МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА

Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»

Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Інформаційні управляючі системи і технології»

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

Форма навчання: заочна

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему: «Цифрова підтримка акселерації інноваційних проєктів у бізнесі»

здобувача Савінкова Нікіти Дмитровича
(ПІБ, підпис)

Науковий керівник: _к.е.н., доцент Денісова О.О._____

Робота допущена до захисту перед екзаменаційною комісією з атестації здобувачів вищої освіти (ЕК)

Завідувач кафедри: к.е.н., доцент Тішков Б.О. _____
(підпис)

Київ 2025

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Інформаційні управляючі системи і технології»

галузь знань 12 Інформаційні технології
спеціальність 122 Комп'ютерні науки

ПОГОДЖЕНО:
Керівник проєктної групи (гарант)
освітньо-професійної програми

_____Устенко С.В.
«____» _____202_ р.

ЗАТВЕРДЖУЮ:
Завідувач кафедри інформаційних
систем в економіці

_____ Тішков Б.О.
«____» _____ 202_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти_____
(прізвище, ім'я, по батькові)

_____ форми навчання
очної (денної), заочної, дистанційної

на підготовку кваліфікаційної магістерської роботи
на тему: *«Цифрова підтримка акселерації інноваційних проєктів у бізнесі»*

Тему затверджено наказом ректора Університету від «__» _____ 202_ р. №_____

Кваліфікаційна магістерська робота виконується на матеріалах
_____

План кваліфікаційної магістерської роботи

Розділ I Характеристика та аналіз предметної галузі

Розділ II Розробка вимог і моделювання інформаційної системи

Розділ III Проектування та реалізація компонентів системи

Об'єкт дослідження: Процеси організації та автоматизації управління інноваційною діяльністю в бізнес-середовищі з використанням цифрових інформаційних систем.
Предмет дослідження: Методи, моделі та інформаційні технології цифрової підтримки процесів подання, аналізу та реалізації інноваційних бізнес-ідей із використанням засобів штучного інтелекту та автоматизованих інформаційних систем.

Мета кваліфікаційної магістерської роботи: Розробка та створення прототипу інформаційної системи, яка забезпечує цифрову підтримку процесів збору, аналізу, класифікації та рекомендації інноваційних ідей у бізнесі із використанням технологій штучного інтелекту, автоматизації та сучасних інформаційних технологій.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:
У розділі I У першому розділі я дослідив теоретичні основи цифрової підтримки інновацій у бізнесі. Проаналізував існуючі платформи, наукову літературу та сучасні підходи до автоматизації процесів подання й аналізу ідей. Визначив основні функції майбутньої системи та обґрунтував актуальність її створення.

У розділі II У другому розділі було спроєктовано прототип інформаційної системи, що підтримує цифрову акселерацію інноваційних проєктів. Я розробив структуру бази даних, алгоритми роботи системи та створив діаграми, які відображають логіку функціонування. Реалізовано взаємодію між модулями: користувацьким інтерфейсом, модулем обробки ідей і підсистемою рекомендацій. Усі компоненти перевірено на тестових прикладах, що підтвердило працездатність запропонованого рішення.

У розділі III У третьому розділі було створено прототип інформаційної системи з реалізацією її основних функціональних модулів. Було налаштовано технічне середовище, обрано мову програмування Python та необхідні бібліотеки для роботи з базою даних і модулем штучного інтелекту. Реалізовано логіку обробки ідей, формування рекомендацій, збереження даних та взаємодії з користувачем через інтерфейс. Проведено тестування працездатності системи на контрольних прикладах. Також визначено напрями для подальшого вдосконалення функціоналу та розширення можливостей системи.

Завдання підготував
науковий керівник _____ _____
(підпис) *(ініціали, прізвище)*

«____» _____ 202_ р.

Завдання одержав
здобувач _____ _____
(підпис) *(ініціали, прізвище)*

«____» _____ 202_ р.

Реферат

Кваліфікаційна магістерська робота містить 81 сторінок, 8 таблиць, 16 рисунків, список використаних джерел з 20 найменувань, додатки.

«Цифрова підтримка акселерації інноваційних проєктів у бізнесі»

Ключові слова: інновації, веб-платформа, штучний інтелект, класифікація ідей, цифрові рішення, працівник, бізнес-процеси, інформаційна система, машинне навчання.

Об'єкт дослідження: процес цифрової підтримки інноваційної діяльності в бізнес-середовищі.

Предмет дослідження: веб-платформа для збору, класифікації та обробки інноваційних ідей працівників із використанням технологій штучного інтелекту.

Мета магістерського кваліфікаційної магістерської роботи полягає у розробці веб-платформи, яка дозволяє працівникам компанії пропонувати ідеї для покращення бізнесу, автоматично класифікує їх за допомогою ШІ, здійснює підбір релевантних курсів або цифрових рішень, оцінює їхню ефективність та вартість, і в результаті пропонує оптимальні шляхи реалізації запропонованих ідей.

Апаратура, використана при дослідженні:

- процесор – IntelCore i9-9400f 2.50ГГц.;
- оперативна пам'ять – 16 Гб;
- відеокарта – AsusArmor 1070, 8 Гб;
- жорсткий диск – HDD, 1ТБ;

Отримані результати: спроєктована платформа для акселерації бізнесу завдяки ідеям співробітників.

Область застосування: будь-який бізнес чи стартап .

Рік виконання випускного магістерського проекту: 2025.

Рік захисту випускного магістерського проекту: 2025.

ЗМІСТ[1]

ВСТУП

У сучасному динамічному бізнес-середовищі здатність адаптуватися та впроваджувати інновації перестала бути конкурентною перевагою — вона стала критичною умовою виживання. Незважаючи на стрімкий розвиток технологій, саме людський капітал залишається джерелом ідей, які можуть трансформувати внутрішні процеси підприємства або навіть вивести його на новий рівень. Проте ці ідеї часто залишаються непочутими або не мають достатнього інструментарію для практичного втілення. Це створює реальний виклик: як зібрати, впорядкувати й перетворити ініціативи співробітників у реалізовані рішення?

Відповіддю на це запитання може стати інтерактивна цифрова платформа, яка виступає посередником між ініціативою людини та конкретними кроками до її реалізації. Використання сучасних підходів до обробки природної мови, машинного навчання та рекомендаційних систем дозволяє автоматизувати ключові етапи цього процесу: від збору пропозицій — до підбору релевантних інструментів чи освітніх ресурсів для їх впровадження.

Обрана тема дипломної роботи зосереджена на створенні такої веб-платформи — ресурсу, який приймає ідеї від співробітників, групує їх за тематичними напрямами з допомогою алгоритмів штучного інтелекту, проводить аналіз доступних цифрових рішень і навчальних матеріалів, оцінює їх ефективність, вартість, і надає користувачу декілька найбільш обґрунтованих варіантів. Такий підхід не лише підвищує ефективність процесу генерації інновацій, але й мінімізує суб’єктивність та затримки, пов’язані з ручною обробкою ідей.

Актуальність обраної теми пояснюється кількома ключовими чинниками. По-перше, спостерігається тенденція до децентралізації процесів прийняття рішень у компаніях, що створює потребу в інструментах, які дозволяють "чути" голос кожного працівника, незалежно від його посади чи підрозділу. По-друге, ринок цифрових рішень є надзвичайно фрагментованим, що ускладнює ручний пошук релевантних інструментів для реалізації певної ідеї. І по-третє, бізнесу потрібні не

лише креативні рішення, але й шляхи їх швидкого втілення — з урахуванням витрат, ризиків і очікуваного результату.

У центрі цієї роботи — не просто побудова чергової платформи для збору зворотного зв’язку. Йдеться про створення смарт-системи, яка функціонує як каталізатор внутрішньої інноваційності: вона виявляє ідеї, бачить їхній потенціал, і надає обґрунтовані варіанти реалізації. Це вже не лише цифрова підтримка, а інтелектуальна інфраструктура для акселерації змін.

Сучасний рівень розвитку штучного інтелекту, зокрема в галузі класифікації текстових даних, семантичного аналізу та рекомендаційних моделей, дає змогу реалізувати систему, яка розуміє зміст людських пропозицій і зіставляє їх з релевантними ресурсами. Такий функціонал відкриває нові можливості для компаній — від підвищення інноваційного потенціалу до зменшення витрат на аналітику та дослідження.

Вибір теми зумовлений практичною значущістю проєкту. Розробка подібної системи дозволяє не лише вдосконалити внутрішні бізнес-процеси, а й створити основу для довгострокового сталого розвитку компаній за рахунок підвищення залученості працівників і системної роботи з їх ідеями. Дипломна робота передбачає створення технічно обґрунтованої архітектури платформи, формування її функціональних модулів, розробку алгоритмів обробки пропозицій, а також оцінку можливостей її впровадження у реальних умовах бізнесу.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз існуючих інформаційних управляючих систем (предметної області)

1.1.1 Дослідження предметної області

У сучасному бізнес-середовищі інноваційна активність набуває ключового значення для забезпечення сталого розвитку, підвищення ефективності внутрішніх процесів та формування конкурентних переваг. Проте традиційні підходи до управління інноваціями виявляються недостатньо гнучкими та повільними в умовах динамічної цифрової трансформації. Саме тому актуальним є використання інформаційних систем, які забезпечують цифрову підтримку акселерації інноваційних проєктів у бізнесі.

Предметна область цього дослідження охоплює процеси подачі, структурованого аналізу, класифікації та супроводу інноваційних ідей, запропонованих працівниками підприємства, за допомогою цифрових платформ і технологій штучного інтелекту. Особливу увагу приділено створенню умов, за яких ці ідеї можуть не лише зберігатися, але й автоматично оброблятися, аналізуватися за змістом, розподілятися за категоріями та отримувати персоналізовані рекомендації щодо їх реалізації.

Акселерація в цьому контексті розуміється як прискорення життєвого циклу ідеї — від її формування до реалізації. В умовах сучасного бізнесу це передбачає не лише внутрішнє адміністрування, але й залучення зовнішніх цифрових рішень, таких як онлайн-курси, інструменти автоматизації, програмні продукти або сервіси, що допомагають реалізувати певну ініціативу.

Цифрова підтримка полягає у використанні веб платформи, на якій кожен співробітник може подати інноваційну ідею через спеціальну форму. Далі ідея потрапляє в систему, де за допомогою модулів штучного інтелекту (зокрема NLP

— Natural Language Processing) вона автоматично обробляється: аналізується її зміст, ключові слова, визначається напрям (наприклад: маркетинг, логістика, IT, клієнтський сервіс тощо). Після цього система звертається до внутрішніх і зовнішніх баз даних — наприклад, каталогів освітніх курсів, онлайн-сервісів або спеціалізованого ПЗ — для пошуку рішень, які можуть сприяти реалізації ідеї.

У межах предметної області вивчається також взаємодія між трьома основними сторонами:

користувачем системи (автором ідеї або адміністратором),

інформаційною системою, яка виконує аналіз, зберігання, рекомендації,

цифровим зовнішнім середовищем, з яким система взаємодіє через API або готові джерела даних.

Технічно предметна область охоплює інформаційні системи підтримки прийняття рішень, системи штучного інтелекту (нейромережі, моделі машинного навчання), вебінтерфейси, бази даних (PostgreSQL), API-інтеграцію та сервіси візуалізації результатів. Архітектура платформи базується на взаємодії між базою даних, серверною логікою на Python та клієнтською частиною.

У межах дослідження було проаналізовано схожі системи, зокрема IdeaScale, Monday.com, ProductBoard, які використовуються в корпоративному середовищі для управління ідеями. Усі вони мають обмеження щодо автоматизації, що підтверджує доцільність створення власного рішення з акцентом на AI-підтримку.

Таким чином, предметна область цієї роботи охоплює комплекс технічних, організаційних та управлінських процесів, які пов’язані з впровадженням цифрової платформи для прискорення інноваційної діяльності в бізнесі. Система, що була спроєктована в межах дипломної роботи, покликана усунути бар’єри на шляху впровадження ініціатив і підвищити загальну ефективність інноваційного розвитку підприємств.

Було досліджено існуючі компоненти акселерації в бізнесі, що будуть представлені далі у тексті. А також були проаналізовані схожі за задумкою та реалізацією вже існуючі системи.

Цифрова інфраструктура акселерації інноваційних проєктів у бізнесі є важливим елементом, що забезпечує необхідні умови для ефективного розвитку стартапів та інноваційних ініціатив. Вона охоплює сукупність технологічних, програмних, організаційних та інформаційних ресурсів, які підтримують стартапи на всіх етапах їхнього розвитку — від ідеї до комерціалізації. Завдяки такій інфраструктурі стартапи можуть швидко адаптуватися до змінюваного ринку, отримувати необхідні ресурси для розвитку і забезпечувати ефективне управління проектами.

Одним із основних компонентів цифрової інфраструктури є технології, зокрема хмарні сервіси, які надають стартапам гнучкість у використанні обчислювальних потужностей та зберіганні даних без необхідності в значних витратах на фізичні сервери. Хмарні платформи, такі як AWS, Google Cloud та Microsoft Azure, дозволяють масштабувати інфраструктуру залежно від потреб стартапу. Також важливою складовою є платформи для автоматизації бізнес-процесів, зокрема CRM-системи, які допомагають стартапам управляти взаємодією з клієнтами, партнерами та інвесторами, а також інструменти для проектного менеджменту, що дозволяють ефективно організовувати робочі процеси, розподіляти завдання і відслідковувати прогрес.

Штучний інтелект та аналіз великих даних також займають важливу роль у цифровій інфраструктурі акселерації. Завдяки великим даним стартапи можуть отримувати важливу інформацію для прийняття рішень щодо ринку, конкурентів та споживачів. Алгоритми машинного навчання допомагають аналізувати велику кількість інформації, що дозволяє стартапам знаходити найкращі стратегії для розвитку і оптимізувати їхні процеси. Крім того, штучний інтелект дозволяє автоматизувати рутинні задачі, такі як обробка даних про клієнтів, прогнозування попиту чи визначення потенційних ризиків, що дає змогу зосередитись на інноваціях і розвитку продукту.

Інформаційні ресурси є важливим компонентом для акселерації стартапів. Вони включають платформи для збору ідей, на яких інноватори можуть подати свої

проєкти, а акселератори оцінюють їхні перспективи. Такі платформи використовують механізми голосування, рейтингу та зворотного зв'язку для того, щоб вибрати найкращі проєкти для подальшої акселерації. Візуалізація даних є ще одним важливим елементом, який дозволяє стартапам ефективно відслідковувати ключові показники успішності проєкту. Використання спеціалізованих платформ, таких як Power BI чи Tableau, дозволяє створювати дашборди для моніторингу фінансових та операційних даних, що є важливим для прийняття стратегічних рішень.

Окрім технологічних та інформаційних компонентів, організаційні аспекти цифрової інфраструктури також мають велике значення для акселерації. Для стартапів важливо мати доступ до комунікаційних інструментів, таких як Slack, Microsoft Teams, Zoom, які дозволяють підтримувати постійний зв'язок між учасниками акселераційної програми, менторами, інвесторами та іншими стейкхолдерами. Такі платформи забезпечують ефективну організацію онлайн-зустрічей, тренінгів та презентацій, що є важливим для навчання та розвитку стартапів. Інструменти для зворотного зв'язку, такі як Surveys чи Google Forms, дозволяють збирати відгуки від учасників програми, що допомагає постійно вдосконалювати процес акселерації.

Не менш важливим є доступ до фінансування. Цифрові платформи для краудфандингу та онлайн-інвестування, такі як Kickstarter, Indiegogo, Seedrs та AngelList, дозволяють стартапам залучати капітал від широкої аудиторії інвесторів. Це дає змогу стартапам отримати необхідні ресурси для розвитку без необхідності звертатися до традиційних джерел фінансування. Крім того, електронні платформи для створення юридичних документів та електронного підпису, такі як LegalZoom і DocuSign, допомагають стартапам автоматизувати процеси юридичних угод та договорів, що забезпечує зручність і безпеку в процесі співпраці з партнерами та інвесторами.

Інформаційні системи управління інноваційними проєктами є важливою складовою цифрової підтримки акселерації інновацій в бізнесі, оскільки вони

дозволяють ефективно організовувати, відслідковувати та контролювати всі етапи реалізації проєкту — від початкової ідеї до комерціалізації результатів. Такі системи забезпечують інтеграцію різних бізнес-процесів, полегшують управлінське рішення, покращують комунікацію між командами та іншими зацікавленими сторонами, а також сприяють швидкому реагуванню на зміни зовнішнього середовища, що є критичним для інноваційних проєктів.

Основною метою інформаційних систем управління інноваційними проєктами є оптимізація управлінських процесів, автоматизація рутинних завдань та забезпечення високого рівня прозорості для всіх учасників проєкту. Це досягається завдяки впровадженню інструментів для планування, моніторингу та аналізу проєктів, а також інтеграції з іншими корпоративними системами (наприклад, системами управління ресурсами підприємства, CRM, ERP).

Одним із основних компонентів таких систем є модулі для планування та організації проєктів. Це включає функціонал для розподілу завдань між учасниками команди, встановлення термінів виконання, визначення відповідальних осіб, а також створення графіків виконання робіт. За допомогою таких модулів проектні менеджери можуть візуалізувати хід виконання завдань, відслідковувати прогрес і за необхідності коригувати плани. Інструменти для створення Gantt-діаграм, Kanban-дошок або інтерактивних дашбордів забезпечують ефективне управління часом, ресурсами та етапами виконання проєкту.

Ще одним важливим аспектом є використання інформаційних систем для моніторингу та контролю результатів. Для цього в таких системах передбачено можливість відстеження ключових показників ефективності (KPI), що дає змогу оцінювати не лише процеси, але й досягнуті результати. Завдяки інтеграції з аналітичними інструментами, інформаційні системи можуть автоматично збирати дані з різних джерел (наприклад, фінансові показники, дані про продажі, показники задоволеності клієнтів), аналізувати їх та генерувати звіти, які допомагають приймати обґрунтовані рішення.

