КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ
УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра технологій управління

Спеціальність 122 «Комп'ютерні науки»
Освітньо-наукова програма «Управління проєктами»

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему:

«Дослідження процесів управління проєктом розробки вебдодатку для управління IT-ресурсами компанії»

Студента 2-го курсу групи УП-21

Всеволода ГЕЛЛИ

(ім'я, прізвище)

(підпис студента)

Науковий керівник:

к.т.н., доцент

(науковий ступінь, вчене звання)

Любов КУБЯВКА

(ім'я, прізвище)

(дата) *(підпис)*

Попередній захист:

(Висновок: "До захисту в Екзаменаційній комісії")

Завідувач кафедри
технологій управління_____ Віктор МОРОЗОВ _____
(підпис) (ім'я, прізвище) (дата)

Київ - 2026

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій

Кафедра технологій управління
Освітній рівень Магістр
Спеціальність 122 Комп'ютерні науки
Освітньо-наукова програма Управління проєктами

ЗАТВЕРДЖУЮ

Завідувач кафедри

професор Віктор МОРОЗОВ

_____

«8» грудня 2025 року

З А В Д А Н Н Я

НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Студент: Гелла Всеволод Вячеславович

Група: УП-21

1. Тема кваліфікаційної роботи: «Дослідження процесів управління проєктом розробки вебдодатку для управління IT-ресурсами компанії».

Затверджена протоколом кафедри ТУ від 5 грудня 2026 року № 4

2. Строк подання студентом готової роботи - «21» травня 2026 р.

3. Цільова установка та вихідні дані до роботи: дослідження різних методів та інструментів для управління проєктом, їх використання у плануванні проєкту, ресурсів, бюджету та управлінні ризиками; дослідження математичних моделей.

4. Зміст роботи: дослідження конкурентного середовища, обґрунтування доцільності та життєздатності проєкту, проведення інвестиційного аналізу проєкту, загальний опис проєкту, побудова дерева проблем і дерева рішень, проведення SWOT-аналізу, ієрархічна структура проєкту, організаційна структура проєкту, аналіз зацікавлених сторін проєкту, вибір методології управління проєктом, календарне планування проєкту, управління ресурсами та бюджетування проєкту, управління ризиками проєкту, розрахунок бюджету проєкту, розробка архітектури системи, опис діаграм потоку даних, створення логічної моделі бази даних, проєктування інтерфейсу користувача.

5. Перелік графічного матеріалу (слайдів): дерево проблем, дерево цілей, WBS та OBS структури проєкту, дорожня карта робіт проєкту, розклад робіт по спринтах, високорівнева архітектура проєкту, діаграми потоку даних, логічна модель бази даних, архітектура клієнтської частини, сторінки вебдодатку.

6. Календарний план виконання роботи:

№ з/п	Назва частин роботи	План виконання роботи
1.	Вивчення літературних джерел з предмету дослідження	12.01.2026-30.01.2026
2.	Складання плану кваліфікаційної роботи магістра	19.01.2026-23.01.2026
3.	Ознайомлення наукового керівника з розгорнутим планом кваліфікаційної роботи. Внесення змін	02.02.2026
4.	Підготовка розділу 1	02.02.2026-27.02.2026
5.	Підготовка розділу 2	02.03.2026-20.03.2026
6.	Підготовка розділу 3	23.03.2026-10.04.2026
7.	Підготовка розділу 4	13.04.26-23.03.2026
8.	Оформлення кваліфікаційної роботи	27.04.2026-01.05.2026
9.	Передача кваліфікаційної роботи науковому керівнику	05.05.2026
10.	Презентація кваліфікаційної роботи рецензенту для рецензування	12.05.2026
11.	Захист магістерської кваліфікаційної роботи	25.05.2026

Дата видачі завдання «10» *грудня* 2025р.

Керівник роботи доцент Любов КУБЯВКА
(посада, ім'я, прізвище)

(підпис)

Завдання прийняв до виконання студент групи УП-21

Всеволод ГЕЛЛА
(ім'я, прізвище)

(підпис)

АНОТАЦІЯ

кваліфікаційної роботи магістра на тему

«Дослідження процесів управління проєктом розробки вебдодатку для управління IT-ресурсами компанії»

Студент: Гелла Всеволод Вячеславович
Науковий керівник: Кубявка Любов Богданівна
Рік захисту: 2026

Метою кваліфікаційної роботи є дослідження процесів управління проєктом та розробка архітектури корпоративного вебдодатку для централізованого управління IT-ресурсами компанії, що об'єднує модулі ITAM, CMDB та ITSM.

Ціль проєкту – розробка ефективного інструменту для централізованого обліку активів, моніторингу корпоративної IT-інфраструктури та оптимізації роботи служби технічної підтримки.

Наукова новизна дослідження полягає в застосуванні гібридного підходу збереження даних у системі управління IT-активами, що поєднує транзакційну реляційну базу для обліку атрибутів із графовою базою для миттєвого рекурсивного розрахунку топології мережі та залежностей між конфігураційними одиницями.

Кваліфікаційної робота складається з анотації, вступу, основної частини, яка включає чотири розділи, висновків, переліку використаних джерел та додатків.

У першому розділі досліджено методологічні моделі та сучасні концепції управління IT-ресурсами компанії. Проведено аналіз конкурентного середовища та обґрунтовано доцільність і життєздатність проєкту. Побудовано дерево проблем і дерево рішень, здійснено SWOT-аналіз та аналіз зацікавлених сторін. Сформовано технічне завдання на розробку у вигляді

паспорта проєкту, а також визначено ключові функціональні та нефункціональні вимоги до вебдодатка.

У другому розділі розроблено загальну концепцію проєкту. Побудовано концептуальну та математичну моделі інформаційної системи, які формалізують сутності платформи та логіку інформаційних потоків між ними.

У третьому розділі здійснено планування та організацію проєкту. Обґрунтовано вибір технології управління проєктом, розроблено ієрархічну структуру робіт (WBS) та організаційну структуру команди з детальним розподілом відповідальності (матриця RACI). Проведено планування ресурсів та бюджету, складено детальний календарний план. Також розроблено стратегію управління ризиками проєкту з визначенням заходів щодо їхньої мінімізації.

У четвертому розділі описано практичну реалізацію проєкту. Розроблено інформаційну структуру вебдодатка. Спроєктовано концептуальну, логічну та детальну фізичну моделі баз даних. Проаналізовано сучасні тренди та вимоги до UI/UX дизайну корпоративних систем, зокрема використання концепції AppShell, на основі чого спроєктовано користувацький інтерфейс та розроблено детальні візуальні прототипи ключових екранів платформи.

Висновок містить підсумки та аналіз проведених досліджень.

Робота містить 96 сторінок без додатків, 30 рисунків та 15 таблиць. Додатки складають 5 сторінок

***Ключові слова**: управління проєктами, управління IT-ресурсами ITAM, CMDB, ITSM, гібридний підхід збереження даних, корпоративний UI/UX дизайн, AppShell,*

ЗМІСТ

ВСТУП

На даний момент важко уявити існування сучасної компанії, яка б не проходила процес цифрової трансформації, в основі якої лежить стрімке збільшення кількості технічного обладнання. Успішна діяльність підприємств безпосередньо залежить від ефективності управління їхніми IT-ресурсами. Враховуючи постійні оновлення та збільшення наявності комп'ютерної техніки, мережевого та серверного обладнання, а також периферійних пристроїв виникає гостра потреба у прозорому та чіткому моніторингу їх життєвого циклу, розподілу та технічного обслуговування. Відсутність належного відстеження IT-активів від моменту їх придбання до виведення з експлуатації призводить до неефективного обслуговування, передчасного зносу та втрати можливостей для економії коштів, тобто з’являються ситуації, коли є невикористані або неправильно використані активи

Вирішенням цієї проблеми є розробка спеціалізованого вебдодатка, здатного консолідувати дані про всі IT-ресурси компанії, що враховуючи вищесказане є вкрай актуальним.

Метою дослідження є теоретичне обґрунтування та розробка стратегії управління проєктом створення корпоративного вебдодатка для консолідованого зберігання, інвентаризації та моніторингу IT-ресурсів компанії. Реалізація цієї мети спрямована на забезпечення ефективної координації процесів розробки, мінімізацію проєктних ризиків, дотримання бюджетних і часових обмежень.

Для досягнення мети необхідно виконати наступні завдання:

- проаналізувати сучасний стан та специфіку процесів управління IT-ресурсами підприємства, а також виявити критичні недоліки в існуючих методах обліку, моніторингу та розподілу технологічного обладнання;
- дослідити методологічні підходи до управління проєктами розробки вебдодатків, обґрунтувавши вибір оптимальної моделі;

- сформувати технічні та функціональні вимоги до системи управління IT-ресурсами, враховуючи необхідність інтеграції різнорідних параметрів: від технічних характеристик апаратного забезпечення до історії його експлуатації;
- розробити концептуальну та архітектурну моделі бази даних вебдодатка;
- побудувати організаційну модель та ієрархічну структуру робіт (WBS) із чітким розподілом повноважень між членами команди, а також підготувати розклад виконання проєкту й обґрунтувати його кошторис на основі потреби в ресурсах;
- розробити UI/UX-дизайн вебдодатка, зокрема створити прототипи інформаційних панелей та дашбордів для інтерактивного моніторингу технічного стану IT-інфраструктури й відображення даних автоматизованого обліку в режимі реального часу.

Об’єктом дослідження є методи, моделі та інструменти управління проєктом щодо розробки вебдодатка для управління IT-ресурсами компанії

Предметом дослідження є процеси управління проєктами розробки вебдодатків з управління IT-ресурсами компаній.

Під час виконання роботи використані такі методи дослідження:

- аналіз наукових джерел та корпоративних практик управління проєктами для вибору оптимальної методології розробки;
- системний аналіз та структурне моделювання для формування вимог та проєктування архітектури бази даних;
- методи декомпозиції робіт (WBS), календарного та бюджетного планування для організації процесів розробки;
- прототипування для проєктування користувацьких інтерфейсів та дашбордів системи.

Наукова новизна дослідження полягає у розробці та обґрунтуванні комплексної моделі управління проєктом створення корпоративного

вебдодатка, орієнтованого саме на консолідоване управління всією сукупністю IT-ресурсів компанії.

Практичне значення: результати дослідження можуть бути використані при розробці та впроваджені подібних вебдодатків, які підвищують ефективність управління IT-ресурсами компанії. Сформовані у роботі концептуальні моделі, технічні вимоги, ієрархічна структура робіт (WBS), організаційна структура команди, календарний план, розрахунки бюджету та план реагування на ризики є готовим пакетом проєктної документації.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА ОБРҐРУНТУВАННЯ ДОЦІЛЬНОСТІ ТА ЖИТТЄЗДАТНОСТІ ПРОЄКТУ

1.1 Методологічні моделі та сучасні концепції управління IT-ресурсами компанії

Для якісного аналізу конкурентного середовища для початку потрібно зрозуміти та проаналізувати теоретичні основи різних концепцій управління IT-ресурсами компанії та визначити їх спільні та відмінні риси. Згідно з міжнародними стандартами на поточний момент розрізняють три основні концепції: «Управління IT-активами» або ж ITAM, «База даних управління конфігураціями» або ж CMDB, «Управління IT-сервісами» або ж ITSM. Розглянемо більш детально кожну з цих концепцій та сформуємо їх порівняльну характеристику.

Управління IT-активами (ITAM) – це концепція, яка зосереджується на фінансовому, договірному та логістичному життєвому циклі IT-ресурсів організації [1]. Згідно з міжнародним стандартом ISO/IEC 19770-1:2017, управління активами визначається як скоординована діяльність компанії, спрямована на реалізацію та максимізацію цінності від будь-якого елемента або ресурсу, що використовується для придбання, обробки, зберігання та розповсюдження цифрової інформації [2]. Тобто ITAM відстежує актив як самостійну одиницю незалежно від того, чи він наразі включений до корпоративної мережі та функціонує, чи знаходиться в резерві на складі, чи перебуває в процесі ремонту. Ця концепція поділяється на два пов’язані між собою напрямки: управління апаратними активами та управління активами програмного забезпечення [3]. Обидва ці напрямки вимагають оптимізації процесів придбання, використання, обслуговування та ліквідації IT-ресурсів. Впровадження цієї концепції гарантує супровід кожного IT-активу компанії через критичні стадії, які зменшують ризик непередбачуваних витрат, штрафів під час ліцензійних аудитів вендорів використовуваного компанією програмного забезпечення, ймовірної крадіжки або втрати даних [1]. Перша

стадія відбувається на етапі закупівлі, де фіксується початкова вартість, постачальник та гарантійні умови. Друга стадія реалізується на етапі розгортання та експлуатації, де відстежується поточне місцезнаходження, відповідальна особа та витрати на амортизацію обладнання [1]. Третя стадія – це етап виведення з експлуатації, де гарантується, що обладнання списується з балансу підприємства коректно, а всі конфіденційні дані на носіях інформації безпечно знищуються.

Якщо ITAM дивиться на активи через призму фінансів та життєвого циклу, то база даних управління конфігураціями (CMDB) розглядає IT-інфраструктуру виключно з операційної та технічної точок зору. Стандарт ISO/IEC 20000-1:2018 визначає поняття конфігураційної одиниці або ж CI як будь-якого компонента, який необхідно суворо контролювати для забезпечення безперебійного надання IT-сервісів [1]. Головним фокусом CMDB є не просто фіксація наявності обладнання, а побудова глибокої карти взаємозв'язків та залежностей між конфігураційними одиницями [2]. Наприклад, CMDB фіксує, як конкретний веб-сервер залежить від певного кластера баз даних, який фізично розміщений на конкретному обладнанні, підключеному до певного порту мережевого комутатора. Ключовим завданням CMDB у рамках операційної діяльності є забезпечення IT-спеціалістів життєво необхідним контекстом під час планування змін, дозволяючи заздалегідь проаналізувати, які бізнес-процеси зупиняться у разі перезавантаження або оновлення певного маршрутизатора [2].

Управління IT-сервісами (ITSM) є глобальною інфраструктурою, яка керує проектуванням, створенням, підтримкою та безперервним вдосконаленням інформаційних технологій [4]. На відміну від ITAM та CMDB, які є скоріше технічними сховищами даних та процесними підсистемами, ITSM виступає в ролі організаційної парадигми, що пов'язує технології з бізнес-цілями підприємства. Найбільш розповсюдженим і авторитетним світовим фреймворком для реалізації ITSM є бібліотека інфраструктури інформаційних технологій або ж ITIL (Information Technology

Infrastructure Library). Найновішим стандартом цього фреймворку є ITIL 4, яка визначає 34 чітко регламентовані управлінські практики, які поділені на три групи: 14 загально-управлінських практик, 17 практик управління сервісами та 3 практики технічного управління [1]. Основною ж концепцією ITIL 4 є система створення цінності сервісів, яка описує як різні компоненти та діяльність компанії пов’язані між собою для створення цінності шляхом надання IT-послуг [5]. Центральним елементом цієї системи є ланцюжок створення цінності та модель безперервного вдосконалення або ж CSI. ITSM визначає IT-департамент як внутрішнього постачальника послуг, який взаємодіє з бізнес-підрозділами на основі чітких угод про рівень надання послуг (SLA) [1].

У якості результату дослідження концепцій управління IT-ресурсами компаній створено їх порівняльну характеристику (таблиця 1.1) у контексті їх впливу на корпоративну архітектуру компанії.

Таблиця 1.1.

	CMDB (База даних управління конфігураціями)	**ITAM (Управління IT-активами)**	**ITSM (Управління IT-сервісами)**
1	**2**	**3**	**4**
Основний фокус системи	Технічні взаємозв'язки та взаємозалежності між активними конфігураційними одиницями	Життєвий цикл, статус та фінансовий стан індивідуального об'єкта	Управління та оптимізація якості IT-послуг для задоволення стратегічних цілей бізнесу
Ціннісна пропозиція для бізнесу	Забезпечення стабільності інфраструктури, аналіз впливу змін, швидка локалізація збоїв мережі	Контроль бюджету, розрахунок амортизації, дотримання ліцензійної чистоти	Підвищення задоволеності користувачів, систематизація запитів, виконання SLA

Продовження табл. 1.1.

1	2	3	4
Необхідний інструментарій	База даних із функціями візуалізації топології	Реєстри обліку, фінансові модулі, інструменти сканування штрих-кодів, база контрактів	Портал обслуговування, система, модулі управління інцидентами та змінами
Регулюючі міжнародні стандарти	Базується на вимогах ISO/IEC 20000-1:2018	Базується на вимогах ISO/IEC 19770-1:2017	Рамкові методології ITIL 4

Багато компаній намагалися впроваджувати ці концепції ізольовано, купуючи окрему програму для інвентаризації, окрему систему для обробки запитів та ведучи облік конфігурацій вручну. Такий розрізнений підхід визнаний сучасними дослідниками вкрай неефективним і таким, що веде до деградації інформаційного середовища [1]. Зберігання фінансової та гарантійної інформації ITAM безпосередньо у класичній базі даних управління конфігураціями CMDB є типовою помилкою. Експерти галузі та аналітичні звіти Gartner відзначають, що до 80% ініціатив із впровадження CMDB зазнають краху саме через так званий підхід «кухонної раковини», тобто коли системні адміністратори намагаються перетворити CMDB на універсальне сховище для всіх можливих атрибутів обладнання, включаючи бухгалтерські коди, номери договорів та дати закінчення ліцензій [1]. Наслідком цього є те, що база даних швидко стає громіздкою, складною для запитів та надзвичайно дорогою в обслуговуванні.

З іншого боку, використання виключно ITAM-інструменту також є недостатнім, оскільки він не здатен демонструвати операційні наслідки. Якщо на сервері виходить з ладу жорсткий диск, ITAM-система покаже, коли цей сервер був придбаний і чи покривається він гарантією виробника, але не зможе

повідомити, що цей сервер входив до кластера, який підтримує головний платіжний шлюз компанії [2].

1.2 Аналіз конкурентного середовища та обґрунтування доцільності та життєздатності проєкту

Для всебічного аналізу конкурентного середовища варто спочатку звернути увагу на ключових гравців на ринку. Верхній сегмент ринку повністю контролюється великими корпоративними системами, які визначають галузеві стандарти та забезпечують широкий функціонал. Найбільш популярними представниками цієї категорії є ServiceNow, ManageEngine ServiceDesk Plus та Ivanti Neurons [6].

Платформа ServiceNow є безумовним лідером для найбільших світових корпорацій. Ця платформа охоплює весь спектр управлінських практик ITIL 4, пропонуючи глибоку автоматизацію, використання штучного інтелекту для маршрутизації завдань, предиктивну аналітику та потужні інструменти управління апаратними та програмними активами [6]. Користувачі ServiceNow високо оцінюють можливості оптимізації ліцензій, автоматизованого вилучення активів та зниження ризику під час впровадження змін [7]. Однак для компаній які відносяться до сегменту середнього і малого бізнесу ServiceNow є непрактичним вибором через низку нездоланних обмежень. До суттєво великої вартості ліцензій та послуг з впровадження, ця платформа побудована на базі мови GlideScript, яка є доволі рідкісною [6]. Це означає, що внесення будь-яких структурних змін до системи вимагає щонайменше наявності кваліфікованого працівника у штаті компанії. Багато підприємств середнього рівня, які інвестували у ServiceNow, згодом страждають від низького рівня використання розширених модулів та високих обсягів звернень до технічної підтримки для вирішення елементарних завдань [8].

ManageEngine позиціонується як потужна платформа ITSM з розширеними функціями інвентаризації активів та модулем CMDB, яка пропонує більш гнучку цінову політику порівняно зі ServiceNow [9].

Перевагою є також те, що цей інструмент має сертифікацію від PinkVERIFY для ключових процесів ITIL та активно впроваджує можливості генеративного штучного інтелекту [10]. Однак, незважаючи на свої переваги ManageEngine стикається зі значною критикою від користувачів середнього сегменту бізнесу. Експерти вказують, що архітектура системи може здаватися «важкою» для компаній, які не мають достатніх адміністративних ресурсів, користувацький інтерфейс часто характеризується як застарілий та менш інтуїтивно зрозумілий порівняно з сучасними хмарними альтернативами [11].

Компанія Ivanti розробила амбітну платформу, яка намагається ліквідувати бар'єри між ITSM, ITAM та кібербезпекою в рамках єдиного середовища [1]. Завдяки глибокому фокусу на автономному виявленні активів та нормалізації даних, Ivanti Neurons забезпечує потужну аналітику життєвого циклу техніки та програмного забезпечення. Однак цей інструмент також має свої недоліки. Зокрема, враховуючи відгуки користувачів, початкове налаштування є вкрай складним, а адміністративний інтерфейс є неінтуїтивним, що створює значну залежність від консультантів на етапі впровадження [10]. Окрім того, це рішення має складнощі з гнучким налаштуванням специфічних персоналізованих звітів. Ivanti Neurons також не пропонує базових безкоштовних рівнів для невеликих організацій, що обмежує її привабливість для компаній, який щойно починає масштабуватися.

З іншого боку великих платформ знаходяться системи управління з відкритим вихідним кодом та спеціалізовані утиліти для обліку. Головними перевагами таких систем для малого та середнього бізнесу є відсутність регулярних ліцензійних платежів, прозорість програмного коду та можливість швидкого самостійного розгортання [12]. Найвідомішими представниками цієї категорії є Snipe-IT, GLPI та Spiceworks.

Система Snipe-IT завоювала популярність завдяки своєму прагматичному підходу. Вона розроблена спеціально для IT-менеджерів, які потребують надійної інвентаризації без перевантаження складними процесами ITIL [12]. Згідно з даними порталів відгуків, користувачі високо оцінюють

легкість налаштування та гнучкість системи, оскільки Snipe-IT дозволяє створювати необмежену кількість користувацьких полів, генерувати штрих-коди та керувати логістикою видачі техніки працівникам. Однак архітектурні обмеження Snipe-IT роблять це рішення недостатнім для компаній, що зростають, адже у загальному це виключно статичний реєстр активів, який не має модуля CMDB [12]. Як наслідок, ця система не здатна відображати взаємозв'язки між програмним забезпеченням та апаратними серверами, більшість даних вноситься до системи вручну або через масовий імпорт електронних таблиць, оскільки інструменти автоматичного мережевого виявлення не є її сильною стороною.

Платформа GLPI представляє собою більш комплексну спробу створити безкоштовну альтернативу великим корпоративним системам [13]. Вона об'єднує функціонал управління IT-активами з базовими механізмами ITSM, а саме службою підтримки, маршрутизацією заявок та обліком проблем. Використовуючи розширення такі як GLPI Agent, система може автоматизовано сканувати мережі та збирати конфігураційні дані обладнання. Незважаючи на свій широкий функціонал, GLPI стикається із серйозними проблемами масштабованості та архітектурної нестабільності [13]. Розширення можливостей платформи критично залежить від екосистеми сторонніх плагінів, які часто конфліктують між собою. Інтерфейс адміністратора вважається перевантаженим і не відповідає сучасним стандартам UX-дизайну.

Spiceworks пропонує безкоштовні інструменти для сканування мережі та базову систему служби підтримки [14]. Її головною цільовою аудиторією є малі компанії з мінімальними IT-бюджетами. Проте функціональні обмеження Spiceworks є значними: інструмент не масштабується для складних гібридних мереж, не підтримує сучасні методології ITIL та позбавлений розвинених механізмів CMDB та контрактного управління [14].

Прогалину між надскладними рішеннями та простими трекерами на ринку намагається закрити категорія систем середнього рівня.

Найпопулярнішими представниками цієї групи є SolarWinds Service Desk та SysAid, а також платформи з фокусом на штучний інтелект, такі як Xurrent, Ravenna [6].

