Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

КОМПЛЕКСНА МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Аудит інформаційної безпеки департаментів Вінницької міської ради. Частина 2. Система підтримки аудиту»

Виконав: студент 2 курсу, групи БС-23 м
спеціальності 125 Кібербезпека та захист інформації
_____ Євгеній РАДЧЕНКО
Керівник: к. т. н., доцент каф. ЗІ
_____ Олеся ВОЙТОВИЧ
«16» 12 2024 р.
Опонент: к. т. н., доц., доц. каф. ПЗ
_____ Галина ЧЕРНОВОЛИК
«16» 12 2024 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____ Володимир ЛУЖЕЦЬКИЙ
«16» 12 2024 р.

Вінниця ВНТУ – 2024 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти II (магістерський)
Галузь знань – 12 Інформаційні технології
Спеціальність – 125 Кібербезпека та захист інформації
Освітньо-професійна програма – Безпека інформаційних і комунікаційних систем

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ,
д. т. н., проф.
_____Володимир ЛУЖЕЦЬКИЙ
«18» 09 2024 року

З А В Д А Н Н Я
НА КОМПЛЕКСНУ МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ
СТУДЕНТУ

Радченко Євгенію Валентиновичу

1. Тема роботи: «Аудит інформаційної безпеки департаментів Вінницької міської ради. Частина 2. Система підтримки аудиту»
керівник роботи: Войтович Олеся Петрівна, к. т. н., доцент кафедри ЗІ,
затверджені наказом ректора ВНТУ від 17 вересня 2024 року №310.

2. Строк подання студентом роботи 16 грудня 2024 р.

3. Вихідні дані до роботи:
– зберігання інформації в базі даних;
– формування звітів про невідповідності;
– програмна реалізація підсистеми підтримки аудиту;
– експериментальне дослідження ефективності розробленого методу.

4. Зміст текстової частини: Вступ. 1. Аналіз джерел за напрямком аудиту кібербезпеки. 2. Удосконалення моделі збору інформації для аудиту кібербезпеки 3. Розробка та експериментальне дослідження підсистеми підтримки аудиту 4. Економічна частина. Висновки. Список використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: Перелік записів категорії «інформаційні ресурси». Детальна інформація про запис «диспетчер гаряча лінія цілодобова варта». Загальний звіт з невідповідностей. Фільтрування звітів за критерієм невідповідності. Схема моделі систематизації даних про інформаційні ресурси. Схема бази даних. Архітектура програмного засобу. Алгоритм отримання і обробки даних. Алгоритм формування звіту.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Олеся Войтович, к.т.н., доц. каф. ЗІ		
2	Олеся Войтович, к.т.н., доц. каф. ЗІ		
3	Олеся Войтович, к.т.н., доц. каф. ЗІ		
4	Олеся Войтович, к.т.н., доц. каф. ЗІ		
5	Ольга РАТУШНЯК, к.т.н., доц., доц. каф. ЕПВМ		

7. Дата видачі завдання 1 вересня 2024 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів комплексної магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітки
1	Аналіз завдання. Вступ	01.09.2024 – 10.09.2024	
2	Аналіз інформаційних джерел за напрямком комплексної магістерської кваліфікаційної роботи	10.09.2024 – 15.09.2024	
3	Науково-технічне обґрунтування	16.09.2024 – 22.09.2024	
4	Розробка технічного завдання	23.09.2024 – 29.09.2024	
5	Аналіз методик та методів проведення аудиту	30.09.2024 – 12.10.2024	
6	Аналіз та формулювання вимог до моделі	13.10.2024 – 02.11.2024	
7	Розробка моделі систематизації даних при проведенні аудиту кібербезпеки	03.11.2024 – 10.11.2024	
8	Розробка розділу економічного обґрунтування доцільності розробки	11.11.2024 – 17.11.2024	
9	Аналіз виконання ТЗ, висновки	18.11.2024 – 24.11.2024	
10	Оформлення пояснювальної записки	25.11.2024 – 30.11.2024	
11	Перевірка комплексної магістерської роботи на наявність текстових запозичень	31.11.2024 – 01.12.2024	
12	Попередній захист та доопрацювання МКР	02.12.2024 – 12.12.2024	
13	Представлення МКР до захисту, рецензування	13.12.2024 – 16.12.2024	
14	Захист МКР	17.12.2024 – 20.12.2024	

Студент _____ Євгеній РАДЧЕНК

Керівник роботи _____ Олеся ВОЙТОВИ

АНОТАЦІЯ

УДК 004.81

Радченко Є.В. Аудит інформаційної безпеки департаментів Вінницької міської ради. Частина 2. Система підтримки аудиту. Комплексна магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2024. 87 с.

Укр. мовою. Бібліогр.: 32 назви; рис.: 21; табл.: 9.

Комплексна магістерська кваліфікаційна робота зосереджена на покращенні процесу аудиту інформаційної безпеки Вінницької міської ради. У процесі виконання дослідження було здійснено аналіз інструментів підтримки аудиту, методів збору та обробки інформації а також стандартів, за якими проводиться аудит кібербезпеки. На основі аналізу було висунуто вимоги та розроблено модель систематизації даних, яку було імплементовано в підсистему підтримки аудиту.

Ілюстративна частина представлена 10 плакатами, які демонструють зовнішній вигляд підсистеми підтримки аудиту, результати розробки моделі та програмного засобу. Вони наочно відображають ключові етапи та результати роботи, що дозволяє краще зрозуміти розроблену модель та оцінити ефективність підсистеми підтримки аудиту.

Економічна частина присвячена оцінці науково-технічного потенціалу та економічної доцільності розробки. В ній доведено, що науково-дослідна робота має високий потенціал та є економічно доцільною.

Ключові слова: аудит, інструмент підтримки аудиту, кібербезпека, моніторинг, відповідність, систематизація даних

ABSTRACT

Radchenko Y.V. Audit of Information Security of Departments of Vinnytsia City Council. Part 2. Audit support system. Comprehensive master's qualification work in the specialty 125 – Cybersecurity, educational program – Security of Information and Communication Systems. Vinnytsia: VNTU, 2024. 87 p.

In Ukrainian. Bibliography: 32 titles; fig.: 21; tab.: 9.

The comprehensive master's qualification work is focused on improving the process of auditing the information security of the Vinnytsia City Council. In the course of the study, an analysis of audit support tools, methods for collecting and processing information, as well as standards for cybersecurity audits was carried out. Based on the analysis, requirements were put forward and a data systematization model was developed, which was implemented in the audit support subsystem.

The illustrative part is represented by 10 posters that demonstrate the appearance of the audit support subsystem, the results of the development of the model and the software tool. They clearly reflect the key stages and results of work, which allows you to better understand the developed model and evaluate the effectiveness of the audit support subsystem.

The economic part is devoted to the assessment of scientific and technical potential and economic feasibility of development. It proves that research work has a high potential and is economically feasible.

Keywords: audit, audit support tool, cybersecurity, monitoring, compliance, data systematization

ЗМІСТ

ВСТУП.....3
1 АНАЛІЗ ДЖЕРЕЛ ЗА НАПРЯМКОМ АУДИТУ КІБЕРБЕЗПЕКИ5
1.1 Аналіз потреби в аудиті кібербезпеки 5
1.2 Аналіз стандартів, за якими проводиться аудит 7
1.3 Визначення вимог до засобу підтримки аудиту 10
1.4 Аналіз сучасних рішень з підтримки аудиту 11
1.5 Постановка задачі..... 18
1.6 Висновки з розділу..... 19
2 УДОСКОНАЛЕННЯ МОДЕЛІ ЗБОРУ ІНФОРМАЦІЇ ДЛЯ АУДИТУ КІБЕРБЕЗПЕКИ..... 20
2.1 Аналіз сучасних моделей збору даних при аудиті кібербезпеки... 20
2.2 Розробка моделі систематизації даних 23
2.3 Розробка структури бази даних 29
2.4 Висновки з розділу..... 33
3 РОЗРОБКА ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПІДСИСТЕМИ ПІДТРИМКИ АУДИТУ 35
3.1 Архітектура програмного засобу..... 36
3.2 Вибір інструментів розробки 45
3.3 Розробка програмного засобу 51
3.4 Експериментальне дослідження підсистеми підтримки аудиту 59
3.5 Висновки з розділу..... 66
4 ЕКОНОМІЧНА ЧАСТИНА 68
4.1 Технологічний аудит розробки..... 69
4.2 Розрахунок витрат на здійснення науково-дослідної роботи..... 73
4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором 80
4.4 Висновки з розділу..... 82
ВИСНОВКИ 83
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 85
ДОДАТКИ..... 88
Додаток А ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ..... **Error! Bookmark not defined.**
Додаток Б Код блоку обробки запитів..... 90
Додаток В Код блоку зв’язку з базою даних..... 91
Додаток Г Код ядра..... 92
Додаток Д Показники оцінювання науково-технологічного рівня та комерційного потенціалу..... 93
Акт впровадження

ВСТУП

В сучасному світі свою ефективність довів новий вид війни – гібридна війна. Сюди входять як всі попередньо відомі способи ведення бойових дій, так і деякі нові, зокрема, найнебезпечніший з них – інформаційна війна. Це як можливість впливати на настрої громадян, збирати інформацію, що раніше була недоступною, або навіть порушувати роботу підприємств на території ворога, не вдаючись до використання зброї [1]. Таким чином, аудит інформаційної безпеки стає критично необхідним інструментом для захисту інформаційної та кібербезпеки держави, що знаходиться в стані війни. Проте, постає інше питання – проведення аудиту.

Як відомо, будь-який аудит є довгим та кропітким процесом, а аудити інформаційної безпеки, завдяки своєму впливу на працездатність всього підприємства, можна вважати ще більш складним для виконання. Цей процес може займати місяці, в залежності від розміру та комплексності підприємства та його інформаційних систем, навіть використовуючи найбільш сучасні та комплексні інструменти [2]. Як результат, постає проблема часу – необхідно виявити можливі вразливості негайно, уникаючи всіх можливих затримок. Проблемами та вразливостями в інформаційних системах, які знаходять експерти, можуть скористатись ще до того, як інформацію про них отримає підприємство, або навіть до того, як їх взагалі знайдуть.

Очевидною є необхідність в пришвидшенні процесу аудиту. І, звісно, найкращим способом пришвидшення процесу є його автоматизація. Здатність покласти частину роботи, наприклад, форматування даних, або заповнення таблиць, на програмний застосунок допоможе значно. Важливо також розуміти необхідність захисту конфіденційності чутливих даних, що обробляються під час проведення аудиту. Для прикладу, використання хмарних сервісів може поставити під питання доцільність та користь від проведення аудиту, якщо дані про інформаційні системи опиняються на непідконтрольних підприємству або аудитору серверах [3].

Об’єктом дослідження є процес аудиту кібербезпеки

Предметом дослідження є методи та засоби підтримки аудиту кібербезпеки

Метою комплексної магістерської кваліфікаційної роботи є покращення та пришвидшення процесу аудиту кібербезпеки за рахунок розробки програмного засобу для систематизації інформації про інформаційні системи підприємства з метою подальшої обробки та аналізу на відповідність вимогам обраного стандарту.

Для досягнення мети необхідно виконати такі завдання:

- проаналізувати сучасний процес аудиту кібербезпеки;
- розробити вимоги до програмного засобу підтримки аудиту кібербезпеки;
- провести порівняльний аналіз наявних рішень в сфері підтримки аудиту кібербезпеки;
- розробити модель систематизації даних;
- розробити підсистему для підтримки аудиту;
- провести експериментальне дослідження системи підтримки аудиту, який реалізує збір, обробку та аналіз даних про інформаційні системи.

Новизна комплексної магістерської роботи полягає в розробці моделі збору та обробки інформації для підтримки аудиту кібербезпеки, що дозволяє прискорити процес збору та обробки інформації під час аудиту.

Результати роботи наведені в тезах конференцій, що були опубліковані в процесі роботи над комплексною магістерською роботою [1-3].

1 АНАЛІЗ ДЖЕРЕЛ ЗА НАПРЯМКОМ АУДИТУ КІБЕРБЕЗПЕКИ

1.1 Аналіз потреби в аудиті кібербезпеки

Аудит кібербезпеки в сучасному світі є вимогою для нормального функціонування великих, середніх та навіть малих підприємств, їх взаємодії з державою та партнерами. Без результатів аудиту, що підтверджують захищеність інформації, яку обробляє підприємство, користувачі не стануть довіряти йому свої дані, інші компанії не стануть співпрацювати, а у випадку підприємств критичної інфраструктури з'являється ризик повної зупинки. Все це – найменш важливі наслідки відсутності підтвердженої сертифікації інформаційної безпеки підприємства. Варто пам'ятати про можливість кібератак, витоків інформації, порушень цілісності та доступності інформації.

Все це пояснює загальну актуальність процесу аудиту. Жодне підприємство не здатне нормально функціонувати на ринку без проведення детального аудиту кібербезпеки. В той же час, умовах військового стану актуальність аудиту лише зростає. З'являються нові ризики, пов'язані як зі збільшенням кількості атак так і зі збільшенням критичності всієї інформації. Раніше неважлива інформація, що мала обмежений рівень захисту, як, наприклад, дані про щільність населення або логістичні маршрути, може раптово набути значної критичності через потенційну шкоду, до якої може призвести її витік.

Під час проходження практики в департаменті інформаційних технологій Вінницької міської ради було проведено аналіз інформаційних систем підприємства та проведено змістовну консультацію з спеціалістами в напрямі захисту інформації.

В процесі проходження практики було визначено найбільш довготривалі та вибагливі до людських ресурсів етапи аудиту кібербезпеки. Було визначено, що найбільш проблематичним етапом є збір інформації про інформаційні системи підприємства.

На даний момент процедура збору інформації полягає в проведенні опитування працівників підприємства про інформаційні системи, які вони

використовуються в роботі. Оскільки підприємства не завжди проводять інвентаризацію власних ресурсів, з’являється проблема в тривалому процесі опитування з метою збору інформації.

Як результат, цей процес може затягуватись на тижні а то і місяці, що є неприпустимим в умовах військового стану. Виникає проблема, яку можна описати як «горло пляшки». Зазвичай цей термін описує ситуацію, коли потік рідини обмежується найбільш вузьким проміжком. В даному випадку «горлом пляшки» виступає процес опитування та збору інформації про інформаційні системи підприємства.

На даний момент існує велика кількість інструментів підтримки аудиту. Вони мають різноманітні призначення, від розв’язання технічних, на кшталт створення та управління контролями безпеки, до організаційних питань, наприклад, тайм-менеджмент, розподіл обов’язків, тощо.

Наявні на ринку програмного забезпечення рішення організаційних питань процесу аудиту користуються значною популярністю, проте вибір робиться більше за рахунок особистих побажань конкретної організації, що проводить аудит інформаційної безпеки. В той же час, більш технічні засоби, такі як масивні GRC-системи, використовуються значно рідше.

Цьому явищу можу бути декілька причин. Зокрема варто виділити відсутність гнучкості такого програмного забезпечення, вузький напрямок застосування, використання хмарних технологій, що ставить під питання безпеку інформації, тощо.

Таким чином, на ринку відсутнє загальне рішення, яке може бути корисним для кожного окремого аудиту інформаційної безпеки, з чого випливає потреба в створенні певних інструментів підтримки аудиту, яке буде універсальним та корисним для більшості аудиторів.

Таким чином, військовий стан та війна загалом привносять корективи в процес аудиту – зменшуються терміни, оскільки результати необхідно отримати негайно, і, як наслідок, падає ефективність даного процесу. В той же час, сам по собі процес проведення аудиту кібербезпеки є далеким від ідеального,

з’являється необхідність в інструментах, що дозволять пришвидшити процес аудиту водночас не нехтуючи точністю та ефективністю. На додаток до цього, на ринку відсутні універсальні рішення для збору даних про інформаційні системи підприємства в процесі аудиту.

Отже, виникає потреба в створенні програмного засобу, що здійснюватиме підтримку аудиту кібербезпеки шляхом спрощення процесу опитування та збору інформації.

1.2 Аналіз стандартів, за якими проводиться аудит

При проведенні будь-якого аудиту, організації, що проводять оцінку, спираються на різноманітні стандарти, що дозволяє їм проводити сертифікацію рівня відповідності підприємств за чіткими метриками. Таким чином, проведення аудиту дозволяє забезпечити точну оцінку відповідності підприємства мінімальним вимогам в тій чи іншій сфері. Проведення аудиту інформаційної безпеки не є виключенням. Спеціалісти використовують велику кількість різноманітних стандартів, за якими вони проводять сертифікацію підприємства в таких напрямках як управління доступом, захист інформації від витоків, захист від втручання, мережева безпека, тощо.

Найголовнішим стандартом проведення аудиту інформаційної безпеки є серія стандартів ISO 27000. Їх затверджено Державною службою спеціального зв'язку та захисту інформації України, в наказах, які цитують, або напряму посилаються на дані стандарти [4]. Таким чином ця серія стандартів є основоположною частиною аудиту інформаційної безпеки, як а має як нормативні так і рекомендаційні положення та дозволяє певною мірою обирати необхідний напрям для стандартизації, що показано на рис. 1.1.

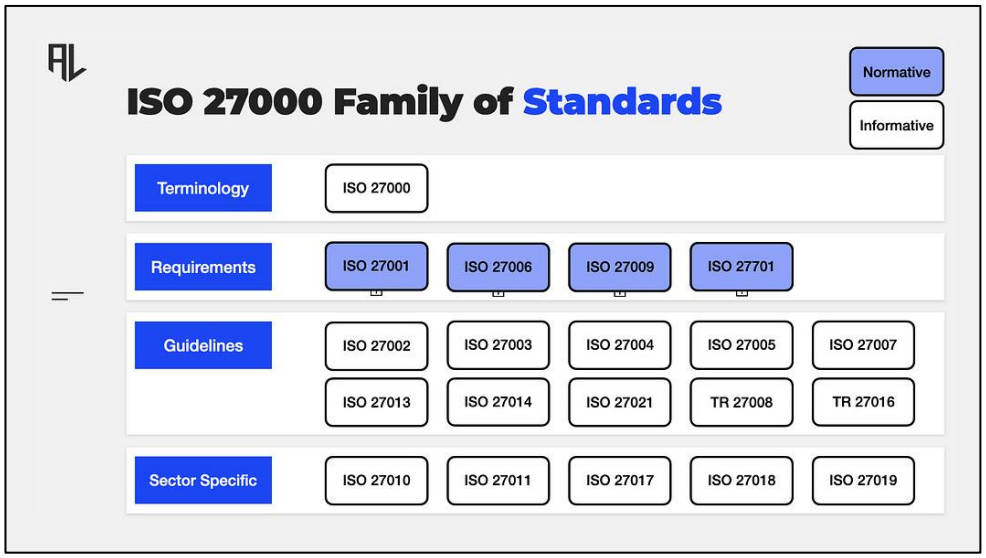


Рисунок 1.1 – Графічне зображення категорій стандартів серії ISO 27000

Загалом стандарти ISO є міжнародно прийнятими для використання в більшості країн світу. Міжнародна організація зі стандартизації пропонує стандарти в багатьох сферах та користується повагою серед міжнародної спільноти. Завдяки багаторічній репутації та постійним оновленням стандартів, організація ISO по праву вважається лідером в переважній більшості сфер застосування стандартів, в тому числі в сфері інформаційної та кібербезпеки [5].

Хоча стандарти ISO 27000 є базовими вказівками з забезпечення мінімального допустимого рівня захисту інформації, вони в той же час є основою в процесі побудови систем захисту, контролю доступу та управління ризиками. Таким чином, стандарти ISO 27000 вважаються обов'язковими для слідування навіть на державному рівні.

Також, в якості рекомендаційних документів, додаткових сертифікацій та для загального покращення практик в сфері захисту інформації та кібербезпеки, використовується стандарт NIST (National Institute of Standards and Technology). Розроблений в США, цей стандарт по праву вважається одним з найдетальніших та найвибагливіших у сфері захисту інформації. Подібно до стандартів ISO 27000, NIST зосереджується на оцінці ризиків та розробці, впровадженні і експлуатації систем захисту [6]. Проте, також він включає в себе рекомендації

щодо реагування на інциденти, відновлення після них та забезпечення безперервності бізнесу.

Основою даного стандарту є NIST CSF, або Cybersecurity Framework, що використовується для управління ризиками при роботі з інформаційними системами. Цей документ поділяє процес роботи з кіберризиками на 5 основних функцій.

- Ідентифікація – виявлення активів та ризиків в організації та за її межами.
- Захист – реалізація засобів та заходів захисту.
- Виявлення – моніторинг та виявлення інцидентів.
- Реагування – реагування на інциденти.
- Відновлення – відновлення бізнесу після атак, забезпечення безперервності бізнес процесів [7].

В результаті формується модель постійної підтримки інформаційної системи, реагування на інциденти кібербезпеки та відновлення після інцидентів. Кожна окрема функція Cybersecurity Framework включає в себе декілька більш специфічних та конкретних вимог та процесів, як показано на рис. 1.2.

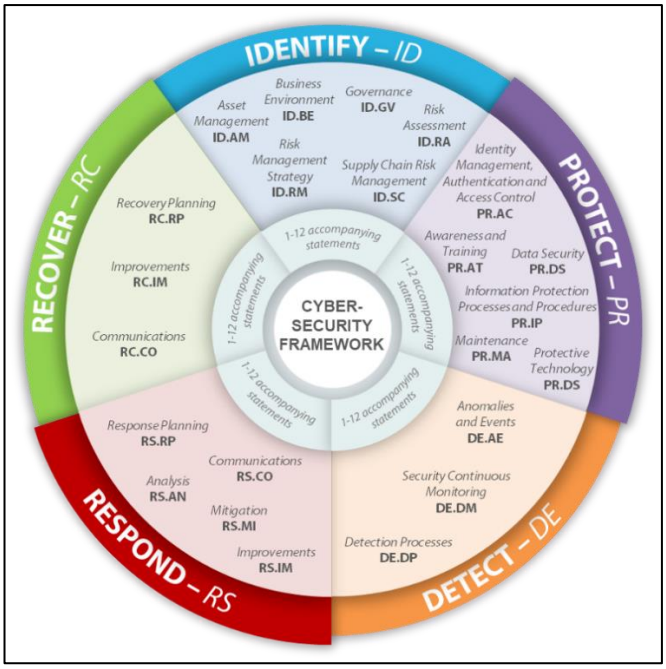


Рисунок 1.2 – Візуальне зображення основ NIST CF

На додаток до CSF, NIST також містить певні документи, що можуть бути корисними для підприємств за межами США. Це, зокрема, NIST SP 800-53, що описує впровадження контролів безпеки, NIST SP 800-37, що описує процес впровадження та моніторингу систем захисту, NIST SP 800-30 – керівництво з управління ризиками, та NIST SP 800-34, що надає рекомендації щодо процесу відновлення після інцидентів та безперервності бізнесу [8].

Оскільки в Україні стандарт NIST не є затвердженим на державному рівні, деякі організації та підприємства нехтують ним, надаючи перевагу серії стандартів ISO 27000. Проте варто зауважити, Більшість підприємств, в тому числі і державних, використовують рекомендації таких стандартів як NIST або COBIT для покращення своїх інформаційних систем.

1.3 Визначення вимог до засобу підтримки аудиту

Таким чином, існує потреба в пришвидшенні процесу збору інформації, зокрема, даних про інформаційні системи на підприємстві. В той же час, для зручної обробки інформації варто використовувати вже перевірені рішення – програмні застосунки від Microsoft, що використовуються при проведенні аудиту на даний момент, є оптимальним рішенням, та не потребують заміни [9].

Іншим питанням, що постає при проведенні аудиту кібербезпеки, є захист інформації, що обробляється. Програмні застосунки Excel, Access та PowerBI можна використовувати локально на ПК або внутрішньому сервері, для того аби не передавати конфіденційну інформацію підприємства за межі внутрішньої мережі. Проте більшість сучасних інструментів підтримки аудиту працюють за допомогою хмарних технологій та надають послуги за моделями SaaS або PaaS, що стає додатковим ризиком та створює можливий вектор атаки на підприємство [10].

Іншими словами, в даній ситуації існує потреба в локальних рішенням – програмних застосунках, які можна використовувати на персональному комп’ютері або розгорнути в внутрішній мережі підприємства. Зокрема використання веб-застосунку, що є самодостатнім для роботи в локальній

мережі, дозволить зменшити навантаження на персональні комп'ютери працівників та задіяти більшу обчислювальну потужність. Оскільки об'єми інформації, що обробляється, можуть бути доволі значними, в залежності від конкретного підприємства, дана можливість може значно покращити продуктивність. Як наслідок – зростає ефективність процесу збору інформації, за рахунок відсутності затримок, пов'язаних з повільним програмним забезпеченням.