Інформаційні системи для управління інноваційними проєктами також включають можливості для комунікації та взаємодії учасників проєкту. Інструменти для чатів, відеоконференцій та спільних робочих просторів сприяють безперервному обміну інформацією між командами, підрядниками, партнерами та іншими зацікавленими сторонами. Завдяки цьому всі учасники проєкту можуть бути в курсі актуальних змін, отримувати оновлення в реальному часі та швидко реагувати на зміни або проблеми.

Особливо важливим є використання таких систем у контексті управління інноваціями, оскільки інноваційні проєкти часто передбачають високу невизначеність і швидкі зміни умов. В таких випадках системи управління можуть допомогти мінімізувати ризики, надаючи можливості для швидкого аналізу даних, що дозволяє вчасно коригувати стратегії і адаптуватися до нових умов. Для цього використовуються різноманітні технології, включаючи штучний інтелект та машинне навчання, які можуть допомогти в прогнозуванні трендів та автоматизації прийняття рішень на основі зібраних даних.

Інтеграція цифрових платформ для управління інноваційними проєктами з іншими корпоративними інформаційними системами, такими як системи управління фінансами (ERP), управління ресурсами (HRM) та управління взаємодією з клієнтами (CRM), дозволяє створити єдину екосистему для всіх операційних аспектів бізнесу. Це дозволяє керівникам отримувати цілісну картину стану проєкту, відстежувати використання ресурсів, оптимізувати витрати і вчасно коригувати стратегію в разі необхідності.

Важливим аспектом є також забезпечення безпеки даних та захисту інтелектуальної власності. Інформаційні системи управління інноваційними проєктами мають вбудовані механізми для захисту конфіденційної інформації, таких як шифрування даних, контроль доступу та механізми аутентифікації користувачів, що є критичними для стартапів, які часто працюють з новітніми технологіями та ідеями.

Крім того, сучасні інформаційні системи управління інноваційними проєктами забезпечують високий рівень гнучкості, що дозволяє стартапам швидко адаптувати систему під специфічні потреби і масштаби проєкту. Завдяки цьому стартапи можуть інтегрувати необхідні інструменти, залежно від етапу розвитку проєкту та специфіки індустрії, в якій вони працюють.

Хмарні технології та API-інтеграції відіграють ключову роль у цифровій підтримці акселерації інноваційних проєктів у бізнесі, оскільки вони забезпечують гнучкість, масштабованість та можливість швидкого впровадження нових функціональностей, що є критичним для стартапів та інноваційних компаній. Використання хмарних платформ та інтеграційних інтерфейсів API дозволяє підприємствам значно спростити процеси зберігання, обробки та обміну даними, забезпечуючи зростання ефективності та прискорення інноваційних процесів.

Хмарні технології є основою для надання підприємствам доступу до необмежених обчислювальних ресурсів, таких як сервери, зберігання даних та мережеві послуги, без необхідності інвестувати в дорогі фізичні інфраструктури. Це дозволяє компаніям значно скоротити витрати на IT-інфраструктуру та спрямувати ресурси на розробку інноваційних продуктів і рішень. Хмарні платформи пропонують широкий спектр сервісів, що забезпечують можливості для масштабування, резервного копіювання, безпеки даних, а також обробки великих обсягів інформації. Вони дозволяють стартапам і підприємствам з різними вимогами гнучко адаптувати свої ресурси під потреби бізнесу та специфіку проєкту.

Завдяки хмарним технологіям стартапи можуть реалізувати рішення на основі передових інструментів для машинного навчання, аналізу даних, розподілених обчислень та великих даних (Big Data). Наприклад, хмарні сервіси дозволяють здійснювати обробку даних у реальному часі, що важливо для інноваційних проєктів, які мають потребу в швидкому реагуванні на зміни ринку або поведінки користувачів. Доступ до хмарних інструментів дозволяє підприємствам тестувати та впроваджувати нові технології без великих

капітальних витрат, що є суттєвим фактором для стартапів, що працюють в умовах високої невизначеності та швидких змін.

API-інтеграції, у свою чергу, є важливою складовою для забезпечення взаємодії між різними системами, застосунками та сервісами. Вони дозволяють об'єднувати різні технологічні рішення в єдину екосистему, що важливо для інтеграції різних функціональних модулів у рамках одного проєкту або бізнес-платформи. API (Application Programming Interface) є інтерфейсом, який дозволяє різним програмним системам обмінюватися даними та взаємодіяти між собою без потреби в прямій інтеграції на рівні коду. Це дає змогу компаніям швидко адаптувати нові функціональності та інтегрувати зовнішні сервіси, такі як платіжні системи, аналітичні інструменти, CRM-системи, платформи для обміну даними тощо.

Інтеграція через API забезпечує більшу гнучкість, дозволяючи компаніям обирати найкращі інструменти для вирішення конкретних завдань. Наприклад, стартапи можуть інтегрувати API для доступу до хмарних баз даних, соціальних мереж, аналітики або навіть для забезпечення платіжних функцій, що дозволяє їм створювати інноваційні рішення без необхідності розробляти все з нуля. Крім того, API дозволяють бізнесам швидко адаптуватися до змін, впроваджуючи нові технології чи сервіси в межах існуючої інфраструктури, що підвищує ефективність і зменшує час виходу на ринок.

Використання хмарних технологій разом з API-інтеграціями дає змогу знизити складність управління інфраструктурою, оскільки більшість процесів з управління ресурсами, безпекою та масштабуванням можна автоматизувати. Це дозволяє зосередитись на розвитку самих продуктів і послуг, а не на технічних питаннях інфраструктури. Крім того, такі технології сприяють інтеграції даних із різних джерел, що є необхідним для прийняття обґрунтованих рішень в умовах високої конкуренції та змінності ринку.

Крім того, хмарні технології та API-інтеграції сприяють розвитку інновацій у бізнесі завдяки їхній здатності забезпечувати високий рівень автоматизації та персоналізації. Вони дозволяють впроваджувати інноваційні рішення, що базуються на великих даних і штучному інтелекті, для покращення продуктів і послуг. Наприклад, інтеграція хмарних платформ для машинного навчання з API-сервісами може допомогти в автоматизації процесів рекомендацій для користувачів або в розробці персоналізованих пропозицій на основі аналізу поведінки споживачів.

Amazon Web Services (AWS) є однією з провідних світових хмарних платформ, яка широко використовується для підтримки цифрової акселерації інноваційних проєктів у бізнесі. Її перевага полягає в тому, що вона надає масштабовану, надійну й безпечну інфраструктуру, яка дозволяє компаніям будь-якого розміру запускати свої рішення без значних початкових інвестицій у фізичні сервери та IT-інфраструктуру.

Один із ключових сервісів — Amazon EC2 — дозволяє створювати віртуальні сервери з гнучкими конфігураціями, що ідеально підходить для швидкого запуску MVP (мінімально життєздатного продукту) чи прототипів інноваційних рішень. Для зберігання даних використовується Amazon S3, який забезпечує доступність, масштабованість і високу надійність — критичні параметри для сучасних цифрових проєктів.

Для обробки великих обсягів даних, автоматизації процесів і впровадження штучного інтелекту AWS пропонує Amazon SageMaker, що дозволяє створювати, навчати та впроваджувати моделі машинного навчання без глибоких знань у програмуванні. Це значно знижує поріг входу для стартапів або команд, які лише розпочинають інноваційний проєкт.

Іншою важливою складовою є безсерверні обчислення через AWS Lambda, які дозволяють запускати окремі функції коду у відповідь на події, що ідеально підходить для побудови швидких, економних та гнучких систем. Це особливо

корисно в акселераційних програмах, де важливо швидко тестувати гіпотези без перевитрат ресурсів.

Крім того, AWS має потужні інструменти для API-інтеграції — зокрема Amazon API Gateway, який дозволяє створювати, розгортати та захищати API, завдяки чому бізнес може легко підключати зовнішні сервіси або інтегруватися з іншими системами.

Для підтримки безперервної розробки та розгортання інновацій AWS надає інструменти DevOps, такі як CodeBuild, CodePipeline і CloudFormation, що дозволяють автоматизувати CI/CD-процеси (безперервну інтеграцію та доставку). Таким чином, цифрові команди можуть швидко оновлювати продукт і впроваджувати нові функції в реальному часі.

Ще однією перевагою є глобальна інфраструктура AWS із десятками дата-центрів по всьому світу, що дозволяє запускати проєкти з мінімальною затримкою незалежно від регіону. AWS також має вбудовані засоби кібербезпеки, шифрування та управління доступом, які забезпечують відповідність стандартам GDPR, ISO, SOC та іншим.

У підсумку, AWS є не лише платформою для розміщення сервісів, а й комплексним інструментом для цифрової трансформації, який дає змогу інноваційним проєктам пройти шлях від ідеї до масштабованого рішення максимально швидко, безпечно й ефективно.

Google Cloud Platform

Amazon Web Services (AWS) є однією з провідних світових хмарних платформ, яка широко використовується для підтримки цифрової акселерації інноваційних проєктів у бізнесі. Її перевага полягає в тому, що вона надає масштабовану, надійну й безпечну інфраструктуру, яка дозволяє компаніям будь-якого розміру запускати свої рішення без значних початкових інвестицій у фізичні сервери та IT-інфраструктуру.

Один із ключових сервісів — Amazon EC2 — дозволяє створювати віртуальні сервери з гнучкими конфігураціями, що ідеально підходить для швидкого запуску MVP (мінімально життєздатного продукту) чи прототипів інноваційних рішень. Для зберігання даних використовується Amazon S3, який забезпечує доступність, масштабованість і високу надійність — критичні параметри для сучасних цифрових проєктів.

Для обробки великих обсягів даних, автоматизації процесів і впровадження штучного інтелекту AWS пропонує Amazon SageMaker, що дозволяє створювати, навчати та впроваджувати моделі машинного навчання без глибоких знань у програмуванні. Це значно знижує поріг входу для стартапів або команд, які лише розпочинають інноваційний проєкт.

Іншою важливою складовою є безсерверні обчислення через AWS Lambda, які дозволяють запускати окремі функції коду у відповідь на події, що ідеально

підходить для побудови швидких, економних та гнучких систем. Це особливо корисно в акселераційних програмах, де важливо швидко тестувати гіпотези без перевитрат ресурсів.

Крім того, AWS має потужні інструменти для API-інтеграції — зокрема Amazon API Gateway, який дозволяє створювати, розгортати та захищати API, завдяки чому бізнес може легко підключати зовнішні сервіси або інтегруватися з іншими системами.

Для підтримки безперервної розробки та розгортання інновацій AWS надає інструменти DevOps, такі як CodeBuild, CodePipeline і CloudFormation, що дозволяють автоматизувати CI/CD-процеси (безперервну інтеграцію та доставку). Таким чином, цифрові команди можуть швидко оновлювати продукт і впроваджувати нові функції в реальному часі.

Ще однією перевагою є глобальна інфраструктура AWS із десятками дата-центрів по всьому світу, що дозволяє запускати проєкти з мінімальною затримкою незалежно від регіону. AWS також має вбудовані засоби кібербезпеки, шифрування та управління доступом, які забезпечують відповідність стандартам GDPR, ISO, SOC та іншим.

У підсумку, AWS є не лише платформою для розміщення сервісів, а й комплексним інструментом для цифрової трансформації, який дає змогу інноваційним проєктам пройти шлях від ідеї до масштабованого рішення максимально швидко, безпечно й ефективно.

1.1.2 Дослідження ринку інформаційних управляючих систем

У процесі підготовки до розробки інформаційної системи було проведено аналіз сучасного ринку програмних рішень, які використовуються в бізнесі для управління інноваціями, збору ідей від працівників, оптимізації внутрішніх процесів і підтримки командної взаємодії. Метою дослідження було визначення функціональних можливостей існуючих платформ, їхніх обмежень, цільових аудиторій та технічних особливостей.

Особливу увагу було приділено сервісам, що забезпечують збір ідей як із внутрішніх джерел компанії, так і від зовнішніх користувачів. Серед найпоширеніших рішень виділяються такі платформи, як IdeaScale, Brightidea, Jira Product Discovery, Monday.com, Miro та ClickUp. Ці системи дозволяють створювати єдиний інформаційний простір для подання ідей, коментування, голосування, обговорення та моніторингу реалізації.

Однак, у ході аналізу виявлено, що більшість існуючих платформ орієнтовані переважно на проєктний менеджмент або координацію командної роботи й практично не містять інструментів інтелектуальної обробки ідей. Функціонал автоматичного визначення тематики, типу проблеми чи генерації релевантних рекомендацій (наприклад, навчальних курсів або цифрових сервісів) або відсутній повністю, або реалізований у вигляді простих інструментів ручного тегування.

Наприклад, система Monday.com забезпечує гнучкість конфігурацій та підтримує численні інтеграції, проте не передбачає автоматичного аналізу змісту поданих ідей. IdeaScale вирізняється розширеними можливостями збирання ідей і проведення голосувань, однак етап обробки інформації здебільшого потребує ручного втручання модераторів. Платформи ClickUp і Asana ефективно організовують роботу над завданнями, однак не містять модуля для контекстного аналізу тексту або генерації рекомендацій.

Також було розглянуто декілька рішень із відкритим вихідним кодом, зокрема Fider та Open Innovation Platform. Встановлено, що більшість із них потребують суттєвого доопрацювання та не мають вбудованих інструментів роботи зі штучним інтелектом, зокрема методами обробки природної мови (NLP).

Окремо було проаналізовано суміжні сервіси, які можуть частково реалізовувати окремі компоненти майбутньої системи. Серед них — платформи для пошуку навчальних курсів (Coursera, Udemy, Prometheus), сервіси для інтеграції (Zapier, Integromat), а також аналітичні рішення (Power BI, Tableau). Виявлено, що сучасні інструменти успішно вирішують вузькоспеціалізовані задачі, однак відсутні комплексні рішення, які б об’єднували всі функції — від збору ідей

до їх аналізу, генерації рішень та формування рекомендацій — у межах єдиної інформаційної системи.

Результати аналізу підтверджують наявність потреби у створенні спеціалізованої системи, орієнтованої на підтримку інноваційної діяльності в бізнес-середовищі. Ключовими особливостями такої системи мають стати інтелектуальна обробка текстової інформації, застосування технологій штучного інтелекту (насамперед NLP), а також здатність генерувати обґрунтовані рекомендації щодо цифрових рішень для реалізації поданих ідей.

Крім того, було встановлено, що значна частина популярних платформ є платними або має закритий код, що обмежує можливість їх впровадження в українських малих та середніх підприємствах. Це підсилює актуальність розробки адаптованої системи, яка могла б функціонувати як у хмарному середовищі, так і на внутрішніх серверах компаній, відповідно до потреб локального ринку.

Таким чином, проведене дослідження ринку інформаційних управляючих систем дозволило сформулювати функціональні вимоги до майбутньої системи, виявити недоліки наявних рішень і підтвердити доцільність створення власного інноваційного програмного забезпечення, спрямованого на цифрову підтримку бізнес-ініціатив нового покоління.

1.2 Обґрунтування вибору підходів і технологій для створення інформаційної управляючої системи

Розробка інформаційної системи цифрової підтримки акселерації інноваційних проєктів у бізнесі вимагає ретельного вибору підходів, інструментів і технологій, які не лише задовольнять функціональні вимоги, а й забезпечать масштабованість, гнучкість, швидкодію та простоту супроводу. На основі аналізу предметної області, потреб кінцевих користувачів та характеристик наявних аналогів було сформовано концепцію архітектури та визначено набір технологічних рішень, що стали основою розробки системи.

Архітектурний підхід: клієнт-серверна модель

Обрана архітектура системи — класична клієнт-серверна. Такий підхід дає змогу розділити логіку системи на незалежні компоненти:

клієнтську частину, відповідальну за взаємодію з користувачем через веб-інтерфейс (форма подання ідей, перегляд результатів, інтерфейс голосування тощо);

серверну частину, яка обробляє дані, класифікує ідеї, здійснює пошук рішень і взаємодіє з базами даних.

Клієнт-серверна архітектура дозволяє масштабувати систему, розміщувати її в хмарному середовищі (наприклад, AWS, Azure) або локально на сервері підприємства. Це особливо важливо для компаній із чутливою інформацією.

Мова програмування: Python

Усі основні серверні компоненти були реалізовані мовою Python, що обумовлено наступними чинниками:

Python є індустріальним стандартом у сферах data science, AI, обробки природної мови (NLP) і побудови прототипів інформаційних систем.

Python має чіткий і лаконічний синтаксис, що спрощує розробку, тестування й масштабування коду.

Широка екосистема бібліотек дозволяє реалізувати складні завдання (обробка тексту, аналіз даних, робота з API, побудова моделей) без створення власних інструментів із нуля.

Інструменти для NLP та AI

Основна цінність системи полягає в автоматизованій обробці ідей, поданих у вигляді тексту. Для цього було використано наступні інструменти:

spaCy — бібліотека для високошвидкісної обробки природної мови. Використовувалася для розпізнавання іменованих сутностей, розбору тексту, токенізації, виділення ключових слів і фраз.

Transformers (Hugging Face) — потужна бібліотека для роботи з передтренованими моделями глибокого навчання (наприклад, BERT, RoBERTa).

Моделі використовувалися для семантичного аналізу ідей, розуміння контексту та класифікації за напрямами бізнесу.

scikit-learn — застосовувався для побудови алгоритмів порівняння ідей із зовнішніми рішеннями, ранжування та побудови рекомендацій на основі векторного подібності або класифікаційних моделей.

Система керування базами даних: PostgreSQL

У якості СКБД було обрано PostgreSQL, яка дозволяє:

ефективно працювати з реляційною структурою даних;

реалізувати складні зв'язки між таблицями (наприклад, user – idea – decision – report);

зберігати великі обсяги текстової інформації, статистики, рейтингів і логів системи;

здійснювати трансакційний контроль та забезпечувати цілісність даних.

PostgreSQL також підтримує JSON, що дозволяє додавати гнучкі поля без порушення цілісності структури.

Користувацький інтерфейс і серверна логіка: Flask + Jinja2

Веб-сервер реалізовано на фреймворку Flask, що забезпечує:

просту та гнучку побудову API;

легке підключення модулів обробки даних, роботи з базами, шаблонізації;

швидкий старт без надлишкового коду.

Для відображення сторінок на боці клієнта було використано Jinja2 — механізм шаблонізації HTML-сторінок, який інтегрується з Flask.

