

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук _____
(повна назва)

Кафедра _____ Інформаційних управляючих систем _____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____ другий (магістерський) _____
Дослідження методів та моделей управління ІТ проєктами
(тема)

Виконав:
Здобувач 2 курсу, групи УПГІТМ-24-1
_____ Кирило МЕДОЛАЗОВ _____
(прізвище, ім'я, по батькові)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
Освітня програма Управління
проєктами в галузі інформаційних
технологій
(повна назва освітньої програми)

Керівник
проф. каф. ІУС Віктор ЛЕВИКІН
(Власне ім'я ПРІЗВИЩЕ)

Допускається до захисту
Зав. кафедри


(підпис)

Костянтин ПЕТРОВ
(Власне ім'я ПРІЗВИЩЕ)

2026 р.

Харківський національний університет радіоелектроніки

Факультет _____ Комп'ютерних наук _____
Кафедра _____ Інформаційних управляючих систем _____
Рівень вищої освіти _____ другий (магістерський) _____
Спеціальність _____ 122 Комп'ютерні науки _____
(код і повна назва)
Тип програми _____ освітньо-наукова _____
(освітньо-професійна або освітньо-наукова)
Освітня програма _____ Управління проектами в галузі інформаційних технологій _____
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«___» _____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачу _____ Медолазову Кирилу Олеговичу _____
(прізвище, ім'я та по батькові)

1. Тема роботи Дослідження методів та моделей управління ІТ проектами
затверджена наказом університету від 23 березня 2026 р. № 231Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20 травня 2026 р.
3. Вихідні дані до роботи: Аналіз концептуальних засад, Проблеми моніторингу та існуючих методів управління ІТ-проектами (Scrum, Agile, Waterfall, Kanban) у динамічному середовищі. Результати дослідження та математичної оцінки ризиків існуючих комбінованих методологій (Water-Scrum-Fall, Waterfall-Agile) за визначеними критеріями ефективності. Інноваційні принципи інтеграції інтелектуальних агентів для автоматичної декомпозиції задач та оцінки трудомісткості. Вимоги до клієнт-серверної архітектури програмної системи на базі Node.js та React, а також алгоритмів розрахунку каскадного зміщення етапів. Результати тестування розробленої гібридної системи управління проектами та оцінка її результативності.
4. Перелік питань, що потрібно опрацювати в роботі: аналіз специфіки сучасних ІТ-проектів; дослідження проблем контролю проєктів у динамічному середовищі; формування системи критеріїв оцінки ефективності методів управління; проведення математичного аналізу ризиків комбінованих методологій та відбір оптимальних структурних елементів; проєктування архітектури комбінованого методу управління з інтеграцією каскадного макропланування, ітеративних спринтів та візуалізації потоку; розробка механізмів взаємодії з ІІІ-агентами для автоматизованої декомпозиції завдань; практична програмна реалізація системи управління (Node.js, React) з алгоритмами каскадного зміщення етапів та інтерактивними дошками Kanban із WIP-лімітами; тестування та оцінка ефективності розробленої програмної системи..

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз існуючих підходів та методів управління проектами	06.04.2026 – 05.04.2026	виконано
2	Вивчена монолітних методологій та парадигм управління та їх характеристик	25.03.2026 – 05.04.2026	виконано
3	Огляд підходів та їх переваг і недоліків	01.04.2026 – 04.04.2026	виконано
4	Дослідження гібридних методів	04.04.2026 – 09.04.2026	виконано
5	Структурне формування універсального гібридного методу спеціалізованому для зручності навчання	11.04.2026 – 12.04.2026	виконано
6	Розробка демонстраційної програмної системи	13.04.2026 – 14.05.2026	виконано
7	Оформлення пояснювальної записки	13.04.2026 – 23.04.2026	виконано
8	Підготовка презентації	24.04.2026 – 26.04.2026	виконано
9	Надання роботи для перевірки на плагіат	15.05.2026 – 17.05.2026	виконано
10	Попередній захист	18.05.2026 – 19.05.2026	виконано
11	Надання роботи на рецензію	18.05.2026 – 19.05.2026	виконано
12	Захист	21.05.2026	виконано

Дата видачі завдання: 06.04.26

Здобувач _____

(підпис)

Керівник роботи _____

(підпис)

проф. каф. ІУС Віктор ЛЕВИКІН

(посада, власне ім'я, ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи містить: 93 с., 24 рис., 8 табл., 31 джерел, 2 додатки.

ГІБРИДНИЙ МЕНЕДЖМЕНТ, ІТ ПРОЄКТИ, КАЛЕНДАРНИЙ ПЛАН, КРИТЕРІЙ ЕФЕКТИВНОСТІ ПРОЄКТУ, МЕТОДОЛОГІЯ, МОНІТОРИНГ, ПЛАНУВАННЯ, ТРИВАЛІСТЬ РОБІТ.

Об'єктом даної кваліфікаційної роботи є процеси та методи управління ІТ-проектами в умовах динамічного технологічного середовища.

Предметом дослідження є розробка комбінованого методу управління проектами, що поєднує принципи каскадного та гнучкого планування, а також його подальша програмна імплементація з інтеграцією інструментів штучного інтелекту.

Завданням кваліфікаційної роботи є удосконалення ефективності гібридних методів управління ІТ-проектами в системах дистанційного навчання на основі інтеграції каскадних (Waterfall) та гнучких (Agile, Kanban) підходів.

Після вивчення існуючих методів управління ІТ проектами була розроблена адаптована програмна система клієнт-серверної архітектури (Node.js, React), яка візуалізує робочі процеси через інтерактивні дошки, автоматизує алгоритми каскадного зміщення етапів та містить інтеграцію з API штучного інтелекту для декомпозиції і оцінки задач.

ABSTRACT

The explanatory note to the qualification work contains: 93 pages, 24 figures, 8 tables, 31 sources, 2 appendices.

HYBRID MANAGEMENT, IT PROJECTS, CALENDAR PLAN, PROJECT EFFICIENCY CRITERIA, METHODOLOGY, MONITORING, PLANNING, WORK DURATION.

The object of this qualification work is the processes and methods of managing IT projects in a dynamic technological environment.

The subject of the research is the development of a combined project management method that combines the principles of cascade and flexible planning, as well as its further software implementation with the integration of artificial intelligence tools.

The task of the qualification work is to improve the efficiency of hybrid IT project management methods in distance learning systems based on the integration of cascade (Waterfall) and flexible (Agile, Kanban) approaches.

After studying existing IT project management methods, an adapted client-server architecture software system (Node.js, React) was developed, which visualizes workflows through interactive boards, automates cascade stage displacement algorithms and contains integration with the artificial intelligence API for task decomposition and evaluation.

ЗМІСТ

Скорочення та умовні позначки	8
Вступ	9
1 Аналіз предметної області та існуючих методів управління ІТ-проєктами.....	12
1.1 Концептуальні засади та специфіка сучасних ІТ-проєктів	12
1.2 Аналіз проблем контролю та моніторингу проєктів у середовищі.....	14
1.3 Порівняльна характеристика інструментальних засобів та методологій (Scrum, Agile, Waterfall, Kanban)	17
1.4 Обґрунтування вибору гібридних підходів для підвищення ефективності управління.....	22
1.5 Постановка задачі дослідження в межах кваліфікаційної роботи	24
2 Дослідження та оцінка ефективності методів комбінованого управління.....	26
2.1 Критерії оцінки ефективності методів управління	26
2.2 Аналіз ризиків існуючих комбінованих методологій (Water-Scrum-Fall, Waterfall-Agile)	29
2.3 Фільтрація та відбір оптимальних структурних елементів управління.....	34
3 Розробка та проєктування комбінованого методу.....	38
3.1 Поєднання моделі Waterfall та методу Kanban	38
3.2 Інноваційна імплементація інтелектуальних агентів (ШІ) для декомпозиції та оцінки задач.....	41
3.3 Опис реалізації та структурної взаємодії елементів розробленого методу.....	42
4 Практична реалізація та тестування гібридного методу	49
4.1 Вибір технологічного стеку та розробка архітектури клієнт-серверного рішення (Node.js, React).....	49
4.2 Програмна реалізація алгоритмів розрахунку тривалості етапів та каскадного	

	7
зміщення (Waterfall)	52
4.3 Наповнення та застосування технологій Kanban та інтеграція API штучного інтелекту (Gemini)	55
4.4 Оцінка результатів впровадження розробленого додатка.....	60
Висновки.....	66
Перелік джерел посилання.....	69
Додаток А Лістинг коду	74
Додаток Б Графічний матеріал кваліфікаційної роботи.....	83

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

IT – Інформаційні технології

K1–K5 – Критерії оцінки ефективності методів управління (K1 - зв'язок з командою, K2 - стабільність завдань, K3 - відсутність перевантажень, K4 - швидкий фідбек, K5 - низький поріг входу)

III – Штучний інтелект

API – Програмний інтерфейс застосунку (Application Programming Interface)

ERP – Корпоративні системи управління ресурсами підприємства (Enterprise Resource Planning)

HMR – «Гаряче» перезавантаження модулів (Hot Module Replacement)

HTTP – Протокол передачі гіпертексту (HyperText Transfer Protocol)

JSON – Текстовий формат обміну даними (JavaScript Object Notation)

KPI – Ключові показники ефективності (Key Performance Indicators)

LMS – Системи дистанційного навчання (Learning Management System)

PERT – Метод оцінки та огляду програм (Program Evaluation Review Technique)

PM – Управління проєктами / проєктний менеджмент (Project Management)

RESTful – Архітектурний стиль створення програмних інтерфейсів (Representational State Transfer)

SPA – Односторінковий застосунок (Single Page Application)

UI – Інтерфейс користувача / дизайн (User Interface)

WBS – Ієрархічна структура робіт (Work Breakdown Structure)

WIP – Незавершена робота / обмеження кількості одночасних завдань у процесі (Work In Progress)

ВСТУП

У сучасному світі управління проєктами відіграє ключову роль у досягненні цілей організацій та забезпеченні їхнього стратегічного успіху на глобальному ринку. ІТ-проєкти у своїй загальній характеристиці являють собою складний продукт, що є результатом розробки за чітко визначеними правилами, впровадження або підтримки програмного забезпечення та інформаційних систем. З огляду на швидкі та безперервні зміни технологічного ландшафту, а також постійно зростаючі вимоги замовників, наявні методології управління проєктами часто зіштовхуються з серйозними проблемами щодо забезпечення належної ефективності та гнучкості. Процес управління такими проєктами вимагає скрупульозного планування, виконання та постійного контролю діяльності з метою досягнення встановлених цілей у межах жорстких обмежень стосовно бюджету, часу та якості фінального результату. Оскільки технології розвиваються надзвичайно стрімко, проєктні команди повинні демонструвати не лише технічну компетентність, але й високу здатність до оперативної адаптації.

Попри наявність широкого спектра інноваційних інструментальних рішень, управління сучасними ІТ-проєктами супроводжується низкою критичних проблем, що ускладнюють процес моніторингу та негативно впливають на результативність розробки. Традиційні підходи, як-от каскадна модель (Waterfall), хоча й забезпечують чітку структуру, детальне документування та високу передбачуваність процесів, часто виявляються занадто негнучкими для адаптації до швидких змін в умовах динамічного середовища. Будь-які коригування вже реалізованої частини проєкту в таких моделях вимагають повернення до початкових фаз, що суттєво підвищує вартість та затягує строки виконання. З іншого боку, гнучкі методології, такі як Agile та Scrum, хоч і пропонують необхідну адаптивність та орієнтованість на швидкий зворотний зв'язок, іноді можуть призвести до втрати контролю над

довготривалим плануванням та загальним обсягом робіт. Постійна волатильність запитів замовника часто призводить до неконтрольованого розширення меж проєкту, що ускладнює прогнозування фінального бюджету.

Окрему увагу варто приділити проблематиці залучення та навчання молодих спеціалістів у галузі проєктного менеджменту. Для початківців існуючі комплексні системи контролю часто створюють надзвичайно високий поріг входу через заплутаність механізмів керування, надмірне бюрократичне навантаження та відсутність безпечного простору для здійснення та виправлення помилок. Таким чином, існує очевидна та гостра потреба у розробці комплексного підходу до управління проєктами, який гармонійно поєднував би структурованість традиційних методів із гнучкістю сучасних практик. Розробка такого комбінованого методу дозволить командам ефективно адаптуватися до змін, водночас забезпечуючи високий рівень продуктивності, стабільність виконання завдань та повне задоволення вимог стейкхолдерів.

Актуальність даного дослідження додатково підсилюється необхідністю інтеграції новітніх технологій штучного інтелекту в процеси проєктного менеджменту. Сучасні тенденції вимагають створення інтелектуальних систем управління, які здатні не лише візуалізувати робочі процеси через інтерактивні дошки, але й автоматизувати аналітичну роботу менеджера. Впровадження інструментів на базі генеративного штучного інтелекту дозволяє автоматично оцінювати декомпозицію етапів проєкту та генерувати логічні підзадачі з визначенням ролей і витрат часу, що значно оптимізує етап планування та знижує ризик людської помилки.

Об'єктом дослідження в межах даної кваліфікаційної роботи виступають процеси та методи управління ІТ-проєктами в умовах динамічного технологічного середовища. Предметом дослідження є розробка комбінованого методу управління проєктами, що поєднує принципи каскадного та гнучкого планування, а також його подальша програмна імплементація з інтеграцією інструментів штучного інтелекту.

Головна мета роботи полягає в детальному дослідженні, науковому обґрунтуванні та практичній розробці комбінованого методу управління ІТ-проектами, який би максимально ефективно поєднував прогнозованість моделі Waterfall із візуалізацією Kanban та адаптивністю Agile, а також у створенні відповідної програмної системи для автоматизації цих процесів. Для досягнення поставленої мети було сформовано комплекс завдань, що передбачають проведення порівняльного аналізу сучасних методологій, вивчення недоліків існуючих систем контролю, проектування архітектури гібридного рішення та безпосередню розробку програмного продукту, що включає клієнтську та серверну частини з реалізованим алгоритмом каскадного зміщення етапів і автоматичною генерацією задач.

Наукова новизна одержаних результатів полягає у вдосконаленні методів проектного управління шляхом структурного поєднання класичних алгоритмів лінійного планування з ітеративними спринтами, що доповнюється інноваційною інтеграцією великих мовних моделей для безперервного аналізу та декомпозиції беклогу завдань. Практичне значення роботи полягає у створенні повноцінно функціонуючої програмної системи, розробленої з використанням сучасного технологічного стеку (Node.js, React), яка може бути використана як у реальних комерційних ІТ-проектах для підвищення прозорості процесів, так і в освітніх цілях для комфортної адаптації та навчання молодих фахівців.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ МЕТОДІВ УПРАВЛІННЯ ІТ-ПРОЄКТАМИ

1.1 Концептуальні засади та специфіка сучасних ІТ-проєктів

Сучасні проєкти в галузі інформаційних технологій у загальній характеристиці являють собою складний продукт, що є результатом розробки за чітко визначеними правилами та вимогами, впровадження або підтримки програмного забезпечення, апаратних засобів та мережевих систем. Спектр діяльності в межах таких проєктів є надзвичайно широким: від створення вебсайтів та розробки мобільних додатків до впровадження комплексних корпоративних систем управління ресурсами підприємства (ERP) чи спеціалізованих систем електронного навчання.

Характерною особливістю ІТ-проєктів є високий рівень терміновості та вимоги до точності виконання завдань, що супроводжується постійною зміною технологічного ландшафту та вимог замовників. Процес управління такими проєктами передбачає планування, виконання та контроль діяльності з метою досягнення встановлених цілей у межах жорстких обмежень щодо бюджету, часу та якості результату. Оскільки технології безперервно розвиваються, проєктні команди повинні демонструвати високу гнучкість та здатність до швидкої адаптації.

Центральне місце в системах управління посідає менеджмент людських ресурсів, оскільки персонал є ключовим активом, чії навички, досвід та мотивація безпосередньо визначають успішність реалізації задуму. Сучасні розробки характеризуються значною складністю, що вимагає залучення фахівців із різноманітними компетенціями: від програмування та архітектурного проєктування до глибокого аналізу даних (рисунок 1.1).



Рисунок 1.1 – Концептуальна модель факторів впливу на успішність ІТ-проєкту

Визначальним компонентом концептуальних засад є обсяг проєкту, який встановлює межі та специфікації робіт. Він слугує фундаментом для моніторингу та контролю, оскільки деталізує цілі, функції та очікувані результати. Чітке визначення обсягу допомагає уникнути непорозумінь із зацікавленими сторонами та запобігає ризику неконтрольованого розширення задач [6].

Специфіка галузі також вимагає особливої уваги до формування команди, яка зазвичай включає:

- розробників та системних архітекторів;
- аналітиків та менеджерів проєкту;
- спеціалістів із тестування та забезпечення якості;
- експертів з управління ризиками [7].

Ефективна взаємодія всередині такої групи базується на розробці прозорих каналів комунікації та координації зусиль під керівництвом менеджера, який відстежує прогрес і забезпечує відповідність робіт стратегічним цілям замовника. Таким чином, успіх ІТ-проєкту залежить від здатності інтегрувати технічні

інструменти, людський потенціал та структуровані процеси планування в єдину цілісну систему [8].

1.2 Аналіз проблем контролю та моніторингу проєктів у динамічному середовищі

Управління сучасними ІТ-проєктами, попри наявність інноваційних технологічних рішень, супроводжується низкою проблем, що ускладнюють процес моніторингу та негативно впливають на результативність. Динамічне середовище розробки вимагає від менеджерів не лише технічних знань, а й здатності оперативно реагувати на непередбачувані виклики, які виникають на різних етапах життєвого циклу продукту.

Одним із найбільш поширених викликів є постійна зміна вимог з боку замовника або кінцевих користувачів. Такі зміни зазвичай призводять до неконтрольованого збільшення обсягу робіт, суттєвих затримок у графіку та перевищення попередньо затвердженого бюджету. Ситуація ускладнюється у випадках, коли цілі проєкту не були чітко визначені на початковому етапі, що провокує непорозуміння щодо очікуваних результатів і невідповідність продукту реальним потребам ринку [9].

Окрему групу чинників становлять ресурсні та комунікаційні обмеження. Недостатність фінансових, людських або технічних ресурсів суттєво гальмує виконання завдань і може призвести до зниження якості програмного забезпечення (таблиця 1.1). Водночас неефективна комунікація між учасниками проєктної команди стає підґрунтям для конфліктів та неузгодженості дій, що загрожує успішному завершенню проєкту.

Технічні складнощі, пов'язані з інтеграцією нових технологій або непередбачуваними труднощами під час кодування, також вимагають посиленого контролю. Часто менеджери недооцінюють складність завдань, формуючи нереалістичні плани, що в поєднанні з недостатнім рівнем управління ризиками призводить до критичних збоїв.

