";
            this->nmrTotalNumber->Size = System::Drawing::Size(80, 20);
            this->nmrTotalNumber->TabIndex = 7;
            this->nmrTotalNumber->ValueChanged += gcnew
System::EventHandler(this, &frmWorker::nmrTotalNumber_ValueChanged);
            //
            // tabHistory
            //
            this->tabHistory->Controls->Add(this->btnClearLogs);
            this->tabHistory->Controls->Add(this->groupOutput);
            this->tabHistory->Controls->Add(this->groupInput);
            this->tabHistory->Location = System::Drawing::Point(4, 22);
            this->tabHistory->Name = L"tabHistory";
            this->tabHistory->Size = System::Drawing::Size(717, 366);
            this->tabHistory->TabIndex = 2;
            this->tabHistory->Text = L"Історія";
            this->tabHistory->UseVisualStyleBackColor = true;
            //
            // btnClearLogs
            //
            this->btnClearLogs->Location = System::Drawing::Point(631, 339);
            this->btnClearLogs->Name = L"btnClearLogs";
            this->btnClearLogs->Size = System::Drawing::Size(75, 23);
            this->btnClearLogs->TabIndex = 0;
            this->btnClearLogs->Text = L"Очистити";
            this->btnClearLogs->UseVisualStyleBackColor = true;
```

КПЗ 2024

```
            this->btnClearLogs->Click += gcnew System::EventHandler(this,
&frmWorker::btnClearLogs_Click);
            //
            // groupOutput
            //
            this->groupOutput->Controls->Add(this->txtOutputHistory);
            this->groupOutput->Location = System::Drawing::Point(6, 166);
            this->groupOutput->Name = L"groupOutput";
            this->groupOutput->Size = System::Drawing::Size(700, 167);
            this->groupOutput->TabIndex = 1;
            this->groupOutput->TabStop = false;
            this->groupOutput->Text = L"Історія виконань";
            //
            // txtOutputHistory
            //
            this->txtOutputHistory->Location = System::Drawing::Point(6, 19);
            this->txtOutputHistory->Multiline = true;
            this->txtOutputHistory->Name = L"txtOutputHistory";
            this->txtOutputHistory->ScrollBars =
System::Windows::Forms::ScrollBars::Both;
            this->txtOutputHistory->Size = System::Drawing::Size(688, 139);
            this->txtOutputHistory->TabIndex = 2;
            //
            // groupInput
            //
            this->groupInput->Controls->Add(this->txtInputHistory);
            this->groupInput->Location = System::Drawing::Point(8, 3);
            this->groupInput->Name = L"groupInput";
            this->groupInput->Size = System::Drawing::Size(700, 157);
            this->groupInput->TabIndex = 0;
            this->groupInput->TabStop = false;
            this->groupInput->Text = L"Історія надходжень";
            //
            // txtInputHistory
            //
            this->txtInputHistory->Location = System::Drawing::Point(6, 19);
            this->txtInputHistory->Multiline = true;
            this->txtInputHistory->Name = L"txtInputHistory";
            this->txtInputHistory->ScrollBars =
System::Windows::Forms::ScrollBars::Both;
            this->txtInputHistory->Size = System::Drawing::Size(688, 130);
            this->txtInputHistory->TabIndex = 1;
            //
            // nmrPageNumber
            //
            this->nmrPageNumber->Location = System::Drawing::Point(139, 162);
            this->nmrPageNumber->Maximum = System::Decimal(gcnew cli::array<
System::Int32 >(4) {65535, 0, 0, 0});
            this->nmrPageNumber->Name = L"nmrPageNumber";
            this->nmrPageNumber->Size = System::Drawing::Size(80, 20);
            this->nmrPageNumber->TabIndex = 6;
            //
            // txtTheme
            //
            this->txtTheme->Location = System::Drawing::Point(139, 128);
            this->txtTheme->Name = L"txtTheme";
            this->txtTheme->Size = System::Drawing::Size(302, 20);
            this->txtTheme->TabIndex = 5;
            //
            // txtISBN
            //
            this->txtISBN->Location = System::Drawing::Point(139, 94);
            this->txtISBN->Name = L"txtISBN";
            this->txtISBN->Size = System::Drawing::Size(163, 20);
            this->txtISBN->TabIndex = 4;
            //
            // txtAuthor
            //
            this->txtAuthor->Location = System::Drawing::Point(139, 60);
```

КОПІЯ 2024

```
            this->txtAuthor->Name = L"txtAuthor";
            this->txtAuthor->Size = System::Drawing::Size(505, 20);
            this->txtAuthor->TabIndex = 3;
            //
            // txtDocName
            //
            this->txtDocName->Location = System::Drawing::Point(139, 26);
            this->txtDocName->Name = L"txtDocName";
            this->txtDocName->Size = System::Drawing::Size(557, 20);
            this->txtDocName->TabIndex = 2;
            //
            // tabReports
            //
            this->tabReports->Controls->Add(this->btnRefresh);
            this->tabReports->Controls->Add(this->groupBox2);
            this->tabReports->Location = System::Drawing::Point(4, 22);
            this->tabReports->Name = L"tabReports";
            this->tabReports->Padding = System::Windows::Forms::Padding(3);
            this->tabReports->Size = System::Drawing::Size(717, 366);
            this->tabReports->TabIndex = 1;
            this->tabReports->Text = L"Звіти";
            this->tabReports->UseVisualStyleBackColor = true;
            //
            // lblFreeNumber
            //
            this->lblFreeNumber->AutoSize = true;
            this->lblFreeNumber->Location = System::Drawing::Point(263, 196);
            this->lblFreeNumber->Name = L"lblFreeNumber";
            this->lblFreeNumber->Size = System::Drawing::Size(103, 13);
            this->lblFreeNumber->TabIndex = 6;
            this->lblFreeNumber->Text = L"Вільно екземплярів";
            //
            // lblTotalNumber
            //
            this->lblTotalNumber->AutoSize = true;
            this->lblTotalNumber->Location = System::Drawing::Point(34, 196);
            this->lblTotalNumber->Name = L"lblTotalNumber";
            this->lblTotalNumber->Size = System::Drawing::Size(105, 13);
            this->lblTotalNumber->TabIndex = 5;
            this->lblTotalNumber->Text = L"Всього екземплярів";
            //
            // lblPagesNumber
            //
            this->lblPagesNumber->AutoSize = true;
            this->lblPagesNumber->Location = System::Drawing::Point(40, 162);
            this->lblPagesNumber->Name = L"lblPagesNumber";
            this->lblPagesNumber->Size = System::Drawing::Size(99, 13);
            this->lblPagesNumber->TabIndex = 4;
            this->lblPagesNumber->Text = L"Кількість сторінок";
            //
            // lblTheme
            //
            this->lblTheme->AutoSize = true;
            this->lblTheme->Location = System::Drawing::Point(108, 128);
            this->lblTheme->Name = L"lblTheme";
            this->lblTheme->Size = System::Drawing::Size(31, 13);
            this->lblTheme->TabIndex = 3;
            this->lblTheme->Text = L"Тема";
            //
            // statusStrip
            //
            this->statusStrip->Items->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(1) {this->lblStatus});
            this->statusStrip->Location = System::Drawing::Point(0, 417);
            this->statusStrip->Name = L"statusStrip";
            this->statusStrip->Size = System::Drawing::Size(722, 22);
            this->statusStrip->TabIndex = 10;
            this->statusStrip->Text = L"statusStrip1";
            //
```

КПІ-2024

```
            // lblStatus
            //
            this->lblStatus->Name = L"lblStatus";
            this->lblStatus->Size = System::Drawing::Size(109, 17);
            this->lblStatus->Text = L"toolStripStatusLabel1";
            //
            // txtFindString
            //
            this->txtFindString->Location = System::Drawing::Point(6, 19);
            this->txtFindString->Name = L"txtFindString";
            this->txtFindString->Size = System::Drawing::Size(609, 20);
            this->txtFindString->TabIndex = 0;
            //
            // tabControl
            //
            this->tabControl->Controls->Add(this->tabDocs);
            this->tabControl->Controls->Add(this->tabReports);
            this->tabControl->Controls->Add(this->tabHistory);
            this->tabControl->Location = System::Drawing::Point(0, 27);
            this->tabControl->Name = L"tabControl";
            this->tabControl->SelectedIndex = 0;
            this->tabControl->Size = System::Drawing::Size(725, 392);
            this->tabControl->TabIndex = 11;
            //
            // tabDocs
            //
            this->tabDocs->Controls->Add(this->btnEdit);
            this->tabDocs->Controls->Add(this->btnDelete);
            this->tabDocs->Controls->Add(this->btnCreate);
            this->tabDocs->Controls->Add(this->groupInfo);
            this->tabDocs->Controls->Add(this->groupFinding);
            this->tabDocs->Location = System::Drawing::Point(4, 22);
            this->tabDocs->Name = L"tabDocs";
            this->tabDocs->Padding = System::Windows::Forms::Padding(3);
            this->tabDocs->Size = System::Drawing::Size(717, 366);
            this->tabDocs->TabIndex = 0;
            this->tabDocs->Text = L"Документи";
            this->tabDocs->UseVisualStyleBackColor = true;
            //
            // btnEdit
            //
            this->btnEdit->Location = System::Drawing::Point(402, 337);
            this->btnEdit->Name = L"btnEdit";
            this->btnEdit->Size = System::Drawing::Size(99, 23);
            this->btnEdit->TabIndex = 14;
            this->btnEdit->Text = L"Редагувати";
            this->btnEdit->UseVisualStyleBackColor = true;
            this->btnEdit->Click += gcnew System::EventHandler(this,
&frmWorker::btnEdit_Click);
            //
            // btnDelete
            //
            this->btnDelete->Location = System::Drawing::Point(507, 337);
            this->btnDelete->Name = L"btnDelete";
            this->btnDelete->Size = System::Drawing::Size(99, 23);
            this->btnDelete->TabIndex = 15;
            this->btnDelete->Text = L"Видалити";
            this->btnDelete->UseVisualStyleBackColor = true;
            this->btnDelete->Click += gcnew System::EventHandler(this,
&frmWorker::btnDelete_Click);
            //
            // btnCreate
            //
            this->btnCreate->Location = System::Drawing::Point(612, 337);
            this->btnCreate->Name = L"btnCreate";
            this->btnCreate->Size = System::Drawing::Size(99, 23);
            this->btnCreate->TabIndex = 16;
            this->btnCreate->Text = L"Створити";
            this->btnCreate->UseVisualStyleBackColor = true;
```

КПІ ІВ-2024

```
            this->btnCreate->Click += gcnew System::EventHandler(this,
&frmWorker::btnCreate_Click);
            //
            // groupInfo
            //
            this->groupInfo->Controls->Add(this->txtPosition);
            this->groupInfo->Controls->Add(this->btnFirst);
            this->groupInfo->Controls->Add(this->btnPrev);
            this->groupInfo->Controls->Add(this->btnNext);
            this->groupInfo->Controls->Add(this->btnLast);
            this->groupInfo->Controls->Add(this->nmrFreeNumber);
            this->groupInfo->Controls->Add(this->nmrTotalNumber);
            this->groupInfo->Controls->Add(this->nmrPageNumber);
            this->groupInfo->Controls->Add(this->txtTheme);
            this->groupInfo->Controls->Add(this->txtISBN);
            this->groupInfo->Controls->Add(this->txtAuthor);
            this->groupInfo->Controls->Add(this->txtDocName);
            this->groupInfo->Controls->Add(this->lblFreeNumber);
            this->groupInfo->Controls->Add(this->lblTotalNumber);
            this->groupInfo->Controls->Add(this->lblPagesNumber);
            this->groupInfo->Controls->Add(this->lblTheme);
            this->groupInfo->Controls->Add(this->lblISBN);
            this->groupInfo->Controls->Add(this->lblAuthor);
            this->groupInfo->Controls->Add(this->lblDocName);
            this->groupInfo->Location = System::Drawing::Point(6, 65);
            this->groupInfo->Name = L"groupInfo";
            this->groupInfo->Size = System::Drawing::Size(702, 266);
            this->groupInfo->TabIndex = 9;
            this->groupInfo->TabStop = false;
            this->groupInfo->Text = L"Інформація про документ";
            //
            // txtPosition
            //
            this->txtPosition->Location = System::Drawing::Point(564, 241);
            this->txtPosition->Name = L"txtPosition";
            this->txtPosition->Size = System::Drawing::Size(58, 20);
            this->txtPosition->TabIndex = 11;
            this->txtPosition->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
            this->txtPosition->KeyDown += gcnew
System::Windows::Forms::KeyEventHandler(this, &frmWorker::txtPosition_KeyDown);
            //
            // btnFirst
            //
            this->btnFirst->Location = System::Drawing::Point(490, 241);
            this->btnFirst->Name = L"btnFirst";
            this->btnFirst->Size = System::Drawing::Size(31, 20);
            this->btnFirst->TabIndex = 9;
            this->btnFirst->Text = L"|<<";
            this->btnFirst->UseVisualStyleBackColor = true;
            this->btnFirst->Click += gcnew System::EventHandler(this,
&frmWorker::btnFirst_Click);
            //
            // btnPrev
            //
            this->btnPrev->Location = System::Drawing::Point(527, 241);
            this->btnPrev->Name = L"btnPrev";
            this->btnPrev->Size = System::Drawing::Size(31, 20);
            this->btnPrev->TabIndex = 10;
            this->btnPrev->Text = L"<<";
            this->btnPrev->UseVisualStyleBackColor = true;
            this->btnPrev->Click += gcnew System::EventHandler(this,
&frmWorker::btnPrev_Click);
            //
            // btnNext
            //
            this->btnNext->Location = System::Drawing::Point(628, 240);
            this->btnNext->Name = L"btnNext";
            this->btnNext->Size = System::Drawing::Size(31, 20);
```

КОПІЯ 2024