Інтерфейс і візуалізація

Інтерфейс було спроєктовано у Figma (етап прототипування), реалізація виконувалась на HTML/CSS із адаптивним дизайном. У тестовій фазі також використовувався Qt Designer для побудови автономного десктопного прототипу.

Для виведення аналітичних результатів, рейтингу рішень і звітів було застосовано matplotlib / seaborn, що забезпечили візуалізацію даних у вигляді графіків, діаграм і таблиць.

Зовнішні API та інтеграції

Система передбачає можливість звернення до зовнішніх джерел даних (каталогів курсів, платформ, інструментів), зокрема через REST API. Для цього використано бібліотеки requests та httpx.

Наприклад, на основі ключових слів ідеї може бути надісланий запит до освітньої платформи (Coursera API), що дозволяє отримати список курсів з коротким описом, рейтингом, вартістю та мовою.

Контроль версій і командна робота: GitHub

Код зберігається в приватному репозиторії на GitHub, що дає змогу:

керувати версіями проєкту;

фіксувати зміни, проводити рев’ю;

формувати документацію та тестові гілки;

легко відновлювати попередні стани у разі помилок.

РОЗДІЛ 2 ХАРАКТЕРИСТИКА ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ ТА МЕТОДИ І МОДЕЛІ

2.1 Структура і характеристика системи

Бізнес-вимоги до системи. Бізнес-вимоги містять високорівневі цілі організації, які спрямовані на покращення роботи нейромережі за допомогою системи. Вони впливають на пріоритети реалізації варіантів використання і пов'язані з ними функціональні вимоги, також істотно впливають на спосіб реалізації вимог.

Також описують основні переваги, які нова система дасть організації і користувачам.

Бізнес вимоги:

- Підвищення ефективності інноваційної діяльності працівників;
- Зменшення витрат на пошук рішень;
- Прискорення реалізації інноваційних ідей.;

Бізнес-вимоги інформаційної системи нейромережею для розпізнавання облич у розумному будинкузображені на рис. 2.1 за допомогою менеджера специфікацій *EnterpriseArchitect*.

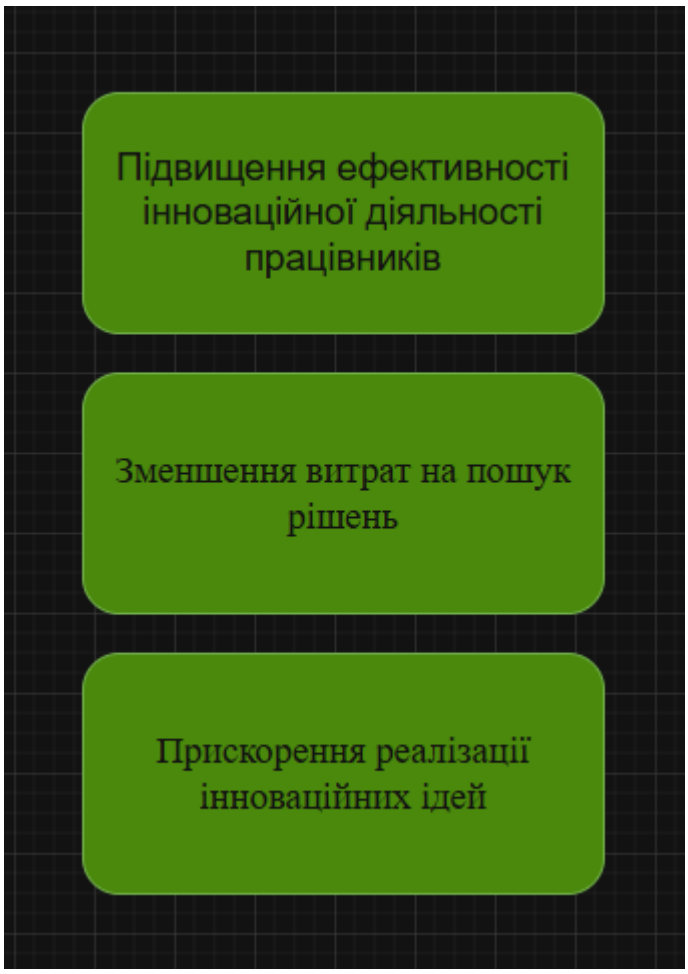


Рис. 2.1. Діаграма бізнес-вимог до веб-платформи цифрової акселерації інновацій в корпоративному середовищі

Функціональні вимоги до системи. Функціональні вимоги — це вимоги до програмного забезпечення, які описують внутрішню роботу системи, її поведінку: калькулювання даних, маніпулювання даними, обробка даних та інші специфічні функції, які має виконувати система.

Функціональні вимоги:

1) Реєстрація та автентифікація користувачів;
2) Подання інноваційної ідеї користувачем;
3) Автоматичне групування ідей за категоріями;
4) Пошук релевантних рішень для реалізації ідей;
5) Оцінювання альтернатив за заданими критеріями;
6) Механізм голосування та оцінювання ідей;
7) Особистий кабінет користувача;

8) Панель адміністратора з управління платформою;

9) Автоматичне формування аналітичних звітів;

Діаграма функціональних вимог для системи роботи інформаційної системи нейромережею для платформи аналізу акселерації у бізнесі *EnterpriseArchitect* та показана на рис. 2.2

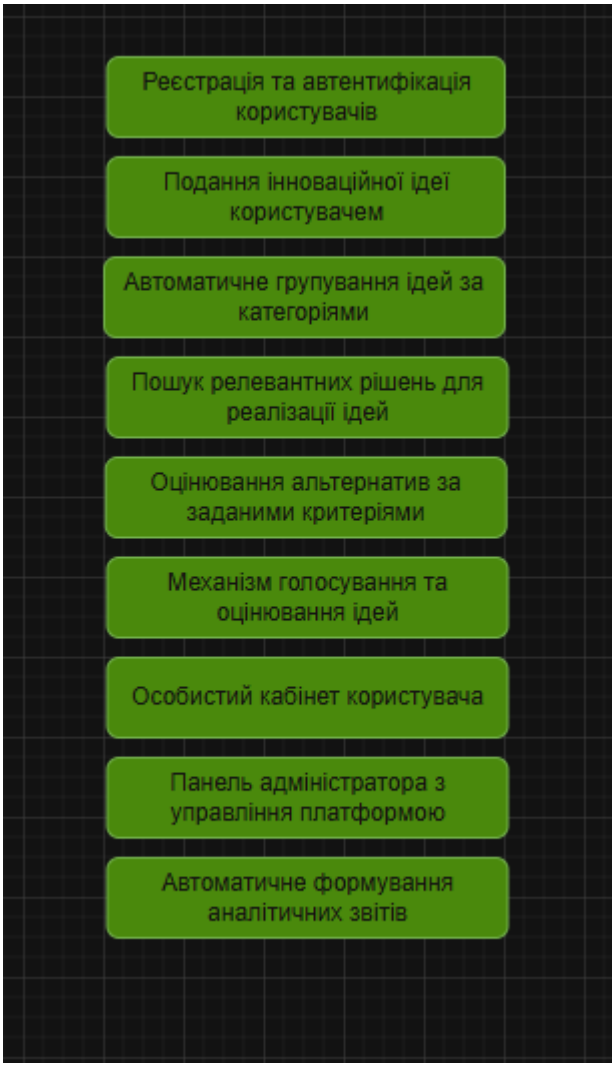


Рис. 2.2. Діаграма функціональних вимог для ІС.

Нефункціональні вимоги до системи. Нефункціональні вимоги — це вимоги до програмного забезпечення, які задають критерії для оцінки якості його роботи. На відміну від функціональних вимог, які визначають що система повинна робити, нефункціональні вимоги визначають якою система повинна бути.

Для забезпечення ефективного функціонування системи цифрової підтримки акселерації інноваційних проєктів, до неї висуваються такі класифікації нефункціональних вимог:

Інтерфейсні вимоги:

Адаптивність до різних пристроїв — система повинна коректно працювати на екранах різного розміру: смартфонах, планшетах, ноутбуках, десктопах, включаючи підтримку адаптивної верстки для комфортної взаємодії;

Інтуїтивно зрозумілий інтерфейс — користувач має змогу легко орієнтуватися в системі без потреби в додаткових інструкціях або навчанні;

Візуальна привабливість — дизайн повинен бути професійним, не перевантаженим, з використанням м'яких кольорів та сучасної графіки, щоб сприяти довготривалому комфортному користуванню;

Операційні вимоги:

Круглодобовий доступ до системи (24/7) — система має бути постійно доступною для користувачів з мінімальними простоями;

Регулярне збереження даних — усі дані повинні зберігатися в базі даних з резервним копіюванням щонайменше раз на добу;

Журналювання подій (логи) — усі дії в системі мають бути зафіксовані у логах для можливості моніторингу, діагностики та аналізу;

Умови розміщення серверної частини — обладнання повинно працювати в стабільних кліматичних умовах без впливу прямого сонячного світла, з хорошою вентиляцією та захистом від вологи.

Інтеграція корпоративної айдентики — система має підтримувати вставку логотипу компанії, корпоративних кольорів та стилю, що відповідає бренду.

Програмні вимоги:

Мова програмування — основна мова розробки: Python;

СУБД — використання систем управління базами даних PostgreSQL або SQLite для збереження інформації про користувачів, ідеї, результати аналізу;

Модульна структура — код має бути розділений за логічними модулями для легшого обслуговування та масштабування;

Актуальні бібліотеки — застосування сучасних бібліотек для реалізації обробки тексту, роботи зі штучним інтелектом, API-запитів, зокрема TensorFlow, Scikit-learn, LangChain тощо;

Операційна сумісність — підтримка встановлення на Windows та Linux-системах;

Пакетне встановлення залежностей — використання менеджера пакетів pip або conda для інсталяції усіх необхідних бібліотек;

Налаштування сервісу БД — передбачене встановлення, запуск і налаштування серверної частини бази даних.

Вимоги до безпеки:

Шифрування даних — усі дані, що передаються або зберігаються в системі, мають бути зашифровані з використанням протоколу не нижче 256-бітного;

Обмеження доступу — доступ до системи мають лише авторизовані користувачі згідно з рівнем прав доступу;

Резервне збереження даних — передбачено дублювання основної БД на резервному сервері;

Архівування звітів — усі важливі аналітичні звіти автоматично зберігаються в цифровому архіві та (за необхідності) виводяться в друковану форму.

Діаграма нефункціональних вимог для системи роботи інформаційної системи нейромережею для розпізнавання облич у розумному будинку реалізована за допомогою менеджера специфікацій *EnterpriseArchitect* та показана на рис. 2.3.

Рис.2.3 Інтерфейсні вимоги

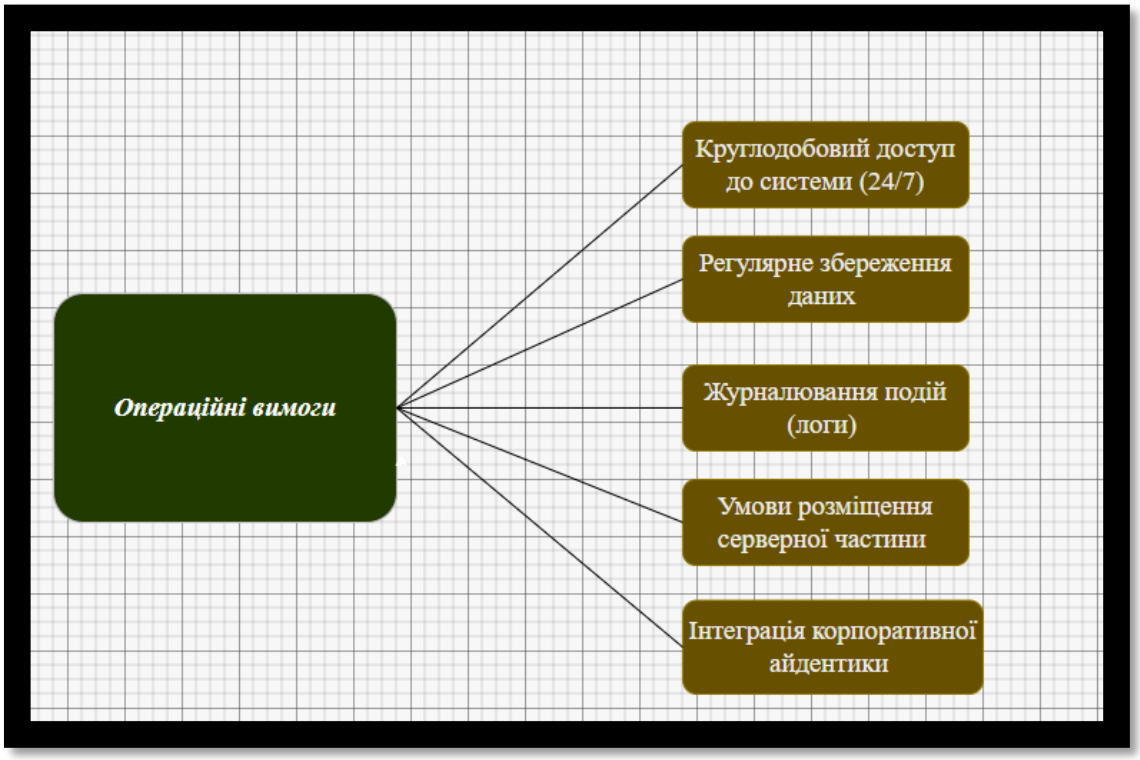


Рис.2.4. Операційні вимоги

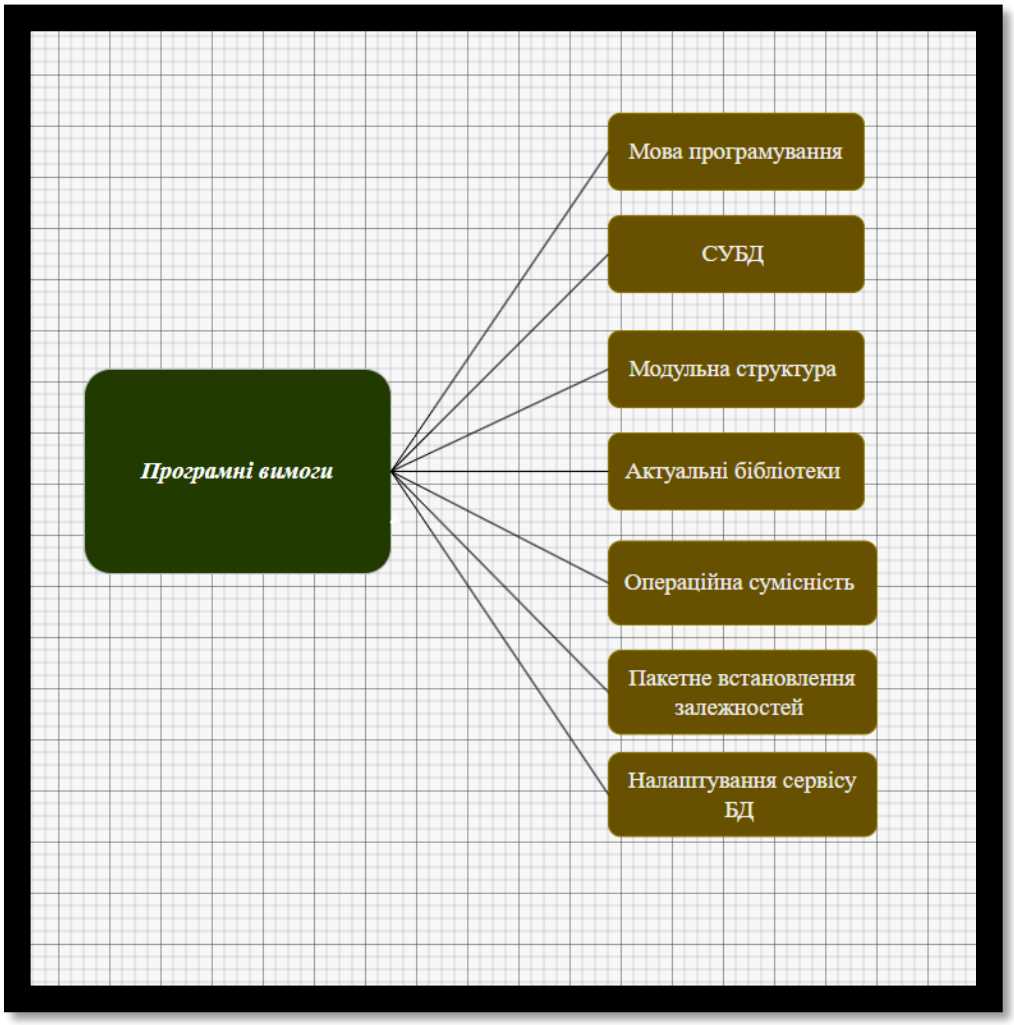


Рис.2.5 Програмні вимоги

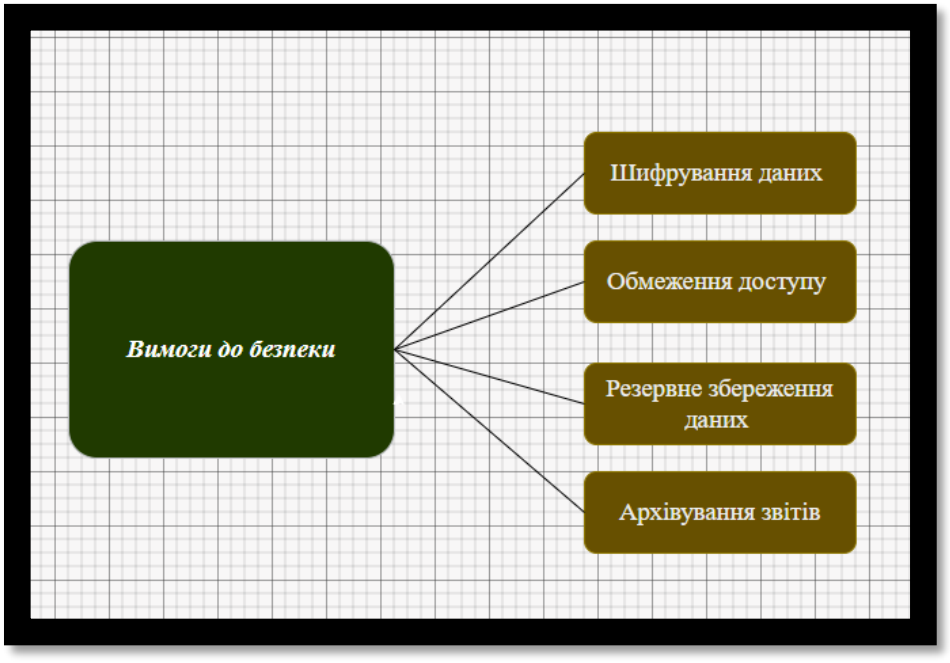


Рис.2.6. Вимоги до безпеки

2.2 Методи та моделі в інформаційних управляючих системах і технологіях

Розробка сучасних інформаційних управляючих систем (ІУС), зокрема в контексті підтримки інноваційної активності в бізнесі, ґрунтується на застосуванні різноманітних методів і моделей, які забезпечують ефективну обробку, збереження, аналіз, класифікацію та представлення інформації. У межах даного дипломного дослідження, ці методи слугують інструментальним підґрунтям для реалізації інтелектуального аналізу поданих ідей, побудови рекомендаційних механізмів та забезпечення цілісного функціонування цифрової системи.