SolarWinds Service Desk та SysAid є потужними проміжними інструментами, проте вони страждають від проблеми ізольованих даних, коли інформація про активи не повною мірою синхронізується [10]. Інтерфейс цих систем часто критикується користувачами як застарілий, а розширені функції автоматизації в таких системах часто блокуються жорсткими тарифними планами, змушуючи клієнтів постійно доплачувати за розширення [10].

Сучасні платформи нового покоління Xurrent та Ravenna, демонструють еволюцію ринку в бік повної автоматизації. Вони відмовляються від традиційних веб-порталів на користь розмовних інтерфейсів та ШІ-маршрутизації [10]. Їхня перевага полягає у миттєвому впровадженні та використанні для всіх підрозділів компанії. Однак фокус цих інструментів значною мірою зосереджений на управлінні комунікаціями, тоді як глибоке інженерне відстеження життєвого циклу серверного обладнання, розрахунок амортизації та побудова детальних зв’язків між обладнанням залишаються другорядними функціями.

Базуючись на проведеному досліджені конкурентного середовища, а також розгляді сучасних концепцій управління IT-ресурсами компанії, можна зробити висновок про незаповнену потребу на ринку якісного рішення для малого та середнього бізнесу. Перевагою та особливістю проєкту розглянутого в рамках магістерської роботи є створення архітектурно збалансованого продукту, який займе цю нішу. Проектована система не буде черговим ізольованим координатором техніки, і водночас не стане громіздким фреймворком для великих корпорацій. Її архітектура органічно об'єднуватиме фінансово-життєвий цикл активів (ITAM), інтерактивну карту технічних залежностей (CMDB) та процесно-орієнтовані робочі процеси управління сервісами (ITSM), адаптовані під операційні потреби та бюджетні обмеження компаній середнього бізнесу.

1.3 Побудова дерева проблем і дерева рішень

В межах проведеного дослідження було виявлено основну проблему проєкту (рис. 1.1), а саме неефективне, децентралізоване та непрозоре управління IT-ресурсами компанії, яка спричинена чотирма основними групами:

- *відсутність регламентованих процесів управління життєвим циклом*: це проявляється через нечіткі процедури введення в експлуатацію та подальшого списання активів, а також через відсутність єдиної, затвердженої політики розподілу техніки між співробітниками;
- *низька комунікація та координація між підрозділами*: воно спричинене постійними розбіжностями в даних про IT-ресурси між різними відділами, зокрема IT та бухгалтерією;
- *обмежений контроль за програмним забезпеченням та ліцензіями*;
- *недостатнє управління технічними та операційними ризиками*: спричинене відсутністю завчасного або ж проактивного попередження про критичну зношеність обладнання.

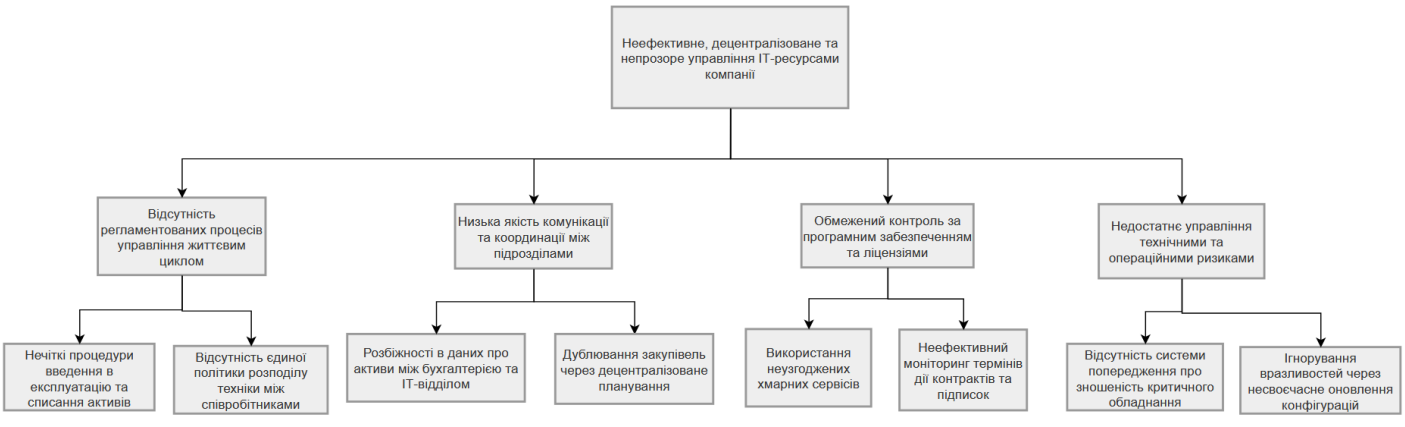

Рисунок 1.1. Дерево причин та наслідків. Частина перша.

Як результат, описані причини призводять до наступних наслідків (рис. 1.2) для компанії:

- *втрата контролю над життєвим циклом активів*: як наслідок надлишкова закупівля обладнання, порушення ліцензійних угод та штрафи, перевитрати бюджету;
- *збільшення тривалості та кількості критичних збоїв*: ручний встановлення зв'язків між обладнанням призводить до простою бізнес-процесів та відповідних збитків;
- *зниження продуктивності IT-відділу*: зайва витрата часу на звірку реєстрів та інвентаризацію;
- *підвищується ризики інформаційної безпеки*: наявність некерованих активів та своєчасних їх оновлень підвищує їх вразливість до кібератак тощо.

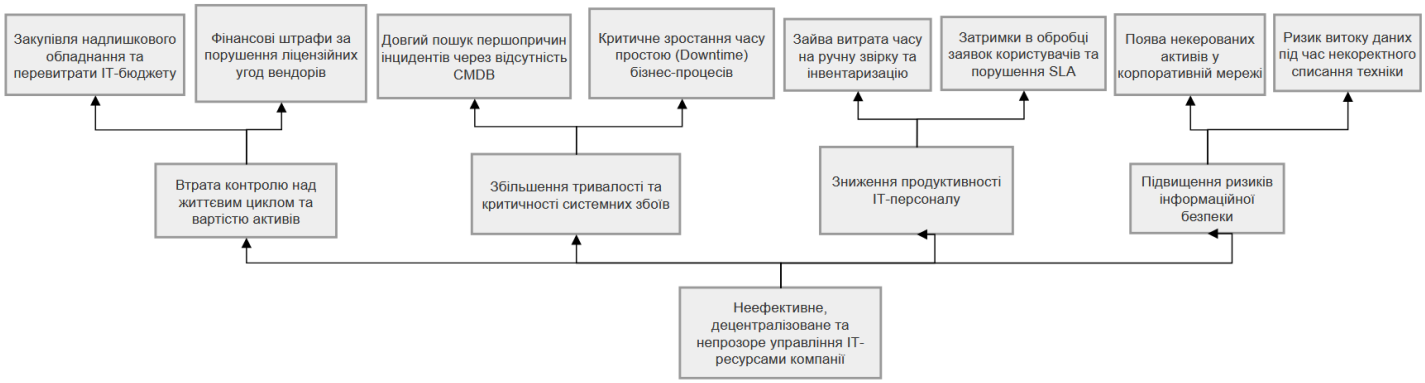

Рисунок 1.2. Дерево причин та наслідків. Частина друга.

Головною метою проєкту (рис. 1.3) є підвищення ефективності управління IT-ресурсами компанії на всіх етапах їх життєвого циклу шляхом створення консолідованого інформаційного середовища.

Розробка та впровадження корпоративного вебдодатка для централізованого обліку й моніторингу IT-активів дозволить спрогнозувати наступні покращення:

- *оптимізація обліку та розподілу активів:* досягається шляхом впровадження централізованого реєстру;
- *забезпечить прозорість IT-інфраструктури:* реалізується через побудову мапи взаємозалежностей її складових;

- *мінімізація технічних та безпекових ризиків:* забезпечиться налаштуванням проактивного моніторингу та впровадженням алгоритмів завчасного виявлення несправності обладнання та автоматичних сповіщень про це;
- *забезпечення діяльності управління проєктом*.

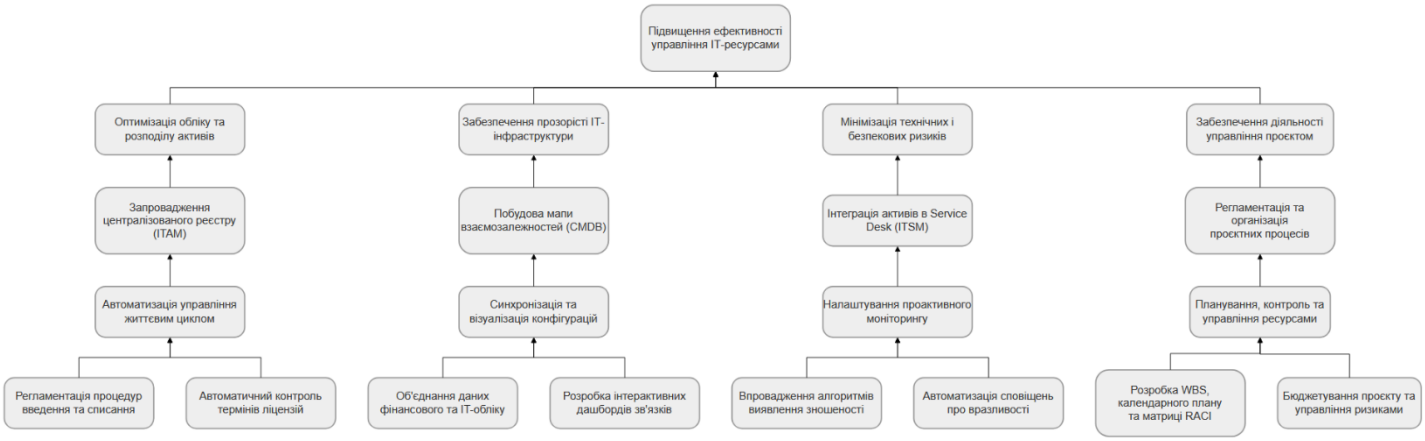

Рисунок 1.3. Дерево цілей.

1.4 Проведення SWOT-аналізу

В управлінні проєктами SWOT-аналіз є критично важливим інструментом на етапі ініціації та стратегічного планування. Його важливість полягає в тому, що він дозволяє об’єктивно оцінити внутрішній потенціал створюваного продукту, у даному випадку вебдодатка, та зіставити його з умовами зовнішнього середовища.

Результати комплексного дослідження внутрішнього та зовнішнього середовищ проєкту розробки системи управління IT-ресурсами наведено у матриці SWOT (таблиця 1.2).

SWOT-аналіз

Таблиця 1.2

Сильні сторони (Strengths)	**Слабкі сторони (Weaknesses)**
1	**2**
Синергія концепцій: органічне поєднання функціоналу ITAM, CMDB та базового ITSM в єдиному консолідованому інтерфейсі	*Обмеженість ресурсів:* брак фінансових та людських ресурсів на початкових етапах розробки
Гнучкість архітектури: можливість швидкої персоналізації під специфічні бізнес-процеси компаній середнього бізнесу	*Відсутність репутації:* продукт є новим на ринку, що може викликати обережність стейкхолдерів порівняно з відомими брендами
Хмарна орієнтованість: зниження витрат на впровадження завдяки відсутності потреби у власному серверному обладнанні	*Обмеженість інтеграцій:* відсутність готових, вбудованих рішень для всіх можливих нішевих сторонніх сервісів
Фокус на UX/UI: сучасний, інтуїтивно зрозумілий інтерфейс та інтерактивні дашборди, що знижують поріг входу для користувачів.	
Можливості (Opportunities)	**Загрози (Threats)**
1	**2**
Зростання попиту: тренд на цифровізацію та необхідність суворого контролю витрат на IT-ресурси в умовах віддаленої роботи.	*Вплив війни в Україні:* ризики блекаутів, пошкодження інфраструктури (втрата зв'язку), а також мобілізація ключових розробників

Продовження табл. 1.2.

1	2
Незакритий ринок: наявність компаній середнього бізнесу, які шукають доступніші альтернативи дорогим Enterprise-гігантам (напр., ServiceNow)	*Конкурентний тиск:* ймовірність випуску спрощених і дешевших версій продуктів великими корпоративними гравцями
Технологічний розвиток: доступність сучасних фреймворків та AI-алгоритмів для швидкої реалізації складного функціоналу	*Економічна нестабільність:* скорочення бюджетів компаній на впровадження нового програмного забезпечення
	Кіберзагрози: зростання кількості хакерських атак на інфраструктурні корпоративні системи

Аналіз «Сили та можливості»:

- можна використати унікальну пропозицію об'єднання ITAM та CMDB в єдиному інструменті для агресивного заміщення застарілих або занадто дорогих (ServiceNow) систем у сегменті середнього бізнесу, позиціонуючи додаток як найкраще співвідношення функціоналу та ціни;
- спираючись на хмарну архітектуру продукту, можна запропонувати компаніям максимально швидке розгортання системи, що ідеально відповідає тренду на підтримку розподілених команд та віддалених IT-ресурсів.

Аналіз «Сили та загрози»:

- нівелювати загрози блекаутів та фізичного знищення серверних потужностей можна шляхом розміщення архітектури вебдодатка та

його резервних копій у надійних хмарних провайдерах таких як AWS або Azure, сервери яких знаходяться за межами України;

- гнучку мікросервісну архітектуру додатка можна використати для швидкого випуску нових функцій у відповідь на появу спрощених версій від конкурентів-гігантів, виграючи за рахунок швидкості прийняття рішень.

Аналіз «Слабкості та можливості»:

- компенсувати недовіру до нового продукту шляхом можна наданням безкоштовного пілотного періоду або базової версії для компаній середнього бізнесу, що дозволить напрацювати кейси успішного впровадження;
- замість розробки готових рішень для інтеграції сторонніх додатків, використати сучасні технології для створення потужного відкритого API, що дозволить клієнтам самостійно і легко інтегрувати платформу у свою екосистему.

Аналіз «Слабкості та загрози»:

- для управління ризиком раптової втрати ключових розробників через мобілізацію або відсутність зв'язку, необхідно впровадити жорстку дисципліну документування коду та перехресне навчання команди, щоб кожен учасник міг підстрахувати іншого.
- через скорочення бюджетів та брак ресурсів, розробка має вестися виключно за принципом MVP (Minimum Viable Product). Спочатку розробляється критичний функціонал інвентаризації, який одразу починає приносити фінансову вигоду замовнику, а складні аналітичні модулі відкладаються до наступних ітерацій.

Результати аналізу свідчать про те, що проєкт має потужний концептуальний фундамент: збалансоване поєднання ITAM та CMDB з орієнтацією на середній бізнес створює чітку конкурентну перевагу. Проте вебдодаток розробляється в умовах значної турбулентності зовнішнього середовища, зокрема через військову агресію в Україні, що генерує критичні

ризики для команди та інфраструктури. Внутрішні слабкості, зокрема обмеженість бюджету та інтеграцій, вимагають впровадження ітеративної моделі розробки, щоб якомога швидше доставити цінність користувачам.

Таблиця 1.3

Фактор середовища (SWOT)	**Стратегічне рішення**	**Вплив на архітектуру та функціонал**
Слабкість: обмеженість готових інтеграцій	Створення відкритого інтерфейсу взаємодії	Розробка документації та архітектури REST API як ключового компонента системи
Загроза: блекаути та ризики втрати зв'язку	Забезпечення безперебійної роботи клієнтської частини	Застосування механізмів локального збереження станів та використання технології SPA (Single Page Application)
Можливість: потреба ринку в економії бюджету	Пріоритезація швидкого досягнення економічної цінності	Визначення модуля ITAM (фінансовий облік) як першочергового для реалізації в MVP
Сила: Синергія ITAM, CMDB та ITSM	Створення цілісного інформаційного поля	Реалізація алгоритму, що пов'язує топологію (CMDB) із сервісними заявками (ITSM)

Основним результатом проведеного SWOT-аналізу є перелік стратегічних рішень, які безпосередньо вплинули на формування функціональних та архітектурних вимог до вебдодатку. Ці рішення продемонстровано у таблиці 1.3.

1.5 Аналіз зацікавлених сторін проєкту

Для того щоб гарантувати успішну реалізацію та подальше масштабування проєкту, вкрай важливо провести ретельний аналіз усіх

ключових зацікавлених сторін, визначивши їхні основні вимоги та інтереси. Головними стейкхолдерами виступають власники компаній та їх вище керівництво, які є безпосередніми ініціаторами впровадження нової системи в компанії. Їхніми пріоритетами є надійність і кіберзахист додатку, його здатність адаптуватися до корпоративних стандартів, а також наявність вичерпної аналітики IT-середовища, необхідної для стратегічного планування.

Наступною важливою категорією є фінансовий департамент, котрий потребує чітких показників оптимізації витрат на IT-ресурси та прогнозованого зниження загальної вартості володіння активами. Для них критичною буде детальна звітність про розрахунок амортизації обладнання, вартість ліцензій та дотримання бюджетних обмежень під час реалізації проєкту.

Для керівного складу IT-відділу та працівників служби технічної підтримки, які стануть основними щоденними користувачами платформи, головною цінністю є отримання зручних механізмів інвентаризації, стабільної CMDB-бази та засобів для оперативного виявлення та локалізації збоїв. Вони очікують інтуїтивно зрозумілого інтерфейсу для обробки заявок, гнучкого налаштування робочих процесів, автоматизації аудиту обладнання та зручної комунікації з іншими підрозділами.

Команда проєкту, яка безпосередньо створює додаток, прагне працювати з детально прописаними технічними завданнями, надійною інфраструктурою для тестування, а також мати змогу безперешкодно вносити послідовні покращення. Для них ключовим є своєчасне отримання зворотного зв'язку від замовників та реалістичні терміни календарних планів.

Окрему увагу слід приділити звичайним співробітникам компанії, який виступає в ролі кінцевих споживачів IT-послуг. Наявність зрозумілого порталу самообслуговування, доступних інструкцій та оперативне виконання їхніх заявок щодо техніки підвищить довіру до продукту та сприятиме його органічному впровадженню без супротиву змінам.

Також слід врахувати зовнішніх постачальників IT-обладнання та вендорів програмного забезпечення. Моніторинг їхніх контрактних вимог та інтеграційних можливостей стимулюватиме нас до постійного вдосконалення системи та впровадження автоматизованого контролю за дотриманням ліцензійних угод, для уникнення штрафів.

Насамкінець, успішний реліз продукту неможливий без підтримки підрозділів операційного забезпечення. Юридичний та HR-відділи мають бути своєчасно залучені для узгодження процедур онбордингу, підготовки нормативної бази та гарантування законності обробки корпоративних даних.

Зведена інформація щодо потреб кожної групи стейкхолдерів та ступеня їхнього впливу на проєкт структурована у таблиці 1.4, що слугує фундаментом для розробки збалансованої стратегії взаємодії та нівелювання комунікаційних ризиків.

Таблиця 1.4

Зацікавлена сторона	**Очікування від системи**	**Вплив на проєкт**
1	**2**	**3**
Власники та вище керівництво компанії	Прозора аналітика IT-інфраструктури, оптимізація витрат, безпека та стабільність системи, гнучка персоналізація під корпоративні стандарти	Високий: затверджують фінансування, приймають стратегічні рішення щодо долі проєкту

Продовження табл. 1.4

1	2	3
Фінансовий департамент	Точний облік матеріальних активів, автоматизований розрахунок амортизації обладнання та вартості ліцензій, дотримання бюджету розробки	Високий: контролюють бюджетні витрати проєкту та фінансові процеси закупівлі IT-ресурсів
IT-відділ та служба технічної підтримки	Автоматизація аудиту обладнання, єдина база залежностей (CMDB), зручний інтерфейс для управління заявками	Високий: основні користувачі та замовники функціоналу, формують технічні вимоги до продукту
Команда проєкту	Чітке технічне завдання, комфортні інструменти розробки, стабільне середовище тестування, реалістичні календарні плани	Середній: безпосередньо створюють продукт, впливають на його технічну якість та швидкість релізу
Кінцеві користувачі	Зрозумілий портал самообслуговування, швидка обробка заявок на заміну/ремонт техніки	Низький: є кінцевими споживачами послуг; визначають загальну задоволеність системою

Завершення табл. 1.4

1	2	3
Постачальники обладнання та IT-вендори	Зрозумілі умови інтеграції, дотримання ліцензійних угод, прозора звітність використання їхнього ПЗ.	Низький: впливають на архітектуру опосередковано, через надання технічних стандартів та вимог
Юридичний та HR-відділи	Дотримання політик конфіденційності, своєчасне керування доступами під час прийому та звільнення працівників	Середній: узгоджують нормативні регламенти та впливають на легальність операційних процесів системи

1.6 Постановка задачі дослідження, формування технічного завдання на розробку у вигляді паспорту проєкту

Згідно з результатами проведеного аналізу, корпоративна IT-інфраструктура стикається з потребою її модернізації. Замість застарілих ручних моделей обліку в електронних таблицях та використання подібних розрізнених програм необхідно впроваджувати сучасні консолідовані системи управління.

Таким чином, метою даного підрозділу є не лише визначення, а й детальна формалізація основних вимог до повноцінної розробки та впровадження у корпоративні системи описаного вебдодатка. Ця розробка повинна забезпечувати синергію управління життєвим циклом активів (ITAM) та реляційної бази конфігурацій (CMDB), що значно підвищить ефективність роботи служби технічної підтримки та сприятиме оптимізації ключових бізнес-процесів.

Поставлена задача: розробити та впровадити консолідований корпоративний вебдодаток з повноцінною клієнтською та серверною частиною для централізованого обліку, моніторингу та управління IT-ресурсами компанії на всіх етапах їх життєвого циклу.

Проблема: необхідно створити сучасний вебдодаток для управління корпоративною IT-інфраструктурою, що об’єднає засоби управління життєвим циклом активів (ITAM), ведення реляційної бази конфігурацій (CMDB) та інтеграції з процесами служби технічної підтримки.

Цілі проєкту визначаються наступним чином:

- розробити та розгорнути базову версію системи (MVP) протягом 3 місяців, а також здійснити повну реалізацію та остаточне впровадження корпоративного вебдодатка, що поєднує ITAM та CMDB, протягом 6 місяців з моменту старту робіт;
- спроєктувати та розробити інтуїтивно зрозумілий користувацький інтерфейс та систему інтерактивних дашбордів для різних ролей, що забезпечить швидкий доступ до аналітики та візуалізацію взаємозв’язків IT-інфраструктури;
- здійснити успішну автоматизовану міграцію 100% розрізнених історичних даних про апаратні та програмні активи компанії в нову єдину базу;
- знизити сукупні операційні витрати на закупівлю надлишкового обладнання та штрафи за порушення ліцензій щонайменше на 15% протягом першого року експлуатації;
- реалізувати інтеграцію процесів інвентаризації та служби технічної підтримки, що дозволить скоротити час обробки стандартних запитів користувачів на 20%;
- забезпечити відповідність архітектури та логіки додатка базовим практикам ITIL 4 та стандарту ISO/IEC 19770-1.

Цільовою аудиторією розроблюваного додатку стануть компанії малого та середнього бізнесу, які проходять через цифрову трансформацію або ж масштабують інфраструктуру, адже вони потребують гнучкої, швидкодіючої альтернативи дорогим та масштабним рішенням, які представлені на ринку. Кінцевими користувачами стануть керівники IT-відділу, системні адміністратори, фахівці технічної підтримки, працівники фінансового відділу та звичайні співробітники цих компаній.

Географія додатку, завдяки хмарній архітектурі яка використовується, технічно не обмежена, однак першочерговий реліз та пілотні впровадження орієнтуються на компанії з України.