```
            this->btnNext->TabIndex = 12;
            this->btnNext->Text = L">>";
            this->btnNext->UseVisualStyleBackColor = true;
            this->btnNext->Click += gcnew System::EventHandler(this,
&frmWorker::btnNext_Click);
            //
            // btnLast
            //
            this->btnLast->Location = System::Drawing::Point(665, 240);
            this->btnLast->Name = L"btnLast";
            this->btnLast->Size = System::Drawing::Size(31, 20);
            this->btnLast->TabIndex = 13;
            this->btnLast->Text = L">>|";
            this->btnLast->UseVisualStyleBackColor = true;
            this->btnLast->Click += gcnew System::EventHandler(this,
&frmWorker::btnLast_Click);
            //
            // lblISBN
            //
            this->lblISBN->AutoSize = true;
            this->lblISBN->Location = System::Drawing::Point(2, 94);
            this->lblISBN->Name = L"lblISBN";
            this->lblISBN->Size = System::Drawing::Size(137, 13);
            this->lblISBN->TabIndex = 2;
            this->lblISBN->Text = L"Вхідний номер документу";
            //
            // lblAuthor
            //
            this->lblAuthor->AutoSize = true;
            this->lblAuthor->Location = System::Drawing::Point(101, 60);
            this->lblAuthor->Name = L"lblAuthor";
            this->lblAuthor->Size = System::Drawing::Size(38, 13);
            this->lblAuthor->TabIndex = 1;
            this->lblAuthor->Text = L"Автор";
            //
            // lblDocName
            //
            this->lblDocName->AutoSize = true;
            this->lblDocName->Location = System::Drawing::Point(44, 26);
            this->lblDocName->Name = L"lblDocName";
            this->lblDocName->Size = System::Drawing::Size(95, 13);
            this->lblDocName->TabIndex = 0;
            this->lblDocName->Text = L"Назва документу";
            //
            // groupFinding
            //
            this->groupFinding->Controls->Add(this->btnFind);
            this->groupFinding->Controls->Add(this->txtFindString);
            this->groupFinding->Location = System::Drawing::Point(6, 6);
            this->groupFinding->Name = L"groupFinding";
            this->groupFinding->Size = System::Drawing::Size(702, 53);
            this->groupFinding->TabIndex = 8;
            this->groupFinding->TabStop = false;
            this->groupFinding->Text = L"Ключові рядки для пошуку";
            //
            // btnFind
            //
            this->btnFind->Location = System::Drawing::Point(621, 16);
            this->btnFind->Name = L"btnFind";
            this->btnFind->Size = System::Drawing::Size(75, 23);
            this->btnFind->TabIndex = 1;
            this->btnFind->Text = L"Знайти";
            this->btnFind->UseVisualStyleBackColor = true;
            this->btnFind->Click += gcnew System::EventHandler(this,
&frmWorker::btnFind_Click);
            //
            // frmWorker
            //
            this->AutoScaleDimensions = System::Drawing::Size(6, 13);
```

КПЗ_2024

```
            this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(722, 439);
            this->Controls->Add(this->menuStrip);
            this->Controls->Add(this->statusStrip);
            this->Controls->Add(this->tabControl);
            this->Name = L"frmWorker";
            this->Text = L"Режим  адміністратора - Віртуальний офіс для потреб
виробництва ";
            this->Load += gcnew System::EventHandler(this,
&frmWorker::frmWorker_Load);
            this->FormClosed += gcnew
System::Windows::Forms::FormClosedEventHandler(this,
&frmWorker::frmWorker_FormClosed);
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrPagesNumber))->EndInit();
            this->menuStrip->ResumeLayout(false);
            this->menuStrip->PerformLayout();
            this->groupBox 2-2->ResumeLayout(false);
            this->groupBox 2-2->PerformLayout();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrFreeNumber))->EndInit();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrTotalNumber))->EndInit();
            this->tabHistory->ResumeLayout(false);
            this->groupOutput->ResumeLayout(false);
            this->groupOutput->PerformLayout();
            this->groupInput->ResumeLayout(false);
            this->groupInput->PerformLayout();
            (cli::safe_cast<System::ComponentModel::ISupportInitialize^  >(this-
>nmrPageNumber))->EndInit();
            this->tabReports->ResumeLayout(false);
            this->statusStrip->ResumeLayout(false);
            this->statusStrip->PerformLayout();
            this->tabControl->ResumeLayout(false);
            this->tabDocs->ResumeLayout(false);
            this->groupInfo->ResumeLayout(false);
            this->groupInfo->PerformLayout();
            this->groupFinding->ResumeLayout(false);
            this->groupFinding->PerformLayout();
            this->ResumeLayout(false);
            this->PerformLayout();

      }
#pragma endregion
/*****************************************************************/
private: System::Void LockingForm()
       {
             // Блокування всіх елементів керування
             btnFind->Enabled = false;
             btnEdit->Enabled = false;
             btnDelete->Enabled = false;
             btnCreate->Enabled = false;
             btnFirst->Enabled = false;
             btnLast->Enabled = false;
             btnNext->Enabled = false;
             btnPrev->Enabled = false;
             txtPosition->Enabled = false;
             menuStrip->Enabled = false;
             tabControl->TabPages[1]->Enabled = false;
             tabControl->TabPages[2]->Enabled = false;
       }
/*****************************************************************/
private: System::Void UnlockingForm()
       {
             // Розблокування всіх елементів керування
             btnFind->Enabled = true;
             btnEdit->Enabled = true;
             btnDelete->Enabled = true;
             btnCreate->Enabled = true;
```

КОПІЯ 2024

```
            btnFirst->Enabled = true;
            btnLast->Enabled = true;
            btnNext->Enabled = true;
            btnPrev->Enabled = true;
            txtPosition->Enabled = true;
            menuStrip->Enabled = true;
            tabControl->TabPages[1]->Enabled = true;
            tabControl->TabPages[2]->Enabled = true;
      }
/*************************************************************************/
private: System::Void UpdateView()
      {
            // Вивести історію
            txtInputHistory->Text = Worker->ViewInputLog();
            txtOutputHistory->Text = Worker->ViewOutputLog();

            // Список не порожній?
            if(listView->Count != 0)
            {
                  // Перевірити індекс поточного елемента
                  if(ddCurrentIndex >= listView->Count)
                  {
                        // Установити покажчик на останній елемент
                        ddCurrentIndex = listView->Count - 1;
                  }

                  // Значення індексу негативно?
                  if(ddCurrentIndex < 0)
                  {
                        // Установити індекс на перший елемент
                        ddCurrentIndex = 0;
                  }

                  // Одержати характеристики документа й заповнити поля форми
                  txtDocName->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetName();
                  txtAuthor->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetAuthor();
                  txtISBN->Text = ((CDoc^)listView[ddCurrentIndex])->GetISBN();
                  txtTheme->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetTheme();
                  nmrPageNumber->Value = ((CDoc^)listView[ddCurrentIndex])-
>GetPages();
                  nmrTotalNumber->Value = ((CDoc^)listView[ddCurrentIndex])-
>GetTotalNumber();
                  nmrFreeNumber->Value = ((CDoc^)listView[ddCurrentIndex])-
>GetFreeNumber();

                  // Установити поточну позицію
                  txtPosition->Text = Convert::ToString(ddCurrentIndex + 1) + "
/ " +
                        Convert::ToString(listView->Count);

                  // Розблокувати кнопки редагування й видалення
                  btnDelete->Enabled = true;
                  btnEdit->Enabled = true;
            }
            else
            {
                  // Очистити всі поля
                  txtDocName->Text = "";
                  txtAuthor->Text = "";
                  txtISBN->Text = "";
                  txtTheme->Text = "";
                  nmrPageNumber->Value = 0;
                  nmrTotalNumber->Value = 0;
                  nmrFreeNumber->Value = 0;
                  // Поточна позиція - 0
                  txtPosition->Text = "0/0";
```

КОПІЯ 2024

```
                    // Установити індекс поточного елемента
                    ddCurrentIndex = 0;
                    // Заблокувати кнопки редагування й видалення
                    btnDelete->Enabled = false;
                    btnEdit->Enabled = false;
                    // Вивести стан
                    lblStatus->Text = "Немає записів для відображення";
              }
        }
/*************************************************************************/
private: System::Void frmWorker_Load(System::Object^  sender, System::EventArgs^
e)
        {
              // Створити об'єкт "Адміністратор"
              Worker = gcnew CWorker();

              // Завантажити список документів
              Worker->LoadDocList();

              // Завантажити історію
              Worker->ReadLogs();

              // Обновити форму
             UpdateView();

             // Вивести стан
             lblStatus->Text = "Для відображення існуючого списку скористайтеся
пошуком";

             // Заповнення випадаючого списку вибору документів по кількості
сторінок
             cmbPagesDirect->Items->Add(PAGES_LESS);
             cmbPagesDirect->Items->Add(PAGES_GREATER);
             cmbPagesDirect->Items->Add(PAGES_EQUAL);
        }
 /*************************************************************************/
private: System::Void btnCreate_Click(System::Object^  sender,
System::EventArgs^  e)
        {
              // Кнопка "Створити" була натиснута раніше?
              if(IsCreateClicked)
              {
                    // Забрати зайві пробіли у всіх полях
                    txtDocName->Text = txtDocName->Text->Trim();
                    txtAuthor->Text = txtAuthor->Text->Trim();
                    txtISBN->Text = txtISBN->Text->Trim();
                    txtTheme->Text = txtTheme->Text->Trim();

                    // Перевірити значимі поля на коректність
                    if(txtDocName->Text == "" || txtISBN->Text == "")
                    {
                          // Обов'язкові поля не заповнені
                          MessageBox::Show("Поля 'Назва документа' і 'вхідний
номер документа' обов'язкові для заповнення!",
                                "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);
                          return;
                    }

                    // Перевірити кількість екземплярів
                    if(Convert::ToInt32(nmrTotalNumber->Value) <= 0)
                    {
                          // Вивести повідомлення про помилку
                          MessageBox::Show("Некоректне значення кількості
екземплярів",
                                "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);
                          return;
                    }
```

КОПЗ-2024

```
                    // Створити новий об'єкт "Документ"
                    CDoc^ NewDoc = gcnew CDoc;
                    // Заповнити поля об'єкта
                    NewDoc->SetName(txtDocName->Text);
                    NewDoc->SetAuthor(txtAuthor->Text);
                    NewDoc->SetISBN(txtISBN->Text);
                    NewDoc->SetTheme(txtTheme->Text);
                    NewDoc->SetPages(Convert::ToInt32(nmrPageNumber->Value));
                    NewDoc->SetTotalNumber(Convert::ToInt32(nmrTotalNumber-
>Value));
                    NewDoc->SetFreeNumber(Convert::ToInt32(nmrTotalNumber-
>Value));
                    // Додати документ у загальний список (якщо документ має
унікальний вхідний номер документа)
                    if(!Worker->AddDoc(NewDoc))
                    {
                          MessageBox::Show("Документ із таким вхідним номером
документа вже існує!",
                                "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);
                          return;
                    }

                    // Додати документ у список для відображення
                    listView->Add(NewDoc);

                    // Обновити поля на формі
                    ddCurrentIndex ++;
                    UpdateView();

                    // Розблокувати елементи керування
                    UnlockingForm();

                    // Перемінити напис на кнопці
                    btnCreate->Text = "Створити";

                    // Вивід стану
                    lblStatus->Text = "Запис доданий";

                   // Змінити значення прапора
                   IsCreateClicked = false;
              }
              else
              {
                    // Перемінити напис на кнопці
                    btnCreate->Text = "Зберегти";

                    // Заблокувати елементи керування
                    LockingForm();

                    // Розблокувати кнопку збереження
                    btnCreate->Enabled = true;

                    // Розблокувати поля для введення
                    txtDocName->Enabled = true;
                    txtAuthor->Enabled = true;
                    txtISBN->Enabled = true;
                    txtTheme->Enabled = true;
                    nmrPageNumber->Enabled = true;
                    nmrTotalNumber->Enabled = true;

                    // Установити початкові значення полів для введення
                    txtDocName->Text = "";
                    txtAuthor->Text = "";
                    txtISBN->Text = "";
                    txtTheme->Text = "";
                    nmrPageNumber->Value = 0;
                    nmrTotalNumber->Value = 1;
```

КПЗ 2024

```
                nmrFreeNumber->Value = 1;

                // Вивести стан
                lblStatus->Text = "Введіть значення полів";

                // Змінити значення прапора натискання по кнопці "Створити"
                IsCreateClicked = true;
            }
        }
/*************************************************************************/
private: System::Void btnDelete_Click(System::Object^  sender,
System::EventArgs^  e)
        {
            // Видалити документ із загального списку
            Int32 ddResult = Worker->RemoveDoc(
                ((CDoc^)listView[ddCurrentIndex])->GetISBNHash());

            // Перевірити код повернення
            switch(ddResult)
            {
            case 0:
                {
                    // Документ вилучений успішно
                    // Видалити документ зі списку для виводу
                    listView->RemoveAt(ddCurrentIndex);
                    // Обновити форму
                    UpdateView();

                    break;
                }
            case 1:
                {
                    // Документ не був вилучений, не всі екземпляри в
системі
                    MessageBox::Show("Видалення неможливо, тому що не всі
документи перебувають у системі",
                        "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);

                    break;
                }
            case 2:
                {
                    // Документ не знайдений
                    MessageBox::Show("Видалення неможливо, документ у
системі відсутній",
                          "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);
                    // Видалити документ зі списку для виводу
                    listView->RemoveAt(ddCurrentIndex);
                    // Обновити форму
                    UpdateView();

                    break;
                }
            }
        }
/*************************************************************************/
private: System::Void btnFind_Click(System::Object^  sender, System::EventArgs^
e)
        {
            // Забрати в рядку пошуку зайві пробіли й перетворити до нижнього
регістра
            String^ strValue = txtFindString->Text->Trim()->ToLower();

            // Перевірити рядок
            if(String::IsNullOrEmpty(strValue))
            {
                // Рядок порожня, запропонувати вивести весь список
```

КПЗ – 2024

```
                  if(MessageBox::Show("Рядок пошуку не заданий, вивести весь
список?",
                                                "Usrs Manager",
MessageBoxButtons::YesNo,
                                                MessageBoxIcon::Question) ==
::DialogResult::Yes)
                  {
                        // Перейти на вивід списку
                        goto OUT_LIST;
                  }
                  // Завершити роботу функції
                  return;
            }

OUT_LIST:   // Здійснити пошук документа по заданих параметрах
            listView = Worker->FindDoc(strValue);

            // Вивести стан
            lblStatus->Text = "Пошук завершений";
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnFirst_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на перший елемент
            ddCurrentIndex = 0;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnPrev_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на попередній елемент
            ddCurrentIndex ---i;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnNext_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на наступний елемент
            ddCurrentIndex ++;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnLast_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на останній елемент
            ddCurrentIndex = listView->Count - 1;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnEdit_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Кнопка "Редагувати" була натиснута раніше?
            if(IsEditClicked)
            {
                  // Забрати зайві пробіли
                  txtDocName->Text = txtDocName->Text->Trim();
                  txtAuthor->Text = txtAuthor->Text->Trim();
                  txtTheme->Text = txtTheme->Text->Trim();
```

КПІ 2024

```
                    // Перевірити значимі поля на коректність
                    if(txtDocName->Text == "")
                    {
                          // Обов'язкові поля не заповнені
                          MessageBox::Show("Поле 'Назва документа' обов'язково
для заповнення!",
                                "Usrs Manager", MessageBoxButtons::OK,
MessageBoxIcon::Error);
                          return;
                    }

                    // Відредагувати об'єкт у списках
                    ((CDoc^)listView[ddCurrentIndex])->SetName(txtDocName->Text);
                    ((CDoc^)listView[ddCurrentIndex])->SetAuthor(txtAuthor-
>Text);
                    ((CDoc^)listView[ddCurrentIndex])->SetTheme(txtTheme->Text);
                    ((CDoc^)listView[ddCurrentIndex])-
>SetPages(Convert::ToInt32(nmrPageNumber->Value));

                    // (???) Видалити документ зі списку перегляду
                    //Worker->RemoveDoc(((CDoc^)listView[ddCurrentIndex])-
>GetISBNHash());
                    // (???) Додати документ у список перегляду
                    //Worker->AddDoc((CDoc^)listView[ddCurrentIndex]);

                    // Обновити форму
                    UpdateView();

                    // Розблокувати елементи керування
                    UnlockingForm();