Іншою вимогою до інструменту підтримки аудиту є можливість автоматичного або автоматизованого аналізу даних. Якщо під час проведення аудиту користувач вносить дані про інформаційний ресурс, серед цих даних гарантовано є такі, які можна перевірити на відповідність автоматично. Для прикладу, програмний застосунок може визначити, що останнє оновлення програмного забезпечення відбувалось надто давно, або що інформація з високим рівнем критичності передається за межі підприємства через незахищений канал. Далі програмний застосунок може повідомити користувача про це, або по завершенню роботи скласти повноцінний звіт про всі невідповідності та порушення.

Варто зауважити також важливість питання зручності інтерфейсу. Саме зручність використання програмного засобу диктує ефективність роботи з ним, і, як наслідок – швидкість виконання поставленої задачі. Проте, оскільки даний програмний застосунок призначений виключно для вирішення технічних питань аудиту, та не розрахований на широку аудиторію, графічний інтерфейс не зобов'язаний виглядати привабливо. Достатньо аби він був інтуїтивно зрозумілим та не заважав роботі, що забезпечується мінімалістичним дизайном. Хоча дані вимоги не є обов'язковими, вони також мають вплив на ефективність програмного засобу як інструменту підтримки аудиту.

1.4 Аналіз сучасних рішень з підтримки аудиту

Згідно найновіших опитувань, більшість аудиторів та організацій, що проводять аудит інформаційної безпеки, використовують в роботі виключно

базові інструменти для роботи з великими обсягами даних, до яких, здебільшого відносяться Microsoft Excel, Power BI та Microsoft Access [11].

Microsoft Excel – програмний застосунок для роботи з файлами в табличному форматі, зокрема xls, xlsx, xlsm, csv та багато інших. Цей застосунок пропонує зручний інтерфейс, великий функціонал, можливість автоматизації та навіть підтримку користувацьких розширень, що дозволяє додати новий функціонал та зробити роботу більш ефективною. Загалом, Excel надає достатньо можливостей для обробки посереднього за розміром набору даних, проте не завжди може впоратись з навантаженням, викликаним великою кількістю рядків таблиці. З урахуванням необхідності зображення всього обсягу даних на екрані в зручному для користувача інтерфейсі, швидкодія застосунку може падати [12].

З іншого боку, Microsoft Access ідеально підходить для обробки великих наборів даних. Це спеціалізований програмний застосунок для роботи з базами даних, отже, він був розроблений спеціально для роботи щ великими обсягами інформації. Можливість зручного написання власних скриптів для роботи з базами даних означає практично необмежений функціонал, що робить Access дуже гнучким інструментом [13]. Також, здатність додавання скриптів може дозволити певною мірою автоматизувати процес перевірки на відповідність.

Power BI, в свою чергу, є інструментом для візуалізації великих обсягів інформації. Працюючи у зв’язці з Excel або Access, він дозволяє отримати дані про інформаційні системи підприємства у зручному та читабельному вигляді, в тому числі – для презентації керівництву підприємства з метою прийняття рішень щодо покращення інформаційної безпеки організації [14].

Ці три основні програмні застосунки створені корпорацією Microsoft, що пояснює можливість ефективної взаємодії між ними. Таким чином забезпечується можливість обробки великих обсягів даних, їх візуалізації та презентації. Проте, з урахуванням того факту, що вони створювались з метою вирішення широкого спектру задач, можна припустити, що ефективність даного програмного забезпечення при зборі інформації, необхідної для проведення

аудиту є посередньою. Підтвердженням цього можна вважати існування великої кількості систем підтримки аудиту, модулів розширення для програмного забезпечення Microsoft, систем автоматизації збору даних, тощо [15].

Не дивлячись на мешу популярність ніж стандартний пакет програмного забезпечення від Microsoft, ефективними також є вузькоспеціалізовані програмні застосунки для проведення аудиту, для прикладу – системи ITAM (IT Asset Management), які дозволяють вести облік та категоризувати активи IT-інфраструктури, в тому числі в контексті інформаційного аудиту чи аудиту кібербезпеки. Ці програмні застосунки покликані замінити стандартні засоби роботи з таблицями або базами даних. Серед переваг – вузька спеціалізація даного програмного забезпечення означає, що порівняно з програмними застосунками широкої спеціалізації, вони повинні бути більш зручними, ефективними та швидкими [16]. Більшість ITAM-систем дійсно мають функціонал, який здатний допомогти при проведенні аудиту, проте не завжди користуються популярністю через необхідність переносу інформації на хмарне середовище. Особливо критичним це є при проведенні аудиту інформаційної безпеки – якщо процес аудиту містить в собі більше ризиків ніж він може усунути, потреба в такому аудиті є сумнівною. Також варто зауважити що ITAM-системи здебільшого створені для аудиту IT-інфраструктури, а не аудиту кібербезпеки. Більшість з цих інструментів не надають повного функціоналу, що необхідний при розгляді вразливостей системи, як, наприклад, можливість аналізу методів доступу до інформаційної системи або статусу шифрування ПК [17]. Загалом такі системи працюють за моделлю SaaS, хоча є і аналоги, здатні працювати локально, не залежно від підключення до мережі Інтернет. Проблемою стає їх найбільша перевага – незалежність. Дані програмні застосунки створюються для моніторингу мережі та інформаційної системи, що нечасто означає можливість експорту даних до програм для обробки великих обсягів даних. Через свій специфічний функціонал ITAM-системи також не мають можливості повноцінно замінити такі програмні застосунки, адже не

надають інструментів для обробки даних про інформаційні системи, хоча деякі автоматично візуалізують систему, що обробляється.

Також варто звернути увагу на можливість масштабування. Переважна більшість ITAM-систем створені для обробки надвеликих масивів даних, тобто, для використання з великими підприємствами або навіть конгломератами. В результаті ціна на подібне програмне забезпечення вказується з розрахунком на використання в великому масштабі, що може бути проблематично для підприємств малого та середнього бізнесу, як показано на рис. 1.3.

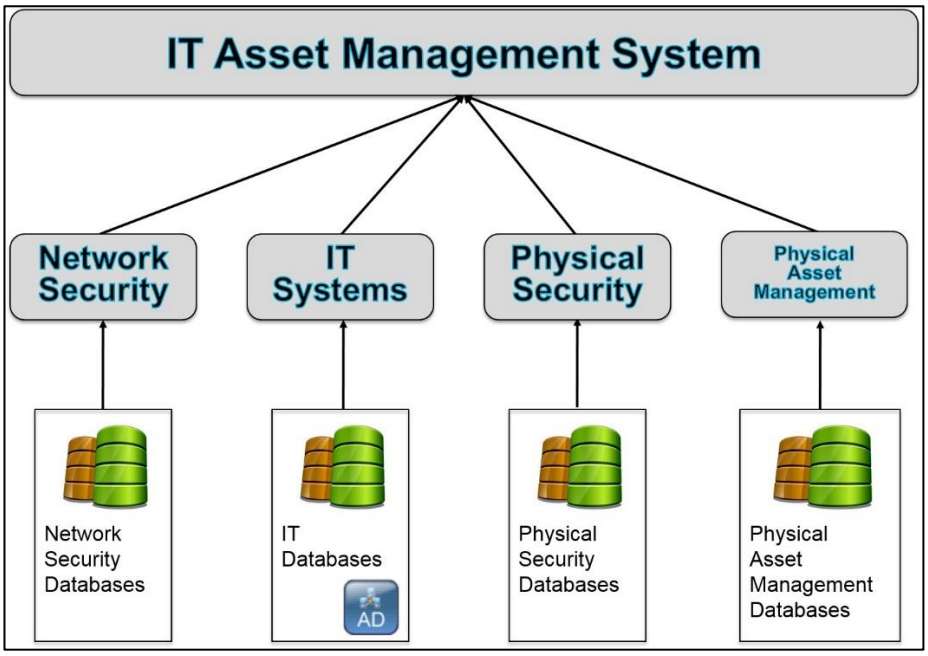


Рисунок 1.3 – Основні принципи роботи ITAM-систем

Також існують певні інструменти оцінки ризиків, які автоматично обробляють дані про систему як ціле та окремі її частини, та на основі алгоритмів визначають ризики того чи іншого доступу, або тієї чи іншої вразливості. Прикладом такої системи можна навести SAP GRC. Вона користується певною популярністю, проте більше підходить для застосування при проведенні аудиту в великих компаніях-холдингах, що мають великі мережі та значні обсяги даних. Іншим недоліком є той факт що даний програмний застосунок обробляє дані про вразливості лише в межах програмного забезпечення від організації SAP [18]. Також варто зауважити, що дане програмне забезпечення розгортається у

внутрішній мережі підприємства, що дозволяє знизити ризик витоку інформації. Структура роботи SAP GRC показана на рис. 1.4.

Рисунок 1.4 – Основні функції та переваги SAP GRC

Також існують більш специфічні програмні засоби, такі як Netwrix Auditor – платформа для аналізу поведінки та оцінки ризиків. Даний програмний застосунок створений з метою допомоги в аудиті змін в IT-системі за допомогою відслідковування стану кожної окремої інформаційної системи, аналізу поведінки користувачів, та складанні звітів з відповідності та звернень користувачів. Таким чином, даний програмний застосунок частково покриває вимоги до засобу підтримки аудиту кібербезпеки шляхом обробки даних про інформаційні системи, проте кінцевий результат роботи не відповідає цілям аудиту – Netwrix Auditor покладається на тривалу взаємодію з інформаційною системою та її користувачами. На додаток до цього, цей програмний застосунок є хмарним сервісом, що ставить під питання доцільність його використання на підприємствах критичної інфраструктури та при роботі з конфіденційними даними [19]. З іншого боку, програмний застосунок має повноцінну інтеграцію з продуктами від компанії Microsoft та навіть підтримує можливість прямого вивантаження даних з віддаленої бази даних в застосунок Excel або Access.

Приклад аналізу поведінки в даному програмному забезпеченні зображено на рис. 1.5.

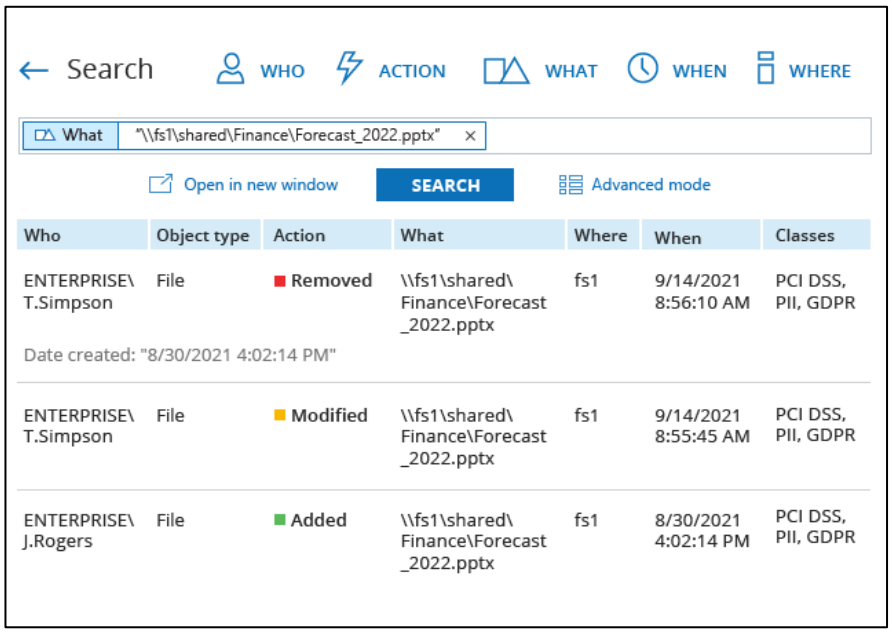


Рисунок 1.5 – Процес роботи з інформаційними системами в Netwrix Auditor

MasterControl - інший програмний застосунок, що позиціонує себе як інструмент підтримки аудиту. Проте його ціль – не збір, обробка та візуалізація даних, а аналіз ризиків та побудова звітів у режимі реального часу. Хоча даний функціонал безсумнівно є корисним для організації, що проводить аудит, він більше відноситься до рішень організаційних питань, ніж технічних [20]. З іншого боку, популярність даного програмного забезпечення серед аудиторів показує, наскільки зручність процесу впливає на його ефективність.

Однією з систем GRC, що широко використовується в аудиті IT є LogicGate – програмний застосунок, що надає можливість моніторингу контролів безпеки в режимі реального часу, управління доступом, менеджменту відповідності, тощо. Проте, серед функцій, що присутні в даному програмному забезпеченні, відсутні такі, що можуть пришвидшити початкові етапи аудиту – всі перелічені функції відносяться до інструментів підтримки інформаційної системи, що застосовуються після проведення аудиту [21]. В такому випадку зручність є

значним фактором, проте питання пришвидшення процесу аудиту вже не є актуальним.

Для визначення відповідності переліченого програмного забезпечення вищевказаним вимогам було проведено порівняльний аналіз, результати якого показані у табл. 1.1.:

Таблиця 1.1 – Відповідність ПЗ вимогам до інструменту підтримки аудиту

Назва програмного застосунку	Підтримує збір даних про інформаційні системи?	Можна використовувати локально?	Підтримує інтеграцію з Excel, Access або PowerBI (або здатний замінити їх)	Зручний інтерфейс	Автоматизація
Microsoft Office	+	+	+	+	-
ITAM-системи	+	+/-	-	+	+
SAP GRC	+	+	+	-	+
Netwrix Auditor	+	-	+	+	-
MasterControl	-	-	-	+	+
LogicGate	-	-	+	+	-

Як видно в таблиці 1.1, сучасне програмне забезпечення, що позиціонується як інструмент підтримки аудиту, не відповідає вищевказаним вимогам. Таким чином, існує попит на підсистему підтримки аудиту, що дозволить:

- швидко та ефективно збирати дані про інформаційні системи;
- форматувати дані таким чином, щоб їх можна було обробляти незалежно;
- виконувати збір даних всередині локальної мережі підприємства, без необхідності підключення до інтернету;
- матиме інтуїтивно зрозумілий інтерфейс;
- буде здатний автоматично перевіряти дані про інформаційні ресурси;

Отже, в результаті аналізу сучасного програмного забезпечення на ринку підтримки процесу аудиту можна сказати, що відсутнє рішення, що дозволить малому або середньому бізнесу збирати, форматувати та обробляти дані про інформаційні системи в локальній мережі. Як результат, існує потреба в створенні такого програмного застосунку.

1.5 Постановка задачі

Як показав аналіз методів та засобів аудиту кібербезпеки, на процес збору інформації витрачається багато часу, що ускладнює аудит, створює затримки та знижує його загальну ефективність. Тобто існує проблема несвоєчасного отримання даних про вразливості в інформаційній системі. Отже, основна мета розробки повинна бути прискоренням процесу збору інформації під час аудиту.

Також було проведено порівняльний аналіз сучасних інструментів підтримки аудиту. Результати показали, що такі програмні застосунки як SAP GRC, MasterControl, LogicGate та Netwix Auditor не відповідають суворим вимогам аудиту кібербезпеки. Ці вимоги полягають в відсутності передачі інформації через мережу Інтернет, можливість збору даних про інформаційні системи та їх експорт в формат, зручний для подальшої роботи. Через сучасний вектор розвитку програмного забезпечення, що спрямований на хмарні сервіси за моделлю SaaS, таке програмне забезпечення створює додаткові ризики, а намагання розробників створити самодостатній продукт не дозволяє інтегрувати його з найпопулярнішими рішеннями в сфері аналізу даних.

Оптимальним варіантом для інструменту з підтримки аудиту в даній ситуації є програмний застосунок, що працює в локальній мережі підприємства, дозволяє користувачам власноруч, без участі аудиторів, вносити дані про інформаційні системи підприємства і вивантажувати ці дані для подальшої обробки аудиторами чи працювати з результатами автоматичного аналізу всередині програмного засобу.

При розробці програмного застосунку варто також врахувати необхідність створення інтуїтивно зрозумілого користувацького інтерфейсу, оскільки користувачами даного програмного засобу можуть бути люди з обмеженим досвідом в сфері IT. Таким чином, необхідно створити інтерфейс, для використання якого не потребуватиметься значних технічних знань та навичок.

З метою забезпечення вільного доступу до функціоналу підсистеми для всіх користувачів необхідно розробити застосунок для підтримки аудиту на основі

веб-інтерфейсу. Це дозволить користувачам мати можливість внесення даних з будь-якої точки локальної мережі, без прив’язки до конкретного ПК або робочого місця.

1.6 Висновки з розділу

В даному розділі було проаналізовано процес аудиту кібербезпеки. На основі даних, отриманих в процесі проходження практики, було визначено, що велику частину затримок при проведенні аудиту кібербезпеки становить процес збору даних про інформаційні системи через проведення опитування особисто аудиторами. Етап збору інформації було визначено як неефективний та такий, що потребує вдосконалення.

При аналізі процесу збору інформації було визначено вимоги до інструменту підтримки аудиту, що дозволить пришвидшити процес збору та, як наслідок, аналізу даних про інформаційні системи. Було проведено порівняльний аналіз програмних засобів, що присутні на ринку та надаються послуги з підтримки аудиту. Було визначено, що існує попит на універсальну підсистему з підтримки аудиту, який не задовольняється.

2 УДОСКОНАЛЕННЯ МОДЕЛІ ЗБОРУ ІНФОРМАЦІЇ ДЛЯ АУДИТУ КІБЕРБЕЗПЕКИ

2.1 Аналіз сучасних моделей збору даних при аудиті кібербезпеки

Збір даних про інформаційні системи полягає в формуванні даних про розміщення, захист, використання, технічні та організаційні характеристики інформаційних систем в зручному для подальшої обробки форматі. В залежності від способів обробки даних, який використовується під час проведення аудиту, дані формати можуть різнитися, проте стандартом в цій сфері вважається програмне забезпечення обробки даних в табличному форматі.

Найпростіший метод, який досі використовується в певних обмежених випадках – ручний збір інформації. Введення даних напряму в таблицю аудитором або працівником, або, в окремих випадках суворої політики безпеки, що не дозволяє використання електронно-обчислювальних пристроїв на території підприємства, заповнення анкет вручну для подальшого перенесення в цифровий вигляд [22].

Цей метод є надзвичайно ненадійним та вимагає співставлення робочих графіків декількох осіб. Що в умовах безперервної роботи підприємства може бути проблематично або неможливо. Головні недоліки даного методу:

- повільний процес запису отриманих даних вручну;
- довготривалий процес;
- вимагає особистої присутності декількох осіб [23];

Варто також згадати про можливість внесення даних напряму в таблицю з метою подальшої обробки даних в зручному для роботи форматі. Зазвичай для цього використовуються програми на кшталт Microsoft Excel, які дозволяють працювати з великими обсягами даних у вигляді таблиці, використовувати формули для автоматизованого або автоматичного підрахунку та аналізу. Звісно, внесення даних одразу до програми, в якій ці дані будуть оброблятись, є перевагою, якщо говорити про уникнення етапу перенесення даних

Абсолютно новим підходом до збору інформації є використання штучного інтелекту в даному процесі. Для прикладу, застосування чат-бота для опитування користувачів та одночасно надання консультацій чи пояснень з певних окремих питань [24]. Такий підхід може здатись ідеальним, адже він виключає необхідність в особистій присутності експертів для опитування та дає працівникам можливість вносити інформацію дистанційно за допомогою звичних засобів зв'язку або персональних комп'ютерів. Однак, використання штучного інтелекту тягне за собою велику кількість нюансів, які частково або повністю нівелюють потенційну користь [25].

Для прикладу, тренування мовної моделі для опитування можна здійснити Двома методами, проте обидва мають як переваги так і недоліки. Якщо взяти за основу вже існуючу мовну модель, таку як ChatGPT, варто звернути увагу на умови використання. Розробники більшості мовних моделей, що можна використати як основу для створення власного штучного інтелекту використовують дані, які обробляються їх клієнтами для подальшого навчання своїх нейромереж. Це означає, що інформація, яка буде вноситись до бази даних за допомогою штучного інтелекту, буде потрапляти також до баз даних третьої сторони, що є прямим витоком інформації навіть при умові, що збір даних відбувається неперсоніфіковано.

З іншого боку, створення мовної модулі з нуля є проблематичним з точки зору вартості та тривалості розробки. Цей процес може займати місяці, а то і роки та мати надзвичайну вартість. Для відносно нескладної задачі, такої як збір даних про інформаційні системи, розробка повноцінної мовної моделі є надмірним рішенням, яке не надасть достатньо вагомих переваг порівняно з більш класичними рішеннями.

Також варто враховувати питання використання штучного інтелекту. Для роботи мовної моделі потрібні значні обчислювальні потужності. Якщо підприємство, аудит якого проводиться, не має на своїй території технічних засобів для підтримки мовної моделі, її використання може бути можливим лише за умови віддаленого підключення, що компрометує конфіденційність та

цілісність інформації, що передається та обробляється [26]. Як відомо, будь яка передача інформації за межі внутрішньої мережі підприємства є небажаною, адже є потенційним джерелом витоку.

Варто зазначити, що штучний інтелект не є ідеальним помічником в будь-якому питанні. Широко відомо про періодичні збої в роботі, що породжують неточні відповіді. Для прикладу, штучний інтелект, створений компанією Google для своєї пошукової системи, відомий через свої неточні а іноді небезпечні відповіді та поради, які часто не відображають реальність [27]. Це стається через необачне тренування мовної моделі, або через технічні помилки, що можуть не залежати від відповідальних осіб. В будь-якому випадку, для надважливих задач, таких як проведення аудиту, використання штучного інтелекту повинно строго регулюватись та контролюватись, адже будь-яка помилка може мати руйнівні наслідки, якщо не помітити її на початкових етапах.

Порівняно з попередніми методами проведення опитування та збору інформації, використання спеціалізованого програмного забезпечення є збалансованим рішенням, що поєднує зручність технологій з надійністю контрольованого середовища, дозволяючи брати найкраще як від класичних методик та методів, так і від більш сучасних.

Для прикладу, система SAP GRC має дуже широкий функціонал управління доступом. Це дозволяє видати необхідні права будь-якому користувачу системи для внесення або редагування даних в системі. Таким чином, якщо система SAP GRC встановлена на підприємстві, аудитори можуть отримати доступ до необхідної їм інформації, не залучаючи працівників підприємства. Це є беззаперечним плюсом з точки зору ефективності [28]. Проте постає питання, чи наявні в таких інструментах підтримки всі функції, необхідні безпосередньо для початкового збору даних про інформаційні системи підприємства.

Проте подібні програмні засоби часто створені для інших цілей. Звісно, певний функціонал можна використовувати для збору інформації або навіть адаптувати під інші задачі, яких вимагає процес аудиту кібербезпеки. Проте не варто забувати, що такі нетрадиційні міри не передбачені розробниками, а отже,

не завжди є надійними. Зокрема, хоча в SAP GRC і присутня функція експорту даних в файловий формат з розділювачами-комами «.csv», внутрішні системи даного інструменту не підтримують створення користувацьких баз даних. Дана система була розроблена як доповнення інших продуктів SAP та дозволяє керувати інформаційними ресурсами та даними, що зберігаються в цих системах, проте це накладає обмеження на обробку інформації про інші системи, які не відносяться до програмного пакету SAP [29].

2.2 Розробка моделі систематизації даних

З огляду на вимоги, висунуті при аналізі сучасних рішень, та наявні на ринку інструменти в сфері підтримки аудиту кібербезпеки, можна сказати що на даний момент відсутнє універсальне рішення для збору та обробки даних про інформаційні системи. Отже необхідно розробити метод проведення опитування за допомогою спеціалізованого програмного забезпечення.

Опираючись на результати переддипломної практики було розроблено метод систематизації даних про інформаційні ресурси шляхом розбиття всієї інформації на 5 категорій:

- дані про інформаційний ресурс;
- дані про програмне забезпечення;
- дані про апаратне забезпечення;
- дані про кадровий склад підприємства;
- дані про способи передачі інформації;

Для демонстрації розробленої моделі було побудовано схему, що зображена на рисунку 2.1.