Основні завдання проєкту:

- вивчення поточної інфраструктури, виявлення її недоліків, складання детального технічного завдання та побудова архітектури бази даних;
- створення прототипів та дизайну інтерфейсів для дашбордів керівника, порталу підтримки та візуальної топології мережі;
- написання програмного коду клієнтської та серверної частин вебдодатка, реалізація логіки парсингу даних, історії змін та системи рівнів допуску до певних наборів операцій;
- розгортання реляційної бази даних, налаштування алгоритмів автоматичного відстеження термінів дії ліцензій;
- проведення навантажувального та приймального тестуваннь із залученням ключових стейкхолдерів;
- фінальна міграція даних у промислову базу, офіційний запуск вебдодатку, навчання персоналу, збір та аналіз зворотного зв'язку.

Очікувані результати будуть наступними:

- зменшення часу на проведення регулярної повної інвентаризації IT-інфраструктури компанії на 40% завдяки впровадженню автоматизованого обліку;

- скорочення середнього часу реагування служби технічної підтримки на критичні збої на 25% завдяки інтерактивній мапі взаємозв’язків між складовими IT-інфраструктури;
- зниження частки неврахованих активів у корпоративній мережі до рівня, що не перевищує 2%;
- досягнення рівня задоволеності кінцевих користувачів до показника у щонайменше 85% завдяки сучасному дизайну порталу самообслуговування.

Продуктом проєкту будуть:

- повністю функціональний, готовий до використання вебдодаток що об’єднає засоби управління життєвим циклом активів (ITAM), ведення реляційної бази конфігурацій (CMDB) та інтеграції з процесами служби технічної підтримки;
- пакет проєктної та технічної документації, що включає архітектурні схеми, API-специфікації та регламенти взаємодії для користувачів системи;
- комплексний план управління проєктом, за яким здійснювалась реалізація.

1.7 Функціональні та нефункціональні вимоги вебдодатка

Враховуючи все, що було описано у попередньому підпункті, були розроблені наступні функціональні (таблиця А.1) вимоги. Ця таблиця описує базовий функціонал, який вебдодаток повинен безпосередньо реалізовувати для задоволення бізнес-потреб користувачів.

Нефункціональні вимоги (таблиця А.2), які були розроблені аналогічно базуючись на викладеній інформації у попередньому підпункті, описують атрибути якості системи, тобто як саме вона повинна працювати, щоб забезпечувати надійність, зручність та безпеку.

РОЗДІЛ 2. РОЗРОБКА КОНЦЕПЦІЇ ПРОЄКТУ

2.1. Розробка концептуальної моделі інформаційної системи

Цінність запропонованого вебдодатку полягає у його здатності трансформувати хаотичні процеси управління корпоративною IT-інфраструктурою у впорядковану, автоматизовану екосистему. Як показано на розробленій концептуальній моделі (рис. 2.1), центром архітектури виступає середовище управління IT-ресурсами, яке функціонує як центральний хаб для всіх операційних завдань. За його межами розташовані зовнішні зацікавлені сторони такі як постачальники апаратного забезпечення та вендори програмних рішень. Саме вони впливають на макропроцеси, формуючи вимоги, та встановлюючи ліцензійні обмеження. У саму ж систему на вхід надходить масив первинної інформації, який являє собою неструктуровані дані про існуюче IT-обладнання, регулярні заявки до служби технічної підтримки, а також повідомлення про технічні збої. Усередині середовища виділено дві підсистеми. Одна з них це корпоративна команда, яка ініціює процеси та підтримує працездатність мережі, друга ж являє собою функціональні модулі керування IT-ресурсами.

Права частина запропонованої моделі зображує роботу самого вебдодатку. Цей технічний фундамент працює паралельно з бізнес-процесами та відповідає за консолідацію інформації про техніку, координацію стосовно виконання заявок, консолідацію даних з різних відділів та запуск алгоритмів, що будують топологію IT-інфраструктури та контролюють терміни дії ліцензій. Сполучною ланкою між корпоративною командою та машинною підсистемами додатку є єдиний API-шар, який гарантує безперебійну та просту комунікацію. Таким чином кожна дія, така як реєстрація нового ноутбука чи повідомлення про збій на сервері, миттєво оброблюється та передається між відповідними складовими середовища управління IT-ресурсами компанії.

У результаті описаної багатошарової взаємодії система забезпечує чіткий результат, адже початкові неструктуровані дані перетворюються на централізований реєстр IT-активів, автоматично оновлювану та актуальну CMDB, а всі зареєстровані заявки успішно вирішуються.

Для спеціалістів служби технічної підтримки та системних адміністраторів вебдодаток пропонує інструменти для занесення нового обладнання до системи, оновлення його статусу, а також прив'язки конкретних заявок та збоїв до проблемних пристроїв. Завдяки інформативним дашбордам фахівці бачать обсяг відкритих заявок у режимі реального часу та отримують push-сповіщення щодо критичних подій, таких як серверні помилки чи закінчення терміну дії гарантії на обладнання чи ліцензії. Це дозволяє мінімізувати час реакції на аварії та підвищує загальну злагодженість роботи відділу технічної підтримки.

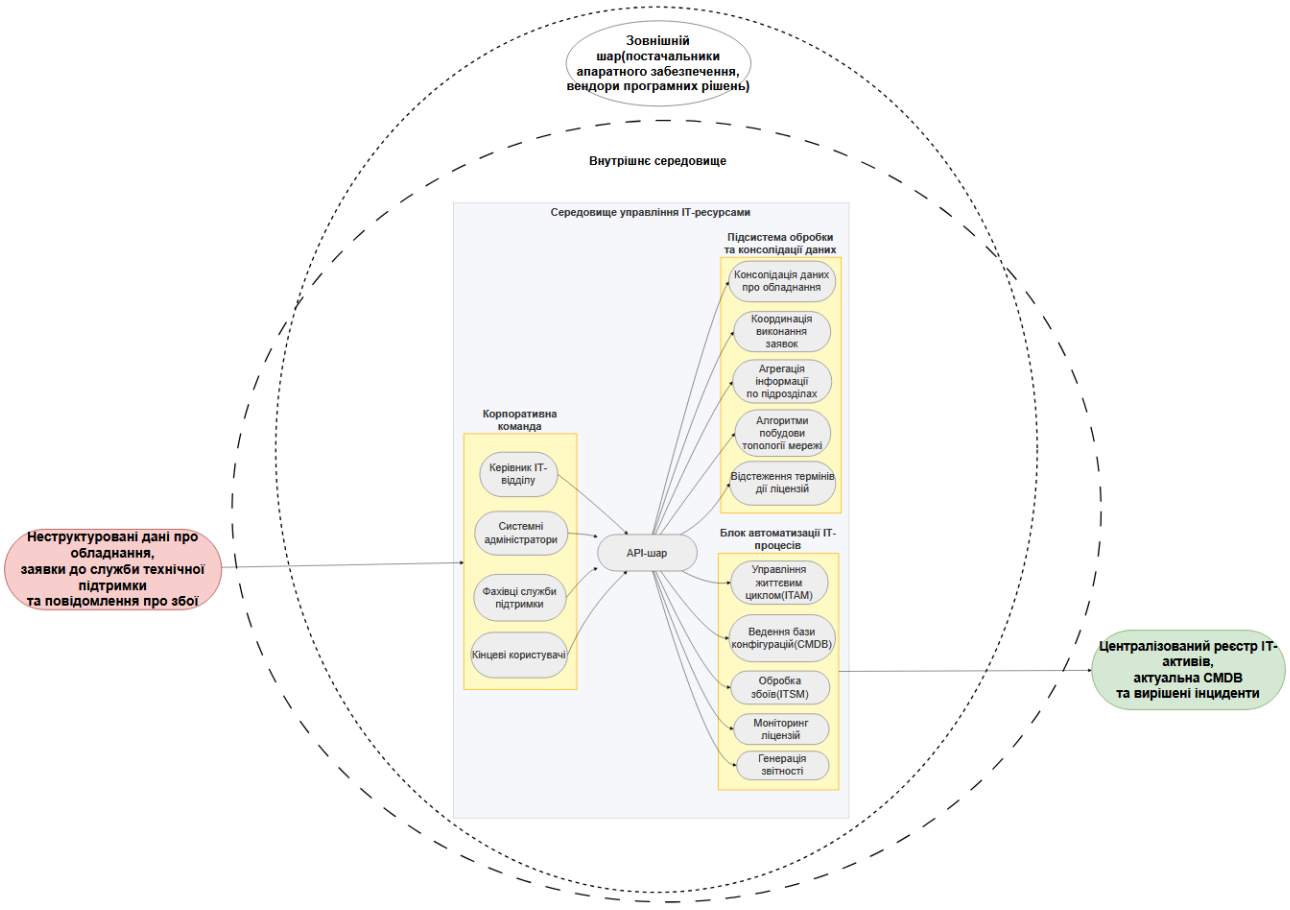

Рисунок 2.1. Концептуальна модель системи

Керівник ІТ-відділу може використовувати можливості вебдодатку для прийняття обґрунтованих управлінських рішень та контролю бюджетів. Інтерфейс надає змогу оцінювати загальну вартість володіння усією технікою та аналізувати витрати на її амортизацію. Найважливішим інструментом на цьому рівні є візуальна мапа CMDB, яка ілюструє взаємозв’язки між кожною складовою всієї ІТ-інфраструктури компанії. Тобто, вона демонструє як потенційна відмова одного комутатора вплине на критичні бізнес-сервіси. Крім того, завдяки єдиній базі даних керівник може гарантувати ліцензійну чистоту ПЗ компанії та точно прогнозувати фінансові потреби на майбутні закупівлі.

Обов'язки системного адміністратора самої платформи зводяться до її технічного налаштування. Цей спеціаліст керує конфігураціями робочих просторів, створюючи матриці доступу для різних відділів компанії, контролює системні метрики вебдодатка, наприклад швидкість обробки запитів у базі, забезпечуючи безперебійність та продуктивність роботи інструменту для всієї організації.

Загалом, така модель забезпечує баланс між щоденним операційним обслуговуванням мережі та глобальним стратегічним ІТ-менеджментом. Вебдодаток надає кожному учаснику процесу спеціалізований інструментарій, перетворюючи розрізнену інфраструктуру на прозорий, надійно захищений та фінансово контрольований актив підприємства.

2.2. Розробка математичної моделі інформаційної системи

Математичне моделювання корпоративного вебдодатка дозволяє формалізувати логіку взаємодії його базових підсистем (ITAM, CMDB, ITSM), уникнути неоднозначностей під час програмування бізнес-логіки та забезпечити коректність роботи алгоритмів обробки даних. Для представлення вебплатформи як цілісного об’єкта, її математичну модель доцільно описати як динамічну систему за допомогою кортежу станів та переходів:

$$W = \langle X, Y, Z, F, G \rangle, \quad (2.1)$$

де X – множина вхідних сигналів (реєстрація нових активів, запити на зміну статусів, звіти про інциденти);

Y – множина вихідних реакцій системи (автоматичні сповіщення, топологічні мапи ураження, звіти про виконання SLA);

Z – простір внутрішніх станів системи, що визначається поточним станом баз даних ITAM, CMDB та ITSM;

$F: Z \times X \to Z$ – функція переходів, що описує зміну станів активів та зв'язків;

$G: Z \times X \to Y$ – функція виходів, що формує реакцію системи на вхідні події.

Загальну структуру інформаційних об'єктів у просторі Z можна подати у вигляді кортежу множин:

$$S = \langle U, A, L, T, \Phi \rangle, \quad (2.2)$$

де S – загальна структура інформаційних об'єктів;

U – множина користувачів системи;

A – множина апаратних активів;

L – множина ліцензій на програмне забезпечення;

T – множина інцидентів (збоїв) та сервісних заявок;

Φ – множина функціональних відображень, що описують бізнес-процеси.

Розглянемо математичну модель *управління активами (ITAM).* Процес закріплення обладнання за працівником описується функцією розподілу активів f_{alloc}, яка ставить у відповідність конкретному активу $a_i \in A$ (i-й апаратний актив з множини активів) одного користувача $u_j \in U$ (j-й користувач з множини працівників):

$$f_{alloc}: A \to U \cup \{\emptyset\}, \quad (2.3)$$

де f_{alloc} – функція розподілу обладнання;

A – множина апаратних активів;

U – множина користувачів системи;

$\emptyset$ – порожня множина.

Якщо $f_{alloc}(a_i) = \emptyset$, тобто значення функції для i-го активу є порожньою множиною, актив знаходиться на складі, або ж інакше кажучи є вільним ресурсом. Для автоматизації фінансового обліку, що включає в себе розрахунок амортизації активу, використовується лінійна модель зносу. Залишкова вартість активу V_{res} для i-го активу в момент часу τ (поточний місяць експлуатації) обчислюється за формулою:

$$V_{res}(a_i, \tau) = max\left(0, V_{init}(a_i) - \frac{V_{init}(a_i)}{\Delta T_{life}(a_i)} \times \tau\right), \quad (2.4)$$

де $V_{res}(a_i, \tau)$ – залишкова вартість i-го активу в момент часу τ;

$V_{init}(a_i)$ – початкова балансова вартість i-го активу при закупівлі;

$\Delta T_{life}(a_i)$ – нормативний термін експлуатації i-го активу (у місяцях);

τ – кількість місяців, що минула з дати введення активу в експлуатацію.

Сукупна вартість володіння TCO одиницею обладнання формується як сума початкової вартості та витрат на обслуговування й ліцензії за весь період експлуатації t:

$$TCO(a_i) = V_{init}(a_i) + \sum_{x=1}^{t}\left(C_{maint}(a_i, x) + C_{lic}(a_i, x)\right), \quad (2.5)$$

де $TCO(a_i)$ – сукупна вартість володіння i-м активом;

$V_{init}(a_i)$ – початкова вартість i-го активу;

t – загальний період експлуатації в місяцях;

x – поточний місяць експлуатації;

$C_{maint}(a_i, x)$ – витрати на технічне обслуговування i-го активу в місяці x;

$C_{lic}(a_i, x)$ – витрати на програмні ліцензії для i-го активу в місяці x.

Тепер розглянемо *графову модель бази конфігурацій (CMDB).*

Основою CMDB є топологія взаємозв'язків між елементами інфраструктури (серверами, роутерами, сервісами). Ця структура найточніше описується за допомогою моделі орієнтованого графа G_{CMDB}:

$$G_{CMDB} = (V, E), \tag{2.6}$$

де G_{CMDB} – модель орієнтованого графа топології інфраструктури;

$V \subseteq A$ – множина вершин графа, що є конфігураційними одиницями;

$E \subseteq V \times V$ – множина дуг, що позначають спрямовані залежності між вузлами.

Наявність дуги $e = (v_i, v_j)$ означає, що працездатність вузла v_i залежить від працездатності вузла v_j.

У разі виникнення збою на вузлі v_{fail}, система повинна визначити множину всіх потенційно уражених сервісів V_{impact}. Ця задача зводиться до знаходження транзитивного замикання графа для вершини v_{fail}.

Якщо існує шлях від v_x до v_{fail}, то $v_x \in V_{impact}$:

$$V_{impact}(v_{fail}) = \{v \in V \mid \exists x: v_x \to v_{fail}\}, \tag{2.7}$$

де $V_{impact}(v_{fail})$ – множина уражених вузлів внаслідок збою на вузлі v_{fail};

v_{fail} – вузол, на якому безпосередньо виникла проблема;

v – довільна вершина з множини всіх конфігураційни одиниць V;

V – загальна множина всіх вершин графа;

$\exists x$: $v_x \rightarrow v_{fail}$ – логічна умова існування орієнтованого шляху від вершини v до вершини v_{fail}.

Система сповіщень про терміни дії ліцензій працює на основі функції перевірки статусів. Нехай $T_{exp}(l_k)$ – дата закінчення k-ї ліцензії, а T_{curr} – поточна дата. Функція генерування тригера сповіщення Ω визначається як:

$$\Omega(l_k) = \{\text{Warning, якщо } T_{exp(l_k)} - T_{curr} \leq 14\}, \quad (2.8)$$

де $\Omega(l_k)$ – функція генерування попередження для k-ї ліцензії;

l_k – k-та програмна ліцензія з множини ліцензій L;

$T_{exp}(l_k)$ – дата закінчення терміну дії k-ї ліцензії;

T_{curr} – поточна системна дата.

Розглянемо *модель управління інцидентами (ITSM)*. Обробка заявок служби підтримки базується на параметрах часу та пріоритетів. Кожен інцидент t_p характеризується моментом створення τ_{open} та моментом закриття τ_{close}. Фактичний час вирішення TTR:

$$TTR(t_p) = \tau_{close}(t_p) - \tau_{open}(t_p), \quad (2.9)$$

де $TTR(t_p)$ – фактичний час вирішення для інциденту t_p;

t_p – p-й інцидент (заявка) з множини T;

$\tau_{close}(t_p)$ – точний час та дата закриття інциденту t_p;

$\tau_{open}(t_p)$ – точний час та дата відкриття інциденту t_p.

Виконання *угоди про рівень послуг* SLA є булевою функцією, яка залежить від максимального допустимого часу вирішення T_{max} , закріпленого за пріоритетом заявки $Pr(t_p)$:

$$SLA(t_p) = \begin{cases} 1, & \text{якщо } TTR(t_p) \leq T_{max}\left(Pr(t_p)\right) \\ 0, & \text{якщо } TTR(t_p) > T_{max}\left(Pr(t_p)\right) \end{cases}, \quad (2.10)$$

де $SLA(t_p)$ – булева функція дотримання угоди SLA для інциденту t_p;

$TTR(t_p)$ – фактичний витрачений час на вирішення інциденту t_p;

$T_{max}(Pr(t_p))$ – максимальний допустимий час вирішення, регламентований для інциденту з пріоритетом $Pr(t_p)$;

$Pr(t_p)$ – рівень пріоритету інциденту t_p.

Для оцінки ефективності роботи IT-відділу за період розраховується показник успішності SLA_{Rate}:

$$SLA_{Rate} = \frac{\sum_{p=1}^{N} SLA(t_p)}{N} \times 100\%, \quad (2.11)$$

де SLA_{Rate} – показник успішності виконання регламентів SLA;

$\sum_{p=1}^{N} SLA(t_p)$ – сумарна кількість успішно вирішених інцидентів в межах регламентованого часу;

N – загальна кількість вирішених інцидентів за обраний звітний період.

Перейдемо до *інтеграційної моделі міжмодульних зв'язків*. Її суть полягає у формалізації прямої залежності між топологією інфраструктури та параметрами обслуговування. Пріоритет інциденту $Pr(t_p)$ є функцією від потужності множини впливу в CMDB:

$$Pr(t_p) = \psi(|V_{impact}(v_{fail})|), \quad (2.12)$$

де ψ – функція ранжування пріоритету (чим більша кількість уражених вузлів, тим вищий пріоритет інциденту). Це, у свою чергу, динамічно змінює обмеження часу вирішення проблеми в моделі ITSM:

$$T_{max}\left(Pr(t_p)\right) \rightarrow min \text{ при } |V_{impact}| \rightarrow \max \quad (2.13)$$

Таким чином, сформована математична модель не просто описує окремі підсистеми, а встановлює аналітичні зв'язки між технічним станом інфраструктури (CMDB), фінансовими показниками (ITAM) та оперативною діяльністю служби підтримки (ITSM), що дозволяє мінімізувати час простою

критичних бізнес-сервісів. Описані теоретико-множинні конструкції будуть використані для проєктування архітектури реляційної бази даних, графова модель стане основою для алгоритмів модуля CMDB, а аналітичні залежності дозволять реалізувати бекенд-логіку для розрахунку амортизації активів та контролю дотримання SLA.

РОЗДІЛ 3. РОЗРОБКА ІНФОРМАЦІЙНОЇ СТРУКТУРИ ПРОЄКТУ

3.1. Вибір технології управління проєктом

Історичний розвиток методів управління ІТ-проєктами пройшов шлях від суворих послідовних систем до адаптивних практик. Загалом існуючі парадигми поділяються на класичні лінійні та гнучкі. Рішення щодо використання певної методології завжди залежить від особливостей майбутнього продукту. Каскадні підходи виправдовують себе там, де всі специфікації чітко визначені на старті, а внесення правок на етапі написання коду коштує надто дорого [15]. Однак під час створення сучасних вебрішень, коли потреби клієнтів динамічно змінюються разом із технологічними трендами, найбільш ефективним та раціональним є використання саме гнучких фреймворків [16].

Враховуючи функціональну спрямованість вебдодатку на роботу з IT-активами (ITAM) та ведення конфігураційної бази (CMDB), найоптимальнішим вибором для управління розробкою є застосування Scrum-фреймворку як частини методології Agile [17-18]. Головна перевага цього вибору полягає в ітеративності: весь цикл створення додатку ділиться на невеликі часові проміжки, які називаються спринтами, що тривають від одного до чотирьох тижнів. Це чудово підходить для побудови архітектури ITAM-продуктів, адже дає змогу випускати та перевіряти систему покроково. Наприклад, перший спринт може завершитися релізом базового обліку техніки та підсистеми авторизації, а наступний додаванням функцій життєвого циклу ресурсів чи розширеного пошуку.

До того ж, проєктування платформи для менеджменту корпоративних ІТ-ресурсів є багатовимірною задачею. Вона передбачає глибоке вбудовування в ІТ-процеси компанії та розробку спеціалізованих компонентів, таких як відстеження ліцензій на ПЗ, фінансовий аналіз хмарних послуг чи сканування мережевого обладнання [18]. Scrum дозволяє отримати оптимальний баланс, адже з одного боку він не скасовує важливості

попереднього проєктування надійної структури бази даних для точного фінансового обліку, а з іншого залишає простір для безболісного коригування вимог завдяки безперервним відгукам замовника. Такий підхід суттєво знижує ймовірність критичних помилок на етапах підключення зовнішніх API або налаштування складної бізнес-логіки.

Ще однією перевагою цього підходу є те, що роботу можна розпочати невеликою багатофункціональною командою для швидкого запуску MVP (мінімально життєздатного продукту). Наприклад, це може бути елементарний реєстр обладнання, а відповідно до того, як додаток трансформуватиметься у комплексну екосистему з модулем ITSM, процеси легко адаптуватимуться під залучення нових розробників без втрати керованості. Запорукою успіху в такій моделі є налагоджена комунікація, яка включає в себе щоденні короткі зустрічі, демонстрації результатів та ретроспективи, які допомагають вчасно знаходити архітектурні недоліки, гнучко керувати пріоритетами завдань і підтримувати абсолютну поінформованість ключових стейкхолдерів.

3.2. Розробка ієрархічної структури проєкту

Наведена модель етапу ініціалізації (рис 3.1) відображає фазу закладення фундаменту проєкту, де абстрактні бізнес-потреби перетворюються на формалізовані цілі. Для розробки корпоративного вебдодатка цей етап є критичним кроком узгодження бачень стейкхолдерів та визначення меж майбутньої системи. За відсутності глибокого аналізу конкурентів (1.1.2) продукт ризикує успадкувати типові недоліки існуючих рішень або виявитися неконкурентоспроможним на рівні архітектури. Недостатнє опрацювання концептуальних вимог (1.1.3) закладе хибний вектор розвитку архітектури, через що ключові модулі інвентаризації (ITAM) та бази конфігурацій (CMDB) можуть виявитись структурно несумісними при спробі їхньої інтеграції в єдину платформу.