                    // Розблокувати некоректируємі поля
                    txtISBN->Enabled = true;
                    nmrTotalNumber->Enabled = true;
                    nmrFreeNumber->Enabled = true;

                    // Перемінити напис на кнопці
                    btnEdit->Text = "Редагувати";

                   // Змінити значення прапора натискання кнопки "Редагувати"
                    IsEditClicked = false;
              }
              else
              {
                    // Перемінити напис на кнопці
                    btnEdit->Text = "Зберегти";

                    // Заблокувати елементи керування
                    LockingForm();

                    // Розблокувати кнопку збереження
                    btnEdit->Enabled = true;

                    // Заблокувати некоректируємі поля
                    txtISBN->Enabled = false;
                    nmrTotalNumber->Enabled = false;
                    nmrFreeNumber->Enabled = false;

                    // Змінити значення прапора натискання кнопки "Редагувати"
                    IsEditClicked = true;
              }
        }
/*********************************************************************/
private: System::Void menuFileSave_Click(System::Object^  sender,
System::EventArgs^  e)
        {
              // Зберегти список документів у файл
              Worker->SaveDocList();
```

КОПЗ_2024

```
            // Зберегти історію
            Worker->WriteLogs();

            // Вивести стан
            lblStatus->Text = "Збереження виконане";
      }
/*************************************************************************/
private: System::Void menuFileExit_Click(System::Object^  sender,
System::EventArgs^  e)
      {
            // Вийти з додатка
            Application::Exit();
      }
/*************************************************************************/
private: System::Void nmrTotalNumber_ValueChanged(System::Object^  sender,
System::EventArgs^  e)
      {
            // Можливе натискання тільки при створенні документа
            // Кількість вільних екземплярів дорівнює загальній кількості
екземплярів
            nmrFreeNumber->Value = nmrTotalNumber->Value;
      }
/*************************************************************************/
   private: System::Void frmWorker_FormClosed(System::Object^  sender,
System::Windows::Forms::FormClosedEventArgs^  e)
     {
            // Вийти з додатка
            Application::Exit();
     }
/*************************************************************************/
private: System::Void btnClearLogs_Click(System::Object^  sender,
System::EventArgs^  e)
      {
            // Очистити історію
            Worker->ClearLogs();

            // Обновити форму
            UpdateView();

            // Вивести стан
            lblStatus->Text = "Очищення виконане";
      }
/*************************************************************************/
private: System::Void btnRefresh_Click(System::Object^  sender,
System::EventArgs^  e)
      {
            // Створити об'єкт "Лічильник"
            CCounter^ Counter = gcnew CCounter();

            // Установлений прапор підрахунку сумарної кількості екземплярів?
            if(chkTotalInstaceFlag->Checked)
            {
                  // Обчислити
                  lblTotalInstanceNumber->Text =
                        Convert::ToString(Counter-
>GetTotalInstanceNumber(listView));
            }

            // Установлений прапор підрахунку сумарної кількості вільних
екземплярів?
            if(chkFreeInstanceFlag->Checked)
            {
                  // Обчислити
                  lblFreeInstanceNumber->Text =
                        Convert::ToString(Counter-
>GetFreeInstanceNumber(listView));
            }
```

КПІ3_2024

```
            // Встановлений прапор підрахунку документів заданого автора?
            if(chkAuthorDocFlag->Checked)
            {
                  // Створити тимчасовий список
                  ArrayList^ listTemp = gcnew ArrayList();

                  // Одержати список документів заданого автора
                  listTemp = Counter->GetDocListOfAuthor(listView,
txtAuthorFilter->Text);

                  // Очистити список на формі
                  lstAuthorDocList->Items->Clear();
                  // Вивести список знайдених документів
                  for(Int32 i = 0; i < listTemp->Count; i++)
                  {
                        // Зчитати параметри документа й додати в список на
формі
                        lstAuthorDocList->Items->Add(
                             Convert::ToString(i + 1) + ". '" +
                             ((CDoc^)listTemp[i])->GetName() + "', '" +
                             ((CDoc^)listTemp[i])->GetAuthor() + "', що входить
номер документа " +
                             ((CDoc^)listTemp[i])->GetISBN());
                  }

                  // Очистити тимчасовий список
                  listTemp->Clear();
            }

            // Установлений прапор виводу списку документів по заданій темі?
            if(chkThemeDocFlag->Checked)
            {
                  // Створити тимчасовий список
                  ArrayList^ listTemp = gcnew ArrayList();

                  // Одержить список документів по заданій темі
                  listTemp = Counter->GetDocListOnTheme(listView,
txtThemeFilter->Text);

                  // Очистити список для виводу на формі
                  lstThemeDocList->Items->Clear();
                  // Вивести список знайдених документів
                  for(Int32 i = 0; i < listTemp->Count; i++)
                  {
                        // Завантажити параметри документа й додати в список на
формі
                        lstThemeDocList->Items->Add(
                             Convert::ToString(i + 1) + ". '" +
                             ((CDoc^)listTemp[i])->GetName() + "', '" +
                             ((CDoc^)listTemp[i])->GetTheme() + "', що входить
номер документа " +
                             ((CDoc^)listTemp[i])->GetISBN());
                  }
                  // Очистити тимчасовий список
                  listTemp->Clear();
            }
            // Установлений прапор виводу списку документів із заданою
кількістю сторінок?
            if(chkPagesFlag->Checked)
            {
                  // Створити тимчасовий список
                  ArrayList^ listTemp = gcnew ArrayList();

                  // Одержати список документів, у яких кількість сторінок
задовольняє
                  //  заданій умові
                  listTemp = Counter->GetDocListByPages(listView,
                        cmbPagesDirect->Text, Convert::ToInt32(nmrPagesNumber-
>Value));
```

КПЗ-2024

```
                    // Очистити список для виводу на формі
                    lstPagesDocList->Items->Clear();
                    // Вивести список знайдених документів
                    for(Int32 i = 0; i < listTemp->Count; i++)
                    {
                          // Завантажити параметри документа й додати в список на
формі
                          lstPagesDocList->Items->Add(
                               Convert::ToString(i + 1) + ". '" +
                               ((CDoc^)listTemp[i])->GetName() + "', " +
                               Convert::ToString(((CDoc^)listTemp[i])-
>GetPages())
                               + " стор., що входить номер документа " +
((CDoc^)listTemp[i])->GetISBN());
                    }
                    // Очистити тимчасовий список
                    listTemp->Clear();
              }
              // Вивести стан
              lblStatus->Text = "Відновлення необхідних звітів закінчене";
        }
/************************************************************************/
private: System::Void txtPosition_KeyDown(System::Object^  sender,
System::Windows::Forms::KeyEventArgs^  e)
        {
              // Натиснута клавіша "Enter"?
              if( e-e->KeyCode == Keys::Enter)
              {
                    try
                    {
                          // Спробувати одержати введений номер документа для
перегляду
                          ddCurrentIndex = Convert::ToInt32(txtPosition->Text) -
1;
                    }
                    catch(...)
                    {
                          // Якщо в процесі перетворення номера з рядка в число
                          //  виникла помилка, те ніяк на це не реагувати
                    }
                    // Обновити форму
                    UpdateView();
              }
              else
              {
                    // Текстове поле містить символ "/"?
                    if(txtPosition->Text->IndexOf("/") != -1)
                    {
                          // Для початку введення нового номера позиції,
необхідно
                          //  очистити текстове поле
                          txtPosition->Text = "";
                    }
              }
        }
/************************************************************************/
private: System::Void menuHelpAbout_Click(System::Object^  sender,
System::EventArgs^  e)
        {
              // Відобразити форму з інформацією про програму
              UsrsManager::frmAbout^ frmNew = gcnew UsrsManager::frmAbout();
              frmNew->Show();
        }
/************************************************************************/
};
}
```

КБТЗ_2024

Файл Reader.cpp - вікно користувача

```
/*****************************************************************************/
#include "StdAfx.h"
#include "Reader.h"
#include "Worker.h"
#include "Logger.h"
/*****************************************************************************/
using namespace System;
using namespace System::IO;
using namespace System::Collections;
using namespace System::Windows::Forms;
/*****************************************************************************/
#define READER_FILE_EXTENSION ".LMR"
/*****************************************************************************/
// NAME:          CReader
// DESCRIPTION:   Конструктор класу
// INPUT:         strUserName - ім'я облікового запису користувача
/*****************************************************************************/
CReader::CReader(String^ strUserName)
{
      // Сформувати ім'я файлу облікового запису
      strFileName = strUserName + READER_FILE_EXTENSION;
      // Створити список для документів користувача
      listMyDoc = gcnew ArrayList();
      // Створити об'єкт для ведення історії
      logUsing = gcnew CLogger();
}
/*****************************************************************************/
// NAME:          Load
// DESCRIPTION:   Функція завантаження списку документів користувача з файлу
// INPUT:         Worker - об'єкт, через який здійснюється одержання інформації
про
//                         книзі по вхідному номеру документа
/*****************************************************************************/
Boolean CReader::Load(CWorker^ Worker)
{
      // Перевірка існування файлу користувача
      if(!File::Exists(strFileName))
      {
            // Видати запит на створення нового файлу
            if(MessageBox::Show("Файл користувача не знайдений, створити
новий?",
                  "Система УЕД", MessageBoxButtons::YesNo,
                  MessageBoxIcon::Warning) == DialogResult::Yes)
            {
                  // Створити файловий потік
                  FileStream^ fsReaderFile = gcnew FileStream(strFileName,
                        FileMode::CreateNew);
                  // Відкрити файл користувача
                  StreamWriter^ swReaderFile = gcnew StreamWriter(fsReaderFile);
                  // Кількість документів у користувача дорівнює 0
                  swReaderFile->WriteLine("0");
                  // Закрити файл
                  swReaderFile->Close();
                  // Закрити файловий потік
                  fsReaderFile->Close();
                  // Закінчити виконання функції, але виконати вхід
                  return true;
            }
            else
            {
                  // Закінчити виконання функції й не виконувати вхід
                  return false;
            }
      }
      // Відкрити файл користувача
```

КОПІЯ - 2024

```
	StreamReader^ srReaderFile = gcnew StreamReader(strFileName);
	// Завантажити кількість документів
	Int32 ddDocsNumber = Convert::ToInt32(srReaderFile->ReadLine());
	// Цикл зчитування інформації про документи
	for(Int32 i = 0; i < ddDocsNumber; i++)
	{
		// Зчитування вхідний номер документа з файлу
		Int32 ddISBNHash = Convert::ToInt32(srReaderFile->ReadLine());

		// Одержання інформації й додавання документи в список документів
користувача
		listMyDoc->Add(Worker->ViewDoc(ddISBNHash));
	}

	// Кінець файлу не досягнуть?
	if(!srReaderFile->EndOfStream)
	{
		// Завантажити історію одержань і повернень документів
		logUsing->strLog = srReaderFile->ReadToEnd();
	}
	else
	{
		// Обнулити рядок історії
		logUsing->strLog = "";
	}

	// Закрити файл користувача
	srReaderFile->Close();

	// Закінчити виконання функції й виконати вхід
	return true;
}
/*************************************************************************/
// NAME:          Save
// DESCRIPTION:   Функція збереження списку документів користувача у файл
// INPUT:         N/D
/*************************************************************************/
void CReader::Save()
{
	// Перевірка існування файлу користувача
	if(File::Exists(strFileName))
	{
		// Видалення старого файлу
		File::Delete(strFileName);
	}
	// Відкрити новий файл користувача
	StreamWriter^ swReaderFile = gcnew StreamWriter(strFileName);

	// Записати кількість документів
	Int32 ddDocsNumber = listMyDoc->Count;
	swReaderFile->WriteLine(Convert::ToString(ddDocsNumber));
	// Цикл запису інформації про документи
	for(Int32 i = 0; i < ddDocsNumber; i++)
	{
		// Запис вхідний номер документа у файл
		swReaderFile->WriteLine(((CDoc^)listMyDoc[i])->GetISBNHash());
	}
	// Записати лог
	swReaderFile->WriteLine(logUsing->strLog);
	// Закрити файл користувача
	swReaderFile->Close();
}
/*************************************************************************/
// NAME:          GetMyDocsList
// DESCRIPTION:   Функція одержання списку документів користувача
// INPUT:         N/D
/*************************************************************************/
ArrayList^ CReader::GetMyDocsList()
{
```

КОПЗ – 2024

```
	return listMyDoc;
}
/*****************************************************************************/
// NAME:		RequireDoc
// DESCRIPTION:	Функція запиту заданого документа
// INPUT:		Worker - об'єкт "Адміністратор", через який здійснюється
одержання
//			ddHash - хеш-значення вхідного номера документа, що
потрібно одержати
/*****************************************************************************/
Boolean CReader::RequireDoc(CWorker^ Worker, Int32 ddHash)
{
	// Запит на одержання документи до "адміністратора"
	Boolean fResult = Worker->GiveDoc(listMyDoc, ddHash);
	// Результат одержання документів позитивний?
	if(fResult)
	{
		// Індекс останньої документи
		Int32 ddIndex = listMyDoc->Count - 1;
		// Записати операцію в лог
		logUsing->WriteEvent("Отриманий документ '" +
((CDoc^)listMyDoc[ddIndex])->GetName() +
			"', Автор '" + ((CDoc^)listMyDoc[ddIndex])->GetAuthor() +
			"', що входить номер документа " +
((CDoc^)listMyDoc[ddIndex])->GetISBN());

		// Операція пройшла успішно
		return true;
	}
	else
	{
		// Такого документа в наявності не є
		return false;
	}
}
/*****************************************************************************/
// NAME:		ReleaseDoc
// DESCRIPTION:	Функція повернення заданого документа
// INPUT:		Worker - об'єкт "Адміністратор", через який здійснюється
повернення
//			ddIndex - індекс документа в списку користувача
/*****************************************************************************/
void CReader::ReleaseDoc(CWorker^ Worker, Int32 ddIndex)
{
	// Повернути документ
	Worker->TakeDoc(listMyDoc, ((CDoc^)listMyDoc[ddIndex])->GetISBNHash());
	// Записати операцію в лог
	logUsing->WriteEvent(
		"Повернутий документ '" + ((CDoc^)listMyDoc[ddIndex])->GetName() +
		"', Автор '" + ((CDoc^)listMyDoc[ddIndex])->GetName() +
		"', що входить номер документа " + ((CDoc^)listMyDoc[ddIndex])-
>GetISBN());
	// Видалити документ зі списку користувача
	listMyDoc->RemoveAt(ddIndex);
}
/*****************************************************************************/
// NAME:		ViewLog
// DESCRIPTION:	Функція перегляду історії
// INPUT:		N/D
/*****************************************************************************/
String^ CReader::ViewLog()
{
	return logUsing->strLog;
}
/*****************************************************************************/
```

КБТЗ_2024

Файл Reader.resx - xml опис вікна користувача