Одним із ключових напрямів є використання методів обробки природної мови (NLP). Завдяки NLP-моделям система може аналізувати текстові описи ідей, автоматично виявляти тематику, ключові слова, інтенції користувача, що суттєво знижує потребу в ручному сортуванні та класифікації. Для цього використовуються моделі типу Bag-of-Words, TF-IDF, а також глибинні мовні моделі на основі трансформерів (наприклад, BERT, DistilBERT).

У системах, де інформація має бути класифікована, зіставлена або оцінена, важливу роль відіграють методи машинного навчання, зокрема:

класифікація — для розподілу ідей за категоріями (напр. маркетинг, логістика, IT);

кластеризація — для групування схожих ідей (алгоритми типу k-means);

аналіз подібності — для пошуку релевантних рішень, продуктів або курсів (cosine similarity, евклідова відстань).

Для обґрунтованого ранжування рекомендованих рішень, у системі можуть бути реалізовані моделі прийняття рішень на основі множинних критеріїв (MCDM), зокрема:

метод аналізу ієрархій (AHP),

метод електрє (ELECTRE),

метод TOPSIS — для вибору найоптимальніших варіантів за кількома параметрами (ціна, рейтинг, доступність, релевантність).

Ще одним важливим аспектом є моделі управління інформаційними потоками, що визначають, як саме ідеї проходять життєвий цикл у системі: від моменту подання, до аналітичної обробки, затвердження, рекомендацій і впровадження. У цьому контексті використовуються процесні моделі (BPMN), діаграми активності (UML), а також моделі станів (state machine models) для формалізації етапів обробки ідеї.

З точки зору збереження та доступу до інформації, застосовуються реляційні моделі даних, що дозволяють ефективно реалізовувати зв'язки між сутностями: користувачами, поданими ідеями, системними рішеннями, історією взаємодії, звітами тощо. На основі таких моделей формуються запити до СКБД, формуються структури даних та проєктується логіка заповнення, редагування і аналізу.

Не менш важливими є і аналітичні моделі, які дозволяють здійснювати оцінку ефективності поданих ініціатив, аналіз популярності тем, відстеження реалізованих ідей та побудову статистичних звітів. Для цього застосовуються математичні методи описової статистики, кореляційного аналізу, а в перспективі — прогностичного моделювання.

Таким чином, інформаційна система, що розробляється в межах даного дослідження, базується на комплексному використанні формальних, статистичних, семантичних та аналітичних методів і моделей, що забезпечують її інтелектуальність, адаптивність і здатність до самонавчання. Саме поєднання сучасних математичних підходів, методів ШІ та технологій управління даними дозволяє реалізувати ефективну цифрову підтримку інноваційних процесів у бізнесі.

. Завданням дипломного проєкту є розробка інформаційної системи у вигляді веб-платформи, яка дозволить співробітникам компанії подавати власні ідеї щодо покращення внутрішніх процесів, сервісів або бізнес-продуктів. Система має реалізовувати автоматизований збір, обробку, класифікацію та аналіз поданих ідей з використанням методів штучного інтелекту. Основною функцією є підтримка цифрової акселерації інноваційних ініціатив за рахунок швидкого пошуку

релевантних рішень — таких як навчальні курси, прикладні IT-продукти або цифрові сервіси.

Інформаційна система повинна оцінювати відповідні варіанти за критеріями вартості, якості та релевантності, і формувати для користувача кілька найоптимальніших пропозицій для реалізації ідеї. Менеджери, аналітики або керівники зможуть здійснювати модерацію, переглядати статистику, затверджувати ідеї та керувати їх реалізацією.

Опис вихідної інформації

Вихідна інформація в системі цифрової підтримки акселерації інноваційних проєктів використовується для формування узагальнених результатів аналізу поданих ідей. На її основі користувач отримує автоматично згенеровані рекомендації щодо можливих способів реалізації своєї ідеї — таких як навчальні курси, цифрові платформи, прикладні сервіси або програмні рішення.

Для менеджерів та адміністраторів вихідна інформація подається у вигляді аналітичних звітів, що включають:

- статистику поданих ідей за напрямами (тематичне групування);
- показники активності користувачів (кількість ідей, голосів, переглядів);
- зведення щодо реалізованих ініціатив;
- ефективність впроваджених рішень у розрізі часу, вартості та очікуваної користі.

Інформація виводиться у вигляді списків, таблиць, діаграм та графіків — у зручному для аналізу вигляді, відповідно до ролі користувача (працівник, експерт, адміністратор). Це дозволяє системі виступати інструментом не лише для генерації ідей, але й для їхньої структурної оцінки та подальшого супроводу у процесі реалізації.

На основі вихідної інформації для адміністратора проводиться контроль за статистикою відвідування: кількість створених постів та зареєстрованих користувачів.

Перелік та опис вихідних повідомлень наведено у таблиці 2.1.

Назва вихідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
Результат аналізу ідеї	IDEA_ANALYSIS_RESULT	Текст, графік, таблиця	Одноразово — після подання ідеї	Модуль штучного інтелекту
Рекомендовані рішення	RECOMMENDED_SOLUTIONS	Таблиця з посиланнями	Одноразово — після аналізу ідеї	База даних + зовнішні API
Рекомендовані навчальні курси	RECOMMENDED_COURSES	Таблиця, гіперпосилання	Одноразово — при кожному новому запиті	EdTech-сервіси (Coursera тощо)
Звіт про активність користувача	USER_ACTIVITY_REPORT	PDF / HTML	Щомісяця або за запитом	База даних
Звіт про популярні категорії ідей	IDEA_TRENDS_REPORT	Графік, діаграма	Щотижня	База даних, AI-модуль
Звіт для менеджера про реалізацію ідей	MANAGEMENT_STATUS_REPORT	Таблиця, коротке зведення	Раз на тиждень / за запитом	База даних
Статус розгляду конкретної ідеї	IDEA_STATUS_INFO	Текстове повідомлення	За запитом користувача	Панель адміністратора
Сповіщення про оновлення або схвалення ідеї	NOTIFICATION_IDEA_UPDATE	Push / Email / Web-сповіщення	Негайно після зміни статусу	Система повідомлень

Таблиця 2.1

Перелік і опис вихідних повідомлень

Назва вхідного повідомлення	Ідентифікатор	Форма подання	Термін і частота надходження	Джерело
Ідея користувача	idea_form	Текст + прикріплені файли	Одноразово при створенні ідеї	Користувач платформи
Форма авторизації/реєстрації	auth_form	Форма входу/реєстрації	Під час входу користувача	Користувач платформи
Користувачі (профілі)	users	Масив даних	Одноразово, оновлення за потреби	База даних системи
Опис ідеї з тегами	description	Текстове поле	З кожною ідеєю	Форма подання ідеї
Ключові слова	keywords	Автоматично витягуються	Автоматично при створенні ідеї	Аналіз тексту ідеї
Історія впроваджених ідей	idea_history	Масив даних	Періодично, за результатами реалізації	Історичні дані
Каталоги курсів	courses	API-запити	Періодично, з оновленням	Зовнішні освітні платформи
База цифрових рішень	solutions	JSON/API	Постійно оновлювана база	Внутрішні та зовнішні сервіси
Оцінки користувачів	ratings	Форма з оцінкою/балами	Після взаємодії з ідеями	Користувачі системи

Опис вхідної інформації. Вхідна інформація використовується для створення авторизації та реєстрації користувачів, збір інформації для аналізу образів.

Перелік та опис вхідних повідомлень наведено у таблиці 2.2.

Таблиця 2.2

Перелік і опис вхідних повідомлень

Використана інформація. Для розв'язаннязадачі керування інформаційної системи нейромережею для розпізнавання облич у розумному будинку у інформаційній системі використовується інформація, що зберігається в масивах у базі даних, а також інформація яку надають користувачііінформаційної системи нейромережею для розпізнавання облич у розумному будинку в процесі використання системи.

Перелік використаних джерел інформації:

1) довідникмешканців;

2) запити на початок перевірки;

3) адміністратори;

4) відеозаписи.

Перелік та опис масивів використовуваної інформації подану в таблиці 2.3.

Таблиця 2.3

Перелік масивів використовуваної інформації

Масив	**Ідентифікатор**	**Максимальна кількість записів**
Ідеї користувачів	ideas	100000
Користувачі платформи	users	10000
Оцінки/голоси за ідеї	ratings	100000
Рекомендовані рішення	solutions	50000
Навчальні курси	courses	50000
Історія реалізації ідей	idea_history	50000
Категорії ідей	categories	1000

Журнал дій адміністратора	admin_log	5000

Результати розв'язання. Результатом реалізації задачі цифрової підтримки акселерації інноваційних проєктів у бізнесі є створення інформаційної системи, яка дозволяє працівникам компанії подавати власні ідеї щодо вдосконалення процесів або розвитку продуктів, а також автоматично отримувати релевантні рішення для їх реалізації.

Перелік і опис масивів результатної інформації подано в табл. 2.4.

Таблиця 2.4.

Перелік масивів результатної інформації

Масив	**Ідентифікатор**	**Максимальна кількість записів**
Результати аналізу ідеї	RESULT_INFO	100000
Аналітичні звіти	REPORT	50000
Рекомендовані рішення	RECOMMENDATIONS	50000
Статус реалізації ідеї	IMPLEMENTATION_STATUS	50000
Історія голосувань	VOTING_HISTORY	100000
Відгуки/оцінки користувачів	USER_FEEDBACK	50000

Алгорит розв'язання задачі на ЕОМ. Алгоритм розв'язання задачі використання ІС.

На початку роботи веб-платформи відображається головна сторінка із формою входу або реєстрації. Після успішної авторизації користувач отримує доступ до персонального кабінету, де може подати нову ідею за допомогою відповідної форми. У формі користувач заповнює назву ідеї, опис, ключові слова, прикріплює файли (за потреби), після чого надсилає її на обробку.

Після подачі ідеї система автоматично ініціює аналіз поданої інформації. За допомогою алгоритмів штучного інтелекту (нейромереж або NLP-моделей) ідея класифікується за напрямом бізнесу, типом проблеми або сфери впровадження. Далі система звертається до баз зовнішніх джерел (каталогів курсів, платформ, сервісів) для пошуку потенційних рішень, які можуть допомогти в реалізації запропонованої ініціативи.

Знайдені рішення оцінюються за критеріями: відповідність суті ідеї, якість, вартість, рейтинги та інші релевантні показники. Система формує 2–3 найкращі рекомендації для користувача. Одночасно запис про ідею, статус її розгляду та результати аналізу заносяться до бази даних.

На наступному етапі користувач отримує повідомлення про результат обробки, а менеджери або адміністратори можуть розглянути ідею, затвердити її або направити на доопрацювання. У разі схвалення — ідея потрапляє до підсистеми впровадження, де формується план дій і розпочинається реалізація.

Весь процес супроводжується автоматичним збереженням історії дій, результатів оцінювання, голосів інших користувачів, що дозволяє вести аналітику та генерувати звіти про ефективність інноваційної активності. На рисунку 2.5. наведено алгоритм вирішення поставленої задачі.

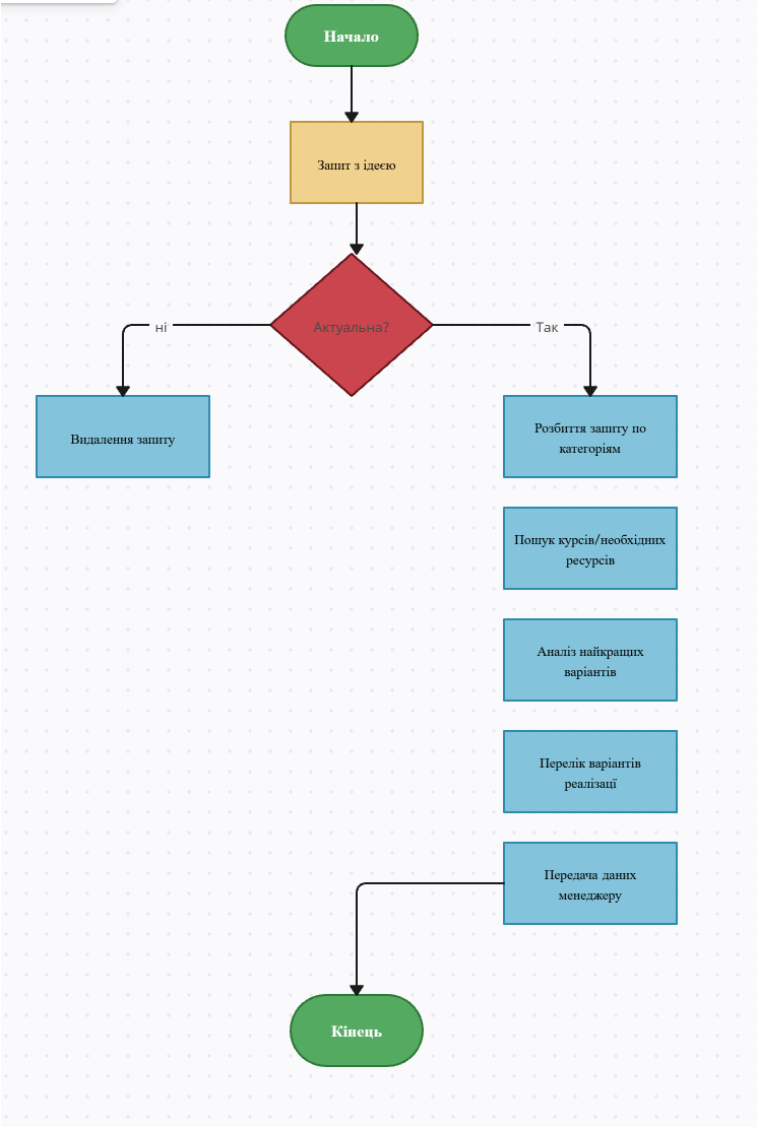


Рис 2.9. Схема алгоритму розв’язання задачі

2.3. Моделювання інформаційної підсистеми

Моделювання поведінки системи

Моделювання поведінки розробленої інформаційної системи цифрової підтримки акселерації інноваційних проєктів у бізнесі представлено у вигляді діаграми прецедентів (Use Case Diagram). Така діаграма дозволяє узагальнено візуалізувати функціональні вимоги до системи через типові сценарії взаємодії між користувачами (акторами) та функціональними компонентами платформи.

Діаграма прецедентів являє собою графічну модель, що містить:

акторів — учасників, які взаємодіють із системою (наприклад, працівник, адміністратор, менеджер);

прецеденти — окремі функціональні можливості системи (наприклад, подати ідею, отримати рекомендацію, розглянути ініціативу);

межі системи — функціональну область, у межах якої реалізуються всі можливості;

асоціації — зв’язки між акторами і прецедентами;

відношення включення, розширення або узагальнення — зв’язки між варіантами використання.

Розроблена діаграма прецедентів створена на основі описаних раніше функціональних вимог до системи, таких як авторизація користувачів, подання інноваційних ідей, аналіз і класифікація ідей, пошук рішень за допомогою ШІ, формування рекомендацій, голосування, перегляд звітів та керування статусами реалізації.

