

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 моделей і методів управління роботами зі створення
допоміжної документації в IT-проекті розробки платформи генерації
цифрового контенту
(тема)

Виконав:
здобувач _____ 2 _____ року навчання,
групи _____ УПГІТм-24-1
Маргарита Д'ЯЧЕНКО
(власне ім'я, прізвище)

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

Тип програми _____ освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма _____ Управління проектами в
галузі IT
(повна назва освітньої програми)

Керівник: _____ проф. каф. ІУС Максим Євланов
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри ІУС



(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)

«06» _____ квітня 20 26 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві _____ Д'яченко Маргариті Володимирівні _____
 (прізвище, ім'я, по батькові)

1. Тема роботи Дослідження моделей і методів управління роботами зі створення допоміжної документації в IT-проекті розробки платформи генерації цифрового контенту

затверджена наказом по університету від « 23 » березня 2026 р. № 231Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії « 19 » травня 2026 р.

3. Вихідні дані до роботи: науково-технічна література, публікації та інтернет-ресурси, що стосуються теми кваліфікаційної роботи, опис процесу документації.

4. Перелік питань, що потрібно опрацювати у роботі: аналіз специфіки процесу створення допоміжної документації в IT-проектах, аналіз існуючих моделей і методів організації робіт зі створення документації, класифікація підходів до автоматизації документування із використанням ШІ, розробка структурної та процесної моделі створення допоміжної документації в IT-проектах, дослідження особливостей реалізації запропонованого методу, апробація моделей і методів управління процесом створення допоміжної документації.

КАЛЕНДАРНИЙ ПЛАН

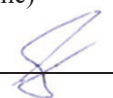
№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз специфіки управління процесом створення допоміжної документації в ІТ-проектах	23.03.2026 – 25.03.2026	Виконано
2	Класифікація та аналіз робіт зі створення допоміжної документації в ІТ-проекті	26.03.2026 – 28.03.2026	Виконано
3	Дослідження існуючих моделей організації процесу документування	01.04.2026 – 04.04.2026	Виконано
4	Розробка моделі розподілу задач між людиною та штучним інтелектом на основі індексу придатності до автоматизації	05.04.2026 – 11.04.2026	Виконано
5	Розробка математичних моделей часових та фінансових витрат процесу документування	13.04.2026 – 17.04.2026	Виконано
6	Дослідження особливостей реалізації технології створення документації в ІТ-проекті розробки платформи генерації цифрового контенту	18.04.2026 – 20.04.2026	Виконано
7	Реалізація етапів технології документування	21.04.2026 – 25.04.2026	Виконано
8	Проведення апробації розробленого методу на прикладі реального ІТ-проекту розробки платформи генерації цифрового контенту	26.04.2026 - 29.04.2026	Виконано
9	Оформлення пояснювальної записки	01.05.2026 – 04.05.2026	Виконано
10	Підготовка презентаційного матеріалу	10.05.2026	Виконано
11	Перевірка роботи на плагіат	14.05.2026	Виконано
12	Захист кваліфікаційної роботи	20.05.2026	Виконано

Дата видачі завдання 06 квітня 2026 р.

Здобувач


(підпис)

Керівник роботи


(підпис)

проф. каф. ІУС Максим ЄВЛАНОВ

(посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 94 с., 11 рис., 18 табл., 1 дод., 21 джерело.

ДОКУМЕНТАЦІЯ, ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ, ІТ-ПРОЄКТ, МЕТОД, ОПТИМІЗАЦІЯ, РЕСУРСИ, СТРУКТУРА МОДЕЛІ, УЧАСНИК ПРОЄКТУ.

Об'єктом дослідження кваліфікаційної роботи є процеси створення допоміжної документації в ІТ-проектах розробки програмного забезпечення.

Предметом дослідження є моделі та методи управління роботами зі створення допоміжної документації в ІТ-проектах з використанням штучного інтелекту.

Метою кваліфікаційної роботи є розробка, модифікація та дослідження моделей і методів управління роботами зі створення допоміжної документації в ІТ-проектах, а також розробка методу оптимізації витрат ресурсів проекту на прикладі ІТ-проекту розробки платформи генерації цифрового контенту.

Методи дослідження: порівняльний аналіз, попарних порівнянь, формалізація.

Основними результатами кваліфікаційної роботи є розробка методу оптимізації витрат ресурсів при управлінні роботами зі створення допоміжної документації в ІТ-проекті розробки платформи генерації цифрового контенту.

Проведено дослідження особливостей реалізації розроблених моделей та методів.

Проведена апробація розробленого методу оптимізації ресурсів, яка підтвердила його ефективність при створенні документації в ІТ-проекті розробки платформи генерації цифрового контенту.

ABSTRACT

Master`s thesis: 94 pages, 11 figures, 18 tables, 1 appendice, 21 sources.

DOCUMENTATION, INFORMATION TECHNOLOGY, IT PROJECT, METHOD, OPTIMIZATION, PROJECT PARTICIPANT, RESOURCES, STRUCTURE OF THE MODEL.

The object of the qualification work is the processes of creating supporting documentation in IT projects of software development.

The subject of the research is models and methods of managing work on creating supporting documentation in IT projects using artificial intelligence.

The purpose of the qualification work is to develop, modify and study models and methods of managing work on creating supporting documentation in IT projects, as well as to develop a method for optimizing project resource costs using the example of an IT project for developing a digital content generation platform.

Research methods: comparative analysis, pairwise comparisons, formalization.

The main results of the qualification work are the development of a method for optimizing resource costs when managing work on creating supporting documentation in an IT project for developing a digital content generation platform.

A study of the implementation features of the developed models and methods was conducted.

The developed resource optimization method was tested, which confirmed its effectiveness in creating documentation in an IT project for developing a digital content generation platform.

ЗМІСТ

	С.
Скорочення та умовні позначки	8
Вступ	9
1 Аналіз предметної області та постановка задачі дослідження	11
1.1 Огляд та загальний аналіз проблеми створення допоміжної документації в ІТ-проєктах	11
1.2 Аналіз існуючих підходів до організації процесу створення допоміжної документації	13
1.3 Постановка задачі дослідження	18
2 Розробка моделей і методів управління роботами зі створення допоміжної документації	21
2.1 Розробка структурної та функціональної моделі процесу створення допоміжної документації	21
2.2 Розробка моделі розподілу задач між людиною та штучним інтелектом у процесі створення документації	32
2.3 Розробка методу оптимізації витрат ресурсів у процесі створення документації	43
3 Дослідження особливостей реалізації технології створення документації в іт-проєкті розробки платформи генерації цифрового контенту	53
3.1 Реалізація технології створення допоміжної документації в ІТ-проєкті розробки платформи генерації цифрового контенту	53
3.2 Реалізація етапу «Збір та аналіз вихідних даних»	55
3.3 Реалізація етапу «Планування структури документації»	58
3.4 Реалізація етапу «Генерація чернетки документації»	60
3.5 Реалізація етапу «Рецензування та доопрацювання документації»	62
3.6 Реалізація етапу «Публікація документації»	67
3.7 Висновки з реалізації технології створення допоміжної документації ..	69

4 Апробація результатів дослідження на прикладі іт-проекту розробки платформи генерації цифрового контенту.....	70
4.1 Підготовка даних для апробації	70
4.2 Результат апробації розробленого методу ефективності створення документації	72
Висновки	77
Перелік джерел посилання	79
Додаток А. Графічний матеріал кваліфікаційної роботи.....	82

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІС – інформаційна система

ІТ – інформаційна технологія

ПЗ – програмне забезпечення

ШІ – штучний інтелект

AIC – Average Iteration Count

API – application Programming Interface

ASI – Automation Suitability Index

CI/CD – Continuous Integration/Continuous Deployment)

FB – Feedback

FPAR – First-Pass Iteration Acceptance Rate

ISO – International Organization for Standardization

LLM service – Large Language Model as-a-Service

NLP – Natural Language Processing

SDK – Software Development Kit

TSR – Time Saving Ratio

PM – Менеджер проектів

UAT – User Acceptance Testing

ВСТУП

Проекти, що існують в галузі інформаційних технологій, як форма проєктної діяльності, вирізняються високим рівнем динаміки, залежністю від людського ресурсу та постійною необхідністю створення та підтримки актуальної технічної документації. У такому середовищі документування є невід'ємною складовою успішної реалізації ІТ-проєкту. Несвоєчасне або неякісне документування призводить до порушення термінів, збільшення витрат, зростання кількості помилок та втрати критично важливих знань про систему.

У кваліфікаційній роботі досліджуються моделі та методи управління роботами зі створення допоміжної документації в ІТ-проєктах з використанням штучного інтелекту (ШІ).

Управління процесом створення допоміжної документації у кваліфікаційній роботі визначається як процес, що дозволяє зібрати вихідні дані про компоненти системи, визначити оптимальний розподіл задач між учасниками проєкту та ШІ, генерувати та валідувати документацію, а також сформувати базу знань, яку менеджер ІТ-проєкту може використовувати для прийняття рішень щодо планування та оптимізації процесу документування у майбутніх проєктах.

При проведенні досліджень розробляється метод управління роботами зі створення допоміжної документації, що заснований на поєднанні структурно-функціональної моделі процесу документування, моделі розподілу задач між людиною та ШІ на основі індексу придатності до автоматизації (Automation Stability Index, ASI), та математичної моделі оптимізації витрат ресурсів. Використання такого методу дозволяє не лише автоматизувати рутинні задачі документування, але й комплексно управляти процесом із урахуванням критичності кожної задачі, навантаження учасників та бюджетних обмежень проєкту.

Результати дослідження сприяють прийняттю обґрунтованих управлінських рішень, спрямованих на скорочення витрат часу та коштів на документування при збереженні прийнятного рівня якості.

Метод, що розробляється, дозволяє не лише оптимізувати поточний процес документування, а й накопичувати дані для формування системи управління документацією у майбутніх проєктах та адаптуватись до специфіки різних типів ІТ-команд.

Метою кваліфікаційної роботи є дослідження моделей і методів управління роботами зі створення допоміжної документації в ІТ-проєктах, розробка методу оптимізації розподілу задач між людиною та штучним інтелектом, а також дослідження особливостей реалізації розробленої технології на прикладі ІТ-проєкту розробки платформи генерації цифрового контенту.

Для досягнення поставленої мети у кваліфікаційній роботі пропонується виконати наступні задачі дослідження:

- провести аналіз існуючих моделей і підходів до організації процесу створення допоміжної документації в ІТ-проєктах та визначити їх обмеження;
- розробити структурну та функціональну модель процесу створення допоміжної документації із визначенням етапів, станів, учасників та інформаційних потоків;
- розробити модель розподілу задач між людиною та штучним інтелектом на основі індексу придатності до автоматизації ASI;
- розробити метод оптимізації витрат ресурсів у процесі створення документації та сформулювати задачу оптимізації;
- дослідити особливості реалізації розробленої технології в ІТ-проєкті розробки платформи генерації цифрового контенту;
- провести апробацію розробленого методу на прикладі реального ІТ-проєкту та оцінити його ефективність.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Огляд та загальний аналіз проблеми створення допоміжної документації в ІТ-проєктах

Сучасні ІТ-проєкти, особливо ті, що спрямовані на створення цифрового контенту засобами ШІ, характеризуються високою складністю, динамічнішими змінами та великою кількістю учасників. У таких умовах важливу роль відіграє допоміжна документація, яка забезпечує ефективну взаємодію між членами команди, спрощує підтримку програмного забезпечення та підвищує якість кінцевого продукту.

Допоміжна документація в ІТ-проєктах охоплює широкий спектр матеріалів, таких як технічний опис системи, документація користувача, інструкції з тестування, описи Application Programming Interface (API), а також внутрішня документація команди, наприкладі результатів зустрічей та технічні нотатки, див.рис 1.1 [1].

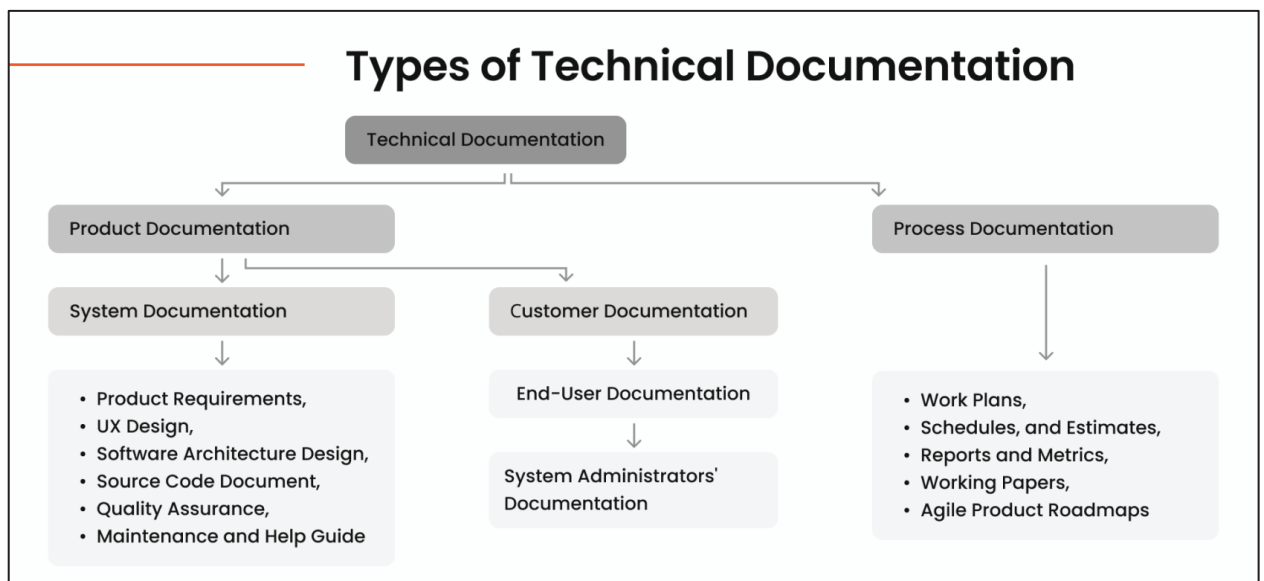


Рисунок 1.1 – Види технічної документації в ІТ-проєктах

У рамках традиційних підходів до організації процесу створення документації, її написання покладається безпосередньо на розробників або тестувальників. З одного боку, залучення фахівців, це ефективно, оскільки вони розроблюють функціональність продукту, краще розуміють його внутрішню логіку. Однак на практиці це призводить до суттєвих недоліків, бо фахівці присвячують значну частину свого робочого часу на виконання задач, які не є їхньою основною діяльністю. Це знижує загальну продуктивність команди та призводить до неефективного використання ресурсів.

Крім того, процес створення документації часто не має чіткої структури та стандартів. Документи можуть створюватися нерегулярно, з різними форматами та деталями щодо їх використання, що може ускладнити їх подальше використання. У більшості випадків документація застаріває швидше, ніж оновлюється, що робить її функціональність менш важливою.

Ще однією проблемою є відсутність ефективного розподілу ролей. Відповідаючі за документацію не розподілені на окремі ролі в більшості ІТ-команд або розподіляються частково та несистематично. Як наслідок, виникає перевантаження ресурсів ключових учасників проєкту – розробників та технічних керівників.

В епоху розвитку інформаційних технологій розробляються нові методи їх вирішення, на прикладі застосування інструментів автоматизації та технологій ШІ. Використання такого типу технологій відкриває перспективи для часткової або повної автоматизації процесу створення допоміжної документації.

Результати робочих зустрічей можна обробляти за допомогою ШІ для генерації технічних описів залежно від вихідного коду та формування чернеток документації на основі необхідних шаблонів. Роль людини в цьому випадку також змінюється: замість розробки документації як такої, фахівці в цьому випадку виконують функції перевірки, перегляду та затвердження матеріалів.

Впровадження інтелектуальних інструментів у процес створення документації допоможуть скоротити витрати часу розробників, підвищити

стандартизацію документів та забезпечити їх актуальність. Разом з тим, ефективність такого підходу значною мірою залежить від правильної організації процесу управління роботами зі створення документації, зокрема розподілу ролей між учасниками, визначення етапів процесу та впровадження засобів контролю якості.

Відповідно, існує об'єктивна необхідність у дослідженні моделей і методів управління роботами зі створення допоміжної документації в ІТ-проєктах з урахуванням сучасних технологічних можливостей, включаючи застосування ІІІ, та можливостей оптимізації використання ресурсів та підвищення ефективності роботи команди. При цьому використання ІІІ потребує формалізацію процесу взаємодії між людиною та інтелектуальною системою, що обумовлює необхідність розробки відповідних моделей управління.

1.2 Аналіз існуючих підходів до організації процесу створення допоміжної документації

Процес створення допоміжної документації в ІТ-проєктах є складною багатокомпонентною задачею, що включає не лише безпосереднє формування текстових матеріалів, але й організацію взаємодії між учасниками проєкту, управління знаннями та забезпечення актуальності інформації. У сучасній практиці відсутній єдиний універсальний підхід до організації цього процесу, що обумовлює необхідність аналізу існуючих моделей з метою виявлення їх переваг та обмежень.

Залежно від організаційної структури команди, масштабу проєкту та рівня готовності процесів, у практиці розробки програмного забезпечення застосовуються різні підходи до створення документації, котрі відображені на рисунку 1.2, роздивимось детальніше:

- документація, створена розробниками;
- залучення технічних письменників;
- візуальне та Collaborative-середовище;
- Agile-підхід;
- використання ШІ.