```
 <?xml version="1.0" encoding=" utf-8"?>
<root>
  <xsd:schema id="root" xmlns="" xmlns:xsd=" " xmlns:msdata="urn: schemas-
microsoft-com: xml-msdata">
    <xsd:import namespace=" " />
    <xsd:element name="root" msdata:IsDataSet="true">
      <xsd:complexType>
        <xsd:choice maxOccurs="unbounded">
          <xsd:element name="metadata">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0" />
              </xsd:sequence>
              <xsd:attribute name="name" use="required" type="xsd:string" />
              <xsd:attribute name="type" type="xsd:string" />
              <xsd:attribute name="mimetype" type="xsd:string" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="assembly">
            <xsd:complexType>
              <xsd:attribute name="alias" type="xsd:string" />
              <xsd:attribute name="name" type="xsd:string" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="data">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
                <xsd:element name="comment" type="xsd:string" minOccurs="0"
msdata:Ordinal="2" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required"
msdata:Ordinal="1" />
              <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
              <xsd:attribute name="mimetype" type="xsd:string"
msdata:Ordinal="4" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="resheader">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required" />
            </xsd:complexType>
          </xsd:element>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
  <resheader name="resmimetype">
    <value>text/ microsoft-resx</value>
  </resheader>
  <resheader name="version">
    <value>2.0</value>
  </resheader>
  <resheader name="reader">
    <value>System.Resources.ResXResourceReader, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <resheader name="writer">
```

КПЗ_2024

```
    <value>System.Resources.ResXResourceWriter, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <metadata name="statusStrip.TrayLocation" type="System.Drawing.Point,
System.Drawing, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a">
    <value>436, 17</value>
  </metadata>
  <metadata name="menuStrip.TrayLocation" type="System.Drawing.Point,
System.Drawing, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b03f5f7f11d50a3a">
    <value>540, 17</value>
  </metadata>
</root>
```

КбПЗ_2024

Файл Reader.h - бібліотека для файлу frmAbout.cpp

```
#pragma once
/*************************************************************************/
#include "Worker.h"
#include "Doc.h"
#include "Reader.h"

#include "frmAbout.h"
/*************************************************************************/
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;
/*************************************************************************/
namespace UsrsManager
{
	public ref class frmReader : public System::Windows::Forms::Form
	{
/*************************************************************************/
	private:
	CWorker^ Worker;		// Об'єкт класу "Адміністратор"
	CReader^ Reader;		// Об'єкт класу "Користувач"

	ArrayList^ listView;	// Поточний відображуваний список документів

	Int32 ddCurrentIndex;	// Індекс поточного відображуваного елемента

	String^ strUserName;	// Поточне ім'я користувача
/*************************************************************************/
	public:
	frmReader(String^ strReaderName)
	{
		// Ініціалізація елементів керування на формі
		InitializeComponent();

		// Створити список відображуваних документів
		listView = gcnew ArrayList();

		// Поточний індекс не визначений
		ddCurrentIndex = -1;

		// Зберегти ім'я користувача
		strUserName = strReaderName;
	}
/*************************************************************************/
protected: ~frmReader()
	{
		if (components)
		{
			delete components;
		}
	}
/*************************************************************************/
	private: System::Windows::Forms::Button^  btnRequest;
	private: System::Windows::Forms::ToolStripStatusLabel^  lblStatus;
	private: System::Windows::Forms::StatusStrip^  statusStrip;
	private: System::Windows::Forms::TabPage^  tabFinding;
	private: System::Windows::Forms::GroupBox^  groupBox1;
	private: System::Windows::Forms::Label^  lblPagesNumber;
	private: System::Windows::Forms::Label^  lblTheme;
	private: System::Windows::Forms::Label^  lblISBN;
	private: System::Windows::Forms::Label^  lblAuthor;
	private: System::Windows::Forms::Label^  lblDocName;
	private: System::Windows::Forms::GroupBox^  groupFinding;
```

КПЗ_2024

```
        private: System::Windows::Forms::Button^  btnFind;
        private: System::Windows::Forms::TextBox^  txtFindString;
        private: System::Windows::Forms::TabControl^  tabControl;
        private: System::Windows::Forms::TabPage^  tabMyDocs;
        private: System::Windows::Forms::Button^  btnReturn;
        private: System::Windows::Forms::GroupBox^  groupBox2;
        private: System::Windows::Forms::ListBox^  lstMyDocs;
        private: System::Windows::Forms::TabPage^  tabHistory;
        private: System::Windows::Forms::GroupBox^  groupBox3;
        private: System::Windows::Forms::TextBox^  txtHistory;
        private: System::Windows::Forms::ToolStripMenuItem^  menuHelp;

        private: System::Windows::Forms::ToolStripMenuItem^  menuHelpAbout;
        private: System::Windows::Forms::ToolStripMenuItem^  menuFileExit;
        private: System::Windows::Forms::MenuStrip^  menuStrip;
        private: System::Windows::Forms::ToolStripMenuItem^  menuFile;
        private: System::Windows::Forms::ToolStripMenuItem^  menuFileSave;
        private: System::Windows::Forms::TextBox^  txtPosition;
        private: System::Windows::Forms::Button^  btnFirst;
        private: System::Windows::Forms::Button^  btnPrev;
        private: System::Windows::Forms::Button^  btnNext;
        private: System::Windows::Forms::Button^  btnLast;
        private: System::Windows::Forms::TextBox^  txtPagesNumber;
        private: System::Windows::Forms::TextBox^  txtTheme;
        private: System::Windows::Forms::TextBox^  txtISBN;
        private: System::Windows::Forms::TextBox^  txtAuthor;
        private: System::Windows::Forms::TextBox^  txtDocName;

        private:
        /// <summary>
        /// Required designer variable.
        /// </summary>
        System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
        /// <summary>
        /// Необхідний метод для підтримки Розроблювача - не модифікувати
        /// зміст цього методу з кодовим редактором.
        /// </summary>
        void InitializeComponent(void)
        {
                this->btnRequest = (gcnew System::Windows::Forms::Button());
                this->lblStatus = (gcnew
System::Windows::Forms::ToolStripStatusLabel());
                this->statusStrip = (gcnew System::Windows::Forms::StatusStrip());
                this->tabFinding = (gcnew System::Windows::Forms::TabPage());
                this->groupBox1 = (gcnew System::Windows::Forms::GroupBox());
                this->txtPosition = (gcnew System::Windows::Forms::TextBox());
                this->btnFirst = (gcnew System::Windows::Forms::Button());
                this->btnPrev = (gcnew System::Windows::Forms::Button());
                this->btnNext = (gcnew System::Windows::Forms::Button());
                this->btnLast = (gcnew System::Windows::Forms::Button());
                this->txtPagesNumber = (gcnew System::Windows::Forms::TextBox());
                this->txtTheme = (gcnew System::Windows::Forms::TextBox());
                this->txtISBN = (gcnew System::Windows::Forms::TextBox());
                this->txtAuthor = (gcnew System::Windows::Forms::TextBox());
                this->txtDocName = (gcnew System::Windows::Forms::TextBox());
                this->lblPagesNumber = (gcnew System::Windows::Forms::Label());
                this->lblTheme = (gcnew System::Windows::Forms::Label());
                this->lblISBN = (gcnew System::Windows::Forms::Label());
                this->lblAuthor = (gcnew System::Windows::Forms::Label());
                this->lblDocName = (gcnew System::Windows::Forms::Label());
                this->groupFinding = (gcnew System::Windows::Forms::GroupBox());
                this->btnFind = (gcnew System::Windows::Forms::Button());
                this->txtFindString = (gcnew System::Windows::Forms::TextBox());
                this->tabControl = (gcnew System::Windows::Forms::TabControl());
                this->tabMyDocs = (gcnew System::Windows::Forms::TabPage());
                this->btnReturn = (gcnew System::Windows::Forms::Button());
                this->groupBox2 = (gcnew System::Windows::Forms::GroupBox());
```

КОПІЯ 2024

```
            this->lstMyDocs = (gcnew System::Windows::Forms::ListBox());
            this->tabHistory = (gcnew System::Windows::Forms::TabPage());
            this->groupBox3 = (gcnew System::Windows::Forms::GroupBox());
            this->txtHistory = (gcnew System::Windows::Forms::TextBox());
            this->menuHelp = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuHelpAbout = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuFileExit = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuStrip = (gcnew System::Windows::Forms::MenuStrip());
            this->menuFile = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->menuFileSave = (gcnew
System::Windows::Forms::ToolStripMenuItem());
            this->statusStrip->SuspendLayout();
            this->tabFinding->SuspendLayout();
            this->groupBox 1-1->SuspendLayout();
            this->groupFinding->SuspendLayout();
            this->tabControl->SuspendLayout();
            this->tabMyDocs->SuspendLayout();
            this->groupBox 2-2->SuspendLayout();
            this->tabHistory->SuspendLayout();
            this->groupBox 3-3->SuspendLayout();
            this->menuStrip->SuspendLayout();
            this->SuspendLayout();
            //
            // btnRequest
            //
            this->btnRequest->Location = System::Drawing::Point(567, 285);
            this->btnRequest->Name = L"btnRequest";
            this->btnRequest->Size = System::Drawing::Size(144, 23);
            this->btnRequest->TabIndex = 12;
            this->btnRequest->Text = L"Здійснити запит";
            this->btnRequest->UseVisualStyleBackColor = true;
            this->btnRequest->Click += gcnew System::EventHandler(this,
&frmReader::btnRequest_Click);
            //
            // lblStatus
            //
            this->lblStatus->Name = L"lblStatus";
            this->lblStatus->Size = System::Drawing::Size(109, 17);
            this->lblStatus->Text = L"toolStripStatusLabel1";
            this->lblStatus->Click += gcnew System::EventHandler(this,
&frmReader::lblStatus_Click);
            //
            // statusStrip
            //
            this->statusStrip->Items->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(1) {this->lblStatus});
            this->statusStrip->Location = System::Drawing::Point(0, 365);
            this->statusStrip->Name = L"statusStrip";
            this->statusStrip->Size = System::Drawing::Size(724, 22);
            this->statusStrip->TabIndex = 13;
            this->statusStrip->Text = L"statusStrip1";
            //
            // tabFinding
            //
            this->tabFinding->Controls->Add(this->btnRequest);
            this->tabFinding->Controls->Add(this->groupBox1);
            this->tabFinding->Controls->Add(this->groupFinding);
            this->tabFinding->Location = System::Drawing::Point(4, 22);
            this->tabFinding->Name = L"tabFinding";
            this->tabFinding->Padding = System::Windows::Forms::Padding(3);
            this->tabFinding->Size = System::Drawing::Size(717, 314);
            this->tabFinding->TabIndex = 0;
            this->tabFinding->Text = L"Пошук";
            this->tabFinding->UseVisualStyleBackColor = true;
            //
```

КПЗ 2024

```
            // groupBox1
            //
            this->groupBox 1-1->Controls->Add(this->txtPosition);
            this->groupBox 1-1->Controls->Add(this->btnFirst);
            this->groupBox 1-1->Controls->Add(this->btnPrev);
            this->groupBox 1-1->Controls->Add(this->btnNext);
            this->groupBox 1-1->Controls->Add(this->btnLast);
            this->groupBox 1-1->Controls->Add(this->txtPagesNumber);
            this->groupBox 1-1->Controls->Add(this->txtTheme);
            this->groupBox 1-1->Controls->Add(this->txtISBN);
            this->groupBox 1-1->Controls->Add(this->txtAuthor);
            this->groupBox 1-1->Controls->Add(this->txtDocName);
            this->groupBox 1-1->Controls->Add(this->lblPagesNumber);
            this->groupBox 1-1->Controls->Add(this->lblTheme);
            this->groupBox 1-1->Controls->Add(this->lblISBN);
            this->groupBox 1-1->Controls->Add(this->lblAuthor);
            this->groupBox 1-1->Controls->Add(this->lblDocName);
            this->groupBox 1-1->Location = System::Drawing::Point(6, 65);
            this->groupBox 1-1->Name = L"groupBox1";
            this->groupBox 1-1->Size = System::Drawing::Size(702, 214);
            this->groupBox 1-1->TabIndex = 9;
            this->groupBox 1-1->TabStop = false;
            this->groupBox 1-1->Text = L"Інформація про документ";
            //
            // txtPosition
            //
            this->txtPosition->Location = System::Drawing::Point(568, 187);
            this->txtPosition->Name = L"txtPosition";
            this->txtPosition->Size = System::Drawing::Size(58, 20);
            this->txtPosition->TabIndex = 9;
            this->txtPosition->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
            this->txtPosition->KeyDown += gcnew
System::Windows::Forms::KeyEventHandler(this, &frmReader::txtPosition_KeyDown);
            //
            // btnFirst
            //
            this->btnFirst->Location = System::Drawing::Point(494, 187);
            this->btnFirst->Name = L"btnFirst";
            this->btnFirst->Size = System::Drawing::Size(31, 20);
            this->btnFirst->TabIndex = 7;
            this->btnFirst->Text = L"|<<";
            this->btnFirst->UseVisualStyleBackColor = true;
            this->btnFirst->Click += gcnew System::EventHandler(this,
&frmReader::btnFirst_Click);
            //
            // btnPrev
            //
            this->btnPrev->Location = System::Drawing::Point(531, 187);
            this->btnPrev->Name = L"btnPrev";
            this->btnPrev->Size = System::Drawing::Size(31, 20);
            this->btnPrev->TabIndex = 8;
            this->btnPrev->Text = L"<<";
            this->btnPrev->UseVisualStyleBackColor = true;
            this->btnPrev->Click += gcnew System::EventHandler(this,
&frmReader::btnPrev_Click);
            //
            // btnNext
            //
            this->btnNext->Location = System::Drawing::Point(632, 186);
            this->btnNext->Name = L"btnNext";
            this->btnNext->Size = System::Drawing::Size(31, 20);
            this->btnNext->TabIndex = 10;
            this->btnNext->Text = L">>";
            this->btnNext->UseVisualStyleBackColor = true;
            this->btnNext->Click += gcnew System::EventHandler(this,
&frmReader::btnNext_Click);
            //
            // btnLast
```

КПЗ_2024