Таким чином, діаграма прецедентів забезпечує формалізоване подання логіки взаємодії користувачів із системою та слугує основою для подальшого проектування архітектури програмного забезпечення.1

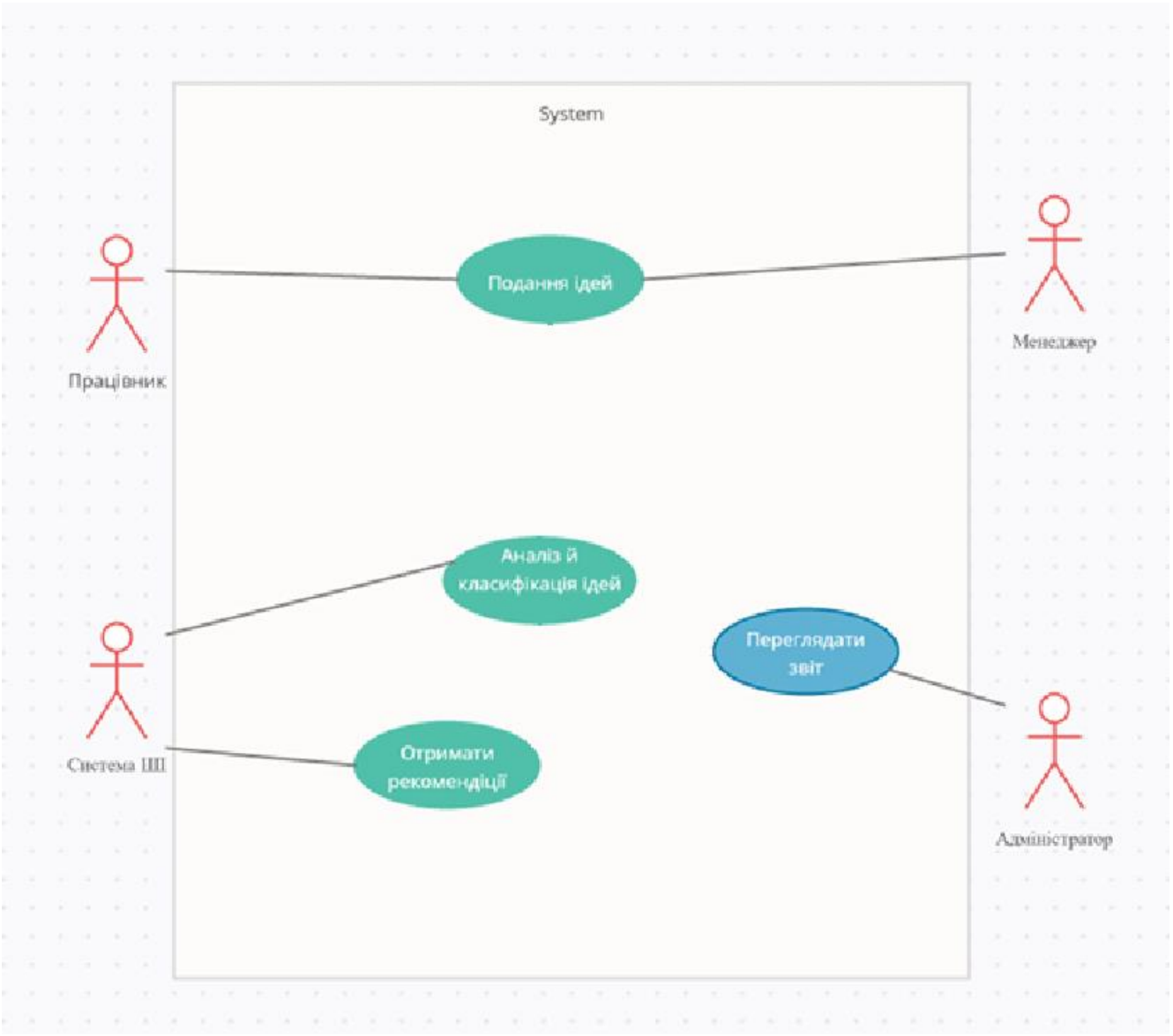


Рис. 2.10. Діаграма прецедентів

Сценарій — це конкретне виконання прецеденту, тобто один повний прохід по потоку подій у межах певного варіанту використання системи. Кожному прецеденту (випадку використання) відповідає множина можливих сценаріїв, які деталізують, як саме актор взаємодіє з системою на певному етапі. Для кожного такого сценарію може бути побудована діаграма активності (UML Activity Diagram), яка графічно описує послідовність дій та переходів між ними.

У межах цієї дипломної роботи розглядаються типові сценарії використання інформаційної системи цифрової підтримки інноваційних проєктів у бізнесі з боку користувача веб-платформи. Зокрема, сценарії охоплюють процеси авторизації,

подачі ідеї, автоматичного аналізу, пошуку відповідних рішень, отримання рекомендацій, голосування та подальшої взаємодії з результатами.

Нижче буде подано перелік базових сценаріїв використання платформи та відповідні до них діаграми активності, що дозволяють візуально відобразити логіку поведінки системи під час кожного з випадків використання.

Математичний опис. Математичний опис — це формалізоване подання реальної або абстрактної системи у вигляді математичних моделей, що дозволяє досліджувати її поведінку, структуру та взаємозв'язки між складовими. У контексті дипломної роботи математичний опис є невіддільною частиною наукового аналізу, яка забезпечує точність, однозначність та можливість проведення об'єктивних розрахунків, прогнозів або моделювання процесів, що вивчаються.

У науковій практиці математичний опис застосовується для побудови моделей, які відображають динаміку, залежності та функціонування об'єкта дослідження. Наприклад, у разі створення інформаційної системи або цифрової платформи, математичний опис може включати формалізацію алгоритмів, логіки взаємодії користувача з системою, моделі обробки даних, оптимізаційні задачі чи стохастичні процеси. Це дозволяє не лише чітко структурувати дослідження, але й забезпечити його реплікацію або верифікацію іншими дослідниками.

Математичний опис, як правило, базується на використанні таких елементів, як:

Змінні — характеристики об'єкта, що змінюються в часі або просторі;

Параметри — сталі величини, що визначають властивості системи;

Функціональні залежності — рівняння або функції, що встановлюють взаємозв'язки між змінними;

Обмеження — умови, що визначають допустимі області існування рішень;

Цільові функції — у випадку оптимізаційних задач визначають мету моделювання (наприклад, мінімізація витрат або максимізація ефективності).

Особливість математичного опису полягає в його універсальності: він може бути застосований у технічних, економічних, соціальних, інформаційних та інших системах. У дипломній роботі його наявність свідчить про глибоке розуміння

предметної області та здатність дослідника до аналітичного мислення й застосування методів формалізації для вирішення прикладних задач.

Таким чином, математичний опис є ключовим інструментом об’єктивного вивчення складних систем, що підвищує наукову якість дослідження та сприяє впровадженню отриманих результатів у практику. Його важливо подавати у чіткому, логічно послідовному вигляді, із поясненням кожного елементу моделі, щоб забезпечити повне розуміння досліджуваного об'єкта.

Загальна модель системи

$$S=\langle U,I,P,R,T,D,O\rangle$$

де:

U — множина користувачів, які взаємодіють із системою. Це можуть бути працівники компанії (ініціатори ідей), менеджери, що ухвалюють рішення, або адміністратори, які керують функціонуванням системи;

I — множина ідей, які вводяться в систему. Кожна ідея подається через спеціальну форму і містить опис, теги, можливу категорію та прикріплені файли;

P — множина параметрів, що характеризують ідею (наприклад: текст опису, ключові слова, напрямок). Вони необхідні для її обробки;

R — множина можливих рішень, які система може рекомендувати. Це можуть бути цифрові продукти, онлайн-курси, автоматизаційні сервіси, CRM-платформи тощо;

T — множина функцій, що реалізують логіку обробки ідей: класифікація, пошук, оцінювання, ранжування;

D — база даних, у якій зберігаються всі записи системи, включно з ідеями, користувачами, результатами аналізу та історією дій;

O — множина результатів, які система формує у відповідь на подану ідею. Це можуть бути рекомендації, звіти, статус реалізації.

Пошук відповідних рішень

$$t2(i)=\{r1,r2,...,rn\}\subseteq R$$

Ці рішення можуть бути знайдені:

У внутрішній базі (внутрішні цифрові ресурси компанії);

Через зовнішні API (наприклад, Coursera, HubSpot, Notion, Airtable);

або з відкритих джерел даних.

Нижче наведено всі можливі сценарії використання системи:

1. Алгоритм реєстрації користувача

- Користувач відкриває форму реєстрації.
- Вводить ПІБ, email, пароль.
- Система перевіряє, чи email не зайнятий.
- Якщо email унікальний — створюється обліковий запис.
- Користувач отримує повідомлення про успішну реєстрацію.

2. Алгоритм авторизації користувача

- Користувач вводить логін та пароль.
- Система перевіряє відповідність у базі даних.
- У разі успішного входу відкривається персональний кабінет.
- Якщо дані не збігаються — виводиться повідомлення про помилку.

3. Алгоритм подачі ідеї

- Користувач відкриває розділ «Подати ідею».
- Заповнює назву, опис, ключові слова.
- Прикріплює за потреби файли.
- Натискає кнопку «Надіслати».
- Ідея зберігається у базі даних зі статусом «Очікує розгляду».

4. Алгоритм класифікації ідеї

- Система аналізує текст ідеї.
- Виділяє ключові слова за допомогою NLP.

- Порівнює із заздалегідь заданими категоріями.
- Визначає напрям ідеї (наприклад, маркетинг, логістика).

5. Алгоритм пошуку рішень

- Система формує пошуковий запит на основі тематики ідеї.
- Звертається до зовнішніх API або баз даних.
- Отримує список відповідних курсів, сервісів або інструментів.
- Створює тимчасовий список рішень для подальшої оцінки.

6. Алгоритм оцінки ефективності рішень

- Для кожного знайденого рішення обчислюється:
    1. якість (за рейтингами),
    2. вартість,
    3. релевантність до ідеї.
- Застосовується формула з ваговими коефіцієнтами.
- Рішення сортуються за результатом оцінки.

7. Алгоритм формування рекомендацій

- Вибираються 2–5 рішень з найвищими оцінками.
- Формується блок з описом та посиланнями.
- Рекомендації відображаються користувачу.
- Результат зберігається у базі даних.

8. Алгоритм голосування за ідеї

- Користувач відкриває перелік ідей інших учасників.
- Обирає ідею, яку хоче підтримати або не підтримати.
- Система фіксує голос.

- Оновлює загальний рейтинг ідеї.

9. Алгоритм перегляду статусу ідеї

- Користувач заходить у розділ «Мої ідеї».
- Система виводить статус кожної (очікує, затверджено, реалізовано).
- Дані автоматично оновлюються після змін у БД.

10. Алгоритм формування аналітичного звіту

- Менеджер відкриває модуль аналітики.
- Система збирає статистику щодо:
    1. кількості ідей,
    2. активності користувачів,
    3. ефективності рішень.
- Будуються таблиці, графіки.
- Звіт експортується у PDF або Excel.

11. Алгоритм керування ролями користувачів

- Адміністратор відкриває профіль користувача.
- Вибирає нову роль (користувач, модератор, адміністратор).
- Підтверджує зміну.
- Система оновлює роль у базі даних.

12. Алгоритм логування дій

- Система автоматично фіксує ключові дії:
    1. логін, реєстрацію, подачу ідеї, голосування.
- Для кожної дії записується час, тип та ID користувача.
- Адміністратор може переглянути журнал активності.

13. Алгоритм резервного копіювання

- За розкладом запускається процес створення резервної копії.

- База даних зберігається на захищеному сервері.
- Створюється запис про копіювання з позначкою часу.

14. Алгоритм надсилання сповіщень

- Система перевіряє події, які потребують сповіщення (зміна статусу, нові коментарі).
- Створює текст повідомлення.
- Надсилає email або push-сповіщення користувачу.

15. Алгоритм фільтрації ідей

- Користувач вибирає фільтр: категорія, статус, популярність.
- Система застосовує умови до всієї множини ідей.
- Виводить лише релевантні записи.

16. Алгоритм архівації ідей

- Система перевіряє активність ідей.
- Ідеї, не змінювані понад 6 місяців, позначаються як архівні.
- Переносяться в окремий архів або таблицю.

17. Алгоритм генерації дашборду

- Система щоденно або в реальному часі агрегує ключові дані.
- Відображає на панелі адміністратора:
    1. кількість нових ідей,
    2. активність користувачів,
    3. реалізовані проєкти,
    4. середню оцінку ефективності.

Моделювання структури системи. Структура інформаційної системи — це сукупність її функціональних елементів та зв’язків між ними, які забезпечують цілісне функціонування системи як єдиного інструменту цифрової підтримки інноваційного розвитку.

Для моделювання структури створюваної інформаційної системи цифрової акселерації інновацій у бізнесі нами було використано **UML-діаграму класів**, яка дозволяє наочно представити основні сутності (класи), їхні атрибути, методи, а також логічні зв’язки між ними.

UML-діаграма класів ілюструє взаємозв’язок між основними компонентами системи, такими як: користувач, ідея, рішення, курс, звіт, адміністратор, система аналізу. Кожен клас описує реальну сутність у системі з відповідними атрибутами та діями, які вона виконує. Взаємозв’язки між класами побудовані за принципами спадкування, агрегації та асоціації.

На рисунку 2.12 наведено діаграму класів, розроблену на основі функціональної моделі системи.

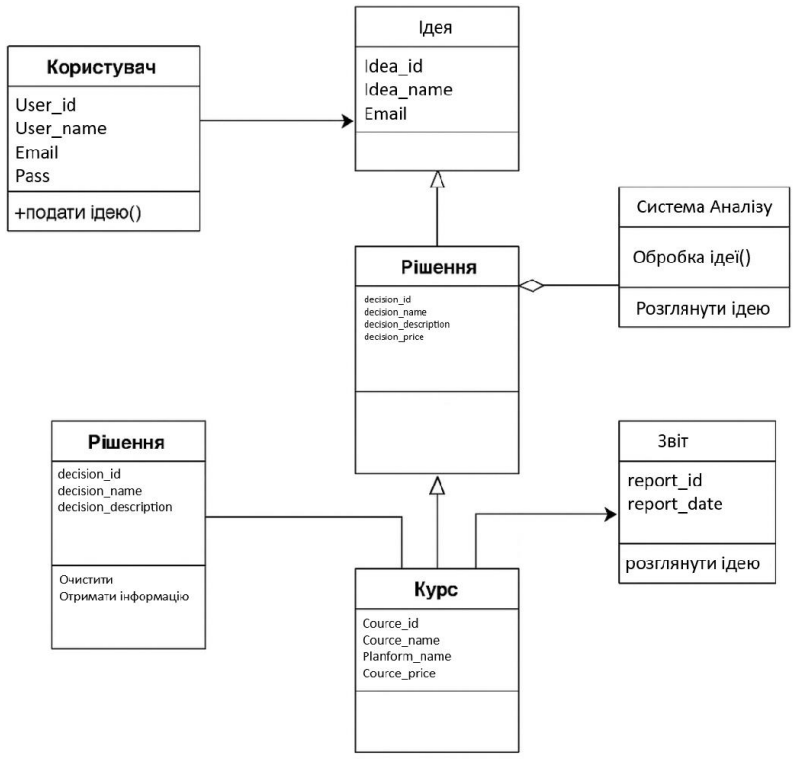


Рис. 2.24. UML-діаграма класів інформаційної системи.

У другому розділі дипломної роботи було проведено комплексне проєктування інформаційної системи цифрової підтримки акселерації інноваційних проєктів у бізнесі. На початковому етапі було здійснено структурний

аналіз предметної області, визначено функціональне призначення системи та сформульовано мету її створення як інструменту автоматизації процесу обробки інноваційних ідей.

Були визначені ключові користувацькі ролі (акторів), описано логіку взаємодії з системою та побудовано діаграму прецедентів (Use Case Diagram), що відображає основні варіанти використання системи у межах визначених функціональних вимог. Дана діаграма дозволила систематизувати основні сценарії взаємодії користувача з платформою: подання ідеї, отримання рекомендацій, перегляд статусів, голосування тощо.

У подальшому було здійснено аналіз функціональних і нефункціональних вимог до інформаційної системи, а також сформульовано бізнес-вимоги, які описують економічну доцільність створення такої платформи для бізнес-середовища. Було розроблено структуровані вимоги до інтерфейсу, операційного середовища, програмного забезпечення, а також визначено критерії продуктивності, безпеки та ефективності.

З метою візуалізації сценаріїв використання системи були побудовані діаграми активностей (Activity Diagrams) для ключових прецедентів. Це дозволило деталізувати кожен потік подій у системі та моделювати відповідні бізнес-процеси.

Окрему увагу було приділено математичному опису системи: наведено формальну множинну модель, описано структуру вхідних і вихідних даних, визначено функції трансформації, класифікації, пошуку та ранжування рішень, а також побудовано формули для оцінки ефективності системних рекомендацій.

Також було реалізовано UML-діаграму класів, яка формалізує структуру системи, відображає основні сутності (користувач, ідея, рішення, звіт, адміністратор тощо), їх атрибути, методи та зв’язки між класами. Діаграма демонструє логічну архітектуру системи та взаємозв’язки між її об’єктами.

Крім того, були детально описані основні алгоритми, що реалізуються в системі: реєстрація та авторизація користувача, подання ідеї, класифікація, пошук рішень, голосування, формування рекомендацій, генерація звітів тощо. Для

кожного алгоритму було подано послідовність дій, що визначає поведінку системи на відповідному етапі.

Таким чином, у межах другого розділу було повністю опрацьовано логічну, структурну та функціональну модель майбутньої системи, що забезпечує основу для її технічної реалізації в наступних етапах роботи.

РОЗДІЛ 3 РОЗРОБЛЕННЯ ПРОЕКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ

3.1 Проектування бази даних та/або сховища даних для інформаційної управляючої системи

У процесі розробки інформаційної системи цифрової підтримки акселерації інноваційних проєктів у бізнесі мною було спроєктовано спеціалізовану базу даних, яка забезпечує ефективне зберігання, структуровану обробку та доступ до інформації, необхідної для функціонування системи.

Розробка бази даних стала важливим етапом, оскільки вся логіка обміну, фіксації й аналітики в системі залежить від того, наскільки грамотно побудовані таблиці, зв’язки між сутностями, типи даних і правила обробки. При цьому я керувався вимогами до функціоналу системи, які були сформовані на основі аналізу аналогів, потреб бізнес-користувачів, а також результатів опрацювання предметної області.

На початковому етапі я створив інфологічну модель, у якій описав основні сутності системи: користувачі, ідеї, рішення, звітність, категорії, а також взаємозв’язки між ними. Така модель дозволила виявити всі атрибути та сформувати логіку взаємодії об’єктів у системі. Для зручності візуалізації використовувалось середовище Draw.io, у якому було зображено всі зв’язки, включно з типами зв’язків: один до одного, один до багатьох, багато до багатьох.

На наступному етапі здійснено перехід до даталогічного рівня, з урахуванням особливостей обраної системи керування базами даних — PostgreSQL. Саме PostgreSQL було обрано як основну СКБД, оскільки вона підтримує всі необхідні можливості: транзакції, зв’язки зовнішніх ключів, перевірки унікальності, індексацію, зберігання JSON-структур, та сумісність із мовою програмування Python, яка використовується в серверній логіці системи.

У базі даних були реалізовані такі основні таблиці:

users — зберігає інформацію про зареєстрованих користувачів системи (ідентифікатор, ім’я, електронна пошта, роль у системі);

ideas — основна таблиця з поданими інноваційними ідеями (назва, опис, дата подачі, ключові слова, автор, статус розгляду);

decisions — таблиця з результатами аналізу ідей, де зберігаються рекомендовані курси, сервіси чи інструменти, які були підібрані автоматично;

reports — модуль звітності для збереження історії активності, взаємодії з ідеєю, коментарів, оцінок;

допоміжні таблиці categories, keywords, logs, access_history.