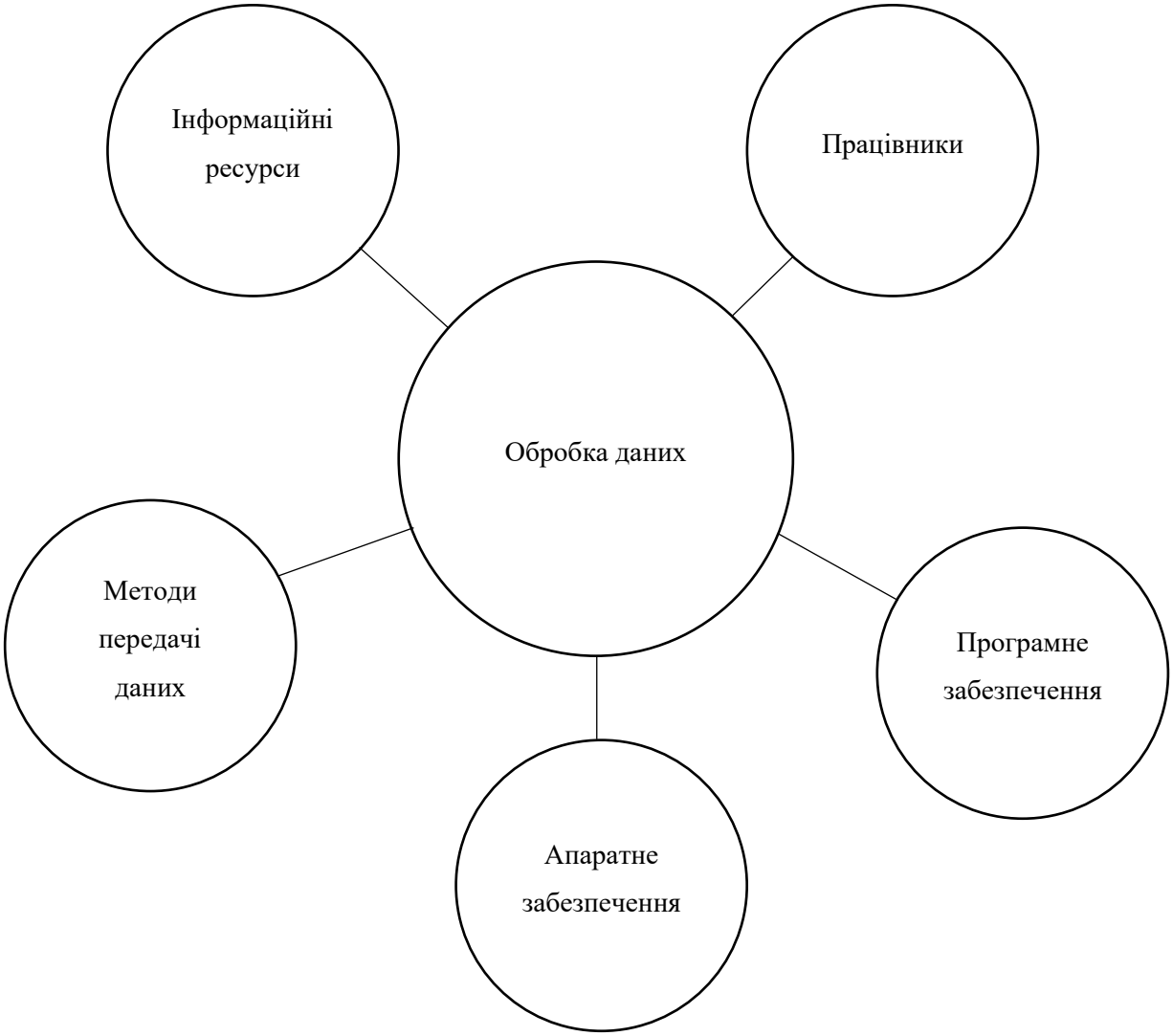


Рисунок 2.1 – Узагальнена модель систематизації даних

Категорія даних про інформаційний ресурс є головною. Ці дані включають в себе посилання на 4 інші категорії в рамках опису програмного та апаратного забезпечення, що використовується для обробки інформації, працівників підприємства, що працюють з цією інформацією, та методів передачі даної інформації. Також варто зазначити необхідність додаткових полів для опису, таких як, наприклад, опис конфіденційності, цілісності та доступності інформації.

Дані про програмне забезпечення включають в себе такі дані як розміщення, методи управління доступом до програмного забезпечення, відповідальні особи, наявність ліцензії, дата останнього оновлення, тощо. Ця категорія призначена

для всебічного опису та класифікації будь-якого програмного забезпечення, що бере участь в зберіганні, передачі або обробці інформації.

Дані про апаратне забезпечення включають інформацію про розташування, перелік осіб, що мають фізичний або логічний доступ до даного апаратного забезпечення, інвентарний номер, операційну систему, тощо. Дана категорія призначена для опису апаратного забезпечення, що присутнє на території підприємства а також за його межами.

Дані про кадровий склад підприємства повинні включати такі ключові поля, як ім’я, контакти працівника, його посаду та відділ. Це дозволить чітко визначити відповідальних осіб та їх посадові обов’язки, на основі яких вони можуть мати право на доступ до певних ресурсів.

Дані про методи передачі інформації включають в себе інформацію про напрямок передачі, використання методів захисту інформації при передачі, програмне та апаратне забезпечення, що використовується для передачі, тощо. Таким чином, можна сформувати повноцінну картину про процес передачі даних для визначення відповідності вимогам того чи іншого стандарту.

Окрім методу систематизації даних про інформаційні ресурси також необхідно розробити новий підхід з точки зору технічних можливостей програмного забезпечення. Так, для забезпечення доступності інструменту підтримки аудиту кібербезпеки необхідно розглянути можливість розміщення даного програмного засобу в локальній мережі підприємства. Таким чином, кожен працівник матиме змогу вносити дані в будь-який зручний час, а розміщення застосунку в локальній мережі підприємства допоможе уникнути додаткових ризиків, пов’язаних з передачею даних за межі внутрішньої мережі підприємства.

Для роботи інструменти підтримки аудиту кібербезпеки необхідна також функція збереження даних. Це може бути зроблено за допомогою вбудованої бази даних, використання баз даних, наявних у внутрішній мережі підприємства або за допомогою збереження проміжних результатів опитування в окремий файл. З точки зору захисту інформації, збереження даних в файл є небажаним, з

огляду на відсутність шифрування, резервного копіювання або логування змін. Таким чином, вирішено використовувати реляційні бази даних.

З точки зору захисту інформації, різниця між вбудованою базою даних та базою даних, що вже використовується на підприємстві, є незначною. Обидві можуть бути захищеними як шифруванням так і стандартними методами управління доступом, обидві можуть мати резервні копії та журнал подій, який буде нотувати будь-які зміни.

З точки зору універсальності програмного засобу, використання вбудованої бази даних має значну перевагу за рахунок відсутності необхідності в створенні нової бази даних в локальній мережі, її налаштуванню та підключенні. Таким чином, інструмент підтримки аудиту можна розгорнути в локальній мережі одразу готовим до використання, що пришвидшить роботу та, як наслідок – збільшить ефективність аудиту.

Процес обробки інформації полягає в аналізі даних про існуючі інформаційні системи та їх відповідність стандартам кібербезпеки, обраним підприємством. Для прикладу, з метою перевірки відповідності інформаційних систем таким вимогам стандарту ISO 27001 як «наявність шифрування при передачі конфіденційної інформації за межі підприємства» можна використати автоматичну перевірку таких полів як «Критичність» в категорії «Дані про інформаційний ресурс» та «Методи захисту» в категорії «Методи передачі». При використанні незахищеного методу передачі даних для конфіденційної інформації підсистема підтримки аудиту може виводити сповіщення або формувати звіт на основі інформації про невідповідність бізнес процесу вимогам стандарту ISO 27001.

Перейдемо до опису математичної моделі процесу систематизації даних про інформаційні ресурси, отриманих під час аудиту інформаційної безпеки:

$$M = \{I, F, O\} \tag{2.1}$$

де I – множина вхідних даних;

O – множина вихідних даних;

F – множина функцій.

Перейдемо до детального опису множин.

Вхідні дані, позначені I – множина даних про інформаційний ресурс. Це включає в себе показники критичності, цілісності та доступності, власників інформації, розміщення в мережі, програмне забезпечення, яке обробляє дану інформацію та методи передачі інформації. Варто зауважити, що вхідні дані отримуються у вигляді п’яти об’єктів, S, R, P, H, та T. Розглянемо детальніше:

- об’єкт R – дані інформаційний ресурс;
- об’єкт S – дані про програмне забезпечення, що обробляє інформацію;
- об’єкт H – дані про апаратне забезпечення, що обробляє або зберігає інформацію;
- об’єкт P – дані про кадровий склад підприємства, залучений до збереження, обробки чи передачі інформації;
- об’єкт T – методи передачі, що використовуються для даної інформації.

Таким чином, маємо множину вхідних даних, описану формулою 2.2.

$$I = \{S, R, P, H, T\} \tag{2.2}$$

Результатом роботи моделі систематизації даних є вихідні дані O. Сюди входить сформований об’єкт A, що містить розгорнуту інформацію про інформаційну систему з урахуванням релевантних даних про пов’язані структури, такі як програмне та апаратне забезпечення, працівники, що мають відношення до даної інформації та методи передачі інформації. Також до вихідних даних O належить звіт про невідповідності Q. Таким чином, множина вихідних даних описана формулою 2.3.

$$O = \{A, Q\} \tag{2.3}$$

Множина функцій F описує набір перетворень та операцій, що застосовуються при обробці інформації для її систематизації. Першою функцією є Populate, функція створення результуючого об’єкту на основі п’яти вхідних об’єктів. В результаті її роботи створюється новий об’єкт A, що містить дані про інформаційний ресурс а також необхідні для аналізу дані з інших об’єктів. Дана функція описується формулою 2.4.

$$A = Populate(S, R, P, H, T) \tag{2.4}$$

Функція ComplianceCheck виконує перевірку об’єкту A на відповідність визначеним вимогам. Відбувається формування об’єкту N, що містить перелік невідповідностей. Даний процес описано в формулі 2.5

$$N = \text{ComplianceCheck}(A) \tag{2.5}$$

Функція FormReport на основі об’єкту C формує звіт Q та надсилає його користувачу. Дана функція описується формулою 2.6.

$$Q = \text{FormReport}(N) \tag{2.6}$$

В результаті маємо формулу 2.7, що описує множину функцій, що застосовуються для систематизації та аналізу інформації.

$$F = \{populate, complianceCheck, formReport\} \tag{2.7}$$

В результаті підстановки всіх формул в формулу, що описує процес систематизації даних, отримуємо деталізований математичний опис, представлений формулою 2.8

$$M = \{\{S, R, P, H, T\}, \{populate, complianceCheck, formReport\}, \{A, Q\}\} \quad (2.8)$$

Отже, розроблена модель систематизації даних про інформаційні системи полягає в:

- Заповненні п’яти об’єктів відповідними даними, що стосуються інформаційного ресурсу
- Форматуванні цих даних у придатний для ручної обробки вигляд
- Аналізу даних на невідповідності
- Формуванні звіту на основі результатів аналізу

Таким чином, даний метод забезпечує ефективну підтримку аудиту завдяки систематизації та автоматичному аналізі даних про інформаційні ресурси.

2.3 Розробка структури бази даних

Для збереження даних про інформаційні системи необхідно створити структуровану базу даних. Зберігання даних в звичайному файлі можливо, проте даний метод позбавлений переваг будь-якої бази даних, для прикладу – структурованості даних, простоти методів запису, читання, редагування та видалення інформації, захисту даних, резервного копіювання, а також можливості віддаленої взаємодії. Звісно, всі ці функції можна імплементувати всередині програмного засобу таким чином, щоб використовувався звичайний файл в директорії застосунку, проте даний метод розробки є непрактичним.

Бази даних можна розділити на реляційні та нереляційні. Вони відрізняються структурою, методом зберігання інформації та призначенням.

Реляційні бази даних, або SQL бази даних, названі на честь Sequence Quarry Language – мови запитів, що використовується в цих базах даних. Реляційні бази даних побудовані з таблиць, що поділені на рядки та стовпчики. Стовпчики мають чітко визначені ім’я та тип даних, що може зберігатись в кожному конкретному стовпчику. Кожен рядок – новий об’єкт або набір інформації, що

має певне значення в кожному рядку. До переваг даного типу баз даних належать висока організованість та структурованість, можливість роботи з великими масивами даних та надійність.

Нереляційні бази даних, в свою чергу, діляться на декілька типів:

- документні;
- бази даних «ключ-значення»;
- графові;
- стовбчикові.

Найбільш наближеними до реляційних баз даних є документні бази даних. Такі бази даних використовують структуру об’єктів, що мають набір полів. Кожен об’єкт має власний набір полів, що можуть мати різні значення. У порівнянні з табличним форматом SQL бази даних, такий підхід виглядає менш організовано, проте має свої переваги, зокрема – можливість зміни властивостей даних без зміни структури бази даних, простота налаштування та адаптивність, а відсутність чіткого формату в певних випадках може стати перевагою.

Бази даних типу «ключ-значення» є найбільш мінімалістичним підходом до зберігання даних, якщо мова йде про бази даних. Ці бази даних мають певну схожість з документними базами даних через те, що вони також є адаптивними за рахунок своєї структури. Бази даних «ключ-значення» містять набір пар ключ-значення, де ключ відображає назву об’єкта або ідентифікатор, за яким виконується його пошук, а значення – сама інформація, що зберігається. Таким чином, забезпечується швидкий доступ до даних з мінімальними затратами обчислювальних потужностей, проте є і недоліки: відсутність можливості структуризації та групування значень, необхідність існування чітко визначеної схеми створення ключів.

Графові бази даних описують зв’язки між об’єктами за допомогою вузлів, ребер та властивостей. Перевагами даного підходу є фокус на зв’язках між елементами, явне відображення даних зв’язків та відсутність обмежень в типах зв’язків, оскільки всі вони описуються користувачами. Оскільки графові бази даних зосереджуються на зв’язках, а не на самих елементах, їх користь в

інструменті збору даних про інформаційні системи при проведенні аудиту є сумнівними, хоча виключати таку можливість не варто.

Стовбчикові бази даних нагадують SQL бази даних, проте не мають таких чітких обмежень в типах даних, адже вся інформація фактично зберігається в стовпчиках. Кожен стовпчик містить набір заголовків, до кожного з яких прив'язаний набір пар ім'я-значення. Даний підхід спрощує сегментацію та забезпечує високу продуктивність.

На основі результатів практики, яку було пройдено на підприємстві «Вінницька міська рада», було визначено, що підприємство середнього бізнесу, зазвичай, має не більше 1000 окремих інформаційних ресурсів, до 200 співробітників та порівняно невелику кількість програмного та апаратного забезпечення. При обробці такого відносно невеликого набору даних різниця в швидкодії між звичайною SQL базою даних та нереляційною базою даних є незначною, що усуває всі потенційні переваги реляційних баз даних. З іншого боку, використання нереляційних баз даних дозволить не лише ефективніше розробляти програмний засіб, що реалізує підсистему підтримки аудиту, а й спростить процес його адаптації під конкретні вимоги підприємства. Для прикладу, введення нових полів в колекцію не вимагатиме внесення змін в саму базу даних.

Структура бази даних заснована на вимогах моделі систематизації даних. Таким чином, відповідно до п'яти категорій, що описують дані про інформаційні ресурси, апаратне та програмне забезпечення, кадровий склад підприємства та методи передачі інформації, необхідно створити п'ять колекцій даних, що будуть відображати ці категорії. Кожна колекція повинна містити основний ключ – ідентифікатор. Певні колекції міститимуть посилання на об'єкти інших колекцій. Таким чином можна буде динамічно отримувати детальну інформацію про пов'язані об'єкти, як, наприклад, посаду користувача, що має доступ до певної інформації, або метод автентифікації, що використовується на програмному забезпеченні, що розміщене на сервері. В результаті розробки бази даних було створено структуру, яка зображена на рис. 2.2.:

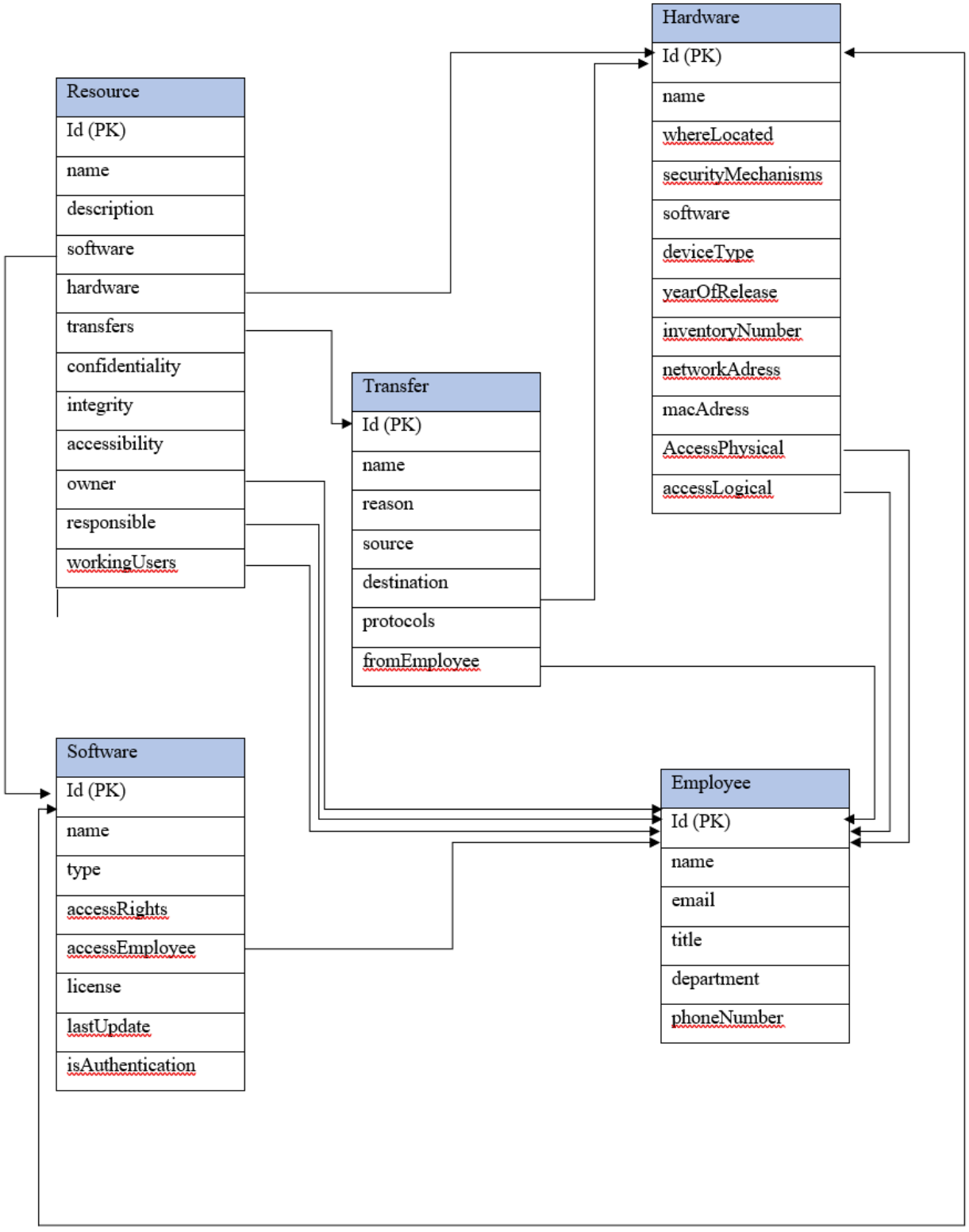


Рисунок 2.2 – Структура бази даних

Отже, було вирішено використати нереляційну базу даних документного типу для розробки програмного застосунку. Адаптивність та відсутність необхідності в чіткій структурі дозволять зробити програмний застосунок

універсальним, а простота налаштування дозволять спростити процес розробки. Загалом це дозволить створити програмний засіб, що забезпечить ефективну підтримку процесу аудиту шляхом збору, зберігання та аналізу даних про інформаційні ресурси.

2.4 Висновки з розділу

Було проаналізовано сучасні та класичні методи збору даних про інформаційні системи. Кожен метод було проаналізовано на прикладах, визначено головні переваги та недоліки.

Класичні методи збору інформації за особистої участі аудиторів довели свою надійність за рахунок безпосередньої участі експертів в процесі. Це дозволяє забезпечити надійність даних та надання консультації при зборі інформації, якщо це необхідно. Проте, дані методи не є ефективними з точки зору витрат часу, адже вони потребують особистої присутності декількох людей, що може значно ускладнити процес.

Найсучасніші методи, на кшталт використання штучного інтелекту також мають як переваги так і недоліки. До переваг можна віднести в першу чергу зручність процесу для користувача, в той час як недоліками вважаються складність реалізації, можливість технічних несправностей штучного інтелекту та інші технічні обмеження, пов’язані з потребою в значних обчислювальних потужностях.

Спеціалізоване програмне забезпечення, в свою чергу, довело свою ефективність та беззаперечну перевагу в процесу збору інформації. Можливість для користувачів вносити дані без участі аудиторів є перевагою як в зручності так і в ефективності процесу аудиту кібербезпеки. Можливість побудови алгоритму відповідно до вимог, без обмежень, які накладаються, для прикладу, штучним інтелектом, є ще однією перевагою та надає достатньо гнучкості для того аби створити універсальний інструмент підтримки аудиту інформаційної безпеки.

Для коректного функціонування програмного застосунку та ефективної обробки даних необхідно розробити та впровадити модель систематизації та аналізу даних, на основі якої буде проводитись обробка даних. Це дозволить забезпечити підтримку аудиту на рівні, який можна порівняти з провідними хмарними сервісами, в той же час не компрометуючи безпеку даних, що обробляються.

Отже, програмний застосунок, розроблений з урахуванням специфічних вимог конкретного підприємства, повинен забезпечити достатній рівень захисту інформації за рахунок його розміщення в локальній мережі. В той же час ефективність інструменту підтримки аудиту буде забезпечуватись шляхом застосування розробленої моделі систематизації даних, що дозволить пришвидшити процес збору та обробки інформації.

3 РОЗРОБКА ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПІДСИСТЕМИ ПІДТРИМКИ АУДИТУ

Поставленою задачею є розробка програмного засобу для збору даних про інформаційні ресурси підприємства. Першим кроком є розробка алгоритму, архітектури та схеми роботи програми. Після цього необхідно обрати інструменти, методи та підходи до розробки програмного застосунку відповідно до визначених вимог.

Оскільки програмний застосунок повинен бути доступним в локальній мережі, основою його архітектури повинен бути мережевий інтерфейс на основі будь-якої мови програмування, що підтримує розробку веб-застосунків.

Враховуючи вимоги до збереження, обробки, зміни та вивантаження даних, бажаним також є використання бази даних. Між різними видами баз даних є велика різниця, що полягає як в їх загальній структурі так і в функціоналі, що потребує більш детального вивчення.

Для забезпечення зручного та ефективного процесу збору даних необхідно також створити зручний та інтуїтивно зрозумілий інтерфейс. Основна частина цього питання більше залежить від обраної мови програмування, проте також постає питання дизайну, яке вирішується окремо та фактично є самостійною проблемою, вирішення якої не прив’язано до конкретних обраних інструментів розробки.

Окрім мови програмування також важливо обрати бібліотеки, фреймворки та модулі, що будуть використані для розробки. Оскільки сучасний стан сфери розробки програмного забезпечення вимагає використання сторонніх бібліотек для введення певного функціоналу у відносно короткі терміни та з достатньою ефективністю, необхідно також обрати додаткові зовнішні інструменти для роботи. Вимоги до цих бібліотек, модулів або фреймворків можна описати наступним чином:

- відсутність залежності від зовнішніх ресурсів для забезпечення конфіденційності всієї інформації, що обробляється застосунком;

- можливість додавання, налаштування або блокування функцій відповідно щодо вимог до роботи програмного застосунку;
- мінімальний вплив на продуктивність програмного застосунку через перенавантаженість модулями.

Останнім кроком перед початком розробки є вибір технічних інструментів розробки. Це може бути середовище розробки, середовище тестування, додаткові засоби проміжного тестування, тощо.

Таким чином, були визначені необхідні вимоги до засобів розробки програмного засобу. Перейдемо до вибору конкретних інструментів.