Таблиця 1.1 - Ключові чинники проблемного контролю в ІТ-проектах

Категорія проблеми	Основні прояви та наслідки
Вимоги	постійна волатильність запитів замовника, розмиття меж проєкту;
Цілепокладання	відсутність чітких критеріїв успіху, розбіжності в очікуваннях;
Ресурси	дефіцит кваліфікованих кадрів, обмеження бюджету чи інфраструктури;
Комунікації	низька прозорість процесів, інформаційні розриви в команді;
Планування	недооцінка технічної складності, ігнорування потенційних ризиків.

Встановлення чіткого контролю над процесами розробки має вирішальне значення для забезпечення відповідності продукту встановленим стандартам якості та вимогам безпеки. Скрупульозний моніторинг дозволяє виявляти проблеми на ранніх стадіях, запобігаючи значним фінансовим втратам (рисунок 1.2). У швидкоплинному технологічному середовищі саме прозорість процесів підвищує довіру з боку замовників та забезпечує стабільність проєкту [10].

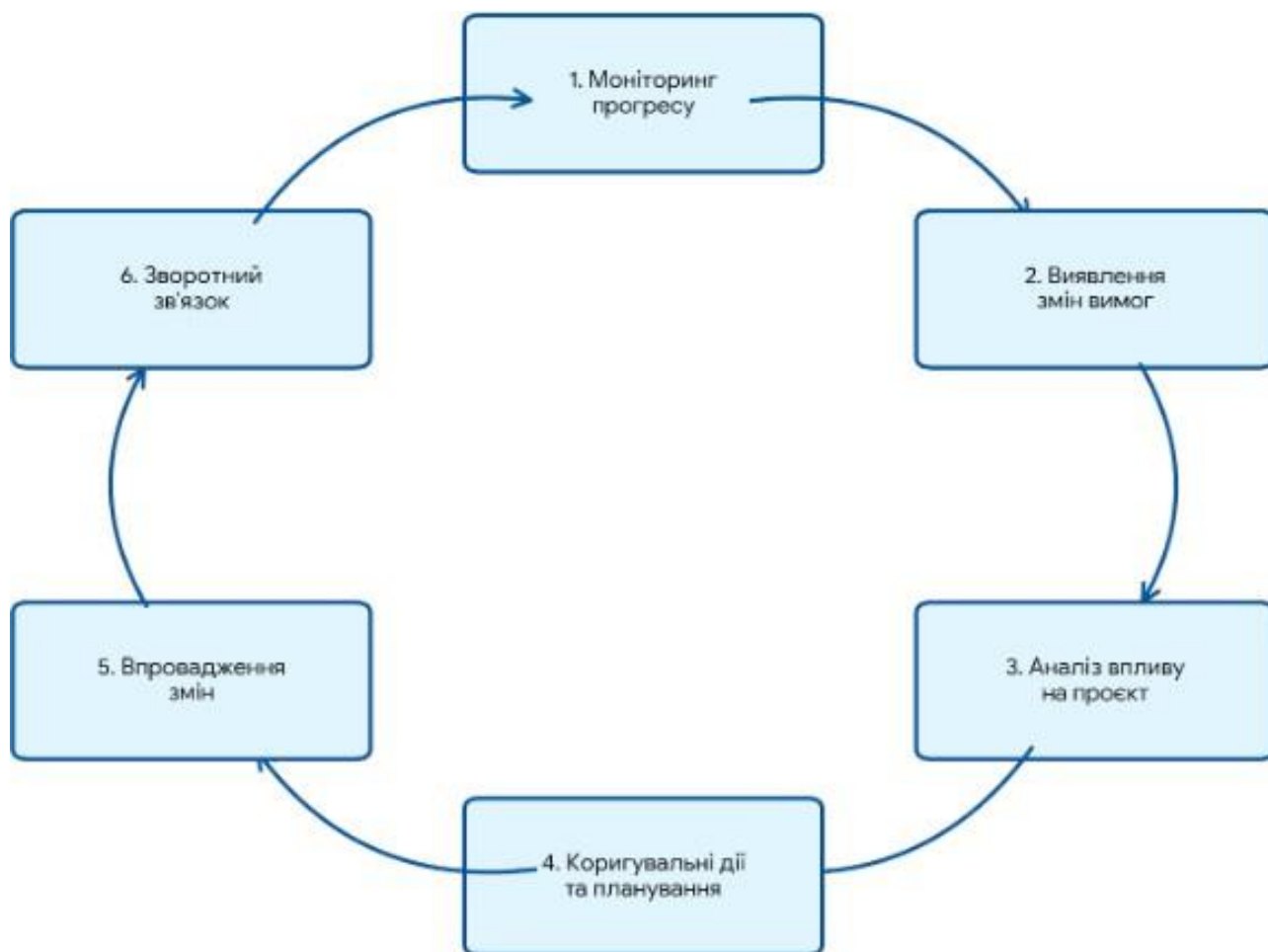


Рисунок 1.2 – Цикл моніторингу та контролю в умовах динамічних змін вимог

Варто зауважити, що для молодих спеціалістів існуючі комплексні системи контролю часто створюють високий поріг входу через заплутаність механізмів керування та відсутність простору для помилок. Тому актуальним завданням залишається пошук збалансованих інструментів, які б поєднували ефективність моніторингу з можливістю поступового навчання та адаптації новачків у галузі менеджменту.

1.3 Порівняльна характеристика інструментальних засобів та методологій (Scrum, Agile, Waterfall, Kanban)

Вибір методології управління проєктами є критичним етапом, що визначає структуру робочих процесів, способи комунікації та механізми контролю якості кінцевого продукту. У сучасній ІТ-індустрії домінують два принципово різні підходи: традиційний лінійний та гнучкий ітеративний, кожен з яких має власну специфіку застосування залежно від визначеності вимог та динаміки середовища.

Метод Waterfall, або водоспадна модель, представляє класичний підхід, що базується на строгій послідовності етапів, де перехід до наступної фази можливий лише після повного завершення попередньої [11]. Ця модель передбачає чітку структуру, що включає визначення вимог, проєктування, безпосередню реалізацію, тестування, впровадження та подальшу підтримку (рисунок 1.3). Такий лінійний підхід забезпечує високу прогнозованість результатів, легкість у плануванні ресурсів та бюджету, а також детальне документування на кожному кроці розробки. Проте Waterfall виявляє низьку адаптивність до змін, оскільки будь-які коригування вже реалізованої частини вимагають повернення до початкових фаз, що суттєво підвищує вартість проєкту.

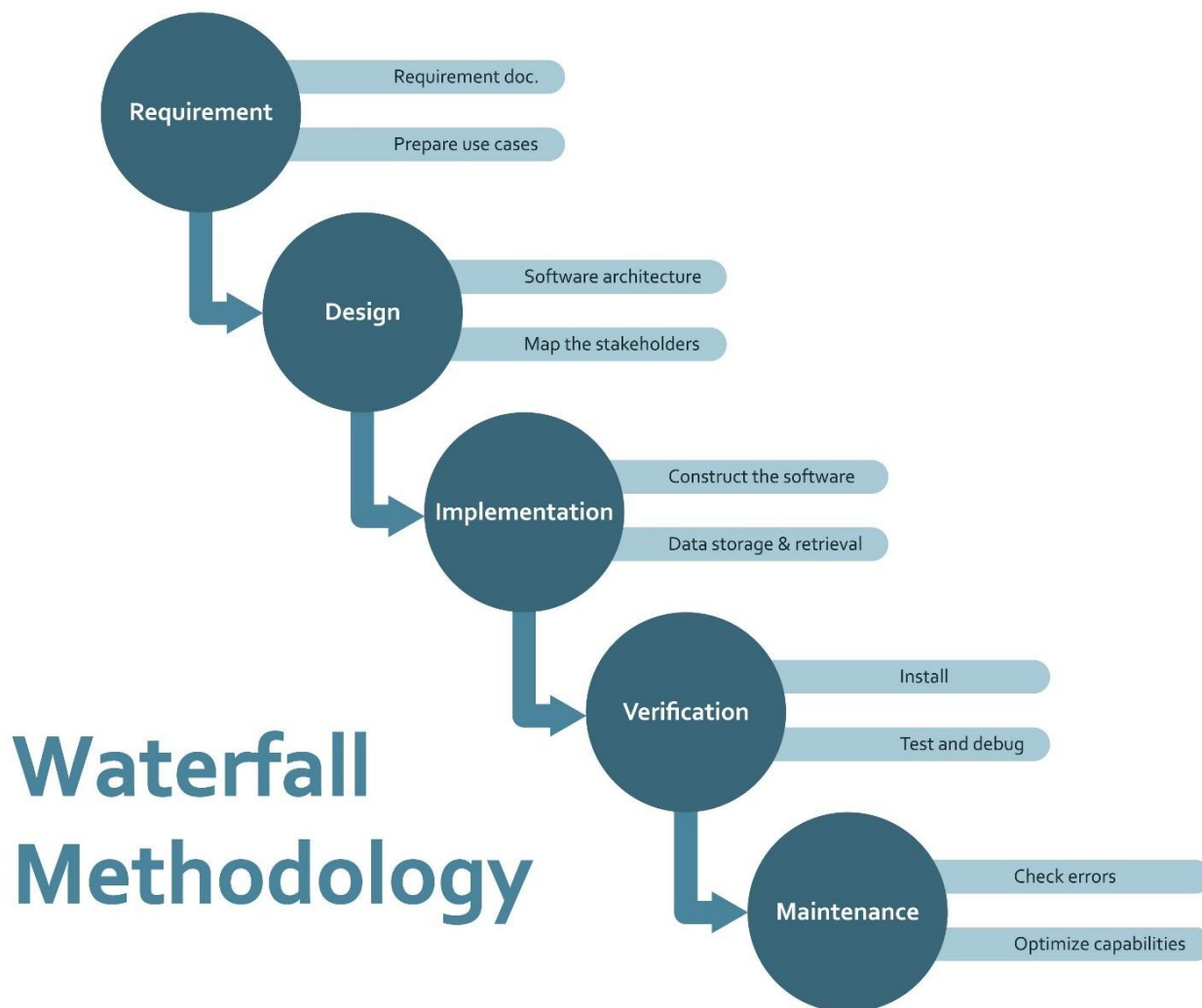


Рисунок 1.3 – Каскадна методологія

На противагу традиційним методам, філософія Agile акцентує увагу на швидкій адаптації до змін, ітеративному розвитку та постійній співпраці із зацікавленими сторонами. Гнучкі методології прагнуть мінімізувати ризики шляхом розбиття процесу розробки на короткі цикли – ітерації або спринти, що тривають від одного до чотирьох тижнів. Основними принципами Agile, закладеними у відповідному Маніфесті, є пріоритет взаємодії між людьми, фокус на функціонуючому програмному забезпеченні та готовність до змін понад слідування початковому плану.

Scrum є однією з найбільш поширених реалізацій Agile, яка базується на використанні крос-функціональних команд та самоорганізації. Процес управління в Scrum чітко регламентований через систему ролей, до яких належать власник продукту (Product Owner), Scrum-майстер та команда розробників (рисунок 1.4). Робота структурується через регулярні події: планування спринту, щоденні стендапи для синхронізації дій, огляд результатів та ретроспективу для вдосконалення внутрішніх процесів. Основними артефактами методу виступають беклог продукту, беклог спринту та інкремент продукту [12].

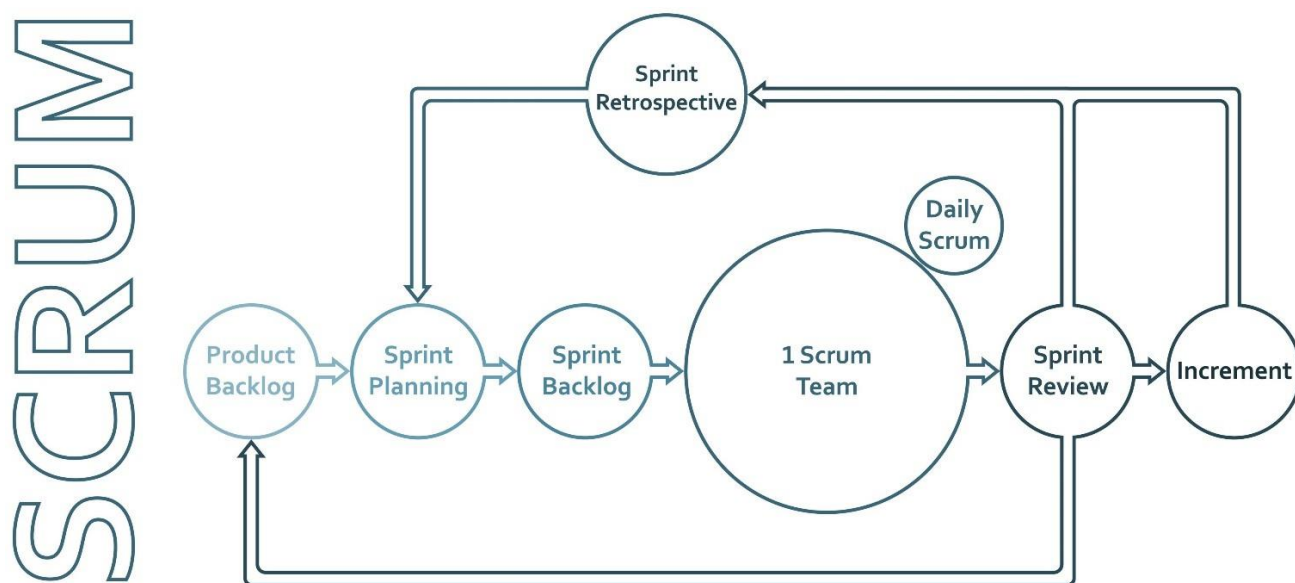


Рисунок 1.4 – Структура та взаємозв'язок елементів Scrum

Методологія Kanban орієнтована на оптимізацію потоку робіт через візуалізацію та прозорість процесів. На відміну від Scrum, Kanban не використовує фіксовані ітерації, а зосереджується на безперервному виконанні завдань (рисунок 1.5) (таблиця 1.2). Ключовими принципами тут є візуалізація роботи на спеціальних дошках, обмеження кількості одночасних завдань у процесі (WIP-ліміти) та управління потоком для мінімізації часу на виконання [13]. Це дозволяє

ідентифікувати «вузькі місця» в системі та забезпечує рівномірне навантаження на команду.

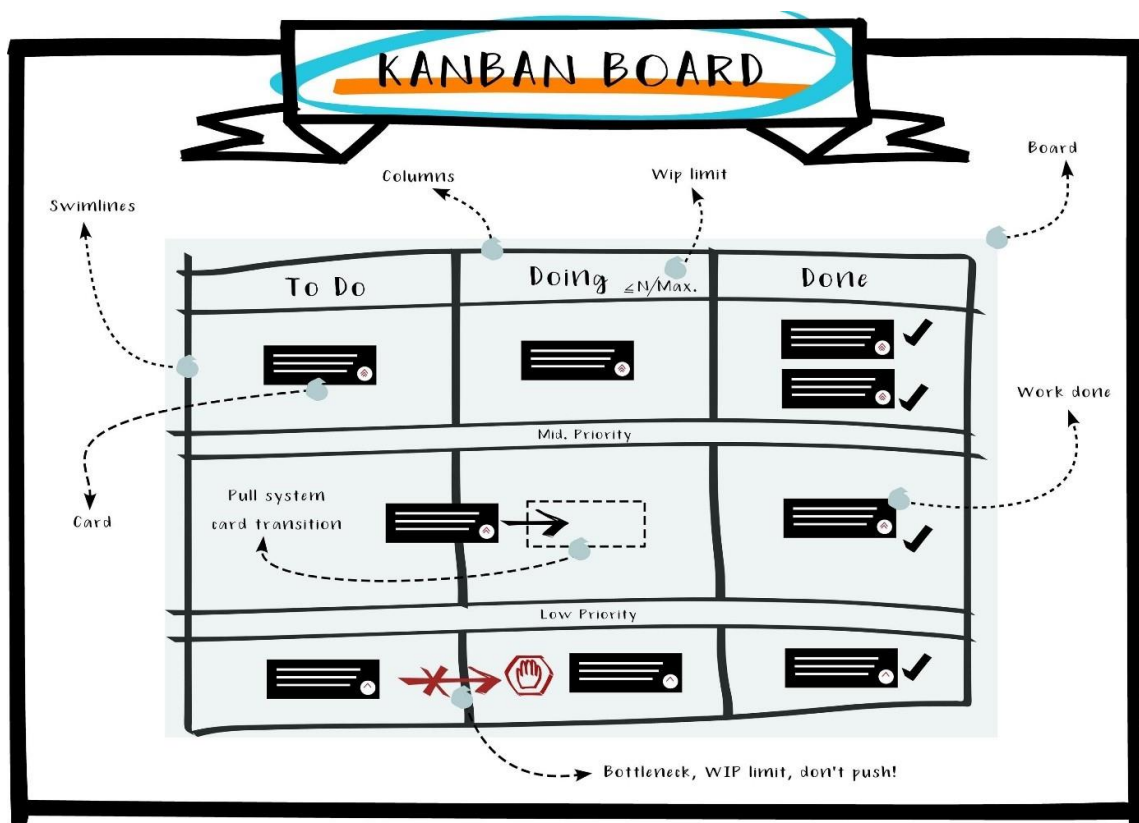


Рисунок 1.5 – Kanban дошка

Таблиця 1.2 – Порівняльний аналіз переваг та недоліків методологій управління

Метод	Переваги	Недоліки
Scrum	гнучкість, прозорість, швидка реакція на зміни, постійне вдосконалення;	вимагає високої дисципліни, складність масштабування, ризик перевантаження;
Waterfall	чітка структура, висока прогнозованість, легкість планування бюджету;	низька гнучкість, затримка тестування, ризик невідповідності очікуванням;
Agile	тісна співпраця з клієнтом, раннє випускання продукту, адаптивність;	важкість прогнозування фінального бюджету, потреба у високій самоорганізації;
Kanban	візуалізація затримок, відсутність перевантажень, універсальність;	менша структурованість, ризик невизначеності ролей у складних проєктах.

Для вибору оптимального підходу в контексті навчання та роботи з початківцями було проведено критеріальне порівняння методів за параметрами зв'язку з командою (K1), стабільності завдань (K2), відсутності перевантажень (K3), швидкості фідбеку (K4) та низького порогу входу (K5) (таблиця 1.3).

Таблиця 1.3 - Порівняння методів за специфічними критеріями ефективності

Методи	K1	K2	K3	K4	K5
Scrum	+	–	–	+	–
Waterfall	+	+	+	–	+
Agile	+	+	–	+	–
Kanban	–	–	+	+	+

Аналіз показує, що жодна з методологій не є абсолютно універсальною. Waterfall забезпечує необхідну стабільність та документальну базу, тоді як Agile та Scrum надають інструменти для швидкої адаптації до змін вимог (рисунок 1.6). У складних проєктах, використовують в рамках гібридної конфігурації.