Рисунок 3.1. Ієрархічна структура робіт. Фаза ініціалізації

Етап планування (рис. 3.2) відображає фазу архітектурного моделювання, на якій загальні концепції перетворюються на деталізовані технічні специфікації, структури даних та логічні схеми. Для комплексного вебдодатку управління IT-активами цей етап є запорукою масштабованості та безвідмовності. За відсутності правильної формалізації графової моделі (1.2.2.3) майбутній модуль CMDB не зможе коректно обробляти складні топології корпоративної мережі. Недолік продуманої математичної моделі системи (1.2.3) закладе критичні помилки на самому старті, що зробить неможливим точний розрахунок амортизації активів та призведе до фінансових втрат під час подальшої експлуатації продукту.

Рисунок 3.2. Ієрархічна структура робіт. Фаза планування

Наведена модель етапу розробки системи (рис. 3.3) відображає фазу безпосередньої технічної реалізації, де закладені на етапі планування архітектурні рішення перетворюються на функціонуюче програмне забезпечення. Для вебдодатку цей етап є критичним моментом об'єднання модулів ITAM, CMDB та ITSM у цілісне інтегроване середовище. Недостатньо якісна реалізація обчислювального ядра (1.3.2.2) призведе до того, що фінансова статистика генеруватиметься з похибками, що знецінить систему для IT-менеджменту. Неефективна логіка обходу графа (1.3.3.2) призведе до того, що під час реальних збоїв в інфраструктурі система не зможе вчасно визначати уражені вузли, зводячи нанівець усю користь від підсистеми управління інцидентами.

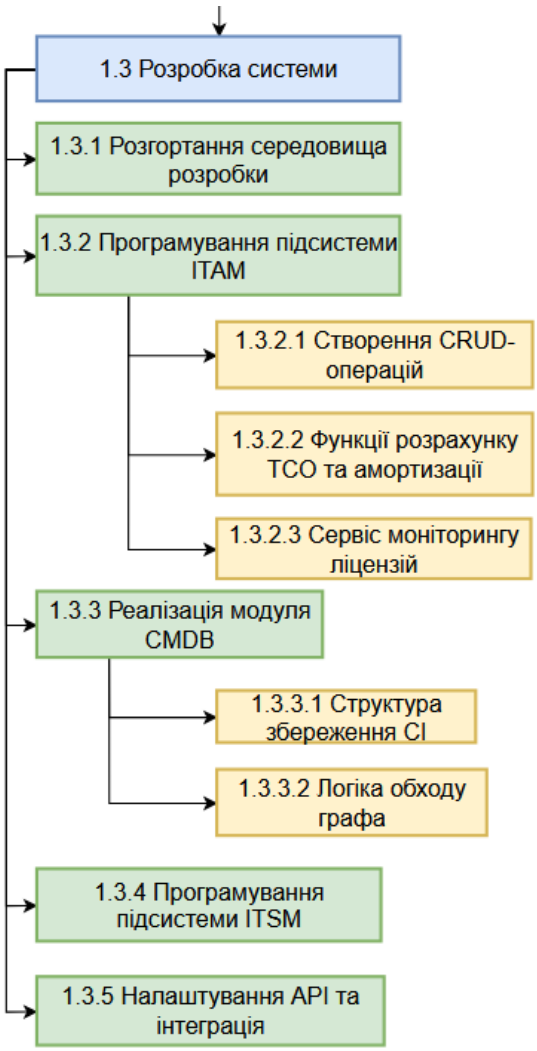

Рисунок 3.3. Ієрархічна структура робіт. Фаза розробки системи

Наведена модель етапу впровадження системи (рис. 3.4) ілюструє процес фінальної валідації, коли програмні компоненти перевіряються на стабільність та інтегруються на серверах замовника. Брак комплексного тестування взаємодії модулів (1.4.1.2) гарантовано спровокує системні колізії між модулем обробки інцидентів (ITSM) та реєстром активів (ITAM), загрожуючи цілісності інвентаризаційних баз. Крім того, невідповідне ставлення до етапу дослідної експлуатації (1.4.4) призведе до відмови від продукту стейкхолдерами через неперсоналізовані та незручні робочі сценарії.

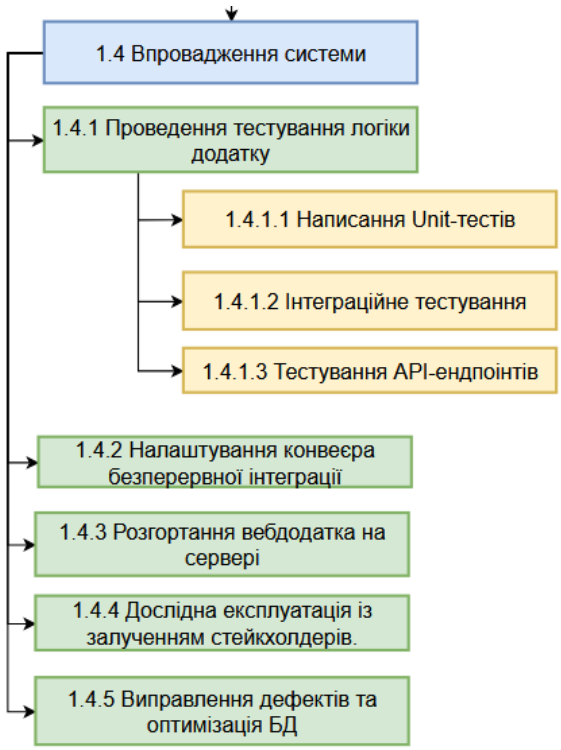

Рисунок 3.4. Ієрархічна структура робіт. Фаза впровадження системи

Наведена модель етапу завершення (рис. 3.5) відображає фазу формального закриття розробки та передачі продукту замовнику. За відсутності якісної експлуатаційної документації (1.5.1) подальше обслуговування вебдодатка стане критично залежним від розробників, унеможливлюючи самостійну підтримку платформи IT-відділом.

У результаті розроблено деталізовану ієрархічну структуру робіт (WBS) для проєкту створення вебдодатку для управління IT-ресурсами компанії. Рівень деталізації до четвертого рівнів вкладеності дозволив не лише структурувати інженерні завдання, але й ідентифікувати критичні точки проєкту на кожній фазі. Зокрема, було чітко розмежовано процеси проєктування баз даних, розробки логіки модулів ITAM, CMDB та ITSM, а також процеси їхнього подальшого тестування та інтеграції.

Рисунок 3.5. Ієрархічна структура робіт. Фаза завершення

3.3. Організаційна структура проєкту та розподіл відповідальності

Для успішної реалізації вебдодатка сформовано гібридну організаційну структуру, яка поєднує вертикаль корпоративного управління (для звітності перед замовником) та горизонтальну взаємодію в межах крос-функціональної Scrum-команди (для гнучкої розробки). Важливою особливістю команди є чіткий поділ інженерних ролей на спеціалізовані напрямки розробки, що гарантує високу якість як серверної логіки, так і клієнтського інтерфейсу.

Нижче наведено детальний склад учасників проєкту та їхні персональні зони відповідальності:

- *Спонсор проєкту* – вищий керівник з боку замовника, який затверджує стратегічні цілі, бюджет та приймає остаточне рішення про готовність системи до промислової експлуатації;
- *Керівник проєкту (Project manager)* – здійснює загальний адміністративний нагляд, звітує перед спонсором проєкту,

контролює терміни та фінансові ресурси, а також управляє комунікацією з усіма стейкхолдерами;

- *Product owner* – відповідає за бізнес-цінність продукту, формує беклог, визначає пріоритетність функцій, узгоджує вимоги із замовником та приймає готові етапи розробки;
- *Scrum master* – забезпечує дотримання Scrum-процесів, організовує командні зустрічі, допомагає усувати перешкоди та сприяє самоорганізації колективу;
- *Головний архітектор (Software architect)* – формує технічну стратегію, відповідає за цілісність системи, вибір технологічного стека та високорівневе проєктування взаємодії всіх модулів бази даних;
- *Бізнес-аналітик* – деталізує вимоги, проводить аналіз конкурентів, описує логіку бізнес-процесів та трансформує їх у технічні завдання;
- *UI/UX дизайнер* – проєктує інтерфейс користувача, створює макети та інтерактивні прототипи;
- *Бекенд-розробник (Backend developer)* – відповідає за серверну логіку, проєктування реляційних та графових баз даних, інтеграцію API та реалізацію алгоритмів обробки даних;
- *Фронтенд-розробник (Frontend developer)* – відповідає за клієнтську частину, а саме реалізацію інтерфейсів на основі макетів, забезпечення адаптивності та взаємодії з сервером;
- *QA-інженер* – здійснює контроль якості, проводить тестування, перевіряє відповідність продукту вимогам та фіксує дефекти;
- *DevOps-інженер* – налаштовує серверне середовище та забезпечує стабільне розгортання додатка.

Організаційна діаграма (рис 3.6) відображає структуру адміністративного підпорядкування в проєкті.

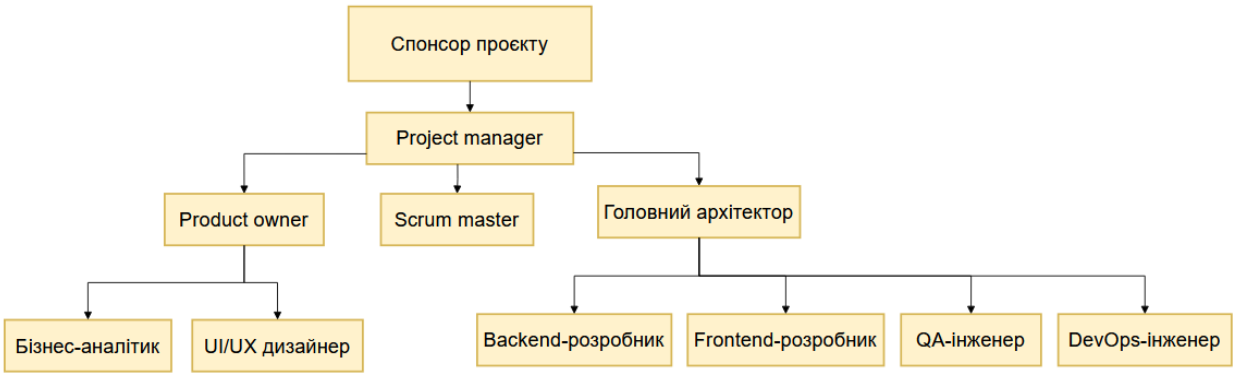

Рисунок 3.6. Організаційна структура проєкту

Для забезпечення прозорості процесів та уникнення дублювання функцій сформовано матрицю RACI (таблиця 3.1) [20].

RACI-матриця

Таблиця 3.1

Процес	**PM**	**PO**	**SM**	**SA**	**BA**	**UI/ UX**	**BD**	**FD**	**QA**	**DO**	**S**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Ініціалізація проєкту та узгодження статуту	A/R	C	I	C	C	I	I	I	I	I	C
Збір та уточнення вимог	I	A	C	C	R	C	C	C	I	I	I
Проєктування технічної архітектури	I	I	I	A	C	I	R	C	C	C	I
UI/UX дизайн	I	A	I	C	C	R	I	C	I	I	I

Завершення табл. 3.1

1	2	3	4	5	6	7	8	9	10	11	12
Планування та формування беклогу	I	A	R	C	C	C	I	I	C	C	I
Розробка серверної логіки та обчислювального ядра	I	A	I	C	C	I	R	C	I	I	I
Розробка WEB-компонентів клієнтської частини	I	S	I	C	I	C	C	R	I	I	I
Тестування	I	I	I	I	C	I	C	C	A/R	I	I
Підготовка та розгортання продукту на сервері	I	I	I	C	I	I	I	I	I	A/R	I
Дослідна експлуатація та навчання	I	A	I	I	R	C	I	I	C	C	C
Реліз продукту в промислову експлуатацію	A	C	I	I	I	I	I	I	I	R	C
Аналіз результатів	A	C	R	C	C	C	C	C	C	C	I

У цій таблиці вище PM, PO, SM, SA, BA, UI/UX, BD, FD, QA, DO та S позначають проєктного менеджера, product owner’a, scrum master’a, головного

архітектора, бізнес-аналітика, UI/UX-дизайнера, бекенд-розробника, фронтенд-розробника, QA-інженера, DevOps-інженера та спонсора проєкту відповідно.

Літери R, A, C та I позначають виконавця (особу що безпосередньо виконує роботу), відповідального (особу, яка приймає остаточне рішення та затверджує результат), консультанта (фахівець, з яким радяться під час виконання) та поінформовану людину (особа, яку повідомляють про результат виконання) відповідно.

3.4. Управління ресурсами та бюджетом проєкту

Ефективність управління життєвим циклом IT-активів та побудови цілісної структури CMDB безпосередньо залежить від професійного кадрового забезпечення та інструментарію розробки платформи. Загальний бюджет проєкту розрахований на 6 місяців і включає в себе витрати на оплату праці залучених фахівців, оренду усієї необхідної інфраструктури для розробки та розгортання вебдодатку та формування резервного фонду для нівелювання проєктних ризиків.

Для успішної реалізації запланованих робіт та досягнення визначених цілей необхідні наступні фахівці:

- Керівник проєкту (Project manager);
- Product owner;
- Scrum Master;
- Головний архітектор;
- Бізнес-аналітик;
- UI/UX Дизайнер;
- 2 Backend-розробники;
- 2 Frontend-розробники;
- QA-інженер;
- DevOps-інженер.

Такий кількісний та якісний склад команди дозволяє одночасно працювати над серверною частиною та клієнтським інтерфейсом, забезпечуючи безперервну інтеграцію та тестування функціоналу.

Для раціонального використання бюджету проєкту розробки вебдодатка тривалість залучення фахівців була оптимізована відповідно до фаз життєвого циклу розробки. Зокрема, термін роботи спеціалістів, чий основний внесок припадає на етапи проєктування та формування вимог, було обмежено часом їх активної фази та подальшої підтримки. Загальні витрати на оплату праці продемонстровані у таблиці 3.2.

Трудові витрати

Таблиця 3.2

Посада	**Місячна заробітня плата (грн)**	**Кількість**	**Термін залучення (міс.)**	**Загальні витрати (грн)**
1	**2**	**3**	**4**	**5**
Керівник проєкту	80 000	1	6	480 000
Product owner	90 000	1	6	540 000
Scrum master	60 000	1	6	360 000
Головний архітектор	140 000	1	6	840 000
Бізнес-аналітик	70 000	1	4	280 000
UI/UX Дизайнер	60 000	1	4	240 000
Backend-розробник	110 000	2	6	1 320 000

Завершення табл. 3.2

Frontend-розробник	90 000	2	6	1 080 000
QA-інженер	50 000	1	6	300 000
DevOps-інженер	100 000	1	6	600 000
Загальна сума				6 040 000

Для забезпечення безперервного процесу розробки, тестування та стабільної роботи вебдодатка до бюджету закладено витрати на матеріально-технічну базу. Використання моделі оренди дозволяє забезпечити команду всім необхідним інструментарієм без значних капітальних інвестицій на старті (таблиця 3.3).

Витрати на матеріальне забезпечення

Таблиця 3.3

Стаття витрат	**Опис**	**Щомісячна вартість (грн)**	**Термін (міс.)**	**Загальні витрати (грн)**
1	**2**	**3**	**4**	**5**
Хмарний хостинг	Оренда серверів та сховища	14 000	6	84 000
Керована БД	Хмарний сервіс Neo4j Aura для CMDB	7 500	6	45 000
Оренда ноутбуків	12 робочих пристроїв	26 400	6	158 400

Завершення табл. 3.3

1	2	3	4	5
Інструменти розробки	Jira, GitHub Team (12 ліцензій)	8 500	6	51 000
ПЗ для дизайну	Професійна ліцензія Figma	2 500	6	15 000
Безпека та домени	SSL, домен, Firewall	2 500	6	15 000
Загальна сума:				368 400

На основі розрахованих витрат на трудові ресурси та інфраструктурне забезпечення сформовано загальний фінансовий план реалізації проєкту (таблиця 3.4).

Загальний бюджет

Таблиця 3.4

Категорія витрат	Сума(грн)
Трудові витрати	6 040 000
Матеріальне забезпечення	368 400
Резерв на ризики (~10%)	640 840
Загально	7 049 240

Запропонований загальний бюджет у розмірі 7 049 240 гривень (без резерву на ризики 6 408 400 гривень) є фінансовим обґрунтуванням для розробки вебдодатку для управління IT-ресурсами. Структура витрат демонструє раціональний підхід, адже понад 85% бюджету спрямовано на залучення висококваліфікованих людських ресурсів, тоді як витрати на

інфраструктуру оптимізовано за рахунок хмарних сервісів та оренди обладнання. Резервування близько 10% коштів гарантує фінансову стійкість проєкту у разі виникнення непередбачуваних технічних обставин під час інтеграції модулів ITAM та CMDB.

3.5. Календарне планування проєкту

Календарне планування розробки вебдодатка базується на фреймворку Scrum. Оскільки проєкт має визначені часові межі, які були вказані в цілях проєкту при формуванні паспорту проєкту, а саме загальна тривалість 6 місяців, з випуском MVP через 3 місяці, розробка організована навколо досягнення ключових бізнес-цілей через серію ітерацій. Весь період поділено на двотижневі спринти (загалом 12 спринтів), у кінці кожного з яких команда постачає робочий інкремент продукту. Датою старту проєкту є 1 червня 2026 року.

Для синхронізації високорівневих цілей команди та очікувань замовника використовується дорожня карта продукту (таблиця 3.5). Вона фокусується на реалізації ключових задач та не прив'язана до жорстких щоденних графіків на макрорівні.

Дорожня карта продукту

Таблиця 3.5

Спринти	Дата проведення	Ключові задачі до виконання	Очікуваний результат
1	2	3	4
Спринти 1-2	01.06.2026 – 28.06.2026	Проєктування архітектури реляційної БД та графа CMDB, створення базового UI Kit	Розгорнута базова інфраструктура, затверджена схема бази даних та дизайн-система

Завершення табл. 3.5

1	2	3	4
Спринти 3-6	29.06.2026 – 23.08.2026	Розробка логіки обліку активів та зв'язків графа, візуалізація зв'язків у CMDB, інтеграція та тестування базового функціоналу	Працююча версія системи для управління основними активами
Спринти 7-10	24.08.2026 – 18.10.2026	Розробка модуля обробки заявок (ITSM), прив'язка заявок до конкретних вузлів графа CMDB, налаштування системи сповіщень та розширених прав доступу	Функціонал обробки заявок, повністю інтегрований з базою активів
Спринти 11-12	19.10.2026 – 15.11.2026	Реалізація модуля аналітики та експорту звітів, навантажувальне тестування, виправлення помилок, дослідна експлуатація та передача системи	Готовий до промислової експлуатації вебдодаток

Такий підхід дозволяє зберегти гнучкість у реагуванні на зміни вимог замовника, але при цьому чітко контролювати прогрес наближення до фінального релізу.

Для оперативного управління задачами в межах ітерацій та візуалізації робочого процесу розробки вебдодатка було обрано платформу ClickUp [21]. Вибір цього хмарного рішення зумовлений його архітектурною гнучкістю та здатністю ефективно поєднувати елементи різних методологій управління в єдиному інформаційному просторі. Основними перевагами для її використання є:

- *мультиформатність візуалізації:* платформа дозволяє працювати з єдиним масивом даних через різні відображення, таким чином задачі спринту можна одночасно планувати у вигляді структурованого списку, відстежувати їхній прогрес на класичній Kanban-дошці (Board view) та контролювати часові рамки й залежності на діаграмі Ганта (Gantt view);
- *гнучка система статусів* надає можливість налаштування унікальних етапів життєвого циклу задачі, що дозволяє точно відобразити реальний процес проходження коду від написання до тестування та фінального затвердження;
- всі учасники команди мають доступ до актуального стану проєкту в режимі реального часу, що мінімізує комунікаційні розриви.

Важливою складовою календарного планування є управління робочим навантаженням, щоб гарантувати рівномірний розподіл задач і відсутність перетинів в обов'язках. Відповідно до визначеного раніше складу команди, у ClickUp використовується подання Workload view, яке дозволяє оцінити завантаженість кожного учасника. У двотижневому спринті на одного фахівця, залученого на повну ставку, закладається 80 робочих годин.

Використання цього інструментарію забезпечить високу прозорість процесу реалізації спринтів та дозволить сформувати наочну звітність щодо статусу виконання ключових задач. Нижче наведений потенційний результат

виконання спринту №4. Його термін проведення буде з 13 липня 2026 року по 26 липня 2026 року. Ціль цього спринта полягає у реалізації можливості створення та редагування IT-активів через інтерфейс та забезпечення першої ітерації відображення зв'язків у графі CMDB.

Під час цього спринту команда розробки зосередилася на реалізації базового функціоналу управління IT-активами (рис 3.7), що є критичним кроком для підготовки MVP. Робота була побудована паралельно: поки UI/UX дизайнер готував макети модальних вікон для відображення детальної інформації про техніку, Backend-команда реалізувала REST API для операцій створення, читання, оновлення та видалення (CRUD) активів, а також заклала логіку автоматичної побудови зв'язків у графовій базі даних Neo4j.

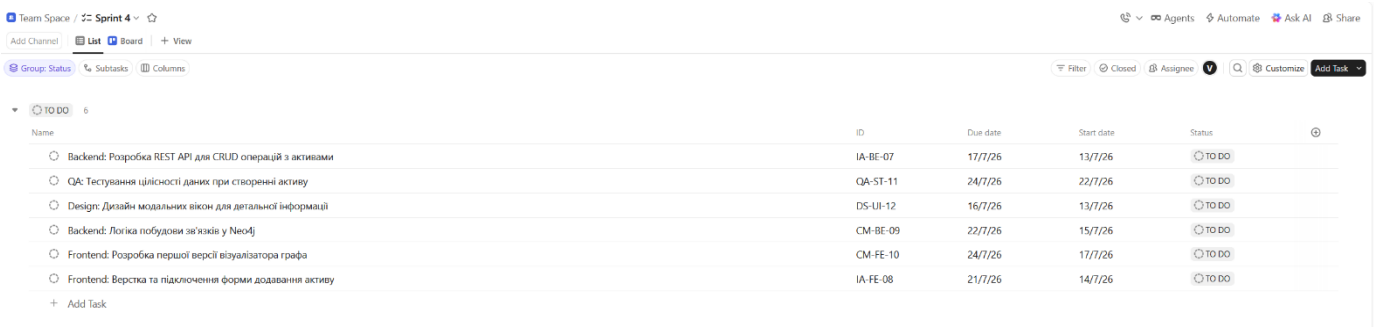

Рисунок 3.7. Перелік задач спринту №4

Frontend-частина команди синхронізувала свої дії з бекендом, успішно виконавши верстку та підключення форм додавання нових активів в інтерфейсі. QA-інженер провів тестування цілісності запису даних одночасно у дві бази (реляційну та графову), що дозволило перевести відповідні задачі у статус Complete. Єдиною задачею, яка на кінець ітерації залишилася у статусі In progress (рис 3.8), стала розробка клієнтської частини візуалізатора графа CMDB. Оскільки відображення складних вузлів та ребер за допомогою сторонньої бібліотеки виявилося ресурсомістким, код передано головному архітектору для фінальної перевірки та оптимізації продуктивності перед остаточним прийняттям задачі.