3.1 Архітектура програмного засобу

Як було визначено в попередньому розділі, програмний засіб, що виконує функцію підтримки аудиту шляхом збору даних про інформаційні ресурси повинен бути веб-застосунком, що розміщується в локальній мережі та працює в браузері, зосереджуючи основну частину програмної логіки, а разом з нею і навантаження, на сервері. Це допоможе забезпечити швидкодію програмного засобу за рахунок більшого ресурсу обчислювальних потужностей.

Першим кроком є розробка узагальнену схему архітектури програмного засобу. Це допоможе визначитись з напрямком роботи, необхідним функціоналом та інструментами, що можуть допомогти під час розробки програмного застосунку.

Як будь-який інший веб-застосунок, що виконує функцію взаємодії між користувачем та сервером, даний програмний засіб необхідно чітко розділити на серверну та клієнтську частини. Таким чином можна буде розмежувати блоки, які відповідатимуть за різні функції та етапи обробки даних.

Після розділення архітектури на дві частини варто визначити, які блоки будуть присутні в кожній частині. Для клієнтської частини необхідний всього один блок – блок графічного інтерфейсу, з яким буде взаємодіяти користувач. З цього блоку до серверної частини будуть надходити запити з даними, на які сервер буде відповідати або надсилаючи нову веб-сторінку з необхідними

даними або взаємодіючи з базою даних відповідно до інструкцій, отриманих від користувача, тобто створювати новий запис відповідно до даних, переданих користувачем, видаляти записи або вивантажувати їх та форматувати в табличний вигляд для завантаження.

Серверна частина програмного засобу повинна містити всю програмну логіку, що робить її набагато більшою за розмірами та більш комплексною, ніж клієнтська частина. Це, в свою чергу, допомагає виконати задачу перенесення навантаження на сервер.

Перейдемо до переліку блоків, які необхідні для функціонування серверної частини:

- блок обробки запитів;
- ядро;
- блок валідації даних;
- блок взаємодії з базою даних;
- база даних

В результаті, маємо відповідну архітектуру програмного засобу, що виконує функцію підтримки аудиту шляхом збору інформації. Узагальнена архітектура зображена на рисунку 3.1.

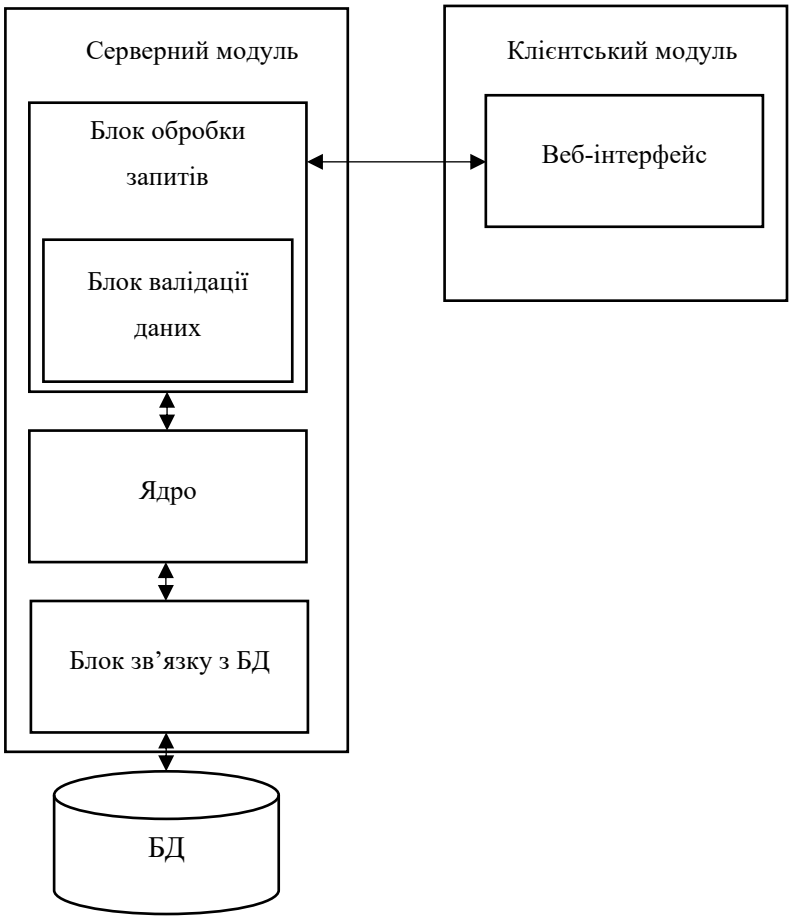


Рисунок 3.1 – Узагальнена архітектура програного засобу

Далі опишемо детальніше кожен з блоків, починаючи з клієнтської частини.

3.1.1 Структура модуля клієнта

Модуль клієнта складається всього з одного блоку – блоку веб-інтерфейсу. Даний блок виконує функцію відображення інформації та керування базою даних за допомогою полів, кнопок та інших елементів користувацького інтерфейсу.

Блок веб-інтерфейсу складається з декількох веб-сторінок, кожна з яких виконує окрему функцію, формується на основі об'єктів, отриманих з бази даних та допомагає користувачу маніпулювати інформацією, що зберігається в базі даних за допомогою зручного та інтуїтивно зрозумілого інтерфейсу.

Головна сторінка повинна відображати список існуючих інформаційних ресурсів та коротку інформацію про них, для прикладу – їх розташування. Кожен запис повинен мати свій власний елемент на сторінці, при натисканні на який можна отримати більше деталей про даний інформаційний ресурс. Також

головна сторінка повинна містити кнопки для таких функцій як вивантаження звіту про невідповідності та внесення нового запису в базу даних.

Сторінка деталей про конкретний інформаційний ресурс повинна містити детальну інформацію про конкретний запис з бази даних та формуватись динамічно на основі існуючих записів. Також, окрім інформації, дана сторінка повинна містити кнопку редагування даних на випадок певних змін в інформаційному ресурсі.

Сторінка зміни даних повинна генеруватись на основі обраного запису в базі даних та містити форму для вводу, попередньо наповнену поточними значеннями бази даних. Таким чином, користувач не повинен вводити всі дані з нуля, а може лише виправити вже існуючу інформацію, таку як, для прикладу, орфографічні помилки.

Сторінка внесення нових даних може бути ідентичною сторінці редагування, за винятком того що поля введення повинні бути порожніми та готовими для введення нової інформації. Обидві сторінки повинні містити кнопку «зберегти», яка надсилає дані до серверу, де вони записуються в базу даних.

Сторінка звіту передбачає проведення перевірки на невідповідність відносно кожного об’єкту бази даних, виведення результатів, можливість швидкого переміщення до об’єктів, які не відповідають визначеним вимогам, а також функцію фільтрування записів за категорією та критерієм перевірки.

Також було додано інформативну сторінку, яка містить дані про розробника та призначення програмного застосунку.

Загальний дизайн веб-застосунку не вимагає складних та комплексних анімацій, візуальних ефектів та значних дизайнерських рішень. Навпаки, єдина вимога до дизайну – відсутність засмічення зайвими ефектами, задля простоти та зручності використання інтерфейсу.

3.1.2 Структура модуля сервера

Модуль сервера складається з наступних блоків:

- блок обробки запитів;
- блок валідації даних;
- ядро;
- блок взаємодії з базою даних;
- база даних

Почнемо з опису блоку обробки запитів. Даний блок отримує запити від графічного інтерфейсу користувача та обробляє їх, формуючи відповідь, направляючи дані до ядра програмного засобу та виконуючи переадресацію, направляючи користувача на певну веб-сторінку. Даний блок є простим набором функцій та інструкцій на основі типу запиту та його пункту призначення, що не обробляє дані, які передаються, натомість передаючи їх до ядра.

Блок валідації даних виконує перевірку правильності вводу користувацьких даних, формування запитів, що будуть виконані блоком зв’язку з базою даних, а також форматування цих даних у вигляд, придатний до завантаження в базу даних. Також цей блок відповідає за формування файлу в форматі «.csv», який користувач завантажує з серверу.

Ядро, в свою чергу, виконує основний масив функцій. Це і генерування веб-сторінки, і ініціалізація та проведення процесу вивантаження, підтримка програмного засобу шляхом створення зв’язків між блоками, передача даних до блоку валідації даних, тощо. Даний блок є найбільш об’ємним, адже містить основну частину програмної логіки застосунку.

Блок зв’язку з базою даних, як зрозуміло з назви, виконує функцію зв’язку з базою даних. Створення запитів, надсилання та отримання даних відповідно до запитів, отриманих від користувача виконуються в рамках даного блоку, який здатний функціонувати лише на основі даних та інструкцій, отриманих від попередніх блоків

Таким чином, було описано модулі клієнтської та серверної частини, їх окремі блоки, їх призначення та функціонал.

3.1.3 Побудова схеми роботи основних функцій

Програмний засіб, що виконує функцію підтримки аудиту шляхом збору даних про інформаційні ресурси, повинен мати чітку логіку та схему роботи, що описує алгоритм. Для побудови схеми необхідно спочатку описати алгоритм роботи веб-застосунку.

При переході за локальною адресою веб-застосунку користувач опиняється на головній сторінці, що містить такі посилання:

- Отримання звіту
- створення нового запису
- перегляд інструкції користувача
- перегляд деталей про конкретний запис

При виборі функції завантаження даних, програмний засіб повинен запусти процес отримання даних з бази даних, їх форматування, запис у файл та надіслати отриманий файл у відповідь на запит користувача. Для нормального функціонування програмного засобу переадресація на додаткові сторінки при виконанні цієї дії чи після неї не потрібна.

При виборі функції створення нового запису, програмний засіб виконує переадресацію на сторінку створення запису, що містить форму для внесення даних. Дана форма складається з 18 полів, що мають бути заповнені. Кожне поле має надпис, що вказує, які дані необхідно внести. Після введення, користувач повинен натиснуту кнопку зберегти. При натисканні даної кнопки надсилається POST запит з введеними користувачем даними. Ці дані вносяться в об’єкт, створений на основі моделі, як частина процесу роботи з MongoDB на основі Mongoose. Після перевірки даних, об’єкт надсилається в базу даних для запису, а користувач за допомогою переадресації опиняється на головній сторінці. Оскільки головна сторінка генерується на основі об’єктів в базі даних, вона міститиме нове посилання на об’єкт, створений користувачем.

При натисканні на будь-який запис на головній сторінці користувача переносить на сторінку з детальним описом інформаційної системи, який він

обрав. Програмний засіб робить запит до бази даних, шукаючи обраний об’єкт за ідентифікатором, та на основі отриманого об’єкту генерує веб-сторінку з усіма даними про інформаційну систему. Також, на цій сторінці повинні бути кнопки внесення змін та видалення об’єкту з бази даних.

Кнопка внесення змін переносить користувача на сторінку, по функціоналу схожу на сторінку внесення даних. Всі поля форми, що розміщена на даній сторінці, повинні бути заповнені поточними даними про обраний об’єкт, для того щоб скоротити час внесення змін. При натисканні користувачем на кнопку збереження, надсилається POST запит з внесеними даними, які сервер формує в об’єкт, що надсилається в базу даних в формі запиту на оновлення. По завершенню внесення змін користувач повинен опинитись на сторінці з деталями про запис, який він редагував. Оскільки контент сторінки є динамічним, користувач одразу побачить зміни в даних про інформаційну систему.

При натисканні користувачем на кнопку видалення запису програмний засіб надсилає запит до бази даних про видалення запису за певним ідентифікатором. Варто зауважити, що у всіх випадках взаємодії з конкретним записом ідентифікатор також використовується у вигляді адреси веб-сторінки з детальною інформацією про кожен окремий запис.

Після словесного опису алгоритму роботи програмного запису, перейдемо до створення схем окремих функцій програмного засобу, починаючи з алгоритму отримання та обробки даних, що зображено на рис. 3.2:

Рисунок 3.2 – Алгоритм отримання і обробки даних

Даний алгоритм виконує отримання користувацьких даних про новий інформаційний ресурс, перевірку даних, їх запис в базу даних, а також генерування головної сторінки на основі отриманих даних, після чого виконує

переадресацію користувача, надсилаючи йому головну сторінку, що містить дані про всі інформаційні ресурси, включно зі щойно створеним записом.

сДалі розглянемо алгоритм формування звітів з відповідності, що зображено на рис. 3.3.

Рисунок 3.3 – Алгоритм формування звіту

Даний алгоритм виконує запит до бази даних, після чого перевіряє отримані дані, та, якщо вони є цілісними та відповідають моделі, формує на основі існуючих полів файл формату «.csv», заповнює його об’єктами, отриманими з

бази даних, та надсилає заповнений файл користувачу разом з відповіддю на запит.

3.2 Вибір інструментів розробки

Оскільки вибір таких засобів розробки, як середовище розробки та додаткові зовнішні інструменти заснований в основному на виборі мови програмування, що використовується, процес розгляду інструментів розробки необхідно почати саме з вибору мови програмування.

3.2.1 Вибір мови програмування

На даний момент в сфері веб-розробки існує поділ на фронтенд та бекенд, які описують зовнішній графічний інтерфейс користувача та логічний блок, що обробляє інформацію та презентує графічний інтерфейс користувачу, відповідно [30].

Для створення фронтенду веб-застосунку використовують в основному мову розмітки тексту HTML та каскадні таблиці стилів CSS. Більшість інших мов програмування просто використовує їх внутрішньо, інтегруючи фронтенд в бекенд, що має як власні переваги та недоліки.

HTML, або Hyper Text Markup Language – мова розмітки тексту, створена в 1991 році, стала міжнародним стандартом [31] в 2000 році. Основою даного стандарту є поділ тексту на окремі теги, що не відображаються в браузері, проте використовуються ним для інтерпретації вмісту сторінки.

Основною перевагою HTML є сумісність з усіма сучасними браузерами. Затверджений як офіційний міжнародний стандарт ISO/IEC 15445, він диктує розвиток всіх браузерів, а різні його специфікації використовуються в 96,5% веб-сторінок [32].

Оскільки HTML небезпідставно вважається найбільш вживаною мовою розмітки тексту для побудови веб-застосунків, для розробки інструменту підтримки аудиту було вирішено обрати саме її. Це дозволить забезпечити сумісність з усіма сучасними браузерами.

HTML зазвичай використовується з CSS – каскадною таблицею стилів, що дозволяє видозмінювати зовнішній вигляд веб-сторінки, працюючи на основі HTML-тегів, класів, ідентифікаторів та імен.

CSS був створений в 1997 році, як доповнення до мови розмітки тексту HTML, та рекомендується W3C (World Wide Web Consortium) до використання як заміна базовим функціям оздоблення, що присутні в HTML.

Безсумнівною перевагою CSS є його сумісність з HTML, проте на додаток до цього, каскадні таблиці стилів дозволяють дуже детально налаштовувати графічний інтерфейс та зовнішній вигляд веб-застосунку за допомогою широкого функціоналу. Для прикладу, такий підхід до побудови як flexbox дозволяє адаптивно налаштовувати великі сторінки зі значною кількістю інформаційних блоків, зручно розташовуючи їх на сторінці в залежності від групування.

Безпосередні функції CSS, що будуть використані для розробки програмного засобу, будуть визначені в процесі розробки виходячи з вимог та функціоналу вказаного засобу. Проте, вибір CSS як інструменту побудови графічного інтерфейсу є очевидним, як з огляду на його сумісність з мовою розмітки тексту HTML, так і враховуючи окремий функціонал каскадних таблиць стилів.

Варто зауважити, що ані HTML, ані CSS не є Тюринг-довершеними правилами маніпуляції даними, що не дозволяє назвати їх повноцінними мовами програмування та, як наслідок, породжує необхідність в використанні додаткових мов програмування для створення бекенду програмного застосунку.

Бекенд, як частина веб-засобу, що виконуватиме збір даних про інформаційні системи підприємства, є більш комплексною проблемою, з урахуванням того, що більша частина програмної логіки буде написана саме на обраній мові програмування. Таким чином, необхідно детально розглянути питання вибору мови програмування для розробки як серверної, так і клієнтської частини програмного засобу.

Оскільки основою більшості веб-застосунків у світі на даний момент є мова розмітки HTML, більшість мов програмування мають певну сумісність з нею, що дозволяє розробити програмний засіб майже на будь-якій сучасній мові. Постає питання, які вимоги висуваються до мови програмування та які критерії необхідно враховувати при виборі.

Першочерговою вимогою є сумісність з мовою розмітки HTML. Очевидним є, що в епоху Web3 більшість мов програмування матимуть той чи інший рівень сумісності з HTML, проте досі існують такі мови, як, наприклад, C та Assembler, імплементація HTML в які потребуватиме значних та невиправданих затрат.

До інших обов'язкових вимог можна віднести наступні:

- Швидкість роботи
- Можливість роботи зі значними обсягами даних
- Можливість виконувати основні функції веб-застосунку без додавання надмірної кількості сторонніх модулів
- Широкий функціонал

Також варто зауважити, що існують і додаткові вимоги до мови програмування, такі як зручність розробки, читабельність, можливість відстеження помилок, тощо. Всі ці функції не мають конкретного прямого впливу на результат роботи – тобто сам програмний засіб, проте дозволять значно полегшити процес його розробки.

Найпопулярнішою мовою програмування, що використовується в веб-розробці, є JavaScript – мова скриптів, створена спеціально для взаємодії з HTML та CSS. JavaScript відрізняється простим синтаксисом, швидкою роботою та наявністю великої кількості модулів, створених для вирішення різноманітних задач. Дана мова програмування беззаперечно є лідером в веб-розробці, хоча останнім часом JavaScript починає поступатись популярністю таким мовам як Python, PHP та навіть Java. Як окрема мова програмування JavaScript має певні недоліки, зокрема – в своєму функціоналі. З іншого боку, всі недоліки можна так чи інакше компенсувати певними фреймворками або бібліотеками. Для прикладу, не дивлячись на те, що JavaScript це здебільшого мова скриптів,

створена для роботи на клієнтській частині, деякі фреймворки, такі як Node.js, дозволяють охопити і серверну частину ввеб-застосунку.

Найбільш використовуваною мовою програмування бекенду веб-сторінок є PHP, або Hypertext Preprocessor – скриптова мова розробки веб-застосунків з відкритим вихідним кодом. Дана мова є здебільшого серверною, використовується для формування HTML-сторінки та надсилання її кінцевому користувачу. Це дозволяє генерувати динамічний контент, збирати дані та зберігати вихідний код сторінки та серверної частини веб-застосунку прихованою від користувача. До недоліків даної мови програмування можна віднести обмежений функціонал, вже мова PHP створена для роботи на серверній частині. З іншого боку, як і у випадку з JavaScript, більшість недоліків можна нівелювати за допомогою сторонніх бібліотек, що більш-менш зрівнює ці дві мови програмування між собою.

Python – мова програмування високого рівня, що означає, що вона є дуже зручною та простою для розробки, проте поступається низькорівневим аналогам як за швидкістю роботи так і за гнучкістю при розробці. Свою популярність мова програмування Python заслужила завдяки можливості роботи з великим обсягами даних – він використовується для навчання мовних моделей, нейромереж та інших виді в штучного інтелекту. Проте, для вирішення задачі розробки програмного засобу для збору даних про інформаційні системи такий функціонал не потребується, адже об’єм даних, що обробляються, є значно меншим ніж такий, при якому Python показує значну різницю в швидкодії, яка б перевершила недоліки даної мови програмування.

Java, що є однією з найпопулярніших мов програмування в світі, також може бути використана для побудови веб-застосунків, проте, аналогічно до Python, це мова високого рівня, що значно зменшує швидкість її роботи. Здебільшого, Java використовується для розробки десктопних застосунків, тобто застосунків, що працюють повністю на персональному комп’ютері користувача, залучаючи здебільшого локальні ресурси. Через це дана мова програмування не лише

страждає від повільної роботи, а й викликає такі проблеми як переповнення пам’яті через неоптимізований процес зберігання проміжної інформації.

Таким чином, вибір постає між мовами скриптів PHP та JavaScript. Враховуючи сучасні фреймворки, дані мови програмування є більш-менш однаковими в плані функціоналу та швидкодії, що робить вибір між ними менш суттєвим. Таким чином, незалежно від конкретного вибору мови програмування, варто зосередитись на мовах скриптів, що були створені безпосередньо для взаємодії з HTML.

З огляду на зручність використання та загальну структуру мови скриптів, було вирішено використати мову JavaScript. Це дозволить імплементувати всі необхідні функції веб-застосунку, не компрометуючи швидкість роботи та безпеку.

3.2.3 Вибір додаткових інструментів розробки

Для побудови повноцінного мережевого додатку, а саме серверної його частини, відомої як «бекенд», недостатньо використання лише стандартного функціоналу мови програмування. Існує безліч фреймворків, бібліотек, сервісів та інших інструментів які здатні як полегшити процес розробки так і покращити досвід використання мережевого застосунку. Для розробки інструменту підтримки аудиту, який буде зручним для використання та ефективним в зборі даних про інформаційні ресурси, необхідно звернути увагу на деякі з цих фреймворків.

Сучасна розробка мережевих додатків йде двома шляхами: створення сторінки, що відображається користувачу та обробка всієї логіки програмного засобу на сервері, або надсилання сторінки з підготовленими скриптами користувачу, для більш швидкої, хоча і менш захищенної роботи веб-сайту, яка, до того ж, збільшує навантаження та, відповідно, залежність від пристрою користувача.

Несумнівною перевагою підходу фокусування на серверній частині застосунку є велика обчислювальна потужність. Якщо всі обрахунки

відбуваються на сервері, не навантажуючи персональний комп’ютер користувача, загальна швидкість роботи застосунку значно збільшується. Тому, варто обрати фреймворк, який дозволить перенести всю логіку додатку на серверну частину, надсилаючи користувачу виключно сторінки з даними.

Для розробки інструменту підтримки аудиту було обрано такий фреймворк як Node.js. Node.js дозволяє перетворити JavaScript із мови скриптів, призначеної для роботи всередині HTML-сторінки на стороні користувача, на повноцінну мову програмування, що здатна підтримувати сервер. Таким чином можливо побудувати серверно-орієнтований веб-застосунок, що відповідатиме всім вимогам до інструменту підтримки аудиту.

На додаток до Node.js було вирішено використовувати також фреймворк Express. Це зручний додаток до Node.js, що дозволяє будувати сторінки з динамічним контентом, налаштовувати детальну взаємодію та переадресацію, а також надсилати користувачу готові сторінки, що не містять нічого окрім HTML-розмітки та стилів CSS.

Для роботи з нереляційною базою даних було обрано MongoDB. Це швидка система керування базами даних, що дозволяє використовувати як мережеве так і локальне підключення. Оскільки ніяких особливих функцій, окрім запису, читання та оновлення даних не потрібно, MongoDB повністю задовольняє всі потреби даного програмного засобу.

В якості фреймворку для роботи з MongoDB було вирішено використати Mongoose. Це зручний фреймворк для взаємодії з MongoDB, що дозволяє зробити процеси розробки та роботи програмного засобу, що взаємодіє з цією системою керування базами даних простим, швидким та надійним. В даному фреймворку присутні всі необхідні функції, що дозволяє отримувати записи з бази даних, фільтрувати або сортувати їх, створювати нові записи на основі підготовленої моделі, вносити зміни до існуючих записів та навіть видаляти їх. Загалом, Mongoose дозволить спростити процес розробки, не жертвуючи продуктивністю кінцевого продукту.

Для покращення процесу розробки також було використано бібліотеки Nodemon та Morgan. Nodemon дозволяє створювати сервер для роботи в режимі реального часу. Даний фреймворк дозволяє вносити зміни в програмний засіб під час роботи на спостерігати за результатом без затримок, автоматично перезавантажуючи сервер в фоновому режимі. Це дозволяє спростити процес розробки, та, особливо – відлагодження. Morgan – бібліотека для відстеження запитів, що надходять на сервер. Це утиліта відслідковування процесу роботи програмного засобу, що дозволяє спростити процес налаштування переадресації, налаштувати посилання та обробку запитів.

Отже, було визначено перелік бібліотек та фреймворків, необхідних для розробки програмного засобу. Також було визначено функціональні інструменти, що можуть покращити та пришвидшити процес розробки програмного засобу. Таким чином, можна перейти до розробки інструменту підтримки аудиту, що здійснює збір даних про інформаційні ресурси.

3.3 Розробка програмного засобу

В основі програмного застосунку, створеного за допомогою Node.js та express лежить декілька основних елементів. За допомогою Express можна розділити код програмного застосунку на окремі елементи. Дані елементи не обов’язково відповідають модулям програмного засобу, вказаним в попередніх підпунктах, адже Express слідує власній чіткій моделі побудови, проте поділ програмного засобу на блоки всередині даних чітко визначених функціональних модулів є можливим.