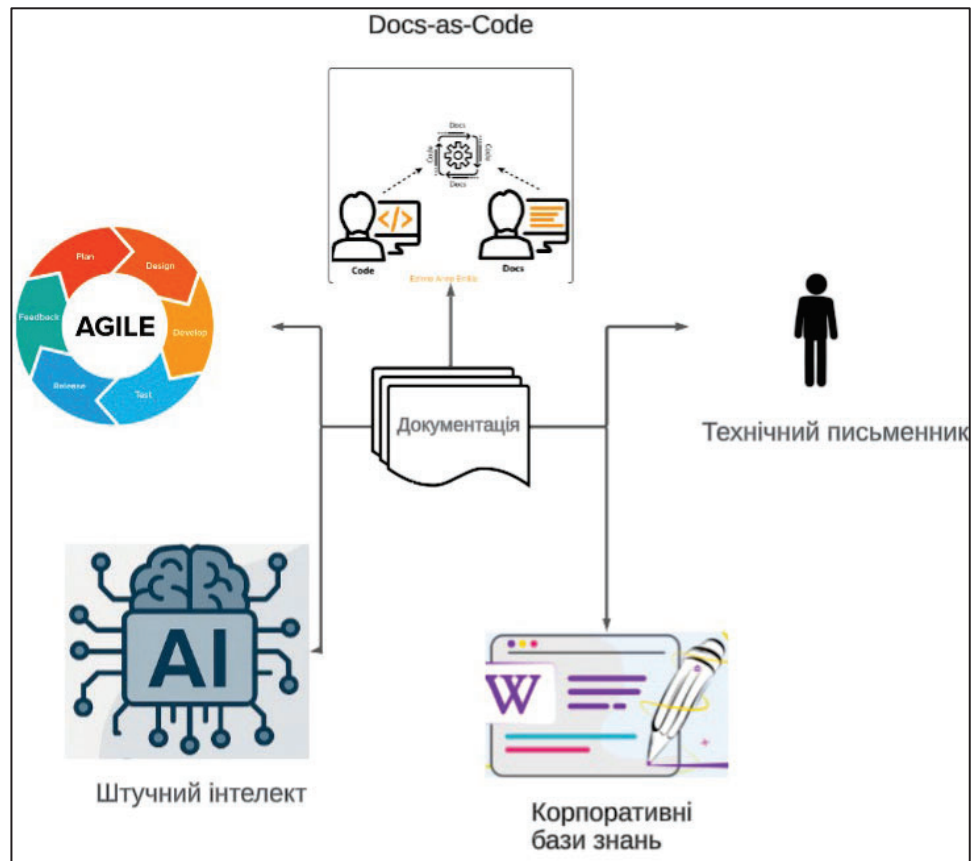


Рисунок 1.2 – Популярні існуючі підходи до організації процесу створення документації

Документація, створена розробниками (Docs-as-Code), створюється або паралельно з розробкою, або після завершення окремих етапів проєкту, формуючись як і код, де використовується розмітка, системи контролю версій (Git) та автоматизована перевірка (CI/CD). До основних недоліків можна віднести неефективне використання висококваліфікованих ресурсів. Розробники, вартість роботи яких є значно вищою порівняно з іншими

спеціалістами, витрачають значну частину часу на виконання задач, що не створюють прямої доданої вартості у вигляді функціоналу [2].

Крім того, документація часто створюється за залишковим принципом, коли основна увага приділяється реалізації функціональних вимог. Це призводить до нерегулярності створення документації, її неповноти або низької якості.

Процес документування завдяки технічним письменникам відокремлюється від процесу розробки, що дозволяє більш чітко розподілити ролі та відповідальність. Технічні письменники, як правило, мають навички структурованого викладу інформації, що забезпечує високу якість документації. Вони використовують стандартизовані підходи, шаблони та методики, що сприяє уніфікації документації в межах проєкту.

Однак ефективність цього підходу значною мірою залежить від організації процесу передачі знань. Розробники повинні витрачати додатковий час на пояснення технічних деталей, що може частково знизити переваги такого підходу. Також, існує ризик втрати контексту або неправильного трактування інформації [3].

Використання корпоративних баз знань типу Confluence, Notion, дозволяють централізувати процес створення та зберігання документації, забезпечуючи доступність інформації для всіх учасників проєкту та сприянню накопичення знань. До того ж, система управління знаннями дозволяє реалізувати механізми контролю версій та історії змін [4 – 5].

В Agile-підході документація базується на принципі розумної достатності, де головним пріоритетом є створення цінності, а не формальне заповнення паперів. Замість об'ємних технічних завдань, які готуються на початку проєкту і швидко втрачають актуальність, команда створює «живі» документи лише тоді, коли в них виникає реальна потреба для розробки чи підтримки продукту. Основний акцент зміщується з детального опису планів на фіксацію ключових архітектурних рішень та бізнес-логіки, що дозволяє зберігати гнучкість і швидко адаптуватися до змін без необхідності

переписувати сотні сторінок застарілих інструкцій. В основі лежить ідея, що найкращим джерелом істини є працюючий програмний код, а супровідні тексти слугують лише допоміжним інструментом для порозуміння між людьми [6].

Використання ШІ – це технологія, яка автоматизує такі завдання, як вилучення даних, аналіз поведінки, форматування контенту та зменшення помилок. Використовуючи передові алгоритми машинного навчання, ШІ пришвидшує обробку документів, швидко та точно обробляючи великі обсяги контенту.

Поєднання машинного навчання з обробкою природної мови (Natural Language Processing, NLP) надає інструментам документування на базі ШІ витягувати ключову інформацію з існуючого технічного контенту. Ці системи розуміють контекст, граматику та стиль тексту, що дозволяє їм створювати високоякісні документи процесу, які автоматично класифікуються відповідно. Це включає будь-яку документацію, створену технічними авторами, таку як документація Software Development Kit (SDK), тестування прийняття користувачем (user acceptance testing, UAT), документація API з коду або посібники користувача зі специфікацій продукту [7 – 8].

Інструменти на базі ШІ створюють узгодженість термінології та покращують ясність, аналізуючи структуру речень, їх плавність та загальну зв'язність. Ці інструменти також допомагають покращити читабельність, скоротити час редагування та полегшити розуміння складної технічної інформації для користувачів. Крім того, ШІ спрощує управління контентом, автоматизуючи теги, категоризацію та застосування метаданих, покращуючи організацію та полегшуючи командам пошук та повторне використання контенту.

Найпереконливішим аргументом на користь інтеграції ШІ у процес документування є його ефективність, котра візуально відображена у таблиці 1.1.

Таблиця 1.1 – Порівняння традиційного та інтегрованого з ШІ підходів до написання документації

Критерій аналізу	Традиційний підхід	Підхід на основі ШІ
Створення контенту	Автори вручну створюють документацію на основі нотаток розробників та зустрічей.	Інструменти ШІ, такі як Mintlify, Swimm та GitHub Copilot, аналізують кодові бази та автоматично генерують документацію API, зменшуючи ручні зусилля.
Синхронізація з кодом	Відбувається в ручну через людський фактор у вигляді моніторингу нових комітів	Автоматична через прописані тригери в моделі при зміні специфікації API або архітектури моделі розробником
Обробка технічних функцій	Низька точність, бо приклади коду часто копіюються вручну розробниками і застарівають.	Висока точність. ШІ витягує сигнатури функцій та типи даних безпосередньо з вихідного коду, одразу після ловлі тригера про зміни
Редагування та узгодженість	Технічні автори витрачають години на коректуру тону, граматики та стилю	Помічники з написання на базі ШІ, такі як Grammarly, DeepL Write, Jasper AI та Writer.com, пропонують покращення для ясності, узгодженості та читабельності
Оновлення документації	Зміни в програмному забезпеченні, API або функціях вимагають ручного редагування, що часто призводить до застарілого контенту	Інтегровані з ШІ інструменти, такі як Redocly та DocGPT, сканують репозиторії коду та позначають застарілу документацію, пропонуючи автоматичні оновлення.
Пошук та доступність	Користувачам важко знайти відповідну документацію, покладаючись на статичні функції пошуку.	Інструменти пошуку на базі ШІ, такі як Algolia, Kogma.ai та вбудовані OpenAI, покращують пошук документації за допомогою NLP.

Створення одного документа з процедурою вручну може зайняти у технічного автора від одного до двох тижнів, що коштуватиме бізнесу тисяч доларів на оплату праці лише за один ресурс. ШІ безпосередньо вирішує цю проблему, автоматизуючи найбільш повторювані та трудомісткі частини процесу. Виконуючи такі завдання, як збір інформації, початкове складання та

організація контенту, ШІ зменшує ручні зусилля, необхідні для запуску документа. Ця автоматизація не лише прискорює створення контенту, але й дозволяє вашим кваліфікованим авторам зосередитися на більш стратегічній роботі, такій як архітектура контенту, рецензування та забезпечення технічної точності. З часом цей перехід призводить до значної економії коштів та більш продуктивної, цілеспрямованої команди з технічного контенту.

Отримані результати аналізу підходів до організації процесу документування підтверджують висновок про те, що сучасні методи та підходи мають суттєві недоліки, включаючи неефективне використання ресурсів; відсутність стандартизованих процедур; та труднощі із забезпеченням того, щоб дані були корисними та актуальними.

1.3 Постановка задачі дослідження

Метою дослідження є розробка науково обґрунтованого та практично застосовного набору моделей і методів управління роботами зі створення допоміжної документації в ІТ-проектах, орієнтованих на платформи генерації цифрового контенту. Ця мета виникає з виявленого протиріччя між зростаючою складністю сучасних програмних систем на основі ШІ та недостатньою розвиненістю спеціалізованих систем управління документацією, здатних враховувати унікальні характеристики таких платформ.

На відміну від інших чітко визначених процесів розробки (включаючи каскадні або класичні Agile-моделі), які розглядають документацію як побічний продукт розробки, ця модель розглядає документацію як гнучкий та еволюційний об'єкт, який розвивається одночасно з функціональністю системи.

Специфіка платформ генерації цифрового контенту, включаючи їхню залежність від конвеєрів машинного навчання, динамічних архітектур API та ітеративних циклів перенавчання моделей, вводить вимоги до допоміжної документації, які принципово відрізняються від тих, що розглядаються в традиційних стандартах документації програмного забезпечення (ISO/IEC 26514, , РМВОК 7). Існуючі підходи розглядають документацію як статичний артефакт, що створюється на визначених фазах проєкту, тоді як платформи цільового класу вимагають постійної еволюції документації паралельно з оновленнями моделі, змінами інтерфейсу та новими моделями взаємодії з користувачем [9 – 11] .

Наукова новизна цього дослідження полягає в розробці інтегрованого підходу до управління роботами зі створення допоміжної документації в ІТ-проєктах, спеціально адаптованого до умов платформ генерації цифрового контенту, та відмінного від відомих рішень.

Крім того, дослідження включатиме створення математичної моделі, яка визначатиме зв'язок між витратами ресурсів, часом, витраченим на створення документації, та якістю створеної документації. З формальної точки зору ця проблема буде виражена за допомогою задачі оптимізації. У цьому випадку необхідно мінімізувати загальну кількість часу, витраченого на виконання робіт, та фінансові ресурси, необхідні для створення документації. Також необхідно буде враховувати обмеження, пов'язані із забезпеченням дотримання необхідного рівня якості документації, а також зменшенням навантаження на розробника. Тому задача формується як «мінімізація функцій часу та витрат за умови заданого рівня якості документації та мінімізація участі розробника у створенні документації».

Для цього час, необхідний для створення документації, це загальна сума часу, необхідного для кожного етапу створення документації, що включає створення документів, редагування документів та перевірку документів. Витрати, пов'язані зі створенням документації, базуються на функції робочого часу конкретного персоналу, залученого до створення документації, та витрат,

понесених на використання інтелектуальних інструментів. Якість документації це складна характеристика, яка може складатися зі ступеня точності; ступеня повноти; ступеня узгодженості; та ступеня відповідності документації фактичному стану використовуваної системи.

Ще одним ключовим аспектом є мінімізація робочого навантаження розробників для досягнення максимальної ефективності проєкту. У рамках цього дослідження пропонуються відповідні показники ефективності, які дозволяють кількісно виміряти переваги застосування запропонованих моделей. Зокрема, співвідношення між часом, витраченим на традиційні методології, та часом, що використовує ІІІ, дасть змогу оцінити, як тривалість процесу покращилася завдяки впровадженню технологій ІІІ. Крім того, співвідношення між фінансовими витратами на традиційні методології та витратами на методології ІІІ також можна оцінити, щоб оцінити економічне обґрунтування впровадження технологій ІІІ.

Сфера застосування охоплює, перш за все, ІТ-проєкти, що розробляють платформи для автоматизованої або напівавтоматизованої генерації цифрового контенту, включаючи системи генерації тексту, конвеєри синтезу зображень, мультимодальні контентні платформи та інструменти редагування на основі ІІІ. Результати дослідження також потенційно застосовні, з відповідною адаптацією, до ІТ-проєктів у суміжних галузях, що характеризуються порівнянними проблемами документації: частими змінами інтерфейсу, ітеративними оновленнями моделей та високою різноманітністю профілів кінцевих користувачів.

2 РОЗРОБКА МОДЕЛЕЙ І МЕТОДІВ УПРАВЛІННЯ РОБОТАМИ ЗІ СТВОРЕННЯ ДОПОМІЖНОЇ ДОКУМЕНТАЦІЇ

2.1 Розробка структурної та функціональної моделі процесу створення допоміжної документації

Моделювання функціонального процесу дає змогу перейти від вільно описового опису проведеної діяльності до більш формалізованої моделі, яку потім можна оптимізувати. У випадку ІТ-проектів, пов'язаних із розробкою платформ генерації цифрового контенту, цей процес створення супровідної документації є особливо складним через динамізм змін, використання компонентів ШІ та необхідність підтримки актуальності інформації.

Для формалізації цього процесу необхідно побудувати його структурну модель, яка відображає логічну послідовність етапів, можливі стани документів та умови переходу між ними. Під структурною моделлю розуміється абстрактне представлення процесу, що визначає його компоненти та статичні зв'язки між ними. Функціональна ж модель доповнює структурну, бо вона описує саме динаміку процесу: яким чином виконуються функції, хто їх виконує та що є результатом кожної функції.

На рисунку 2.1 відображено процес створення допоміжної документації, котрий поділяється на шість послідовних етапів, кожен з яких має чітко визначені входи, виходи та відповідальних учасників. На відміну від традиційного підходу, у якому документація створюється нерегулярно та не має чіткої структури, запропонована модель передбачає впорядкований процес, що включає як автоматизовані, так і експертні етапи.

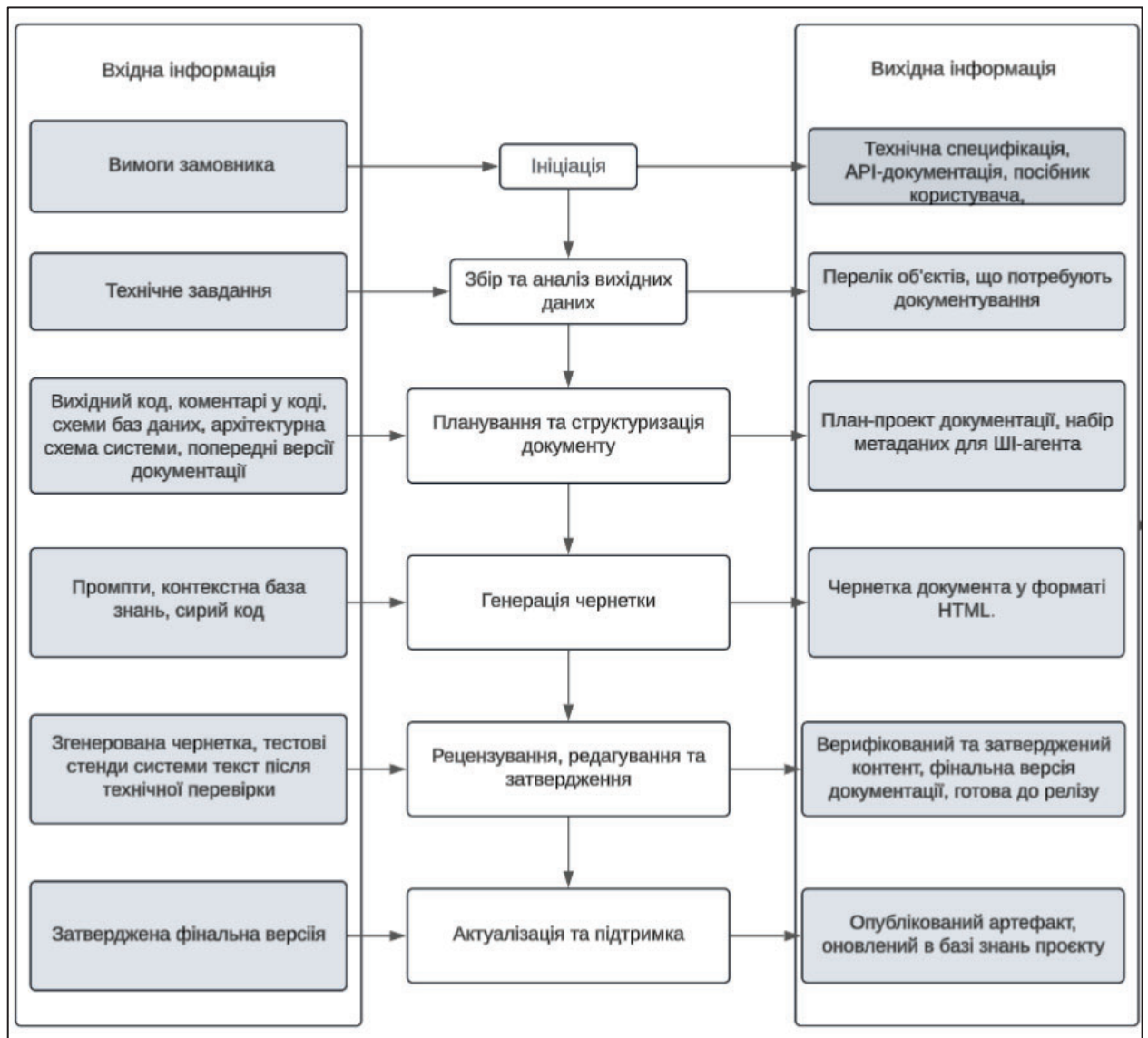


Рисунок 2.1 – Основні етапи створення технічної документації в IT-проєкті

Ініціація процесу створення допоміжної документації виступає початковою стадією, на якій формується потреба у створенні або оновленні документів. У межах сучасних IT-проєктів така потреба зазвичай виникає як реакція на зміни у програмному продукті або запити зацікавлених сторін. Зокрема, це може бути пов'язано з початком нового спринту, появою нового функціонального модуля, змінами у вимогах або архітектурі системи, а також із необхідністю усунення прогалин у вже існуючій документації. На цьому етапі визначається тип майбутнього документа, його призначення, цільова аудиторія та рівень деталізації. Також встановлюються пріоритет і строки

виконання роботи. Результатом етапу є сформоване та зафіксоване у системі управління проєктом завдання, яке стає основою для подальшої організації процесу.