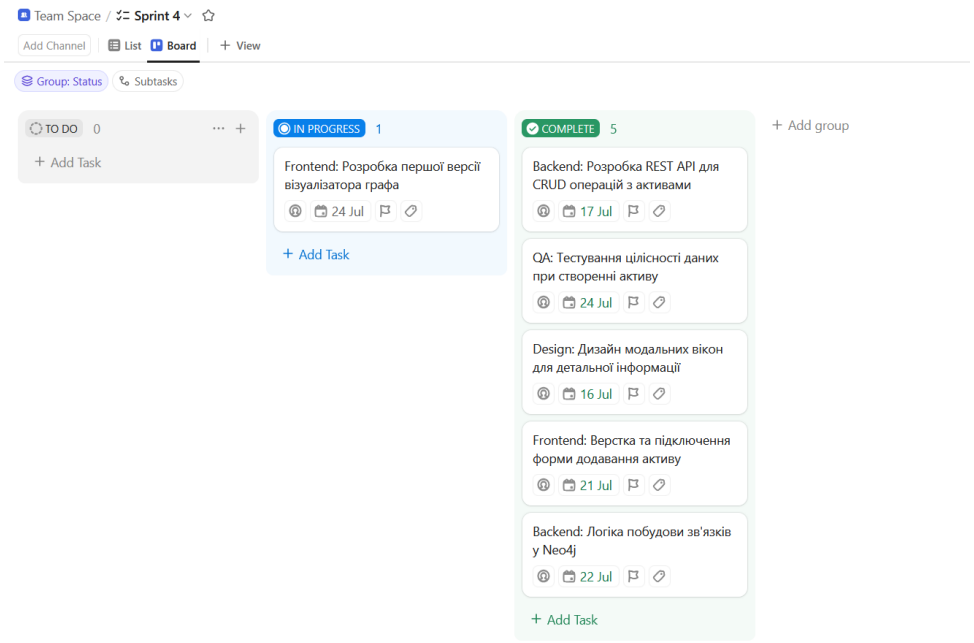

Рисунок 3.8. Результат спринту №4

На рис. 3.9 зображена діаграма Ганта яка показує розподіл робіт виконаних під час спринту №4.

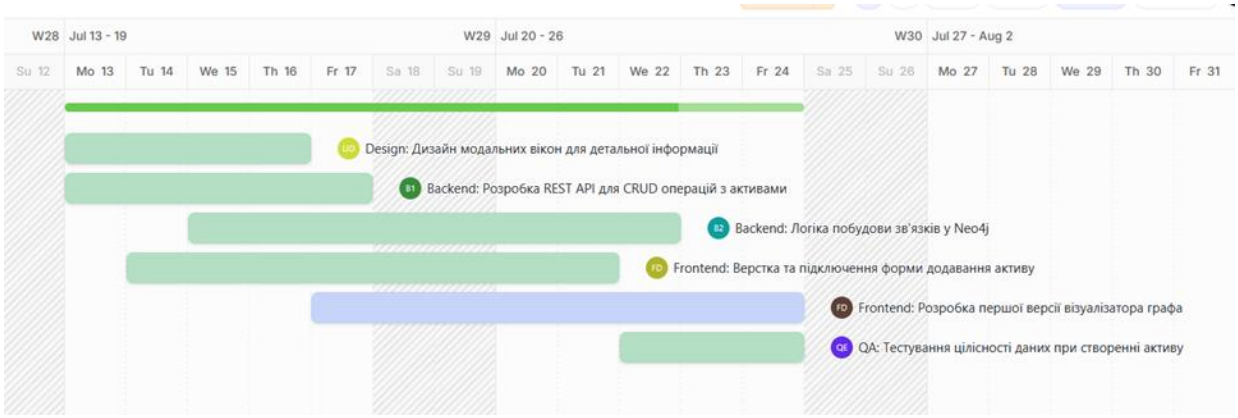

Рисунок 3.9. Діаграма Ганта спринту №4

Ефективність розподілу ресурсів продемонстровано на рисунку Б.1. Як видно з візуалізації, завантаженість кожного спеціаліста не перевищує нормативні 8 робочих годин на день. Задачі розподілені таким чином, щоб уникнути простоїв: наприклад, тестувальник (QA Engineer) залучається до активної фази лише після завершення базової розробки бекенду та фронтенду, що в свою чергу гарантує оптимальне використання оплачуваного часу спеціалістів.

Розглянемо ще спринт №8, головною метою ітерації було створення механізму обробки заявок на обслуговування з жорсткою прив'язкою інцидентів до конкретних елементів інфраструктури (CMDB). Термін його проведення буде з 7 вересня 2026 року по 20 вересня 2026 року.

Backend-розробники успішно спроєктували нову схему даних для збереження заявок та реалізували складну логіку зв'язування інциденту з унікальним ідентифікатором пристрою в системі. Паралельно було налаштовано мікросервіс для відправки Email-сповіщень при зміні статусу заявки. Зі свого боку Frontend-розробники завершили створення окремого дашборду служби технічної підтримки, де користувачі можуть зручно фільтрувати свої заявки (рис 3.10). Якість реалізованого бекенду була підтверджена навантажувальним тестуванням (імітація масового створення заявок), і всі ці задачі успішно перейшли у колонку Done на Канбан-дошці.

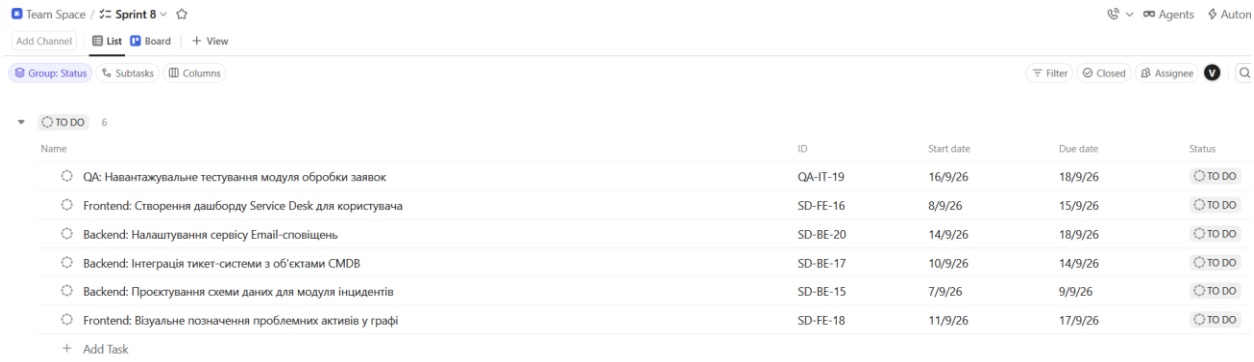

Рисунок 3.10. Перелік задач спринту №8

Якість реалізованого бекенду була підтверджена навантажувальним тестуванням, і всі ці задачі успішно перейшли у колонку Complete на Канбан-дошці (рис. 3.11).

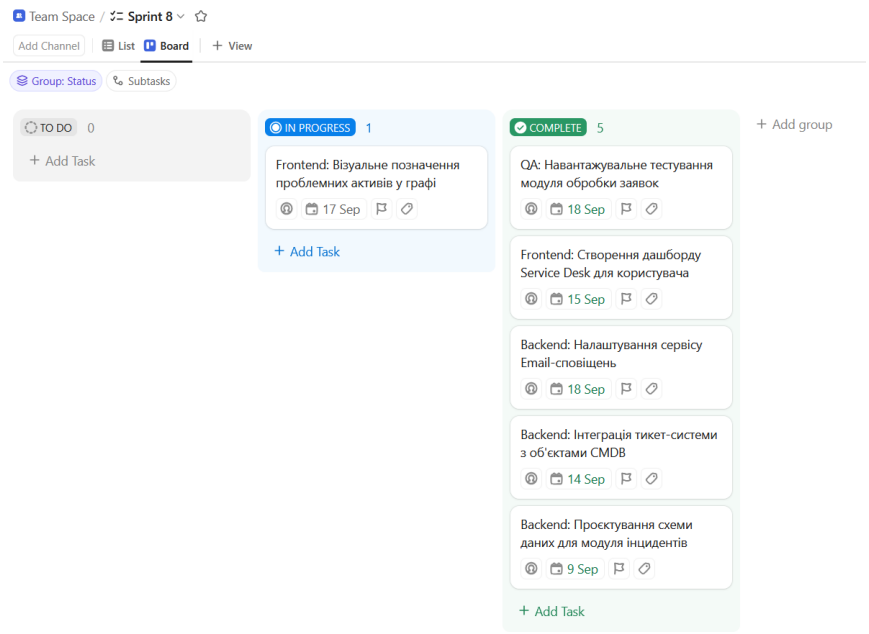

Рисунок 3.11. Результат спринту №8

На етапі перевірки наразі знаходиться задача з візуального позначення проблемних активів. Frontend-команда реалізувала механізм, який змінює колір вузла графа, якщо до цього активу прив'язано відкритий інцидент.

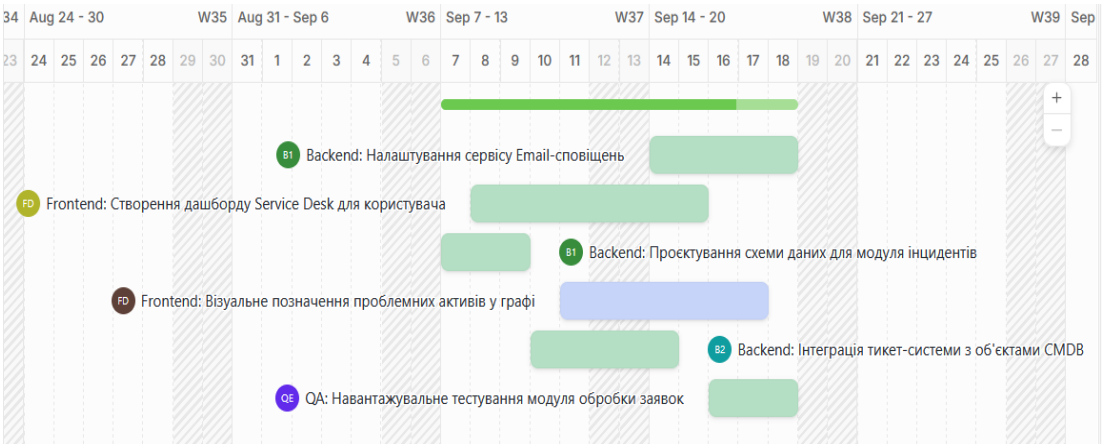

Рисунок 3.12. Діаграма Ганта спринту №8

На рисунку 3.12 показана діаграма Ганта яка демонструє розподіл робіт виконаних під час спринту №8.

На рисунку Б.2 наведено побудований системою графік завантаженості для команди під час спринту №8. Як і в попередніх спринтах, планування здійснено з суворим дотриманням норми у 8 робочих годин на день для кожного спеціаліста. Візуалізація підтверджує, що паралельне виконання задач та чітке дотримання часових оцінок дозволило уникнути критичних

затримок та простоїв на етапі інтеграції бекенду з клієнтською частиною та забезпечило своєчасну передачу функціоналу на фінальне тестування без необхідності понаднормової роботи.

3.6. Управління ризиками проєкту

Успішна реалізація комплексних IT-проєктів, зокрема розробка корпоративного вебдодатка із поєднанням модулів ITAM та графової структури CMDB, безпосередньо пов'язана з умовами високої невизначеності. Ігнорування потенційних загроз на етапах проєктування та розробки може призвести до критичних наслідків, зокрема зриву узгоджених термінів , перевищення затвердженого бюджету або невідповідності фінального продукту бізнес-вимогам замовника. Саме тому складений завчасний план управління ризиками є невід’ємною складовою життєвого циклу проєкту.

На першому етапі здійснюється комплексний аналіз потенційних загроз, що передбачає виявлення ймовірних ризикових подій, оцінку ступеня їхнього деструктивного впливу на результати проєкту, а також визначення рівня керованості цими факторами (таблиця 3.6).

Ідентифікація ризиків

Таблиця 3.6

Категорія ризику	**Ризикова подія**	**Ступінь впливу**	**Керованість**
1	**2**	**3**	**4**
Програмні	Складність інтеграції реляційної та графової баз даних	Висока	Середня

Продовження табл. 3.6

1	2	3	4
Програмні	Критичні помилки або вразливості у сторонніх бібліотеках	Висока	Висока
	Несумісність API між Backend та Frontend на етапі злиття	Середня	Висока
Апаратні	Вихід з ладу орендованої техніки	Середня	Висока
	Збої або тимчасова недоступність хмарних серверів	Висока	Низька
Внутрішні	Втрата ключових фахівців (мобілізація, хвороба, звільнення)	Висока	Середня
	Неточна оцінка трудовитрат та відставання від графіка розробки	Висока	Середня
	Погана комунікація в команді	Середня	Висока

Завершення табл. 3.6

1	2	3	4
Зовнішні	Неконтрольована зміна вимог замовником	Висока	Середня
	Затримки замовника у наданні даних або доступів	Середня	Середня
Форс-мажорні обставини	Тривалі перебої з електропостачанням (блекаути) та зв'язком	Висока	Низька
	Обставини непереборної сили (ескалація бойових дій, надзвичайні ситуації)	Висока	Низька

Для проведення кількісного та якісного аналізу ризиків проєкту використано систему кількісної оцінки (табл. 3.8) за чотирма показниками, а саме ймовірність, часовий дефіцит, економічні втрати та функціональна деградація (рівень зниження якості або обсягу можливостей системи, ФД).

Кожен показник оцінюється за бальною шкалою, наведеною у таблиці 3.7, де у відповідність простій оцінці ставляться деталізована розширена та числова квазікількісна оцінки.

Шкала оцінювання ризикових подій

Таблиця 3.7

Проста	Розширена	Квазікількісна оцінка
Відсутня	В	0
Низька	НН (низька-низька)	1
	НС (низька-середня)	2
	НВ (низька-висока)	3
Середня	СН (середня-низька)	4
	СС (середня-середня)	5
	СВ (середня-висока)	6
Висока	ВН (висока-низька)	7
	ВС (висока-середня)	8
	ВВ (висока-висока)	9
Катастрофічна	К	10

Оцінювання ризиків проєкту

Таблиця 3.8

Ризикова подія	Ймовірність		Часовий дефіцит		Економічні втрати		ФД		Загальний бал
	ЯО	КО	ЯО	КО	ЯО	КО	ЯО	КО	
1	2	3	4	5	6	7	8	9	10
Складність інтеграції PostgreSQL та Neo4j	С	5	В	8	СН	4	ВН	7	24

Завершення табл. 3.8

1	2	3	4	5	6	7	8	9	10
Критичні помилки або вразливості у сторонніх бібліотеках	СН	4	СВ	6	НВ	3	С	5	18
Несумісність API між Backend та Frontend	СВ	6	СН	4	Н	2	НВ	3	15
Вихід з ладу орендованої техніки	Н	2	НВ	3	СН	4	НН	1	10
Збої або тимчасова недоступність хмарних серверів	НВ	3	ВН	7	С	5	В	8	23
Втрата фахівців (мобілізація, хвороба, звільнення)	СВ	6	ВВ	9	ВН	7	В	8	30
Неточна оцінка трудовитрат та відставання від графіка розробки	В	8	В	8	СВ	6	С	5	27
Погана комунікація в команді	НВ	3	СН	4	Н	2	НВ	3	12
Неконтрольована зміна вимог замовником	ВВ	9	В	8	СВ	6	ВН	7	30
Затримки замовника у наданні даних або доступів	ВН	7	СВ	6	СН	4	СН	4	21
Тривалі перебої з електропостачанням (блекаути) та зв'язком	ВВ	9	В	8	С	5	СВ	6	28
Обставини непереборної сили (ескалація бойових дій)	НВ	3	К	10	В	8	К	10	31

Для забезпечення стабільності розробки вебдодатка наступним кроком стало формування комплексного плану реагування на ідентифіковані загрози. Цей план базується на трирівневій моделі управління: запобіганні, відстеженні індикаторів та розробці алгоритмів дій на випадок кризи. Такий підхід дозволяє не лише превентивно нівелювати загрози на ранніх етапах, але й мати готові сценарії мінімізації збитків у разі їх фактичного настання.

Нижче наведено розроблений план протиризикових заходів (табл. 3.9) для всіх визначених раніше загроз.

Протиризикові заходи

Таблиця 3.9

Ризикова подія	**Профілактика**	**Симптом**	**Дії при симптомі**	**Дії при проблемі**
1	**2**	**3**	**4**	**5**
Складність інтеграції PostgreSQL та Neo4j	Розробка методу ранньої перевірки ідеї на старті, залучення досвідченого архітектора БД.	Затримки у виконанні задач бекенду на етапі побудови зв'язків графа	Проведення технічного рев'ю, оптимізація та рефакторинг запитів	Відкат до стабільної версії, тимчасове спрощення архітектури графа
Критичні помилки або вразливості у сторонніх бібліотеках	Використання стабільних версій перевірених бібліотек	Попередження від систем безпеки GitHub, падіння автотестів	Оновлення пакета до патч-версії, перевірка сумісності коду	Заміна бібліотеки на аналог, або написання власного кастомного рішення

Продовження табл. 3.9

1	2	3	4	5
Несумісність API між Backend та Frontend	Використання Swagger/OpenAPI, спільне затвердження контрактів до написання коду	Помилки формату 400/500 при інтеграційних перевірках	Синхронізація на зустріч BE та FE, термінове оновлення документації API	Створення тимчасових Mocks для FE, оперативне виправлення контролерів BE
Вихід з ладу орендованої техніки	Укладання договору оренди з гарантією заміни	Скарги на перегрів, зависання обладнання	Технічна діагностика, звернення до компанії-орендодавця	Оперативна заміна обладнання за договором, перехід на резервний ПК
Збої або тимчасова недоступність хмарних серверів	Вибір надійного провайдера, налаштування	Зниження швидкості відповіді, алерти від моніторингу	Аналіз логів навантаження, збільшення обчислювальних потужностей	Розгортання резервних інстансів з бекапів, інформування замовника

Продовження табл. 3.9

Втрата фахівців (мобілізація, хвороба, звільнення)	Підтримка якісної документації коду, перехресне навчання	Часті відгули, зниження продуктивності, отримання повістки	Передача критичних знань іншому розробнику, початок пошуку заміни	Перерозподіл задач між командою, терміновий найм нового фахівця
Неточна оцінка трудовитрат та відставання від графіка розробки	Планування через Planning Poker, декомпозиція задач, закладання резервів часу	Систематичне невиконання цілей спринту, накопичення задач	Ретроспективний аналіз причин затримки, зменшення обсягу на наступний спринт	Скорочення функціоналу на користь MVP, залучення овертаймів
Погана комунікація в команді	Обов'язкові щоденні зустрічі, чітке ведення статусів	Дублювання роботи, непорозуміння вимог, пропуски мітингів	Проведення тет-а-тет зустрічей, нагадування регламенту комунікації	Жорсткий мікроменеджмент з боку PM, додаткові синхронізаційні зустрічі

Продовження табл. 3.9

1	2	3	4	5
Неконтрольована зміна вимог замовником	Жорстка фіксація ТЗ, процедура управління змінами	Постійні пропозиції нових функцій під час демонстрації інкременту	Оцінка вартості нових вимог, фіксація їх у беклозі на майбутні релізи	Офіційна відмова від реалізації поза межами поточного бюджету та термінів
Затримки замовника у наданні даних або доступів	Завчасні запити, фіксація обов'язків клієнта	Ігнорування листів, перенесення зустрічей щодо передачі інформації	Підняття питання на рівень вищого керівництва замовника	Тимчасове заморожування залежних задач та переведення команди на розробку ізольованих модулів системи
Тривалі перебої з електропостачанням (блекаути)	Забезпечення команди зарядними станціями	Поява або посилення графіків відключень світла	Робота з коворкінгів або відповідно до графіків	Перехід в асинхронний режим, компенсація годин у вихідні дні

Завершення табл. 3.9

1	2	3	4	5
Обставини непереборної сили (ескалація бойових дій)	Зберігання всього коду та баз даних виключно в захищених хмарних середовищах	Погіршення безпекової ситуації в регіоні перебування членів команди	Перевірка цілісності бекапів, підготовка до можливої релокації	Призупинення проєкту, активація протоколу безпеки персоналу

Підсумовуючи результати етапу планування ризиків, слід зазначити, що розроблений комплекс заходів формує надійний механізм захисту проєкту від впливу дестабілізуючих факторів різної природи (технічних, організаційних, зовнішніх та форс-мажорних). Практична імплементація даного плану забезпечить адаптивність команди розробки в умовах змін, мінімізує ймовірність критичних відхилень від затвердженого календарного графіка та бюджету і, як наслідок, гарантуватиме успішне та своєчасне введення корпоративної системи в промислову експлуатацію.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ ПРОЄКТУ

4.1. Розробка інформаційної структури проєкту

Високорівнева архітектура (рис. 4.1) вебдодатка побудована за класичною багаторівневою клієнт-серверною моделлю (N-Tier Architecture) [22]. Враховуючи вимоги до інтеграції модулів ITAM, CMDB та ITSM, систему спроєктовано як модульний моноліт або сервіс-орієнтовану архітектуру з використанням гібридного підходу до зберігання даних.

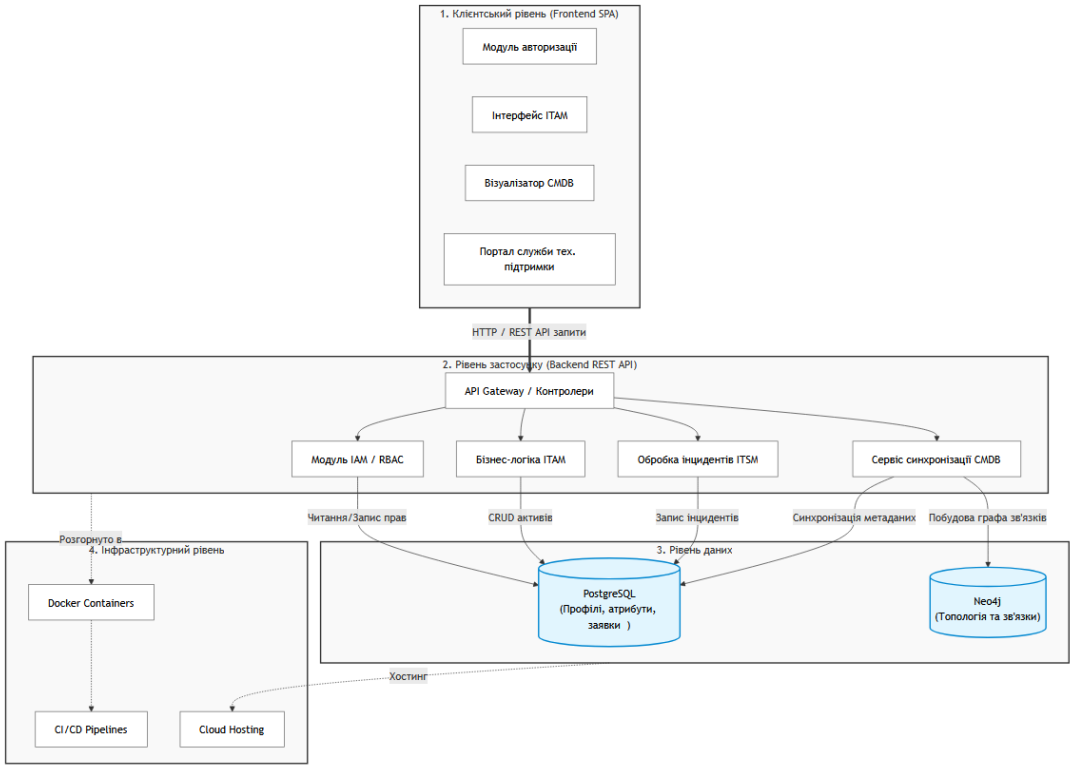

Рисунок 4.1. Високорівнева архітектура вебдодатка

Інформаційна структура поділяється на чотири основні рівні: клієнтський, рівень застосунку (серверний), рівень даних та інфраструктурний рівень.