Усі таблиці пов’язані між собою через зовнішні ключі. Наприклад, ideas.user_id зв’язується з users.id, а кожен запис у таблиці decisions обов’язково пов’язаний з конкретною ідеєю через idea_id. Така структура дозволяє не лише ефективно зберігати дані, а й легко здійснювати вибірку, сортування, аналітику та побудову звітів.

Під час проєктування я виконав нормалізацію структури бази даних до третьої нормальної форми (3НФ), що дозволило уникнути дублювання інформації та прискорити виконання запитів. Була передбачена індексація по найчастіше використовуваних полях — таких як idea_status, user_email, date_created.

Особливу увагу я приділив масивам результатної інформації, зокрема — автоматично згенерованим рекомендаціям, які можуть бути збережені не лише у вигляді тексту, а й у форматі JSON — для можливості швидкої обробки клієнтською частиною.

Окремим пунктом було опрацювання питань безпеки: у структурі бази даних були передбачені поля для шифрування паролів, логування доступів, фіксації авторизацій і блокування небажаних дій.

Результатом цього етапу стало створення гнучкої, масштабованої та легко підтримуваної структури бази даних, яка є основою функціонування всієї цифрової платформи. Надалі база може бути розширена за рахунок додавання нових сутностей — наприклад, метрик ефективності впроваджених ідей або модуля для візуалізації KPI на основі зібраних звітів.

Таким чином, база даних є не лише сховищем, а й інтелектуальним ядром системи, яке забезпечує збереження знань, історію рішень та зв'язок між користувачами, ідеями та інструментами впровадження. Це дозволяє реалізувати повноцінну цифрову підтримку інноваційної діяльності у бізнес-середовищі.

3.2 Проектування бази знань інформаційної управляючої системи та/або засоби інтелектуального аналізу даних

3.2.1 Організація та проведення інтелектуального аналізу даних з метою пошуку знань в базі даних

Загальна характеристика інформаційного забезпечення.

Інформаційне забезпечення (ІЗ) є складовою частиною функціонування інформаційної системи та охоплює сукупність структурованих даних, форм подання інформації, нормативної бази, а також технічних рішень щодо обсягів, способів обробки, збереження, захисту та доступу до інформаційних ресурсів у межах створеної платформи.

Інформаційне забезпечення розроблене з урахуванням функціонального призначення системи цифрової підтримки інноваційних ідей, що передбачає обробку пропозицій працівників, класифікацію за напрямами, пошук рішень і формування рекомендацій. Процес проєктування ІЗ було здійснено з дотриманням таких принципових вимог:

Інформаційне забезпечення повинно гарантувати повну підтримку усіх функцій системи, включаючи подання ідей, їх класифікацію, збереження, пошук, оцінку рішень, голосування та генерацію звітів.

Усі інформаційні масиви організовано у вигляді реляційної бази даних, яка включає взаємопов’язані таблиці для ідей, користувачів, рішень, голосів, звітів та дій адміністратора.

Система передбачає регулярне оновлення, перевірку та резервне копіювання інформації, а також захист від втрати або пошкодження даних у разі виходу з ладу технічного середовища або серверного забезпечення.

Інформація має бути цілісною та ідентичною у всіх пов’язаних модулях, що гарантує відсутність дублювання або розбіжностей у логічних записах бази даних.

Інформаційне забезпечення платформи передбачає захист даних від несанкціонованого доступу шляхом впровадження багаторівневої системи авторизації, обмеження доступу до окремих даних за ролями, а також логування всіх дій користувачів.

Гнучкість ІЗ дозволяє розширювати структуру бази даних відповідно до потреб бізнесу, впроваджувати нові типи ідей, категорій, типів рішень або оновлювати методики оцінювання ефективності.

У структурі інформаційного забезпечення центральне місце займають інформаційні об’єкти — логічні сутності предметної області, які мають певні атрибути та описують типову одиницю інформації, з якою працює система.

До ключових інформаційних об’єктів системи належать:

Користувачі — містять інформацію про авторів ідей, їх контактні дані, роль у системі, історію активності.

Ідеї — записи з описом пропозицій користувачів, тегами, категоріями, статусами, датою створення.

Рішення — варіанти цифрових продуктів або курсів, які система рекомендує для реалізації ідеї.

Голосування — дані про підтримку ідей іншими користувачами.

Адміністратори та менеджери — мають доступ до модерації, перегляду статистики, зміни статусів.

Звіти — агреговані аналітичні дані для управління інноваційною діяльністю.

Кожен інформаційний об’єкт містить набір атрибутів (реквізитів), які є логічно неподільними одиницями — наприклад, для об’єкта «Ідея» це: id, назва, опис, теги, автор, дата створення, статус.

Таким чином, інформаційне забезпечення відіграє ключову роль у забезпеченні стабільної, структурованої та безпечної роботи системи, створюючи основу для прийняття управлінських рішень на основі оброблених інноваційних ініціатив.

3.2.2 Проєктування та реалізація бази знань

Даталогічна модель бази даних. Після формування інфологічної моделі даних інформаційної системи, наступним етапом проєктування є побудова даталогічної (логічної) моделі бази даних, яка відображає структуру даних у термінах обраної системи керування базами даних (СКБД). Логічна модель є результатом процесу відображення інфологічної моделі на рівень, сумісний із технічними засобами зберігання та обробки інформації.

Даталогічна модель представляє структуру бази даних у формі таблиць, полів, типів даних, зв’язків між таблицями та обмежень цілісності, що формуються відповідно до вимог обраної СКБД. Така модель має бути повністю орієнтована на реальні технічні засоби реалізації, підтримувати індексування, первинні та зовнішні ключі, механізми транзакцій і ролей доступу.

Перед виконанням логічного проєктування нами було обрано конкретне середовище для реалізації бази даних — наприклад, PostgreSQL, що є потужною реляційною СКБД з підтримкою розширеної типізації, перевірки цілісності, тригерів і засобів оптимізації запитів. Саме вибір СКБД визначає набір допустимих операцій, типів даних, обмежень і засобів розгортання, що безпосередньо впливає на структуру та логіку побудови моделі.Під час переходу до логічного рівня враховано: вимоги до збереження великих обсягів текстової інформації (наприклад, описів ідей); потребу у зв’язках між таблицями (реалізовано за допомогою зовнішніх ключів); обов’язковість підтримки ролей користувачів та захисту даних (через механізми контролю доступу); цілісність даних, яка забезпечується обмеженнями NOT NULL, UNIQUE, CHECK та FOREIGN KEY.

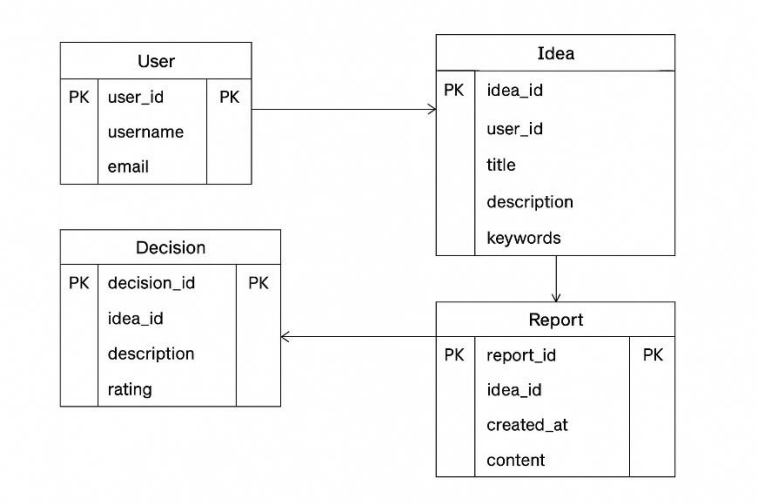


Рис. 3.2.Даталогічна модель

Для створення та для наступної роботи з БД було вибрано ресурс SQLite, оскільки він не є ресурсозатратним, простим для використання. Інтерфейс інтуїтивно-зрозумілий, тому проблеми з ним відсутні. Є можливість реалізації усіх необхідних запитів.

Передача даних перш за все відбудеться завядки міграції інформації з БД вже існуючого ОСББ, якщо таке відсутнє то з БД ЖЕКу, що відповідає за певну територію. Оскільки надання інформації не є проблемою, створення фото може стати проблемою, проте ксерокопії паспортів є, тому її скан для отримання фотокартки не є труднощами, додавання нових користувачів відбувається шляхом створення знімку та заповненнямз адреси та квартири, до якої йому надають доступ. Адміністратор додається за інформаціюєю, що він надав голові чи головам ОСББ. Запис відео відбувається постійно, запис з відеокамери прямує на жорсткий диск, та шлях до файлу зберігається до БД за для швидкого знаходження необхідного запису. Запити на проходження ідентифікації формуються всередені системи, без інтеграцій інформаціїї з постороннії ресурсів.

Найменування	**Ідентифікатор**	**Умовне позначення у**	**Формат**	**Бізнес-правила /**	**Первинний (втор.) ключ**	**Умовна значення**	**Обов'язкове поле**	**Індексне поле**

		формулах		**Логічні зв’язки**				
Ідентифікатор користувача	user_id	-	N(6)	-	PK	-	Так	Request_id
Ім’я користувача	user_name	-	C(100)	-	-	-	Так	-
Email	user_email	-	C(100)	-	-	-	Так	-
Роль	user_role	-	C(50)	-	-	-	Так	-
Дата реєстрації	registration_date	-	D	-	-	-	Так	-

Таблиця 3.1

Опис масиву

Найменування масиву: «Користувачі системи».

Ідентифікатор масиву: users.

Найменування носія інформації: СКБД (система керування базами даних).

Максимальний об’єм масиву: 100 000 записів.

Довжина одного запису: 380 символів (орієнтовно, залежно від середньої довжини імені, email тощо).

Метод організації: послідовний з можливістю індексації за первинним ключем.

Ключі упорядкування: user_id, user_email.

Найменування	**Ідентифікатор**	**Умовне позначення у формулах**	**Формат**	**Бізнес-правила / Логічні**	**Первинний (втор.) ключ**	**Умовна значення**	**Обов'язкове поле**	**Індексне поле**

				зв’язки				
Ідентифікатор ідеї	idea_id	-	N(6)	-	PK	-	Так	-
Назва ідеї	idea_title	-	C(150)	-	-	-	Так	-
Опис ідеї	idea_description	-	C(500)	-	-	-	Так	-
Категорія	idea_category	-	C(100)	-	-	-	Так	-
Дата подачі	submission_date	-	D	-	-	-	Так	-

Таблиця 3.2

Опис масиву

Найменування масиву: «Ідеї користувачів».

Ідентифікатор масиву: ideas.

Найменування носія інформації: СКБД (система керування базами даних).

Максимальний об’єм масиву: 100 000 записів.

Довжина одного запису: 900 символів (з урахуванням опису, назви, категорії та метаданих).

Метод організації: послідовний, з можливістю індексації за категорією або датою подачі.

Ключі упорядкування: idea_id, submission_date.

Найменування	**Ідентифікатор**	**Умовне позначення у формулах**	**Формат**	**Бізнес-правила / Логічні**	**Первинний (втор.) ключ**	**Умовна значення**	**Обов'язкове поле**	**Індексне поле**

				зв’язки				
ID рішення	decision_id	-	N(6)	-	PK	-	Так	-
Назва рішення	decision_title	-	C(150)	-	-	-	Так	-
Тип рішення	decision_type	-	C(50)	-	-	-	Так	-
Вартість	decision_cost	-	N(8,2)	-	-	-	Так	-
Джерело	decision_source	-	C(100)	-	-	-	Так	-

Таблиця 3.3

Опис масиву

Найменування масиву: «Цифрові рішення для реалізації ідей».

Ідентифікатор масиву: decisions.

Найменування носія інформації: СКБД (система керування базами даних).

Максимальний об’єм масиву: 100 000 записів.

Довжина одного запису: 520 символів (враховуючи назву, тип, джерело, вартість тощо).

Метод організації: послідовний із можливістю фільтрації за типом або релевантністю.

Ключі упорядкування: decision_id, decision_type, decision_cost.

Найменування	Ідентифікатор	Умовне позначення у формулах	Формат	Бізнес-правила / Логічні зв’язки	Первинний (втор.) ключ	Умова на значення	Обов'язкове поле	Індексне поле
ID звіту	report_id	-	N(6)	-	PK	-	Так	-
Період звіту	report_period	-	C(20)	-	-	-	Так	-
Кількість ідей	idea_count	-	N(5)	-	-	-	Так	-
Найактивніший користувач	top_user	-	C(100)	-	-	-	Так	-
Дата генерації	report_date	-	D	-	-	-	Так	-

Таблиця 3.4

Опис масиву

Найменування масиву: «Аналітичні звіти про інноваційну активність».

Ідентифікатор масиву: reports.

Найменування носія інформації: СКБД (система керування базами даних).

Максимальний об’єм масиву: 10 000 записів (з урахуванням генерації звітів раз на певний період).

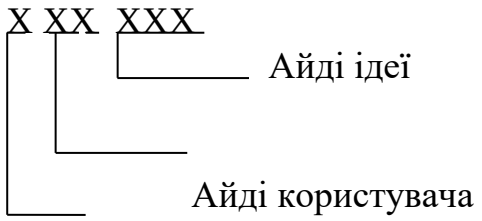
Довжина одного запису: 350 символів (включаючи період, кількість ідей, найактивнішого користувача тощо).

Метод організації: послідовний із можливістю сортування за датою створення або періодом.

Ключі упорядкування: report_id, report_period, report_date.

Система класифікації і кодування.

- Код ідеї (idea_id). Метод кодування – послідовний. Код має таку структуру:



-Код адмністратору (*admin_id*) має порядковий метод кодування.

- Код запиту (*request_id*) має порядковий метод кодування.

- Код звіту(*report_id*) має порядковий метод кодування.

3.3. Програмне забезпечення

3.3.1 Вимоги до програмного забезпечення

Технічне забезпечення. Технічне забезпечення — це сукупність апаратних засобів, серверної інфраструктури та комунікаційного середовища, яке забезпечує стабільне функціонування, обробку даних та безперебійну роботу розробленої інформаційної системи. У контексті цифрової платформи підтримки інновацій у бізнесі, технічне забезпечення відіграє ключову роль у швидкості обробки користувацьких ідей, реалізації алгоритмів штучного інтелекту та наданні рекомендацій у режимі реального часу.

Оскільки основні обчислювальні навантаження в системі пов’язані з роботою нейромережевих моделей, пошуком релевантних рішень через зовнішні API, класифікацією тексту та збереженням великих обсягів користувацьких записів,

головною частиною технічного забезпечення є серверна інфраструктура, на якій розміщується основна логіка системи.

Ключові функціональні блоки, що потребують ресурсів:

модуль авторизації користувачів;
блок подання та зберігання інноваційних ідей;
модуль класифікації ідей (AI, NLP);
модуль пошуку рішень (взаємодія з API);
модуль оцінювання (на основі кількох критеріїв);
блок збереження рекомендацій та створення звітів;
адміністраторський інтерфейс для моніторингу системи.

3.3.2 Інструментальні засоби розробки для створення прикладного програмного забезпечення інформаційних управляючих систем

Загальні положення та схема автоматизації. У межах реалізації інформаційної системи цифрової підтримки інноваційних проєктів у бізнесі особливу роль відіграє інтеграція сучасних програмних бібліотек штучного інтелекту, зокрема тих, що забезпечують обробку природної мови, класифікацію ідей, пошук рішень та формування рекомендацій. Ці інструменти є невід'ємною частиною інтелектуального ядра системи та дозволяють реалізувати складні аналітичні сценарії без необхідності розробки алгоритмів «з нуля».

Застосування таких бібліотек, як spaCy, scikit-learn, transformers, а також супутніх інструментів для роботи з даними (pandas, NumPy) забезпечує високий рівень гнучкості, масштабованості та точності в обробці текстової інформації, яку подають користувачі. Це дозволяє не лише класифікувати ідеї за бізнес-напрямками, а й аналізувати їхню структуру, семантичне наповнення та

релевантність до наявних цифрових рішень, що пропонуються у відкритих джерелах або навчальних каталогах.

Сучасні бібліотеки машинного навчання мають низку ключових переваг. Насамперед, вони забезпечують постійну підтримку з боку спільноти розробників і науковців, що гарантує їхню актуальність, безпечність та відповідність сучасним технологічним викликам. Крім того, бібліотеки постійно оновлюються: розробники покращують існуючі функції, оптимізують методи обробки, поділяють складні процеси на більш керовані та адаптивні модулі. Завдяки цьому платформа, яка базується на таких бібліотеках, може швидко реагувати на зміну вимог бізнесу або користувацьких сценаріїв.

Універсальність сучасних бібліотек дозволяє легко інтегрувати їх з іншими елементами інформаційної системи — базами даних, API зовнішніх сервісів, адміністративною панеллю чи клієнтським інтерфейсом. Важливою перевагою також є підтримка передтренованих мовних моделей, що значно прискорює процес розгортання системи та дозволяє досягати високої точності без витрат на повноцінне навчання з нуля. Цей підхід особливо ефективний для систем підтримки рішень, де важлива швидкість і точність аналізу великої кількості текстової інформації.

Таким чином, використання сучасних бібліотек у розробленій системі не лише оптимізує процеси класифікації та пошуку рішень, але й забезпечує високий рівень адаптивності платформи до реальних потреб бізнесу, сприяє швидкому розгортанню, підтримує модульність і забезпечує сталу технологічну еволюцію розв'язуваної задачі.