Другий етап передбачає збір та аналіз вихідних даних, необхідних для створення документації. Його значення полягає у забезпеченні інформаційної бази, на основі якої формується зміст документа. Джерела інформації можуть мати різну природу: від технічних артефактів, таких як програмний код, специфікації API та архітектурні схеми, до результатів комунікації між учасниками проєкту, включаючи протоколи нарад, службове листування та записи обговорень. Отримані дані підлягають аналізу на повноту, достовірність та узгодженість. У випадку виявлення недостатності або суперечливості інформації ініціюється уточнення даних шляхом взаємодії з відповідними учасниками команди. Таким чином забезпечується якісна основа для подальших етапів розробки документації.

Третій етап полягає у плануванні та структуризації майбутнього документа. До початку написання визначається структура майбутнього документа: розділи, підрозділи, послідовність викладення матеріалу. На цьому етапі обирається шаблон документа відповідно до його типу та стандартів організації. Також визначається стратегія генерації: яка частина документа буде створена вручну, яка – автоматично за допомогою ШІ, а яка – комбінованим способом.

Четвертий етап пов'язаний із формуванням чернетки документа, що є першим матеріалізованим результатом процесу. В цей період відбувається створення тексту на основі зібраних даних і визначеної структури. Залежно від обраної стратегії генерація може здійснюватись вручну розробником, автоматично засобами ШІ на основі зібраних даних та гібридно, завдяки ШІ, котрий генерує базову структуру та текст, людина доповнює контекстними деталями. Водночас важливим залишається контроль якості вхідних даних і коректність постановки завдань для системи генерації. У результаті

формується чернетка документа, яка містить основний зміст і потребує подальшого доопрацювання.

П'ятий етап включає рецензування, редагування та затвердження документації. Його метою є забезпечення відповідності документа фактичному стану системи, а також досягнення необхідного рівня якості викладення. Чернетка проходить перевірку на технічну коректність (відповідає дійсному стану системи), повноту (охоплює всі необхідні аспекти) та якість викладення (зрозумілість, відповідність стандартам оформлення). Рецензування може відбуватися в кілька раундів – між чернеткою та фінальною версією документ може повертатися на доопрацювання. Після успішного рецензування документ затверджується відповідальною особою та публікується у визначеному сховищі (Confluence, GitHub Wiki, корпоративний портал тощо).

Важливою особливістю є поєднання автоматизованих інструментів та експертної оцінки, що дозволяє забезпечити баланс між ефективністю та якістю. Після завершення перевірки документ затверджується відповідальною особою, що свідчить про його готовність до використання.

Шостий етап пов'язаний із публікацією документації та підтримкою її актуальності. Після затвердження документ інтегрується у відповідне інформаційне середовище та стає доступним для користувачів. Водночас документація розглядається не як завершений продукт, а як динамічний елемент системи, який потребує постійного оновлення. У міру змін у програмному продукті виникає необхідність коригування змісту документів або створення нових версій.

Таким чином, процес підтримки документації набуває циклічного характеру та триває протягом усього життєвого циклу ІТ-проєкту.

Далі необхідно формалізувати життєвий цикл документа як динамічну систему.

Процес створення документації розглядається як перехід інформаційного об'єкта через множину дискретних станів.

$$S = \{s_0, s_1, s_2, s_3, s_4, s_5, s_6\},$$

де S – множина всіх можливих станів документу в системі;

s – стан документу.

Кожен стан характеризує ступінь готовності документа та визначає доступні операції управління.

Для математичного моделювання та автоматизації в системі управління проектом стани класифікуються згідно з таблицею 2.1.

Таблиця 2.1 – Опис станів життєвого циклу документу

Позначення	Стан	Опис та умови перебування
1	2	3
s_0	Відсутній	Початкова точка. Потреба у документі існує, але ще не зафіксована в системі управління.
s_1	Запланований	Створено завдання. Визначено тип документа, пріоритет та призначено відповідальну роль – людина або ШІ-агент.
s_2	В розробці	Процес активної генерації. На цьому етапі ШІ-агент створює чернетку на основі технічного контексту платформи.
s_3	На рецензії	Чернетка сформована та передана технічному експерту або розробнику для верифікації точності.
s_4	Допрацювання	Рецензент виявив невідповідності або "галюцинації" ШІ. Об'єкт повертається на етап редагування з новими вказівками.
s_5	Затверджений	Документ пройшов перевірку, опублікований у базі знань і є доступним для кінцевих користувачів.
s_6	Застарілий	Система зазнала змін, які не відображені в документі. Потребує повернення до стану s_1 «Запланований»

Взаємодія між станами описується орієнтованим графом, відображеним на рисунку 2.2. Особливістю інтелектуальної системи є наявність циклів зворотного зв'язку між станами $S_3 \rightarrow S_4 \rightarrow S_2$, що дозволяє системі самонавчатися на основі правок експерта. Також можна відмітити ще одну властивість моделі, а саме наявність зворотного переходу $S_6 \rightarrow S_1$, що відображає циклічну природу підтримки документації.

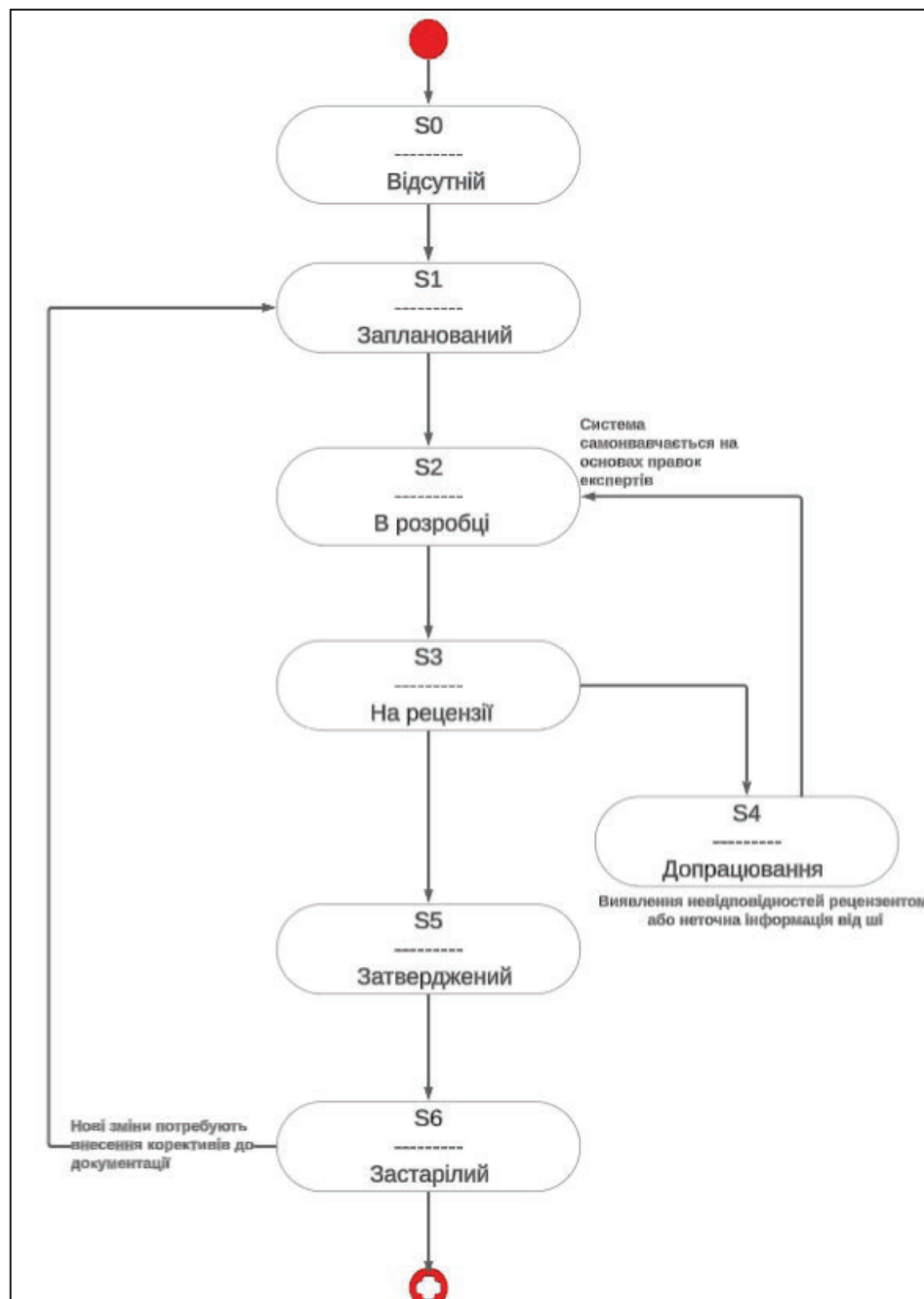


Рисунок 2.2 – Взаємодія між станами життєвого циклу документу

У процесі створення документації залучені учасники з різними функціями, повноваженнями та зонами відповідальності:

- менеджер проєктів (Project Manager, PM) відповідає за ініціацію процесу документування та його організаційну підтримку, визначає пріоритети документів у контексті загального плану проєкту, призначає виконавців, контролює дотримання термінів;

- розробник надає доступ до вихідного коду, пояснює архітектурні рішення, описує поведінку модулів, але у традиційних моделях написання документації створює значне навантаження, відволікаючи розробників від основних задач;

- технічний письменник, відповідає за якість і структуру документів, перетворює технічну інформацію від розробників на зрозумілі, структуровані тексти, визначає стиль, формат та стандарти документації в організації, проте в моделі з використанням ШІ-агенту, технічний письменник переходить від ролі «автора» до ролі «редактора та куратора»;

- рецензент може суміщатися з роллю розробника або технічного письменника, відповідаючи за перевірку технічної коректності та відповідності стандартам, формує зворотний зв'язок і затверджує документ до публікації;

- ШІ-агент виконує задачі генерації тексту, структурування інформації та первинного контролю якості, він не є самостійним учасником команди, бо діє в межах, визначених іншими ролями, і потребує людського нагляду.

Для того, щоб було можливо чітко визначити всі операції, котрі будуть призначатися для конкретної ролі учасника команди у процесі створення документації, було розроблено матрицю відповідальності, діаграму RACI (таблиця 2.2)

Таблиця 2.2 – Матриця відповідальності в форматі RACI (В – Відповідальний; П – Підзвітний; К – Консультант; І – Проінформований)

Етап	РМ	Розробник	Тех.письменник	Рецензент	ІІІ-агент
Ініціація	В/П	К	К	І	-
Збір даних	І	В	П	-	В
Планування	І	К	В/П	-	К
Генерація	І	К	П	-	В
Рецензування, редагування, затвердження	І	В/І	К	В/П/К	К
Актуалізація	П	К	В	К	В

Аналіз побудованої матриці показує, що менеджер проекту відіграє ключову роль на етапах ініціації, планування та актуалізації, забезпечуючи загальну координацію процесу. Розробник є основним джерелом інформації на етапах збору даних та рецензування, тоді як технічний письменник бере активну участь у плануванні, генерації та перевірці документації. ІІІ-агент концентрується переважно на етапі генерації, виконуючи функції автоматизованого створення контенту. Рецензент забезпечує контроль якості на фінальних етапах.

Для формалізації взаємодії між ІІІ та учасниками команди розробки, всі інформаційні потоки класифікуються за трьома засадничими ознаками:

- за напрямком руху;
- за типом та структурю носія;
- за терміновістю.

Класифікація інформаційних потоків дозволяє систематизувати їх та визначити особливості обробки залежно від типу та призначення. З точки зору напрямку руху інформації виділяються вхідні, внутрішні та вихідні потоки.

Вхідні потоки формують інформаційну основу процесу та надходять із зовнішніх джерел, таких як вихідний код, вимоги замовника або специфікації системи.

Внутрішні потоки забезпечують взаємодію між учасниками процесу та включають передачу чернеток документів, зауважень, коментарів та уточнень.

Вихідні потоки є результатом обробки інформації та представлені у вигляді готової документації, що передається кінцевим користувачам або інтегрується у систему знань проєкту.

За типом носія інформаційні потоки поділяються на структуровані, напівструктуровані та неструктуровані.

Структуровані дані мають чітко визначену форму та легко піддаються автоматизованій обробці, що є особливо важливим для інтеграції з інструментами ІІІ.

Напівструктуровані дані, такі як Markdown-документи або коментарі у коді, містять як формалізовану, так і вільну інформацію, що потребує додаткової обробки.

Неструктуровані дані, включаючи усні пояснення або неформальне листування, є найскладнішими для формалізації, однак часто містять критично важливий контекст.

Класифікація за терміновістю дозволяє враховувати часові характеристики інформаційних потоків.

Синхронні потоки передбачають обмін інформацією в режимі реального часу, що характерно для інтеграції із системами ІІІ або автоматизованими сервісами. Асинхронні потоки, навпаки, мають затримку у передачі та обробці, що є типовим для комунікації через системи управління проєктами або системи контролю версій..

Особливе значення має аналіз ключових інформаційних потоків, що забезпечують функціонування системи.

Потік від вихідного коду до ІІІ-агента є базовим для автоматизованої генерації документації, оскільки саме він формує первинний інформаційний контекст. Ефективність цього потоку визначає якість початкової чернетки документа та безпосередньо впливає на подальші етапи.

Передача інформації від розробника до технічного письменника забезпечує доповнення формалізованих даних контекстними знаннями, які не відображені у коді.

Потік від ІІІ-агента до технічного письменника відображає результат автоматизованої генерації та є центральним елементом моделі з використанням ІІІ. Саме на цьому етапі визначається баланс між автоматизацією та участю людини, оскільки від якості згенерованого контенту залежить обсяг подальшого редагування.

Зворотний потік від рецензента до технічного письменника або ІІІ-агента забезпечує механізм корекції та вдосконалення документації, де він може використовуватися не лише для виправлення помилок, але й для навчання інтелектуальної системи, що сприяє підвищенню якості генерації в майбутньому.

Потік інформації від системи управління проєктом до всіх учасників виконує координаційну функцію, забезпечуючи синхронізацію дій та своєчасне реагування на зміни. Він виступає тригером для запуску нових циклів створення або оновлення документації, що є особливо важливим в умовах ітеративної розробки.

На рисунку 2.3 DFD-діаграми процес створення допоміжної документації представлено як систему взаємопов'язаних процесів, зовнішніх сутностей і сховищ даних. Зовнішніми сутностями виступають замовник, система контролю версій і система управління проєктом, які формують вхідну інформацію для запуску та підтримки процесу документування. Основними процесами є збір даних, генерація чернетки, рецензування та публікація документації.

Процес збору даних акумулює технічні та організаційні відомості, необхідні для формування змісту документа. На етапі генерації чернетки використовуються дані з бази знань, що містить шаблони, попередні версії документів і контекст проєкту. Процес рецензування забезпечує перевірку згенерованого матеріалу та формує зворотний зв'язок, який у разі потреби

повертається на етап генерації. Після затвердження документа здійснюється його публікація у репозиторії документації, а всі зміни фіксуються у журналі версій.

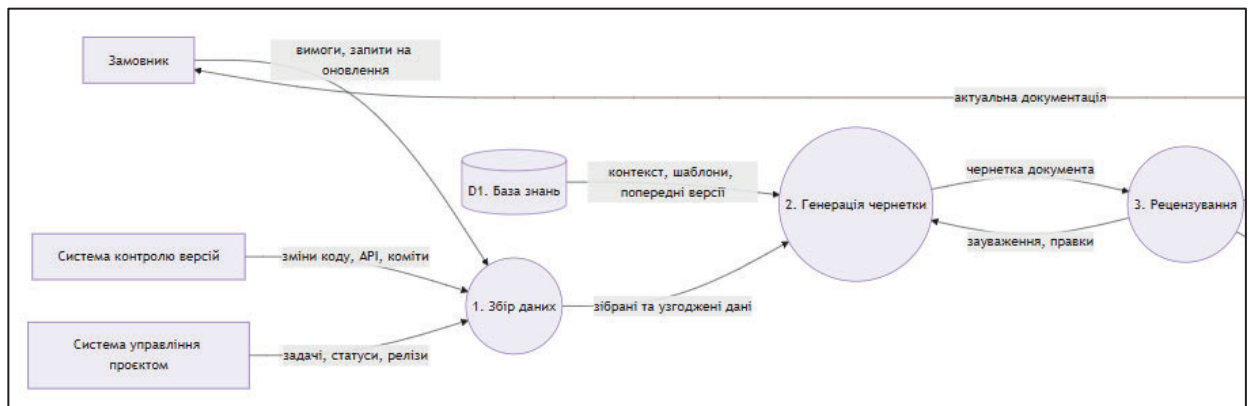


Рисунок 2.3 – Діаграма потоків даних (частина 1)



Рисунок 2.3 – Діаграма потоків даних(частина 2)

У розділі 2.1 було розроблено структурну та функціональну модель процесу створення допоміжної документації в ІТ-проєкті. Процес представлено як систему взаємопов'язаних етапів, станів, ролей та інформаційних потоків, що дозволило формалізувати його та підготувати до подальшого математичного аналізу.