```
            //
            this->btnLast->Location = System::Drawing::Point(669, 186);
            this->btnLast->Name = L"btnLast";
            this->btnLast->Size = System::Drawing::Size(31, 20);
            this->btnLast->TabIndex = 11;
            this->btnLast->Text = L">>|";
            this->btnLast->UseVisualStyleBackColor = true;
            this->btnLast->Click += gcnew System::EventHandler(this,
&frmReader::btnLast_Click);
            //
            // txtPagesNumber
            //
            this->txtPagesNumber->Location = System::Drawing::Point(139, 161);
            this->txtPagesNumber->Name = L"txtPagesNumber";
            this->txtPagesNumber->ReadOnly = true;
            this->txtPagesNumber->Size = System::Drawing::Size(48, 20);
            this->txtPagesNumber->TabIndex = 6;
            //
            // txtTheme
            //
            this->txtTheme->Location = System::Drawing::Point(139, 128);
            this->txtTheme->Name = L"txtTheme";
            this->txtTheme->ReadOnly = true;
            this->txtTheme->Size = System::Drawing::Size(302, 20);
            this->txtTheme->TabIndex = 5;
            //
            // txtISBN
            //
            this->txtISBN->Location = System::Drawing::Point(139, 94);
            this->txtISBN->Name = L"txtISBN";
            this->txtISBN->ReadOnly = true;
            this->txtISBN->Size = System::Drawing::Size(163, 20);
            this->txtISBN->TabIndex = 4;
            //
            // txtAuthor
            //
            this->txtAuthor->Location = System::Drawing::Point(139, 60);
            this->txtAuthor->Name = L"txtAuthor";
            this->txtAuthor->ReadOnly = true;
            this->txtAuthor->Size = System::Drawing::Size(505, 20);
            this->txtAuthor->TabIndex = 3;
            //
            // txtDocName
            //
            this->txtDocName->Location = System::Drawing::Point(139, 26);
            this->txtDocName->Name = L"txtDocName";
            this->txtDocName->ReadOnly = true;
            this->txtDocName->Size = System::Drawing::Size(557, 20);
            this->txtDocName->TabIndex = 2;
            //
            // lblPagesNumber
            //
            this->lblPagesNumber->AutoSize = true;
            this->lblPagesNumber->Location = System::Drawing::Point(40, 164);
            this->lblPagesNumber->Name = L"lblPagesNumber";
            this->lblPagesNumber->Size = System::Drawing::Size(99, 13);
            this->lblPagesNumber->TabIndex = 4;
            this->lblPagesNumber->Text = L"Кількість сторінок";
            //
            // lblTheme
            //
            this->lblTheme->AutoSize = true;
            this->lblTheme->Location = System::Drawing::Point(108, 128);
            this->lblTheme->Name = L"lblTheme";
            this->lblTheme->Size = System::Drawing::Size(31, 13);
            this->lblTheme->TabIndex = 3;
            this->lblTheme->Text = L"Тема";
            //
            // lblISBN
```

КОПІЯ 2024

```
            //
            this->lblISBN->AutoSize = true;
            this->lblISBN->Location = System::Drawing::Point(2, 94);
            this->lblISBN->Name = L"lblISBN";
            this->lblISBN->Size = System::Drawing::Size(137, 13);
            this->lblISBN->TabIndex = 2;
            this->lblISBN->Text = L"Вхідний номер документу";
            //
            // lblAuthor
            //
            this->lblAuthor->AutoSize = true;
            this->lblAuthor->Location = System::Drawing::Point(101, 60);
            this->lblAuthor->Name = L"lblAuthor";
            this->lblAuthor->Size = System::Drawing::Size(38, 13);
            this->lblAuthor->TabIndex = 1;
            this->lblAuthor->Text = L"Автор";
            //
            // lblDocName
            //
            this->lblDocName->AutoSize = true;
            this->lblDocName->Location = System::Drawing::Point(44, 26);
            this->lblDocName->Name = L"lblDocName";
            this->lblDocName->Size = System::Drawing::Size(95, 13);
            this->lblDocName->TabIndex = 0;
            this->lblDocName->Text = L"Назва документу";
            //
            // groupFinding
            //
            this->groupFinding->Controls->Add(this->btnFind);
            this->groupFinding->Controls->Add(this->txtFindString);
            this->groupFinding->Location = System::Drawing::Point(6, 6);
            this->groupFinding->Name = L"groupFinding";
            this->groupFinding->Size = System::Drawing::Size(702, 53);
            this->groupFinding->TabIndex = 8;
            this->groupFinding->TabStop = false;
            this->groupFinding->Text = L"Ключові рядки для пошуку";
            //
            // btnFind
            //
            this->btnFind->Location = System::Drawing::Point(621, 19);
            this->btnFind->Name = L"btnFind";
            this->btnFind->Size = System::Drawing::Size(75, 23);
            this->btnFind->TabIndex = 1;
            this->btnFind->Text = L"Знайти";
            this->btnFind->UseVisualStyleBackColor = true;
            this->btnFind->Click += gcnew System::EventHandler(this,
&frmReader::btnFind_Click);
            //
            // txtFindString
            //
            this->txtFindString->Location = System::Drawing::Point(6, 19);
            this->txtFindString->Name = L"txtFindString";
            this->txtFindString->Size = System::Drawing::Size(609, 20);
            this->txtFindString->TabIndex = 0;
            //
            // tabControl
            //
            this->tabControl->Controls->Add(this->tabFinding);
            this->tabControl->Controls->Add(this->tabMyDocs);
            this->tabControl->Controls->Add(this->tabHistory);
            this->tabControl->Location = System::Drawing::Point(0, 27);
            this->tabControl->Name = L"tabControl";
            this->tabControl->SelectedIndex = 0;
            this->tabControl->Size = System::Drawing::Size(725, 340);
            this->tabControl->TabIndex = 14;
            //
            // tabMyDocs
            //
            this->tabMyDocs->Controls->Add(this->btnReturn);
```

КПЗ 2024

```
            this->tabMyDocs->Controls->Add(this->groupBox2);
            this->tabMyDocs->Location = System::Drawing::Point(4, 22);
            this->tabMyDocs->Name = L"tabMyDocs";
            this->tabMyDocs->Padding = System::Windows::Forms::Padding(3);
            this->tabMyDocs->Size = System::Drawing::Size(717, 314);
            this->tabMyDocs->TabIndex = 1;
            this->tabMyDocs->Text = L"Документи";
            this->tabMyDocs->UseVisualStyleBackColor = true;
            //
            // btnReturn
            //
            this->btnReturn->Location = System::Drawing::Point(622, 285);
            this->btnReturn->Name = L"btnReturn";
            this->btnReturn->Size = System::Drawing::Size(85, 23);
            this->btnReturn->TabIndex = 1;
            this->btnReturn->Text = L"Повернути";
            this->btnReturn->UseVisualStyleBackColor = true;
            this->btnReturn->Click += gcnew System::EventHandler(this,
&frmReader::btnReturn_Click);
            //
            // groupBox2
            //
            this->groupBox 2-2->Controls->Add(this->lstMyDocs);
            this->groupBox 2-2->Location = System::Drawing::Point(8, 6);
            this->groupBox 2-2->Name = L"groupBox2";
            this->groupBox 2-2->Size = System::Drawing::Size(699, 273);
            this->groupBox 2-2->TabIndex = 0;
            this->groupBox 2-2->TabStop = false;
            this->groupBox 2-2->Text = L"Список документів користувача";
            this->groupBox 2-2->Enter += gcnew System::EventHandler(this,
&frmReader::groupBox2_Enter);
            //
            // lstMyDocs
            //
            this->lstMyDocs->FormattingEnabled = true;
            this->lstMyDocs->HorizontalScrollbar = true;
            this->lstMyDocs->Location = System::Drawing::Point(6, 19);
            this->lstMyDocs->Name = L"lstMyDocs";
            this->lstMyDocs->Size = System::Drawing::Size(687, 251);
            this->lstMyDocs->TabIndex = 0;
            //
            // tabHistory
            //
            this->tabHistory->Controls->Add(this->groupBox3);
            this->tabHistory->Location = System::Drawing::Point(4, 22);
            this->tabHistory->Name = L"tabHistory";
            this->tabHistory->Size = System::Drawing::Size(717, 314);
            this->tabHistory->TabIndex = 2;
            this->tabHistory->Text = L"Історія";
            this->tabHistory->UseVisualStyleBackColor = true;
            //
            // groupBox3
            //
            this->groupBox 3-3->Controls->Add(this->txtHistory);
            this->groupBox 3-3->Location = System::Drawing::Point(8, 8);
            this->groupBox 3-3->Name = L"groupBox3";
            this->groupBox 3-3->Size = System::Drawing::Size(700, 303);
            this->groupBox 3-3->TabIndex = 2;
            this->groupBox 3-3->TabStop = false;
            this->groupBox 3-3->Text = L"Історія перегляду документів";
            //
            // txtHistory
            //
            this->txtHistory->Location = System::Drawing::Point(6, 19);
            this->txtHistory->Multiline = true;
            this->txtHistory->Name = L"txtHistory";
            this->txtHistory->ReadOnly = true;
            this->txtHistory->ScrollBars =
System::Windows::Forms::ScrollBars::Both;
```

КОПІЯ 2024

```
            this->txtHistory->Size = System::Drawing::Size(688, 278);
            this->txtHistory->TabIndex = 0;
            //
            // menuHelp
            //
            this->menuHelp->DropDownItems->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(1) {this->menuHelpAbout});
            this->menuHelp->Name = L"menuHelp";
            this->menuHelp->Size = System::Drawing::Size(60, 20);
            this->menuHelp->Text = L"Довідка";
            //
            // menuHelpAbout
            //
            this->menuHelpAbout->Name = L"menuHelpAbout";
            this->menuHelpAbout->Size = System::Drawing::Size(166, 22);
            this->menuHelpAbout->Text = L"Про програму...";
            this->menuHelpAbout->Click += gcnew System::EventHandler(this,
&frmReader::menuHelpAbout_Click);
            //
            // menuFileExit
            //
            this->menuFileExit->Name = L"menuFileExit";
            this->menuFileExit->Size = System::Drawing::Size(152, 22);
            this->menuFileExit->Text = L"Вихід";
            this->menuFileExit->Click += gcnew System::EventHandler(this,
&frmReader::menuFileExit_Click);
            //
            // menuStrip
            //
            this->menuStrip->Items->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(2) {this->menuFile, this->menuHelp});
            this->menuStrip->Location = System::Drawing::Point(0, 0);
            this->menuStrip->Name = L"menuStrip";
            this->menuStrip->Size = System::Drawing::Size(724, 24);
            this->menuStrip->TabIndex = 12;
            this->menuStrip->Text = L"menuStrip1";
            //
            // menuFile
            //
            this->menuFile->DropDownItems->AddRange(gcnew cli::array<
System::Windows::Forms::ToolStripItem^  >(2) {this->menuFileSave,
                this->menuFileExit});
            this->menuFile->Name = L"menuFile";
            this->menuFile->Size = System::Drawing::Size(45, 20);
            this->menuFile->Text = L"Файл";
            //
            // menuFileSave
            //
            this->menuFileSave->Name = L"menuFileSave";
            this->menuFileSave->Size = System::Drawing::Size(152, 22);
            this->menuFileSave->Text = L"Зберегти";
            this->menuFileSave->Click += gcnew System::EventHandler(this,
&frmReader::menuFileSave_Click);
            //
            // frmReader
            //
            this->AutoScaleDimensions = System::Drawing::Size(6, 13);
            this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;
            this->ClientSize = System::Drawing::Size(724, 387);
            this->Controls->Add(this->statusStrip);
            this->Controls->Add(this->tabControl);
            this->Controls->Add(this->menuStrip);
            this->MaximizeBox = false;
            this->MinimizeBox = false;
            this->Name = L"frmReader";
            this->Text = L"Режим  користувача - Віртуальний офіс для потреб
виробництва ";
            this->Load += gcnew System::EventHandler(this,
&frmReader::frmReader_Load);
```

КПЗ 2024

```
            this->FormClosed += gcnew
System::Windows::Forms::FormClosedEventHandler(this,
&frmReader::frmReader_FormClosed);
            this->statusStrip->ResumeLayout(false);
            this->statusStrip->PerformLayout();
            this->tabFinding->ResumeLayout(false);
            this->groupBox 1-1->ResumeLayout(false);
            this->groupBox 1-1->PerformLayout();
            this->groupFinding->ResumeLayout(false);
            this->groupFinding->PerformLayout();
            this->tabControl->ResumeLayout(false);
            this->tabMyDocs->ResumeLayout(false);
            this->groupBox 2-2->ResumeLayout(false);
            this->tabHistory->ResumeLayout(false);
            this->groupBox 3-3->ResumeLayout(false);
            this->groupBox 3-3->PerformLayout();
            this->menuStrip->ResumeLayout(false);
            this->menuStrip->PerformLayout();
            this->ResumeLayout(false);
            this->PerformLayout();

      }
#pragma endregion
 /*************************************************************************/
 private: System::Void UpdateView()
       {
             // Список не порожній?
             if(listView->Count != 0)
             {
                   // Перевірити індекс поточного елемента
                   if(ddCurrentIndex >= listView->Count)
                   {
                         // Установити покажчик на останній елемент
                         ddCurrentIndex = listView->Count - 1;
                   }

                   // Поточний індекс менше нуля?
                   if(ddCurrentIndex < 0)
                   {
                         // Установити індекс на перший елемент
                         ddCurrentIndex = 0;
                   }

                   // Одержати характеристики документа й заповнити поля форми
                   txtDocName->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetName();
                   txtAuthor->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetAuthor();
                   txtISBN->Text = ((CDoc^)listView[ddCurrentIndex])->GetISBN();
                   txtTheme->Text = ((CDoc^)listView[ddCurrentIndex])-
>GetTheme();
                   txtPagesNumber->Text =
Convert::ToString(((CDoc^)listView[ddCurrentIndex])->GetPages());

                   // Документ є в наявності?
                   if(((CDoc^)listView[ddCurrentIndex])->GetFreeNumber() > 0)
                   {
                         // Розблокувати кнопку запросу
                         btnRequest->Enabled = true;
                   }
                   else
                   {
                         // Заблокувати кнопку запросу
                         btnRequest->Enabled = false;
                   }

                   // Установити й вивести поточну позицію
                   txtPosition->Text = Convert::ToString(ddCurrentIndex + 1) + "
/ " +
```

КБПЗ-2024

```
                        Convert::ToString(listView->Count);
            }
            else
            {
                  // Очистити всі поля
                  txtDocName->Text = "";
                  txtAuthor->Text = "";
                  txtISBN->Text = "";
                  txtTheme->Text = "";
                  txtPagesNumber->Text = "";
                  // Поточна позиція - 0
                  txtPosition->Text = "0/0";
                  // Установити індекс поточного елемента
                  ddCurrentIndex = 0;
                  // Вивести стан
                  lblStatus->Text = "Немає записів для відображення";
            }

            // Вивести історію
            txtHistory->Text = Reader->ViewLog();

            // Одержати список документів користувача
            ArrayList^ listTemp = gcnew ArrayList();
            listTemp = Reader->GetMyDocsList();

            // Очистити список на формі
            lstMyDocs->Items->Clear();
            // Кількість документів не дорівнює нулю?
            if(listTemp->Count != 0)
            {
                  // Заповнити список документів на формі
                  for(Int32 i = 0; i < listTemp->Count; i++)
                  {
                        // Одержати характеристики документа й вивести їх у
список
                        lstMyDocs->Items->Add(Convert::ToString(i + 1) + ". '"
+
                              ((CDoc^)listTemp[i])->GetName() + "', '" +
                              ((CDoc^)listTemp[i])->GetAuthor() + "', вхідний
номер документа " +
                              ((CDoc^)listTemp[i])->GetISBN());
                  }
            }
      }
/*********************************************************************/
private: System::Void frmReader_Load(System::Object^  sender, System::EventArgs^
e)
      {
            // Створити об'єкт "Адміністратор"
            Worker = gcnew CWorker();
            // Завантажити список документів
            Worker->LoadDocList();

            // Створити об'єкт класу "Користувач"
            Reader = gcnew CReader(strUserName);
            // Завантажити список документів, що перебувають у користувача
            Reader->Load(Worker);