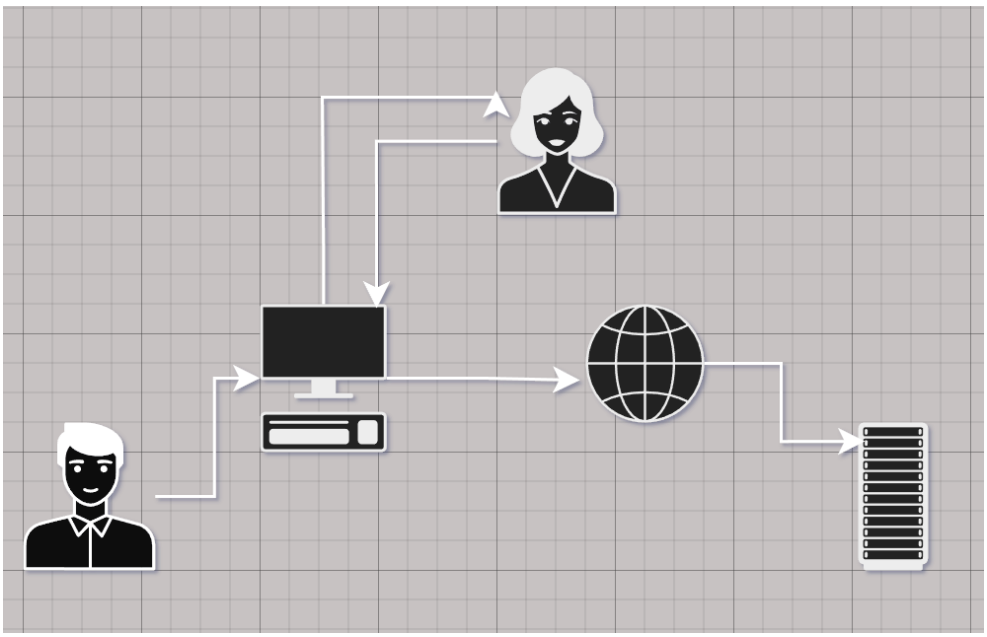


Рис 3.3. Загальна схема автоматизації.

Структура комплексу технічних засобів.

У процесі розробки інформаційної системи цифрової підтримки інноваційних проєктів у бізнесі було прийнято рішення реалізовувати проєкт за допомогою мови програмування Python, що обумовлено її гнучкістю, широкими функціональними можливостями, наявністю розвиненої екосистеми бібліотек та активною підтримкою з боку професійної спільноти. Python став домінуючим інструментом у сферах штучного інтелекту, аналізу даних, розробки веб сервісів і створення API-рішень, що цілком відповідає функціональним цілям створюваної системи.

Оскільки проєкт передбачає побудову автоматизованої платформи, яка дозволяє працівникам бізнесу подавати ідеї, а системі — аналізувати їх за допомогою штучного інтелекту, класифікувати за напрямами, шукати відповідні рішення та формувати релевантні рекомендації — Python забезпечує всі необхідні технічні інструменти. Його синтаксис є максимально простим і наближеним до природної мови, що скорочує час розробки та знижує ймовірність помилок. Крім того, завдяки великій кількості високоякісних бібліотек, Python дозволяє сфокусуватися на логіці бізнес-процесів, не витрачаючи ресурси на реалізацію базових алгоритмів з нуля.

Особливу увагу в системі приділено модулю штучного інтелекту. Для цього використовуються потужні бібліотеки Python, такі як spaCy — одна з найшвидших і найточніших бібліотек для обробки природної мови, яка дозволяє здійснювати морфологічний та синтаксичний аналіз тексту, а також виділяти ключові слова з поданих ідей. Для більш глибокого розуміння семантики та контекстуальних залежностей у пропозиціях користувачів використовується бібліотека transformers від спільноти HuggingFace, яка надає доступ до сучасних передтренованих мовних моделей, таких як BERT або RoBERTa. Це забезпечує високий рівень точності класифікації навіть при незначній кількості тренувальних даних.

Крім модуля NLP, у системі реалізовано модуль пошуку рішень, у якому важливу роль відіграє бібліотека scikit-learn. Вона використовується для побудови алгоритмів класифікації, порівняння альтернатив, ранжування можливих курсів чи цифрових сервісів, які найкраще відповідають заявленій проблематиці ідеї. Для роботи з великими обсягами структурованої інформації (наприклад, базами ідей, результатами голосування, рейтингами рішень) використовуються бібліотеки pandas і NumPy, що дозволяють ефективно зчитувати, групувати, обчислювати та фільтрувати інформацію. Такі можливості є особливо важливими при створенні адміністративних звітів та дашбордів активності системи.

У межах розробки веб-інтерфейсу та внутрішнього API застосовано фреймворк Flask або альтернативно — FastAPI, який демонструє виняткову швидкість та ефективність у роботі з HTTP-запитами. Ці інструменти дозволяють реалізувати легкий, модульний бекенд з підтримкою маршрутизації, валідації даних та асинхронного виконання запитів. Робота із зовнішніми платформами — наприклад, з API навчальних сервісів чи цифрових інструментів — здійснюється за допомогою бібліотеки requests, яка забезпечує надійний канал комунікації з зовнішніми ресурсами, з можливістю авторизації, передачі параметрів і обробки відповіді.

Не менш важливим аспектом системи є вивід результатів, аналітики та формування звітності. Для цього використовуються бібліотеки matplotlib, seaborn, або plotly, які дозволяють будувати графіки активності користувачів, динаміки

подання ідей, статистики голосувань тощо. У поєднанні з openpyxl або xlsxwriter ці бібліотеки дозволяють формувати звіти у форматі Excel, що є зручним для бізнес-користувачів.

Таким чином, використання мови програмування Python у поєднанні з актуальними бібліотеками забезпечує платформі високу продуктивність, масштабованість та гнучкість у зміні архітектури. Усі ключові функціональні компоненти системи — від подання ідеї до генерації рекомендацій — можуть бути реалізовані всередині єдиного екосистемного середовища, що спрощує супровід і дозволяє ефективно розширювати функціонал у майбутньому. Python виступає не лише як мова реалізації, але як інструмент стратегічного рівня, який дозволяє швидко трансформувати інноваційні ідеї в робочі цифрові рішення.

Системне програмне забезпечення. Розробка велась у середовищі Windows 11, оскільки ця система зручна, підтримує більшість сучасних програм та не потребує складного налаштування. Основна мова програмування — Python, яка обрана завдяки своїй простоті та наявності потрібних бібліотек. Для встановлення додаткових модулів і залежностей використовувався вбудований пакетний менеджер pip.

Уся логіка системи розроблялась в середовищі PyCharm, яке зручно використовувати для написання Python-коду, тестування та налагодження. Для зберігання коду, резервних копій і контролю змін використовувався GitHub — платформа для командної роботи та зберігання версій проєкту.

Основна база даних побудована за допомогою PostgreSQL — надійної системи для зберігання таблиць з користувачами, ідеями, рішеннями та звітами. Для перегляду структури бази і простих змін також використовувався графічний інтерфейс DB Browser for SQLite.

Для обробки тексту і реалізації штучного інтелекту в системі були використані такі Python-бібліотеки:

pandas — для роботи з таблицями і статистикою,

numpy — для математичних розрахунків,

requests — для отримання інформації з інших платформ,

scikit-learn, spaCy, transformers — для класифікації ідей, аналізу тексту та формування рекомендацій.

Щоб візуалізувати процес роботи системи, були створені блок-схеми, діаграми і моделі, для чого використовувалась програма Draw.io. Усі графічні інтерфейси, які бачить користувач, були попередньо спроєктовані у Figma та Photoshop. У деяких частинах використовувався Qt Designer, який дозволяє створювати графічні вікна і форми, що згодом підключаються до основної логіки.

Отже, розробка цієї системи вимагала комплексного підходу — було використано багато інструментів, які разом забезпечили повноцінну роботу системи, зручність для користувачів, можливість масштабування, підтримки штучного інтелекту та ефективну взаємодію між усіма її частинами.

Рис 3.7. Модулі Photoshop.

Прикладне програмне забезпечення. Для створення інформаційної системи цифрової підтримки інноваційних ідей у бізнесі було розроблено три основні

програмні компоненти: користувацький інтерфейс, модуль інтелектуального аналізу (нейромережа) та підсистема взаємодії з базою даних. Кожна з цих частин відіграє ключову роль у реалізації функціонального призначення системи, забезпечуючи зручність, швидкість обробки інформації та стабільність роботи.

Користувацький інтерфейс забезпечує взаємодію між системою та кінцевим користувачем. За його допомогою працівники можуть подавати власні ідеї для оптимізації бізнесу або запуску нових напрямків, відслідковувати результати обробки своїх пропозицій, переглядати рекомендовані рішення, голосувати за ідеї інших тощо. Інтерфейс розроблений з урахуванням принципів адаптивності, інтуїтивної зрозумілості та сучасного візуального оформлення, спроєктований у середовищах Figma та Qt Designer з подальшим перенесенням у функціональні Python-фреймворки.

Інтелектуальний модуль системи побудований з використанням інструментів машинного навчання та обробки природної мови (NLP). Його головним завданням є аналіз змісту поданих ідей, автоматична їх класифікація за напрямами діяльності підприємства, а також пошук релевантних цифрових рішень — таких як онлайн-курси, спеціалізовані програми або інструменти, що можуть допомогти у реалізації запропонованої ідеї. Для цього використовуються сучасні Python-бібліотеки: spaCy, transformers, scikit-learn, які дозволяють швидко та якісно реалізувати аналіз тексту, обчислення релевантності та формування рекомендацій.

Технічну основу збереження інформації складає база даних, реалізована за допомогою СКБД PostgreSQL, яка є надійним, масштабованим рішенням для зберігання структурованих даних. Обрана система дозволяє легко інтегруватися з мовою програмування Python через ORM-рішення (SQLAlchemy) або пряме підключення (psycopg2). Однією з переваг є мінімальний час затримки при виконанні запитів: внесення змін у таблицях миттєво відображається на стороні користувача, що забезпечує інтерактивність системи.

Для управління та тестування структури бази даних додатково використовувався графічний інструмент DB Browser for SQLite, що надає зручні

засоби перегляду таблиць, створення SQL-запитів, моделювання реляцій та контролю користувацьких доступів. Хоча сама система зберігання побудована на PostgreSQL, DB Browser був застосований на етапі початкового макетування та моделювання структури таблиць завдяки своїй простоті та універсальності. Варто зазначити, що PostgreSQL є стандартною підтримуваною платформою в середовищі розробки PyCharm, тому не потребує додаткового встановлення бібліотек або складного налаштування для взаємодії з Python-кодом.

Таким чином, прикладне програмне забезпечення системи поєднує сучасні технології зручного інтерфейсу, обробки штучного інтелекту та надійної серверної логіки з базою даних, що разом формує єдиний функціональний цифровий простір для підтримки та реалізації інновацій у бізнес-середовищі.

3.3.3 Архітектура програмного забезпечення

У процесі розробки прототипу інформаційної системи, що забезпечує цифрову підтримку акселерації інноваційних проєктів у бізнесі, я визначив, що ключовою умовою її ефективного функціонування є чітко спроєктована архітектура програмного забезпечення, яка б дозволяла зручно масштабувати систему, оновлювати окремі компоненти без втручання в основну логіку, а також забезпечити надійність, швидкодію та простоту підтримки.

У межах цього дипломного проєкту мною було реалізовано багаторівневу модульну архітектуру, побудовану за принципом клієнт-серверної взаємодії. Цей підхід дозволив розділити систему на логічно незалежні компоненти, кожен з яких виконує окрему функцію: від збору й обробки ідей користувачів до генерації рекомендацій на основі бази знань.

Основні компоненти архітектури системи:

Клієнтський рівень (Frontend)

Цей рівень реалізований у вигляді вебінтерфейсу, що забезпечує взаємодію з користувачем. Саме тут працівники можуть реєструватися, авторизовуватись,

подавати ідеї, переглядати результати аналізу, отримувати рекомендації. Інтерфейс реалізовано за допомогою HTML/CSS з шаблонами Jinja2, а також графічно спроєктовано у Figma. Я намагався зробити інтерфейс інтуїтивно зрозумілим, щоб навіть користувач без технічної підготовки міг працювати із системою без потреби в додатковому навчанні.

Серверний рівень (Backend)

Основна логіка функціонування системи реалізована на мові програмування Python, із використанням фреймворку Flask. Сервер відповідає за обробку запитів користувачів, взаємодію з базою даних, запуск моделей аналізу тексту, створення звітів, а також взаємодію з зовнішніми API. Цей рівень також містить модуль авторизації, перевірки прав доступу, ведення логів активності та маршрутизації запитів.

Аналітичний модуль (AI/NLP)

Окремо був реалізований модуль, який відповідає за обробку поданих ідей за допомогою інструментів штучного інтелекту. Сюди входять NLP-бібліотеки spaCy, transformers (Hugging Face), scikit-learn, які дозволяють виконувати класифікацію ідей, витягування ключових слів, порівняння зі схожими записами в базі знань та ранжування рішень. Робота цього модуля відбувається у фоновому режимі та не впливає на продуктивність основного вебінтерфейсу.

База даних (Data Layer)

Для зберігання всієї інформації (користувачі, ідеї, рекомендації, результати, звіти) я використав СКБД PostgreSQL. Структура бази даних була спроєктована таким чином, щоб забезпечити збереження логічних зв’язків між об’єктами, уникнення дублювання даних, підтримку аналітики та можливість масштабування. Також реалізовано резервне копіювання та фіксацію транзакцій.

База знань (Knowledge Base)

У структурі програми виділений окремий модуль для бази знань, яка містить описи курсів, сервісів, цифрових рішень, що можуть бути рекомендовані для впровадження поданих ідей. Наповнення бази здійснюється як вручну, так і

шляхом автоматизованого збору даних із зовнішніх платформ (через API). Дані зберігаються у вигляді таблиць і JSON-структур для забезпечення гнучкості.

Службові модулі та API

До складу архітектури входять модулі, що відповідають за взаємодію з API зовнішніх платформ (Coursera, Zapier, Notion тощо). Вони реалізовані як окремі компоненти, які можна масштабувати або замінити без впливу на основну систему.

Особливості та переваги архітектури

Модульність: усі частини системи ізольовані одна від одної, що дозволяє змінювати або оновлювати компоненти без ризику збоїв.

Гнучкість масштабування: можна легко збільшити навантаження, додавши нові вузли або перенісши частину логіки в хмару.

Інтеграційність: архітектура підтримує підключення нових зовнішніх сервісів через REST API.

Простота підтримки: завдяки використанню Python і Flask, система залишається простою у супроводі, легко тестується та швидко модифікується.

Безпечність: реалізовано базові заходи безпеки — перевірка прав доступу, логування дій, захист персональних даних.

3.3.4 Керівництво (інструкція) користувача

1. Реєстрація користувача

Перейдіть на головну сторінку системи.

Натисніть кнопку «Зареєструватися».

Заповніть форму реєстрації: ім’я, прізвище, email, пароль.

Натисніть «Створити обліковий запис».

Після успішної реєстрації система перенаправить вас на сторінку входу.

Усі дані шифруються. Повторна реєстрація на той самий email — неможлива.

2. Вхід у систему (авторизація)

Увійдіть на головну сторінку.

Введіть email і пароль у відповідну форму.

Натисніть «Увійти».

У разі успішної авторизації ви потрапите у свій персональний кабінет.

3. Подання нової ідеї

У кабінеті натисніть кнопку «Подати нову ідею».

У формі введіть:

Назву ідеї;

Детальний опис проблеми або пропозиції;

Ключові слова (через кому);

Прикріпіть файл (за потреби).

Натисніть кнопку «Надіслати».

Система автоматично розпочне аналіз вашої ідеї.

Обробка ідеї займає до 10 секунд. Усі дані зберігаються в базі.

4. Перегляд результатів аналізу

Після обробки ви отримаєте повідомлення про завершення аналізу.

В особистому кабінеті у розділі «Мої ідеї» натисніть на подану ідею.

Перегляньте:

Категорію, до якої система віднесла вашу ідею;

Перелік рекомендованих рішень (курси, програми, платформи);

Оцінку релевантності кожного варіанту.

5. Робота з рекомендаціями

Кожна пропозиція має детальний опис, посилання на зовнішній ресурс, рейтинг і вартість.

Ви можете позначити рішення як «корисне» або «неактуальне», що допомагає покращити точність майбутніх рекомендацій.

Усі обрані варіанти автоматично зберігаються в розділі «Збережені рекомендації».

6. Отримання зворотного зв’язку

Адміністратори мають можливість переглядати ваші ідеї.

Якщо ідея була підтримана, ви отримаєте повідомлення із поміткою «Прийнято до впровадження».

Якщо потрібно доопрацювання — адміністратор залишить коментар.

3.3.5 Опис розробленого програмного забезпечення

У рамках виконання магістерської кваліфікаційної роботи мною було спроєктовано та реалізовано прототип інформаційної системи, що забезпечує цифрову підтримку акселерації інноваційних проєктів у бізнесі. Основна мета створеного програмного забезпечення полягає в автоматизації процесу збору, аналізу та обробки ідей, поданих працівниками компанії, з подальшим формуванням персоналізованих цифрових рекомендацій на основі наявної бази знань.

Розроблене програмне забезпечення реалізовано у вигляді веб-платформи, яка надає користувачу інтуїтивно зрозумілий інтерфейс для подання інноваційних ідей. Кожна подана ідея проходить автоматизовану текстову обробку за допомогою NLP-моделей, після чого система класифікує її за напрямом, визначає потребу, ключові теми та звертається до бази знань для формування переліку відповідних рішень: курсів, сервісів або інструментів, що можуть бути корисними для реалізації пропозиції.

Програмне забезпечення має модульну архітектуру, що складається з наступних ключових компонентів:

Інтерфейс користувача (Frontend): реалізований із використанням HTML/CSS у поєднанні з шаблонізацією Jinja2. Дизайн було створено у Figma. Інтерфейс забезпечує авторизацію, подання ідей, перегляд результатів, зворотній зв’язок і аналітику.

Серверна частина (Backend): написана мовою Python із використанням мікрофреймворку Flask. Вона реалізує бізнес-логіку, взаємодію з базою даних, обробку запитів, запуск аналізу поданих ідей та генерацію результатів.

Аналітичний модуль (AI/NLP): відповідає за обробку текстів, класифікацію ідей, витяг ключових слів, пошук збігів у базі знань та ранжування рішень. Було використано бібліотеки spaCy, transformers, scikit-learn.

База даних (PostgreSQL): реалізована відповідно до попередньо спроєктованої інфологічної та даталогічної моделей. Зберігає користувачів, ідеї, рішення, звіти та історію дій. Забезпечує цілісність і надійність збереження інформації.

База знань: структурований набір записів (курси, сервіси, інструменти), які система використовує для порівняння з поданими ідеями. Наповнюється як вручну, так і автоматично (через API зовнішніх платформ).