Запропонована модель враховує сучасні особливості ІТ-проєктів, зокрема використання ІІІ на етапі генерації документації та ітеративний характер процесу. Це дозволило визначити ключові точки управління, зони

відповідальності учасників і вузькі місця, пов'язані з обробкою інформації та рецензуванням.

2.2 Розробка моделі розподілу задач між людиною та штучним інтелектом у процесі створення документації

Інтеграція ШІ у процес створення допоміжної документації не повинна розглядатися як заміна людської участі, а радше як еволюція підходу до організації роботи. У сучасних ІТ-проєктах, особливо тих, що пов'язані з розробкою платформ генерації цифрового контенту, ефективність досягається не за рахунок повної автоматизації, а шляхом побудови гібридної системи взаємодії між людиною та інтелектуальними інструментами. У такій системі ШІ бере на себе виконання формалізованих, повторюваних задач, що не потребують глибокого контекстного аналізу, тоді як людина зосереджується на задачах, які вимагають критичного мислення, інтерпретації, узгодження суперечливих даних та прийняття рішень.

Ефективність такої інтеграції безпосередньо залежить від чіткого визначення ролі ШІ в межах процесу. Недостатньо лише впровадити інструмент – необхідно формалізувати точки його застосування, визначити режими функціонування та забезпечити узгоджену взаємодію з іншими учасниками системи. Інтеграція ШІ розглядається на трьох рівнях, які відповідають різним глибинам втручання у процес:

- рівень 1 – інструментальний;
- рівень 2 – процесний;
- рівень 3 – системний.

На інструментальному рівні ШІ використовується як допоміжний інструмент без змін у загальній структурі процесу. Учасники звертаються до ШІ за власною ініціативою для виконання окремих мікрозадач: генерації

абзацу, перефразування, перевірки граматики. Процес залишається людино-центричним, ІІІ є аналогом розширеного текстового редактора.

Процесний рівень являє собою вбудову ІІІ у визначені етапи процесу як обов'язковий компонент. Наприклад, кожна нова задача у трекері автоматично ініціює генерацію чернетки документа засобами ІІІ, яка потім передається технічному письменнику. Процес перебудовується навколо ІІІ як центрального виконавця рутинних задач.

В системному рівні ІІІ інтегрується не лише у виконання задач, а й в управління процесом: аналізує стан документації, виявляє застарілі документи, пріоритизує задачі на оновлення, генерує звіти про покриття документацією. На цьому рівні ІІІ виконує функції, що раніше належали менеджеру або аналітику якості.

В межах даного дослідження у запропонованій моделі обирається комбінація другого та третього рівня інтеграції ІІІ у процес.

Залежно від характеру задачі та ступеня її формалізації ІІІ може функціонувати в одному з трьох режимів:

- автоматичний режим відбувається через виконання задачі повністю самостійно, без залучення людини, де застосовується лише для задач із високим ступенем формалізації та низькою ціною помилки;

- режим асистента дає змогу ІІІ генерувати варіанти або пропозиції, застосовуючись для задач, що вимагають контекстного розуміння або суб'єктивної оцінки, таких як генерація кількох варіантів опису функціональності для вибору технічним письменником, підказки при формулюванні технічних вимог або виявлення потенційних прогалин у документації з рекомендаціями щодо їх усунення, де людина тільки приймає рішення,;

- аналітичний режим дозволяє аналізувати наявні дані та формує звіти, оцінки або рекомендації без генерації тексту документів, де результат може виводитись у форматі оцінки повноти покриття функціоналу документацією,

виявлення суперечностей між різними документами та оцінкою читабельності та відповідності термінологічним стандартам.

На основі моделі процесу, побудованої у підрозділі 2.1, розрахуємо конкретні точки інтеграції для порівняльного аналізу трудомісткості процесу створення допоміжної документації у двох сценаріях: традиційному (без використання ІІІ) та оптимізованому (з використанням ІІІ).

Вхідні параметри моделі процесу створення допоміжної документації отримані шляхом методу експертних оцінок, де було опитано 5 спеціалістів, та середнє значення було прийнято як базове. Збір здійснювався у двох умовах – базова модель (без використання ІІІ) та автоматизована модель (з використанням ІІІ). Об'єктом дослідження слугував модуль авторизації та управління користувачами у веб-застосунку, котрий включав 8 функцій, 3 API-ендпоінти та 2 таблиці даних. Збір даних проводився шляхом іксації фактичних витрат часу учасників процесу на кожному етапі створення документації. Отримані дані наведені у таблицях 2.3 та 2.4.

Для розрахунку кожного етапу буде використовуватись формула

$$\Delta T_i = \frac{T_i^{manual} - T_i^{AI}}{T_i^{manual}} \times 100\%, \quad (2.1)$$

де T_i^{manual} – тривалість виконання етапу без використання ІІІ,

T_i^{AI} – тривалість цього ж етапу після інтеграції ІІІ.

Таблиця 2.3 – Дані про витрати часу на етапах документування модуля авторизації у базовій моделі написання документації

Етап процесу	Склад робіт	Виконавець	Витрати часу, год	Примітки
1	2	3	4	5
Збір даних	Ручний перегляд вихідного коду, читання коментарів,	Розробник	3-5	Час на з'ясування деталей

Кінець таблиці 2.3

1	2	3	4	5
Планування структури	Визначення розділів документа, вибір шаблону, погодження структури з командою	Розробник	2	-
Генерація чернетки	Написання опису модуля, функцій, параметрів, прикладів використання	Розробник	5	-
Рецензування	Перевірка технічної коректності, повноти, відповідності коду	Розробник	3	-
Доопрацювання	Внесення правок за результатами рецензування, повторна перевірка	Розробник	2	2-4 правки від тех.ліда
Публікація	Форматування, завантаження у Confluence, оновлення навігації	Розробник	1	Ручне оновлення всіх посилань на документ
Актуалізація	Відстеження змін у коді, ручне порівняння з документом, оновлення	Розробник	3	Виконується при кожному релізі

У базовій моделі найбільш витратним є етап генерації чернетки, майже 5 год, що пояснюється необхідністю ручного формулювання та структурування технічної інформації. Значну частку також займають збір даних та актуалізація. Сумарно розробник витрачає 7.5 год на задачі, що безпосередньо не пов'язані з написанням коду, що свідчить про суттєве відволікання ключового ресурсу команди.

У таблиці 2.4 наведено дані про витрати часу на різних етапах документування модуля авторизації.

Таблиця 2.4 – Дані про витрати часу на етапах документування модуля авторизації в авторизованій моделі написання документації

Етап процесу	Склад робіт	Виконавець	Підготовка запиту, хв	Виконання ІІІ, хв	Перевірка людиною, хв	Разом, год
1	2	3		4	5	
Збір даних	Автоматичний парсинг коду/коментарів/ схеми БД,	ІІІ + розробник	20	40	30	1,5
Планування структури	Генерація структури документа на основі зібраних даних, шаблон	ІІІ + розробник	20	15	40	1,2
Генерація чернетки	Автоматичне написання опису функцій	ІІІ + розробник	20	30	70	2
Рецензування	Автоматична перевірка форматування та повноти, змістова перевірка людиною	ІІІ + розробник	5	15	100	2
Доопрацювання	ІІІ пропонує правки, людина затверджує	ІІІ + розробник	10	15	35	1
Публікація	Автоматичне форматування та оновлення артефактів	ІІІ + розробник	5	20	25	0,8
Актуалізація	Автоматичний моніторинг змін у коді, сповіщення про застарілі розділи документу	ІІІ + розробник	5	55	30	1,5

За отриманими даними, середній час збору технічних даних для одного модуля становить 4 години, в автоматизованій – 1,5 години. Тоді ефект становить

$$\Delta T_1 = \frac{4 - 1.5}{4} \times 100\% = 62.5\%$$

Для етапу планування взято 2 години без ІІІ та 1,2 години з ІІІ. Тоді

$$\Delta T_2 = \frac{2 - 1.2}{2} \times 100\% = 40\%.$$

Для етапу генерації чернетки отримуємо

$$\Delta T_3 = \frac{5 - 2}{5} \times 100\% = 60\%$$

Для етапу рецензування час скорочується не так суттєво, оскільки тут залишається критично важлива людська участь. Якщо взяти 3 години без ІІІ та 2 години з ІІІ, отримаємо

$$\Delta T_4 = \frac{3 - 2}{3} \times 100\% \approx 33.3\%$$

Однак основний ефект на цьому етапі полягає не лише у скороченні часу, а у зменшенні кількості формальних помилок, які переходять у наступні ітерації.

Для доопрацювання документу отримуємо

$$\Delta T_5 = \frac{2 - 1}{2} \times 100\% = 50\%$$

а для його публікації

$$\Delta T_6 = \frac{1 - 0.8}{1} \times 100\% = 20\%$$

Підсумуючи результати розрахунків у таблиці 2.5, можна побачити етапи, та порівняти, де інтеграція ІІІ дає найбільший ефект

Таблиця 2.5 – Розрахунок точок інтеграції

Етап процесу	Точка інтеграції	Режим ІІІ	Час без ІІІ, год	Час з ІІІ, год	Очікуваний ефект
1	2	3	4	5	6
Збір даних	Парсинг коду та коментарів	Авто	4	1,5	Скорочення часу збору на 60 – 70%
Планування	Генерація структури документа	Асист	2	1,2	Скорочення часу планування на 45%
Генерація чернетки	Написання первинного тексту	Авто / Асист	5	2	Скорочення часу написання на 50 – 65%
Рецензування	Автоматична перевірка якості	Авто	3	2	Виявлення 70 – 80% формальних помилок, скорочення часу перевірки
Доопрацювання	Корекція після зауважень	Асист	2	1	Скорочення часу доопрацювання на 50%
Публікація	Автоматизація оновлення артефактів	Авто	1	0.8	Незначне скорочення часу публікації
Актуалізація	Моніторинг змін у коді	Авто	-	-	Своєчасне виявлення застарілої документації

Розрахунок очікуваного ефекту для кожної точки інтеграції ІІІ виконано шляхом порівняння тривалості відповідного етапу у традиційній та автоматизованій моделях. Для цього використано показник відносного скорочення часу

Розподіл задач між людиною та ІІІ є ключовим проєктним рішенням моделі управління. Некоректний розподіл призводить до двох типів неефективності:

- надмірна автоматизація: ІІІ виконує задачі, що вимагають людського судження, що призводить до помилок і зниження якості документації
- недостатня автоматизація: людина виконує рутинні задачі, що можуть бути делеговані ІІІ, що призводить до надмірного навантаження та зростання витрат

Для уникнення обох крайнощів пропонується формальний метод оцінки задач за набором критеріїв, котрий оцінюється за чотирма ключовими характеристиками:

- ступінь формалізації (F), котрий відображає, наскільки задача може бути описана у вигляді чітких правил або алгоритмів;
- потреба у контекстному розумінні (C), визначає ступінь залежності задачі від неформалізованих знань, таких як бізнес-логіка, командні домовленості або стратегічні цілі проєкту;
- критичність помилки (E), характеризує потенційні наслідки некоректного результату;
- обсяг повторюваності (V), відображає частоту виконання задачі у процесі.

На основі цих критеріїв формується інтегральний показник придатності задачі до автоматизації – індекс ASI(Automation Suitability Index), наведений у таблиці 2.6

$$ASI_i = w_1 \times F_i + w_2 \times (1 - C_i) + w_3 \times (1 - E_i) + w_4 \times V_i ,$$

де ASI_i – індекс придатності задачі до автоматизації ;

w_n – вагові коефіцієнти критеріїв;

F_i – ступінь формалізації описання задачі;

C_i – потреба у контекстному розумінні задачі;

E_i – критичність помилки задачі;

V_i – обсяг повторюваності задачі, що відображає чистоту її викнання у процесі документування;

Вагові коефіцієнти w_1 - w_4 визначені методом попарних порівнянь критеріїв, що є стандартним підходом у задачах багатокритеріального прийняття рішень

$$w_1 + w_2 + w_3 + w_4 = 1$$

де w_n – вагові коефіцієнти критеріїв;

Для кожної пари критеріїв встановлювалась відносна важливість за дев'ятибальною шкалою з урахуванням специфіки процесу документування в ІТ-проектах. Так, критерії формалізації (F) та критичності помилки (E) отримали однакову вагу $w_1=w_3=0.30$, оскільки обидва є визначальними факторами при прийнятті рішення про автоматизацію: формалізація визначає принципову можливість делегування задачі ШІ, а критичність визначає допустимість такого делегування з точки зору ризику.

Критерій контекстного розуміння(C) отримав вагу $w_2=0.25$, оскільки є важливим, але другорядним відносно перших двох, бо задача з помірною залежністю від контексту може бути делегована ШІ за умови низької критичності помилки. Критерій повторюваності(V) отримав найменшу вагу $w_4=0.15$, оскільки є радше економічним аргументом на користь автоматизації, ніж її необхідною умовою.

Таблиця 2.6 – Інтерпретація значень ASI

Діапазон ASI	Рішення	Режим ІІІ
1	2	3
$0.75 \leq ASI \leq 1$	Повна автоматизація	Авто
$0.50 \leq ASI < 0.75$	Часткова автоматизація	Асистент
$0.25 \leq ASI < 0.50$	Мінімальна підтримка ІІІ	Аналітичний
$0 \leq ASI < 0.25$	Виконується людиною	-

Отримане значення ASI в таблиці 2.7 дозволяє визначити режим виконання задачі. Високі значення індексу свідчать про доцільність повної автоматизації, середні – про необхідність асистивного режиму, а низькі – про пріоритет ручного виконання із можливим аналітичним супроводом ІІІ. Таким чином, формується адаптивна матриця розподілу задач, яка враховує специфіку кожного типу робіт у процесі документування.

Таблиця 2.7 – Матриця розподілу задач після розрахунків

Задача	F	C	E	V	ASI	Режим ІІІ
1	2	3	4	5	6	7
Парсинг коду та коментарів	0.95	0.05	0.25	0.95	0.88	Авто
Генерація структури документа	0.70	0.30	0.40	0.60	0.64	Асистент
Написання технічного опису API	0.85	0.20	0.50	0.85	0.74	Асистент
Опис архітектурних	0.25	0.85	0.70	0.30	0.22	Людина
Перевірка форматування	0.95	0.05	0.20	0.90	0.89	Авто
Рецензування змісту	0.30	0.80	0.85	0.70	0.27	Людина
Актуалізація	0.95	0.10	0.30	0.95	0.86	Авто

Запропонована модель управління процесом створення документації з використанням ІІІ базується на концепції Human-in-the-Loop, яка передбачає

збереження активної ролі людини у процесі функціонування інтелектуальної системи. На відміну від повністю автономних рішень, такий підхід забезпечує контроль якості, адаптивність та підвищує довіру до результатів роботи ІІІ [12].

Взаємодія між людиною та ІІІ організована у вигляді багаторівневого циклу. На оперативному рівні відбувається безпосередня взаємодія у межах окремої задачі: користувач формує запит, система генерує результат, після чого здійснюється оцінка якості. У разі невідповідності результату очікуванням відбувається уточнення запиту та повторна генерація. Формально цей процес описується як ітеративний цикл переходів між станами генерації та оцінювання.

На тактичному рівні розглядається повний цикл створення документа. Тут взаємодія включає узгодження різних версій, рецензування та інтеграцію результатів у загальну структуру документації. Результатом цього рівня є завершений документ та накопичений досвід у вигляді зауважень і коригувань.

Стратегічний рівень охоплює аналіз функціонування системи в цілому. На цьому рівні здійснюється оцінка ефективності використання ІІІ, коригування параметрів моделі, оновлення шаблонів та вдосконалення правил розподілу задач.

Ключовим елементом моделі є механізм зворотного зв'язку (feedback, FB), який формалізується у вигляді:

$$FB = \langle \text{тип задачі, оцінка якості, тип помилки, витрачений час на виправлення} \rangle$$

Накопичення та аналіз цих даних дозволяє системі адаптуватися, зменшувати кількість помилок та підвищувати ефективність роботи.

Для кількісної оцінки ефективності взаємодії людини та ІІІ вводиться система показників. Коефіцієнт прийняття з першої ітерації (2.2) визначається як відношення кількості задач, прийнятих без доопрацювання, до загальної кількості задач:

$$FPAR = \frac{\text{кількість задач, прийнятих без доопрацювання}}{\text{загальна кількість задач}} \times 100\% \quad (2.2)$$

Середня кількість ітерацій на задачу характеризує трудомісткість процесу за формулою

$$AIC = \frac{\sum_{i=1}^n k_i}{n}, \quad (2.3)$$

де k_i -кількість ітерацій для окремої задачі;

n – загальна кількість задач.