           // Вивести стан
           lblStatus->Text = "Для відображення списку потрібних документів
скористайтеся пошуком";
            // Обновити форму
           UpdateView();
      }
 /*********************************************************************/
private: System::Void btnFind_Click(System::Object^  sender, System::EventArgs^
e)
      {
```

КбПЗ_2024

```
            // Забрати з рядку пошуку пробіли, перетворити до нижнього регістра
й зберегти
            String^ strValue = txtFindString->Text->Trim()->ToLower();

            // Перевірка рядка
            if(String::IsNullOrEmpty(strValue))
            {
                  // Рядок порожня, запропонувати вивести весь список
                  if(MessageBox::Show("Рядок пошуку не завдань, вивести весь
список?",
                                                "Система УЕД",
MessageBoxButtons::YesNo,
                                                MessageBoxIcon::Question) ==
::DialogResult::Yes)
                  {
                        // Перейти на вивід списку
                        goto OUT_LIST;
                  }
                  // Завершити роботу функції
                  return;
            }

OUT_LIST:   // Виконати функцію пошуку
            listView = Worker->FindDoc(strValue);

            // Вивести стан
            lblStatus->Text = "Пошук завершено";
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnFirst_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на перший елемент
            ddCurrentIndex = 0;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnPrev_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на попередній елемент
            ddCurrentIndex ---i;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnNext_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на наступний елемент
            ddCurrentIndex ++;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void btnLast_Click(System::Object^  sender, System::EventArgs^
e)
      {
            // Установити індекс на останній елемент
            ddCurrentIndex = listView->Count - 1;
            // Обновити форму
            UpdateView();
      }
/*************************************************************************/
private: System::Void txtPosition_KeyDown(System::Object^  sender,
System::Windows::Forms::KeyEventArgs^  e)
```

КбПЗ_2024

```
	{
		// Натиснута клавіша "Enter"?
		if( e->KeyCode == Keys::Enter)
		{
			try
			{
				// Спробувати одержати введений номер документа для
перегляду
				ddCurrentIndex = Convert::ToInt32(txtPosition->Text) -
1;
			}
			catch(...)
			{
			// Якщо в процесі перетворення номера з рядка в число
			//  виникла помилка, те ніяк на це не реагувати
			}

			// Обновити форму
			UpdateView();
		}
		else
		{
			// Текстове поле містить символ "/"?
			if(txtPosition->Text->IndexOf("/") != -1)
			{
				// Для початку введення нового номера позиції,
необхідно
				//  очистити текстове поле
				txtPosition->Text = "";
			}
		}
	}
/************************************************************************/
private: System::Void btnRequest_Click(System::Object^  sender,
System::EventArgs^  e)
	{
		// Запросити документ, попередньо перетворивши значення вхідного
номера документа в текстовому
		//  поле до нижнього регістра й забравши бічні пробіли
		if(Reader->RequireDoc(Worker, txtISBN->Text->ToLower()->Trim()-
>GetHashCode()))
		{
			// Запит пройшов вдало, видалити документ із поточного списку
перегляду
			listView->RemoveAt(ddCurrentIndex);

			// Обновити форму
			UpdateView();
			// Вивести стан
			lblStatus->Text = "Документ успішно отримано";
		}
		else
		{
			// Неможливо одержати документ, вивести повідомлення про
цьому
			lblStatus->Text = "Вибачте, але на даний момент немає жодного
документу по даному запиту";
		}
	}
/************************************************************************/
private: System::Void btnReturn_Click(System::Object^  sender,
System::EventArgs^  e)
	{
		// У списку документів є обраний елемент?
		if(lstMyDocs->SelectedIndex != -1)
		{
			// Повернути обрану документ
			Reader->ReleaseDoc(Worker, lstMyDocs->SelectedIndex);
```

КОПЗ_2024

```
                // Обновити форму
                UpdateView();
                // Вивести стан
                lblStatus->Text = "Документ виконано";
            }
            else
            {
                // Вивести повідомлення про помилку
                lblStatus->Text = "Спочатку виберіть документ для виконання";
            }
        }
/*************************************************************************/
private: System::Void menuFileSave_Click(System::Object^  sender,
System::EventArgs^  e)
        {
             // Збереження модифікованого списку документів у файл
            Worker->SaveDocList();

            // Збереження списку документів користувача
            Reader->Save();
        }
/*************************************************************************/
private: System::Void menuFileExit_Click(System::Object^  sender,
System::EventArgs^  e)
        {
            // Закрити додаток
            Application::Exit();
        }
/*************************************************************************/
private: System::Void frmReader_FormClosed(System::Object^  sender,
System::Windows::Forms::FormClosedEventArgs^  e)
        {
            // Закрити додаток
            Application::Exit();
        }
/*************************************************************************/
private: System::Void menuHelpAbout_Click(System::Object^  sender,
System::EventArgs^  e)
        {
            // Відобразити форму з інформацією про програму
            UsrsManager::frmAbout^ frmNew = gcnew UsrsManager::frmAbout();
            frmNew->Show();
        }
/*************************************************************************/
private: System::Void groupBox2_Enter(System::Object^  sender,
System::EventArgs^  e) {
        }
private: System::Void lblStatus_Click(System::Object^  sender,
System::EventArgs^  e) {
        }
};
}
```

КПЗ - 2024

Файл frmLogin.resx - xml опис форми «Вхід у систему УЕД»

```
<?xml version="1.0" encoding=" utf-8"?>
<root>
    <xsd:schema id="root" xmlns="" xmlns:xsd=" " xmlns:msdata="urn: schemas-
microsoft-com: xml-msdata">
    <xsd:import namespace=" " />
    <xsd:element name="root" msdata:IsDataSet="true">
      <xsd:complexType>
        <xsd:choice maxOccurs="unbounded">
          <xsd:element name="metadata">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0" />
              </xsd:sequence>
              <xsd:attribute name="name" use="required" type="xsd:string" />
              <xsd:attribute name="type" type="xsd:string" />
              <xsd:attribute name="mimetype" type="xsd:string" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="assembly">
            <xsd:complexType>
              <xsd:attribute name="alias" type="xsd:string" />
              <xsd:attribute name="name" type="xsd:string" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="data">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
                <xsd:element name="comment" type="xsd:string" minOccurs="0"
msdata:Ordinal="2" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required"
msdata:Ordinal="1" />
              <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
              <xsd:attribute name="mimetype" type="xsd:string"
msdata:Ordinal="4" />
              <xsd:attribute ref="xml:space" />
            </xsd:complexType>
          </xsd:element>
          <xsd:element name="resheader">
            <xsd:complexType>
              <xsd:sequence>
                <xsd:element name="value" type="xsd:string" minOccurs="0"
msdata:Ordinal="1" />
              </xsd:sequence>
              <xsd:attribute name="name" type="xsd:string" use="required" />
            </xsd:complexType>
          </xsd:element>
        </xsd:choice>
      </xsd:complexType>
    </xsd:element>
  </xsd:schema>
  <resheader name="resmimetype">
    <value>text/ microsoft-resx</value>
  </resheader>
  <resheader name="version">
    <value>2.0</value>
  </resheader>
  <resheader name="reader">
    <value>System.Resources.ResXResourceReader, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
  <resheader name="writer">
    <value>System.Resources.ResXResourceWriter, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
</root>
```

ІПЗ_2024

Файл frmLogin.h - бібліотека для файлу frmLogin.resx

```
/*************************************************************************/
#pragma once
/*************************************************************************/
#include "frmWorker.h"
#include "frmReader.h"
#include "Usrs.h"
/*************************************************************************/
using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;
/*************************************************************************/
#define WORKER_LOGIN    "worker"
/*************************************************************************/
namespace UsrsManager
{
      public ref class frmLogin : public System::Windows::Forms::Form
      {
/*************************************************************************/
      private:
            CUsrs^ Usrs;                  // Керуючий об'єкт класу "Система"
/*************************************************************************/
public:     frmLogin(void)
            {
                  InitializeComponent();
                  //
                  //Додавання цього коду в конструктор
                  //
            }
/*************************************************************************/
protected: ~frmLogin()
            {
                  if (components)
                  {
                        delete components;
                  }
            }
/*************************************************************************/
      private: System::Windows::Forms::GroupBox^  groupBox;
      private: System::Windows::Forms::TextBox^  txtPassword;
      private: System::Windows::Forms::TextBox^  txtLogin;
      private: System::Windows::Forms::Label^  lblPassword;
      private: System::Windows::Forms::Label^  lblLogin;
      private: System::Windows::Forms::Button^  btnEnter;
      private: System::Windows::Forms::Button^  btnExit;
      private: System::Windows::Forms::Button^  btnDelete;
      private: System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
            /// <summary>
            /// Необхідний метод для підтримки Розроблювача - не модифікувати
            /// зміст цього методу з кодовим редактором.
            /// </summary>
            void InitializeComponent(void)
            {
                  this->groupBox = (gcnew System::Windows::Forms::GroupBox());
                  this->txtPassword = (gcnew System::Windows::Forms::TextBox());
                  this->txtLogin = (gcnew System::Windows::Forms::TextBox());
                  this->lblPassword = (gcnew System::Windows::Forms::Label());
                  this->lblLogin = (gcnew System::Windows::Forms::Label());
                  this->btnEnter = (gcnew System::Windows::Forms::Button());
                  this->btnExit = (gcnew System::Windows::Forms::Button());
                  this->btnDelete = (gcnew System::Windows::Forms::Button());
                  this->groupBox->SuspendLayout();
                  this->SuspendLayout();
```

КБГЗ_2024

```
                    //
                    // groupBox
                    //
                    this->groupBox->Controls->Add(this->txtPassword);
                    this->groupBox->Controls->Add(this->txtLogin);
                    this->groupBox->Controls->Add(this->lblPassword);
                    this->groupBox->Controls->Add(this->lblLogin);
                    this->groupBox->Location = System::Drawing::Point(12, 11);
                    this->groupBox->Name = L"groupBox";
                    this->groupBox->Size = System::Drawing::Size(237, 80);
                    this->groupBox->TabIndex = 8;
                    this->groupBox->TabStop = false;
                    this->groupBox->Text = L"Параметри облікового запису";
                    //
                    // txtPassword
                    //
                    this->txtPassword->Location = System::Drawing::Point(57, 45);
                    this->txtPassword->Name = L"txtPassword";
                    this->txtPassword->PasswordChar = '*';
                    this->txtPassword->Size = System::Drawing::Size(174, 20);
                    this->txtPassword->TabIndex = 1;
                    //
                    // txtLogin
                    //
                    this->txtLogin->Location = System::Drawing::Point(57, 19);
                    this->txtLogin->Name = L"txtLogin";
                    this->txtLogin->Size = System::Drawing::Size(174, 20);
                    this->txtLogin->TabIndex = 0;
                    //
                    // lblPassword
                    //
                    this->lblPassword->AutoSize = true;
                    this->lblPassword->Location = System::Drawing::Point(6, 48);
                    this->lblPassword->Name = L"lblPassword";
                    this->lblPassword->Size = System::Drawing::Size(44, 13);
                    this->lblPassword->TabIndex = 1;
                    this->lblPassword->Text = L"Пароль";
                    //
                    // lblLogin
                    //
                    this->lblLogin->AutoSize = true;
                    this->lblLogin->Location = System::Drawing::Point(13, 19);
                    this->lblLogin->Name = L"lblLogin";
                    this->lblLogin->Size = System::Drawing::Size(33, 13);
                    this->lblLogin->TabIndex = 0;
                    this->lblLogin->Text = L"Логін";
                    //
                    // btnEnter
                    //
                    this->btnEnter->Location = System::Drawing::Point(12, 98);
                    this->btnEnter->Name = L"btnEnter";
                    this->btnEnter->Size = System::Drawing::Size(75, 23);
                    this->btnEnter->TabIndex = 2;
                    this->btnEnter->Text = L"Вхід";
                    this->btnEnter->UseVisualStyleBackColor = true;
                    this->btnEnter->Click += gcnew System::EventHandler(this,
&frmLogin::btnEnter_Click);
                    //
                    // btnExit
                    //
                    this->btnExit->Location = System::Drawing::Point(174, 98);
                    this->btnExit->Name = L"btnExit";
                    this->btnExit->Size = System::Drawing::Size(75, 23);
                    this->btnExit->TabIndex = 3;
                    this->btnExit->Text = L"Вихід";
                    this->btnExit->UseVisualStyleBackColor = true;
                    this->btnExit->Click += gcnew System::EventHandler(this,
&frmLogin::btnExit_Click);
                    //
```

КПЗ-2024

```
                    // btnDelete
                    //
                    this->btnDelete->Location = System::Drawing::Point(93, 98);
                    this->btnDelete->Name = L"btnDelete";
                    this->btnDelete->Size = System::Drawing::Size(75, 23);
                    this->btnDelete->TabIndex = 4;
                    this->btnDelete->Text = L"Видалення";
                    this->btnDelete->UseVisualStyleBackColor = true;
                    this->btnDelete->Click += gcnew System::EventHandler(this,
&frmLogin::btnDelete_Click);
                    //
                    // frmLogin
                    //
                    this->AutoScaleDimensions = System::Drawing::Size(6, 13);
                    this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
                    this->ClientSize = System::Drawing::Size(260, 129);
                    this->Controls->Add(this->groupBox);
                    this->Controls->Add(this->btnEnter);
                    this->Controls->Add(this->btnExit);
                    this->Controls->Add(this->btnDelete);
                    this->MaximizeBox = false;
                    this->MinimizeBox = false;
                    this->Name = L"frmLogin";
                    this->Text = L"Вхід у систему УЕД";
                    this->groupBox->ResumeLayout(false);
                    this->groupBox->PerformLayout();
                    this->ResumeLayout(false);

            }
#pragma endregion
/*************************************************************************/
private: System::Boolean PrepareLoginPassword()
            {
                    // Перетворити ім'я до нижнього регістра
                    txtLogin->Text = txtLogin->Text->ToLower();
                    // Забрати в поле ім'я бічні пробіли
                    txtLogin->Text = txtLogin->Text->Trim();

                    // Поле для введення ім'я не містить значення?
                    if(String::IsNullOrEmpty(txtLogin->Text))
                    {
                          // Видати повідомлення про те, поле не містить значення
                          MessageBox::Show("Введіть ім'я користувача.", "Usrs
Manager",
                                MessageBoxButtons::OK, MessageBoxIcon::Warning);
                          // Вийти
                          return false;
                    }

                    // Перевірити введені символи
                    for(Int32 i = 0; i < txtLogin->Text->Length; i++)
                    {
                          // Перевіряється символ
                          Char chTest = txtLogin->Text[i];

                          // Перевірка на коректність
                          if(!(chTest >= 'A' && chTest <= 'Z') &&
                               !(chTest >= 'a' && chTest <= 'z') &&
                               !(chTest >= 'A' && chTest <= 'Я') &&
                               !(chTest >= 'a' && chTest <= 'я') &&
                               !(chTest == '_'))
                          {
                               // Видати повідомлення про те, поле містить
неприпустимий символ
                               MessageBox::Show("Логін містить неприпустимі
символи!",
                                     "Usrs Manager", MessageBoxButtons::OK,
                                     MessageBoxIcon::Warning);
```

КПЗ-2024