Основними елементами фреймворку є:

- набір шаблонів для HTML-сторінок;
- контролер, що виконує функцію ядра;
- «app.js», головний файл, що містить більшу частину програмної логіки, поєднуючи блоки обробки запитів, валідації даних та взаємодії з БД;
- Додаткові бібліотеки, такі як моделі для об’єктів бази даних, роутер, тощо;

Набір шаблонів для HTML-сторінок являє собою набір файлів з розширенням «.ejs», які можуть містити як HTML-розмітку так і код JavaScript, який за допомогою Express конвертується в повноцінну сторінку, перетворюючи JavaScript код на частину розмітки, таким чином генеруючи динамічний контент.

Всі сторінки веб-застосунку містять навігаційну панель, що складається з посилання на головну сторінку, посилання на створення нового запису та на сторінку, що містить інструкцію до роботи з програмним засобом. Додавання навігаційної стрічки виконується за допомогою Express та елементу <%- include("./partials/nav.ejs") %>, який фактично виконує ін’єкцію коду в сторінку, «вписуючи» навігаційну панель в код сторінки перед її формуванням. Це дозволяє створити одну навігаційну панель для всіх сторінок та використовувати її, не копіюючи код, що повторюється, в кожен файл. Таким чином, зберігається чистота коду.

Основний блок контенту, що міститься на головній сторінці, містить фрагмент коду, який створює новий елемент <a> для кожного запису в базі даних:

```
    <% resources.forEach(resource => { %>
  <a class="single" href="/resources/<%= resource._id %>">
    <h3 class="title"><%= resource.name %></h3>
    <p class="snippet"><%= resource.description %></p>
  </a>
<% }) %>
```

В даному фрагменті коду є скрипт, написаний мовою JavaScript, що починається з команди «blogs.forEach». Дана команда запускає цикл, що для кожного елементу масиву «resources» створює елемент <a>, що складається з назви інформаційного ресурсу та його опису.

Таким чином, вміст сторінки генерується динамічно на основі записів в базі даних. В результаті маємо веб-сторінку, що містить в собі навігаційну стрічку, кнопку завантаження та посилання на детальну інформацію про кожен запис. Головна сторінка застосунку зображена на рис. 3.4.

Рисунок 3.4 – Головна сторінка веб-застосунку

Сторінка деталей про інформаційний ресурс генерується лише за умови вже існування обраного інформаційного ресурсу. Таким чином, необхідності в додаткових функціях на кшталт «forEach» немає.

Замість цього сторінка використовує теги <p> з динамічними значеннями, що отримуються з об’єкту. Через те, що даний скрипт обробляється на сервері, дані перетворюються на рядок до того, як потрапити на клієнтську частину застосунку, адже клієнт отримує повністю сформований HTML-файл. Для прикладу розглянемо фрагмент коду, що описує генерування показника конфіденційності при описі інформаційного ресурсу:

```
<div class="details-component">
            <p class="headerDetails">Конфіденційність:</p>
            <p>
              <%= resource.confidentiality %>
            </p>
          </div>
```

Як видно на фрагменті коду, поле «resource.confidentiality» використовуються для наповнення тегу <p>. Таким чином, один і той самий шаблон здатен генерувати сторінку для будь-якого запису з бази даних, як зображено на рис. 3.5

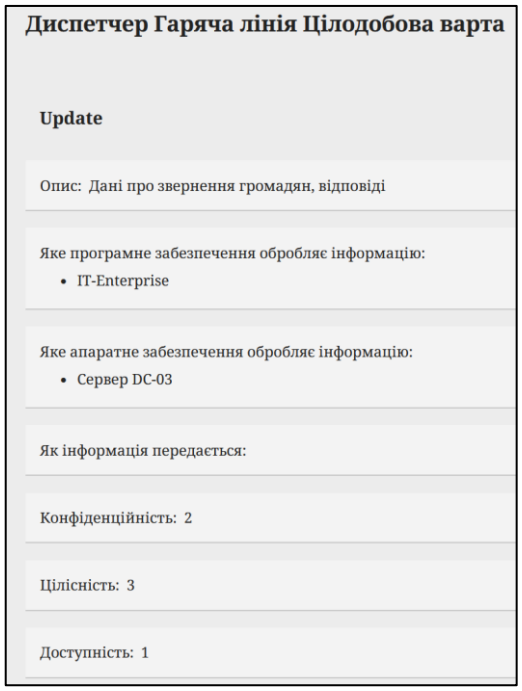


Рисунок 3.5 – Сторінка деталей про запис

Сторінка введення даних містить форму для введення даних та кнопку підтвердження, так само як і стрічку навігації. Стрічка навігації, тобто «header», додається аналогічно до «footer», за допомогою функції Express «include». Таким чином, досягається використання одного і того ж блоку коду для кожної сторінки:

```
<body><%- include("./partials/nav.ejs") %>
```

В результаті отримуємо сторінку з формою для введення та стандартними для веб-застосунку елементами, як зображено на рис. 3.6

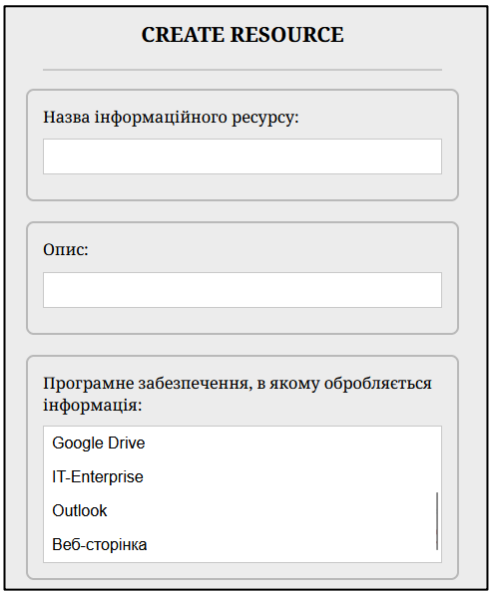


Рисунок 3.6 – Сторінка введення даних

Внутрішня логіка програми реалізована мовою JavaScript з використанням Node.js та Express. Таким чином, функції обробки даних та реагування на запити написані за допомогою методів даних фреймворків.

Для прикладу, обробка запитів використовує методи Express «get», «post» та «delete» для перехоплення користувацьких запитів та їх переадресації або генерування сторінок:

Вказуючи відносні URL-адреси сторінок в якості параметрів відповідних функцій можна перехоплювати запити, що надходять за цими адресами та запускати відповідні функції.

Для прикладу, при надсиланні запиту типу «GET» на адресу «/createResource» запускається метод «createResource», який відповідає за генерування сторінки внесення нового запису в базу даних.

Даний метод використовує функцію «render» для генерування сторінки та надсилання її кінцевому користувачу:

```
    .then(([resources, employees, softwares, hardwares, transfer]) => {
        res.render('resource/createResource', { resources, employees,
softwares,hardwares,transfer,  title: 'Create resource' });
```

В результаті користувач отримує HTML-сторінку з формою для введення даних. Аналогічним чином, при натисканні кнопки «Submit» надсилається запит типу «POST» на адресу «/resources». Під час обробки запиту викликається метод «resourceSave», який використовує метод «save» для збереження даних ,після чого виконує переадресацію на головну сторінку:

```
const resource = new Resource(req.body);
resource.save()
  .then(result => {
    res.redirect('/resources');
  })
```

Відображення деталей про інформаційний ресурс виконується за допомогою асинхронної операції вивантаження записів з бази даних. Метод «finById» запускає асинхронний процес отримання об’єкту з бази даних. Метод «populate» також вивантажує вкладені об’єкти, що дозволяє отримати інформацію не тільки про інформаційний ресурс, а і про пов’язані з ним елементи, наприклад, програмне забезпечення або працівники, що мають

відповідний доступ. За допомогою методу «then» генерування сторінки на основі динамічного контенту починається тільки після того як контент було отримано. Таким чином, формується сторінка, заповнена детальною інформацією про конкретний об’єкт.

```
Resource.findById(req.params.id)
    .populate('software')
    .populate('hardware')
    .populate('transfers')
    .populate('owner')
    .populate('responsible')
    .populate('workingUsers')
    .sort({ name: 1 })
    .then(resource => {
      res.render('resource/resourceDetails', {
        resource,
        title: 'Resource details',
      });
    })
```

Після отримання даних, на основі шаблону формується сторінка. За допомогою функціоналу Express існує можливість вбудовувати параметри, що передаються при генеруванні сторінки, в HTML-код. Для прикладу, використовуючи синтаксис <%=x%>, можна вставити змінну «x» в текст. Таким чином, при генеруванні сторінки вона автоматично наповнюється даними про обраний об’єкт.

Формування сторінки для внесення змін відбувається шляхом використання шаблону створення елементу та попереднього заповнення полів для вводу поточними значеннями. Для цього використовується Express, за допомогою якого заповнюється параметр «value» елементу «input». Таким чином, при генеруванні сторінки, всі поля вже заповнені поточними значеннями та готові до редагування користувачем:

```
<div>
                        <label for="confidentiality">Конфіденційність:</label>
                        <input type="number" id="confidentiality" name="confidentiality"
min="1" max="3" required
                            value="<%= resource.confidentiality %>">
                    </div>
```

Генерування звітів виконується за допомогою набору перевірок, які звіряють об’єкти в базі даних з певними критеріями. Кожна окрема перевірка становить об’єкт, що містить наступні поля:

- поле або поля об’єкту, що перевіряються;

- набір функцій перевірки;
- опис критерію перевірки;
- індекс критерію перевірки.

Об’єкт масиву «checks» виглядає наступним чином:

```
{
            fields: ["confidentiality", "software.isAuthentication"],
            condition: (resource) => resource.confidentiality > 2 &&
                resource.software?.some(software => software.isAuthentication === false),
            description: "Конфіденційні дані обробляються в ПЗ без автентифікації",
            index: [0, 0]
        },
```

Поля об’єкту застосовуються в методі «performChecks», який вивантажує вказані поля об’єкту, наприклад, інформаційної системи. За допомогою набору функцій значення вивантажених полів порівнюються з зазначеними. У випадку знаходження невідповідності, відомості про об’єкт записуються в масив «results», який після проходження всіх перевірок, повертається як результат роботи функції. В масив «results» входять наступні поля:

- ім’я об’єкту;
- опис критерію перевірки;
- категорія об’єкту перевірки;
- ідентифікатор об’єкту перевірки;
- індекс критерію перевірки.

Перевірка на відповідність здійснюється за допомогою розбиття вхідних даних про поля, що перевіряються, на частини для отримання значень конкретних полів.

```
for (const part of fieldParts) {
                    if (Array.isArray(currentValue)) {
                        currentValue = currentValue.find(obj => obj[part] !== undefined);
                    } else if (currentValue) {
                        currentValue = currentValue[part];
                    } else {
                        currentValue = undefined;
```

Після цього запускається перевірка, яка записує результат перевірки у змінну «isConditionMet». Якщо перевірку на відповідність не було пройдено, відомості записуються в масив «results».

```
if (currentValue !== undefined && check.condition(resource)) {
                    isConditionMet = true;
                    break;
                }
```

```
                }
                if (isConditionMet) {
                    results.push({
                        name: resource.name || "Unknown Name",
                        check: check.description,
                        source: model.modelName,
                        id: resource.id,
                        index: check.index
                    });
```

Отриманий масив передається до сторінки «/report», де відбувається генерування сторінки звіту. Для цього використовуються згадані раніше функції фреймворку Express.

```
    <% if (resources.length> 0) { %>
                <% resources.forEach(res=> { %>
                  <%if(filter.length===res.index.length && filter.every((value, index)=> value
=== res.index[index])){%>
                    <a class="single" href="/<%=res.source%>s/<%= res.id %>">
                      <h3 class="title">
                        <%= res.name %>
                      </h3>
                      <p class="snippet">
                        <%= res.check %>
                          <%= res.nestedObjectValue %>
                      </p>
                    </a>
```

Як результат, формується сторінка, що не тільки містить в собі перелік невідповідностей в окремих об’єктах бази даних, а й посилання на об’єкти, за допомогою якого можна переглянути детальну інформацію або відредагувати некоректні дані.

Також існує можливість перегляду загального звіту по всім категоріям даних. Результат такої перевірки виводиться у вигляді сторінки з загальним переліком всіх невідповідностей.

Також підсистема підтримки аудиту містить функцію фільтрації невідповідностей. За допомогою дворівневого випадаючого списку обирається категорія та невідповідність, яка цікавить користувача. В результаті генерується сторінка, що містить дані тільки про ті об’єкти, в яких була виявлена дана невідповідність.

Це відбувається завдяки передачі індексу критерію, який в процесі генерування сторінки порівнюється з обраним користувачем фільтром. Якщо фільтри співпадають, дані про об’єкт потрапляють на сторінку. Результатом стає сторінка, що містить тільки обрані користувачем помилки, наприклад,

відсутність прав доступу до програмного забезпечення. Існуючі фільтри показані на рис. 3.7.

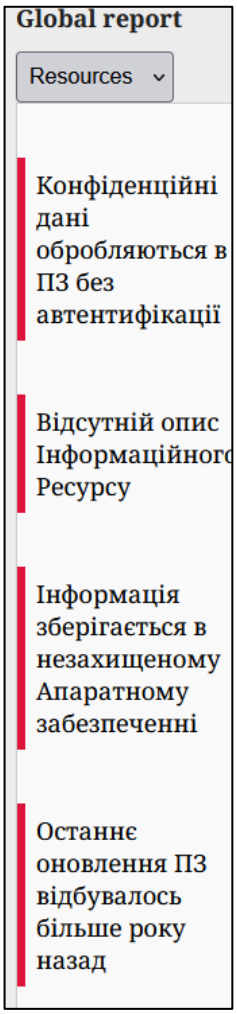


Рисунок 3.7 – Перелік фільтрів категорії «Інформаційні ресурси»

Як результат, було розроблено ефективний та зручний застосунок з веб-інтерфейсом, що може бути розміщений локально для проведення збору, систематизації та обробки даних про інформаційні ресурси та їх відповідність вимогам та стандартам.

3.4 Експериментальне дослідження підсистеми підтримки аудиту

Для підтвердження практичної користі розробленої моделі та дослідження ефективності роботи підсистеми підтримки аудиту необхідно чітко визначити вимоги до функціоналу даної підсистеми та провести перевірку відповідності

розробленого функціоналу даним вимогам. Для ефективної підтримки аудиту інформаційної системи необхідні наступні функції:

- внесення даних про інформаційні ресурси та пов’язані об’єкти
- перегляд списку записів кожної категорії;
- перегляд детальної інформації про кожен окремий запис;
- видалення записів;
- внесення змін до записів;
- перегляд детальної інформації про пов’язані записи;
- проведення перевірки записів на відповідність;
- виведення результатів перевірки на відповідність;
- фільтрування результатів перевірки за категорією та критерієм;

Таким чином, маємо перелік функцій підсистеми підтримки аудиту, які необхідні для ефективного збору, систематизації та обробки інформації. Перейдемо до експериментального дослідження.

Для визначення працездатності функції внесення даних необхідно заповнити базу даних об’єктами кожної з категорій. Для цього використаємо графічний інтерфейс веб-застосунку.

Для перевірки будуть внесені дані про інформаційний ресурс «Система колективної роботи та управління процесами», що обробляється програмним забезпеченням «Microsoft Teams». Дане програмне забезпечення розміщене на апаратному забезпеченні «Сервер DC-01», доступ до якого має інженер відділу Інформаційних технологій «Станіславський Є. А.». Метод передачі, який застосовується для даного інформаційного ресурсу – «Обмін повідомленнями в чаті». Перейдемо до внесення даних.

Для заповнення об’єкту необов’язково одразу встановлювати всі зв’язки. Підсистема підтримки аудиту підтримує часткове наповнення об’єкту даними та можливість встановлення зв’язків між двома об’єктами шляхом редагування будь-якого з них. Це дозволяє уникнути проблеми кругової залежності, де для внесення даних про програмне забезпечення необхідно вказати, яке апаратне забезпечення використовується для нього, а для внесення даних про апаратне

забезпечення необхідно вказати, яке програмне забезпечення на ньому розміщене.

Розпочнемо з заповнення даних про працівника, як зображено на рис. 3.8.

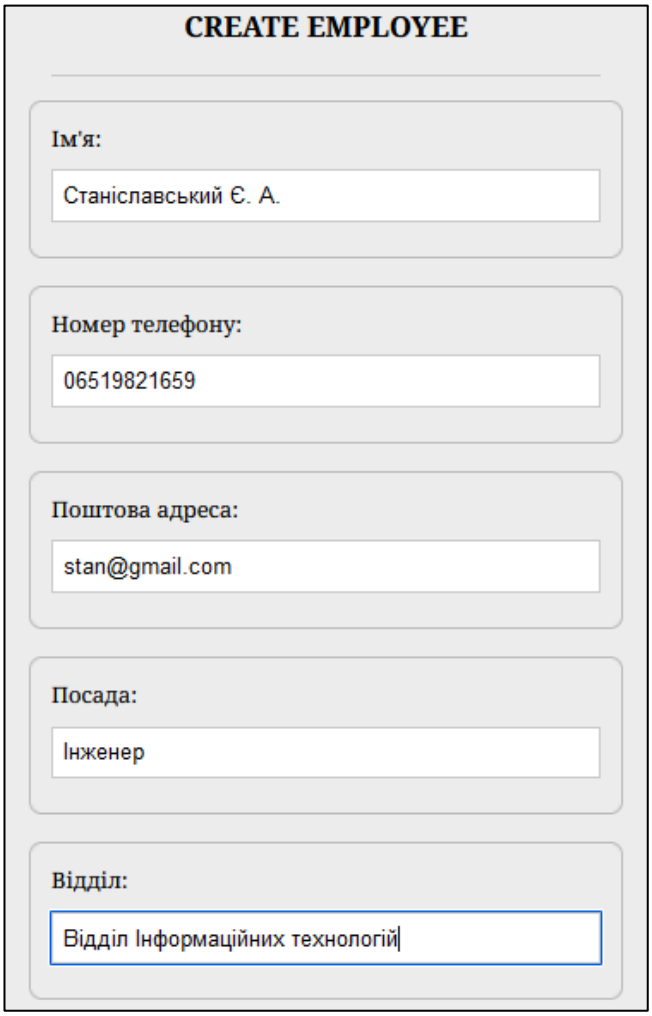


Рисунок 3.8 – Внесення даних про працівника

Заповнення решти полів, які реалізують зв’язки з іншими об’єктами, на даному етапі непотрібно. Об’єкти, зв’язки з якими необхідно налаштувати, відсутні в базі даних, а встановити зв’язки можна буде під час їх створення.

Після підтвердження вводу шляхом натискання кнопки «Submit», отримуємо новий об’єкт в базі даних, як показано на рис. 3.9.

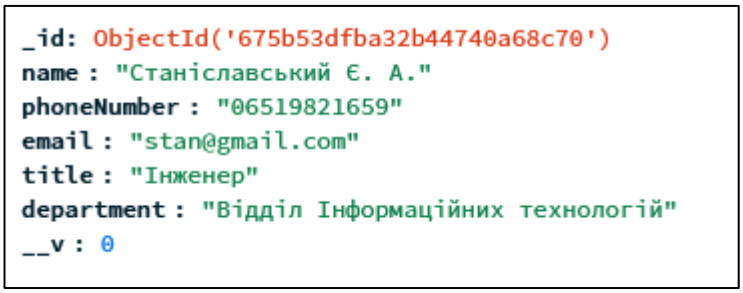


Рисунок 3.9 – Запис про працівника в базі даних

Як бачимо, внесення даних працює коректно та дозволяє користувачу вносити інформацію до бази даних за допомогою графічного інтерфейсу. Перейдемо до тестування можливості перегляду записів у вигляді списку.

Для цього перейдемо за посиланням, після чого очікується побачити перелік інформаційних ресурсів, дані про які внесені до бази даних. В результаті отримуємо сторінку, як зображено на рис. 3.10.

Рисунок 3.10 – Перелік інформаційних ресурсів

Таким чином, підтверджується можливість отримання переліку об’єктів кожної категорії.

Перейдемо до функції перегляду детальної інформації про кожен запис. Для цього в переліку записів необхідно натиснути на той, що цікавить користувача, що змінює сторінку на динамічно згенеровану сторінку детальної інформації про обраний об’єкт. Для прикладу, переглянемо дані про апаратне забезпечення «Сервер DC-01». Сторінка детальної інформації містить зв’язки з працівником «Станіславський Є. А.» а також з програмним забезпеченням «Microsoft Teams». Окрім цього, вона виводить дані інших полів, таких як «Місце розміщення» та «Тип пристрою».

Дана сторінка генерується окремо для кожного об’єкту кожної колекції. Таким чином, кожна окрема сторінка здатна відображати релевантні дані щодо будь-якої категорії. Також сторінки з детальною інформацією слугують для переходу до функцій видалення або внесення змін в існуючі записи.

Веб-застосунок відображає не тільки поля загального значення, а й зв’язки з іншими об’єктами. Це дозволяє не тільки розуміти зв’язки, а й зручно переходити до детальної інформації про ці записи. Для приладу, при натисканні на назву інформаційного ресурсу, який обробляється в даному програмному забезпеченні, відбувається переадресація на сторінку з детальною інформацією про даний інформаційний ресурс, як зображено на рис. 3.11.

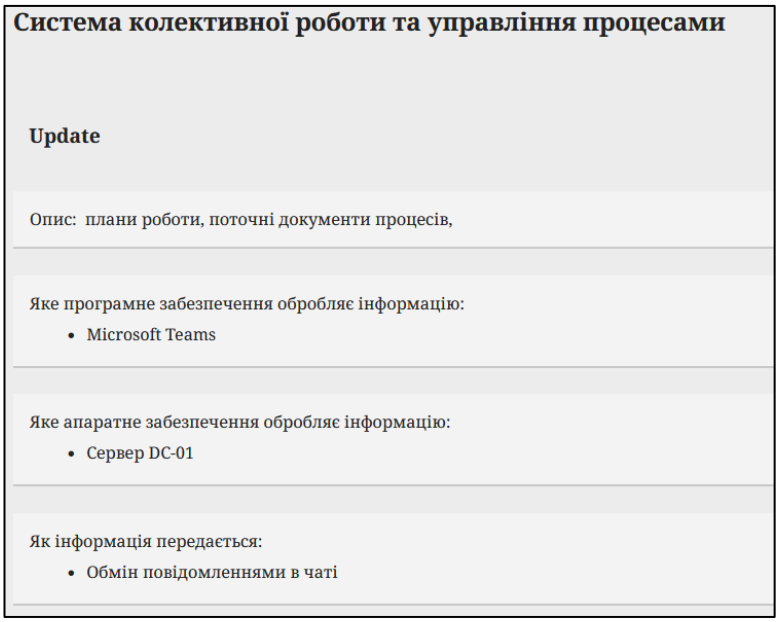


Рисунок 3.11 – Детальна інформація про інформаційний ресурс

Для видалення запису необхідно відкрити сторінку детальної інформації про запис та натиснути кнопку видалення. Після цього запис повинен зникнути з бази даних. В результаті видалення запису про працівника «Станіславський Є. А.», даний запис зник з бази даних, як показано на рис. 3.12

Рисунок 3.12 – Наслідок видалення запису

Внесення змін відбувається за допомогою кнопки «Оновити», яка присутня на кожній сторінці з детальною інформацією. При цьому відбувається перехід на сторінку, що є аналогічною сторінці внесення даних. Різниця полягає в тому, що всі поля заповнені поточними даними запису. Таким чином, є можливість вносити невеликі зміни до запису та коригувати значення полів, не видаляючи об’єкт чи створюючи новий.

Основною функцією підсистеми підтримки аудиту та веб-застосунку, що реалізує її є перевірка на відповідність. Шляхом створення ряду перевірок можна заздалегідь визначити норми, стандарти та вимоги до інформаційних ресурсів, програмного забезпечення та інших елементів моделі. Програмний застосунок повинен виконувати перевірки та виділяти об’єкти, які мають невідповідності.

Також необхідно виводити на екран звіт з відповідності для всіх об’єктів кожної категорії. Додатковою вимогою можна висунути необхідність виведення

звіту з всіх об’єктів всіх категорій одразу. Таким чином, користувач має вибір та гнучкі можливості для перегляду поточного стану інформаційних ресурсів та пов’язаних сутностей. Для прикладу, сторінка звіту з відповідності категорії «Апаратне забезпечення» зображена на рис. 3.13.