Коефіцієнт економії часу дозволяє оцінити ефект від використання ІІІ

$$TSR = 1 - \frac{T_{AI+Human}}{T_{Human_only}}, \quad (2.4)$$

де $T_{AI+Human}$ – час виконання задачі у гібридному режимі,

T_{Human_only} – час виконання тієї ж задачі без ІІІ.

2.3 Розробка методу оптимізації витрат ресурсів у процесі створення документації

Ефективне управління процесом створення документації неможливе без кількісної оцінки витрат, що виникають на кожному його етапі. З цією метою у підрозділі розробляється метод оптимізації ресурсів, що охоплює два взаємопов'язані аспекти: часові витрати учасників процесу та пов'язані з ними фінансові витрати.

Модель фінансових витрат будується як сума трьох складових, що мають різну природу та по-різному залежать від обраної стратегії розподілу задач. Перша складова це витрати на оплату праці учасників, є основною статтею у

традиційній моделі та скорочується пропорційно до обсягу задач, делегованих ІІІ. Друга складова це витрати на використання ІІІ-інструментів, що є відносно невеликою і визначається тарифікацією хмарних сервісів за обсягом оброблених даних. Третя складова це витрати на виправлення помилок, яку традиційні підходи до оцінки ефективності нерідко залишають поза увагою, однак саме вона відіграє вирішальну роль при порівнянні гібридної моделі з повною автоматизацією: чим більше критичних задач делегується ІІІ без належного контролю, тим більшими стають ці витрати.

На основі побудованих моделей часових та фінансових витрат формулюється задача оптимізації, мета якої полягає у знаходженні такого розподілу задач між людиною та ІІІ, який мінімізує зважену суму обох видів витрат при виконанні практичних обмежень. До таких обмежень належать: вимога щодо обов'язкового виконання людиною задач із високою критичністю помилки, обмеження на максимальне навантаження кожного учасника, а також бюджетні та часові обмеження конкретного проєкту. Вагові коефіцієнти у цільовій функції дозволяють гнучко налаштовувати пріоритети: для проєктів із жорсткими дедлайнами більша вага надається скороченню часу, для проєктів з обмеженим бюджетом це призводить скороченню фінансових витрат.

Для розв'язання сформульованої задачі пропонується покроковий алгоритм на основі індексу ASI, розробленого у підрозділі 2.2. На кожній ітерації задачам призначається виконавець відповідно до поточного порогового значення індексу, після чого перевіряється виконання всіх обмежень. Якщо обмеження порушені, то поріг підвищується і процес повторюється.

Для формалізації часових витрат процесу документування вводиться бінарна змінна x_i , що описує вибір виконавця для кожної задачі i : $x_i = 1$, якщо задачу виконує людина, $x_i = 0$, якщо задачу виконує ІІІ. Ця змінна була введена у підрозділі 2.2 у контексті індексу ASI та тут використовується як основна змінна задачі оптимізації.

Сукупні часові витрати процесу документування визначаються за формулою

$$T_{total}(x) = \sum_{i=1}^n [x_i \times T_i^H + (1 - x_i) \times T_i^{AI}]$$

де T_{total} – сукупні часові витрати;

T_i^H – час виконання задачі ілюдиною,

x_i – бінарна змінна x_i , що описує вибір виконавця для кожної задачі i ;

T_i^{AI} – сукупний час виконання задачі i із залученням ШІ, що включає підготовку запиту, генерацію результату та його перевірку людиною.

Слід зазначити, що формула 2.15 розраховує сумарні витрати у людино-годинах. У реальному проєкті частина задач може виконуватися паралельно, що додатково скорочує календарний час завершення процесу. Зокрема, поки ШІ виконує парсинг коду, розробник може паралельно готувати шаблон документа. Таким чином, автоматизація задає додатковий ефект за рахунок паралельності, але є важливим на практиці.

Числові значення параметрів T_i^H та T_i^{AI} для кожного етапу отримані на основі даних, зібраних у таблицях 2.3 та 2.4, і наведені у таблиці 2.8.

Таблиця 2.8 – Часові витрати на задачі процесу документування

Етап процесу	T_i^H , год	T_i^{AI} , год	Економія, год	ΔT_i , %
1	2	3	4	5
Збір даних	4	1,5	2,5	62,5
Планування структури	2	1,2	0,8	40
Генерація чернетки	5	2	3	60
Рецензування	3	2	1	33,3
Доопрацювання	2	1	1	50
Публікація	1	0.8	0,2	20

Фінансові витрати процесу документування складаються з трьох категорій, кожна з яких має різну природу та по-різному залежить від обраної стратегії розподілу задач.

Загальна модель фінансових витрат записується у вигляді

$$C_{total}(x) = C^H(x) + C^{AI}(x) + C^{err}(x) \quad (2.5)$$

де $C^H(x)$ – витрати на оплату праці учасників,

$C^{AI}(x)$ – витрати на використання ШІ-інструментів,

$C^{err}(x)$ – очікувані витрати на виправлення помилок, що виникають внаслідок некоректних результатів автоматизації.

Для порівняльного аналізу розглядаються три сценарії: традиційна модель без ШІ (сценарій А), повна автоматизація (сценарій Б) та запропонована гібридна модель (сценарій В). Результати розрахунку фінансових витрат за кожним сценарієм наведені у таблиці 2.9.

Щоб розрахувати фінансові витрати за кожним сценарієм, необхідно визначити вартісні параметри учасників процесу та інструментів, що використовуються. В межах даного дослідження розглядається команда, до складу якої входить розробник з погодинною ставкою 20 USD/год, що є типовим значенням для middle-senior спеціаліста на ринку праці України у грудні 2025 році за даними DOU.ua. Витрати на використання ШІ визначаються тарифікацією обраного LLM-сервісу і є відносно незначними порівняно з вартістю людської праці [13 – 14] .

При місячному фонді робочого часу 178 годин та курсі долара 44.2 грн/USD погодинна ставка у гривневому еквіваленті становить

$$c^H = 20 \times 44.2 = 884 \text{ грн/год}$$

Підставляючи погодинну ставку розробника у модель фінансових витрат разом із даними про витрати часу з таблиці 2.8 та очікуваними витратами на

виправлення помилок для кожного сценарію, отримуємо фінансові витрати, наведені у таблиці 2.9. Витрати на рецензування присутні в усіх трьох сценаріях, оскільки людський контроль є обов'язковим незалежно від ступеня автоматизації. Витрати на виправлення помилок у сценарії повної автоматизації є найвищими, що підтверджує висновок про недоцільність беззастережного делегування всіх задач ІІІ.

Таблиця 2.9 – Порівняння фінансових витрат за сценаріями

Стаття витрат	Сценарій А	Сценарій Б	Сценарій В
Оплата праці	15028 грн	-	9724 грн
Використання ІІІ	-	247 грн	118 грн
Виправлення помилок	480 грн	2764 грн	1018 грн
Рецензування	2652 грн	2652 грн	2652 грн
Разом	18160 грн	5663 грн	13512 грн

Оплата праці розраховується як добуток сумарного часу виконання задач людиною та погодинної ставки. У сценарії А всі задачі виконує розробник, тому витрати становлять 15028грн.

У гібридній моделі людина виконує задачі, що не були делеговані ІІІ відповідно до матриці розподілу таблиці 2.7, загальним обсягом 11 год та витратами 9724грн.

Рецензування присутнє в усіх трьох сценаріях, оскільки людський контроль результатів є обов'язковим незалежно від ступеня автоматизації. Час рецензування у автоматизованій моделі становить 3 год і витрати 2652грн.

Використання ІІІ розраховується на основі кількості оброблених токенів та тарифної ставки обраного LLM-сервісу. При повній автоматизації обробляється більший обсяг даних, ніж у гібридній моделі, що і пояснює різницю між 247 грн та 118 грн.

Виправлення помилок є найбільш ваговим, бо у сценарії повної автоматизації вона зростає майже у шість разів порівняно з традиційною

моделлю – з 480 грн до 2 764 грн, – що підтверджує недоцільність беззастережного делегування всіх задач ІІІ без урахування критичності кожної з них.

Діаграма на рисунку 2.4 демонструє, що сценарій Б має найменші загальні витрати, однак містить непропорційно велику частку витрат на виправлення помилок, тоді як гібридна модель В забезпечує прийнятний баланс між економією та якістю.

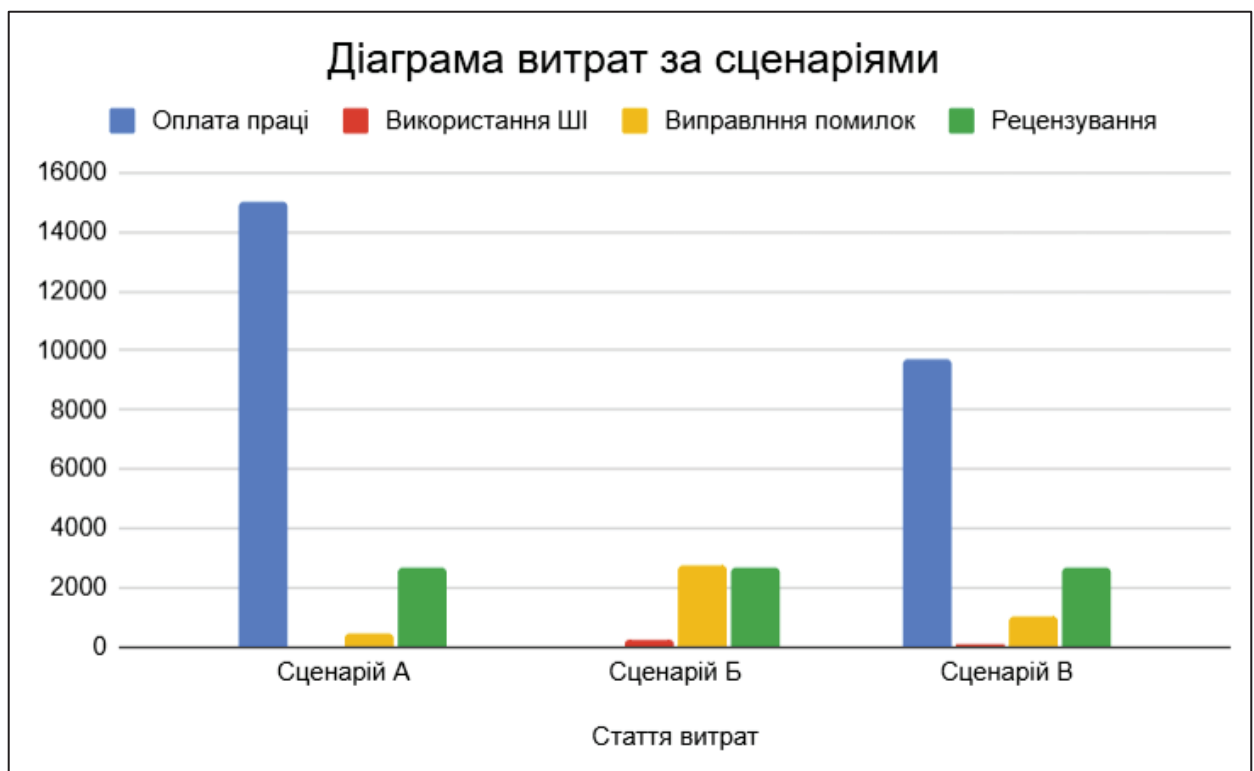


Рисунок 2.4 – Діаграма витрат за сценаріями

Аналіз даних таблиці 2.9 показує, що гібридна модель забезпечує скорочення загальних витрат на 25.6% порівняно з традиційною моделлю при збереженні прийнятного рівня витрат на виправлення помилок. Водночас повна автоматизація, попри найнижчі абсолютні витрати, супроводжується зростанням витрат на виправлення помилок майже у шість разів, що робить цей сценарій неприйнятним для проєктів із підвищеними вимогами до якості документації.

На основі розрахунку часових витрат на процес створення документації (2.4) та розрахунку загальної моделі фінансових витрат (2.5) формулюється задача вибору оптимального розподілу задач, що мінімізує зважену суму часових та фінансових витрат

$$Z(x) = \alpha \cdot \frac{T_{total}(x)}{T_0} + \beta \cdot \frac{C_{total}(x)}{C_0} \quad (2.6)$$

де Z – розподіл задач;

$\alpha + \beta = 1$ – вагові коефіцієнти, що відображають відносний пріоритет скорочення часу та вартості у конкретному проєкті;

T_0 та C_0 – відповідні витрати у традиційній моделі (сценарій А), що використовуються як нормуючі константи.

Нормування є принципово важливим, оскільки час і вартість мають різні одиниці виміру. Після нормування обидві складові є безрозмірними числами, що відображають частку від базових витрат. Наприклад, значення $Z = 0.65$ означає, що при обраному розподілі задач сумарні витрати становлять 65% від витрат традиційної моделі, тобто досягається скорочення на 35%.

Цільова функція (2.6) мінімізується з урахуванням системи практичних обмежень, що відображають реальні вимоги до процесу документування:

- задачі з критичністю помилки $E_i > 0.70$ виконуються людиною незалежно від значення ASI, що гарантує якість у найбільш відповідальних частинах процесу;
- сумарне навантаження кожного учасника не перевищує його доступного фонду робочого часу за аналізований період;
- загальні фінансові витрати не перевищують виділеного бюджету на документування;
- календарний час завершення процесу не перевищує встановленого дедлайну.

Для практичного розв'язання пропонується покроковий алгоритм, відображений на рисунку 2.5, на основі індексу ASI, розробленого у підрозділі 2.2, таблиці 2.6

Перевагою цього підходу є можливість його застосування без спеціалізованого програмного забезпечення при збереженні достатньої для практики точності результату:

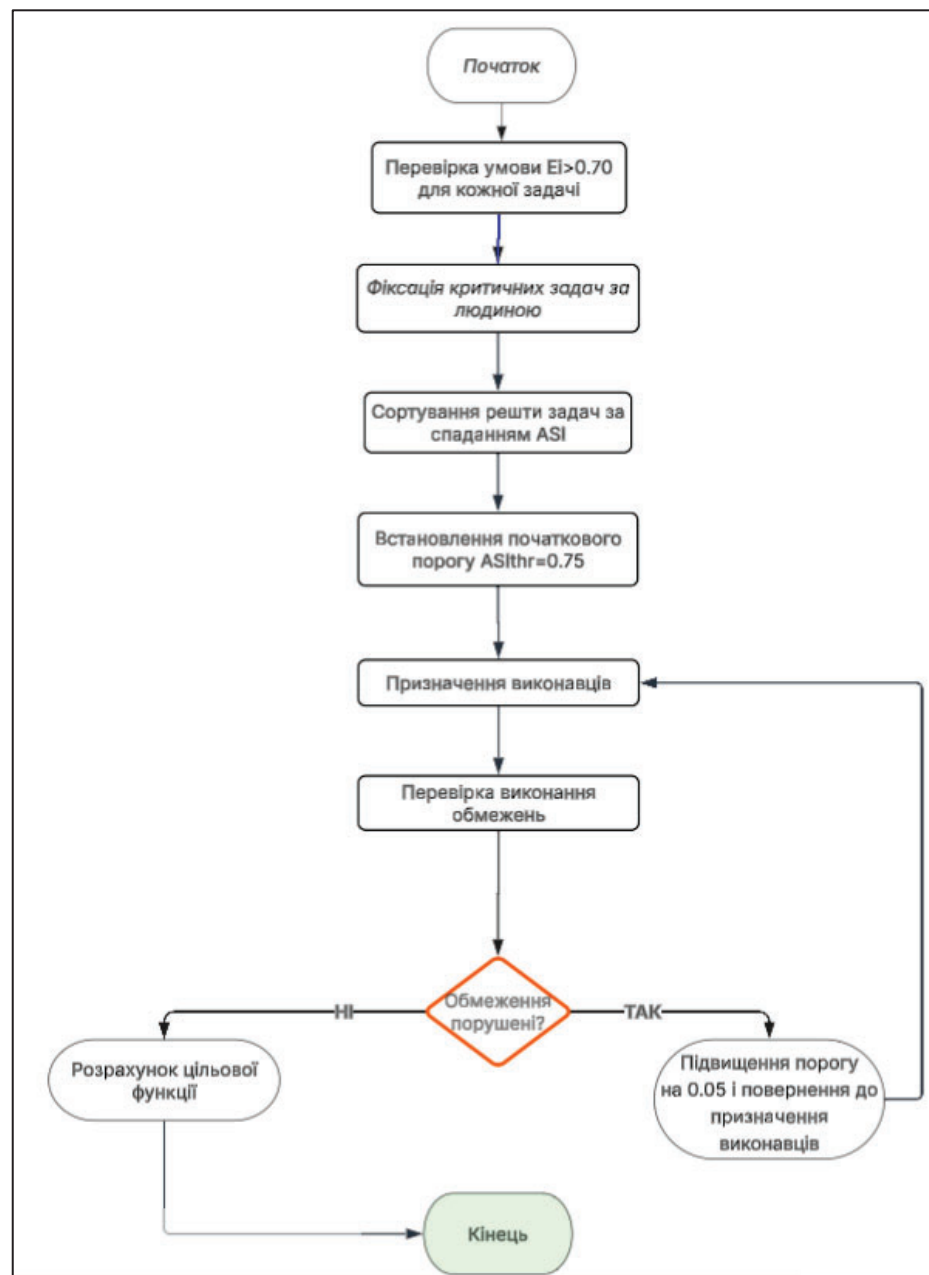


Рисунок 2.5 – Блок-схема алгоритму розподілу задач на основі індексу ASI

– першим етапом йде фіксація критичних задач. На першому кроці для всіх задач з $E_i > 0.70$ встановлюється $x_i = 1$ незалежно від значення ASI. Це обмеження є пріоритетним і не може бути знято навіть у випадку значної потенційної економії витрат. У розглянутому прикладі під це обмеження підпадають задачі «Опис архітектурних рішень» ($E = 0.70$) та «Рецензування змісту» ($E = 0.85$), що відповідає результатам матриці розподілу задач із підрозділу 2.2;

– другим етапом є впорядкування задач. Задачі, що не потрапили під обмеження першого кроку, впорядковуються за спаданням значення ASI. Задачі з вищим ASI розглядаються як пріоритетні кандидати на автоматизацію, оскільки вони є найбільш придатними до виконання ІІІ за сукупністю критеріїв формалізації, контекстності, критичності та повторюваності;

– третій етап це призначення виконавців, де встановлюється початкове порогове значення $ASI_{thr} = 0.75$. Задачам з $ASI_i \geq ASI_{thr}$ призначається ІІІ ($x_i = 0$), іншим призначається людина ($x_i = 1$). Вибір початкового порогу 0.75 обґрунтований таблицею 2.6 інтерпретації значень ASI, де це значення є нижньою межею діапазону повної автоматизації;

– четвертий етап це перевірка обмежень. На цьому етапі перевіряється виконання обмежень на навантаження розробника, бюджет та дедлайн. Якщо всі обмеження виконані, то алгоритм завершено. Якщо хоча б одне обмеження порушено, порогове значення ASI_{thr} підвищується на 0.05 і крок 3 повторюється. Підвищення порогу означає, що меншій кількості задач призначається ІІІ, що знижує навантаження на систему і зменшує ризики, однак одночасно зменшує і ефект від автоматизації;

– останнім етапом буде оцінка результату, де розраховується значення цільової функції Z , за формулою (2.6) та порівнюється з базовим значенням $Z = 1$, що відповідає традиційній моделі без ІІІ.