```
                        // Вийти
                        return false;
                    }
                }

                return true;
            }
/*************************************************************************/
private: System::Void btnEnter_Click(System::Object^  sender, System::EventArgs^
e)
            {
                // Підготовка значень
                if(!PrepareLoginPassword())
                {
                    return;
                }

                // Уміст поля ім'я - логін адміністратора?
                if(txtLogin->Text == WORKER_LOGIN)
                {
                    // Увійти в обліковий запис адміністратора
                    if(Usrs->WorkerLogin(txtPassword->Text))
                    {
                        // Відобразити форму адміністратора
                        UsrsManager::frmWorker^ frmNew = gcnew
UsrsManager::frmWorker;
                        frmNew->Show();
                    }
                    else
                    {
                        // Вхід неможливий, закінчити роботу функції
                        return;
                    }
                }
                else
                {
                    // Увійти в обліковий запис користувача
                    if(Usrs->ReaderLogin(txtLogin->Text, txtPassword-
>Text))
                    {
                        // Відобразити форму для користувача
                        UsrsManager::frmReader^ frmNew =
                            gcnew UsrsManager::frmReader(txtLogin-
>Text);
                        frmNew->Show();
                    }
                    else
                    {
                        // Вхід неможливий, закінчити роботу функції
                        return;
                    }
                }

                // Сховати форму входу
                this->Hide();
            }
/*************************************************************************/
private: System::Void btnExit_Click(System::Object^  sender, System::EventArgs^
e)
            {
                // Закрити додаток
                Application::Exit();
            }
/*************************************************************************/
private: System::Void btnDelete_Click(System::Object^  sender,
System::EventArgs^  e)
            {
                // Одержати подтвержение на видалення
                if(MessageBox::Show("Ви дійсно хочете зробити видалення?",
```

КбПЗ-2024

```
                        "Usrs Manager", MessageBoxButtons::YesNo,
                        MessageBoxIcon::Information) == ::DialogResult::Yes)
                  {
                        // Підготовка значень логіна й пароля
                        if(!PrepareLoginPassword())
                        {
                              // Якщо виникли помилки, то завершити роботу
функції
                              return;
                        }

                        // Уміст поля ім'я - логін адміністратора?
                        if(txtLogin->Text == WORKER_LOGIN)
                        {
                                    // Обліковий запис адміністратора видалити
не можна
                                    MessageBox::Show("Неможливо видалити
обліковий запис адміністратора!",
                                          "Usrs Manager", MessageBoxButtons::OK,
                                          MessageBoxIcon::Error);
                        }
                        else
                        {
                              // Видалити обліковий запис користувача
                              if(Usrs->RemoveReader(txtLogin->Text,
txtPassword->Text))
                              {
                                    // Видалення пройшло успішно
                                    MessageBox::Show("Обліковий запис успішно
вилучений",
                                          "Usrs Manager",
MessageBoxButtons::OK,
                                          MessageBoxIcon::Information);

                              }
                              else
                              {
                                    // При видаленні відбулася помилка
                                    MessageBox::Show("Неможливо видалити
обліковий запис",
                                          "Usrs Manager",
MessageBoxButtons::OK,
                                          MessageBoxIcon::Error);
                              }
                        }
                  }
            }
/*************************************************************************/
};
}
```

КОПЗ_2024

Файл Doc.cpp - робота з документами

```
/*****************************************************************************/
#include "StdAfx.h"
#include "Doc.h"
/*****************************************************************************/
using namespace System;
/*****************************************************************************/
// NAME:          CDoc
// DESCRIPTION:   Конструктор класу
// INPUT:         N/D
// OUTPUT:        N/D
/*****************************************************************************/
CDoc::CDoc(void)
{
      ddTotalNumber = 0;      // Загальна кількість екземплярів дорівнює 0
      ddFreeNumber = 0; // Кількість вільних екземплярів дорівнює 0
}
/*****************************************************************************/
// NAME:          GetName
// DESCRIPTION:   Функція одержання назви документів
// INPUT:         N/D
// OUTPUT:        Назва документа
/*****************************************************************************/
String^ CDoc::GetName()
{
      return strName;
}
/*****************************************************************************/
// NAME:          SetName
// DESCRIPTION:   Функція установки назви документів
// INPUT:         strValue - назва документа
// OUTPUT:        TRUE - Назва документа установлена
//                      FALSE - Назва не встановлена, рядок-параметр порожня
/*****************************************************************************/
Boolean CDoc::SetName(String^ strValue)
{
      // Перевірити, чи не порожній рядок
      if(!String::IsNullOrEmpty(strValue))
      {
            // Установити значення
            strName = strValue;
            return true;
      }
      else
      {
            return false;
      }
}
/*****************************************************************************/
// NAME:          GetAuthor
// DESCRIPTION:   Функція одержання автора документа
// INPUT:         N/D
// OUTPUT:        Автор документи
/*****************************************************************************/
String^ CDoc::GetAuthor()
{
      // Повернути значення
      return strAuthor;
}
/*****************************************************************************/
// NAME:          SetAuthor
// DESCRIPTION:   Функція установки автора документа
// INPUT:         strValue - ім'я автора документа
// OUTPUT:        N/D
/*****************************************************************************/
void CDoc::SetAuthor(String^ strValue)
```

КОПЗ-2024

```
{
      // Установити значення
      strAuthor = strValue;
}
/*************************************************************************/
// NAME:          GetISBN
// DESCRIPTION:   Функція одержання вхідний номер документа
// INPUT:         N/D
// OUTPUT:        вхідний номер документа
/*************************************************************************/
String^ CDoc::GetISBN()
{
      // Повернути значення
      return strISBN;
}
/*************************************************************************/
// NAME:          SetISBN
// DESCRIPTION:   Функція установки вхідний номер документа
// INPUT:         strValue - вхідний номер документа
// OUTPUT:        TRUE - вхідний номер документа встановлений
//                      FALSE - вхідний номер документа не встановлений, рядок-
параметр порожня
/*************************************************************************/
Boolean CDoc::SetISBN(String^ strValue)
{
      // Перевірити, чи не порожній рядок
      if(!String::IsNullOrEmpty(strValue))
      {
            // Установити значення
            strISBN = strValue;
            // Обчислити й установити хеш-код вхідний номер документа
            ddISBNHash = strISBN->GetHashCode();
            return true;
      }
      else
      {
            return false;
      }
}
/*************************************************************************/
// NAME:          GetTheme
// DESCRIPTION:   Функція одержання теми
// INPUT:         N/D
// OUTPUT:        Тема документи
/*************************************************************************/
String^ CDoc::GetTheme()
{
      // Повернути значення
      return strTheme;
}
/*************************************************************************/
// NAME:          SetTheme
// DESCRIPTION:   Функція установки теми
// INPUT:         strValue - тема документи
// OUTPUT:        N/D
/*************************************************************************/
void CDoc::SetTheme(String^ strValue)
{
      // Установити значення
      strTheme = strValue;
}
/*************************************************************************/
// NAME:          GetPages
// DESCRIPTION:   Функція одержання кількості сторінок
// INPUT:         N/D
// OUTPUT:        Кількість сторінок у книзі
/*************************************************************************/
Int32 CDoc::GetPages()
{
```

КПЗ_2024

```
	// Повернути значення
	return ddPages;
}
/*************************************************************************/
// NAME:		SetPages
// DESCRIPTION:	Функція установки кількості сторінок
// INPUT:		ddValue - кількість сторінок
// OUTPUT:		TRUE - кількість сторінок установлена
//			FALSE - кількість сторінок установлена, неприпустиме
значення аргументу
/*************************************************************************/
Boolean	CDoc::SetPages(Int32 ddValue)
{
	// Параметр має припустиме значення?
	if(ddValue >= 0)
	{
		// Установити значення
		ddPages = ddValue;
		return true;
	}
	else
	{
		return false;
	}
}
/*************************************************************************/
// NAME:		GetTotalNumber
// DESCRIPTION:	Функція одержання загальної кількості екземплярів
// INPUT:		N/D
// OUTPUT:		Загальна кількість екземплярів документи
/*************************************************************************/
Int32 CDoc::GetTotalNumber()
{
	// Повернути значення
	return ddTotalNumber;
}
/*************************************************************************/
// NAME:		SetTotalNumber
// DESCRIPTION:	Функція установки загальне кількості екземплярів
// INPUT:		ddValue - загальна кількість екземплярів
// OUTPUT:		TRUE - кількість екземплярів установлена
//			FALSE - кількість екземплярів установлена, негативне
значення
//			аргумента або аргумент перевищує кількість екземплярів,
що перебуває в
//			користувачів
/*************************************************************************/
Boolean	CDoc::SetTotalNumber(Int32 ddValue)
{
	// Перевірка параметра
	if(ddValue >= 0 && ddValue >= ddTotalNumber - ddFreeNumber)
	{
		// Установити значення
		ddTotalNumber = ddValue;
		return true;
	}
	else
	{
		return false;
	}
}
/*************************************************************************/
// NAME:		GetFreeNumber
// DESCRIPTION:	Функція одержання кількості вільних екземплярів
// INPUT:		N/D
// OUTPUT:		Кількість вільних екземплярів
/*************************************************************************/
Int32 CDoc::GetFreeNumber()
{
```

КОПІЯ - 2024

```
      // Повернути значення
      return ddFreeNumber;
}
/*****************************************************************************/
// NAME:          SetFreeNumber
// DESCRIPTION:   Функція установки кількості вільних екземплярів
// INPUT:         ddValue - кількість вільних екземплярів
// OUTPUT:        TRUE - кількість вільних екземплярів установлене
//                      FALSE - значення параметра перевищує загальна кількість
екземплярів
/*****************************************************************************/
Boolean     CDoc::SetFreeNumber(Int32 ddValue)
{
      // Порівняння параметра із загальною кількістю екземплярів
      if(ddTotalNumber >= ddValue)
      {
            // Установити значення
            ddFreeNumber = ddValue;
            return true;
      }
      else
      {
            // ddFreeNumber = ddTotalNumber;
            return false;
      }
}
/*****************************************************************************/
// NAME:          GetISBNHash
// DESCRIPTION: Функція одержання хеш-коду від вхідного номера документа
// INPUT:         N/D
// OUTPUT: Хеш-         Код вхідний номер документа
/*****************************************************************************/
Int32 CDoc::GetISBNHash()
{
      // Повернути хеш-значення вхідного номера документа
      return ddISBNHash;
}
/*****************************************************************************/
// NAME:          DecFreeNumber
// DESCRIPTION:   Функція зменшення кількості вільних екземплярів
// INPUT:         N/D
// OUTPUT:        TRUE - кількість вільних екземплярів декрементовано
//                FALSE - зменшення неприпустимо, кількість вільних екземплярів
дорівнює 0
/*****************************************************************************/
Boolean     CDoc::DecFreeNumber()
{
      // Кількість вільних екземплярів не дорівнює 0?
      if(ddFreeNumber != 0)
      {
            // Зменшити значення
            ddFreeNumber ---i;
            return true;
      }
      else
      {
            return false;
      }
}
/*****************************************************************************/
// NAME:          IncFreeNumber
// DESCRIPTION:   Функція збільшення кількості вільних екземплярів
// INPUT:         N/D
// OUTPUT:        TRUE - кількість вільних екземплярів інкрементовано
//                FALSE - значення неприпустимо, кількість вільних екземплярів
дорівнює
//                      загальній кількості екземплярів
/*****************************************************************************/
Boolean     CDoc::IncFreeNumber()
```

КПЗ_2024

```
{
	// Кількість вільних екземплярів не дорівнює загальній кількості
екземплярів?
	if(ddFreeNumber != ddTotalNumber)
	{
		// Збільшити значення
		ddFreeNumber ++;
		return true;
	}
	else
	{
		return false;
	}
}
/*************************************************************************/
```

КбПЗ_2024

Файл Doc.h - бібліотека для файлу Doc. cpp

```
     #pragma once
/*************************************************************************/
using namespace System;
/*************************************************************************/
ref class CDoc                                          // Клас "Документ"
{
private:
     String^ strName;                                   // Назва
     String^ strAuthor;                                 // Автор
     String^ strISBN;                                   // вхідний номер документа
     String^ strTheme;                                  // Тема
     Int32 ddPages;                                     // Кількість сторінок
     Int32 ddTotalNumber;                               // Загальна кількість
екземплярів
     Int32 ddFreeNumber;                                // Кількість вільних
(перебувають в
                                                        // системі) екземплярів
     Int32 ddISBNHash;                                  // Значення хеш-функції від
вхідний номер документа

public:
     CDoc(void);                                        // Конструктор класу

     String^ GetName();                                 // Функція одержання назви
документів
     String^ GetAuthor();                               // Функція одержання автора
документа
     String^ GetISBN();                                 // Функція одержання
вхідний номер документа
     String^ GetTheme();                                // Функція одержання теми
     Int32  GetPages();                                 // Функція одержання
кількості сторінок
     Int32  GetTotalNumber();                           // Функція одержання
загальне кількості екземплярів
     Int32  GetFreeNumber();                            // Функція одержання
кількості вільних екземплярів
     Int32  GetISBNHash();                              // Функція одержання хеша-
коду від вхідний номер документа

     Boolean SetName(String^ strValue);         // Функція установки назви
документів
     void SetAuthor(String^ strValue);          // Функція установки автора
документа
     Boolean SetISBN(String^ strValue);         // Функція установки вхідний
номер документа
     void SetTheme(String^ strValue);           // Функція установки теми
     Boolean      SetPages(Int32 ddValue);            // Функція установки
кількості сторінок
     Boolean      SetTotalNumber(Int32 ddValue);      // Функція установки
загальне кількості екземплярів
     Boolean      SetFreeNumber(Int32 ddValue); // Функція установки кількості
вільних екземплярів

     Boolean      DecFreeNumber();                    // Функція зменшення
кількості вільних екземплярів
     Boolean      IncFreeNumber();                    // Функція збільшення
кількості вільних екземплярів
};
/*************************************************************************/
```

КОТЗ_2024

Файл Usrs.cpp - робота з обліковими записами

```
    /*************************************************************************/
#include "StdAfx.h"
#include "Usrs.h"
/*************************************************************************/
using namespace System;
using namespace System::IO;
using namespace System::Windows::Forms;
using namespace System::Collections;
/*************************************************************************/
#define USER_LIST_FILE              "USERLIST.TXT"
#define WORKER_NAME                       "ADMINISTRATOR"
#define WORKER_FILE_EXTENSION ".LMW"
#define READER_FILE_EXTENSION ".LMR"
/*************************************************************************/
// NAME:          CUsrs
// DESCRIPTION:   Конструктор класу
// INPUT:         N/D
/*************************************************************************/
CUsrs::CUsrs(void)
{
}
/*************************************************************************/
// NAME:          WorkerLogin
// DESCRIPTION:   Функція входу й завантаження даних про адміністратора
// INPUT:         strPassword - пароль адміністратора
/*************************************************************************/
Boolean CUsrs::WorkerLogin(System::String ^strPassword)
{
    // Перевірити існування файлу зі списком імен і паролів
    if(File::Exists(USER_LIST_FILE))
    {
     // Відкрити файл
     StreamReader^ srUserList = gcnew StreamReader(USER_LIST_FILE);