ХАРАКТЕРИСТИКА	WATERFALL	SCRUM	AGILE	KANBAN
ВИМОГИ ТА ОБСЯГ	Чітко визначені, стабільні	Змінювані, ітеративні	Гнучкі, еволюційні	Будь-які, пріоритетні
КЛІЄНТ ТА ЗАМОВНИК	Низька залученість (на старті/фініші)	Постійна участь, фідбек	Високий рівень колаборації	Фокус на потоці цінності
СКЛАДНІСТЬ ТА ТЕХНОЛОГІЇ	Низька або середня	Висока, інноваційна	Висока, динамічна	Середня, адаптивна
КОМАНДА ТА МАСШТАБ	Велика, ієрархічна	Мала, крос-функціональна	Будь-яка, самоорганізована	Будь-яка, стабільна

Рисунок 1.6 – Матриця вибору методології залежно від характеристик проєкту

1.4 Обґрунтування вибору гібридних підходів для підвищення ефективності управління

Необхідність розгляду гібридних підходів у менеджменті ІТ-проєктів зумовлена динамічністю сучасного середовища та специфікою вимог зацікавлених сторін. Жодна з класичних методологій не є універсальною, оскільки Waterfall демонструє ефективність лише за умови стабільних вимог, тоді як Agile орієнтований на проєкти, що потребують швидкої адаптації до потреб користувачів. У більшості реальних сценаріїв розробка отримує значні переваги саме від комбінації цих підходів, що дозволяє збалансувати строгість планування із гнучкістю реалізації окремих завдань [14].

Використання змішаних методів дозволяє мінімізувати проєктні ризики на різних стадіях життєвого циклу продукту. Зокрема, застосування принципів Waterfall на ранніх етапах забезпечує передбачуваність та стабільність архітектурних рішень, а інтеграція елементів Agile на етапах безпосередньої розробки дає змогу оперативно вносити корективи та оптимізувати внутрішні процеси (таблиця 1.4). Такий підхід забезпечує дотримання договірних зобов'язань та часових рамок при одночасному збереженні високої якості програмного забезпечення.

Результати проведеного аналізу ризиків для різних гібридних конфігурацій підтверджують доцільність вибору моделі Waterfall-Agile. Математичне моделювання за допомогою розрахунку агрегованого ризику R продемонструвало суттєву перевагу цього підходу.

Таблиця 1.4 - Порівняльна оцінка агрегованих ризиків гібридних моделей

Гібридна модель	Значення агрегованого ризику (R)	Характеристика стабільності
Water-Scrum-Fall	7,8	потребує додаткових заходів мінімізації;
Waterfall-Agile	4,4	демонструє значно вищу ефективність.

Використовується гнучкість Agile, візуалізація Kanban та структурованість Waterfall. На початковому етапі Waterfall допомагає чітко визначити обсяг робіт і розробити план, що знижує ймовірність виникнення непорозумінь у майбутньому. Протягом виконання робіт візуальні таблиці Kanban забезпечують неперервний моніторинг прогресу та управління пріоритетами, а ітеративні спринти Agile дозволяють підтримувати постійний зворотний зв'язок із замовником.

Застосування такої гібридної моделі дозволяє:

- забезпечити чітке розуміння цілей проєкту та вимог до нього з самого початку розробки;
- підтримувати високу прозорість процесів для всіх учасників проєктної команди;
- ефективно масштабувати управління залежно від складності та динаміки задач;
- гарантувати якісне завершення проєкту та відповідність критеріям приймання замовника.

Отже, поєднання ітераційного планування, візуальних інструментів контролю та фазового підходу створює міцну основу для успішної реалізації ІТ-проєктів у сучасних умовах, особливо в межах систем дистанційного навчання.

1.5 Постановка задачі дослідження в межах кваліфікаційної роботи

Об'єктом дослідження в межах цієї кваліфікаційної роботи виступають методи управління IT-проектами, що спрямовані на формування архітектурних рішень та програмних інтерфейсів (API) для їхньої подальшої імплементації в сучасні системи дистанційного навчання (LMS). Предметом дослідження є процеси адаптації та інтеграції гібридних методологій управління, зокрема таких як Agile, Scrum та Kanban, для забезпечення ефективної та злагодженої роботи проєктних команд в умовах специфічного освітнього середовища.

Головна мета роботи полягає в детальному дослідженні та науковому обґрунтуванні вибору оптимальних гібридних підходів, які б максимально враховували особливості дистанційного навчання для підвищення загальної продуктивності, гнучкості та ймовірності успішного завершення проєктів.

Для досягнення поставленої мети в межах магістерської кваліфікаційної роботи необхідно вирішити такі наукові та практичні завдання:

- провести комплексний порівняльний аналіз сучасних методологій та інструментальних засобів управління проєктами в IT-галузі;
- вивчити специфічні чинники впливу динамічного середовища на процеси прийняття управлінських рішень та виявити недоліки існуючих підходів до моніторингу робіт;
- розробити вдосконалений гібридний метод управління IT-проектами на основі інтеграції каскадних (Waterfall) та гнучких (Agile, Kanban) підходів;
- спроектувати архітектуру та API системи управління проєктами, що забезпечуватиме візуалізацію робочих процесів через інтерактивні дошки задач;

- розробити функціональні рішення для інтеграції інтелектуальних агентів (персональних чатів на базі ШІ) з метою підвищення інтерактивності та зручності доступу до проєктної інформації [15];
- сформувати систему критеріїв і ключових показників ефективності (KPI) для оцінювання результативності впровадженого гібридного методу;
- провести експериментальну перевірку запропонованих рішень на прикладі реальних IT-проєктів у сфері дистанційної освіти.

Фінальним результатом дослідження є розробка гібридного методу, який дає можливість оптимізувати процеси управління шляхом поєднання високої технічної функціональності із низьким порогом входження для фахівців-початківців, що забезпечує прозорість контролю та стабільність результатів розробки програмних продуктів.

2 ДОСЛІДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ МЕТОДІВ КОМБІНОВАНОГО УПРАВЛІННЯ

2.1 Критерії оцінки ефективності методів управління

Успішна реалізація будь-якого ІТ-проєкту, незалежно від його масштабу та складності, безпосередньо залежить від обраної методології управління. Однак, в умовах постійної трансформації технологічного середовища та специфічних вимог систем дистанційного навчання, застосування суто класичних (монолітних) підходів часто виявляється недостатньо ефективним. Це обґрунтовує необхідність переходу до гібридних або комбінованих моделей, здатних інтегрувати переваги різних парадигм. Для того, щоб вибір структурних елементів такого комбінованого методу був об'єктивним та науково обґрунтованим, необхідно сформувати комплексну систему критеріїв оцінки ефективності. Визначення цих критеріїв слугує міцним фундаментом для вибору найбільш дієвих інструментів управління, забезпечуючи здатність проєктної команди досягати стратегічних цілей в умовах жорстких обмежень та динамічних змін.

Особлива увага під час формування критеріальної бази приділяється специфіці освітнього середовища та проблематиці адаптації молодих спеціалістів (початківців), які долучаються до розробки ІТ-продуктів. Критерії ефективності повинні дозволяти оцінювати не тільки фінальні показники проєкту (час, бюджет, якість), але й ефективність самих процесів управління на всіх етапах життєвого циклу розробки. Вони мають враховувати адаптивність проєктної команди до стресових змін, здатність проєкту відповідати зростаючим вимогам замовника, а також рівень задоволеності всіх учасників робочого процесу. Встановлення такої системи на етапі теоретичного дослідження створює міцну основу для об'єктивного аналізу різних методологій та інструментів, що сприяє розробці більш виважених рішень щодо їхньої практичної імплементації у програмну систему.

Враховуючи вищезазначене, було визначено п'ять ключових критеріїв (K1–K5), які відіграють вирішальну роль у створенні умов, що сприяють ефективній роботі та швидкому освоєнню навичок проєктного менеджменту:

- критерій K1 (Наявність зв'язку з досвідченою командою / Інститут наставництва): Акцент робиться на активній участі та постійній взаємодії всіх членів команди. Наявність досвідчених менеджерів, які здатні ділитися своїми знаннями та виступати в ролі наставників, є надзвичайно важливою умовою для підтримки новачків та забезпечення стабільного робочого процесу [16] ;

- критерій K2 (Стабільність та чітка структурованість завдань): Акцент робиться на передбачуваності робочого процесу. Стабільність завдань забезпечує зрозумілість процесу для початківців, запобігаючи ситуаціям, коли команда втрачає орієнтири через постійні, хаотичні зміни вимог чи нечіткість поставлених цілей;

- критерій K3 (Відсутність перевантажень учасників та процесу): Оцінює здатність методології запобігати професійному вигоранню та хаосу. Чітко визначена послідовність дій, яка не обтяжена надмірною кількістю бюрократичних етапів та обмежує обсяг одночасно виконуваних робіт, сприяє концентрації зусиль на якості продукту [17];

- критерій K4 (Швидкий зворотний зв'язок / Фідбек): Відображає можливість команди оперативно отримувати конструктивний зворотний зв'язок щодо виконаних завдань. Швидкий фідбек дозволяє ефективно вчитися на помилках, оперативно вносити необхідні корективи в код чи архітектуру та прискорює загальний прогрес розробки;

- критерій K5 (Низький поріг входу у систему контролю): Визначає ступінь доступності та зрозумілості інструментів управління для нових користувачів. Відсутність надто складних процедурних правил дозволяє новачкам швидше інтегруватися в робочий процес і почати ефективно працювати без тривалого попереднього навчання.

Для об'єктивізації вибору найбільш придатних інструментів, було проведено порівняльний аналіз класичних методологій контролю проєктів на предмет їхньої відповідності визначеним критеріям. Результати цього аналізу дозволяють виявити сильні та слабкі сторони кожної парадигми в контексті нашого дослідження (таблиця 2.1).

Таблиця 2.1 – Порівняння існуючих методів управління за критеріями ефективності

Модель або методологія управління	K1 (Зв'язок)	K2 (Стабільність)	K3 (Відсутність перевантажень)	K4 (Швидкий фідбек)	K5 (Низький поріг входу)
Scrum	+	-	-	+	-
Waterfall	+	+	+	-	+
Agile	+	+	-	+	-
Kanban	-	-	+	+	+

Аналізуючи отримані результати, можна констатувати, що методологія Agile та її специфічна реалізація Scrum, використовуючи такі елементи, як щоденні наради (Daily Stand-Up) та ретроспективи, ефективно задовольняють критеріям K1 та K4, забезпечуючи тісну комунікацію та швидкий фідбек. Проте динамічність спринтів часто суперечить критерію стабільності (K2), а інтенсивність процесів підвищує ризик перевантаження команди, що ускладнює поріг входу (K5). Метод Kanban, зосереджений на візуалізації через дошки завдань та встановленні WIP-лімітів (Work in Progress), ідеально відповідає критеріям відсутності перевантажень (K3) та низького порогу входу (K5), однак може страждати від браку жорсткої структуризації (K2). Каскадна модель Waterfall, своєю чергою, демонструє відмінні показники щодо стабільності фаз (K2) та уникнення перевантажень завдяки

послідовності (K3), але має критичний недолік у вигляді відсутності швидкого зворотного зв'язку (K4), оскільки тестування відбувається лише наприкінці циклу розробки.

Таким чином, результати критеріального порівняння підтверджують гіпотезу про те, що жоден з ізолюваних методів не здатен повною мірою задовольнити всі вимоги до сучасної системи управління ІТ-проектами. Це створює обґрунтоване підґрунтя для подальшого конструювання гібридного методу, який шляхом ретельного відбору та інтеграції елементів (наприклад, планування від Waterfall, адаптивності від Agile та візуального лімітування від Kanban) зможе забезпечити оптимальний баланс та задовольнити всім п'яти встановленим критеріям ефективності [18].

2.2 Аналіз ризиків існуючих комбінованих методологій (Water-Scrum-Fall, Waterfall-Agile)

Перехід від класичних ізолюваних методологій до гібридних підходів в управлінні ІТ-проектами є логічною відповіддю на неспроможність монолітних систем задовольнити весь спектр вимог сучасного динамічного середовища. Здатність гібридних методологій інтегрувати різні парадигми дозволяє збалансувати строгість планування із необхідною гнучкістю під час безпосередньої розробки коду. У межах нашого дослідження особливої уваги заслуговують дві найбільш поширені гібридні конфігурації: Water-Scrum-Fall та Waterfall-Agile. Для визначення найбільш оптимальної з них необхідно не лише проаналізувати їхній життєвий цикл, але й провести математичну оцінку агрегованих ризиків, що супроводжують їх імплементацію.

Методологія Water-Scrum-Fall являє собою спробу інтеграції гнучких фаз у жорсткий каркас традиційної каскадної моделі. Реалізація гібридного підходу в рамках діаграми (рисунок 2.1) демонструє детальну логіку послідовних переходів між різними парадигмами: спочатку здійснюється перехід від глобального архітектурного проєктування до ітеративного циклу розробки за методологією Scrum, після чого управління трансформується у фінальну каскадну стадію суворого тестування та операцій розгортання готового продукту. На початковому етапі відбувається детальне каскадне планування: збір вимог, формування бюджету та фіксація обсягу робіт. Це мінімізує початкові сумніви та забезпечує документальну базу. Далі розробка здійснюється ітеративними спринтами, що дозволяє команді адаптуватися до технічних викликів. Проте завершальна стадія проєкту (тестування, прийняття та розгортання) знову повертається до методології Waterfall, встановлюючи жорсткі "контрольні ворота" (gates) для релізів. Хоча такий підхід є гарним перехідним етапом для консервативних компаній, повернення до каскадного тестування наприкінці проєкту нівелює швидкість випуску продукту та затримує отримання фінального фідбеку від користувачів.

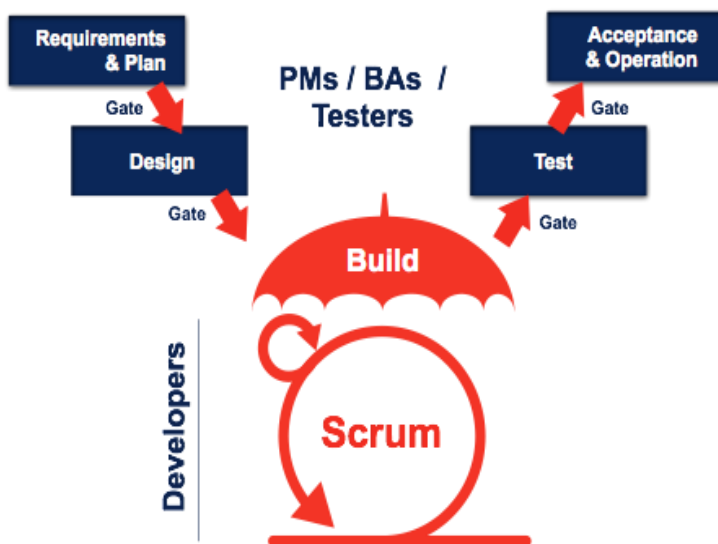


Рисунок 2.1 – Реалізація гібридного підходу в рамках діаграми Water-Scrum-Fall

Натомість, методологія Waterfall-Agile пропонує більш прогресивний рівень інтеграції. На відміну від попередньої моделі, де гнучкість обмежується лише фазою написання коду, у Waterfall-Agile традиційний підхід використовується виключно на етапі первинного бізнес-планування та збору архітектурних вимог [20]. Після завершення цього етапу весь подальший цикл – проєктування дизайну, реалізація, тестування та розгортання – відбувається за гнучкими ітеративними принципами Agile (рисунок 2.2). Це означає, що тестування та інтеграція не відкладаються на кінець проєкту, а виконуються безперервно під час кожного спринту. Такий підхід суттєво скорочує час доставки готових модулів системи та значно полегшує збір відгуків на ранніх стадіях, що дозволяє продукту краще відповідати мінливим вимогам клієнтів.

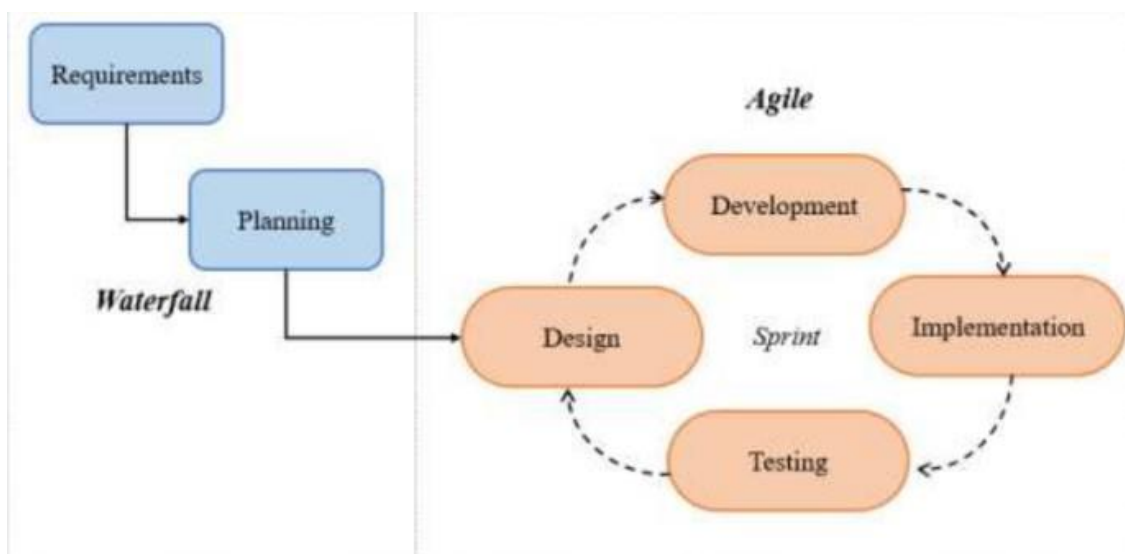


Рисунок 2.2 – Діаграма Waterfall-Agile

Для об'єктивного вибору кращого з цих двох підходів було використано існуючу математичну модель оцінки ризиків, яка враховує сукупний вплив різноманітних дестабілізуючих факторів на хід проєкту. Дослідження базувалося на статистичних дата-сетах організацій ISACA та OWASP [21], а обробка даних здійснювалася за допомогою аналітичної бібліотеки Pandas мовою Python. Однією зі

складових математичного моделювання стало використання методу PERT (Program Evaluation Review Technique) для розрахунку очікуваного часу завершення проєкту з урахуванням ризикових затримок [22]:

$$T_{exp} = \frac{o + 4m + p}{6}$$

де o – оптимістичний час завершення (без урахування ризиків);

m – найімовірніший час завершення;

p – песимістичний час завершення (з урахуванням можливих затримок).

Основою ж для порівняння став показник агрегованого ризику (R), який розраховується як сум добутоків ймовірності виникнення ризику на силу його впливу для всіх ідентифікованих категорій [23]:

$$R = \sum_{i=1}^n p_i \times u_i$$

де u_i – ймовірність виникнення i -го ризику (в діапазоні від 0.1 до 1.0);

p_i – рівень впливу i -го ризику на проєкт (оцінюється за шкалою від 1 до 5).