Застосування описаного алгоритму до задачі документування модуля авторизації при рівному пріоритеті часових та фінансових витрат ($\alpha = \beta =$

0.5) дає оптимальний розподіл задач за результатами алгоритму, наведений у таблиці 2.10.

Таблиця 2.10 – Оптимальний розподіл задач за результатами алгоритму

Задача	ASI	Ei	Рішення	Обґрунтування
Парсинг коду	0,88	0,25	ШІ	$ASI \geq 0.75$, $E \leq 0.70$
Перевірка форматування	0,89	0,20	ШІ	$ASI \geq 0.75$, $E \leq 0.70$
Генерація структури	0,64	0,40	Людина	$ASI < 0.75$
Написання API	0,74	0,50	Людина	$ASI < 0.75$
Опис архітектури	0,22	0,70	Людина	$E > 0.70$
Актуалізація	0,86	0,30	ШІ	$ASI \geq 0.75$, $E \leq 0.70$
Рецензування змісту	0,27	0,85	Людина	$E > 0.70$

Побудована структурно-функціональна модель процесу документування, формалізує процес створення документації як керовану систему з визначеними станами, учасниками та інформаційними потоками

Розроблена модель розподілу задач між людиною та ШІ на основі індексу ASI, що враховує формалізованість, контекстність, критичність та повторюваність кожної задачі. Вагові коефіцієнти визначено методом попарних порівнянь.

Розроблений метод оптимізації витрат ресурсів, що поєднує модель часових витрат і модель фінансових витрат з урахуванням витрат на оплату праці, використання ШІ та виправлення помилок.

3 ДОСЛІДЖЕННЯ ОСОБЛИВОСТЕЙ РЕАЛІЗАЦІЇ ТЕХНОЛОГІЇ СТВОРЕННЯ ДОКУМЕНТАЦІЇ В ІТ-ПРОЄКТІ РОЗРОБКИ ПЛАТФОРМИ ГЕНЕРАЦІЇ ЦИФРОВОГО КОНТЕНТУ

3.1 Реалізація технології створення допоміжної документації в ІТ-проєкті розробки платформи генерації цифрового контенту

Платформа генерації цифрового контенту об'єднує кілька незалежних модулів для автоматичного створення та обробки різних типів цифрового контенту, таких як генерація зображень і тексту, обробка аудіо-файлів та інше. Кожен модуль вирішує окрему прикладну задачу та має власну архітектуру, набір залежностей і специфічні вимоги до його документації. Загальна архітектура платформи представлена на рисунку нижче.(див.рис.3.1)

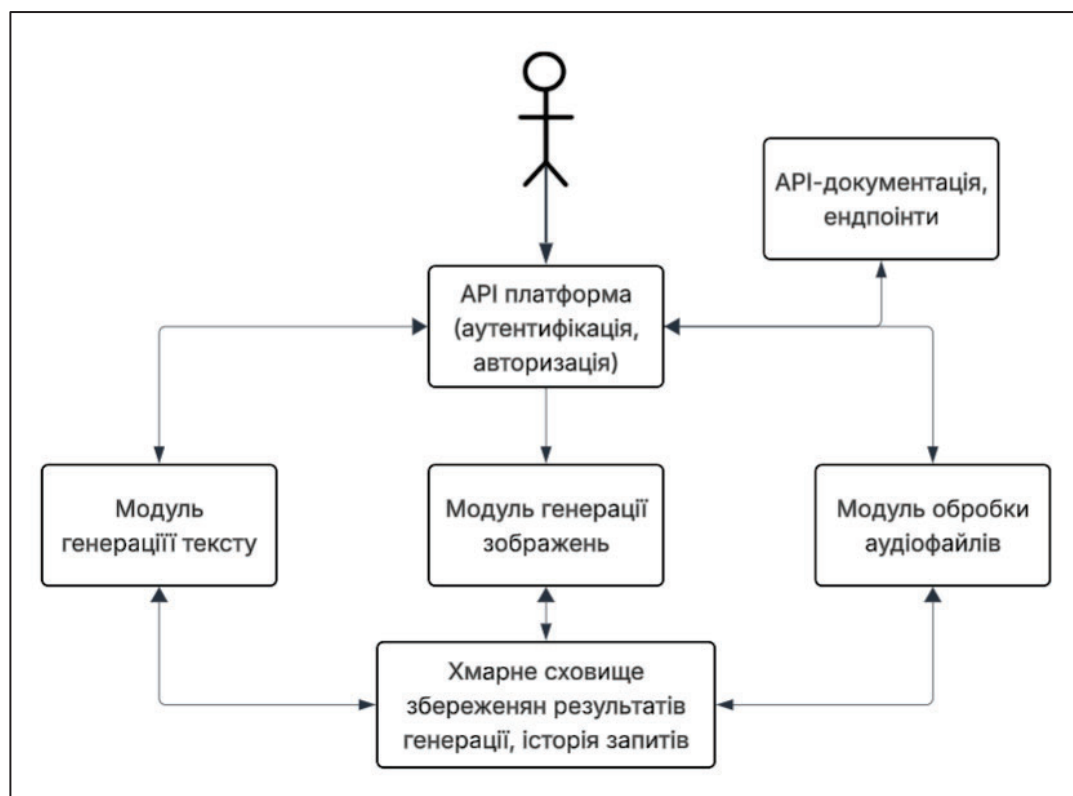


Рис.3.1 – Архітектура платформи генерації цифрового контенту

Модуль генерації тексту створює текстовий контент на основі запитів користувача. Технічно модуль реалізується через інтеграцію з великими мовними моделями, що приймають на вхід текстовий запит та набір параметрів генерації: температуру, максимальну довжину, контекстне вікно і повертають згенерований текст.

Модуль генерації зображень реалізує функціональність створення графічного контенту на основі текстових описів, інтегрується з генеративними моделями дифузійного типу, що приймають текстовий опис бажаного зображення.

Модуль обробки аудіофайлів забезпечує автоматичне перетворення аудіозаписів на текстові транскрипції, де приймає аудіофайли у різних форматах, обробляє їх через модель автоматичного розпізнавання мовлення та повертає структурований текст із можливим розбиттям на часові мітки та визначенням мовців.

Усі три модулі взаємодіють із зовнішнім середовищем через єдину API платформу, що є спільною точкою входу для клієнтських застосунків. Вона об'єднує функціональність усіх модулів за єдиним інтерфейсом та забезпечує аутентифікацію, авторизацію і управління лімітами запитів.

Описана архітектура платформи має деякі специфічні особливості, що впливають на процес написання документації:

- висока частота змін у модулях генерації, бо документація модулів генерації тексту та зображень потребує оновлення значно частіше, ніж документація модуля обробки аудіо;
- зміна API без своєчасного оновлення документації призводить до порушення роботи клієнтських застосунків, що є прямою фінансовою шкодою для замовника. Це визначає найвищу критичність помилки $E_i = 0.95$ для задач документування API, що автоматично закріплює їх за розробником згідно з обмеженням $E_i > 0.70$.

З урахуванням архітектурних особливостей та специфіки модулів визначається перелік типів документації, що підлягають створенню. Кожен

тип документа має власний шаблон у базі знань системи та характеристики з точки зору критичності помилки та режиму роботи ІІІ.

Таблиця 3.1 – Типи документації платформи генерації цифрового контенту

Тип документа	Модуль платформи	E_i	ASI	Режим ІІІ-асистента
Опис параметрів генерації тексту	Генерація тексту	0.8	0.34	Валідація людиною
Опис параметрів генерації зображень	Генерація зображень	0.85	0.28	Валідація людиною
Опис форматів вхідних даних аудіо	Обробка аудіофайлів	0.6	0.6	Асистент
Опис обмежень модуля обробки аудіо	Обробка аудіофайлів	0.65	0.54	Асистент
Документація API	Всі модулі	0.95	0.18	Валідація людиною
Опис внутрішніх фіч модулів	Всі модулі	0.5	0.76	Авто
Інструкція розгортання проекту	Архітектура	0.65	0.55	Асистент

Бачимо, що документи з найвищою критичністю помилки це опис параметрів генеративних моделей та документація API, що закріплюються за розробником відповідно до обмеження $E_i > 0.70$ задачі оптимізації. Документи із середньою критичністю виконуються в асистивному режимі, а опис внутрішніх функцій модулів, виконується у автоматичному режим ІІІ.

3.2 Реалізація етапу «Збір та аналіз вихідних даних»

Репозиторій вихідного коду є основним джерелом технічної інформації. Система автоматично аналізує вихідний код модулів платформи,

конфігураційні файли генеративних моделей та специфікації API у форматі OpenAPI. Для кожного з трьох модулів платформи парсер коду застосовує відповідну стратегію аналізу: для файлів Python та JavaScript – аналіз синтаксичного дерева з метою витягу функцій та їх сигнатур, для конфігураційних файлів використовує структурний парсинг параметрів генеративних моделей, для специфікацій API – аналіз ендпоінтів та форматів запитів і відповідей. [15] .

Таск-менеджер, такий як Jira чи Asana, є джерелом організаційної інформації у вигляді описів функціональних вимог поточного спринту, коментарів команди та зв'язків між задачами і компонентами коду. Для поточного IT-проекту особливо важливими є задачі, пов'язані зі зміною параметрів генеративних моделей або оновленням API, оскільки вони є прямими тригерами для ініціації документування відповідних компонентів [16 – 17] .

Існуюча документація використовується для збереження термінологічної узгодженості між версіями. Система звертається до попередніх версій документів та глосарію проекту, що містить специфічні терміни предметної області, такі як параметри генеративних моделей, формати контенту, назви компонентів платформи.

Кожен модуль платформи має власну специфіку збору вихідних:

- модуль генерації тексту, для нього система аналізує конфігураційний файл мовної моделі: назву моделі, параметри генерації за замовчуванням, обмеження на обсяг тексту;
- модуль генерації зображень є найбільш складним через збір даних та підтримувані розміри зображень;
- модуль обробки аудіофайлів, для котрого система збирає перелік підтримуваних мов та форматів файлів, обмеження на розмір і тривалість аудіо, а також параметри налаштування якості транскрипції.

Збір даних запускається автоматично при виникненні одного з трьох тригерів: фіксації змін у репозиторії коду, появи нової задачі у системі управління проєктом або ручного запиту розробника.

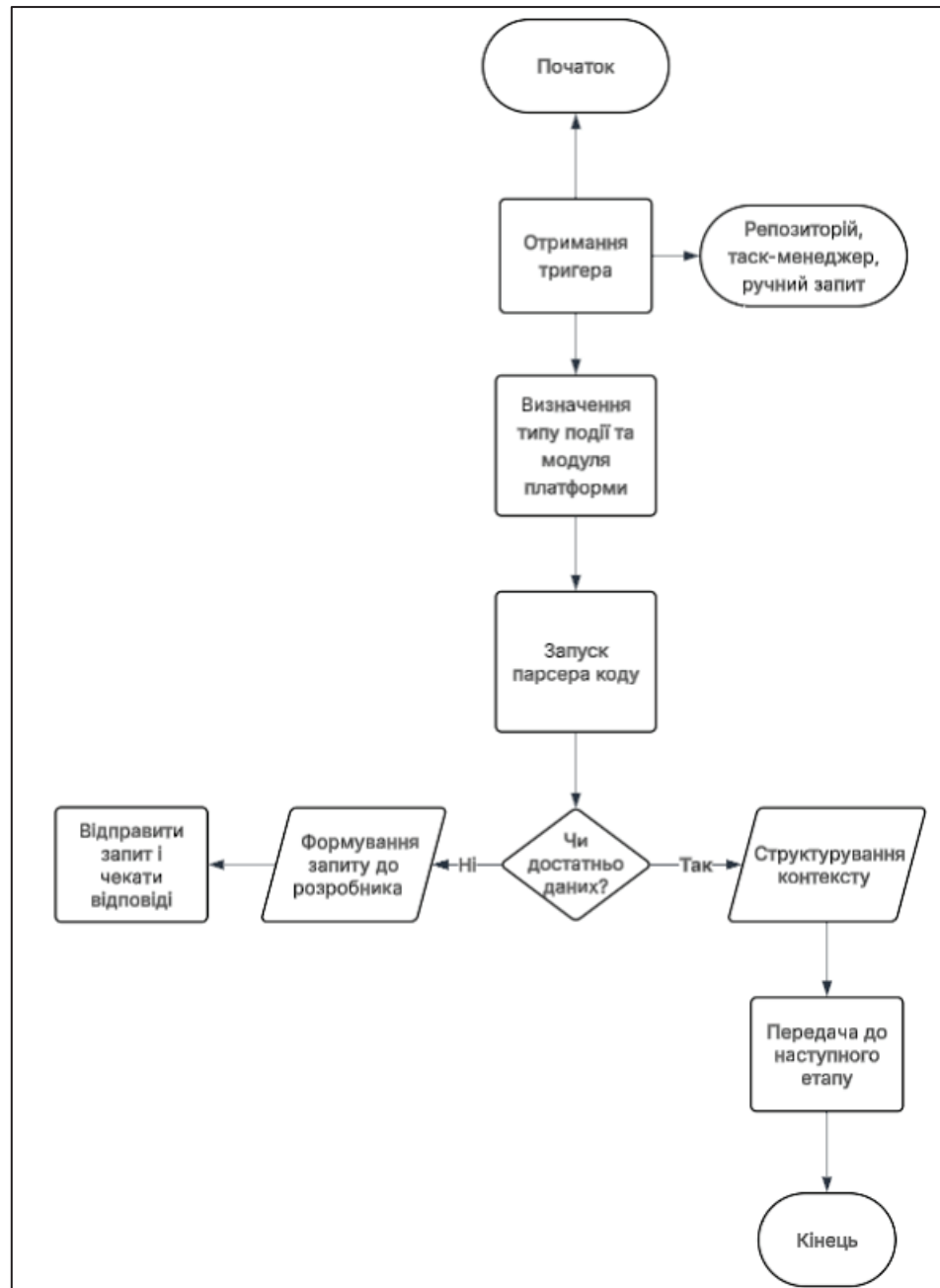


Рис 3.2 – Алгоритм збору вихідних даних

Після отримання тригера система визначає тип події та відповідний модуль платформи. Залежно від типу змін ініціюється документування

конкретного компонента: зміна конфігурації генеративної моделі спричиняє оновлення опису її параметрів, зміна специфікації API – оновлення API-документації, зміна вихідного коду функцій – оновлення опису відповідних функцій.

На етапі закінчення автоматичного збору, система перевіряє повноту даних шляхом порівняння з переліком обов'язкових елементів для відповідного типу документа з таблиці 3.1. За результатами перевірки формується структурований контекст документування, що передається до наступного етапу планування структури.

3.3 Реалізація етапу «Планування структури документації»

Основою планування структури документу є база знань шаблонів, котрий представляє себе як набір попередньо визначених структур для кожного типу документації платформи. Вибір шаблону відбувається автоматично на основі типу модуля та події, що ініціювала документування.

Для платформи генерації цифрового контенту визначено чотири базові шаблони:

- шаблон опису генеративної моделі використовується для документування модулів генерації тексту та зображень. Структура включає в себе призначення моделі, перелік параметрів із допустимими значеннями, формат вхідних даних, формат вихідних даних, приклади використання та відомі обмеження;
- шаблон опису API має загальний опис ендпоінту, метод та шлях запиту, параметри запиту з типами та обмеженнями, формат відповіді, коди помилок та приклади запитів і відповідей;
- шаблон опису модуля обробки даних для документування модуля обробки аудіофайлів (призначення модуля, підтримувані формати вхідних

даних, обмеження на розмір та тривалість, опис функцій модуля та приклади використання);

– шаблон опису внутрішніх функцій (назва та призначення функції, перелік параметрів із типами, повернене значення та приклад виклику).

Після вибору базового шаблону ІІІ в асистивному режимі адаптує його під специфіку конкретного модуля на основі структурованого контексту, отриманого з попереднього етапу. Адаптація полягає у наповненні шаблону конкретними назвами розділів, що відповідають параметрам модуля, та визначенні обов'язкових і необов'язкових розділів залежно від наявності відповідних даних у контексті.