     // Завантажити перший рядок з ім'ям адміністратора
     String^ strLogin = srUserList->ReadLine();

     // Зрівняти введене ім'я з ім'ям у файлі
     if(String::Compare(strLogin, WORKER_NAME) != 0)
     {
      // Логіни не збігаються
      MessageBox::Show("Обліковий запис не знайдено.");
      // Повернути помилку
      return false;
     }

     // Завантажити другий рядок з паролем адміністратора
     String^ strPasswordInFile = srUserList->ReadLine();

     // Закрити файл
     srUserList->Close();

     // Зрівняти введений пароль із паролем у файлі
     if(String::Compare(strPassword, strPasswordInFile) != 0)
     {
      // Паролі не збігаються
      MessageBox::Show("Невірний пароль!", "Система УЕД",
            MessageBoxButtons::OK, MessageBoxIcon::Error);
      // Повернути помилку
      return false;
     }

     // Вхід виконаний
     return true;
    }
    else
```

КПЗ–2024

```
    {
     // Створити файл зі списком користувачів
     StreamWriter^ swUserList = gcnew StreamWriter(USER_LIST_FILE);

     // Зберегти ім'я користувача й пароль
     swUserList->WriteLine(WORKER_NAME);
     swUserList->WriteLine(strPassword);

     // Закрити файл
     swUserList->Close();
    }

    return true;
}
/*************************************************************************/
// NAME:          ReaderLogin
// DESCRIPTION:   Функція входу й завантаження даних про користувача
// INPUT:         strName - ім'я облікового запису користувача
//          strPassword - пароль для входу
/*************************************************************************/
Boolean CUsrs::ReaderLogin(System::String^ strName, System::String^ strPassword)
{
    // Прапор удалого входу
    Boolean IsLoginValid = false,
      IsPasswordValid = false;

    // Перевірка існування файлу зі списком користувачів
    if(!File::Exists(USER_LIST_FILE))
    {
     // Видати повідомлення про помилку
     MessageBox::Show("Помилка! Для початку роботи необхідно створити обліковий
запис.",
      "Система УЕД", MessageBoxButtons::OK, MessageBoxIcon::Error);
     // Повернути помилку
     return false;
    }

    // Відкрити файл
    StreamReader^ srUserList = gcnew StreamReader(USER_LIST_FILE);

    do
    {
     // Завантажити ім'я й пароль
     String^ strNameInFile = srUserList->ReadLine();
     String^ strPasswordInFile = srUserList->ReadLine();

     if(String::Compare(strName, strNameInFile) == 0)
     {
      // Ім'я користувача знайдене
      IsLoginValid = true;

      // Зрівняти введений пароль із паролем у файлі
      if(String::Compare(strPassword, strPasswordInFile) == 0)
      {
            // Пароль збігається
            IsPasswordValid = true;
            // Закінчити цикл пошуку
            break;
      }
      else
      {
            // Ім'я знайдене, але пароль не збігається
            IsPasswordValid = false;
            // Закінчити цикл пошуку
            break;
      }

     }
    }
```

КБПЗ_2024

```
    while(srUserList->EndOfStream == false);

    // Закрити файл
    srUserList->Close();

    // Якщо логін збігається, а пароль не збігається, то перервати виконання
    if(IsLoginValid && !IsPasswordValid)
    {
     // Повідомити користувача
     MessageBox::Show("Невірний пароль!", "Система УЕД",
      MessageBoxButtons::OK, MessageBoxIcon::Error);
     // Повернути помилку
     return false;
    }

    // Якщо ні логін, ні пароль не збігаються, то створити новий обліковий запис
    if(!IsLoginValid && !IsPasswordValid)
    {
     // Запропонувати створити пользоватею новий обліковий запис
     if (MessageBox::Show("Облікового запису з таким ім'ям користувача не
знайдено, створити новий обліковий запис?",
                          "Система УЕД", MessageBoxButtons::YesNo,
                          MessageBoxIcon::Question) == DialogResult::Yes)
     {
      // Створити новий обліковий запис
      if(!AddReader(strName, strPassword))
      {
            MessageBox::Show("При створенні нового облікового запису виникла
помилка", "Система УЕД", MessageBoxButtons::OK,
                                    MessageBoxIcon::Error);
            // Не виконувати вхід
            return false;
      }
     }
     else
     {
      // Не виконувати вхід
      return false;
     }
    }

    return true;
}
/*********************************************************************/
// NAME:         AddReader
// DESCRIPTION:  Функція додавання нового облікового запису користувача
// INPUT:        strName - ім'я користувача
//          strPassword - пароль
/*********************************************************************/
Boolean CUsrs::AddReader(System::String ^strName, System::String ^strPassword)
{
    // Створити файловий потік
    FileStream^ fsUserList = gcnew FileStream(USER_LIST_FILE, FileMode::Append);
    // Відкрити файл зі списком користувачів
    StreamWriter^ swUserList = gcnew StreamWriter(fsUserList);

    // Записати ім'я й пароль
    swUserList->WriteLine(strName + Convert::ToChar(13) + Convert::ToChar(10) +
strPassword);

    // Закрити файл
    swUserList->Close();
    // Закрити файловий потік
    fsUserList->Close();

    // Створити файловий потік
    FileStream^ fsReaderFile = gcnew FileStream(strName->ToUpper() +
READER_FILE_EXTENSION,
                                                FileMode::CreateNew);
```

КбПЗ_2024

```
    // Відкрити файл користувача
    StreamWriter^ swReaderFile = gcnew StreamWriter(fsReaderFile);
    // Кількість документів у користувача дорівнює 0
    swReaderFile->WriteLine("0");
    // Закрити файл
    swReaderFile->Close();
    // Закрити файловий потік
    fsReaderFile->Close();

    return true;
}
/*************************************************************************/
// NAME:          RemoveReader
// DESCRIPTION:   Функція видалення облікового запису користувача
// INPUT:         strName - ім'я облікового запису
//                strPassword - пароль
/*************************************************************************/
Boolean CUsrs::RemoveReader(System::String ^strName, System::String
^strPassword)
{
    // Створити файловий потік
    FileStream^ fsUserList = gcnew FileStream(USER_LIST_FILE, FileMode::Open);

    // Пошук позиції рядка із заданим ім'ям користувача

    // Тимчасовий рядок
    String^ strTmp = "";
    // Лічильник символів
    Int64 dqCounter = 0;
    // Непарні рядки - рядка з іменами
    Boolean IsLoginString = true;
    // Цикл пошуку
    do
    {
     // Завантажити байт
     Char dbChar = fsUserList->ReadByte();
     // Якщо байт службовий або досягнуть кінець файлу
     if(dbChar == 0x0D || fsUserList->Position == fsUserList->Length)
     {
      // Якщо рядок ім'я
      if(IsLoginString)
      {
            // Зрівняти ім'я користувача з виділеним рядком
            if(String::Compare(strTmp, strName) == 0)
            {
                  // Скорегувати покажчик
                  dqCounter -= strTmp->Length;
                  // Перервати виконання циклу
                  break;
            }
      }

      // Обнулити рядок
      strTmp = "";
      // Інвертувати прапор рядка ім'я
      IsLoginString = !IsLoginString;
      // Якщо не кінець файлу
      if(fsUserList->Position + 1 != fsUserList->Length)
      {
            // Пропустити службовий символ
            fsUserList->ReadByte();
            dqCounter ++;
      }
     }
     else
     {
      // Додати лічений символ до тимчасового рядка
      strTmp += dbChar;
     }
```

КбПЗ 2024

```
      // Збільшить покажчик
      dqCounter ++;
    }
    // Продовжувати, поки не кінець файлу
    while(fsUserList->Position != fsUserList->Length);

    // Переміститися у файлі до знайденого рядка
    fsUserList->Position = dqCounter;
    // Заповнити ім'я пробілами
    for(Int32 i = 0; i < strTmp->Length; i++)
    {
     fsUserList->WriteByte(' ');
    }

    // Закрити файловий потік
    fsUserList->Close();
    // Видалити файл користувача
    File::Delete(strName + READER_FILE_EXTENSION);

    return true;
}
/*****************************************************************************/
```

КбПЗ_2024

Файл Usrs.h - бібліотека для файлу Usrs.cpp

```
     /*************************************************************************
/
#pragma once
/*************************************************************************/
using namespace System;
/*************************************************************************/
ref class CUsrs   // Клас реєстрації, видалення й авторизації користувачів
{
public:
     // Конструктор класу
     CUsrs(void);

     // Функція додавання нового облікового запису користувача
     Boolean AddReader(System::String^ strName, System::String^ strPassword);
     // Функція видалення облікового запису користувача
     Boolean RemoveReader(System::String^ strName, System::String^ strPassword);

     // Функція входу й завантаження даних про користувача
     Boolean ReaderLogin(System::String^ strName, System::String^ strPassword);
     // Функція входу й завантаження даних про адміністратора
     Boolean WorkerLogin(System::String^ strName);
};
/*************************************************************************/
```

КбПЗ_2024

Файл frmAbout.resx - xml опис форми «про програму...»

```
 <?xml version="1.0" encoding=" utf-8" ?>
- <root>
- <xsd:schema id="root" xmlns="" xmlns:xsd=" " xmlns:msdata="urn: schemas-
microsoft-com: xml-msdata">
  <xsd:import namespace=" " />
- <xsd:element name="root" msdata:IsDataSet="true">
- <xsd:complexType>
- <xsd:choice maxOccurs="unbounded">
- <xsd:element name="metadata">
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element name="value" type="xsd:string" minOccurs="0" />
  </xsd:sequence>
  <xsd:attribute name="name" use="required" type="xsd:string" />
  <xsd:attribute name="type" type="xsd:string" />
  <xsd:attribute name="mimetype" type="xsd:string" />
  <xsd:attribute ref="xml:space" />
  </xsd:complexType>
  </xsd:element>
- <xsd:element name="assembly">
- <xsd:complexType>
  <xsd:attribute name="alias" type="xsd:string" />
  <xsd:attribute name="name" type="xsd:string" />
  </xsd:complexType>
  </xsd:element>
- <xsd:element name="data">
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1"
/>
  <xsd:element name="comment" type="xsd:string" minOccurs="0" msdata:Ordinal="2"
/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" msdata:Ordinal="1"
/>
  <xsd:attribute name="type" type="xsd:string" msdata:Ordinal="3" />
  <xsd:attribute name="mimetype" type="xsd:string" msdata:Ordinal="4" />
  <xsd:attribute ref="xml:space" />
  </xsd:complexType>
  </xsd:element>
- <xsd:element name="resheader">
- <xsd:complexType>
- <xsd:sequence>
  <xsd:element name="value" type="xsd:string" minOccurs="0" msdata:Ordinal="1"
/>
  </xsd:sequence>
  <xsd:attribute name="name" type="xsd:string" use="required" />
  </xsd:complexType>
  </xsd:element>
  </xsd:choice>
  </xsd:complexType>
  </xsd:element>
  </xsd:schema>
- <resheader name="resmimetype">
  <value>text/ microsoft-resx</value>
  </resheader>
- <resheader name="version">
  <value>2.0</value>
  </resheader>
- <resheader name="reader">
  <value>System.Resources.ResXResourceReader, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
- <resheader name="writer">
```

КОПІЯ_2024

  <value>System.Resources.ResXResourceWriter, System.Windows.Forms,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089</value>
  </resheader>
- <data name="txtInfo.Text" xml:space="preserve">

  <value>МАГІСТЕРСЬКА ДИПЛОМНАРОБОТА На тему: Дослідження та програмна
реалізація системи архітектурного рішення інформаційних та комп’ютерних систем
віртуального офісу Керівник: Буравченко К.О. Розробив: студент Лагута Ігор
Анатолійович гр. КН-23М, ЦНТУ, Кропивницький, 2024</value>
  </data>
  </root>

КбПЗ_2024

Файл frmAbout.h - бібліотека для файлу frmAbout.resx

```
#pragma once

using namespace System;
using namespace System::ComponentModel;
using namespace System::Collections;
using namespace System::Windows::Forms;
using namespace System::Data;
using namespace System::Drawing;


namespace UsrsManager
{

	/// <summary>
	/// Summary for frmAbout
	///

	/// </summary>
	public ref class frmAbout : public System::Windows::Forms::Form
	{
/*************************************************************************/
public:	frmAbout(void)
		{
			InitializeComponent();
			//
			//Додавання цього коду в конструктор
			//
		}
/*************************************************************************/
protected: ~frmAbout()
		{
			if (components)
			{
				delete components;
			}
		}
/*************************************************************************/
	private: System::Windows::Forms::Button^  btnOk;
	private: System::Windows::Forms::TextBox^  txtInfo;

	private: System::ComponentModel::Container ^components;

#pragma region Windows Form Designer generated code
		/// <summary>
		/// Необхідний метод для підтримки Розроблювача - не модифікувати
		/// зміст цього методу з кодовим редактором.
		/// </summary>
		void InitializeComponent(void)
		{
			System::ComponentModel::ComponentResourceManager^  resources =
(gcnew System::ComponentModel::ComponentResourceManager(frmAbout::typeid));
			this->btnOk = (gcnew System::Windows::Forms::Button());
			this->txtInfo = (gcnew System::Windows::Forms::TextBox());
			this->SuspendLayout();
			//
			// btnOk
			//
			this->btnOk->Location = System::Drawing::Point(205, 216);
			this->btnOk->Name = L"btnOk";
			this->btnOk->Size = System::Drawing::Size(75, 23);
			this->btnOk->TabIndex = 0;
			this->btnOk->Text = L"OK";
			this->btnOk->UseVisualStyleBackColor = true;
			this->btnOk->Click += gcnew System::EventHandler(this,
&frmAbout::btnOk_Click);
			//
```

КПЗ_2024

```
                    // txtInfo
                    //
                    this->txtInfo->Location = System::Drawing::Point(7, 12);
                    this->txtInfo->Multiline = true;
                    this->txtInfo->Name = L"txtInfo";
                    this->txtInfo->ReadOnly = true;
                    this->txtInfo->Size = System::Drawing::Size(273, 198);
                    this->txtInfo->TabIndex = 3;
                    this->txtInfo->Text = resources->GetString(L"txtInfo.Text");
                    this->txtInfo->TextAlign =
System::Windows::Forms::HorizontalAlignment::Center;
                    //
                    // frmAbout
                    //
                    this->AutoScaleDimensions = System::Drawing::Size(6, 13);
                    this->AutoScaleMode =
System::Windows::Forms::AutoScaleMode::Font;
                    this->ClientSize = System::Drawing::Size(287, 243);
                    this->Controls->Add(this->btnOk);
                    this->Controls->Add(this->txtInfo);
                    this->MaximizeBox = false;
                    this->MinimizeBox = false;
                    this->Name = L"frmAbout";
                    this->Text = L"Про програму...";
                    this->ResumeLayout(false);
                    this->PerformLayout();

            }
#pragma endregion
/*************************************************************************/
private: System::Void btnOk_Click(System::Object^  sender, System::EventArgs^
e)
             {
                    // Закрити форму
                    frmAbout::Close();
             }
/*************************************************************************/
      };
}
```

КбПЗ – 2024