Рисунок 3.13 – Звіт з невідповідностей категорії «Апаратне забезпечення»

Для більш зручного користування функцією формування звіту, підсистема підтримки аудиту також містить функцію фільтрування звітів за критерієм. Таким чином, можна переглянути всі записи, які не відповідають конкретній вимозі, для того аби скласти звітність про результати роботи або розділити процес актуалізації інформаційних ресурсів на окремі кроки, які дозволять виставити пріоритети та спланувати процес виправлення помилок.

Робота функції фільтрування зображена на рис. 3.14.

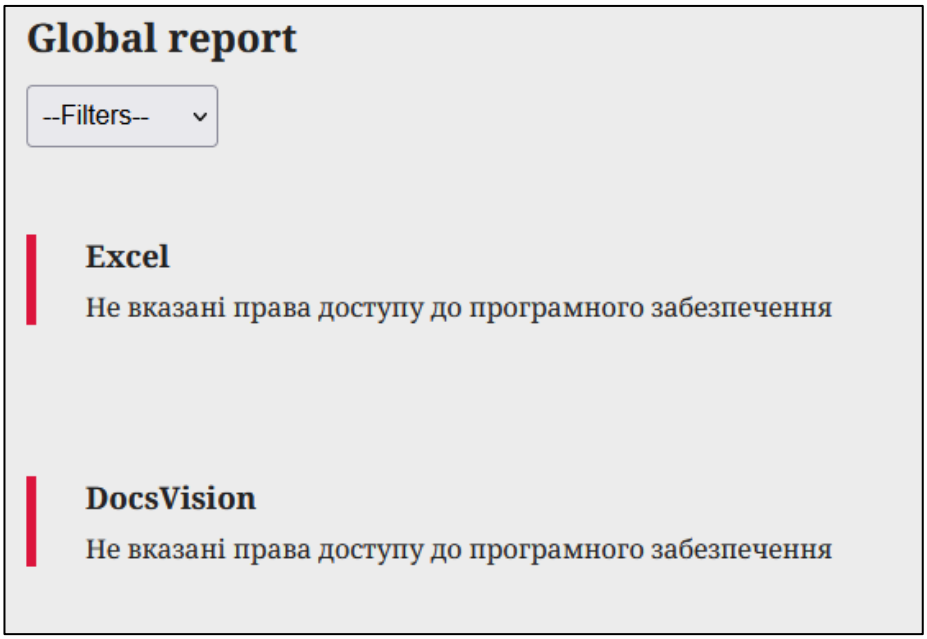


Рисунок 3.14 – Фільтрування за критерієм «Не вказані права доступу до ПЗ»

Таким чином, бачимо, що всі функції, яких потребує підсистема підтримки аудиту, реалізована за допомогою веб-застосунку, що розміщений в локальній мережі підприємства, реалізовані. Це дозволяє користувачам використовувати підсистему без будь-яких навичок в сфері інформаційних технологій та роботи з базами даних або сховищами даних у вигляді таблиць.

3.5 Висновки з розділу

В процесі виконання комплексної магістерської роботи було розроблено веб-застосунок, що реалізує підсистему підтримки аудиту. Дана підсистема використовує розроблену модель систематизації даних про інформаційні ресурси. Підсистема підтримки аудиту має такі функції як внесення, перегляд, видалення та коригування записів про інформаційні ресурси, програмне та апаратне забезпечення, методи передачі даних та працівників. Також присутні функції перевірки даних на відповідність вимогам, формування звіту з перевірки та фільтрування даного звіту за критеріями. Даний функціонал дозволяє

користувачам працювати з даними про інформаційні системи в більш зручному середовищі ніж звичайні підходи, для прикладу, Microsoft Excel.

Також, варто врахувати, що розміщення програмного засобу в локальній мережі підприємства є значною перевагою з точки зору захисту конфіденційної інформації або інформації з обмеженим доступом. Дані про інформаційні системи, їх недоліки та відповідність стандартам, беззаперечно, відноситься до конфіденційної інформації підприємства.

Використання розробленої підсистеми дозволяє підвищити ефективність та скоротити час виконання аудиту інформаційної безпеки за рахунок автоматизації процесу аналізу інформаційних ресурсів на відповідність. Програмний засіб має зручний користувацький інтерфейс, що значно зменшує мінімально необхідний для проведення інформаційного аудиту рівень знань та навичок, що дозволяє залучати до обробки даних менш кваліфікований персонал. В свою чергу, це дозволяє більш кваліфікованим спеціалістам виділити час на інші задачі аудиту. Таким чином, збільшується ефективність процесу.

Таким чином, було визначено, що програмний засіб відповідає вимогам, поставленим перед підсистемою підтримки аудиту, та ефективно виконує поставлені задачі, дозволяючи зменшити витрати часу, матеріалів та коштів.

4 ЕКОНОМІЧНА ЧАСТИНА

В процесі наукових досліджень необхідно враховувати як потенційні витрати на проведення дослідницького процесу, так і безпосередні результати, які визначають доцільність проведення дослідження. Отримані результати характеризують отриманий кінцевий продукт а також наукові знання, які можуть бути застосовані для подальшого розвитку науки.

Комплексна магістерська кваліфікаційна робота на тему «Аудит інформаційної безпеки департаментів Вінницької міської ради», а саме її друга частина, що описує розробку системи підтримки аудиту, відноситься до науково-технічних робіт, що впроваджується розробником на підприємстві «Вінницька міська рада». Основною метою цієї роботи є покращення процесу проведення аудиту шляхом розробки програмного застосунку, що виконує систематизацію та аналіз даних. Тобто, комплексна магістерська кваліфікаційна робота носить прикладний характер, та має на меті отримання технічного продукту, що буде в подальшому впроваджений на підприємстві. Для розрахунку доцільності та ефективності інвестицій, вкладених в дану роботу, необхідно провести такі етапи робіт:

1) проведення технологічного аудиту власної науково-технічної розробки, тобто встановлення її науково-технічного рівня (без комерціалізації);
2) розрахунок витрат на здійснення науково-технічної розробки;
3) розрахунок економічної ефективності науково-технічної розробки у випадку її впровадження розробником (замовником) на власному підприємстві та обґрунтування економічної доцільності впровадження розробником (замовником) розробленого у комплексній магістерській кваліфікаційній роботі науково-технічного проекту.

4.1 Технологічний аудит розробки

Технологічний аудит розробки проводиться за допомогою визначених критеріїв оцінювання науково-технологічного рівня та комерційного потенціалу розробки. Оцінювання проводиться за п’ятибальною системою з 12-ма критеріями оцінки. Таким чином, найвища можлива оцінка складає 100 балів. Критерії оцінювання включають:

- технічна здійсненність концепції;
- ринкові переваги (наявність аналогів);
- ринкові переваги (ціна продукту);
- ринкові переваги (технічні властивості);
- ринкові переваги (експлуатаційні витрати);
- ринкові перспективи (розмір ринку);
- ринкові перспективи (конкуренція);
- практична здійсненність (наявність фахівців);
- практична здійсненність (наявність фінансів);
- практична здійсненність (необхідність нових матеріалів);
- практична здійсненність (термін реалізації);
- практична здійсненність (розробка документів).

Для проведення оцінювання необхідно отримати оцінки експертів відповідно кожного критерія і обчислити середню арифметичну оцінку, яка визначатиме науково-технічний рівень та комерційний потенціал розробки.

Експертами виступають Лукічов Віталій Володимирович, доцент кафедри ЗІ, ВНТУ, Войтович Олеся Петрівна, доцент кафедри Захисту Інформації, ВНТУ, Каплун Валентина Аполінаріївна, старший викладач кафедри ЗІ, ВНТУ. Оцінювання проведено за показниками, що наведені в Додатку Д. Результати оцінювання наведено в таблиці 4.1:

Таблиця 4.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки

Критерії	Експерт		
	Лукічов.В.В.	Войтович О.П.	Каплун В.А.
	Бали		
1. Технічна здійсненність концепції;	4	4	4
2. Ринкові переваги (наявність аналогів);	2	2	2
3. Ринкові переваги (ціна продукту);	5	5	5
4. Ринкові переваги (технічні властивості);	4	3	3
5. Ринкові переваги (експлуатаційні витрати);	3	4	3
6. Ринкові перспективи (розмір ринку);	3	4	4
7. Ринкові перспективи (конкуренція);	2	3	2
8. Практична здійсненність (наявність фахівців);	5	4	4
9. Практична здійсненність (наявність фінансів);	5	5	5
10. Практична здійсненність (необхідність нових матеріалів);	4	4	5
11. Практична здійсненність (термін реалізації);	4	4	4
12. Практична здійсненність (розробка документів);	5	5	5
Сума балів:	46	47	46
Середньоарифметична сума балів:	46,33		

Середньоарифметична сума балів оцінюється за наступною формулою:

$$СБ_С = \frac{\sum_{i=1}^{3} СБ_i}{3} = \frac{46+47+46}{3} = 46{,}33 \qquad (4.1)$$

де $СБ_С$ – середньоарифметична сума балів; $СБ_i$ – сума балів і-го експерта.

За результатами оцінювання робиться висновок про науково-технічний рівень та комерційний потенціал розробки. Висновок ґрунтується на оцінках кожного рівня, що наведені в таблиці 4.2:

Таблиця 4.2 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вищий середнього
21…30	Середній
11…20	Нижчий середнього
0…10	Низький

В результаті бачимо, що розробка має високий потенціал. З комерційної точки зору, така оцінка обумовлена набагато нижчою ціною розробки, порівняно з аналогами, малою кількістю аналогів на помірному ринку а також відносно швидким терміном реалізації. З точки зору ефективності практичного

застосування, розробка має великий потенціал завдяки визначенні функціоналу відповідно до вимог підприємства. З точки зору технічної та практичної здійсненності, розробка має відносно високі експертні оцінки.

Проведемо також оцінку конкурентоспроможності розробки шляхом порівняння з аналогами. Для порівняння візьмемо найбільш популярний інструмент підтримки аудиту на ринку – ITAM-системою ninjaOne. Оскільки параметри оцінювання не є жорсткими та не можуть бути виміряні кількісно, оцінка якісних показників проводиться на п’ятибальною шкалою. Оцінювання проведено головою відділу Інформаційних Технологій Вінницької міської ради, Романенко Володимиром Борисовичем.

Для порівняння економічних показників буде використано 2 показники – ціна продукту та вартість підтримки роботи продукту протягом року.

Для обчислення одиничних параметричних індексів застосовується формула, як показано на приклад оцінки індексу зміни значення параметра «Захист даних»:

$$q_{\text{зд}} = \frac{P_{\text{баз зд}}}{P_{\text{зд}}} = \frac{4}{3} = 1{,}33 \qquad (4.2)$$

Результати порівняння науково-технічної розробки та аналогу на ринку наведено в таблиці 4.3.

Таблиця 4.3 – Оцінки якісних показників

Параметр	Одиниця виміру	Аналог	Нова розробка	Індекс зміни значення параметра	Коефіцієнт вагомості
Технічні					
Захист даних	5-бальна шкала	3	4	1,33	0,5
Рівень автоматизації	5-бальна шкала	4	3	0,75	0,3
Формування звітності	5-бальна шкала	4	4	1	0,1
Доступність	5-бальна шкала	4	5	1.25	0,1
Економічні					
Ціна	грн	50000	80000	1,66	0,4
Вартість підтримки (рік)	грн	50000	12000	0,24	0,6

Для визначення групового параметричного індексу за технічними показниками конкурентоспроможності застосуємо наступну формулу:

$$\mathrm{I_{ТП}} = \sum_{i=1}^{4} q_i * \alpha_i = 1.33 * 0.5 + 0.75 * 0.3 + 1 * 0.1 + 1.25 * 0.1 = 1.115 \quad (4.3)$$

Для визначення групового параметричного індексу за економічними показниками конкурентоспроможності застосуємо наступну формулу:

$$\mathrm{I_{ЕП}} = \sum_{i=1}^{2} q_i * \beta_i = 1{,}66 * 0{,}4 + 0{,}24 * 0{,}6 = 0{,}78 \quad (4.4)$$

Інтегральний показник конкурентоспроможності обчислюється за формулою:

$$\mathrm{К_{ІНТ}} = \mathrm{I_{НП}} * \frac{\mathrm{I_{ТП}}}{\mathrm{I_{ЕП}}} = 1 * \frac{1.115}{0{,}78} = 1.429 \quad (4.5)$$

Отриманий інтегральний показник $К_{ІНТ} = 1.429$ свідчить про перспективність конкурентоспроможності даної розробки

З огляду на оцінки експертів та оцінку конкурентоспроможності розробки, можна зробити висновок про високий науково-технічний рівень, комерційний потенціал та перспективу конкурентоспроможності. В такому випадку можна зробити висновок про потенційну користь від науково-технічної розробки.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

Для обрахунку витрат на здійснення науково-дослідної роботи необхідно розглянути такі статті витрат:

- витрати на оплату праці
- відрахування на соціальні заходи
- програмне забезпечення для наукових та експериментальних робіт
- амортизація обладнання
- накладні (загальновиробничі) витрати

Перейдемо до обрахунку витрат на оплату праці.

4.2.1 Витрати на оплату праці

Витрати на оплату праці можна умовно поділити на 2 категорії. Це виплати дослідникам, що проводять науково-дослідну роботу, що є підґрунтям для подальшої розробки програмного застосунку, що реалізує напрацьовану теорії. Витрати на заробітну плату дослідників розраховуються на основі формули:

$$З_{\text{п}} = \sum_{i=1}^{n} \frac{M_{ni} * t_i}{T_p} \quad (4.6)$$

де k – кількість посад дослідників, залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – кількість днів роботи конкретного дослідника, дн.;

T_p – середня кількість робочих днів в місяці, T_p=21…23 дні.

Розрахунки наведено в таблиці 4.3:

Таблиця 4.4 - Витрати на заробітну плату дослідників

Посада	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Кількість днів роботи	Витрати на заробітну плату, грн
Керівник проекту	25000	1136,36	4	4544
Провідний спеціаліст	19850	902,27	22	19850
Розробник	23000	1045,45	22	23000
Лаборант	10500	477,27	10	4772,7
Тестувальник	15750	715	3	2147
Всього				54313

Для обчислення витрат на заробітну плату робітників необхідно врахувати наступні статті витрат:

- погодинну тарифну ставку за виконання роботи;
- час роботи робітника на виконання роботи

Для визначення погодинного тарифу необхідно використати наступну формулу:

$$C_i = \frac{M_м * K_i * K_c}{T_р * t_{зм}} \quad (4.7)$$

де $M_м$ – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати (залежно від діючого законодавства), грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б);

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати. (табл. Б.1, додаток Б)

$T_р$ – середня кількість робочих днів в місяці, приблизно $T_р$ = 21…23 дні, було вирішено взяти середнє значення 22 дні;

$t_{зм}$ – тривалість зміни, згідно стандартизованого робочого дня дорівнює 8 годин.

Проведемо розрахунок погодинної ставки. Для розрахунків візьмемо мінімальну заробітну плату $М_м = 8000{,}00$ грн. В результаті отримані погодинні тарифні ставки для кожного виду робіт наведено разом з розрахунками витрат в таблиці 4.4

Таблиця 4.4 - Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна ставка, грн	Величина оплати на робітника, грн
Встановлення обладнання	3	2	1,10	75	225
Інсталяція програмного забезпечення	1,5	3	1,35	101,25	151,875
Підготовка дослідження	5	4	1,50	122,72	613,6
Розробка моделі	8	5	1,70	139	1112
Розробка застосунку	10	4	1,50	122,72	1227,2
Тестування	3,5	2	1,10	75	262,5
Всього:					3591,575

Після обчислення витрат на основну заробітну плату робітників необхідно також визначити величину витрат на додаткову заробітну плату дослідників та робітників, що складає 10 … 12% від суми витрат за основну заробітну плату дослідників та робітників. Візьмемо середнє значення 11%:

$$З_{дод} = (З_о + З_р) * \frac{н_{дод}}{100\%} = (54313{,}7 + 3591) * \frac{11}{100}\% = 6369{,}51 \qquad (4.8)$$

Таким чином, загальні витрати на оплату праці становлять 62610 грн.

4.2.2 Відрахування на соціальні заходи

Відрахування на соціальні заходи включає в себе відрахування внеску на загальнообов’язкове державне соціальне страхування та для здійснення заходів

щодо соціального захисту населення. Цей внесок обраховується як 22% від суми основної та додаткової заробітної плати дослідників та робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) * \frac{н_{зп}}{100\%} = (54313{,}7 + 3591 + 6369) * \frac{22}{100}\% = 14140 \quad (4.9)$$

де $н_{зп}$ – норма нарахування на заробітну плату.

Таким чином, бачимо, що сума відрахувань на соціальні заходи становить 14140 грн.

4.2.3 Програмне забезпечення для науково-технічних робіт

До даної статті витрат належать витрати на розробку та/або придбання спеціального програмного забезпечення, що необхідно для проведення досліджень та розробку продукту, а також витрати на проектування, формування та встановлення. Витрати на інсталяцію зазвичай складають 10 … 12% від вартості програмного забезпечення.

Балансова вартість програмного забезпечення обчислюється за наступною формулою:

$$В_{прг} = \sum_{i=1}^{k} Ц_{іпрг} * С_{прг} * К_і \quad (4.10)$$

Де $Ц_{іпрг}$ – ціна придбання одиниці програмного засобу цього виду, грн;

$С_{прг}$– кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

$К_і$ – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, (К і = 1,10…1,12). Для обчислень візьмемо $К_і = 1.10$;

k – кількість найменувань програмних засобів.

Таким чином, результати обчислень наведені в таблиці 4.5:

Таблиця 4.5 – Вартість програмних застосунків

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Коефіцієнт вартості інсталяції	Вартість з урахуванням інсталяції, грн
Atlas	1	4000	1,10	4400
Середовище розробки	1	2000	1,10	2200
Всього				6600

Таким чином, отримуємо вартість програмного забезпечення, що складає 6600 грн.

4.2.4 Амортизація обладнання, програмних засобів та приміщень

До статті «Амортизація обладнання, програмних засобів та приміщень» відносять амортизаційні відрахування по кожному виду обладнання, устаткування та інших приладів і пристроїв, а також програмного забезпечення для проведення науково-дослідної роботи, за його наявності в дослідній організації або на підприємстві. В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$\mathrm{А_{обл}} = \frac{\mathrm{Ц_б}}{\mathrm{Т_в}} * \frac{t_{\text{вик}}}{12} \tag{4.11}$$

де $\mathrm{Ц_б}$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{\text{вик}}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$\mathrm{Т_в}$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Проведені розрахунки наведено в таблиці 4.6:

Таблиця 4.6. – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання, місяців	Амортизаційні відрахування
Сервер	15000	2	2	1250
Ноутбук Acer	18000	2	1	666
Ноутбук Dell	17000	2	2	1416
Всього				3332

Таким чином, отримуємо вартість амортизації обладнання, яка складає 3332 грн.

4.2.5 Сировина та матеріали

Для обчислення ціни матеріалів використовується формула:

$$M = \sum_{i=1}^{n} H_j * Ц_j * K_j - \sum_{j=1}^{n} B_j * Ц_{в\ j} \qquad (4.12)$$

де H_j – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

K_j– коефіцієнт транспортних витрат, (Кj = 1,1);

B_j – маса відходів j-го найменування, кг;

$Ц_{в\ j}$ – вартість відходів j-го найменування, грн/кг.

Для обрахунку дані було зведено до таблиці 4.7. :

Таблиця 4.7 – Обчислення вартості матеріалів

Найменування	Ціна за одиницю, грн	Норма витрат	Величина відходів	Ціна відходів	Вартість втраченого матеріалу
Блокнот	80	2	0	0	176
Ручка	20	4	0	0	88
Папір	0,5	50	0	0	27,5
Всього					291.5

Вартість витраченого матеріалу, на прикладі першої позиції, обчислюється наступним чином:

$$H_j * Ц_j * K_j = 80 * 2 * 1{,}1 = 176 \quad (4.13)$$

В сумі витрати на матеріали становлять 291,5 грн.

4.2.6 Накладні (загальновиробничі) витрати

Накладні витрати розраховуються виходячи з основної заробітної плати дослідників та робітників, становлячи 100 … 150% від суми цих витрат. Сюди входять витрати, пов'язані з управлінням організацією, витрати на винахідництво та раціоналізацію, витрати на підготовку (перепідготовку) та навчання кадрів, витрати, пов'язані з набором робочої сили, витрати на оплату послуг банків, витрати, пов'язані з освоєнням виробництва продукції, витрати на науково-технічну інформацію та рекламу та ін.

Ці витрати обчислюються за формулою:

$$В_{нзв} = (З_0 + З_р) * \frac{Н_{нзв}}{100\%} = (54313{,}7 + 3591) * \frac{125}{100\%} = 70507 \quad (4.12)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

Якщо припустити, що накладні витрати становлять середнє значення діапазону, тобто 125%, в результаті обчислень отримаємо 70507 грн.

Загальні витрати на проведення науково-технічної роботи розраховуються як сума всіх попередніх статей витрат. Таким чином, використавши значення, отримані при попередніх розрахунках, можемо визначити, що загальні витрати на проведення науково-технічної роботи становлять 159143 грн, як зазначено в формулі:

$$В_{нзв} = З_0 + З_р + З_{дод} + M + З_н + В_{прг} + А_{обл} + В_{нзв} = (54313{,}7 + 3591 + 6369 + 14140 + 291{,}5 + 6600 + 3332 + 70507) = 159143. \quad (4.8)$$

Для визначення кінцевої вартості завершення науково-технічної роботи необхідно визначити етап виконання науково-технічної роботи. Дана науково-технічна розробка знаходиться на останніх етапах стадії розробки промислового зразка, тому буде застосовано коефіцієнт η=0,9. Таким чином, застосовуючи формулу отримуємо:

$$ЗВ = \frac{В_{заг}}{\eta} = 159143 / 0{,}9 = 176825 \quad (4.13)$$

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

Дана розробка може бути виведена на ринок для подальшої реалізації. Для оцінки економічної ефективності даної науково-технічної розробки необхідно врахувати такі показники як час впровадження розробки, очікуваний час отримання позитивних результатів для інвестора, величина попиту та суб'єкти цього попиту, ціна реалізації на ринку розробок з аналогічними функціями.

Для прикладу наведемо обчислення прибутку інвестора в першому році після реалізації:

$$\Delta П_i = (\pm \Delta Ц_0 * N + Ц_0 * \Delta N)_i * \lambda * \rho * \left(1 - \frac{\vartheta}{100}\right) = (80000 * 12 + 80000 * 0) * 0{,}8333 * 0{,}35 * (1 - 0{,}18) = 229590.8 \quad (4.14)$$

Далі необхідно розрахувати приведену вартість всіх чистих прибутків ПП за формулою:

$$ПП = \sum_{i=1}^{T} \frac{ΔП_i}{(1+\tau)^t} = \frac{229590}{1,15^1} + \frac{459181}{1,15^2} + \frac{688772}{1,15^3} = 199643 + 347206 + 452878 = 999729$$

(4.8)

Необхідно розрахувати також початкові інвестиції PV. Обчислення проводиться за формулою:

$$PV = k_{інв} * ЗВ = 2 * 176825 = 353651 \quad (4.15)$$

Також необхідно обчислити приведений дохід за формулою:

$$E_{абс} = ПП - PV = 999729 \ - 353651 = 646725 \quad (4.16)$$

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність $E_в$ або показник внутрішньої норми дохідності (IRR, Internal Rate of Return) вкладених інвестицій за формулою:

$$E_в = \sqrt[Тж]{1 + \frac{E_{абс}}{PV}} - 1 = \sqrt[3]{1 + \frac{646725}{353004}} - 1 = 0.4148190 \quad (4.17)$$

Для визначення бар'єрної ставки, з якою необхідно порівняти внутрішню економічну дохідність, використовується формула:

$$\tau_{мін} = d + f = 0.3 + 0.1 = 0.4 \quad (4.18)$$

Оскільки внутрішня економічна дохідність перевищує мінімальну, потенційний інвестор можу бути зацікавлений в даній розробці.

Також обчислимо період окупності інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в} = \frac{1}{0.4148190} = 2.41 \quad (4.19)$$

Таким чином, термін окупності складає 2 роки та 5 місяців. Даний термін є привабливим для інвесторів.