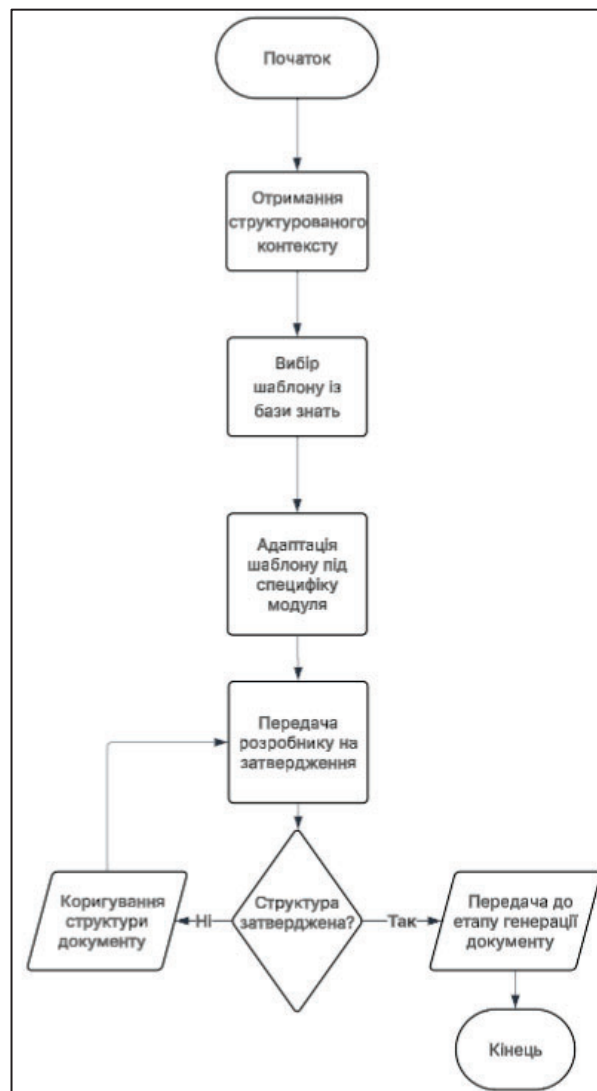


Рис 3.3 – Алгоритм генерації структури документа

Планування структури реалізується у три послідовні кроки(див.рис.3.3):

- перший крок, на котрому система визначає тип документа на основі події, що надійшла з попереднього етапу, та обирає відповідний базовий шаблон із бази знань.
- на другому кроці ІІІ адаптує шаблон під специфіку конкретного модуля платформи, наповнюючи його конкретними назвами розділів на основі структурованого контексту.
- на останньому кроці запропонована структура передається розробнику на затвердження. Він може прийняти структуру без змін, скоригувати перелік чи порядок розділів, або запросити альтернативний варіант структури. Затверджена структура передається до наступного етапу генерації чернетки.

3.4 Реалізація етапу «Генерація чернетки документації»

Якість згенерованої чернетки залежить від якості промπτу, що передається ІІІ-агенту. Він формується автоматично на основі трьох компонентів:

- системна інструкція визначає роль ІІІ-агента, стиль викладення та загальні вимоги до документа: технічна точність, використання термінології глосарію проєкту, структурованість та стислість формулювань;
- затверджена структура визначає перелік розділів, що мають бути присутні у документі, та обов'язкові елементи змісту кожного розділу;
- структурований контекст містить конкретні технічні дані: назви функцій, параметри генеративних моделей, формати вхідних та вихідних даних. Саме цей компонент забезпечує технічну коректність згенерованого тексту.

Для різних типів документів застосовуються різні стратегії генерації, що визначаються специфікою модуля та критичністю помилки E_1 :

Для модуля генерації тексту використовується покрокова стратегія генерації, де спочатку генерується загальний опис моделі та її призначення, після чого для кожного параметра генерації окремо формується опис його призначення та допустимих значень.

Для модуля генерації зображень застосовується розширена стратегія генерації з обов'язковим включенням пояснень практичного впливу параметрів на результат. ШІ-агент генерує не лише технічний опис параметра, а й пояснення того, як його зміна впливає на якість та характеристики згенерованого зображення.

Для модуля обробки аудіофайлів застосовується стандартна стратегія генерації з акцентом на опис обмежень та можливих помилок.

Для документації API використовується підійде стратегія генерації, що суворо дотримується формату OpenAPI. Для кожного ендпоінту окремо генерується опис методу, параметрів запиту, можливих відповідей та кодів помилок.

На рисунку 3.4 представлений алгоритм генерації чернетки. Наступним кроком після введення пром프트, ШІ-агент генерує текст документа відповідно до затвердженої структури. Система автоматично перевіряє наявність усіх обов'язкових розділів у згенерованому тексті. Якщо певний розділ відсутній або є надто коротким відносно встановленого мінімального обсягу, промпт автоматично уточнюється і генерація цього розділу повторюється окремо.

Після успішної генерації всіх розділів система фіксує метадані: час генерації, використану стратегію, версію ШІ-моделі та кількість ітерацій уточнення.

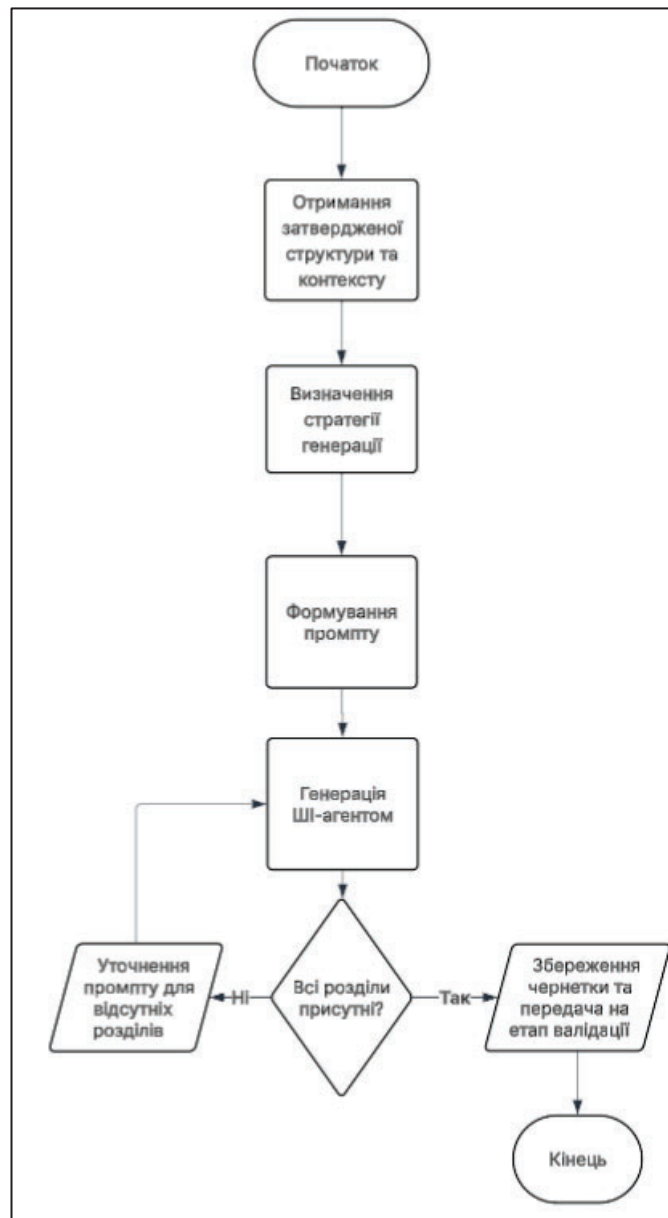


Рис 3.4 – Алгоритм генерації чернетки

3.5 Реалізація етапу «Рецензування та доопрацювання документації»

Рецензування та доопрацювання є 4 і 5 етапом в технології створення документації. Відповідно до матриці розподілу задач(табл.2.7), рецензування являє собою задачу із критичністю 0.85, що автоматично закріплює її за розробником згідно обмеження $E_i > 0.7$. Етап доопрацювання виконується в

асистивному режимі, де ШІ пропонує варіанти виправлень, а розробник їх вже коригує чи затверджує.

До передачі чернетки розробнику на рецензування система виконує автоматичну валідацію якості. Це дозволяє усунути формальні невідповідності ще до залучення розробника, що скорочує час рецензування та дозволяє зосередитись на змістовій перевірці. Алгоритм відображено на рисунку 3.5.

Автоматична валідація перевіряє чернетку за чотирма критеріями послідовно:

- структурна повнота, котру ШІ перевіряє на наявність усіх обов'язкових розділів відповідно до затвердженої структури. Якщо певний розділ відсутній або є порожнім, система повертає чернетку на доопрацювання до ШІ-агента без залучення розробника;
- термінологічна узгодженість – система перевіряє, чи використовуються у документі затверджені назви параметрів генеративних моделей та компонентів платформи;
- повнота технічного опису. Перевірка на актуальність, чи описані всі елементи, що були виявлені парсером коду на етапі збору даних.
- відсутність плейсхолдерів, пошук незаповнених полів шаблону типу «вставити опис» або «TO DO».

Після успішної автоматичної валідації чернетка передається розробнику на змістове рецензування. Розробник перевіряє технічну коректність документа.

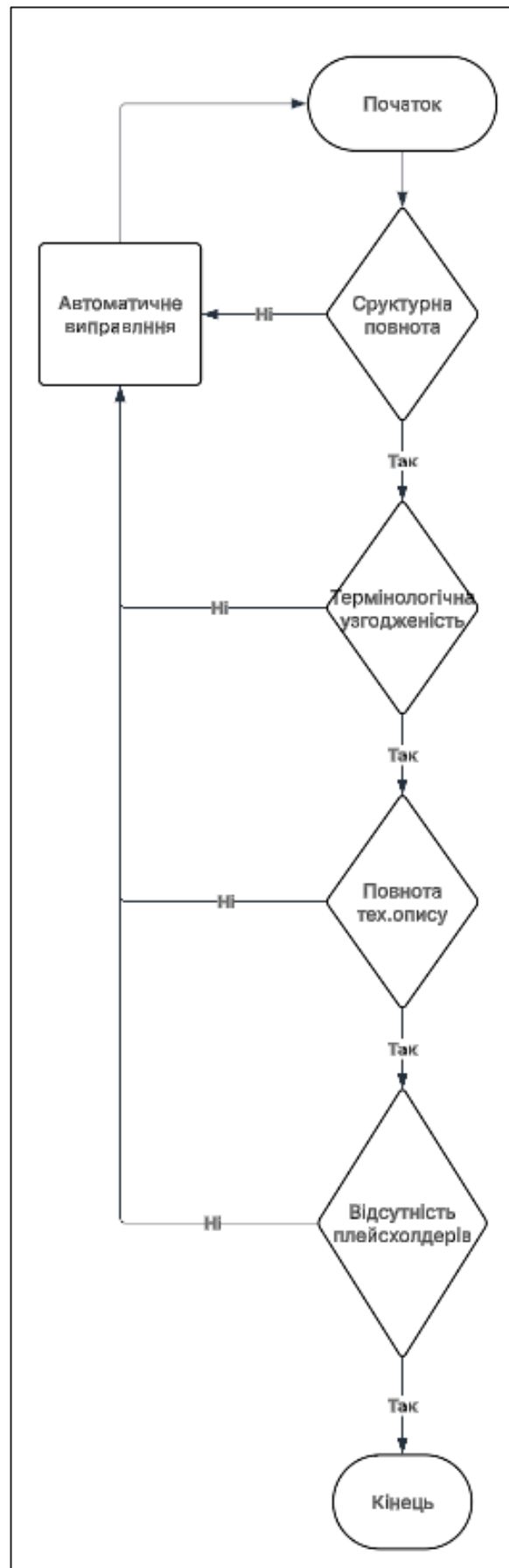


Рисунок 3.5 – Алгоритм автоматичної валідації

За результатами рецензування розробник формує перелік зауважень із зазначенням типу кожного: технічна неточність, неповний опис, некоректний приклад або термінологічна помилка.

Після отримання зауважень від розробника система переходить до етапу доопрацювання, алгоритм відображено на рисунку 3.6. Відповідно до матриці розподілу задач, доопрацювання виконується в асистивному режимі з $ASI=0.42$

Для кожного зауваження ІІІ-агент формує варіант виправлення на основі типу зауваження та контексту відповідного розділу. Розробник розглядає запропоноване виправлення та приймає одне з трьох рішень: затвердити запропонований варіант, скоригувати його або відхилити і внести правку вручну.

Зауваження та прийняті рішення фіксуються у журналі якості як частина механізму зворотного зв'язку та допомагають системі покращувати якість генерації аналогічних документів у майбутніх ітераціях, виявляючи помилки для конкретних типів документів платформи.

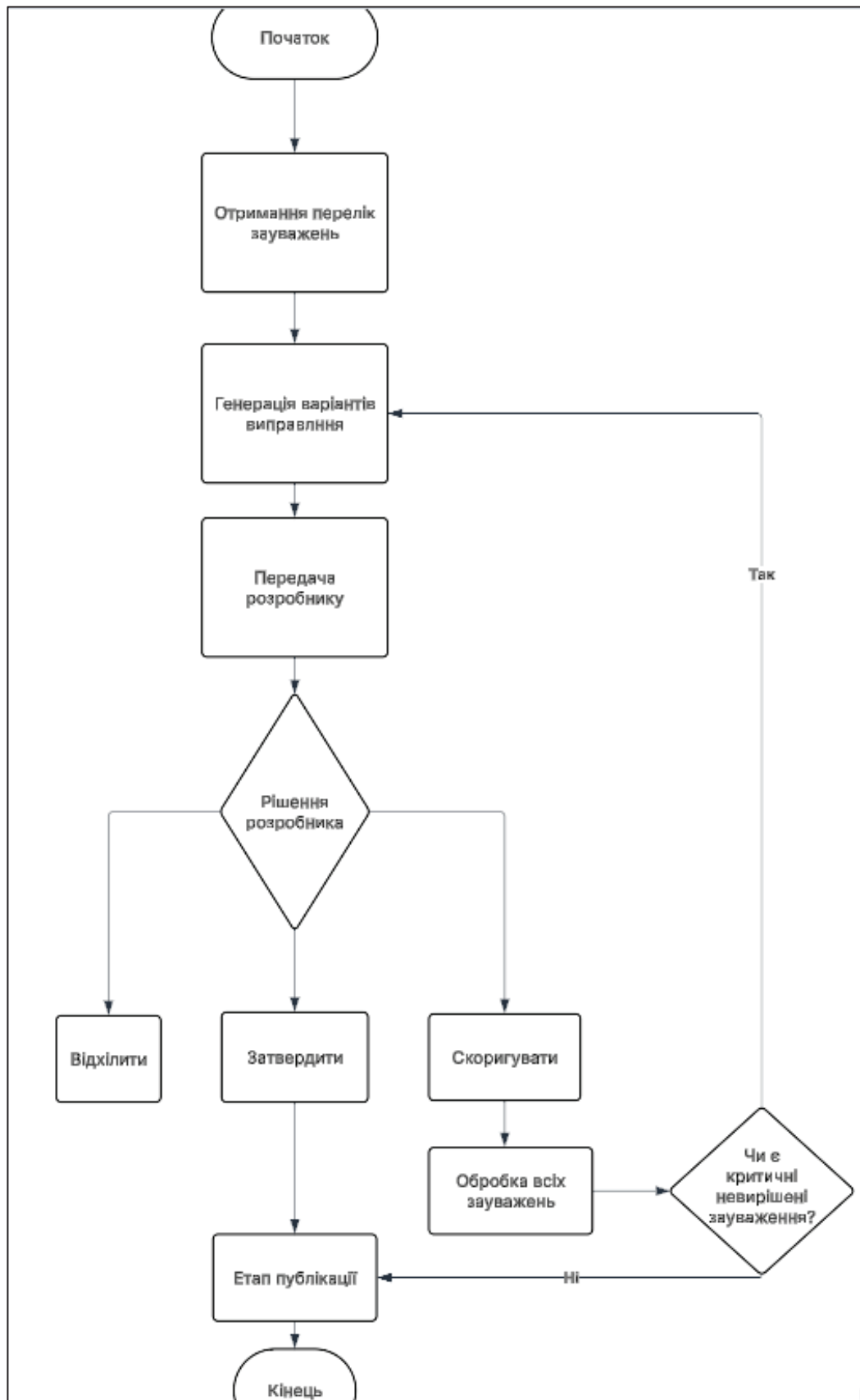


Рисунок 3.6 – Алгоритм допрацювання документації

3.6 Реалізація етапу «Публікація документації».

Після затвердження документа на етапі рецензування система виконує його автоматичну публікацію у сховищі документації проєкту, по типу Confluence. Публікація включає в себе декілька послідовних дій.

По-перше, система автоматично форматує документ відповідно до корпоративних стандартів оформлення – застосовує визначені стилі заголовків, форматування коду та таблиць, додає стандартний колонтитул із датою створення, версією документа та посиланням на відповідний модуль у репозиторії.

По-друге, документ розміщується у визначеній структурі сховища відповідно до типу модуля платформи. Документація модуля генерації тексту, генерації зображень та обробки аудіофайлів зберігається у відповідних розділах сховища, що забезпечує зручну навігацію для розробників.

По-третє, система автоматично оновлює навігаційні посилання – індексну сторінку розділу та перехресні посилання між пов'язаними документами. Наприклад, при публікації опису нового параметра генеративної моделі система автоматично оновлює посилання на цей параметр у документації API, якщо воно там присутнє.