Клієнтський рівень відповідає за взаємодію кінцевого користувача із системою. Він реалізується у вигляді Single Page Application (SPA) [23], що забезпечує високу швидкість роботи та плавність інтерфейсу без постійного перезавантаження сторінок. Модуль авторизації та роутингу потрібен для перевірки прав доступу та відображення відповідних елементів навігації.

Інтерфейс управління активами (ITAM UI) являє собою таблиці з підтримкою пагінації, сортування та фільтрації для перегляду і редагування IT-активів, включає форми для внесення нових пристроїв та ліцензій. Модуль візуалізації графа це спеціалізований компонент, який використовує бібліотеки на кшталт Cytoscape.js для інтерактивного відображення топології мережі та зв'язків між конфігураційними одиницями. Портал служби технічної підтримки (ITSM UI) являє собою дашборд для подачі заявок, відстеження статусів заявок та комунікації з технічною підтримкою.

Рівень застосунку це серверна частина системи, яка обробляє всі бізнес-процеси, здійснює валідацію вхідних даних та керує логікою взаємодії з базами даних. Зв'язок із клієнтським рівнем відбувається через REST API [24]. API Gateway потрібен для забезпечення маршрутизації та обробки HTTP-запитів. Модуль аутентифікації та авторизації (IAM) відповідає за управління сесіями користувачів, хешування паролів та контроль доступу на основі ролей (RBAC). Модуль бізнес-логіки ITAM необхідний для обробки операцій життєвого циклу активів, такі як закупівля, введення в експлуатацію, списання. Сервіс синхронізації CMDB це ключовий архітектурний вузол, який забезпечує транзакційну цілісність під час збереження об'єкта, адже він при створенні нового активу паралельно записує його метадані в реляційну БД, а структурний вузол і зв'язки в графову БД. Модуль ITSM та сповіщень реалізує логіку маршрутизації інцидентів, прив'язки заявок до конкретних конфігураційних одиниць та інтеграцію із зовнішніми сервісами для сповіщень.

На рівні даних, основою архітектури вебдодатка є гібридна модель збереження даних, яка використовує дві різні СКБД для вирішення специфічних задач. Реляційна база даних (PostgreSQL) [25] використовується як основне сховище для структурованих даних, зокрема вона містить дані про профілі користувачів, права доступу, детальні атрибути IT-активів (серійні номери, гарантійні терміни, вартість), текстові описи тощо. Графова база даних (Neo4j) [26] використовується виключно для розрахунку топології та

аналізу впливу, тобто вона зберігає вузли та ребра які є зв’язками між інфраструктурними одиницями, що в свою чергу дозволяє миттєво визначати, які сервіси вийдуть з ладу при падінні конкретного сервера.

Інфраструктурний рівень забезпечує середовище для розгортання, масштабування та безперебійної роботи вебдодатка. Контейнеризація (Docker) відповідає за розміщення Frontend, Backend та баз даних в ізольованих контейнерах для гарантії ідентичності середовищ розробки та продакшену. CI/CD Pipeline [27] відповідає за автоматизацію процесів тестування та розгортання коду при кожному впроваджені змін. Хмарний хостинг використовується для хостингу контейнерів, а також сервісів об'єктного зберігання для прикріплених до заявок файлів.

Для детального розуміння логіки роботи системи було розроблено діаграми послідовностей, які є різновидом діаграм інформаційних потоків. Вони ілюструють ключові сценарії та демонструють покроковий обмін даними між користувачем, клієнтською частиною, сервером та базами даних.

Перший сценарій (рис 4.2), який мною розглядається це реєстрація нового IT-активу. Він демонструє головну архітектурну особливість проєкту паралельний запис даних у дві різні бази.

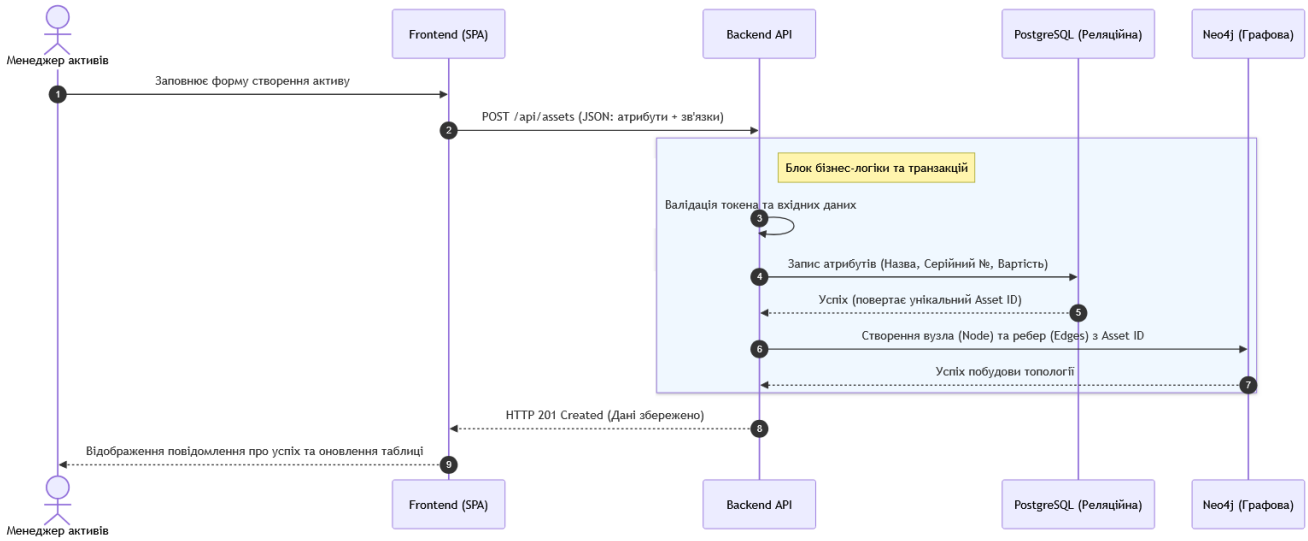

Рисунок 4.2. Сценарій реєстрації нового IT-активу

Процес розпочинається з того, що менеджер активів вводить дані про новий IT-ресурс через клієнтський інтерфейс та вказує його апаратні чи

мережеві підключення. Frontend-додаток формує відповідний запит і відправляє його на серверний API, де відбувається валідація вхідних даних та перевірка прав доступу. Після цього бізнес-логіка ініціює збереження текстових та фінансових атрибутів у реляційній базі даних. Отримавши підтвердження успішного запису та унікальний ідентифікатор активу, система паралельно створює відповідний структурний вузол та прокладає зв'язки в графовій базі. На фінальному етапі користувач отримує системне повідомлення про успішне збереження даних, а інтерфейс таблиці оновлюється для відображення актуальної інформації.

Другий сценарій (рис 4.3), який мною розглядається це обробка інциденту (збою) та аналіз впливу. Він показує, як модуль служби технічної підтримки використовує графову базу даних для того, щоб зрозуміти, на кого вплине поломка конкретного пристрою.

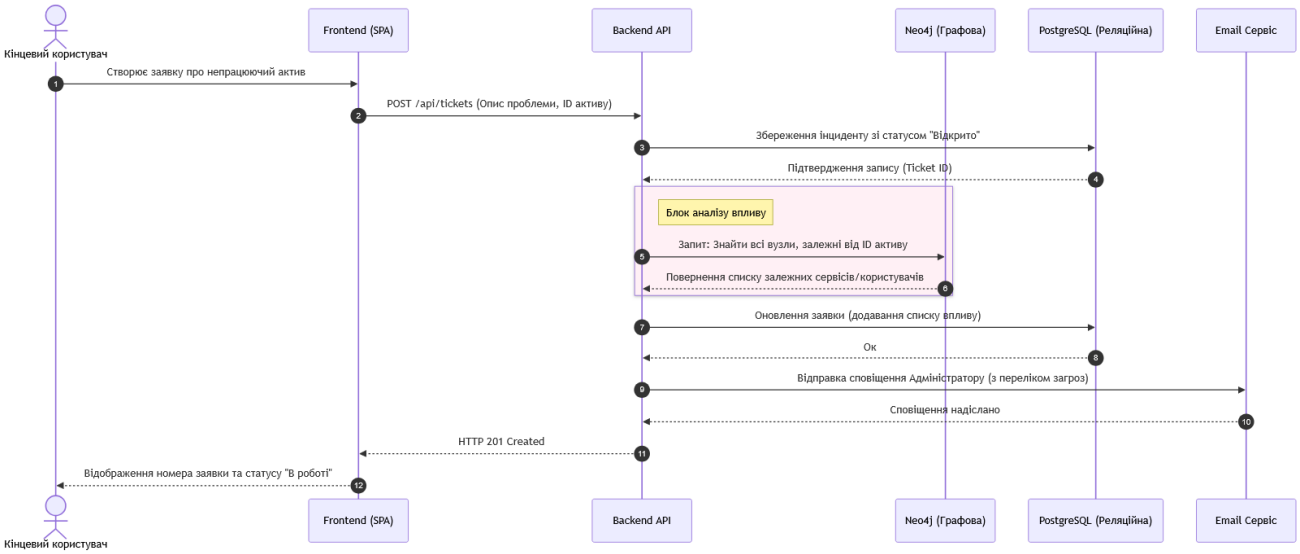

Рисунок 4.3. Сценарій обробки інциденту

Даний сценарій ініціюється кінцевим користувачем, який через портал служби технічної підтримки створює заявку щодо несправності певного обладнання. Система одразу реєструє цей інцидент у реляційній базі зі статусом відкритого. Ключовим етапом є звернення бекенду до графової бази для проведення аналізу впливу, де система автоматично визначає всі залежні сервіси, пристрої чи відділи, робота яких може бути порушена через цю

несправність. Отриманий список критичних залежностей додається до інформації про заявку, після чого ініціюється автоматична розсилка електронних сповіщень адміністраторам із деталізацією потенційних загроз інфраструктурі. Завершується процес тим, що клієнтський інтерфейс відображає користувачу номер його заявки та підтвердження її прийняття в роботу.

4.2. Розробка концептуальної та логічної моделей баз даних

Як було описано у попередньому пункті, особливістю архітектури проєкту є використання підходу збереження даних, що передбачає розділення даних між двома різними типами систем керування базами даних залежно від їхньої структури та призначення. Для зберігання транзакційних та довідкових даних використовується реляційна база, а для роботи з ієрархіями, мережевими топологіями та аналізу впливу графова. Відповідно, процес проєктування даних розділено на створення загальної концептуальної моделі та двох окремих логічних моделей.

Концептуальна модель описує предметну область на найвищому рівні абстракції, виділяючи ключові сутності системи та бізнес-зв'язки між ними, без прив'язки до конкретних технологій зберігання.

До базових сутностей системи належать «Користувач» (User), його «Роль» (Role), «ІТ-актив» (Asset), «Тип активу» (Asset Type), фізична «Локація» (розташування) та «Заявка» (Ticket) служби технічної підтримки. На концептуальному рівні визначено, що «Користувач» створює «Заявку», яка обов'язково стосується конкретного «Активу». Кожен «Актив» класифікується за певним «Типом» та закріплюється за визначеною «Локацією». Візуалізацію концептуальної моделі наведено на рис. 4.4.

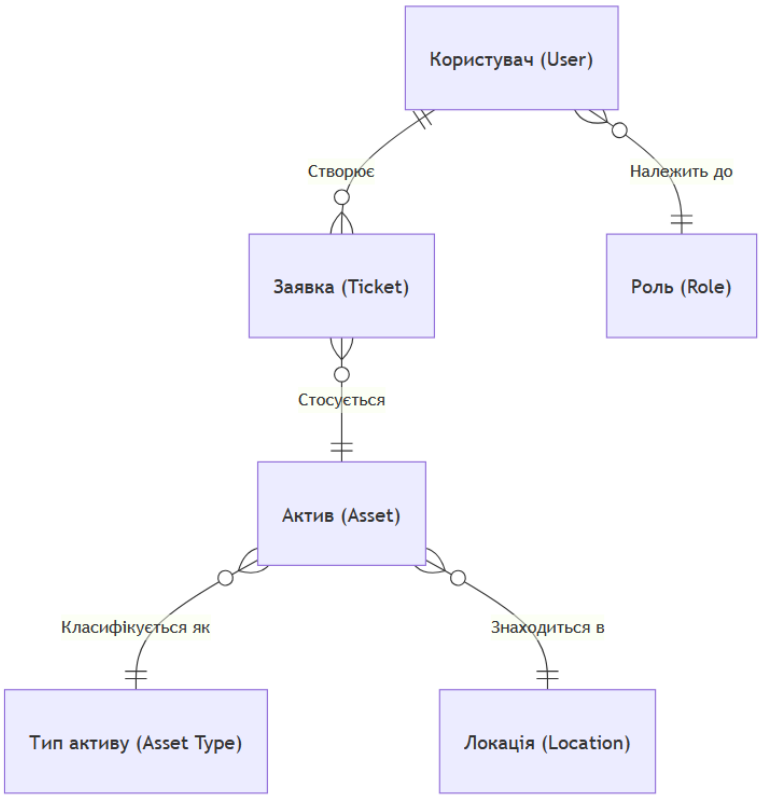

Рисунок 4.4. Концептуальна модель

Логічна модель для реляційної бази даних деталізує концептуальну структуру шляхом визначення конкретних атрибутів, первинних (PK) та зовнішніх (FK) ключів, необхідних для підтримки цілісності даних.

Ця частина системи зберігає текстові описи, фінансові показники активів, історію інцидентів та профілі користувачів. Структуру реляційних таблиць та зв'язків між ними наведено на діаграмі (рис. 4.5).

Оскільки реляційна модель погано підходить для швидкого обходу ієрархій та мережевих топологій, логічна модель для графічної бази проєктується у вигляді властивостей вузлів (Nodes) та спрямованих ребер (Relationships).

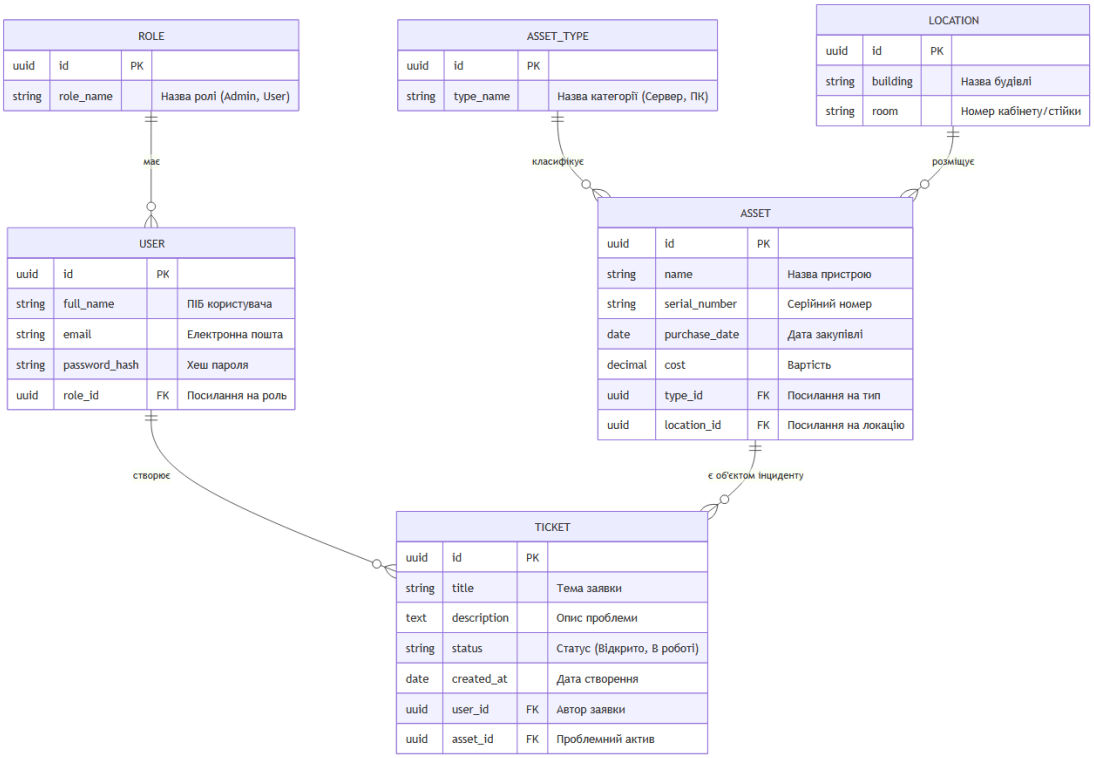

Рисунок 4.5. Логічна модель реляційної БД

У графовій базі ми не дублюємо всі текстові атрибути, а зберігаємо лише унікальний ідентифікатор (спільний з реляційною БД) та базовий тип для швидкого пошуку:

- *(:Asset {id: UUID, name: String, type: String})*
- *(:User {id: UUID, name: String})*
- *(:Location {id: UUID, name: String})*

До ключових типів зв'язків (Relationships) у системі належать:

- *CONNECTED_TO*: відображає фізичне або мережеве підключення між апаратними засобами;
- *DEPENDS_ON*: описує логічну залежність програмних сервісів від апаратної інфраструктури;
- *ASSIGNED_TO*: визначає персональну відповідальність користувача за конкретний актив.

Графова логічна модель, що ілюструє типи вузлів та зв'язків, представлена на рис. 4.6.

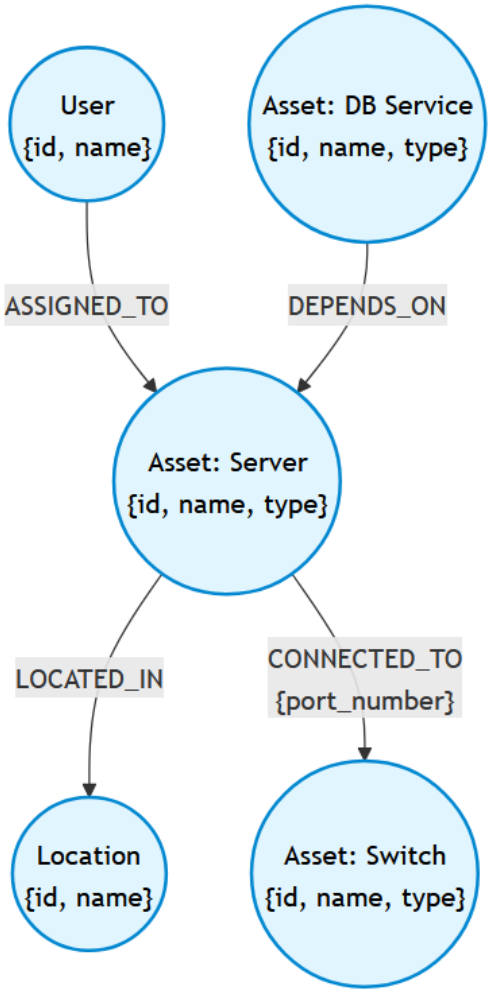

Рисунок 4.6. Логічна модель графової БД

4.3. Проєктування фізичної моделі бази даних

Фізична модель бази даних є завершальним етапом проєктування інформаційної структури. Вона перетворює логічну модель на конкретні об'єкти обраних систем керування базами даних (СКБД), враховуючи їхні архітектурні особливості, типи даних, обмеження цілісності та механізми індексування для забезпечення оптимальної швидкодії системи.

Нижче наведено детальний опис таблиць реляційної бази даних, яка є основним фізичним сховищем структурованої інформації в проєкті, а також схему зв'язків між ними.

Для забезпечення цілісності та унікальності записів у розподіленій системі, як основний тип для первинних ключів обрано UUID [28]. Це гарантує безпроблемну синхронізацію з іншими модулями системи.

Таблиця **Roles**:

- id (UUID, PK) – унікальний ідентифікатор;
- role_name (VARCHAR (50), NOT NULL) – назва ролі.

Таблиця **Users**:

- id (UUID, PK) – унікальний ідентифікатор;
- full_name (VARCHAR (150), NOT NULL) – повне ім’я користувача;
- email (VARCHAR (100), UNIQUE, NOT NULL) – електронна адреса (логін);
- password_hash (VARCHAR (255), NOT NULL) – захешований пароль;
- role_id (UUID, FK) – посилання на роль.

Таблиця **Asset_types:**

- id (UUID, PK) – унікальний ідентифікатор;
- type_name (VARCHAR (100), NOT NULL) – назва (напр. «Сервер»).

Таблиця **Locations**:

- id (UUID, PK) – унікальний ідентифікатор.
- building (VARCHAR (100), NOT NULL) – назва будівлі або корпусу.
- room (VARCHAR (50)) – номер кабінету, серверної або стійки.

Таблиця **Assets:**

- id (UUID, PK) – унікальний ідентифікатор;
- name (VARCHAR (150), NOT NULL) – мережеве або системне ім'я активу;
- serial_number (VARCHAR (100), UNIQUE, NOT NULL) – серійний номер виробника;
- purchase_date (DATE) – дата придбання;
- cost (DECIMAL (12,2)) – балансова вартість активу;
- type_id (UUID, FK) – посилання на тип активу;
- location_id (UUID, FK) – посилання на локацію;
- created_at (TIMESTAMP) – час додавання запису.

Таблиця **Tickets**:

- id (UUID, PK) – унікальний ідентифікатор заявки;
- title (VARCHAR (200), NOT NULL) – короткий опис проблеми;
- description (TEXT, NOT NULL) – детальний опис інциденту;
- status (VARCHAR (50), DEFAULT 'Open') – поточний стан;
- created_at (TIMESTAMP, DEFAULT CURRENT_TIMESTAMP) – дата та час створення;
- user_id (UUID, FK) – посилання на автора заявки;
- asset_id (UUID, FK) – посилання на актив, з яким пов’язана проблема.

Візуально фізична схема реляційної бази даних вебдодатку для управління IT-ресурсами компанії зображена на рисунку 4.7.

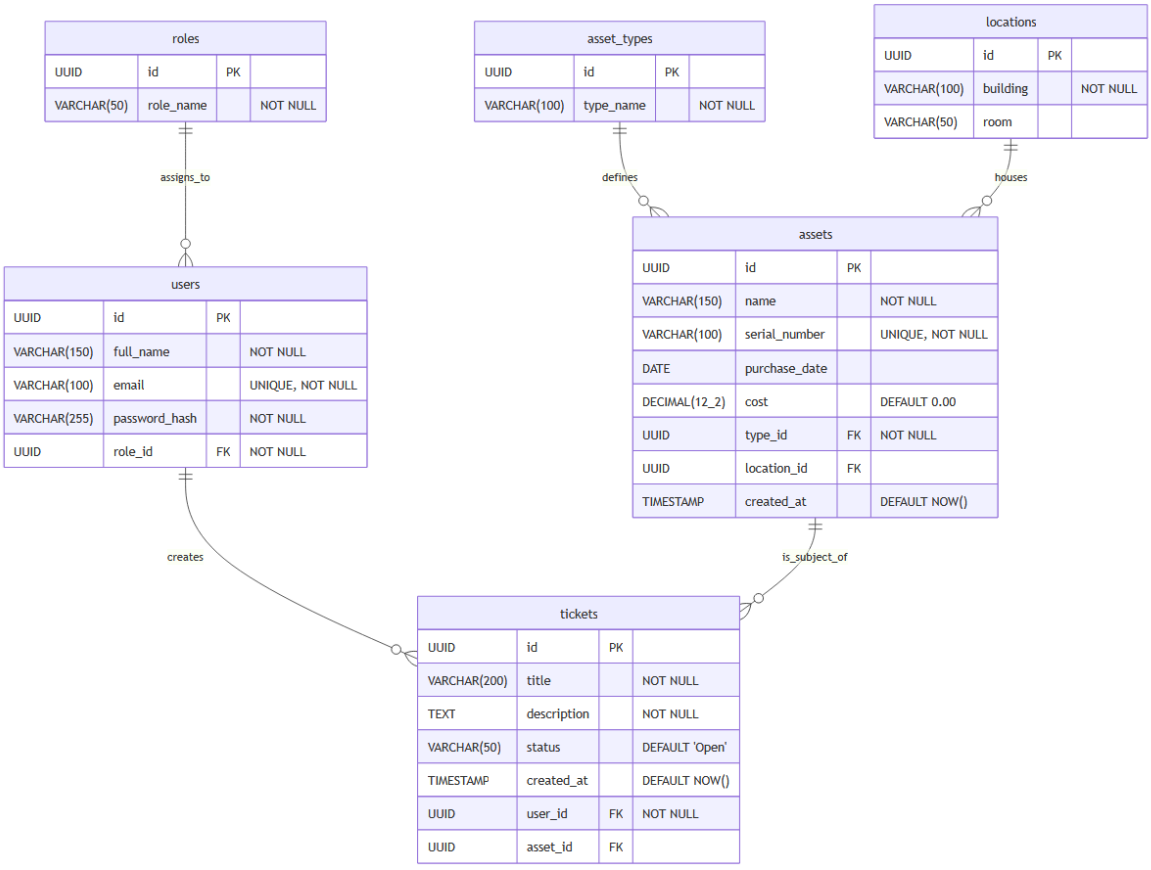

Рисунок 4.7. Схема фізичної БД

Готові таблиці можна запросто імпортувати у обрану СКБД та мати повністю готову та функціонально повну реляційну базу даних.