Враховуючи п'ять ключових категорій ризиків (технологічні, організаційні, комунікаційні, ринкові та оперативні), було проведено розрахунки для двох конкретних конфігурацій комбінованого управління: гібридної моделі Water-Scrum-Fall та методології Waterfall-Agile. Цей розрахунок повністю базується на експертних оцінках ймовірностей та рівнів впливу факторів дестабілізації. Результати цього обчислення зведено до порівняльної таблиці для наочної демонстрації різниці в ефективності (таблиця 2.2).

Таблиця 2.2 – Порівняльна оцінка параметрів ризику для комбінованих методологій

Категорія ризику	Water-Scrum-Fall: Ймовірність (ui)	Water-Scrum-Fall: Вплив (pi)	Waterfall-Agile: Ймовірність (ui)	Waterfall-Agile: Вплив (pi)
Технологічні	0.4	4	0.3	3
Організаційні	0.3	3	0.2	2
Комунікаційні	0.5	3	0.4	2
Ринкові	0.2	4	0.1	3
Оперативні	0.6	5	0.5	4

Відповідно до наведених у таблиці 2.2 даних, розрахунок агрегованого ризику (R) для методології Water-Scrum-Fall має такий вигляд:

$$R = (0.4 * 4) + (0.3 * 3) + (0.5 * 3) + (0.2 * 4) + (0.6 * 5) = 7.8$$

Загальний показник агрегованого ризику для цієї моделі, визначений експертним шляхом, складає 7.8. Таке значення є відносно високим і вимагає додаткових, посиленних заходів щодо мінімізації ризиків, оскільки воно свідчить про високу ймовірність виникнення оперативних затримок та "вузьких місць", особливо під час фінального етапу каскадного тестування.

Здійснимо аналогічний розрахунок агрегованого ризику для моделі Waterfall-Agile, використовуючи відповідні коефіцієнти, що були встановлені на основі експертних оцінок:

$$R = (0.3 * 3) + (0.2 * 2) + (0.4 * 2) + (0.1 * 3) + (0.5 * 4) = 4.4$$

Отриманий загальний ризик проєкту для методології Waterfall-Agile складає 4.4. Цей показник, обчислений експертним шляхом, є значно нижчим за попередній підхід, що прямо свідчить про значно вищу стабільність та загальну ефективність цієї комбінованої моделі в умовах розробки ІТ-проєктів.

Проведений математичний аналіз доводить, що загальний ризик проєкту при використанні методології Waterfall-Agile дорівнює 4.4, що є значно нижчим за

показник альтернативного підходу (рисунок 2.3). Оскільки ці чисельні значення ризиків визначалися саме експертним шляхом, вони відображають реальну специфіку розробки. Таким чином, проведені розрахунки дозволяють зробити конкретний висновок щодо вибору базової методології: модель Waterfall-Agile показала себе значно краще, забезпечивши зниження показника агрегованого ризику з 7.8 (для конфігурації Water-Scrum-Fall) до 4.4. Зниження ймовірності та сили впливу ризиків досягається завдяки тому, що тестування та інтеграція в цій моделі не відкладаються на кінець, а виконуються паралельно з розробкою, що нівелює критичні оперативні затримки. Дослідження однозначно вказує на те, що методологія Waterfall-Agile є значно ефективнішою та стабільнішою основою для подальшого проєктування нашої цільової системи управління ІТ-проєктами.

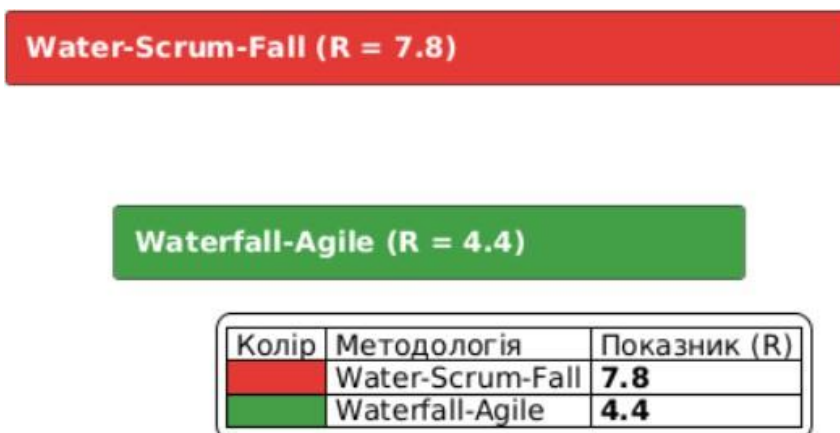


Рисунок 2.3 – Візуалізація агрегованого ризику гібридних методологій

2.3 Фільтрація та відбір оптимальних структурних елементів управління

Процес формування комбінованого методу управління вимагає не просто механічного поєднання різних підходів, а ретельного відбору тих структурних

елементів, які в сукупності забезпечать максимальну синергію та відповідність встановленим критеріям ефективності. Після проведення математичного аналізу ризиків та теоретичного порівняння методологій, наступним логічним кроком стало проведення прикладного дослідження за моделлю Кано (KANO) [24]. Основна ідея цього дослідження полягала у класифікації атрибутів системи управління за рівнем їхнього впливу на задоволеність користувачів (зокрема спеціалістів-початківців) та визначенні того, які функції є базовими, які – лінійними (перформанс-атрибути), а які – захоплюючими, що створюють додаткову цінність.

В межах дослідження буловедено опитування експертів та практикуючих менеджерів для оцінки вісімнадцяти потенційних елементів управління за двома векторами: вплив на швидкість адаптації нових членів команди та забезпечення загальної цілісності проєкту (рисунок 2.4). Результати анкетування, візуалізовані у формі гістограм, дозволили виявити чіткі тенденції важливості певного функціоналу. Зокрема, було встановлено, що деякі елементи традиційних систем сприймаються як надмірно бюрократизовані, тоді як інші – як критично необхідні для підтримки ритму розробки в умовах дистанційної взаємодії. На основі отриманих даних було проведено процедуру фільтрації структурних елементів управління вимогами проєкту. Під цим поняттям мається на увазі детальний аналіз та відсіювання другорядних управлінських функцій з метою виділення лише тих практичних інструментів, які безпосередньо впливають на прозорість фіксації завдань та швидкість адаптації персоналу. За результатами цієї фільтрації для наскрізного управління вимогами проєкту було відібрано шість базових елементів.

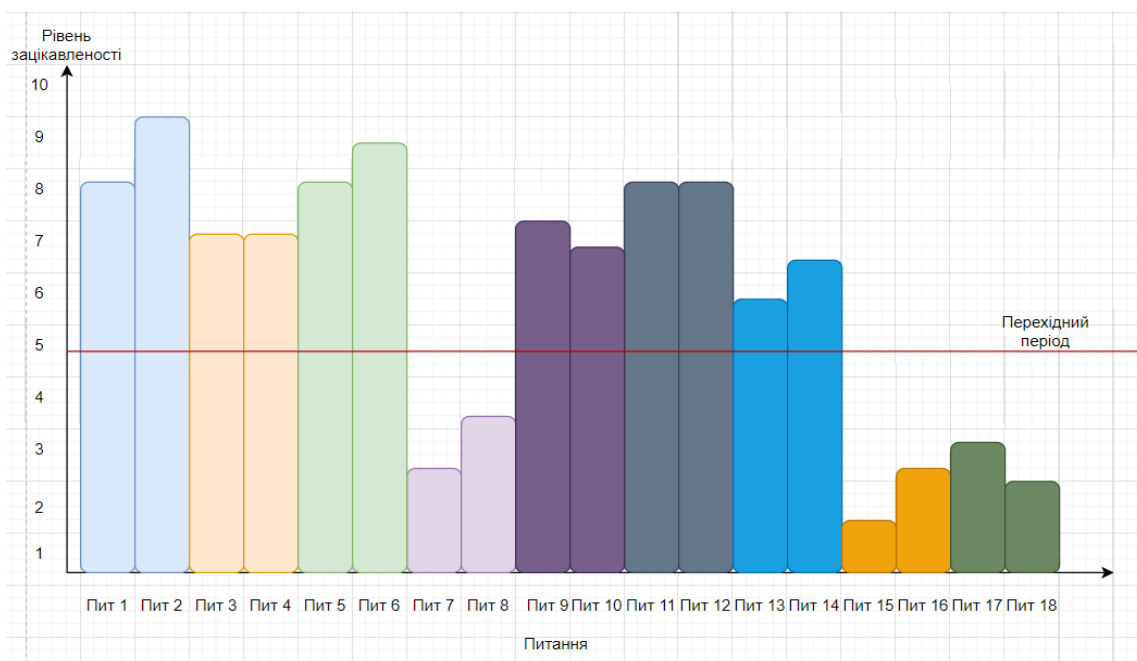


Рисунок 2.4 – Гістограма оцінки користувачами елементів управління вимогами проекту за моделлю Кано

Першим і фундаментальним обраним елементом стали інтерактивні дошки завдань (Task Boards) з обов'язковим впровадженням лімітів на кількість незавершеної роботи (WIP limits) [25]. Цей інструмент, запозичений з методології Kanban, дозволяє візуалізувати весь потік створення цінності, ідентифікувати затримки на ранніх стадіях та, що найважливіше, запобігти перевантаженню спеціалістів, забезпечуючи стабільний темп розробки. Встановлення WIP-лімітів виступає запобіжником від хаотичного перемикавання між задачами, що є критично важливим для збереження фокуса уваги розробників-початківців.

Другою групою відібраних елементів стали щоденні наради (Daily Stand-Up) та беклог спринту (Sprint Backlog). Ці інструменти Agile-парадигми забезпечують виконання критерію K1 (зв'язок із командою) та K4 (швидкий фідбек). Нарadi Daily Stand-Up дозволяють синхронізувати зусилля розподіленої команди в реальному часі, обговорюючи не тільки виконані завдання, а й перешкоди, що виникають. Беклог спринту (Sprint Backlog), своєю чергою, надає команді чіткий,

структурований список завдань на короткий проміжок часу, що забезпечує необхідну гнучкість без втрати загального напрямку руху. Для забезпечення довгострокового планування та структурованості (критерій K2) до фінального набору було включено стенд для планування (Planing Stand) та каскадне фазування проєкту (Waterfall stages). Використання фазової структури на етапі ініціації та архітектурного проєктування дозволяє зафіксувати загальні межі проєкту та етапи його релізу, що дає початківцям чітке розуміння "дорожньої карти" розробки. Такий підхід нівелює основний недолік чистого Agile – відсутність чіткої стратегічної перспективи на ранніх стадіях.

Для забезпечення довгострокового планування та структурованості (критерій K2) до фінального набору було включено стенд для планування (Planing Stand) та каскадне фазування проєкту (Waterfall stages). Використання фазової структури на етапі ініціації та архітектурного проєктування дозволяє зафіксувати загальні межі проєкту та етапи його релізу, що дає початківцям чітке розуміння "дорожньої карти" розробки. Такий підхід нівелює основний недолік чистого Agile – відсутність чіткої стратегічної перспективи на ранніх стадіях.

Окремо слід виділити впровадження інноваційного елемента, який отримав високу оцінку під час дослідження – інтеграцію інтелектуальних агентів (ІІІ) через API для автоматизованої декомпозиції задач та оцінки витрат часу. Це рішення дозволяє значно знизити поріг входу (критерій K5), оскільки штучний інтелект виступає в ролі цифрового помічника, який допомагає правильно розбити великі архітектурні фази на зрозумілі технічні підзадачі, призначаючи їм відповідні ролі (Frontend, Backend, DevOps) та орієнтовну тривалість. Така інтерактивна взаємодія з базою знань у формі чатів дозволяє користувачам миттєво отримувати контекстуальну допомогу без очікування фідбеку від старших колег.

Сформований набір елементів – інтерактивні дошки з WIP-лімітами, щоденні синхронізації, ітеративні беклоги, фазове планування та інтелектуальна підтримка ІІІ – утворює збалансовану систему координат комбінованого методу.

3 РОЗРОБКА ТА ПРОЄКТУВАННЯ КОМБІНОВАНОГО МЕТОДУ

Спираючись на результати теоретичного аналізу предметної області та математичної оцінки ризиків, описаних у попередніх розділах, наступним етапом дослідження є безпосереднє проєктування цільового комбінованого (гібридного) методу управління. Головним завданням цього розділу є формування концептуальної та логічної архітектури, яка дозволить органічно об'єднати відібрані управлінські інструменти в єдину екосистему. Спроєктована модель слугуватиме алгоритмічним фундаментом для подальшої розробки програмного забезпечення, орієнтованого на потреби систем дистанційного навчання та адаптацію молодих фахівців в ІТ-галузі.

3.1 Поєднання моделі Waterfall та методу Kanban

Поєднання концепцій Waterfall та методу Kanban у єдину систему управління вимагає чіткого розмежування зон їхньої відповідальності на різних етапах життєвого циклу проєкту. Суть розробленого підходу полягає у тому, що каскадна модель (Waterfall) використовується на макрорівні стратегічного управління, а гнучкі підходи (Kanban) – на мікрорівні (у період безпосереднього написання коду). Таке поєднання дозволяє задовольнити різнопланові потреби проєкту на різних його стадіях, забезпечуючи баланс між строгістю документації та свободою розробки.

Реалізація розробленого комбінованого методу базується на послідовному виконанні таких концептуальних етапів:

- першого етапу (макрорівень стратегічного планування), який передбачає застосування інструментарію каскадної моделі (Waterfall) на початкових стадіях

життєвого циклу – під час ініціації, збору детальних вимог, формування специфікацій та глобального архітектурного проектування;

- другого етапу (мікрорівень оперативного управління), що полягає у переході процесу розробки на гнучкі рейки методу Kanban безпосередньо у період написання коду, де завдання візуалізуються на інтерактивних дошках із впровадженням жорстких WIP-лімітів;

- третього етапу (інтелектуальна автоматизація та декомпозиція), який полягає в інтеграції інтелектуальних агентів на базі сучасних архітектур штучного інтелекту для автоматизації аналітичних процесів планування та декомпозиції складних завдань.

Впровадження інтелектуальних агентів на основі Gemini API для автоматичної декомпозиції задач у межах комбінованого методу реалізується через такі послідовні підетапи:

- формування контексту фази проєкту, що включає збір поточної інформації про завдання, наявні обмеження та вимоги до кінцевого продукту;

- промпт-інжинірингу та запиту, який передбачає передачу сформованого контекстуального промпту через Gemini API (модель Gemini 3 Flash) до інтелектуального агента;

- інтелектуальної генерації, під час якої ШІ виконує аналіз повноти декомпозиції та формує логічно обґрунтований список підзадач із розгорнутим технічним описом, призначенням IT-ролей та оцінкою трудомісткості в годинах;

- структурування та синхронізації, що забезпечує отримання відповіді у форматі JSON, автоматичне оновлення візуальних дошок Kanban та динамічний перерахунок тривалості етапів моделі Waterfall.

Функціональна структура розробленого гібридного підходу на стику стратегічного та операційного рівнів передбачає наскрізне виконання таких практичних процесів:

- визначення IT-ролей для чіткого розподілу зон відповідальності та

навантаження між учасниками розподіленої команди;

- побудови реактивних дошок Kanban для візуалізації поточного стану завдань та контролю WIP-лімітів у реальному часі;
- автоматичної декомпозиції високорівневих фаз проєкту на атомарні підзадачі з розрахунком їхньої трудомісткості;
- обробки асинхронних подій для забезпечення динамічного перерахунку загальних термінів каскадної моделі при зміні статусу будь-якого завдання.

Детальний розбір механізму автоматичної декомпозиції дозволяє чітко простежити алгоритмічний взаємозв'язок між гнучким плануванням та каскадним контролем термінів (рисунок 3.1).

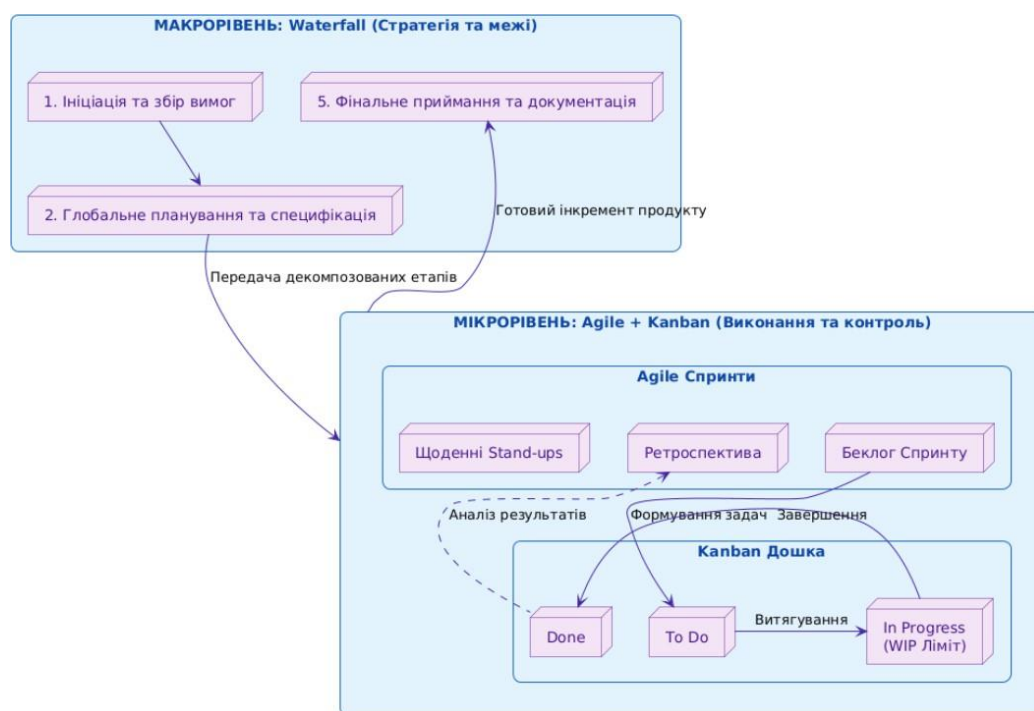


Рисунок 3.1 – Схема поєднання моделі Waterfall та методу Kanban

В результаті розроблено метод управління Waterfall-Kanban, який функціонує як єдиний, збалансований механізм. Модель Waterfall гарантує дотримання стратегічного вектора, чіткість бюджету та прозорість критеріїв приймання. Метод

Kanban, своєю чергою, забезпечує безперервну візуалізацію процесів, управління пріоритетами, гнучку реакцію на зміни під час написання коду та захищає команду від перевантаження через нормування поточного навантаження. Такий синергетичний підхід дозволяє ефективно масштабувати управління залежно від складності поставлених задач, роблячи його оптимальним рішенням для освітніх та комерційних ІТ-проектів.

3.2 Інноваційна імплементація інтелектуальних агентів (ІІІ) для декомпозиції та оцінки задач

Наукова новизна та практична значущість розробленого комбінованого методу управління проектами суттєво підсилюється шляхом інтеграції інтелектуальних агентів на базі сучасних архітектур штучного інтелекту (ІІІ) [28]. У межах даного дослідження імплементація ІІІ розглядається не як допоміжний інструмент, а як інтегрований компонент системи, що виконує роль цифрового консультанта та автоматизує найбільш складні аналітичні процеси на етапі планування та декомпозиції. Таке рішення дозволяє розв'язати критичну проблему високого порогу входу для фахівців-початківців (критерій K5), надаючи їм інтелектуальну підтримку під час формування ієрархічної структури робіт (WBS).