Ведення версій документів також існує на етапі публікації. Кожна нова версія документа отримує унікальний номер версії, що відповідає версії відповідного компонента платформи. Попередня версія документа не видаляється, а архівується із відповідним позначенням. Це дозволяє розробникам, що працюють зі старішими версіями платформи, звертатися до актуальної для них документації.

При публікації нової версії документа система автоматично формує короткий журнал змін, що містить перелік оновлених розділів та стислий опис внесених змін.

Після публікації документ переходить у режим моніторингу, де відбувається систематичне відстеження змін у відповідних компонентах платформи для своєчасного виявлення застарілих розділів. Відповідно до матриці розподілу задач, моніторинг має значення $ASI=0.86$ і виконується у автоматичному режимі III.

Система відстежує зміни у репозиторії коду та порівнює їх із змістом опублікованих документів. При виявленні розбіжності між кодом та документацією система класифікує її за рівнем критичності.(див.рис.3.7)

Критична розбіжність виникає, коли змінився параметр генеративної моделі або ендпоінт API. У цьому випадку система негайно надсилає сповіщення відповідальному розробнику та позначає відповідний розділ документа як застарілий. Розробник отримує сповіщення із зазначенням конкретного розділу, що потребує оновлення, та посиланням на відповідну зміну у репозиторії.

Некритична розбіжність виникає при незначних змінах, що не впливають на функціональність описаного компонента. У цьому випадку система фіксує розбіжність у журналі моніторингу без негайного сповіщення та включає її до планового переліку оновлень наступного спринту.



Рисунок 3.7 – Цикл моніторингу актуальності документації

3.7 Висновки з реалізації технології створення допоміжної документації

У результаті дослідження особливостей реалізації технології створення допоміжної документації в ІТ-проєкті розробки платформи генерації цифрового контенту було визначено специфіку платформи як об'єкта документування.

Етап збору даних виконується в автоматичному режимі на основі трьох джерел: репозиторію вихідного коду, системи управління задачами та існуючої документації

На етапі планування структури в асистивному режимі, визначено чотири базові шаблони документів для модулів платформи та описано механізм їх адаптації під специфіку конкретного модуля з обов'язковим затвердженням розробником.

Генерація чернетки має чотири стратегії генерації для різних типів документів платформи з урахуванням специфіки кожного модуля та критичності помилки. Описано механізм формування промпту та автоматичної перевірки повноти згенерованого тексту.

Знайдено реалізацію етапів рецензування та доопрацювання як дворівневої системи контролю якості: автоматична валідація за чотирма критеріями до передачі розробнику та змістове рецензування розробником з урахуванням специфіки кожного типу документа платформи.

Етап публікації включає автоматичне форматування та розміщення документа, версіонування із формуванням журналу змін та безперервний моніторинг актуальності документації.

Реалізація технології підтвердила застосовність розроблених у розділі 2 моделей та методів до конкретного ІТ-проєкту з урахуванням специфіки платформи генерації цифрового контенту. Отримані результати є основою для апробації та кількісної оцінки ефективності у розділі 4.

4 АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ НА ПРИКЛАДІ ІТ-ПРОЕКТУ РОЗРОБКИ ПЛАТФОРМИ ГЕНЕРАЦІЇ ЦИФРОВОГО КОНТЕНТУ

4.1 Підготовка даних для апробації

Для апробації розробленого методу управління роботами зі створення допоміжної документації обрано ІТ-команду з семи учасників, що працює над розробкою платформи генерації цифрового контенту. До складу команди входять чотири розробники, два тестувальники та один менеджер проєкту. Команда працює за методологією Scrum із двотижневими спринтами.

Розроблений метод створення допоміжної документації складається з таких етапів:

- «Збір та аналіз вихідних даних» (етап 1);
- «Планування структури документації» (етап 2);
- «Генерація чернетки документації» (етап 3);
- «Рецензування документації» (етап 4);
- «Доопрацювання документації» (етап 5);
- «Публікація документації» (етап 6).

Апробація проводилась протягом двох послідовних спринтів. У першому спринті документування велось за традиційною моделлю без використання ШІ, де розробники самостійно створювали документацію після завершення кожного модуля. Це дозволило зафіксувати базові показники витрат часу та фінансових витрат для подальшого порівняння. У другому спринті застосовувалась розроблена технологія документування з використанням ШІ відповідно до реалізації, описаної у розділі 3.

Збір даних проводився комбінованим методом, що включав фіксацію витрат часу на кожному етапі документування у системі управління задачами, нотатуванням кількості ітерацій уточнення при генерації кожного документа засобами ШІ, а також фіксацією зауважень розробників і тестувальників за

результатами рецензування. Менеджер проєкту здійснював загальний контроль дотримання процесу документування протягом обох спринтів та надавав експертну оцінку якості отриманих документів. Загальна кількість документів, створених протягом двох спринтів, наведена у таблиці 4.1.

Таблиця 4.1 – Обсяг документування в рамках апробації

Модуль платформи	Кількість функцій	Кількість документів	Спринт
1	2	3	4
Генерація тексту	7	3	1
Генерація зображень	5	3	2
Обробка аудіофайлів	4	2	2
Зовнішній API	0	2	1
Сума	16	10	

Для розрахунку показників ефективності у підрозділі 4.2 визначено конкретні значення параметрів моделі відповідно до умов проєкту, що наведені у таблиці 4.2.

Таблиця 4.2 – Параметри моделі апробації

Параметр	Значення	Джерело
1	2	3
Погодинна ставка розробника	884грн/год	Рзрахунок формули
Сумарно робочих годин в місяць	178 год	Умови проєкту
Курс USD/UAH	44.2	Ринковий курс
Вартість III API	0.0003грн/токен	Тариф сервісу

4.2 Результат апробації розробленого методу ефективності створення документації

На протязі двох спринтів, було отримано результати апробації про витрати часу на створення документації та її фінансові витрати. Перший спринт відповідав сценарію А, де процес документації відбувався лише за допомогою людських ресурсів. Другий спринт йшов за сценарієм В, коли використовувалась гібридна модель у вигляді розробник + ШІ-агент.

На першому спринті створення документації велось за сценарієм А. Розробники самостійно виконували всі етапи документування від збору даних до публікації. Тестувальники залучались на етапі рецензування для перевірки коректності описів модулів з точки зору тестування, а менеджер проєкту здійснював загальний контроль дотримання термінів.

За результатами першого спринту, відображених у таблиці 4.3, зафіксовані витрати часу на кожному етапі для всіх п'ятьох документів

Таблиця 4.3 – Фактичні витрати часу у першому спринті за сценарієм А

Етап документування	Розробник, год	Тестувальник, год	Сума витрачених годин за етапом
1	2	3	4
Збір даних	4	0,5	4,5
Планування структури	2	0,5	2,5
Генерація чернетки	5	0,5	5,5
Рецензування	2	1	3
Доопрацювання	2	0,5	2,5
Публікація	1	0,5	1,5
Разом	16	3,5	19,5

Порівнюючи дані з фактичних витрат часу у сценарії А, можна побачити, що в середньому при традиційному підході відповідальний розробник за спринт витрачає близько 16 годин на задачі документування.

Протягом першого спринту на етапі рецензування зафіксовано два випадки виявлення помилок у документації (опис параметра модуля генерації зображень та документація API), виправлення помилок зайняло додатково одну годину.

На другому спринті створення документації про модулі обробки аудіо-файлів та генерації зображень велось за гібридним сценарієм В. Розробники і тестувальники делегували частину етапів документації ІІІ-агенту. Менеджер проєкту здійснював загальний контроль дотримання термінів.

За результатами другого спринту, відображених у таблиці 4.4, зафіксовані витрати часу на кожному етапі для всіх п'ятьох документів

Таблиця 4.4 – Фактичні витрати часу у другому спринті за сценарієм В

Етап документування	Розробник, год	Тестувальник, год	ІІІ-агент, год	Сума витрачених годин за етапом
1	2	3	4	5
Збір даних	0,5	0	1	1,5
Планування структури	0,5	0	0,5	1
Генерація чернетки	0,5	0,5	1	2
Рецензування	2	1	0	3
Доопрацювання	0,5	0	0,5	1
Публікація	0,5	0	0	0,5
Разом	4,5	1,5	3	9

За результатами аналізу витрат часу визначено, що у данному спринті відбувся суттєвий перерозподіл навантаження між учасниками проєкту. На

72% скоротилися витрати часу розробників, та на 2 години менше навантаження отримали тестувальники.

У таблиці 4.5 наведено порівняння витрат часу за протязі двох спринтів. Розрахунок $T_{\text{факт}}$, та $T_{\text{теор}}$ здійснювалася за формулою (2.1).

Таблиця 4.5 – Порівняння фактичних часових витрат за двома спринтами

Етап документування	Спринт 1, год	Спринт 2, год	Економія часу, год	$T_{\text{факт}}, \%$	$T_{\text{теор}}, \%$
Збір даних	4,5	1,5	3	66,7	62,5
Планування структури	2,5	1	1,5	60	40
Генерація чернетки	5,5	2	3,5	63,6	60
Рецензування	3	3	0	0	0
Доопрацювання	2,5	1	1,5	60	50
Публікація	1,5	0,5	1	66,7	20
Разом	19,5	9	10,5	-	-

Аналізуючи дані таблиці 4.5, можна зробити висновок, що найбільш ефективним за економією часу є етап збору даних та генерації чернетки.

На етапі рецензування скорочення часу не відбулось, оскільки цей етап повністю виконується людиною і не підлягає автоматизації із-за обмеження $E_i > 0.7$.

Наступним кроком був розрахунок фінансових витрат у проекті за формулою (2.5) та відображення результатів у таблиці 4.6

Таблиця 4.6 – Порівняння фактичних фінансових витрат за спринтами

Стаття витрат	Спринт 1	Спринт 2
1	2	3
Оплата праці(розробник та тестувальник)	17 238 грн	5304 грн

Кінець таблиці 4.6

1	2	3
Рецензування	2652 грн	2653 грн
Використання ШІ	0	119 грн
Виправлення помилок	1326 грн	442 грн
Разом($C_0 C_{total}$)	21 216 грн	8517 грн

Роблячи підсумок порівняння витрат за двома спринтами бачимо, що фактичне скорочення витрат відбулося на 59,8%.

Останнім кроком апробації є оцінка ефективності гібридної моделі документування (сценарій В), через розрахунок цільової функції. Для розрахунку використовувались фактичні дані, отримані за результатами двох спринтів.

При рівному пріоритеті часових та фінансових витрат, де $\alpha = \beta = 0.5$, підставимо значення у формулу (2.6):

$$Z(x) = 0.5 \times \frac{9}{19,5} + 0,5 \times \frac{8517}{21216} = 0,431, \quad (4.1)$$

де Z – цільова функція, що дозволяє оцінити ефективність запропонованого методу.

Результат розрахунку означає, що витрати гібридної моделі написання документації становлять лише 43,1% від початкового бюджету проекту.

З десяти створених документів на протязі двох спринтів, лише сім документів були прийняті в роботу без повернення на доопрацювання через технічну складність модуля та високу критичність помилки. Згідно розрахунків за формулою (2.2), значення FPAR 70%.

Середня кількість ітерацій на документ розраховується за формулою (2.3), де $k=1$ для семи документів, що були прийняті з першої ітерації, та $k=2$ для трьох документів, що потребували доопрацювання. За розрахунками значення AIC вийшло 1.3 ітерації / на документ.

Останнім показником для розрахунку був TSR (2.4). Даний коефіцієнт часу розраховувався шляхом відношення часових витрат спринту 2 до загальних часових витрат на створення документації. 53,8% становить економія часу, якщо використовувати сценарій В в подальших спринтах при написанні документації.

Апробація розробленого методу управління роботами зі створення допоміжної документації на прикладі ІТ-проєкту розробки платформи генерації цифрового контенту з командою із семи учасників(чотирьох розробників, двох тестувальників та менеджера проєкту) проводилась протягом двох спринтів: у першому документування велось за традиційною моделлю без використання ШІ, у другому із застосуванням розробленої технології. Отримані результати підтвердили ефективність розробленого методу за всіма визначеними показниками.

ВИСНОВКИ

У ході кваліфікаційної роботи проведено дослідження з метою ознайомлення зі: специфікою процесів створення допоміжної документації в ІТ-проектах; існуючими моделями та підходами до організації документування; сучасними методами автоматизації із застосуванням ШІ.

У ході виконання завдань кваліфікаційної роботи було проведено аналіз особливостей організації процесу створення допоміжної документації в ІТ-проектах розробки платформ генерації цифрового контенту.

У ході виконання аналізу предметної області виявлено, що існуючі моделі управління процесу документації мають суттєві обмеження через надмірне навантаження на ресурси проекту.

Розроблений метод управління роботами зі створення допоміжної документації забезпечує комплексний підхід до організації процесу документування, оскільки дозволяє сформувати як структурну модель процесу із визначенням виконавців, режимів роботи ШІ та критичності кожної задачі, так і модель оптимізації витрат, що відображає економічну ефективність гібридного підходу.

Даний метод розроблений для оптимізації процесу створення документації в активній фазі розробки ІТ-проекту, але при модифікації етапу збору та аналізу вихідних даних його можна використовувати для проактивного планування документування ще до початку реалізації нового модуля платформи.

Графічний матеріал кваліфікаційної роботи надається в додатку А.

За тематикою кваліфікаційної роботи опубліковано тези доповіді на тему «Оптимізація процесу створення технічної документації в іт-проектах із застосуванням ШІ» на 30-ому Міжнародному молодіжному форумі «Радіoeлектроніка і молодь у ХХІ столітті» [18].

Кваліфікаційна робота виконувалась згідно з вимогами методичних вказівок з організації та виконання кваліфікаційної роботи [19] та національним стандартом ДСТУ 3008:2015 «Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлення» [20].

Перелік джерел посилання оформлено згідно з національним стандартом ДСТУ 8302:2015 «Бібліографічне посилання. Загальні положення та правила складання» [21].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kambala Gireesh. Natural Language Processing for IT Documentation and Knowledge Management // *International Journal of Scientific Research and Management (IJSRM)*. – 2021. №2(11) – ISSN 2321-3418.
2. Gentle Anne. Docs Like Code. URL: https://books.google.pl/books/about/Docs_Like_Code.html/ (дата звернення: 18.03.2026).
3. Chinchilla Chris. Technical Writing for Software Developers. Enhance communication, improve collaboration, and leverage AI tools for software development. – ISBN 978-18-350-8391-8
4. Confluence Cloud resources URL: <https://support.atlassian.com/confluence-cloud/resources/> (дата звернення: 18.03.2026).
5. Notion. A story of tools and the future of work URL: <https://www.notion.com/about> (дата звернення: 18.03.2026).
6. Documentation in Agile methodology. Agile project management blog. URL: <https://www.mindspire-consulting.com/blog/project-management-blog/documentation-in-agile-methodology/> (дата звернення: 18.03.2026)
7. Taylor Ren. How AI for Technical Documentation Boosts Productivity URL: <https://www.heretto.com/blog/ai-documentation-for-improving-technical-content#what-is-ai-for-technical-documentation/> – Дата публікації: 17.03.2025. (дата звернення: 21.03.2026).
8. Kambala Gireesh. Natural Language Processing for IT Documentation and Knowledge Management // *International Journal of Scientific Research and Management (IJSRM)*. – 2021. №2(11) – ISSN 2321-3418.
9. ISO/IEC/IEEE 26514:2016. Systems and software engineering — Design and development of software user documentation. – Geneva: ISO, 2022. – 84 с

10. Сьоме видання Настанови до зводу знань з управління проєктами (Настанова РМВОК). Вид. офіц. Ньютаун Сквер, Пенсильванія, 2021. 368 с.
11. What is an API (application programming interface). IBM. URL:<https://www.ibm.com/think/topics/api> (дата звернення 04.04.2026).
12. What is human-in-the-loop?. IBM. URL:<https://www.ibm.com/think/topics/human-in-the-loop> (дата звернення 12.04.2026).
13. Статистика зарплат Dou для .Net middle URL:<https://jobs.dou.ua/salaries/?period=2025-12&group=1&position=Software%20Engineer> (дата звернення 17.04.2024)
14. LLM as a Service: The Future of AI-Powered Solution. Apoorv Gehlot URL:<https://medium.com/@apoorv-gehlot/llm-as-a-service-the-future-of-ai-powered-solutions-e6b4c31ba0c9> (дата звернення 17.04.2024)
15. Benefits of working with code repositories. Develtio. URL:<https://develtio.com/blog/knowledge/benefits-of-working-with-code-repositories/> . (дата звернення 18.04.2026)
16. JiraUser Guide URL:<https://www.atlassian.com/software/jira/guides/getting-started/introduction#what-is-jira-software> (дата звернення 19.04.2026)
17. Asana User Guide URL:https://help.asana.com/s/article/all-asana-features?language=en_US (дата звернення 19.04.2026)
18. Д'яченко М., Євланов М. Оптимізація процесу створення технічної документації в іт-проєктах із застосуванням штучного інтелекту // 30-й Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті». Зб. матеріалів форуму. Т. 6. Харків: ХНУРЕ. 2026. С. 155–157
19. Методичні вказівки щодо розробки та оформлення кваліфікаційної роботи другого (магістерського) рівня вищої освіти спеціальності 122 Комп'ютерні науки за освітньо-науковою програмою «Управління проєктами в галузі інформаційних технологій» / уклад. К. Е. Петров та ін. Харків : ХНУРЕ, 2025. 24 с

20. ДСТУ 3008-2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлювання. Чинний від 22.06.2015. Київ: ДП «УкрНДНЦ», 2016. 31 с.

21. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ : УкрНДНЦ, 2016. 16 с.