4.4 Висновки з розділу

Отже, для визначення економічної ефективності науково-технічної розробки було проведено оцінювання науково-технічного рівня. Оцінювання проводилось трьома експертами, в результаті чого було отримано оцінку 46,33, що дозволяє оцінити розробку як таку, що має високий потенціал. Було проведено аналіз рівня конкурентоспроможності розробки, в результаті якого було визначено переваги технічних та економічних показників розробки порівняно з аналогами та інтегральний показник конкурентоспроможності, який складає 1,429. Також було проведено розрахунок витрат на здійснення науково-дослідної роботи та розрахунок економічної ефективності розробки. В результаті було визначено, що науково-технічна розробка є привабливою для інвесторів як з точки зору прибутку так і з точки зору терміну окупності. Зокрема, прогнозована внутрішня економічна дохідність становить 0,414819, а термін окупності складає 2 роки та 5 місяців.

ВИСНОВКИ

Питання покращення процесу аудиту залишається надзвичайно актуальним. Зокрема, процес збору, систематизації, обробки та аналізу даних про інформаційні ресурси, який, на сьогоднішній день, все ще потребує участі експертів, які здатні оцінити відповідність інформаційних ресурсів вимогам стандартів.

Ця частина комплексної магістерської кваліфікаційної роботи присвячена покращенню даного процесу. Першим кроком був аналіз сучасних тенденцій проведення аудиту інформаційної безпеки.

В результаті аналізу визначено, що більшість організацій, що надають послуги з проведення аудиту та сертифікації, використовують застарілі методи, що вимагає участі висококваліфікованих працівників. Таким чином, збільшуються витрати коштів та часу. Також, через надмірну залежність від людського фактору збільшується вірогідність помилок, що може призвести до неактуальних результатів аудиту.

Визначено, що попри існування систем підтримки аудиту, більшість з них мають певні недоліки, які так чи інакше компрометують конфіденційність інформації, не надають достатніх переваг, які можуть виправдати їх використання або не підходять для використання за дизайном чи функціоналом. Для прикладу, найбільш популярні сервіси менеджменту активів інформаційних технологій, або ITAM-системи, здебільшого надають свої послуги у вигляді сервісів SaaS, що при роботі з конфіденційною інформацією держустанов чи інших підприємств критичної інфраструктури, є надмірним ризиком з точки зору конфіденційності інформації, що обробляється.

Результатом аналізу стала розроблена модель систематизації даних, що полягає в розділенні даних про інформаційні ресурси на п’ять категорій, до яких входять дані про інформаційні ресурси, дані про апаратне забезпечення, дані про працівників, дані про методи передачі інформації та дані про програмне

забезпечення. Також модель включає в себе функції обробки даних та формування звітів з відповідності. Таким чином, модель дозволяє ефективно обробляти дані про інформаційні системи та формувати звіти про відповідність визначеним заздалегідь стандартам.

Для реалізації підсистеми підтримки аудиту було розроблено веб-застосунок, що реалізує розроблену модель за допомогою веб-інтерфейсу та серверу, що може бути розміщений в локальній мережі підприємства. Даний підхід дозволяє забезпечити доступність програмного застосунку будь-якому користувачу внутрішньої мережі підприємства, в той же час не передаючи інформацію за межі даної мережі. Таким чином, працівники підприємства, які не мають досвіду роботи в сфері інформаційних технологій або збору та обробки даних для аудиту інформаційної безпеки, можуть вносити дані про інформаційні системи, які вони використовують. В той же час адміністратор системи може переглядати звіт, що автоматично оновлюється залежно від нових записів.

Розроблена підсистема підтримки аудиту, яка реалізує модель систематизації даних на основі веб-інтерфейсу являє собою прийнятний компроміс між доступністю хмарних технологій та захистом конфіденційності, що забезпечується локальними інструментами підтримки аудиту. Наявний функціонал є корисним для підвищення ефективності процесу аудиту інформаційної безпеки та зменшення витрат часу та коштів на проведення аналізу відповідності інформаційних ресурсів стандартам кібербезпеки.

Таким чином, можна сказати, що завдання комплексної магістерської кваліфікаційної роботи було виконано повністю. Мету роботи, а саме є покращення та пришвидшення процесу аудиту інформаційної безпеки, було досягнуто.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Радченко Є.В., Пилявець І.Ю. Використання спеціальних технічних засобів отримання інформації: сутність та особливості унормування/ Наук. керівник Л.О. Майданевич. Вінниця, ВНТУ 2024 р. URL: https://conferences.vntu.edu.ua /index.php/mn/mn2024/paper/viewFile/21640/17886 (дата звернення: 13.12.2024)

2. Пилявець І.Ю., Радченко Є.В. Використання штучного інтелекту в аудиті інформаційної безпеки/ Наук. керівник О.П. Войтович. Вінниця, ВНТУ 2024. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/viewFile/22704/187 43 . (дата звернення: 13.12.2024)

3. Радченко Є.В., Пилявець І.Ю. Актуальність використання локальних інструментів підтримки аудиту/ Наук. керівник О.П. Войтович. Вінниця, ВНТУ 2024. Режим доступу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2021/paper/view/22918

4. ДСТУ ISO/IEC 27001:2023. ДСТУ ISO/IEC 27001:2023 Інформаційна безпека, кібербезпека та захист конфіденційності. Системи керування інформаційною безпекою. Вимоги (ISO/IEC 27001:2022, IDT). На заміну ДСТУ ISO/IEC 27001:2015 ; чинний від 2023-08-22. Вид. офіц. 2023.

5. ISO - International Organization for Standardization. ISO. URL: https://www.iso.org/home.html (дата звернення: 05.11.2024).

6. ISO 27001 vs. NIST Cybersecurity Framework Comparison. PrivacyEngine Data Protection Software and Consultancy. URL: https://www.privacyengine.io/blog/iso-27001-vs-nist-cybersecurity-framework/ (дата звернення: 05.11.2024).

7. NIST G. M. NIST Cybersecurity Framework 2.0:. Gaithersburg, MD : National Institute of Standards and Technology, 2024. URL: https://doi.org/10.6028/nist.sp.1301 (дата звернення: 05.11.2024).

8. Akbar B. I., Sucahyo Y. G., Gandhi A. Disaster recovery plan design in energy government agency using NIST SP 800-34 guidelines. THE 2ND INTERNATIONAL CONFERENCE OF SCIENCE AND INFORMATION TECHNOLOGY IN SMART ADMINISTRATION (ICSINTESA 2021), м. Balikpapan, Indonesia. 2022. URL: https://doi.org/10.1063/5.0105726 (дата звернення: 05.11.2024).

9. adcyber. How Excel boosts cyber security analytics? - Cyber Insight. Cyber Insight. URL: https://cyberinsight.co/how-excel-is-used-in-cyber-security/ (дата звернення: 05.11.2024).

10. OANCEA C.-V. Cybersecurity in Cloud Computing Context. International Journal of Information Security and Cybercrime. 2014. Т. 3, № 2. С. 43–48. URL: https://doi.org/10.19107/ijisc.2014.02.05 (дата звернення: 05.11.2024).

11. Emley B. IT audit: The ultimate guide [with checklist] | Zapier. Automate without limits | Zapier. URL: https://zapier.com/blog/it-audit/ (дата звернення: 05.11.2024).

12. Microsoft Excel. Microsoft. URL: https://www.microsoft.com/en-us/microsoft-365/excel (дата звернення: 05.11.2024).

13. Microsoft Access. Microsoft. URL: https://www.microsoft.com/en-us/microsoft-365/access (дата звернення: 05.11.2024).

14. Power BI - Data Visualization | Microsoft Power Platform. Microsoft. URL: https://www.microsoft.com/en-us/power-platform/products/power-bi/ (дата звернення: 05.11.2024).

15. Noble W. Automation for reports, charts, and dashboards | Zapier. Automate without limits | Zapier. URL: https://zapier.com/blog/reports-charts-and-dashoards/ (дата звернення: 05.11.2024).

16. Martins L. M. O. P. R. Software asset management in an organization : master's thesis. 2015. URL: http://hdl.handle.net/10071/11184 (дата звернення: 05.11.2024).

17. IT Asset Management / M. Stone та ін. ; ред. L. Kauffman. U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, 2018. 47 с.

18. SAP Solutions for Governance, Risk, and Compliance (GRC). SAP Help Portal. URL: https://help.sap.com/docs/SAP_GRC (дата звернення: 05.11.2024).

19. IT Audit Software from Netwrix. Netwrix. URL: https://www.netwrix.com/auditor.html (дата звернення: 05.11.2024).

20. Manufacturing Execution & Quality Management Software Solutions. MasterControl. URL: https://www.mastercontrol.com/ (дата звернення: 05.11.2024).

21. Cyber Risk Management Platform | LogicGate Risk Cloud. LogicGate. URL: https://www.logicgate.com/solutions/it-security-risk/ (дата звернення: 05.11.2024).

22. Fetters J. L. Sample Audit Reports. The Handbook of Lighting Surveys & Audits. 2018. С. 65–81. URL: https://doi.org/10.1201/9780203736647-7 (дата звернення: 07.11.2024).

23. United States. Defense Contract Audit Agency. Security inspections and surveys. Alexandria, Va : Defense Contract Audit Agency, 1991.

24. Artificial Intelligence Co-Piloted Auditing / H. Gu et al. SSRN Electronic Journal. 2023. URL: https://doi.org/10.2139/ssrn.4444763 (дата звернення: 07.11.2024).

25. Reddy R. Three open problems in AI. Journal of the ACM. 2003. вид. 50, № 1. с. 83–86. URL: https://doi.org/10.1145/602382.602407 (дата звернення: 07.11.2024).

26. Crawford K. The hidden costs of AI. New Scientist. 2021. вид. 249, № 3327. с. 46–49. URL: https://doi.org/10.1016/s0262-4079(21)00524-8 (дата звернення: 07.11.2024).

27. Grant N. Google's A.I. Search Errors Cause a Furor Online. New York Times. URL: https://www.nytimes.com/2024/05/24/technology/google-ai-overview-search.html (дата звернення: 07.11.2024).

28. Broady D. V., Roland H. A. SAP GRC for Dummies. Wiley & Sons, Incorporated, John, 2011. 342 с.

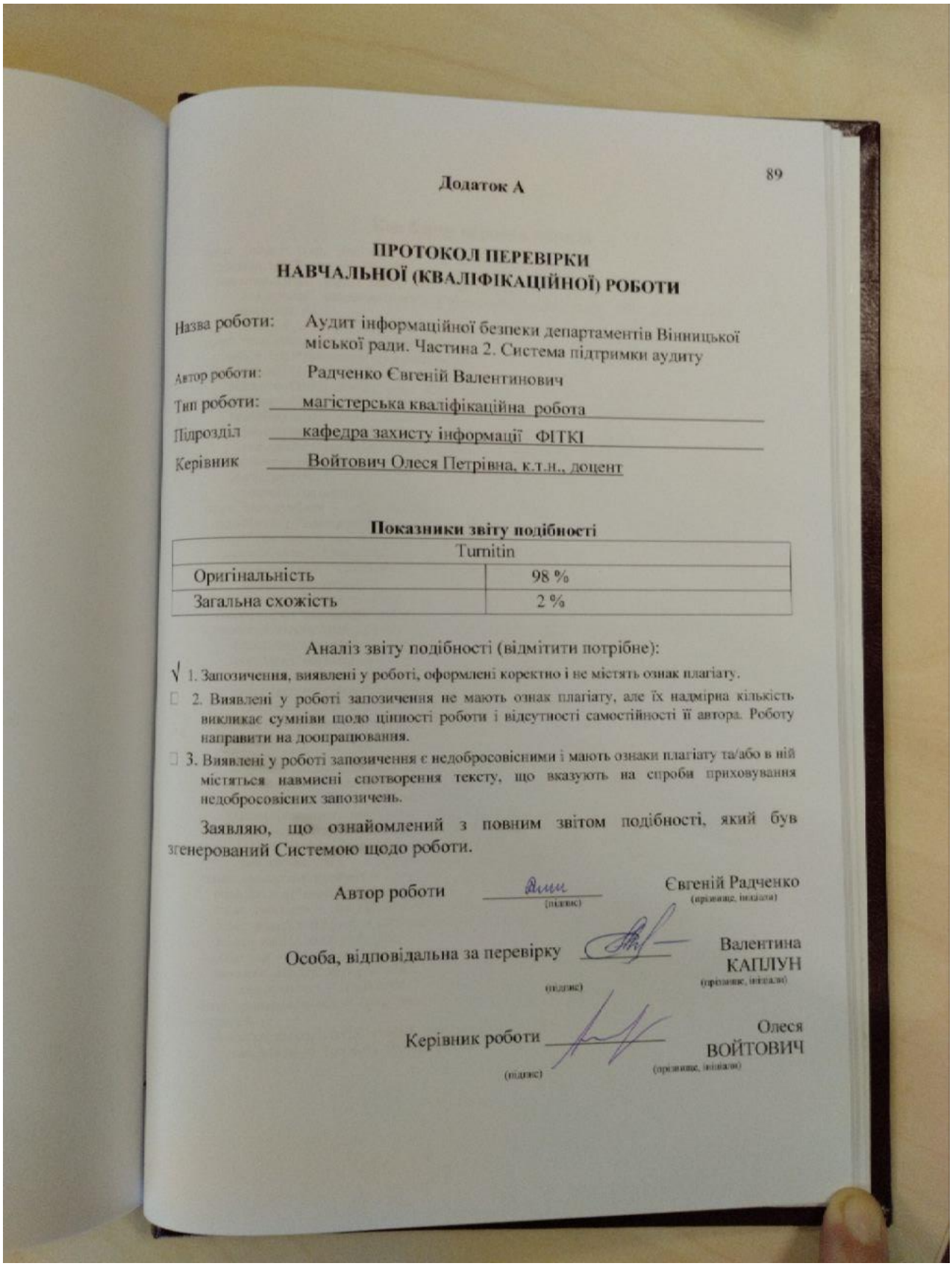
29. GRC. SAP Security Configuration and Deployment. 2009. P. 269–300. URL: https://doi.org/10.1016/b978-1-59749-284-3.00005-3 (дата звернення: 14.12.2024).

30. Mardan A. Putting Frontend and Backend Together. Full Stack JavaScript. Berkeley, CA, 2018. P. 257–287. URL: https://doi.org/10.1007/978-1-4842-3718-2_8 (дата звернення: 14.12.2024).

31. Mann M., Robischon R. HTML Standards-History and Future. The Serials Librarian. 1999. Vol. 36, no. 1-2. P. 51–57. URL: https://doi.org/10.1300/j123v36n01_10 (дата звернення: 14.12.2024).

32. Usage Statistics and Market Share of HTML for Websites, December 2024. W3Techs - extensive and reliable web technology surveys. URL: https://w3techs.com/technologies/details/ml-html_any (дата звернення: 14.12.2024).

ДОДАТКИ



Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Аудит інформаційної безпеки департаментів Вінницької міської ради. Частина 2. Система підтримки аудиту

Автор роботи: Радченко Євгеній Валентинович

Тип роботи: магістерська кваліфікаційна робота

Підрозділ кафедра захисту інформації ФІТКІ

Керівник Войтович Олеся Петрівна, к.т.н., доцент

Показники звіту подібності

Turnitin	
Оригінальність	98 %
Загальна схожість	2 %

Аналіз звіту подібності (відмітити потрібне):

√ 1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

☐ 3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлений з повним звітом подібності, який був згенерований Системою щодо роботи.

Автор роботи _____ (підпис) Євгеній Радченко (прізвище, ініціали)

Особа, відповідальна за перевірку _____ (підпис) Валентина КАПЛУН (прізвище, ініціали)

Керівник роботи _____ (підпис) Олеся ВОЙТОВИЧ (прізвище, ініціали)

Додаток Б

Код блоку обробки запитів

```
      const index = (req, res) => {
        console.log("resource/index")
        Resource.find().sort({ name: -1 }).then(result => {
            res.render('resource/index', { resources: result, title: 'All records' });})
          .catch(err => {
            console.log(err);});}
      const createResource = (req, res) => {
        console.log("resources/createResource")
        Promise.all([
          Resource.find().sort({ name: 1 }), Employee.find().sort({ name: 1 }),
          Software.find().sort({ name: 1 }), Hardware.find().sort({ name: 1 }),
          Transfer.find().sort({ name: 1 })])
          .then(([resources, employees, softwares, hardwares, transfer]) => {
            res.render('resource/createResource', { resources, employees,
softwares,hardwares,transfer,  title: 'Create resource' });})
          .catch(err => {
            console.log(err);});}
      const resourceSave = (req, res) => {
        console.log("resourceSave")
        const resource = new Resource(req.body);
        resource.save().then(result => {
            res.redirect('/resources');}).catch(err => {
            console.log(err);})}
      const resourceDetails = (req, res) => {
        console.log("resourceDetails")
        Resource.findById(req.params.id).populate('software').populate('hardware')
          .populate('transfers').populate('owner').populate('responsible')
          .populate('workingUsers').sort({ name: 1 }).then(resource => {
            res.render('resource/resourceDetails', {resource, title: 'Resource details'});})
          .catch(err => {console.log(err);
            res.render('404', { title: 'Record not found' });})}
      const resourceUpdate = (req, res) => {
        console.log("resourceUpdate")
        Promise.all([
          Resource.findById(req.params.id).sort({ name: 1 }),Employee.find().sort({ name: 1 }),
          Software.find().sort({ name: 1 }),Hardware.find().sort({ name: 1 }),
          Transfer.find().sort({ name: 1 }),])
          .then(([resource, employees, softwares, hardwares, transfers]) => {
            res.render('resource/updateResource', { resource, employees, softwares, hardwares,
transfers, title: 'Update Software' });}).catch(err => {console.log(err);
            res.render('404', { title: 'Record not found' });});}
      const resourceUpdateSend = (req, res) => {
        console.log("resourceUpdateSend")
        const resource = new Resource(req.body);
        const sanitizedObject = JSON.parse(JSON.stringify(resource));
        delete sanitizedObject._id;
        const id = req.params.id;
        Resource.findByIdAndUpdate(id, sanitizedObject, { new: true }).then(result => {
            res.redirect("/resources/" + req.params.id)}).catch(err => {console.log(err);});}
      const resourceDelete = (req, res) => {console.log("resourceDelete")const id =
req.params.id;
        Resource.findByIdAndDelete(id).then(result => {res.json({ redirect: '/resources' });})
          .catch(err => {console.log(err);});}
```

Додаток В

Код блоку зв’язку з базою даних

```
const mongoose = require('mongoose');
const Schema = mongoose.Schema;
const resourceSchema = new Schema({
    name: {
        type: String,
        required: true,
    },    description: String,
    software: [{ type: mongoose.Schema.Types.ObjectId, ref: 'software' }],
    hardware: [{ type: mongoose.Schema.Types.ObjectId, ref: 'hardware' }],
    transfers: [{ type: mongoose.Schema.Types.ObjectId, ref: 'transfer' }],
    confidentiality: {
        type: Number,
    },    integrity: {
        type: Number,
    },    accesability: {
        type: Number,
    },    owner:{
        type: mongoose.Schema.Types.ObjectId,
        ref: 'employee'
    },    responsible:{
        type: mongoose.Schema.Types.ObjectId,
        ref: 'employee'
    },    workingUsers: [{ type: mongoose.Schema.Types.ObjectId, ref: 'employee' }]});
const Resource = mongoose.model('resource', resourceSchema);
module.exports = Resource;
```

Додаток Г

Код ядра

```
const express = require('express');
const morgan = require('morgan');
const mongoose = require('mongoose');
const resourceRoutes = require('./routes/resourceRoutes');
// express app
const app = express();
// connect to mongodb & listen for requests
const dbURI = "mongodb+srv://Blogs_adm:UI41b8nu3x6ZAcGH@pet-01.1kakx.mongodb.net/Audit";
mongoose.connect(dbURI, { useNewUrlParser: true, useUnifiedTopology: true}).then(result =>
{
    console.log("we good")
    app.listen(3000)})
  .catch(err => {
    console.log("We f-d up")
    console.log(err)});
// register view engine
app.set('view engine', 'ejs');
// middleware & static files
app.use(express.static('public'));
app.use(express.urlencoded({ extended: true }));
//app.use(morgan('dev'));
app.use((req, res, next) => {
  res.locals.path = req.path;
  next();});
// blog routes
app.use('/', resourceRoutes);
// 404 page
app.use((req, res) => {res.status(404).render('404', { title: '404' });});
```

Додаток Д

Показники оцінювання науково-технологічного рівня та комерційного потенціалу

Бали (за п’ятибальною шкалою)					
	0	1	2	3	4
1	2	3	4	5	6
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат–ність продукту в реальних умовах
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо нижча за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижча за ціни аналогів	Ціна продукту значно нижча за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші ніж в аналогів	Технічні та споживчі властивості і продукти трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатацій-них витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти	Необхідне незначне навчання фахівців та збіьшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї

		та час на навчання наявних фахівців			
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більший 5-ти років	Термін реалізації ідеї менший 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менший 3-х років. Термін окупності інвестицій менший 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що потребує значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту потребує незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та Реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

ІЛЮСТРАТИВНА ЧАСТИНА

АУДИТ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ ДЕПАРТАМЕНТІВ ВІННИЦЬКОЇ МІСЬКОЇ РАДИ.
ЧАСТИНА 2. СИСТЕМА ПІДТРИМКИ АУДИТУ

Виконав: студент групи 1БС-23м
спеціальності 125 Кібербезпека
та захист інформації
_____ Євгеній РАДЧЕНКО
_____ 2024 р.
Керівник: к. т. н., доцент каф. ЗІ
_____ Олеся ВОЙТОВИЧ
_____ 2024 р.