Технологічна реалізація даного компонента базується на використанні програмного інтерфейсу Gemini API (модель Gemini 3 Flash), інтегрованого у серверну частину розробленої системи. Алгоритмічна логіка взаємодії передбачає передачу контекстуальної інформації про поточну фазу проекту та перелік уже існуючих завдань до інтелектуального агента. Проектне рішення спрямоване на те, щоб ІІІ виступав у ролі «Senior Project Manager», який проводить критичний аналіз повноти декомпозиції етапу та пропонує логічно обґрунтовані підзадачі. Кожна

згенерована задача включає не лише стислу назву, а й розгорнутий технічний опис, призначення конкретної ІТ-ролі (Frontend, Backend, DevOps тощо) та орієнтовну оцінку трудомісткості в годинах.

Процес імплементації взаємодії з інтелектуальними агентами побудований на принципах промпт-інжинірингу, що гарантує отримання структурованих даних у форматі JSON для забезпечення повної сумісності з внутрішньою базою даних системи [29]. Це дозволяє автоматично оновлювати візуальні дошки Kanban та перераховувати тривалість каскадних етапів у моделі Waterfall на основі отриманих від ШІ оцінок часу. Таким чином, використання інтелектуальних агентів забезпечує безперервну валідацію планів розробки та підвищує стабільність структурованості завдань (критерій K2), мінімізуючи ризики пропуску критичних підзадач на ранніх стадіях проєкту.

Крім автоматичної декомпозиції, інноваційним аспектом системи є можливість якірного прикріплення посилань на персональні діалоги з ChatGPT або іншими ШІ-моделями безпосередньо до об'єктів системи управління. Такий підхід забезпечує збереження контекстуальної пам'яті проєкту та дозволяє користувачам миттєво переходити до релевантних обговорень архітектурних рішень чи фрагментів коду, пов'язаних із конкретною задачею. Інтеграція інтелектуальних агентів у такий спосіб створює адаптивне інформаційне середовище, яке створює умови для самостійного навчання розробників та забезпечує високу якість фінального ІТ-продукту при зниженні навантаження на досвідчених менеджерів.

3.3 Опис реалізації та структурної взаємодії елементів розробленого методу

Практична реалізація спроектованого комбінованого методу базується на тісній системній взаємодії всіх його компонентів, де кожен елемент виконує

специфічну функцію в межах життєвого циклу розробки програмного забезпечення для систем дистанційного навчання та адаптації молодих ІТ-фахівців. Компонентами методу, що виступають як взаємопов'язані етапи, є:

- компонент стратегічного планування, який відповідає за фіксацію базових термінів, етапів та меж проєкту;
- компонент оперативного візуального контролю, що забезпечує гнучкий розподіл та моніторинг поточних завдань між виконавцями;
- компонент інтелектуального аналізу, який здійснює автоматизовану декомпозицію та предиктивну оцінку трудомісткості робіт.

Основним інженерним викликом під час розробки стало забезпечення динамічного зв'язку між жорстким фазовим плануванням Waterfall та оперативною гнучкістю Kanban-дошок. Структурна цілісність методу досягається за допомогою архітектури клієнт-серверної взаємодії, де бекенд-логіка (Node.js) автоматично перераховує часові параметри проєкту при зміні статусу будь-якої дрібної задачі (рисунок 3.2).

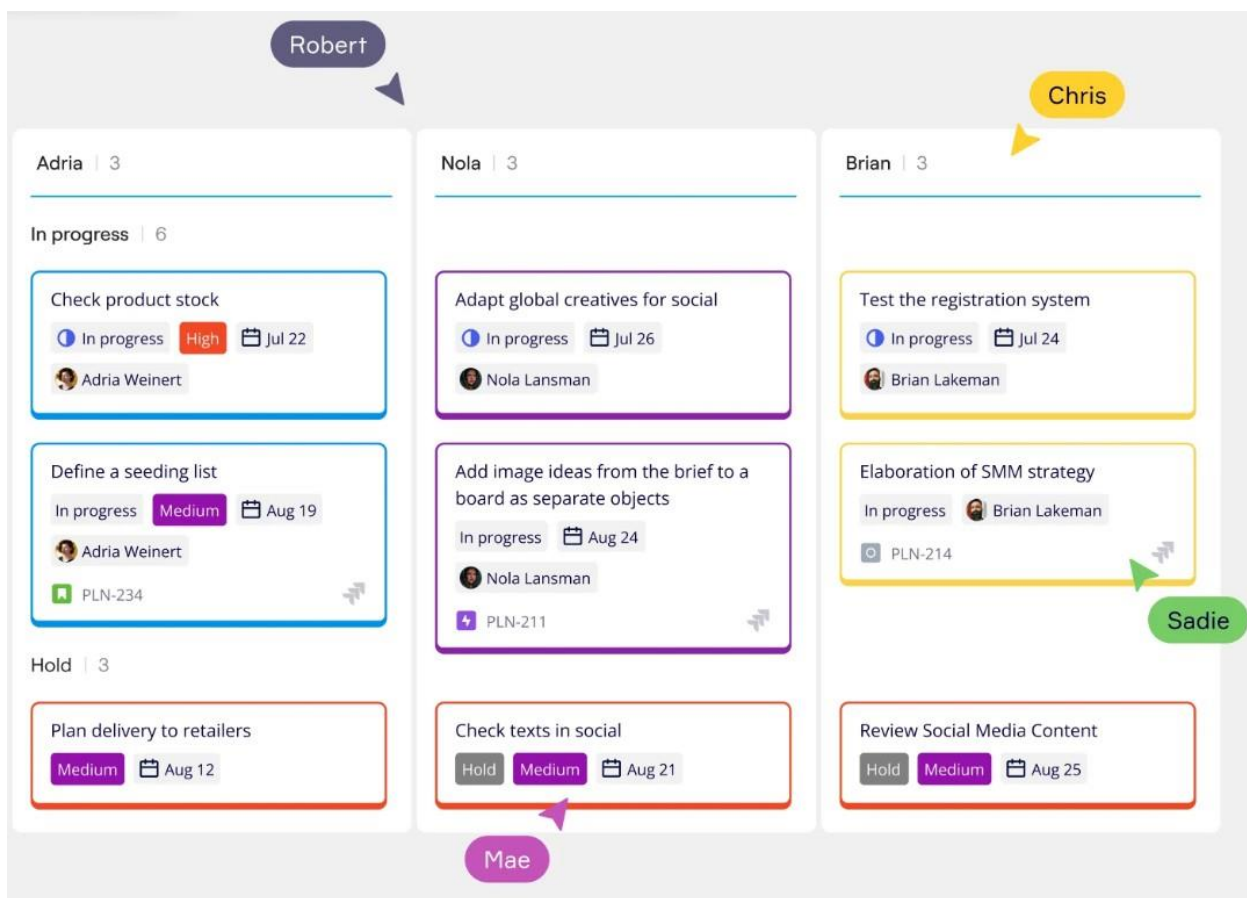


Рисунок 3.2 – Схема каскадного планування з часовими зміщеннями

Реалізація компонента Waterfall у системі базується на алгоритмі каскадного зміщення («offset logic») [30]. Для кожної фази проєкту (наприклад, «Ініціація», «Проектування», «Розробка») розраховується параметр `startOffset`, який визначає точну дату початку етапу відносно завершення попереднього. Особливістю методу є те, що тривалість фази (`duration`) не є статичною величиною: вона автоматично обчислюється на основі сумарної оцінки годин усіх підзадач, що входять до цієї фази, за формулою, де сума годин ділиться на стандартний 8-годинний робочий день.

На наведеній схемі каскадного планування використовуються специфічні параметри, які потребують текстового пояснення для однозначного тлумачення логіки системи: зокрема, параметр `address` позначає унікальний ідентифікатор або ім'я співробітника, який виконує поточну задачу, що дозволяє системі точно

прив'язувати часові зміщення до ресурсних обмежень конкретного виконавця; параметр id визначає унікальний номер завдання в загальній ієрархічній структурі робіт (рисунок 3.3). Такий підхід забезпечує повну прозорість планування та дозволяє замовнику бачити реальні терміни релізу, що базуються на фактичному обсязі робіт.

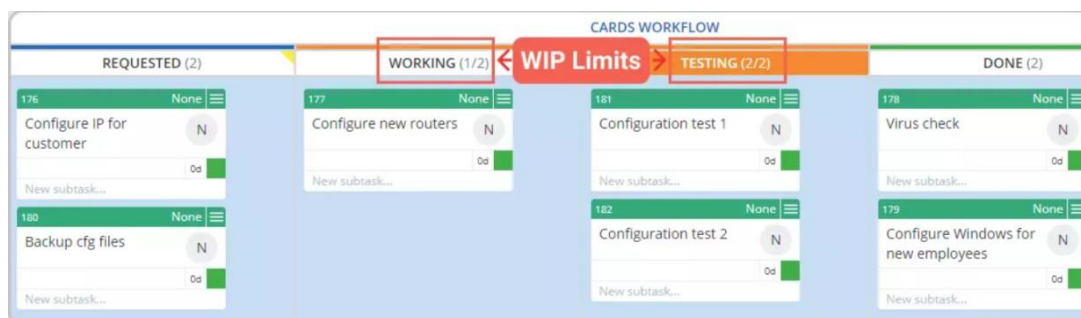


Рисунок 3.3 – Інтерактивна Канбан-дошка з встановленими WIP-лімітами

Взаємодія на рівні виконання реалізована через цифрові Kanban-дошки, які інтегровані з беклогами спринтів. Задля підтримки дисципліни розробки та запобігання перевантаженню команди (критерій K3) у системі впроваджено механізм жорстких WIP-лімітів. WIP-ліміт (Work in Progress Limit) – це встановлене максимальне обмеження на кількість завдань, які можуть одночасно перебувати в певній колонці робочого процесу або виконуватися одним розробником. На програмному рівні це реалізовано як сувора валідація при спробі перемістити задачу в колонку «InProgress»: якщо ліміт для конкретного розробника або для всієї колонки вичерпано, система повністю блокує дію користувача. Зворотний зв'язок у системі реалізовано через миттєве відображення візуального попередження та блокування інтерфейсу перетягування (Drag-and-Drop), що сигналізує про виникнення «вузького місця» у процесі розробки. Це змушує команду

концентруватися на завершенні поточних задач, а не на накопиченні нових, що є суттєвим для підтримки ритму в ітеративній моделі.

Ядром системи планування є багаторівнева структура автоматизації, яка охоплює три послідовні рівні:

- перший рівень, який забезпечує базове ручне введення та первинну фіксацію параметрів проєкту користувачем;
- другий рівень (програмний), що здійснює автоматичний розрахунок часових зміщень, тривалості фаз та перерахунок критичного шляху на основі алгоритмічної логіки бекенду;
- третій рівень (інтелектуальний), який використовує можливості штучного інтелекту для предиктивного аналізу, декомпозиції та оцінювання завдань.

Саме цей третій рівень реалізовано як модуль взаємодії з інтелектуальними агентами. Структурно цей елемент взаємодіє з беклогами через спеціальний API-ендпоінт (рисунок 3.4), який активується при запиті на генерацію задач. Система передає ШІ-моделі контекст фази, і у відповідь отримує структурований JSON-масив із декомпозованими завданнями. Прикметно, що кожна задача автоматично отримує унікальний ідентифікатор та прив'язку до конкретної ролі, що дозволяє системі миттєво розподіляти навантаження між учасниками проєкту. Такий рівень автоматизації дозволяє підтримувати високу стабільність структурованості (критерій K2) навіть у проєктах із високим рівнем невизначеності.

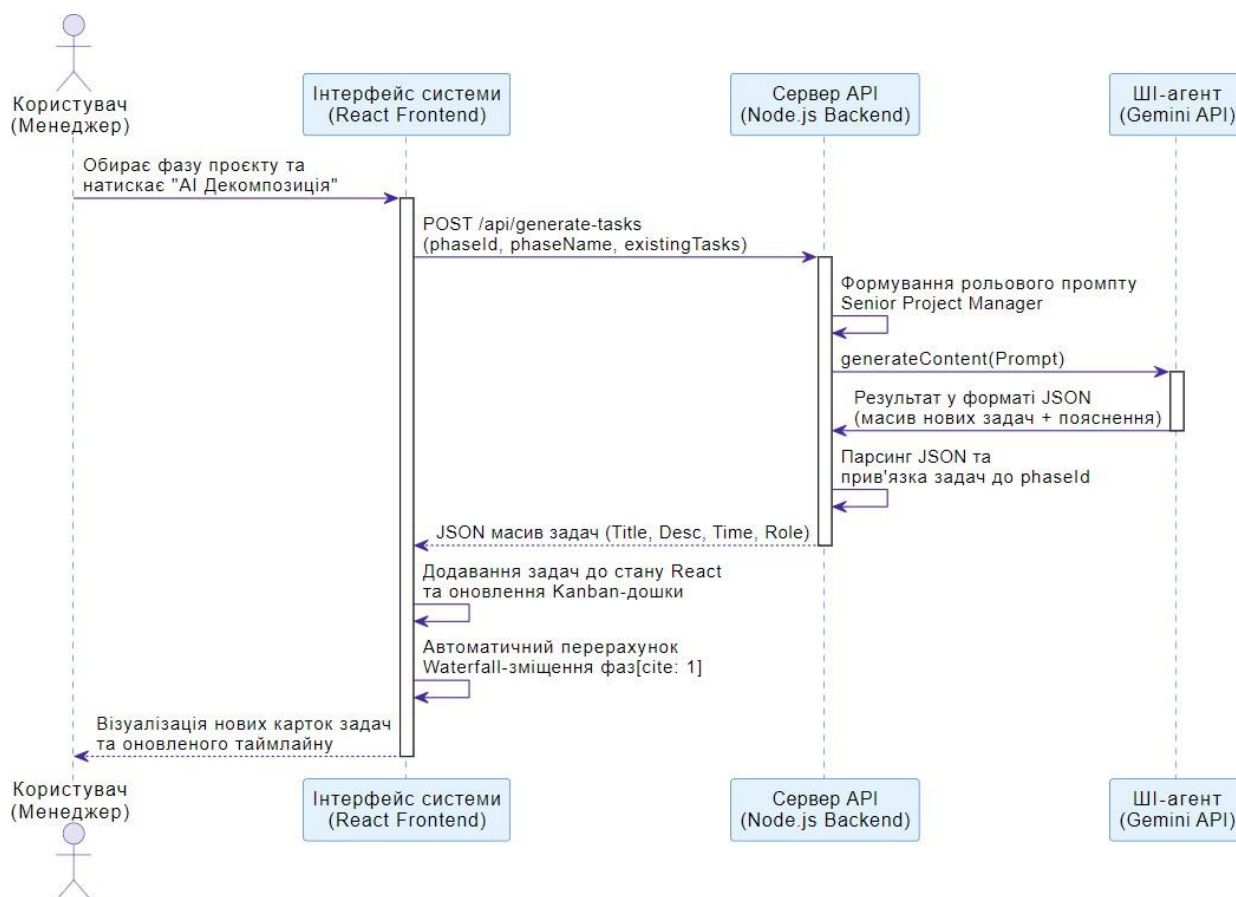


Рисунок 3.4 – Діаграма послідовності взаємодії користувача з ШІ-агентом у системі

Розгорнутий текстовий аналіз діаграми послідовності дозволяє детально описати прецеденти (Use Cases) та функції API як інтелектуального агента. Основним прецедентом використання є «Автоматична декомпозиція етапу проєкту». У межах цього прецеденту користувач ініціює запит через клієнтський інтерфейс, після чого система збирає метадані поточного контексту розробки. Функції API як інтелектуального агента полягають у виконанні ролі експертного аналітика, який бере на себе когнітивне навантаження з інтерпретації вимог: агент аналізує надісланий промпт, проводить семантичний розбір та декомponує високорівневу задачу на атомарні підзадачі.

Взаємодія користувача з Gemini API побудована за асинхронною схемою: клієнт відправляє запит до сервера, сервер трансформує його у валідований JSON-

запит до API, отримує структуровану відповідь, перевіряє її на відповідність внутрішній схемі даних та повертає результат на фронтенд, де відбувається динамічне рендерення нових карток завдань.

Для забезпечення безперервного фідбеку (критерій K4) та навчальної підтримки, реалізовано механізм якірних посилянь на ШІ-чати. Кожна задача на дошці має кнопку швидкого доступу до персоналізованого контексту в ChatGPT, де розробник може отримати роз'яснення щодо технічної реалізації конкретного таска. Вся структурна взаємодія елементів – від каскадного плану до інтелектуальної декомпозиції – замикається в єдиний цикл моніторингу, де оновлення одного показника ініціює перерахунок всього дерева проєкту. Це створює живу, адаптивну систему управління, яка поєднує строгість академічного підходу з гнучкістю сучасних ІТ-практик.

4 ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ГІБРИДНОГО МЕТОДУ

У даному розділі описується процес технічного втілення спроектованого комбінованого методу управління ІТ-проєктами. Основна увага приділяється обґрунтуванню вибору інструментальних засобів розробки, опису архітектурних рішень та програмній реалізації ключових модулів системи: каскадного планування, інтерактивних дошок з WIP-лімітами та блоку взаємодії з інтелектуальними агентами штучного інтелекту.

4.1 Вибір технологічного стеку та розробка архітектури клієнт-серверного рішення (Node.js, React)

Для реалізації програмної системи, що поєднує в собі гнучкість Agile та структурність Waterfall, було обрано сучасний стек технологій, який забезпечує високу швидкість розробки, масштабованість та можливість легкої інтеграції зі сторонніми API. Основним критерієм вибору стала здатність технологій підтримувати асинхронну взаємодію в реальному часі, що є критичним для систем моніторингу проєктів.

В основу розробки клієнт-серверного рішення покладено архітектурну модель Single Page Application (SPA) вебдодатка «Hybrid Manager», де клієнтська та серверна частини функціонують незалежно, взаємодіючи через протокол HTTP за допомогою RESTful API.

Нижче наведено перелік основних програмних залежностей проєкту додатка «Hybrid Manager». Ці залежності безпосередньо забезпечують технічну підтримку таких інструментів та конструктів методу, як автоматичний розрахунок термінів фаз,

ведення беклогу завдань, калькуляція навантаження та інтелектуальна декомпозиція робіт:

- Node.js та Express.js (Backend): Серверна частина реалізована на платформі Node.js, яка використовує неблокуючу модель введення-виведення, що ідеально підходить для обробки великої кількості одночасних запитів. Фреймворк Express.js обрано за його мінімалізм та високу продуктивність при створенні API-ендпоінтів для маніпуляції даними фаз та задач;
- React.js (Frontend): Для побудови інтерфейсу користувача використано бібліотеку React.js. Компонентний підхід дозволяє створити реактивні елементи управління, такі як динамічні дошки Kanban та інтерактивні графіки Waterfall, що миттєво оновлюються при зміні стану даних;
- Vite: Виступає як інструмент збірки та середовище розробки, що забезпечує швидке «гаряче» перезавантаження модулів (HMR) та оптимізацію фінального коду застосунку;
- Tailwind CSS: Використовується для стилізації інтерфейсу за принципом utility-first, що дозволяє швидко створювати адаптивний та сучасний дизайн (UI), орієнтований на зручність користувачів-початківців;
- Google Generative AI (Gemini): Бібліотека `@google/generative-ai` інтегрована для забезпечення роботи інтелектуальних агентів, що виконують автоматичну декомпозицію проектних задач.