4.4. Аналіз сучасних трендів та вимог до UI/UX дизайну

Проєктування інтерфейсів для промислових та корпоративних систем кардинально відрізняється від споживчих застосунків. Такі платформи оперують у високостресових середовищах, де надійність, надзвичайно висока щільність даних та швидкість прийняття рішень є критичними факторами безперервності бізнесу. Погано спроєктований корпоративний інтерфейс знижує продуктивність, створює відчутні операційні ризики та безпосередньо призводить до фінансових втрат [29]. Відтак, ефективний Enterprise UI вимагає ретельного планування архітектури та масштабованості з самого початку, оскільки системи повинні підтримувати одночасну роботу тисяч співробітників і обробляти масиви даних (наприклад, сотні тисяч одиниць обладнання) без затримок у рендерингу сторінок [30].

Для вирішення проблем масштабованості на рівні фронтенду, сучасні корпоративні додатки відмовляються від завантаження масивних таблиць, натомість смарт-дизайн спирається на застосування віртуалізованих таблиць та лінивого завантаження, завдяки чому рендериться лише той контент, який безпосередньо потрапляє у видиму зону екрана користувача [30]. Стратегічне кешування часто використовуваних даних та можливості інтерактивної фільтрації без перезавантаження сторінки значно покращують досвід взаємодії (UX), дозволяючи аналітичним панелям залишатися високочутливими навіть під значним навантаженням. Зі зростанням системи традиційна ієрархічна навігація стає неефективною, тому на перший план виходить підхід «Search-first», або ж навігація через пошук із функцією випереджального введення, що дозволяє миттєво переходити до конкретних документів, клієнтських записів чи конфігураційних одиниць [30].

Основою архітектури складного корпоративного вебдодатку є концепція AppShell. Це уніфікований структурний шаблон, який забезпечує глобальну навігацію, контекст і візуальну консистентність на всіх сторінках системи. Навігація розподіляється на кілька ієрархічних рівнів [31]. Нульовий рівень це уніфікований заголовок навігації, який забезпечує глобальний досвід роботи з

платформою, консолідуючи елементи управління на рівні всього екземпляра системи: глобальний пошук, сповіщення, налаштування профілю та меню застосунків [31]. Це усуває розрізненість і значно підвищує ефективність перемикання між модулями.

На першому рівні розташовується первинна панель навігації, зазвичай у вигляді меню з іконками ліворуч або зверху, що є фундаментальною частиною оболонки робочого простору. На другому рівні система пропонує вибір структурних навігаційних патернів, серед яких домінують Tab-based (на основі вкладок) та Breadcrumb-based (на основі ієрархічних ланцюжків) моделі [31].

Навігація на основі вкладок оптимізована для забезпечення фокусу на багатозадачності, що є абсолютним імперативом для агентів служби підтримки (ITSM). Вона дозволяє спеціалісту одночасно тримати відкритими кілька сутностей у різних сесійних вкладках під головним заголовком, дозволяючи перемикатися між ними без втрати поточного контексту та без постійного перезавантаження сторінок [31]. З іншого боку, навігація на основі ієрархічних ланцюжків забезпечує фокус на лінійному потоці. Вона найкраще підходить для глибокого занурення в ієрархічні структури (що є типовим сценарієм під час дослідження дерева залежностей у CMDB), оскільки чітко показує користувачеві його поточне місце розташування в складній інформаційній архітектурі та дозволяє в один клік повернутися на макрорівень [31]. Для сегментації великого обсягу даних у межах одного запису застосовуються внутрішні вкладки записів, що ділять інформацію на логічні блоки та запобігають нескінченному прокручуванню екрана [31].

Критично важливим елементом інтерфейсу для складних застосунків є бічна панель [32]. У контексті запису ця панель виступає гнучким інструментом для відображення пов'язаного контенту або допоміжних засобів, економлячи цінний простір на екрані. Вона складається з контекстного меню (з іконками стану) та контейнера панелі, який може містити вкладки, пошукові поля та кнопки дій [33]. Дизайн таких панелей передбачає їх постійну присутність або тимчасовий характер залежно від сценарію.

4.5. Проєктування користувацького інтерфейсу та розробка візуальних прототипів

Розробка користувацького інтерфейсу базуватиметься на підходах, які були описані у попередньому підпункті та у додатку Figma [34]. Я сфокусуюсь на тих сторінках, які найкраще розкривають суть трьох модулів розроблюваного вебдодатка (ITAM, CMDB, ITSM).

Головна інформаційна панель (рис. 4.8) є центральним елементом управління вебдодатка, розробленим спеціально для IT-адміністраторів, керівників відділів технічної підтримки та менеджменту компанії. Основна мета цього екрана це забезпечити миттєвий зріз стану всієї IT-інфраструктури, дозволяючи фахівцям швидко оцінювати ситуацію та приймати рішення.

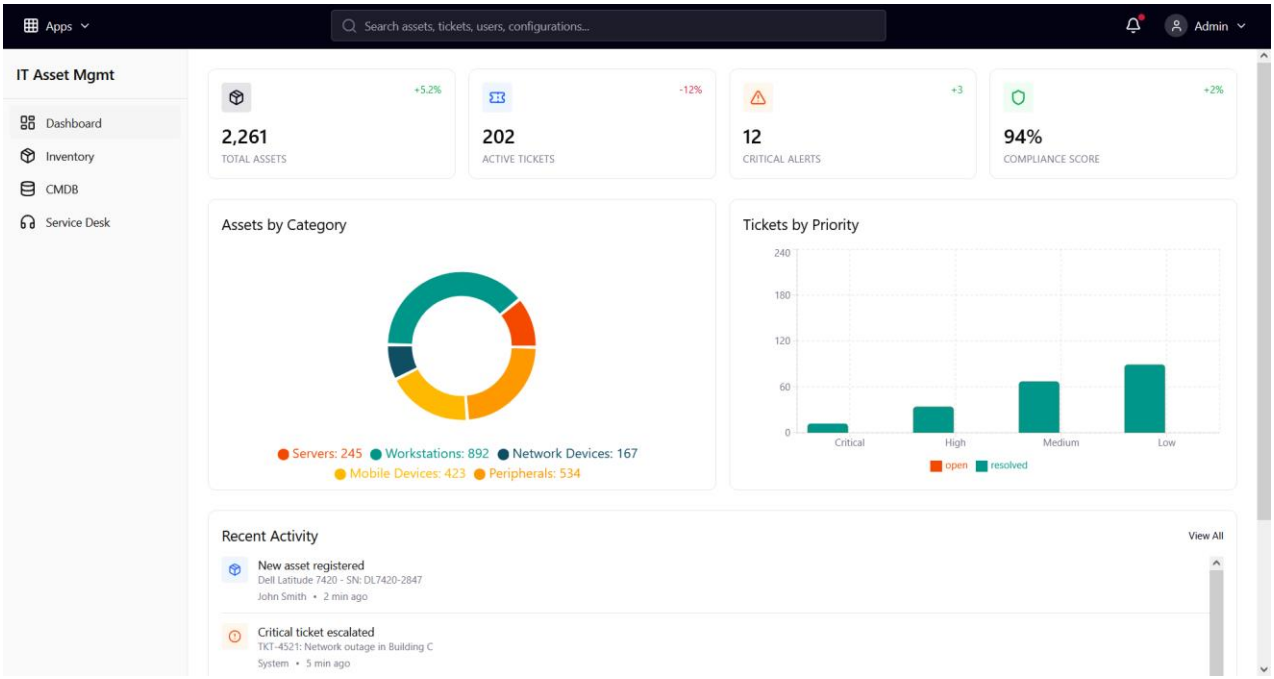

Рисунок 4.8. Інформаційна панель додатку

Візуальна архітектура екрана суворо дотримується сучасної концепції AppShell. Інтерфейс розділено на чіткі ієрархічні рівні, що забезпечують глобальний контекст. Нульовий рівень представлений темним глобальним заголовком, який контрастує з основним робочим простором. Центральне місце в ньому займає масивний рядок глобального пошуку, що реалізує парадигму «Search-first». Замість того, щоб змушувати користувача здійснювати множинні переходи через багаторівневі меню, система дозволяє

миттєво знайти потрібний сервер, заявку чи профіль співробітника за допомогою випереджального введення. Також на нульовому рівні консолідовані інструменти глобального сповіщення та налаштування профілю користувача. На першому рівні архітектури ліворуч розташована первинна панель навігації. Її спроєктовано з використанням мінімалістичних іконок та чіткої типографіки, що дозволяє швидко перемикатися між основними модулями системи (Inventory, CMDB, Service Desk), не перевантажуючи візуальний простір. Робоча область інформаційної панелі оптимізована для забезпечення високої щільності даних без втрати їхньої читабельності, що є критичним для корпоративних застосунків.

Екран «Реєстр IT-активів» (рис 4.9) є основним робочим простором для менеджерів з управління інфраструктурою, призначеним для наскрізного контролю життєвого циклу обладнання.

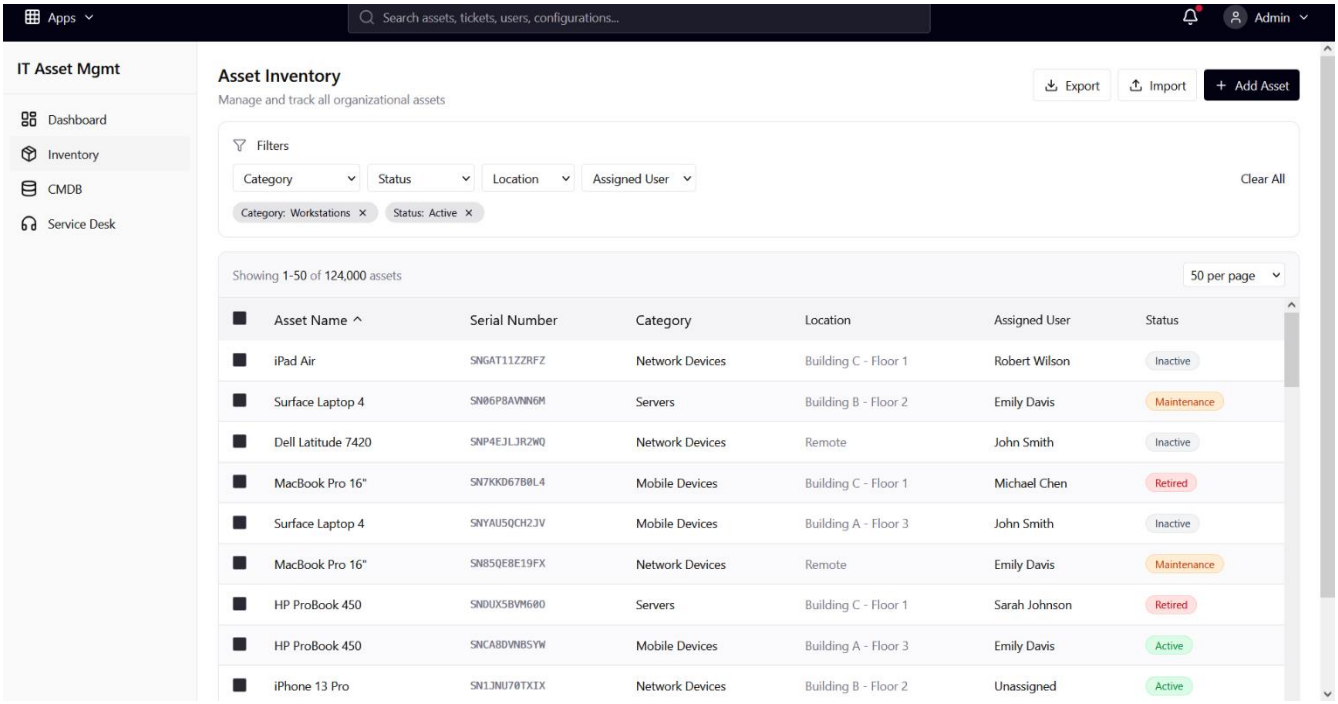

Рисунок 4.9. Реєстр IT-активів

Інтерфейс продовжує використовувати концепцію AppShell, зберігаючи єдиний глобальний заголовок та навігацію. Головним завданням при проєктуванні цього екрана було вирішення проблеми фронтенд-масштабованості. Тому основою екрана є віртуалізована таблиця даних із застосуванням лінивого завантаження. Браузер відмальовує лише ті 50 рядків,

які безпосередньо знаходяться у видимій зоні користувача, забезпечуючи миттєвий відгук інтерфейсу.

Для швидкого пошуку потрібних конфігураційних одиниць реалізовано панель інтерактивної фільтрації. Використання тегів-фільтрів дозволяє сегментувати дані без повного перезавантаження сторінки. Незважаючи на високу щільність інформації, таблиця залишається легкою для сприйняття завдяки використанню кольорових статусних маркерів, які допомагають спеціалісту фокусувати увагу на проблемних активах.

Екран деталей активу (рис. 4.10) демонструє детальну інформацію про конкретну конфігураційну одиницю і є класичним прикладом оптимізації простору в складних корпоративних застосунках.

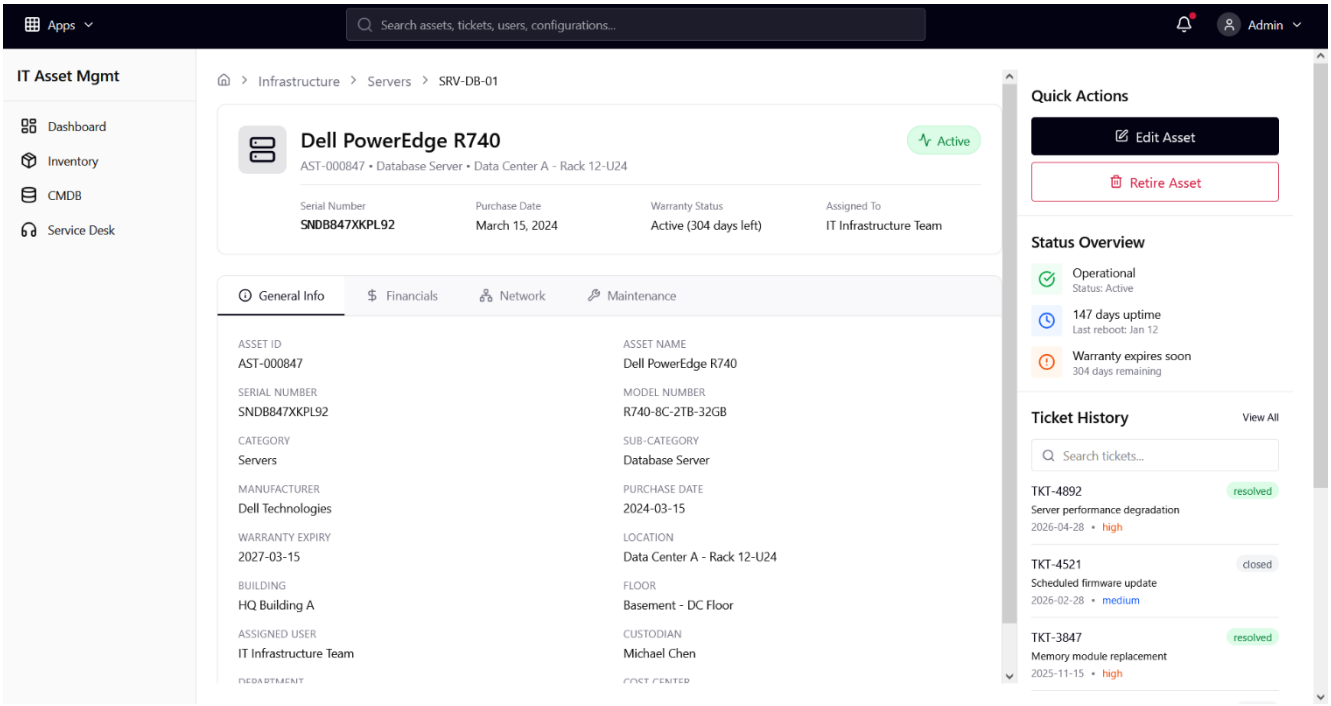

Рисунок 4.10. Екран деталей IT-активу

Ключовою навігаційною особливістю цього екрана є використання моделі на основі ієрархічних ланцюжків на другому рівні AppShell. Вона чітко показує користувачеві його поточне місцезнаходження в масиві даних і дозволяє в один клік повернутися на загальний рівень, що є критично важливим для збереження лінійного потоку при дослідженні інфраструктури.

Для вирішення проблеми нескінченного вертикального прокручування, яка часто виникає через надмірну кількість атрибутів активу, центральну

область екрана розбито на логічні блоки за допомогою внутрішніх вкладок записів.

Праворуч розташована контекстна бічна панель. Вона виступає як гнучкий допоміжний інструмент, що економить простір та консолідує найважливіші елементи керування: кнопки швидких дій (редагування, списання), поточні індикатори стану та історію пов'язаних ITSM-заявок. Такий підхід дозволяє адміністратору отримати повний контекст щодо пристрою та виконати необхідні операції без жодного переходу на інші сторінки системи.

Інтерактивна мапа інфраструктури (CMDB) є візуальним ядром (рис. 4.11) модуля CMDB, розробленим для глибокого дослідження дерева залежностей корпоративної IT-інфраструктури. Основна мета інтерфейсу дозволити інженерам миттєво зрозуміти топологію мережі та оцінити наслідки виходу з ладу окремих вузлів.

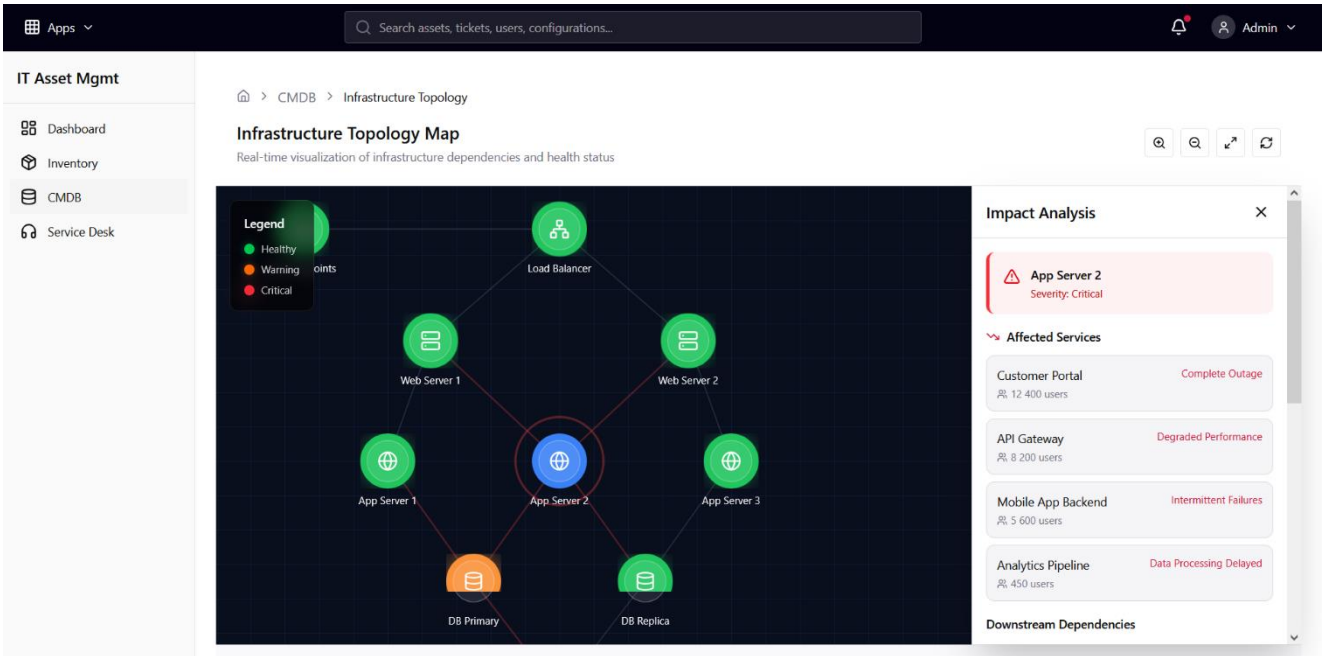

Рисунок 4.11. Інтерактивна мапа інфраструктури

Основою екрана є інтерактивне темне полотно, на якому візуалізуються вузли та зв'язки, що підтягуються безпосередньо з графової бази даних. Використання темного фону з технічною сіткою забезпечує максимальний контраст для кольорових індикаторів стану (зелений – стабільно, червоний – критичний збій), що дозволяє миттєво ідентифікувати проблемну зону.

Портал заявок служби технічної підтримки (рис 4.12) є основним робочим інструментом для агентів служби технічної підтримки та яскравим прикладом оптимізації корпоративного інтерфейсу для умов високої багатозадачності.