СХЕМА МОДЕЛІ СИСТЕМАТИЗАЦІЇ ДАНИХ ПРО ІНФОРМАЦІЙНІ РЕСУРСИ

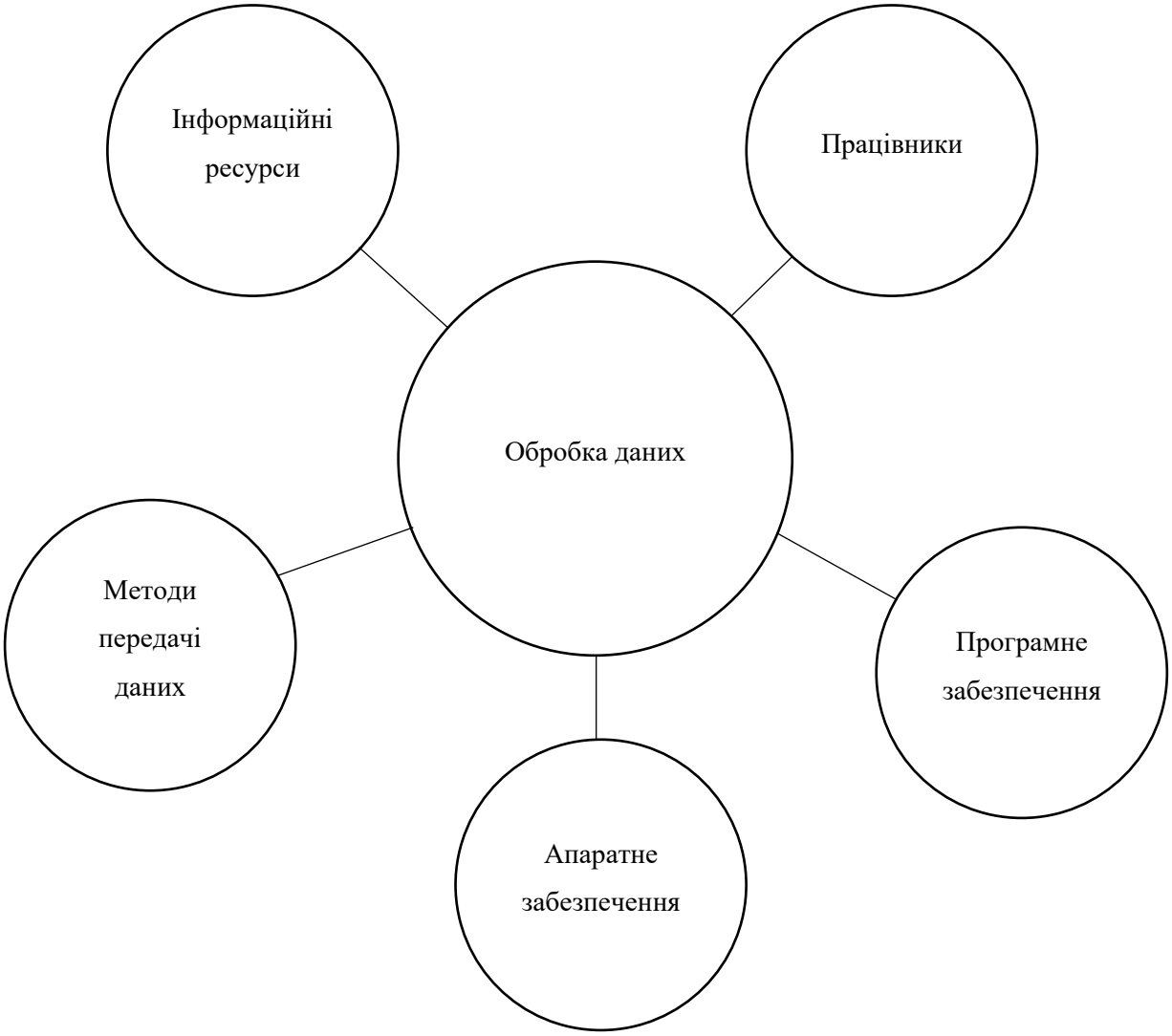
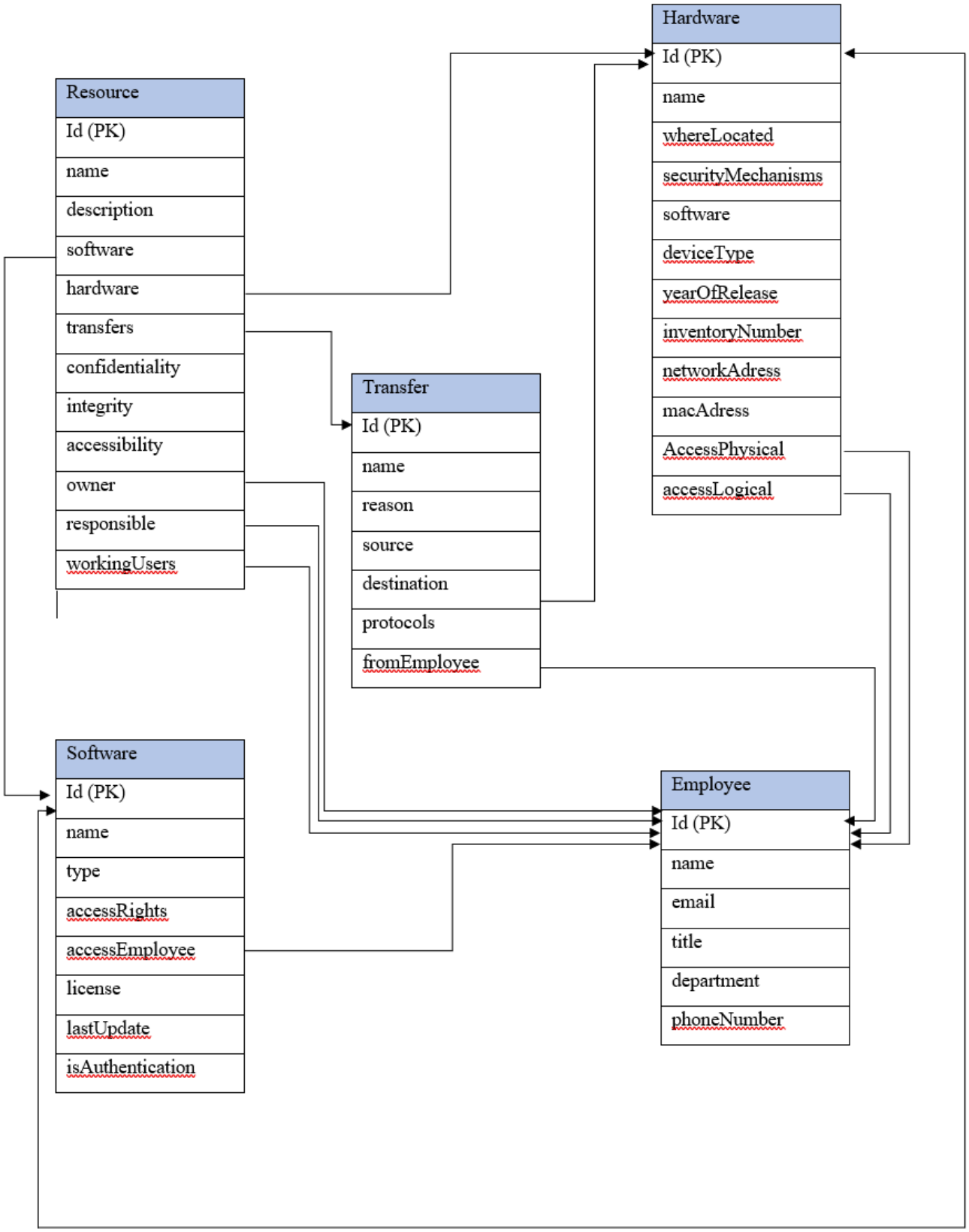
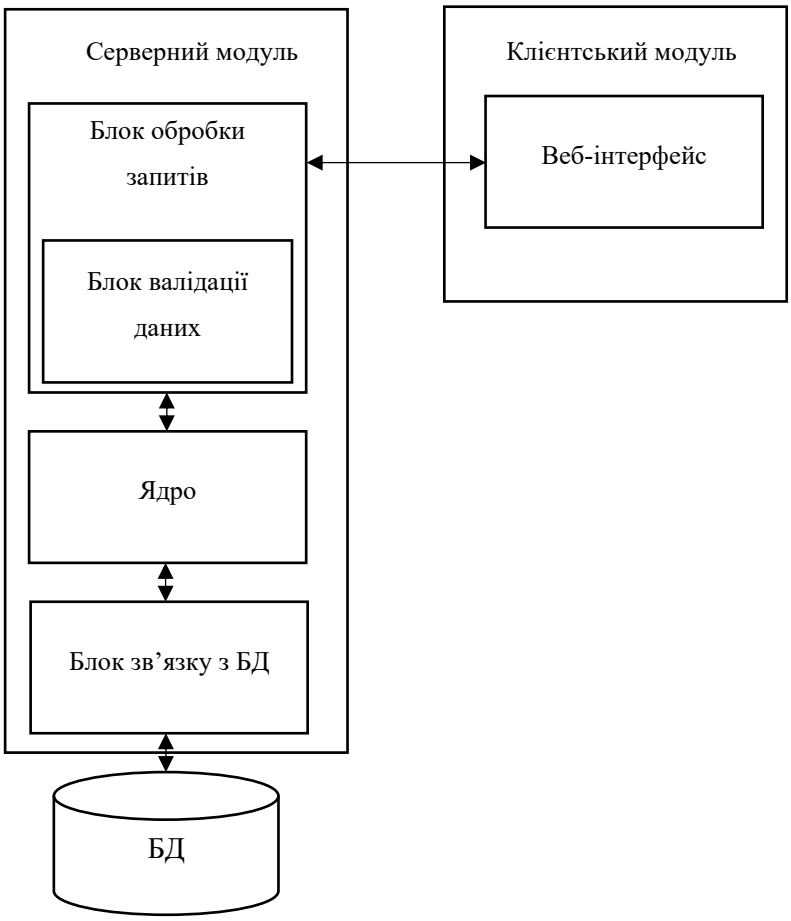


СХЕМА БАЗИ ДАНИХ



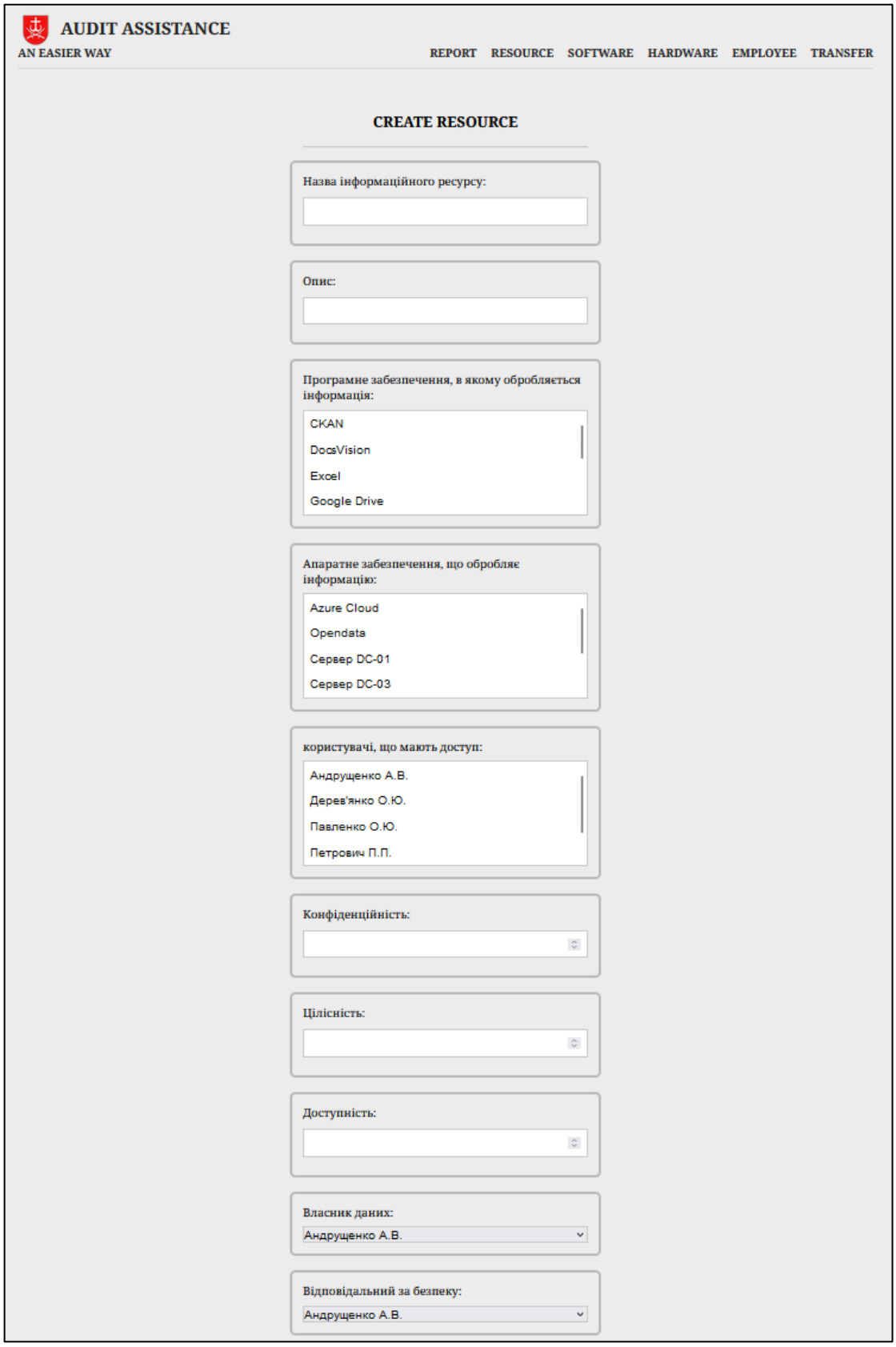
АРХІТЕКТУРА ПРОГРАМНОГО ЗАСОБУ


Серверний модуль
Блок обробки запитів
Блок валідації даних
Ядро
Блок зв’язку з БД
БД
Клієнтський модуль
Веб-інтерфейс

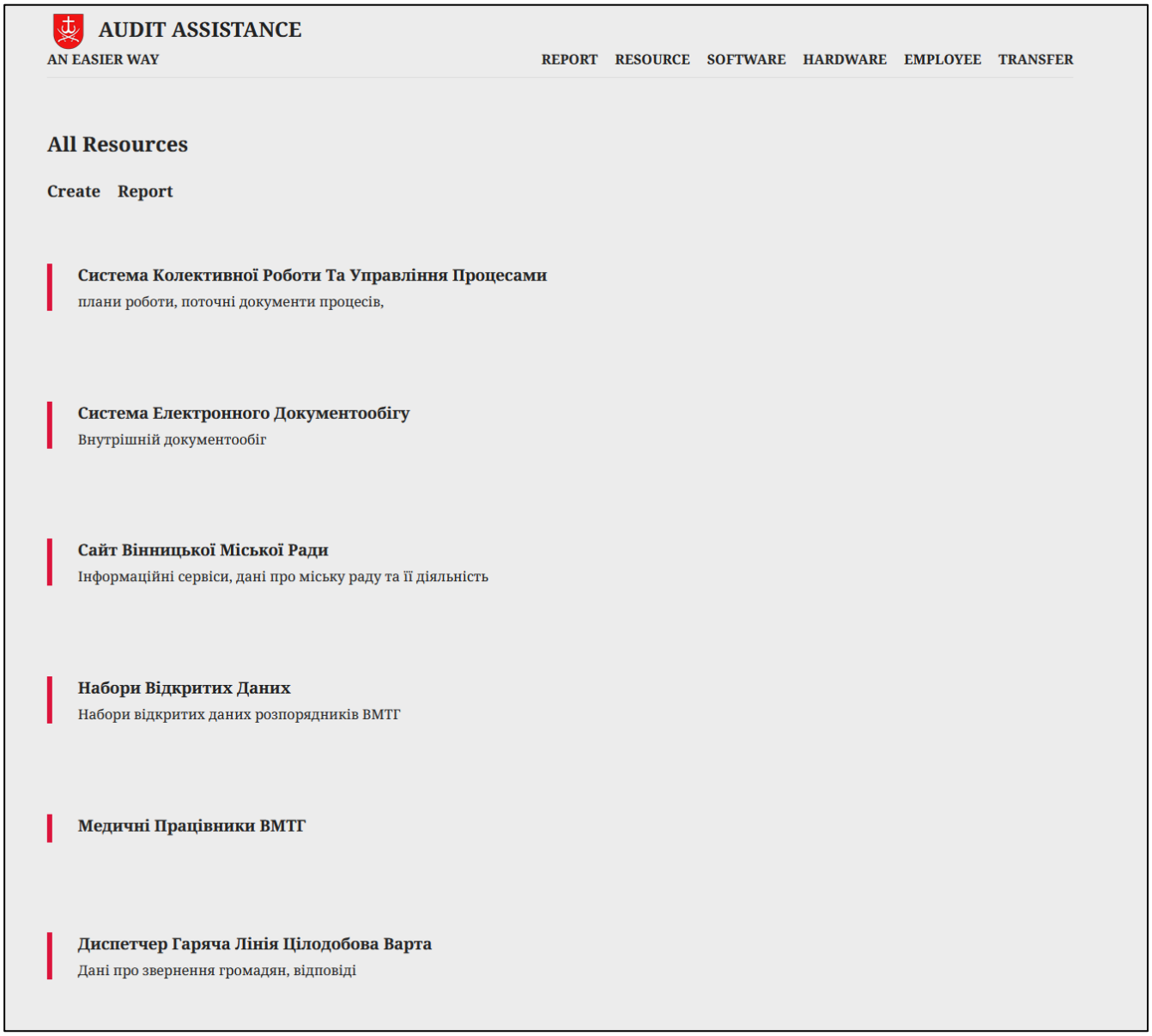
АЛГОРИТМ ОТРИМАННЯ І ОБРОБКИ ДАНИХ

АЛГОРИТМ ФОРМУВАННЯ ЗВІТУ

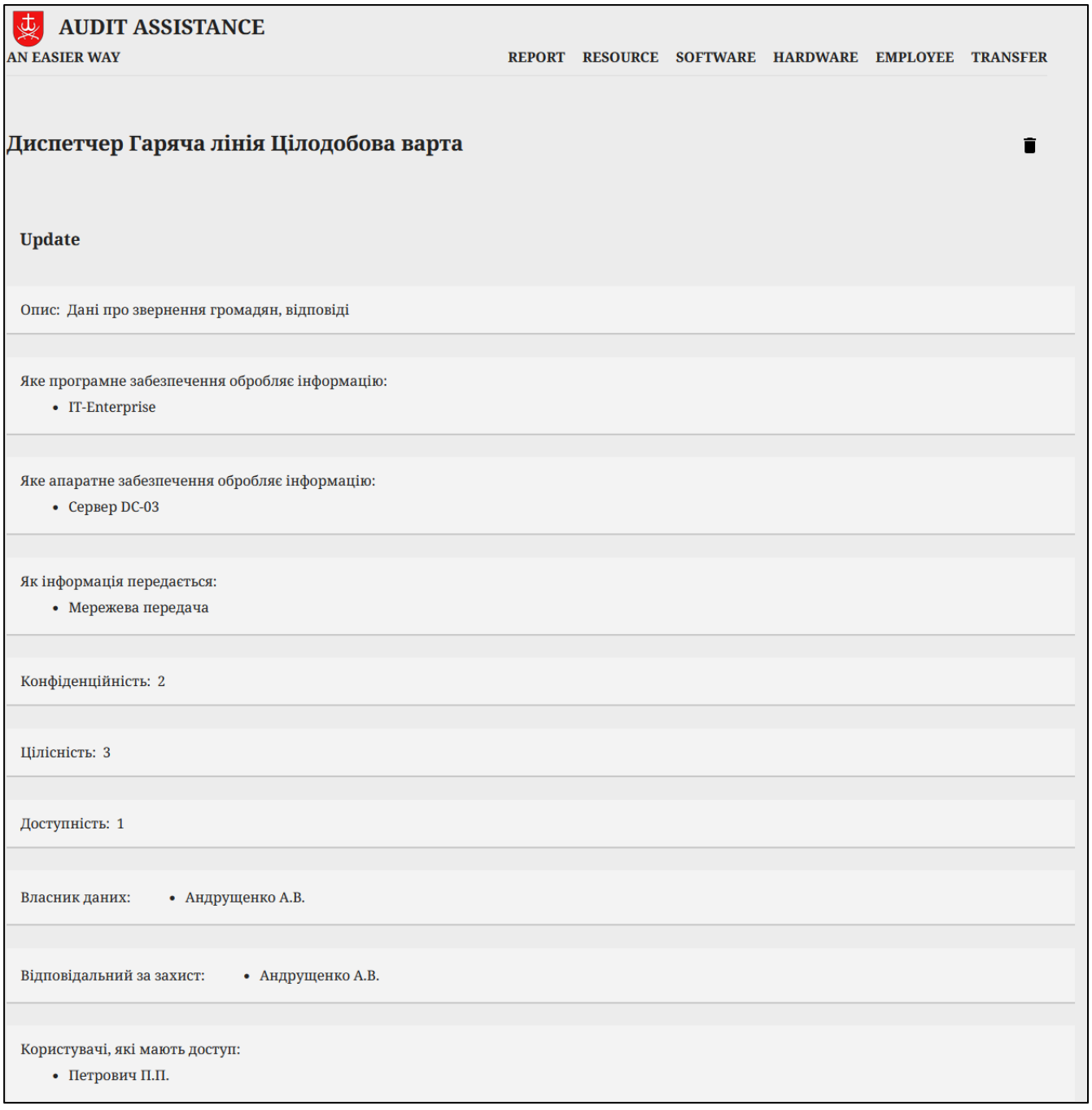
ВНЕСЕННЯ ДАНИХ ДО ПІДСИСТЕМИ ПІДТРИМКИ АУДИТУ



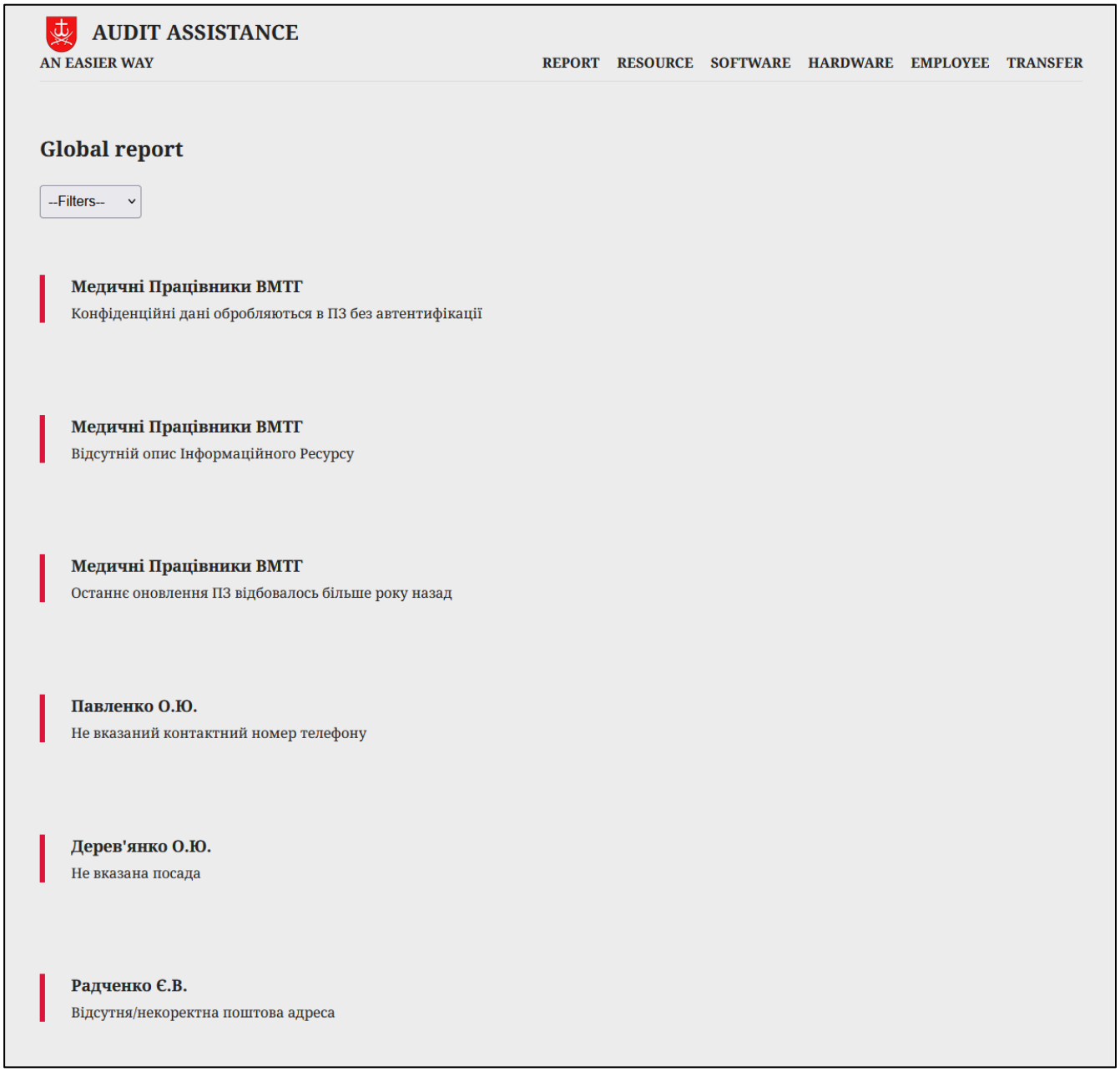
ПЕРЕЛІК ЗАПИСІВ КАТЕГОРІЇ «ІНФОРМАЦІЙНІ РЕСУРСИ»



ДЕТАЛЬНА ІНФОРМАЦІЯ ПРО ЗАПИС «ДИСПЕТЧЕР ГАРЯЧА ЛІНІЯ ЦІЛОДОБОВА ВАРТА»



ЗАГАЛЬНИЙ ЗВІТ З НЕВІДПОВІДНОСТЕЙ



ФІЛЬТРУВАННЯ ЗВІТІВ ЗА КРИТЕРІЄМ НЕВІДПОВІДНОСТІ

ДЕПАРТАМЕНТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
ВІННИЦЬКОЇ МІСЬКОЇ РАДИ

АКТ
ВПРОВАДЖЕННЯ

Цим актом підтверджується, що результати магістерської кваліфікаційної роботи на тему «Аудит інформаційної безпеки департаментів Вінницької міської ради. Частина 2. Система підтримки аудиту», яку виконано Радченком Євгенієм Валентиновичем за спеціальністю 125 «Кібербезпека та захист інформації» (Освітня програма «Безпека інформаційних і комунікаційних систем»), яка виконувалася з 2 вересня 2024 р. по 9 листопада 2024 р., впроваджені у Департаменті інформаційних технологій Вінницької міської ради.

Цей документ не є підставою для фінансових розрахунків.

Директор департаменту
інформаційних технологій
Вінницької міської ради — Володимир РОМАНЕНКО