Нижче наведено перелік основних залежностей проекту, які визначають склад технологічного стеку.

Лістинг 4.1 – Конфігурація основних залежностей системи (фрагмент package.json)

```
JSON
{
  "dependencies": {
    "@google/generative-ai": "^0.1.0",
```

```
    "cors": "^2.8.5",
  },
  "devDependencies": {
    "@tailwindcss/vite": "^4.0.0",
    "tailwindcss": "^4.0.0",
    "vite": "^5.4.1"
  }
}
```

Архітектура рішення побудована за принципом чіткого розподілу обов'язків (Separation of Concerns) у межах додатка «Hybrid Manager». Серверна частина відповідає за бізнес-логіку: розрахунок тривалості фаз, управління станами задач та комунікацію з ШІ. Клієнтська частина забезпечує візуалізацію даних та інтерактивну взаємодію користувача з елементами управління. Наведена схема відображає архітектурну взаємодію інструментів, що призначені для підтримки комбінованих технологій управління в системі «Hybrid Manager»

Для систематизації вибору компонентів стеку розроблено порівняльну таблицю, яка відображає, як саме етапи методу підтримуються на рівні програмного коду, бекенду, фронтенду та сторонніх бібліотек додатка «Hybrid Manager» (таблиця 4.1).

Таблиця 4.1 – Призначення технологічних інструментів у системі

Компонент	Технологія	Програмна підтримка етапів методу (код, бібліотеки)
Backend Runtime	Node.js	забезпечення виконання серверного коду для обробки асинхронних запитів;
API Framework	Express.js	створення ендпоінтів бекенду для маршрутизації та маніпуляцій з даними;
UI Library	React	реалізація фронтенд-компонентів для динамічного відображення інтерфейсу;
Styling	Tailwind CSS	застосування утилітарних стилів для гнучкого адаптивного дизайну клієнтської частини;
AI Integration	Gemini API	інтеграція зовнішньої бібліотеки штучного інтелекту

Розроблена архітектура дозволяє додатку «Hybrid Manager» стабільно функціонувати в умовах динамічної зміни вимог. Використання Node.js забезпечує швидкий перерахунок каскадних зміщень Waterfall на сервері, а React гарантує миттєве відображення цих змін на клієнтській стороні, що підтримує високу прозорість моніторингу проєкту. Такий підхід повністю відповідає академічним вимогам до сучасних управляючих рішень та забезпечує практичну цінність розробленого програмного продукту.

4.2 Програмна реалізація алгоритмів розрахунку тривалості етапів та каскадного зміщення (Waterfall)

Однією з ключових функціональних особливостей розробленого додатка є забезпечення прогнозованості проєкту за допомогою каскадної моделі Waterfall. В основі реалізації лежить логіка динамічного розрахунку тривалості кожної фази та автоматичного обчислення часового зміщення етапів у календарному плані. Це дозволяє підтримувати актуальну «дорожню карту» проєкту, яка автоматично реагує на будь-які зміни в обсязі робіт на мікрорівні.

Програмна логіка реалізована на стороні сервера в межах обробника маршруту GET /api/phases. Процес розрахунку часових параметрів складається з декількох послідовних кроків:

- агрегації даних, що передбачає для кожної фази проведення фільтрації всіх асоційованих із нею задач та збір даних про оціночний час їх виконання;
- розрахунку трудомісткості, за якого система сумує години, вказані в описах задач, трансформуючи їх у числовий формат;

- обчислення тривалості, де загальна кількість годин конвертується у робочі дні за формулою, де сума годин ділиться на стандартний 8-годинний робочий день із округленням до більшого цілого числа;
- каскадного зміщення, де для забезпечення лінійної послідовності Waterfall початок кожної наступної фази (startOffset) встановлюється рівним сумі тривалостей усіх попередніх етапів.

Математично розрахунок тривалості фази (D) можна представити наступним чином:

$$D = \max \left(\left\lceil \frac{\sum_{i=1}^n H_i}{8} \right\rceil, 1 \right)$$

де H_i – оціночна кількість годин для i -ї задачі в межах фази, а 1 день є мінімально допустимою тривалістю етапу.

Нижче наведено програмну реалізацію описаного розрахунку на мові JavaScript із використанням методів масивів для агрегації даних.

Лістинг 4.2 – Програмна реалізація каскадного розрахунку фаз (server.js)

```
- app.get('/api/phases', (req, res) => {;
- let currentOffset = 0; // Накопичувальне зміщення для каскаду (Waterfall).

const enriched = phases.map(p => {
  // Відбір задач, що належать до поточної фази
  const pTasks = tasks.filter(t => t.phaseId === p.id);
  const completed = pTasks.filter(t => t.status === 'done').length;

  // Сумування годин з усіх задач етапу (парсинг оцінки типу "16 год")
  const totalHours = pTasks.reduce((sum, task) => {
    const hours = parseInt(task.time) || 0;
    return sum + hours;
  }, 0);

  // Розрахунок тривалості в днях (8-годинний робочий день)
  const calculatedDuration = Math.max(Math.ceil(totalHours / 8), 1);
```

```

const phaseWithTime = {
  ...p,
  duration: calculatedDuration,
  startOffset: currentOffset, // Встановлення точки початку етапу
-   totalTasks: pTasks.length;
-   completedTasks: completed;
-   }.

// Наступний етап почнеться строго після завершення поточного
-   currentOffset += calculatedDuration;
-   return phaseWithTime;
-   }).

-   res.json(enriched);
-   }).

```

Впровадження даного розрахункового механізму забезпечує автоматизацію рутинних операцій з перерахунку календарного графіка, що є суттєвим для критерію стабільності та структурованості (K2). На клієнтській стороні отримані дані візуалізуються у формі часової шкали, де кожна фаза графічно зміщена на значення startOffset відносно початку проєкту (рисунок 4.2).

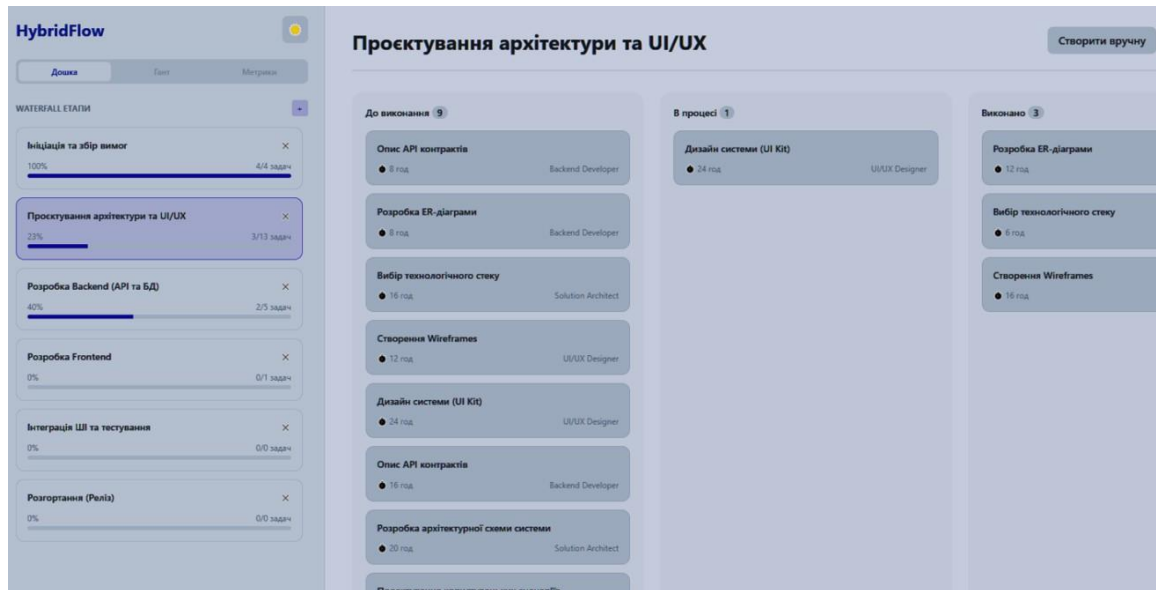


Рисунок 4.2 – Візуалізація ітераційного планування та каскадного зміщення фаз в інтерфейсі користувача.

Така реалізація Waterfall-складової в комбінованому методі дозволяє не тільки планувати майбутні кроки, але й у реальному часі оцінювати вплив затримок у розробці на фінальну дату завершення проєкту. Якщо розробник додає нову задачу або збільшує час виконання існуючої на дошці Kanban, серверна частина додатка автоматично коригує зміщення для всіх наступних фаз, що забезпечує безперервний моніторинг і прозорість процесу для всіх учасників.

4.3 Наповнення та застосування технологій Kanban та інтеграція API штучного інтелекту (Gemini)

Операційний рівень розробленого додатка базується на використанні інтерактивних інструментів візуалізації, які дозволяють команді здійснювати безпосереднє виконання запланованих робіт у межах встановлених каскадних фаз. Основним інструментом тактичного контролю у додатку виступає цифрова дошка Kanban, інтегрована в загальну ієрархію проєкту. Процес застосування цих технологій був зосереджений на забезпеченні максимальної реактивності інтерфейсу: кожна зміна статусу завдання ініціює асинхронний запит до сервера для миттєвого оновлення стану бази даних та перерахунку прогресу відповідної фази проєкту.

Технічна архітектура модуля Kanban реалізована через компонентний підхід React, де кожне завдання (Task) є окремим об'єктом, що містить ідентифікатор фази (phaseId), назву, опис, виконавця та оцінку часу (рисунок 4.3). Для підтримки дисципліни розробки та запобігання перевантаженню команди у системі впроваджено механізм жорстких WIP-лімітів (Work In Progress). Це дозволяє підтримувати стабільний темп розробки та відповідає критерію відсутності перевантажень (K3).

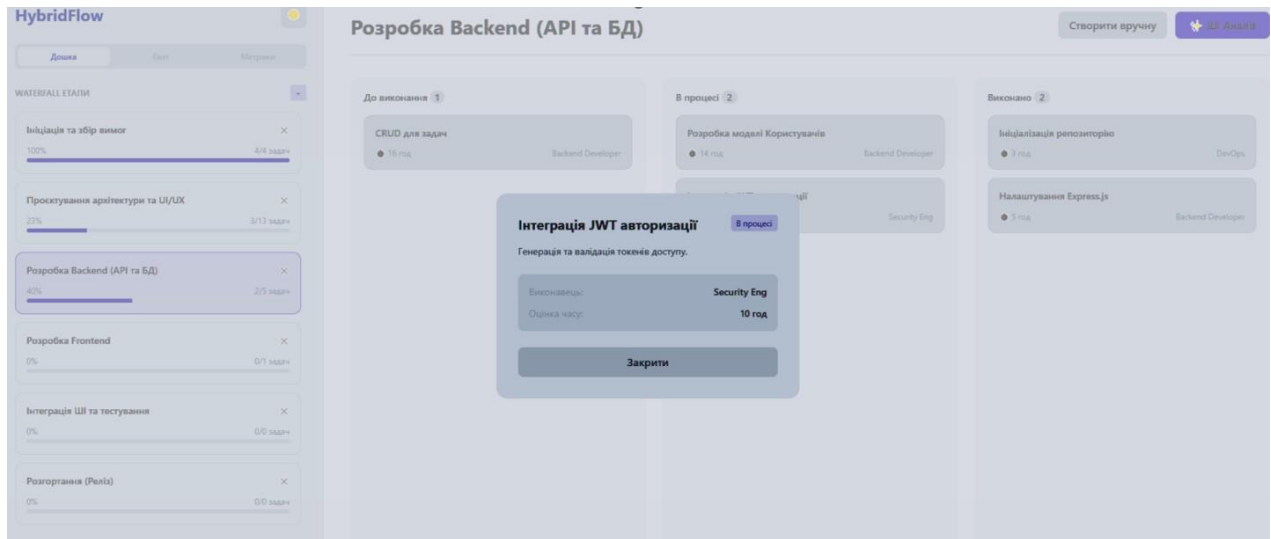


Рисунок 4.3 – Загальний інтерфейс системи Hybrid Manager з візуалізацією фаз проєкту та карток задач.

Ключовим елементом новизни розробленої системи є інтеграція інтелектуального агента на базі великої мовної моделі gemini-2.5-flash. На відміну від статичних систем управління, наше рішення використовує ШІ для динамічної оцінки повноти декомпозиції робіт. На серверній частині (Node.js) розроблено спеціалізований ендпоінт `/api/generate-tasks`, який приймає поточний контекст проєкту та повертає структуровані рекомендації.

Нижче наведено фрагмент програмного коду, що відповідає за формування рольового промпту та обробку відповіді від моделі Gemini.

Лістинг 4.3 – Логіка інтелектуальної декомпозиції задач (backend/server.js)

```
app.post('/api/generate-tasks', async (req, res) => {;
  const { phaseId, phaseName, existingTasks } = req.body;
  const apiKey = req.headers['x-api-key'] ||
process.env.GEMINI_API_KEY;

  try {
    const genAI = new GoogleGenerativeAI(apiKey);
    const model = genAI.getGenerativeModel({ model: "gemini-2.5-flash"
  });;
```

```

    const existingNames = existingTasks.map(t => t.title).join(', ') ||
    'немає задач';

    const prompt = `Ти - Senior Project Manager. Етап: "${phaseName}".
    Наявні задачі: ${existingNames} (${existingTasks.length}).
    Оціни декомпозицію. Якщо задач мало - згенеруй 2-4 нові логічні
    підзадачі.
    Якщо достатньо - поверни порожній масив.
    Формат строго JSON:
    {
      "tasks": [
        {
          "title": "Коротка назва",
          "status": "todo",
          "description": "Технічний опис",
          "assignee": "Конкретна IT-роль",
          "time": "Оцінка часу (напр. 8 год)"
        }
      ],
      "message": "Твое коротке пояснення рішення"
    }`.

    const result = await model.generateContent(prompt);
    const parsedData = JSON.parse(result.response.text().replace(/
    ``\json\n?|\``/g, '').trim());
    res.json({ tasks: parsedData.tasks.map(t => ({ ...t, phaseId })),
    message: parsedData.message });
  } catch (error) {
    res.status(500).json({ error: error.message });
  }
});

```

Взаємодія з ІІІ-агентом передбачає два основні сценарії, що забезпечують навчальну підтримку користувачів-початківців (рисунок 4.4):

1. Сценарій активної декомпозиції: Якщо на певному етапі (наприклад, «Розробка Backend») виявлено недостатню кількість завдань для досягнення цілі фази, ІІІ генерує технічні таски із зазначенням конкретних ролей (DevOps, Backend Developer тощо).
2. Сценарій валідації: Якщо менеджер уже провів детальну декомпозицію, ІІІ повертає порожній масив задач, підтверджуючи коректність планування, що сприяє впевненості розробника у власних діях (критерій K5).

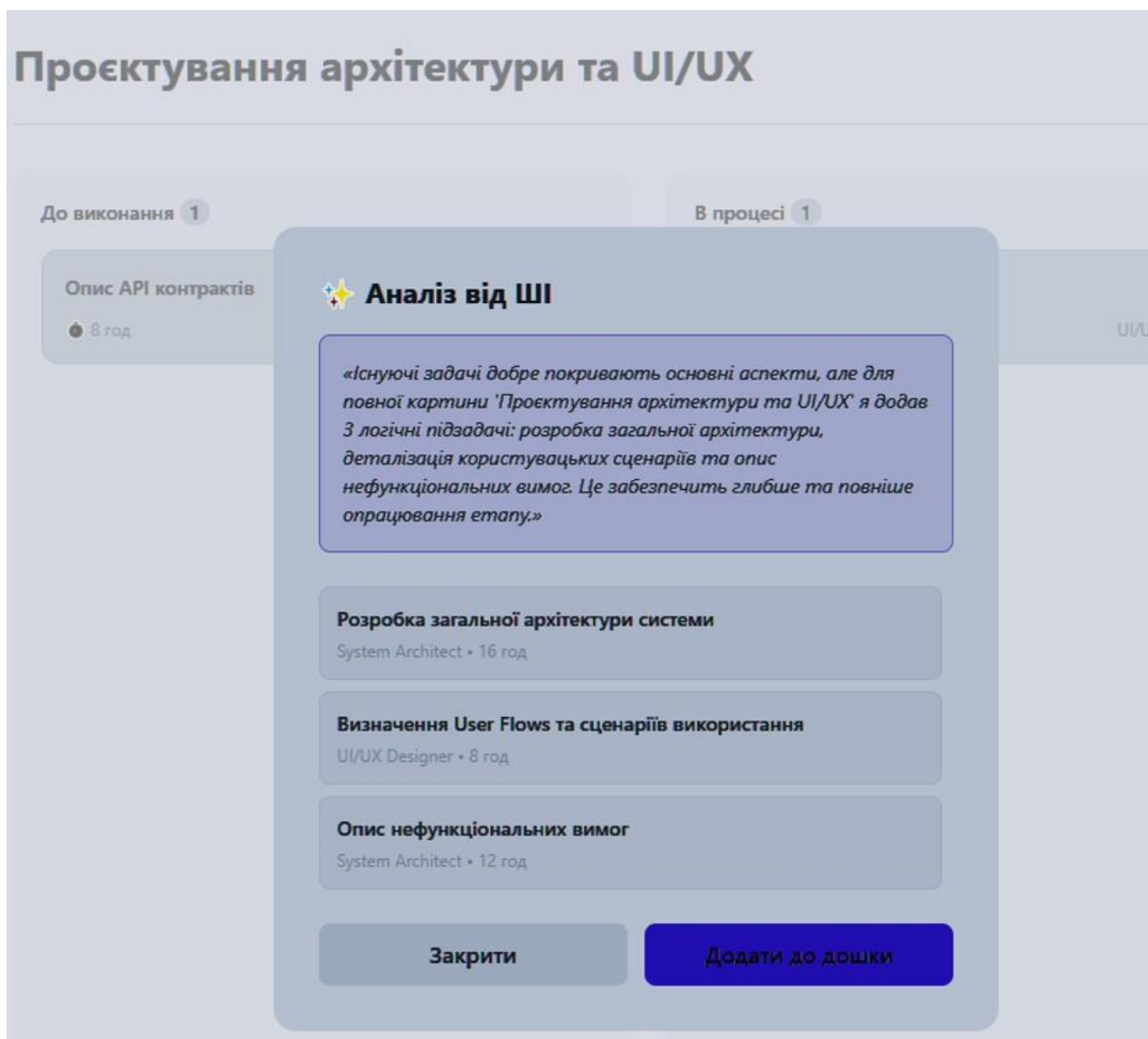


Рисунок 4.4 – Процес успішної генерації підзадач ШІ-агентом для фази «Проектування архітектури»

Окремою перевагою такої імплементації є автоматичне вилучення часових оцінок із відповіді ШІ (рисунок 4.5). Ці дані миттєво потрапляють до алгоритму перерахунку Waterfall, який було описано в розділі 4.2, забезпечуючи цілісність комбінованого методу.