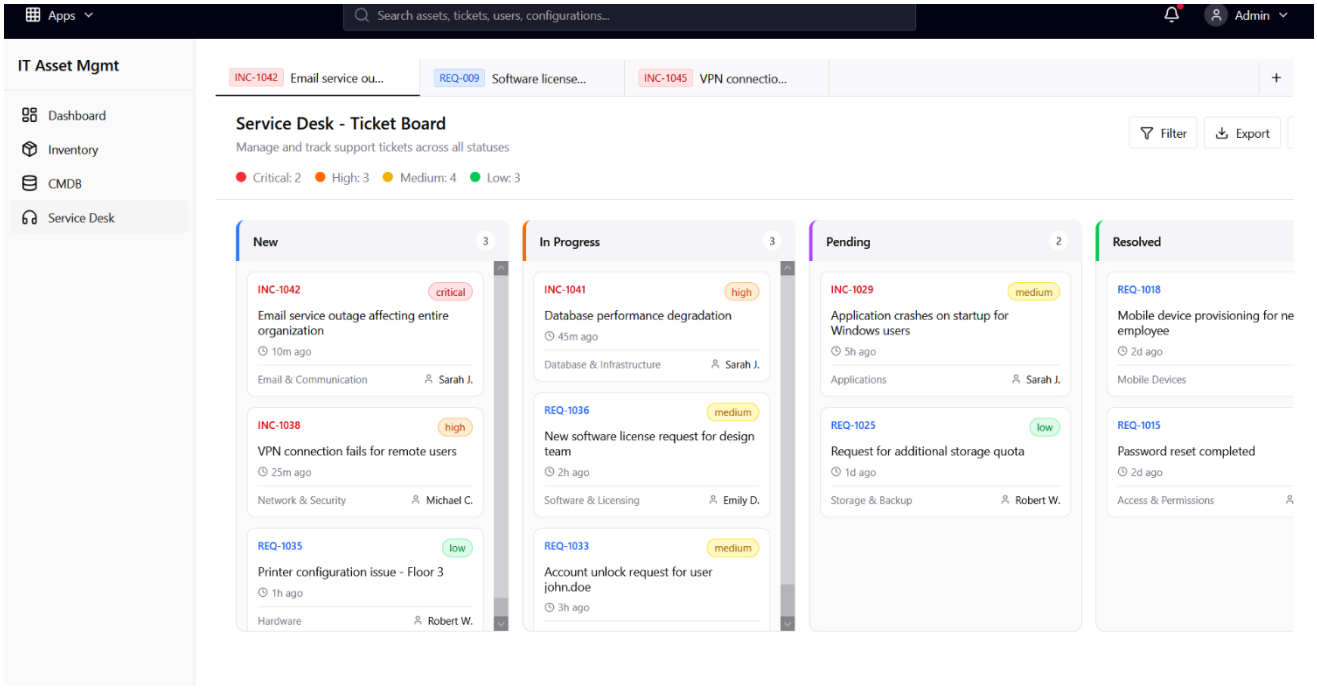

Рисунок 4.12. Портал заявок служби технічної підтримки

Ключовим архітектурним рішенням тут є впровадження навігації на основі вкладок на другому рівні оболонки AppShell. Як видно на макеті, під глобальним пошуком розташована панель активних сесій. Це дозволяє спеціалісту паралельно працювати з кількома складними інцидентами, перемикаючись між ними миттєво і без втрати поточного робочого контексту чи необхідності постійного перезавантаження сторінок.

Центральну частину екрана займає інтерактивна Kanban-дошка, що групує заявки за статусами виконання. Дизайн самих карток задач спроєктовано з урахуванням високої щільності даних, але без шкоди для читабельності.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи «Дослідження процесів управління проєктом розробки вебдодатку для управління IT-ресурсами компанії» було вирішено наступні завдання:

1. Проаналізовано сучасний стан та специфіку процесів управління IT-ресурсами підприємства. Виявлено критичні недоліки в існуючих методах обліку та моніторингу обладнання, що дозволило обґрунтувати доцільність розробки нового продукту. Встановлено, що розроблена система заповнює вільну нішу, пропонуючи унікальне безшовне поєднання модулів ITAM, CMDB та ITSM в єдиному консолідованому інтерфейсі;
2. Досліджено методологічні підходи до управління проєктами розробки корпоративних інформаційних систем. На основі проведеного аналізу обґрунтовано вибір «гнучких» (Agile) методологій, зокрема поєднання практик Scrum та Kanban, як найбільш оптимальної моделі для управління даним проєктом в умовах турбулентного зовнішнього середовища;
3. Сформовано технічні та функціональні вимоги до системи управління IT-ресурсами, а також проведено комплексний SWOT-аналіз із розробкою плану реагування на ризики. Визначено необхідність глибокої інтеграції модуля служби технічної підтримки (ITSM) з інфраструктурними даними для забезпечення автоматизованого виявлення уражених вузлів та оперативного реагування на збої;
4. Розроблено концептуальну, математичну та архітектурну моделі системи. Для реалізації платформи обрано багаторівневу клієнт-серверну архітектуру (N-Tier Architecture) з використанням контейнеризації Docker. Запропоновано інноваційну гібридну модель баз даних: реляційне сховище застосовано для

транзакційних сутностей, а графову базу даних – для відображення складної топології мережі та алгоритмічного обходу зв’язків;

5. Побудовано ієрархічну структуру робіт (WBS) та організаційну структуру команди з чітким розподілом повноважень за допомогою матриці RACI. Розроблено детальний календарний план розробки тривалістю 6 місяців (розбитий на двотижневі спринти) із застосуванням інструментів візуалізації навантаження, а також сформовано та обґрунтовано кошторис проєкту з урахуванням потреби у фінансових та людських ресурсах;
6. Спроєктовано користувацькі інтерфейси вебдодатка корпоративного рівня на базі шаблону AppShell. Створено інтерактивні дашборди для моніторингу технічного стану, інтегровано віртуалізовані таблиці для обробки великих масивів даних та впроваджено Tab-based навігацію для забезпечення зручної багатозадачної роботи фахівців служби підтримки.

Таким чином, кінцеві користувачі (системні адміністратори, менеджери IT-активів та фахівці технічної підтримки) отримують комплексний інструмент для ефективного обліку, моніторингу та обробки інцидентів. Всі встановлені цілі кваліфікаційної роботи магістра з проєктування та валідації такої платформи вважаються успішно досягнутими.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. What's the Difference Between ITAM vs CMDB | Ivanti. [Електронний ресурс]. Режим доступу: https://www.ivanti.com/blog/itam-vs-cmdb
2. Comparing CMDB vs Asset Management for Efficient IT Operations. [Електронний ресурс]. Режим доступу: https://www.device42.com/cmdb-best-practices/cmdb-vs-asset-management/
3. Optimizing IT asset management with ServiceNow: A data-driven approach to HAM and SAM. [Електронний ресурс]. Режим доступу: https://wjaets.com/sites/default/files/fulltext_pdf/WJAETS-2025-1209.pdf
4. ITSM In Small And Medium Enterprises - Meegle. [Електронний ресурс]. Режим доступу: https://www.meegle.com/en_us/topics/it-service/itsm-in-small-and-medium-enterprises
5. Improving IT Assets Management with ITIL 4 Framework. [Електронний ресурс]. Режим доступу: https://jiki.cs.ui.ac.id/index.php/jiki/article/view/1195/518
6. Best ITSM Software 2026: 10 Top Tools Compared | Flamingo. [Електронний ресурс]. Режим доступу: https://www.flamingo.run/blog/best-itsm-software
7. Compare ServiceNow IT Asset Management vs. Snipe-IT | G2. [Електронний ресурс]. Режим доступу: https://www.g2.com/compare/servicenow-it-asset-management-vs-snipe-it
8. Hidden Costs of Poor ServiceNow Adoption: Low ROI & Productivity Loss - Apty AI. [Електронний ресурс]. Режим доступу: https://apty.ai/blog/the-hidden-costs-of-poor-servicenow-adoption-wasted-software-spend-lost-productivity/
9. ManageEngine ServiceDesk Plus vs. Freshservice: Help Desk Leaders Face Off | PCMag. [Електронний ресурс]. Режим доступу: https://www.pcmag.com/comparisons/manageengine-service-desk-plus-vs-freshservice-help-desk-leaders-face-off

10. 10 Top Best ITSM Tools in 2026 | Xurrent. [Електронний ресурс]. Режим доступу: https://www.xurrent.com/blog/top-itsm-tools
11. Service/Help Desk solution - Manageengine vs Spiceworks vs OTRS? : r/sysadmin - Reddit. [Електронний ресурс]. Режим доступу: https://www.reddit.com/r/sysadmin/comments/1mr16cn/servicehelp_desk_solution_manageengine_vs/
12. Top 5 open source IT asset management software | Virima. [Електронний ресурс]. Режим доступу: https://virima.com/blog/top-5-open-source-it-asset-management-software/
13. Snipe-IT vs GLPI | Best Platform for IT Asset Management in 2026? - YouTube. [Електронний ресурс]. Режим доступу: https://www.youtube.com/watch?v=c02w_lSeITY
14. ManageEngine vs. Spiceworks: Ease of Use, Scalability, and Reviews. [Електронний ресурс]. Режим доступу: https://blog.invgate.com/manageengine-vs-spiceworks
15. Beyond the Waterfall: A Review of Project Management Methodologies in STEM - OAsis: UNLV's - University of Nevada, Las Vegas. [Електронний ресурс]. Режим доступу: https://oasis.library.unlv.edu/cgi/viewcontent.cgi?article=1000&context=eed_ug_research
16. Project management in IT: Agile, Scrum, Kanban, Waterfall - Serverspace.io. [Електронний ресурс]. Режим доступу: https://serverspace.io/about/blog/project-management-in-it/
17. A Complete Guide to Project Management Methodologies. [Електронний ресурс]. Режим доступу: https://www.freshworks.com/it-project-management/methodologies/
18. What is IT Asset Management (ITAM)? - ServiceNow. [Електронний ресурс]. Режим доступу: https://www.servicenow.com/products/it-asset-management/what-is-itam.html

19. A systematic literature review of the agile methodology applied during construction project design - University of Johannesburg. [Електронний ресурс]. Режим доступу: https://ujcontent.uj.ac.za/esploro/outputs/graduate/A-systematic-literature-review-of-the/9913444507691
20. RACI: як створити матрицю відповідальності. [Електронний ресурс]. Режим доступу: https://hurma.work/blog/matriczya-vidpovidalnosti-raci/
21. ClickUp: The Everything App for Work. [Електронний ресурс]. Режим доступу: https://clickup.com
22. Understanding N-Tier Architecture: Building Robust and Scalable Applications. [Електронний ресурс]. Режим доступу: https://medium.com/@segekaratas/understanding-n-tier-architecture-building-robust-and-scalable-applications-62db30a40b5
23. Як працює SPA (Single Page Application) – усе, що потрібно знати. [Електронний ресурс]. Режим доступу: https://dou.ua/forums/topic/50894/
24. What is REST API? [Електронний ресурс]. Режим доступу: https://www.ibm.com/think/topics/rest-apis
25. PostgreSQL: The World's Most Advanced Open Source Relational Database. [Електронний ресурс]. Режим доступу: https://www.postgresql.org/
26. Neo4j Graph Database & Analytics. [Електронний ресурс]. Режим доступу: https://neo4j.com
27. What is continuous integration/continuous delivery (CI/CD)? [Електронний ресурс]. Режим доступу: https://www.ibm.com/think/topics/ci-cd-pipeline
28. Розбираємо UUID у всьому його різноманітті. [Електронний ресурс]. Режим доступу: https://dou.ua/forums/topic/52305/
29. Strategic UI/UX Design for Industrial and Enterprise Systems. [Електронний ресурс]. Режим доступу: https://medium.com/@mullztech/strategic-ui-ux-design-for-industrial-and-enterprise-systems-5e41a84b16c5

30. Enterprise UI Design in 2026: Principles, Trends & Best Practices. [Електронний ресурс]. Режим доступу: https://hashbyt.com/blog/enterprise-ui-design
31. Navigation | Horizon Design System. [Електронний ресурс]. Режим доступу:https://horizon.servicenow.com/workspace/patterns/navigation/navigation-pattern
32. Side Drawer UI: A Guide to Smarter Navigation - Design Monks. [Електронний ресурс]. Режим доступу: https://www.designmonks.co/blog/side-drawer-ui
33. Contextual side bar pattern - Horizon Design System - ServiceNow. [Електронний ресурс]. Режим доступу: https://horizon.servicenow.com/workspace/patterns/contextual-side-bar/contextual-side-bar-patterns
34. Figma: The Collaborative Interface Design Tool. [Електронний ресурс]. Режим доступу: https://www.figma.com/
35. Command Palette UI Design - Mobbin. [Електронний ресурс]. Режим доступу: https://mobbin.com/glossary/command-palette
36. 4 B2B SaaS Color Palettes That Stand Out in 2026 | Tentackles. [Електронний ресурс]. Режим доступу: https://tentackles.com/blog/b2b-saas-color-palettes-2026-that-stand-out
37. Kanban, Hierarchy, List, Timeline: task board views - Holaspirit Help Center. [Електронний ресурс]. Режим доступу: https://help.holaspirit.com/en/article/kanban-hierarchy-list-timeline-task-board-views-27u9tw/
38. Kanban vs Scrum in Agile Project Management - University of Phoenix. [Електронний ресурс]. Режим доступу: https://www.phoenix.edu/articles/business/kanban-vs-scrum-in-agile-project-management.html
39. Agile-hybrid delivery approaches for complex design and engineering projects: an integrated case study - Emerald Publishing. [Електронний

ресурс]. Режим доступу: https://www.emerald.com/sasbe/article/doi/10.1108/SASBE-04-2024-0125/1272279/Agile-hybrid-delivery-approaches-for-complex

40. Ivanti vs Spiceworks 2026 | Gartner Peer Insights. [Електронний ресурс]. Режим доступу: https://www.gartner.com/reviews/market/it-infrastructure-and-iot/compare/ivanti-vs-spiceworks
41. Best IT Service Management Tools: User Reviews from April 2026 - G2. [Електронний ресурс]. Режим доступу: https://www.g2.com/categories/it-service-management-itsm-tools
42. Integrating ITAM with ITSM for Improved Operational Efficiency. [Електронний ресурс]. Режим доступу: https://www.lansweeper.com/blog/itam/integrating-itam-with-itsm-for-improved-operational-efficiency/
43. Тімінський О.Г., Коломієць А.С. Управління ризиками та можливостями проєкту [Текст]: методичні вказівки до виконання практичних, лабораторних робіт та самостійної роботи для студентів освітньої програми «Управління проєктами» спеціальності 122 «Комп'ютерні науки» для денної і заочної форм навчання. – К. : КНУ імені Тараса Шевченка, 2021, 40 с.

ДОДАТОК А

Функціональні вимоги

Таблиця А.1

Ідентифікатор	Назва	Опис	Пріорітет
1	**2**	**3**	**4**
FR-1	Управління життєвим циклом IT-ресурів (ITAM)	Вебдодаток має надавати функціонал створення, редагування, архівування та видалення записів про апаратні та програмні активи компанії	Високий
FR-2	Візуалізація взємозв’язків між складовими IT-інфраструктури (CMDB)	Реалізація інтерактивної мапи взаємозв'язків між серверами, мережевим обладнанням та бізнес-сервісами на основі внесених даних	Високий
FR-3	Інтеграція із службою технічної підтримки	Прив'язка заявок служби підтримки до конкретних активів із бази даних	Високий
FR-4	Управління ліцензіями ПЗ	Автоматичне відстеження термінів дії програмних ліцензій зі сповіщенням за 14 та днів до закінчення	Високий
FR-5	Рольова модель доступу	Підтримка розподілу прав доступу за ролями	Середній

Завершення табл. А.1

1	2	3	4
FR-6	Портал самообслуговування	Реалізація інтерфейсу для кінцевих користувачів, де вони можуть переглянути закріплену за ними техніку та створити заявку на її заміну чи ремонт	Високий
FR-7	Аналітичні звіти	Формування фінансових та операційних звітів із можливістю експорту	Низький
FR-8	Дашборди для керівництва	Реалізація інтерактивних панелей, що відображають ключові метрики IT-інфраструктури в режимі реального часу	Середній

Нефункціональні вимоги

Таблиця А.2

Ідентифікатор	Назва	Опис	Пріорітет
1	**2**	**3**	**4**
NFR-1	Продуктивність	Час відгуку системи на стандартні запити користувача не більше 2 секунд	Високий

Завершення табл. А.2

1	2	3	4
NFR-2	Безпека даних	Шифрування конфіденційних даних, підтримка двофакторної автентифікації	Високий
NFR-3	Зручність використання	Інтерфейс має бути інтуїтивно зрозумілим та адаптивним, розробленим відповідно до сучасних дизайн-систем	Високий
NFR-4	Масштабованість	База даних та архітектура додатка мають стабільно обробляти більше 10 000 записів IT-активів без зниження продуктивності	Високий
NFR-5	Надійність та доступність	Гарантований час безперебійної роботи повинен складати не менше 97% на рік	Високий
NFR-6	Резервне копіювання	Система повинна здійснювати регулярне автоматичне резервне копіювання бази даних	Середній
NFR-7	Сумісність	Вебдодаток повинен коректно працювати у всіх сучасних браузерах	Середній

ДОДАТОК Б

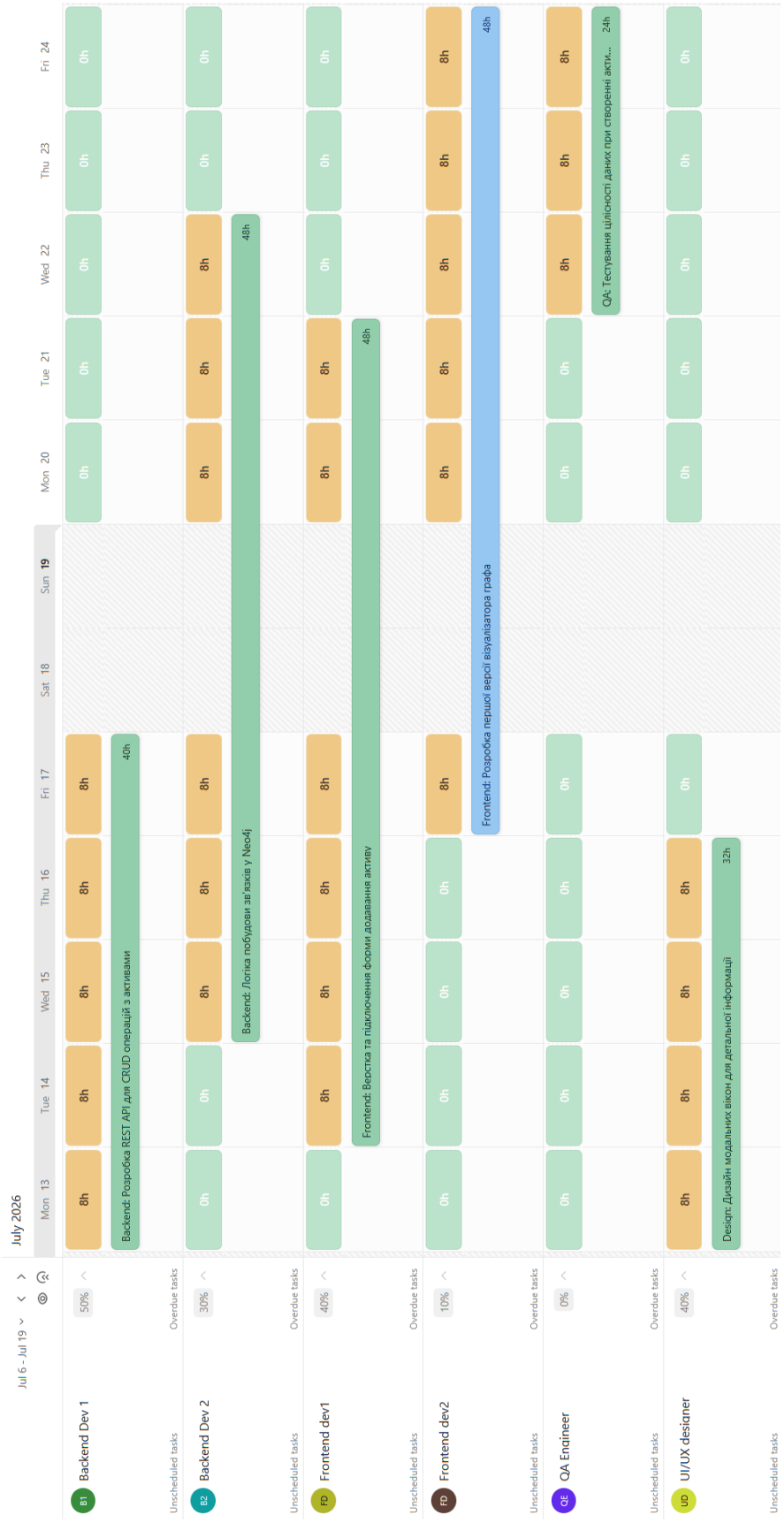

Рисунок Б.1 Графік розподілу робочого навантаження команди під час виконання спринту №4

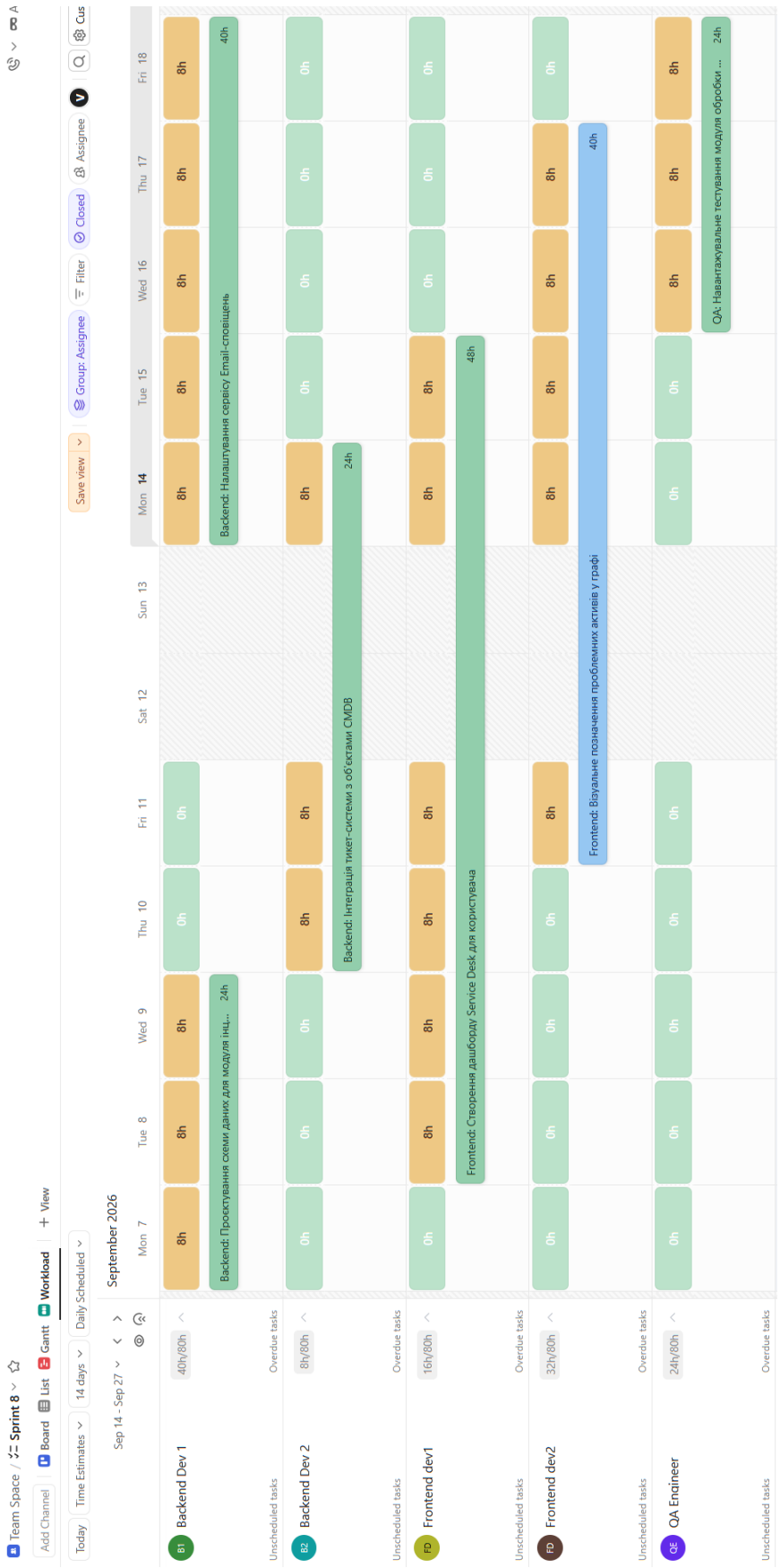

Рисунок Б.2 Графік розподілу робочого навантаження команди під час виконання спринту №8