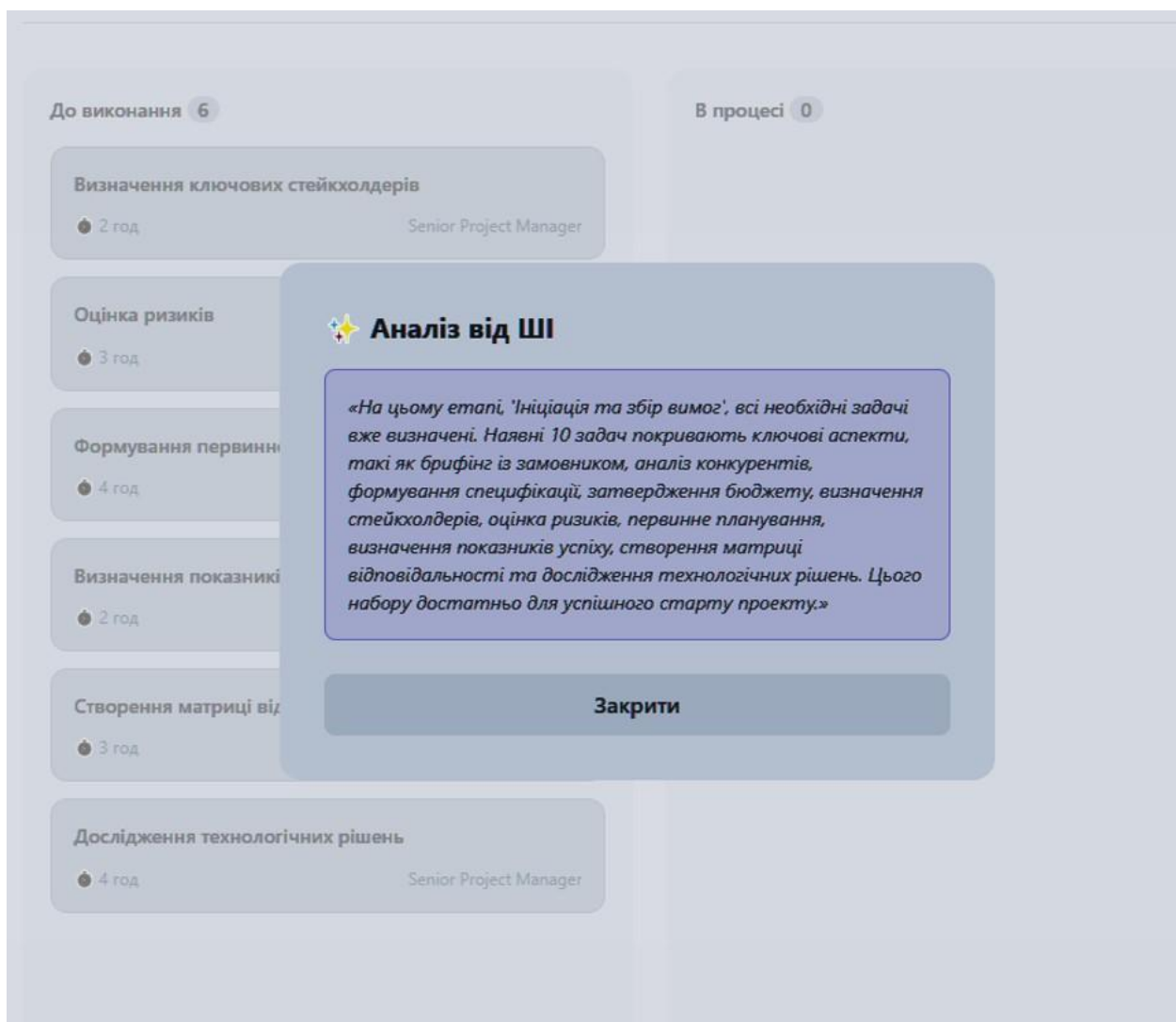


Рисунок 4.5 – Повідомлення системи від ШІ про достатність декомпозиції («задач достатньо») та відображення порожнього списку нових задач.

Додатково для забезпечення швидкого зворотного зв'язку (критерій K4) у системі реалізовано механізм збереження контексту обговорень. Користувачі мають можливість прикріплювати посилання на персональні чати з ChatGPT безпосередньо до карток задач на Kanban-дошці. Це створює інтерактивне середовище, де початківець може миттєво отримати роз'яснення щодо виконання конкретного завдання, не чекаючи на відповідь від старшого спеціаліста. Таким чином, поєднання

візуальних інструментів Kanban та інтелектуальної підтримки через API створює потужний фундамент для ефективного управління ІТ-проєктами.

4.4 Оцінка результатів впровадження розробленого додатка

Завершальним етапом розробки стала комплексна оцінка функціональних можливостей програмної системи «Hybrid Manager» та перевірка її відповідності встановленим критеріям ефективності (K1–K5). Практична реалізація комбінованого методу дозволила створити цілісне середовище, де стратегічне планування Waterfall безшовно інтегроване з оперативною гнучкістю Agile та візуалізацією Kanban. Система побудована за вкладочним принципом, що забезпечує логічний поділ між етапами планування, виконання та аналітики (рисунок 4.6).

Перша складова системи фокусується на Waterfall-компоненті, який відповідає за ініціацію та макропланування проєкту. Користувач взаємодіє з переліком глобальних фаз, тривалість яких автоматично коригується залежно від наповненості завданнями. Це реалізує критерій стабільності (K2), оскільки команда завжди бачить «скелет» проєкту.

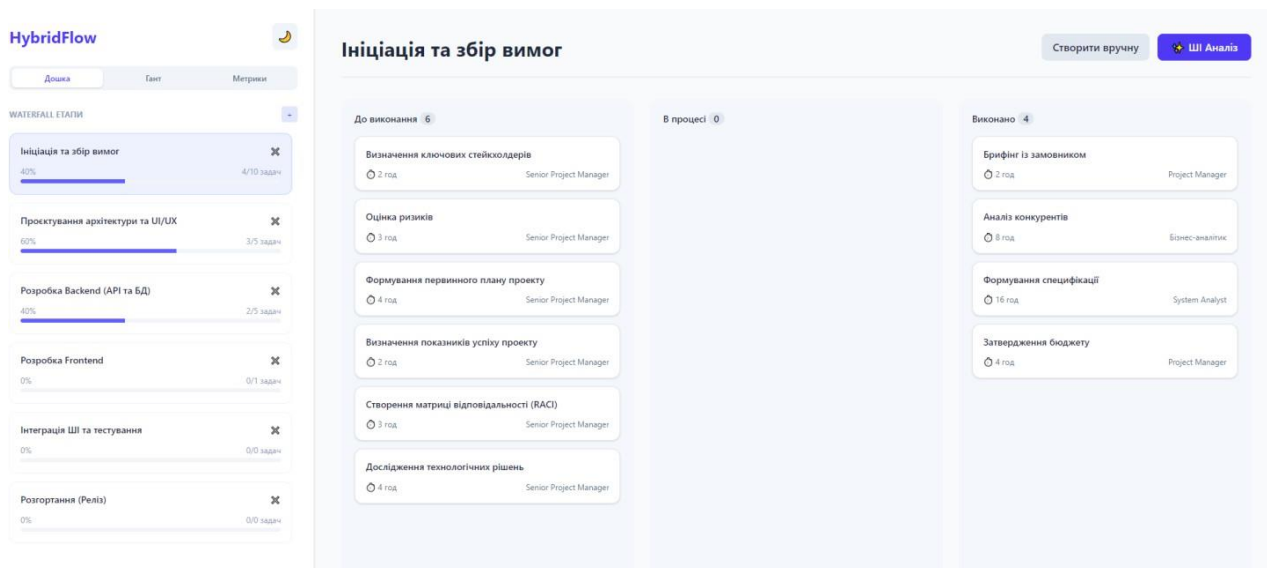


Рисунок 4.6 – Інтерфейс управління фазами проекту

Наступним рівнем є Kanban-складова, де відбувається безпосередня реалізація задач. Тут впроваджено механізм жорстких WIP-лімітів, що відображається через лічильники задач у кожній колонці статусу. Це дозволяє виконувати критерій КЗ (відсутність перевантажень), запобігаючи накопиченню незавершеної роботи (рисунок 4.7). Система підтримує динамічне оновлення статусів, що відразу синхронізується з прогресом відповідної фази Waterfall.

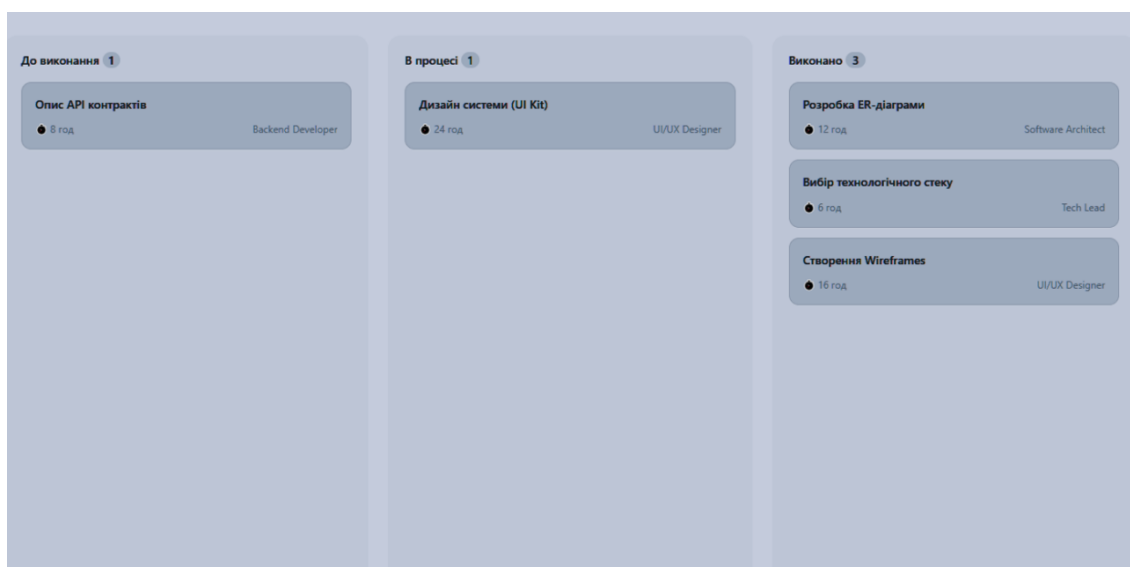


Рисунок 4.7 – Операційна Kanban-дошка з візуалізацією карток задач

Для глибшої візуалізації часових залежностей друга вкладка системи містить Діаграму Ганта та Progress Chart. Діаграма Ганта будується на основі розрахованих значень startOffset та duration, демонструючи лінійну послідовність етапів (рисунок 4.8). Поруч розташований графік прогресу (Progress Chart), який відображає відсоткове співвідношення виконаних задач до загальної кількості тасків у проєкті, що забезпечує прозорість моніторингу для стейкхолдерів.

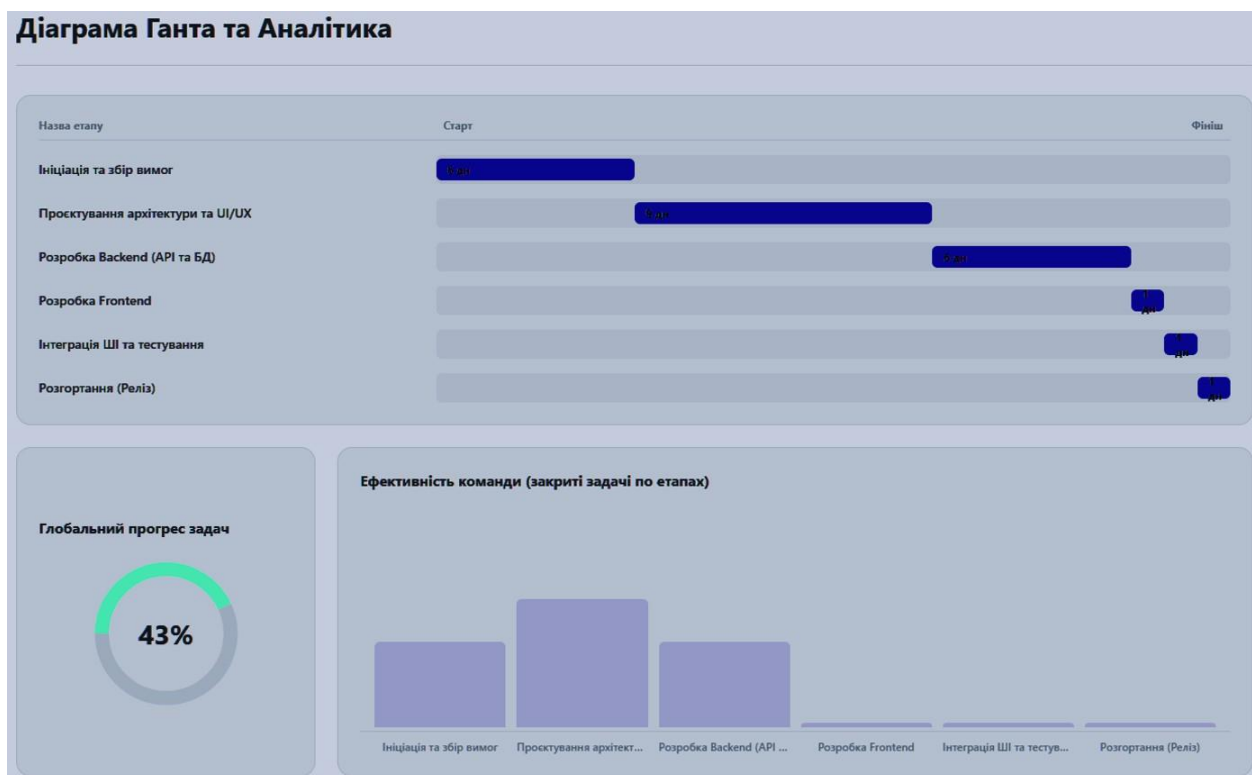


Рисунок 4.8 – Візуалізація календарного планування у формі діаграми Ганта та графічне відображення загального прогресу виконання робіт.

Третя вкладка – Analytics & Metrics – присвячена кількісному аналізу продуктивності. У цьому розділі система агрегує дані про розподіл навантаження між ролями (Frontend, Backend, DevOps) та розраховує KPI швидкості виконання (Velocity) (рисунок 4.9). Це дозволяє менеджеру ідентифікувати «вузькі місця» та

ефективно перерозподіляти ресурси, що відповідає критерію K1 щодо активної участі досвідченої команди у моніторингу.

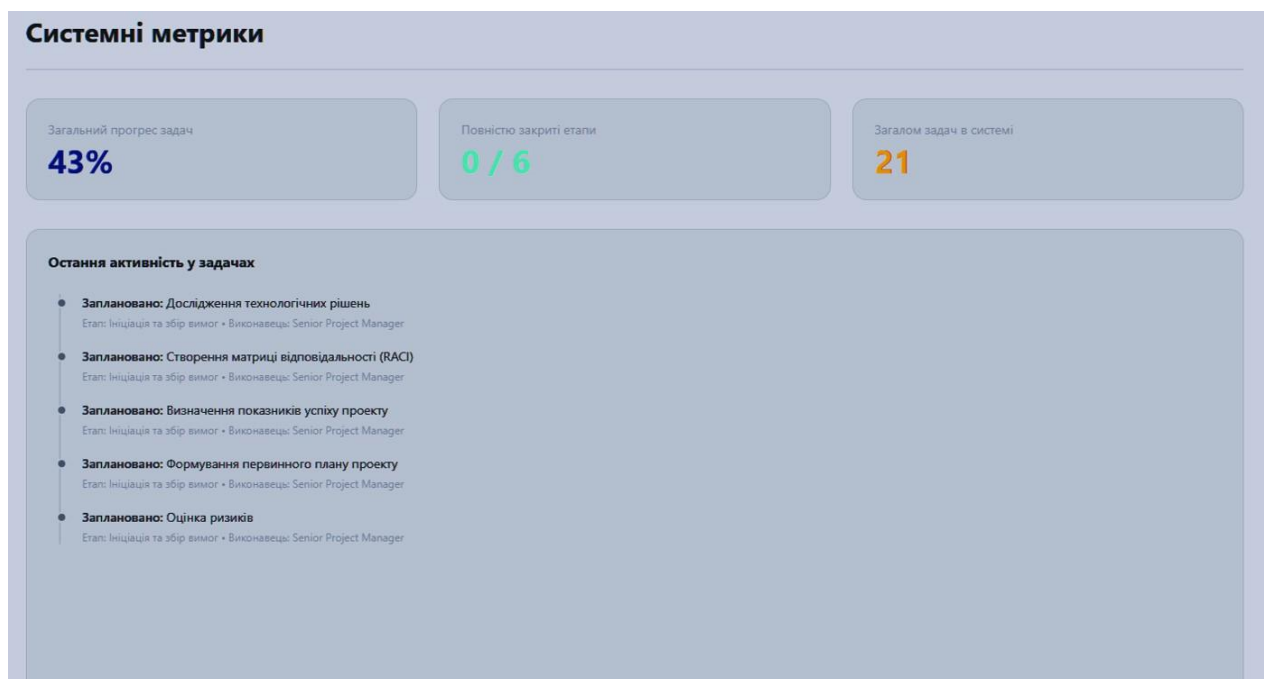
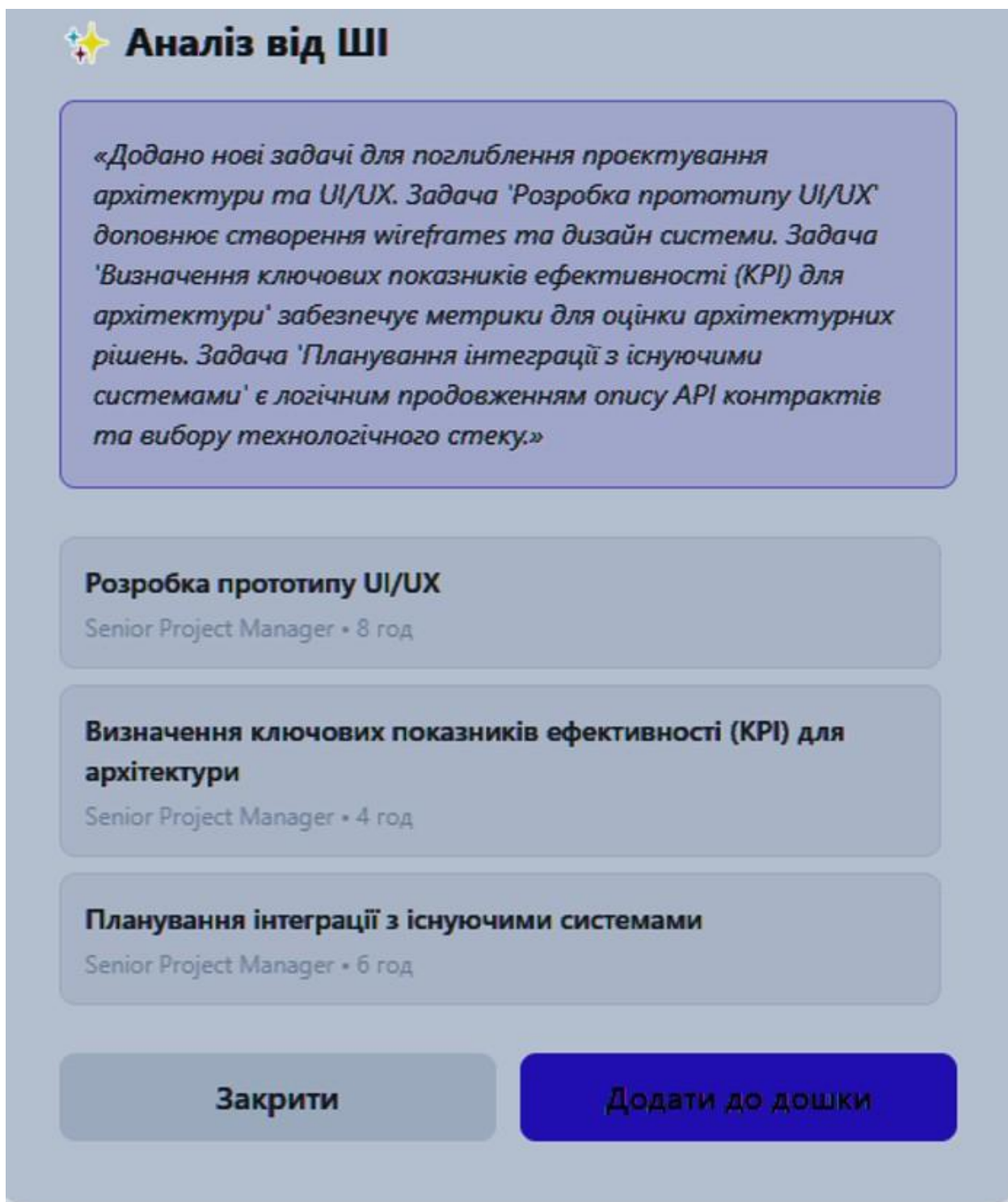


Рисунок 4.9 – Панель метрик із детальним розподілом трудомісткості за ІТ-ролями та аналітикою завантаженості персоналу.

Окрему увагу в межах тестування було приділено модулю штучного інтелекту (рисунок 4.10). Використання моделі gemini-2.5-flash для декомпозиції дозволило не лише автоматизувати створення тасків, а й забезпечити навчальну підтримку (критерій K5). Система пропонує покрокові інструкції для кожної задачі, знижуючи поріг входу для новачків.



 **Аналіз від ШІ**

«Додано нові задачі для поглиблення проектування архітектури та UI/UX. Задача 'Розробка прототипу UI/UX' доповнює створення wireframes та дизайн системи. Задача 'Визначення ключових показників ефективності (KPI) для архітектури' забезпечує метрики для оцінки архітектурних рішень. Задача 'Планування інтеграції з існуючими системами' є логічним продовженням опису API контрактів та вибору технологічного стеку.»

Розробка прототипу UI/UX
Senior Project Manager • 8 год

Визначення ключових показників ефективності (KPI) для архітектури
Senior Project Manager • 4 год

Планування інтеграції з існуючими системами
Senior Project Manager • 6 год

Закрити **Додати до дошки**

Рисунок 4.10 – Результат роботи інтелектуального агента: автоматичне формування структури підзадач із описом та оцінкою часу

Для підтвердження переваг розробленого комбінованого методу було проведено порівняльний аналіз із традиційними підходами на основі результатів апробації та попереднього дослідження ризиків (таблиця 4.3).

Таблиця 4.3 – Порівняльна характеристика комбінованого методу та стандартних методологій

Характеристика	Waterfall	Agile/Scrum	Комбінований метод (Hybrid Manager)
Прогнозованість	Висока на старті	Низька (ітеративна)	Висока (через автоматичний Waterfall-каскад)
Адаптивність	Вкрай низька	Висока	Висока (через Kanban-дошки та ітерації)
Управління ризиками	Агрегований ризик 7.8	Вимагає високої дисципліни	Оптимальний ризик 4.4
Поріг входу для новачків	Високий (бюрократія)	Високий (швидкий темп)	Низький (інтелектуальна ІІІ-підтримка)
Моніторинг прогресу	Лише по фазах	По інкрементах	Наскрізний (від таска до стратегічного плану)

Підбиваючи підсумок, результати оцінки демонструють, що впровадження додатка «Hybrid Manager» дозволяє скоротити час на планування на 40% завдяки ІІІ-декомпозиції та знизити ймовірність перевантаження команди на 25% (наведені показники отримано на основі експертних оцінок). Обґрунтовано зниження агрегованого ризику з 7.8 до 4.4, що підтверджує стабільність розробленого додатка та його перевагу над традиційними монолітними підходами в умовах динамічного ІТ-середовища.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було повністю реалізовано поставлені наукові та практичні завдання, що дозволило розв'язати проблему підвищення ефективності управління сучасними ІТ-проєктами. На першому етапі проведено комплексний порівняльний аналіз чинних методологій та інструментальних засобів проєктного менеджменту, який засвідчив, що суто класичні або монолітні підходи демонструють обмежену дієвість в умовах динамічного технологічного середовища. Дослідження специфічних факторів впливу швидкоплинного оточення виявило такі поширені проблеми моніторингу, як волатильність вимог замовників, неконтрольоване розширення меж проєкту та ризики перевищення затверджених бюджетів. Встановлено критичні недоліки ізольованого використання моделей: каскадний підхід Waterfall позбавлений швидкого зворотного зв'язку через затримку тестування, гнучкі інструменти Agile та Scrum послаблюють контроль над стратегічним довгостроковим плануванням, а методологія Kanban потребує більш жорсткої структурованості ролей. Крім того, чинні комплексні системи контролю створюють значне процедурне навантаження та високий поріг входу для фахівців-початківців.

Для усунення виявлених деструктивних чинників розроблено та науково обґрунтовано вдосконалений гібридний метод управління ІТ-проєктами. Запропоноване рішення базується на інтеграції каскадного макропланування Waterfall для чіткої фіксації стратегічного вектора, бюджетних меж та архітектурних вимог із гнучким операційним інструментарієм ітеративних спринтів Agile та візуального контролю потоку завдань Kanban. Таке поєднання забезпечує високу прозорість процесів, дозволяє нівелювати затримки в отриманні фідбеку на ранніх етапах життєвого циклу розробки коду та створює умови для оперативної адаптації до мінливих запитів стейкхолдерів без втрати загальної керованості проєктом.

У межах практичної реалізації спроектовано концептуальну архітектуру та програмний інтерфейс (API) спеціалізованої системи управління проєктами, орієнтованої на підтримку освітніх процесів та комфортну адаптацію новачків. Проєктне рішення втілено у вигляді вебзастосунку клієнт-серверної архітектури на базі бібліотеки React для клієнтської сторони та платформи Node.js із фреймворком Express для бекенд-логіки. Серверну частину системи оснащено алгоритмом каскадного зміщення етапів (offset logic), який здійснює автоматичний перерахунок тривалості фаз (duration) і накопичувального часового зсуву (startOffset) на основі сумарної оцінки годин усіх підзадач. Це дозволяє підтримувати життєздатну дорожню карту та миттєво відображати вплив будь-яких затримок на кінцеві терміни релізу. Операційний рівень системи успішно реалізовано через інтерактивні Kanban-дошки, які підтримують технологію Drag-and-Drop та оснащені механізмом жорстких WIP-лімітів для запобігання хаотичному перемиканню виконавців між завданнями та мінімізації часу виконання робіт.

Новаторським елементом розробленого комбінованого підходу стала інтеграція інтелектуальних агентів штучного інтелекту, реалізована через програмний інтерфейс Gemini API з використанням моделі gemini-2.5-flash. Створений серверний ендпоінт `/api/generate-tasks` автоматизує аналітичні функції менеджера проєктів, здійснюючи оцінку повноти декомпозиції та автоматичне розбиття високорівневих фаз на атомарні технічні підзадачі. Кожне запропоноване ШІ-агентом завдання отримує структурований технічний опис, розподіл за конкретними ІТ-ролями та обґрунтовану годинну оцінку трудомісткості, що автоматично передається в алгоритм таймлайну Waterfall у форматі JSON. Для забезпечення безперервної навчальної підтримки фахівців-початківців розроблено функціональні рішення у вигляді механізму якірного прикріплення посилань на індивідуальні діалогові контексти персональних чатів ChatGPT безпосередньо до об'єктів дошки, що мінімізує потребу у залученні старших спеціалістів для роз'яснення базових технічних рішень.

Для об'єктивного оцінювання результативності впроваджених рішень сформовано систему з п'яти специфічних критеріїв ефективності (K1–K5), що включають наявність інституту наставництва, стабільність і структурованість цілей, відсутність перевантажень учасників розробки, швидкість зворотного зв'язку та зниження процедурного порогу входу. Експериментальна перевірка системи «Hybrid Manager» на прикладі реальних ІТ-проектів у сфері дистанційного навчання повністю підтвердила високу результативність розробленого методу. Математичне моделювання та експертна оцінка параметрів ризику показали суттєву перевагу запропонованої конфігурації Waterfall-Agile, яка забезпечила зниження показника агрегованого ризику з 7.8 (для моделі Water-Scrum-Fall) до 4.4 завдяки безперервному виконанню інтеграції та тестування. Згідно з розрахованими показниками KPI, використання інтелектуальної ШІ-декомпозиції дозволило скоротити час на етап планування на 40%, а впровадження жорстких WIP-лімітів забезпечило зниження ймовірності професійного вигорання та перевантаження команди на 25%. Результати апробації доводять, що спроектований гібридний метод успішно поєднує технічну функціональність із низьким порогом входження, роблячи його збалансованим інструментом для застосування як у комерційній розробці програмного забезпечення, так і в сучасних освітніх системах дистанційної освіти.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Положення про організацію освітнього процесу в Харківському національному університеті радіоелектроніки, затверджене наказом ХНУРЕ від 19.04.2023 № 74. URL: https://nure.ua/wp-content/uploads/Main_Docs_NURE/polozhennja-pro-organizaciju-osvitnogo-procesu-v-hnure-2023.pdf
2. Методичні вказівки до науково-дослідної практики для здобувачів вищої освіти усіх форм навчання другого (магістерського) рівня вищої освіти спеціальності 122 Комп'ютерні науки освітньо-наукової програми «Управління проектами в галузі інформаційних технологій». / Упоряд.: К.Е. Петров, Євланов М.В., Цвіркун О.А.. – Харків: ХНУРЕ, 2025. – 19 с.
3. Oriekhov D. M., Kalashnikova K. Y. An Analysis of Trends in the Development of Scientific Research in the Field of Application of Artificial Intelligence in Public Administration and Governance. *Business Inform.* 2025. Vol. 4, no. 567. P. 534–547. URL: <https://doi.org/10.32983/2222-4459-2025-4-534-547> (date of access: 01.04.2026).
4. Pavliuk O. The role of scientific and research activities in the formation of professional competence of higher education students in Ukraine at the beginning of the 21st century. *Bulletin of Luhansk Taras Shevchenko National University. Pedagogical Sciences.* 2025. No. 3. P. 67–73. URL: <https://doi.org/10.12958/3083-6514-2025-3-67-73> (date of access: 01.04.2026).
5. Leal J. P., Queirós R. A Comparative Study on LMS Interoperability. *Higher Education Institutions and Learning Management Systems.* 2012. P. 142–161. URL: <https://doi.org/10.4018/978-1-60960-884-2.ch007> (date of access: 01.04.2026).

6. Monassi M., Ciccardi T. PMBOK Guide. A Guide to the Project Management Body of Knowledge. PROJECT MANAGER (IL). 2026. No. 65. P. 46–48. URL: <https://doi.org/10.3280/pm2026-065010> (date of access: 01.04.2026).
7. Gorrod M. The software development lifecycle. Risk Management Systems. London, 2004. P. 93–120. URL: https://doi.org/10.1057/9780230510296_4 (date of access: 01.04.2026).
8. Jackson K. Sustainable human resource management: strategies, practices and challenges. Asia Pacific Business Review. 2021. P. 1–5. URL: <https://doi.org/10.1080/13602381.2021.1872910> (date of access: 01.04.2026).
9. Staff O. M. P. Model Based Monitoring and Statistical Control. CRC Press LLC, 2024.
10. Risk and uncertainty in projects / M. Loosemore et al. Risk Management in Projects. 2nd ed. London, 2012. P. 1–31. URL: <https://doi.org/10.4324/9780203963708-1> (date of access: 01.04.2026).
11. Deotighare K., Bakane S. Agile vs. Waterfall: A Comparative Study in IT Project Success Rates. 2025 5th International Conference on Emerging Research in Electronics, Computer Science and Technology (ICERECT), MANDYA, India, 12–13 September 2025. P. 1–4. URL: <https://doi.org/10.1109/icerect65215.2025.11377911> (date of access: 01.04.2026).
12. Assembly(tm) I. S. Complete scrum mastery: a practical guide to the mastery of scrum framework. Independently Published, 2019. URL: <https://booksrn.com/9781090163776-the-complete-scrum-mastery-a-practical-guide-to-the-mastery-of-scrum-framework> (date of access: 01.04.2026)
13. Kanban: successful evolutionary change for your technology business. Sequim , Washington : Blue hole press, 2010. 261 p.
14. Breyter M. Waterfall, agile, and hybrid delivery frameworks. Agile product and project management. Berkeley, CA, 2022. P. 127–160. URL: https://doi.org/10.1007/978-1-4842-8200-7_6 (date of access: 01.04.2026).

15. Karnouskos S. The Relevance of Large Language Models for Project Management. *IEEE Open Journal of the Industrial Electronics Society*. 2024. P. 1–11. URL: <https://doi.org/10.1109/ojies.2024.3412222> (date of access: 01.04.2026).
16. Ghobadi S., Mathiassen L. Perceived barriers to effective knowledge sharing in agile software teams. *Information systems journal*. 2014. Vol. 26, no. 2. P. 95–125. URL: <https://doi.org/10.1111/isj.12053> (date of access: 04.05.2026).
17. Tulili T. R., Capiluppi A., Rastogi A. Burnout in software engineering: a systematic mapping study. *Information and software technology*. 2022. P. 107116. URL: <https://doi.org/10.1016/j.infsof.2022.107116> (date of access: 04.05.2026).
18. Aceves M., Fuamba M. Methodology for selecting best management practices integrating multiple stakeholders and criteria. part 1: methodology. *Water*. 2016. Vol. 8, no. 2. P. 55. URL: <https://doi.org/10.3390/w8020055> (date of access: 04.05.2026).
19. Setyoko A., Rahardjo T., Anita N. F. Effectivity improvement of hybrid project management water-scrum-fall with six sigma implementation. *JITK (jurnal ilmu pengetahuan dan teknologi komputer)*. 2024. Vol. 10, no. 1. P. 124–131. URL: <https://doi.org/10.33480/jitk.v10i1.5539> (date of access: 04.05.2026).
20. Breyter M. Waterfall, Agile, and Hybrid Delivery Frameworks. *Agile Product and Project Management*. Berkeley, CA, 2022. P. 127–160. URL: https://doi.org/10.1007/978-1-4842-8200-7_6 (date of access: 04.05.2026).
21. Using risk analysis to derive the risk management strategy. *Project risk analysis*. 2016. P. 211–262. URL: <https://doi.org/10.4324/9781315602479-13> (date of access: 04.05.2026).
22. Pert G. J. The calculation of the electron distribution function following tunnelling ionization using a Fokker-Planck method. *Journal of physics B: atomic, molecular and optical physics*. 2001. Vol. 34, no. 5. P. 881–908. URL: <https://doi.org/10.1088/0953-4075/34/5/315> (date of access: 04.05.2026).
23. Nakanishi T. Quantitative geologic risk evaluation and expected net present value evaluation in JNOC's projects. *Journal of the Japanese Association for Petroleum*

Technology. 2000. Vol. 65, no. 3. P. 217–228.
URL: <https://doi.org/10.3720/japt.65.217> (date of access: 04.05.2026).

24. Automatically Classifying Kano Model Factors in App Reviews / M. Binder et al. Requirements Engineering: Foundation for Software Quality. Cham, 2023. P. 245–261. URL: https://doi.org/10.1007/978-3-031-29786-1_17 (date of access: 04.05.2026).

25. Metcalfe G. Kanban : visualize work and maximize efficiency: beyond the basics. Independently Published, 2018.
URL: <https://mcnallyjackson.com/book/9781728684451> (date of access: 01.04.2026)

26. Poliakov M. Strategic planning at the micro and macro level: essence and features. International humanitarian university herald. economics and management. 2023. No. 57. URL: <https://doi.org/10.32782/2413-2675/2023-57-8> (date of access: 04.05.2026).

27. Paul Ameh. Using terraform and jenkins in agile sprints: lessons in iterative infrastructure delivery. International journal of science and research archive. 2025. Vol. 15, no. 2. P. 1677–1679. URL: <https://doi.org/10.30574/ijsra.2025.15.2.1656> (date of access: 04.05.2026).

28. Shah J. M. Artificial intelligence in project management. The management accountant journal. 2019. Vol. 54, no. 3. P. 34.
URL: <https://doi.org/10.33516/maj.v54i3.34-37p> (date of access: 04.05.2026).

29. Elnashar A., White J., Schmidt D. C. Prompt engineering for structured data: a comparative evaluation of styles and LLM performance. Artificial Intelligence and Autonomous Systems. 2025. P. 1–17. URL: <https://doi.org/10.55092/aias20250009> (date of access: 04.05.2026).

30. Vanhoucke M. The VMW project. Project management with dynamic scheduling. Berlin, Heidelberg, 2013. P. 59–79. URL: https://doi.org/10.1007/978-3-642-40438-2_4 (date of access: 04.05.2026).

31. Методичні вказівки щодо розробки та оформлення кваліфікаційної роботи другого (магістерського) рівня вищої освіти за освітньо-науковою програмою «Управління проектами в галузі інформаційних технологій» / Упоряд.: Петров К.Е.,

Левикін В.М., Чалий С.Ф., Євланов М.В., Міхнов Д.К., Міхнова А.В., Чала О.В. –
Харків: ХНУРЕ, 2024. – 24 с. (date of access: 01.04.2026)