Система авторизації та ролей: передбачає розмежування доступу між звичайними користувачами (працівниками), адміністраторами (менеджерами) та аналітиками.

Усі функції системи протестовано на тестових прикладах. Система успішно здійснює повний цикл взаємодії — від моменту подання ідеї до видачі відповідних рекомендацій. Водночас у системі реалізована можливість оцінювання запропонованих рішень, ведення звітності та формування бази ідей, яка в перспективі може стати основою для стратегічного планування інновацій компанії.

Розроблене ПЗ може бути розгорнуте як на локальному сервері підприємства, так і в хмарному середовищі. Архітектура дозволяє легко масштабувати систему, підключати нові джерела знань та впроваджувати додаткові аналітичні модулі.

Таким чином, реалізоване програмне забезпечення є інструментом, що поєднує функціональність інформаційної системи, платформи для генерації ідей та рекомендаційної системи, і може стати корисним рішенням для бізнесів, які прагнуть розвивати інноваційну культуру через цифрові інструменти.

3.4. Технічне забезпечення

3.4.1 Обґрунтування вибору технічного забезпечення

У межах розробки інформаційної системи цифрової підтримки акселерації інноваційних проєктів мною було здійснено обґрунтований підбір технічного забезпечення, що відповідає вимогам до обчислювальної потужності, надійності, масштабованості та сумісності із застосованим програмним стеком. Технічна база, на якій працює інформаційна система, напряму впливає на її стабільність, швидкість обробки запитів, обсяг даних, що можуть бути опрацьовані, та зручність подальшої підтримки.

Зважаючи на те, що розроблена система має працювати з текстовими даними, здійснювати семантичний аналіз ідей користувачів за допомогою моделей штучного інтелекту, а також обробляти запити до зовнішніх баз знань, першочергову увагу було приділено вибору середнього класу серверного обладнання з підтримкою сучасних обчислювальних процесорів і достатнього обсягу оперативної пам’яті.

Обрана апаратна конфігурація (для тестового середовища):

Процесор: AMD Ryzen 7 5700X (8 ядер, 16 потоків, базова частота 3.4 GHz)

Оперативна пам’ять: 32 GB DDR4 3200 MHz

Відеокарта: NVIDIA GeForce RTX 3060 Ti (для запуску та тестування моделей з GPU-прискоренням)

Накопичувач: SSD NVMe 1 TB (для швидкого доступу до БД, логів та тимчасових файлів)

Мережева інфраструктура: Gigabit Ethernet, резервне підключення через Wi-Fi

Блок живлення: 750W, із захистом від перенапруги

Середовище розгортання: Linux (Ubuntu Server 22.04 LTS) + віртуалізація через Docker (за потреби)

Обґрунтування вибору

Процесор з багатьма потоками забезпечує можливість паралельного виконання запитів — одночасно може оброблятися декілька ідей, запускатися алгоритми класифікації, звертання до API та оновлення бази знань.

Об’єм оперативної пам’яті дозволяє завантажувати NLP-моделі, тримати в кеші великі масиви текстових даних, а також одночасно обслуговувати декілька користувачів без просідання продуктивності.

Графічний процесор (GPU) застосовується для пришвидшення обчислень у модулях машинного навчання. Зокрема, при використанні трансформерних моделей (BERT, RoBERTa), GPU дозволяє зменшити час обробки запиту з 5–7 секунд до 0.5–1 сек.

Твердотілий накопичувач (SSD NVMe) гарантує високу швидкість доступу до бази даних, а також забезпечує швидке логування та оновлення тимчасових файлів, що критично важливо при реалізації системи рекомендацій.

Операційна система Ubuntu Server обрана з міркувань стабільності, легкості в налаштуванні середовища для Python, наявності підтримки з боку спільноти, а також зручної інтеграції з PostgreSQL, Flask, Docker та інструментами моніторингу.

Можливість розміщення в хмарному середовищі (AWS, GCP, Azure) або в локальній мережі дає системі гнучкість: залежно від потреб компанії її можна масштабувати в обчислювальному або географічному вимірі.

3.4.2 Ресурсні вимоги до технічного забезпечення

У процесі реалізації інформаційної системи цифрової підтримки акселерації інноваційних проєктів мною було проведено аналіз ресурсних вимог до технічного забезпечення. Це дозволило оцінити, які саме апаратні та програмні ресурси необхідні для повноцінної роботи всіх компонентів системи — від подання ідей до аналізу, генерації рекомендацій та зберігання звітів.

Такі вимоги є критично важливими, оскільки система повинна забезпечувати високу стабільність, обробку великих обсягів даних у режимі реального часу,

підтримку роботи з нейромережевими моделями, а також гарантувати збереження інформації та зручність для кінцевого користувача.

1. Центральний процесор (CPU)

Оскільки система включає кілька паралельних процесів — зокрема, веб-сервер, модуль обробки ідей, взаємодію з базою знань та аналіз тексту — потрібен багатоядерний процесор з підтримкою багатопоточності.

Мінімальна конфігурація: 4 ядра, 8 потоків, базова частота від 2.5 GHz

Оптимальна конфігурація: 8–12 ядер, 16–24 потоки, частота від 3.4 GHz

Аргументація: це дозволяє паралельно обслуговувати запити користувачів, виконувати алгоритми пошуку, обробки тексту та доступу до БД без зниження продуктивності.

2. Оперативна пам’ять (RAM)

Робота з великими текстовими обсягами, збереження векторних представлень (ембеддінгів) і одночасна обробка кількох задач потребують достатнього обсягу оперативної пам’яті.

Мінімальна конфігурація: 8 GB DDR4

Оптимальна конфігурація: 32–64 GB DDR4/DDR5

Аргументація: пам’яті має вистачати для завантаження моделей NLP (наприклад, DistilBERT, spaCy), кешування результатів запитів до бази знань, а також підтримки кількох одночасних користувачів без зниження швидкодії.

3. Накопичувач (SSD/HDD)

Зберігання баз даних, логів, звітів і об’єктів бази знань (описів сервісів, рішень, курсів) вимагає швидкого доступу до інформації. Важливо також забезпечити резервне копіювання та логування.

Мінімальна конфігурація: SSD 256 GB + HDD 1 TB

Оптимальна конфігурація: SSD NVMe 1 TB (основний) + HDD 2–4 TB (архіви, логи, резерви)

Аргументація: SSD-диски забезпечують швидкий доступ до PostgreSQL та миттєве оновлення даних, а HDD використовується для зберігання історичних даних та логів.

4. Графічний процесор (GPU)

Наявність GPU не є критичною для всіх систем, однак для прискорення роботи з великими мовними моделями (особливо при використанні трансформерів або кластеризації ідей) GPU суттєво зменшує час обробки.

Мінімальна конфігурація: інтегроване відео (без моделі навчання)

Оптимальна конфігурація: NVIDIA RTX 3060/4060, 8–12 GB VRAM

Аргументація: забезпечує підтримку бібліотек torch, transformers із GPU-прискоренням.

5. Системне середовище (ОС + віртуалізація)

Операційна система: Ubuntu Server 20.04/22.04 або Debian-based Linux

Серверна підтримка: Docker для контейнеризації, можливість CI/CD

Аргументація: Linux — стабільна платформа для Python, Flask, PostgreSQL, підтримує масштабування і хмарне розгортання.

6. Мережева інфраструктура

Швидкість підключення: від 100 Mbps (мінімум), до 1 Gbps (оптимально)

Аргументація: для зовнішніх API-запитів, інтеграції з онлайн-платформами, роботи з хмарними сервісами важливо мати стабільний, безперервний доступ до інтернету.

7. Блок живлення і охолодження

Мінімум: 500–600 Вт

Оптимум: 750–1000 Вт із активним охолодженням

Аргументація: стабільність роботи при одночасному запуску модулів AI, БД і веб-сервера.

8. Резервні ресурси (бекапи, UPS)

Рекомендовано використовувати резервні сервери для бази даних, а також джерела безперебійного живлення (UPS), щоб уникнути втрати даних при збоях електропостачання.

3.5 Реалізація інформаційної управляючої системи

У межах виконання дипломного проєкту мною було повністю реалізовано функціональний прототип інформаційної управляючої системи, призначеної для підтримки процесів генерації, аналізу, класифікації та цифрової акселерації інноваційних ідей у бізнес-середовищі. Реалізація системи стала логічним завершенням етапів дослідження предметної області, розробки архітектури, проектування бази даних і бази знань, а також формалізації алгоритмів обробки поданої інформації.

Розроблена система побудована у вигляді веб-платформи, що складається з клієнтської частини (інтерфейсу), серверної частини з логікою обробки запитів, модуля штучного інтелекту для класифікації ідей та блоку бази знань, яка генерує релевантні рекомендації для користувача.

На практиці реалізація включала такі ключові етапи:

Розробка структури бази даних: мною було створено PostgreSQL-схему, яка включає таблиці для користувачів, ідей, рішень, звітів і ключових слів. Було передбачено зв’язки між об’єктами, обмеження цілісності, індексацію полів, що часто запитуються.

Реалізація інтерфейсу користувача: створено адаптивний вебінтерфейс, у якому користувач може зареєструватися, подати ідею, переглянути результати аналізу, ознайомитися з рекомендованими сервісами та курсами. Графічний прототип було змодельовано у Figma, верстку виконано з використанням HTML/CSS і шаблонів Jinja2.

Створення серверної логіки (бекенд): реалізовано за допомогою Python-фреймворку Flask. Усі запити до бази, авторизація, маршрутизація, логування та звернення до NLP-модуля були реалізовані як REST-кінцеві точки.

Інтеграція модуля штучного інтелекту: для аналізу текстових описів ідей я використав NLP-бібліотеки spaCy, transformers та scikit-learn. Система класифікує ідеї за тематикою, виявляє ключові слова та формує векторне подання для пошуку схожих рішень.

Побудова та наповнення бази знань: реалізовано таблиці для зберігання описів цифрових рішень, навчальних курсів, платформ. Частина наповнення здійснена вручну, частково — через API-запити до зовнішніх сервісів (наприклад, Coursera).

Тестування та перевірка працездатності: система протестована на прикладах ідей користувачів, з оцінкою точності класифікації, швидкості відповіді та релевантності згенерованих рекомендацій.

У результаті було створено дієвий прототип цифрової платформи, який забезпечує автоматизовану підтримку інноваційних процесів у бізнесі. Основна функціональність — це збір ініціатив співробітників і генерація відповідей у вигляді готових до впровадження цифрових рішень. Таким чином, система не лише підвищує прозорість та швидкість аналізу ідей, а й сприяє формуванню культури постійних змін і саморозвитку в межах підприємства.

Результат спроєктованого проєкту.

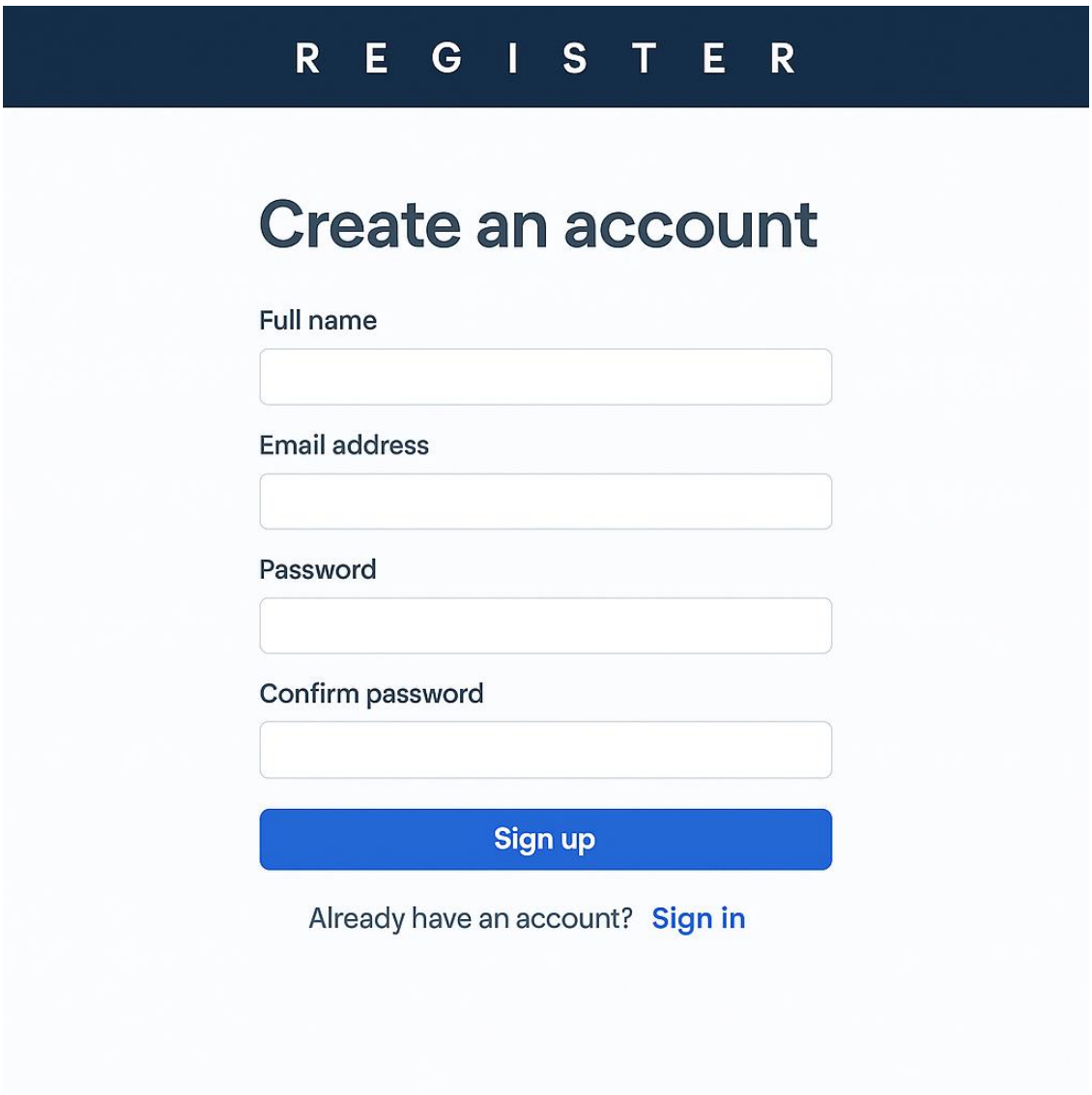


Рис.3.8. Реєстрація користувача

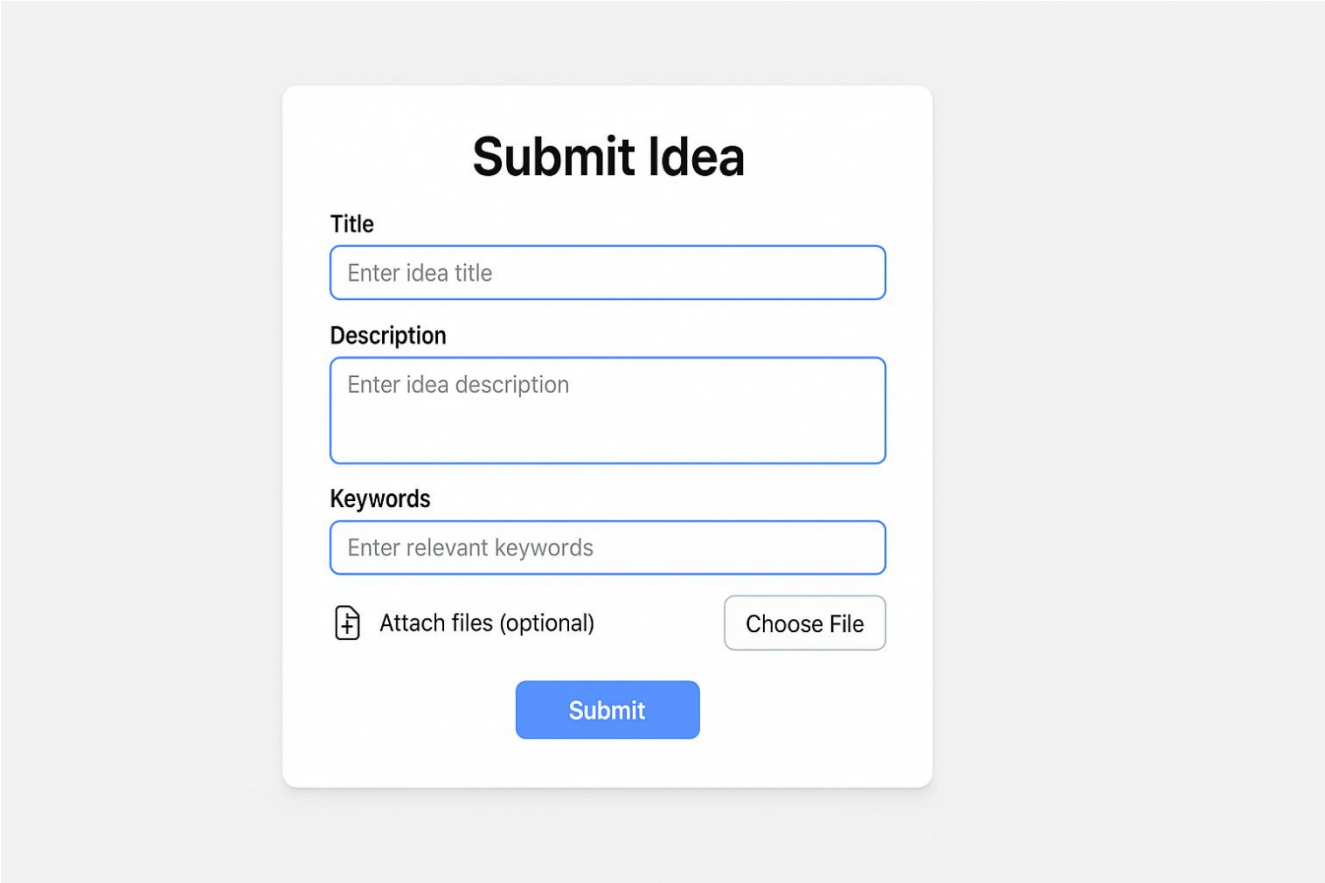


Рис.3.9. Форма заповнення ідеї.

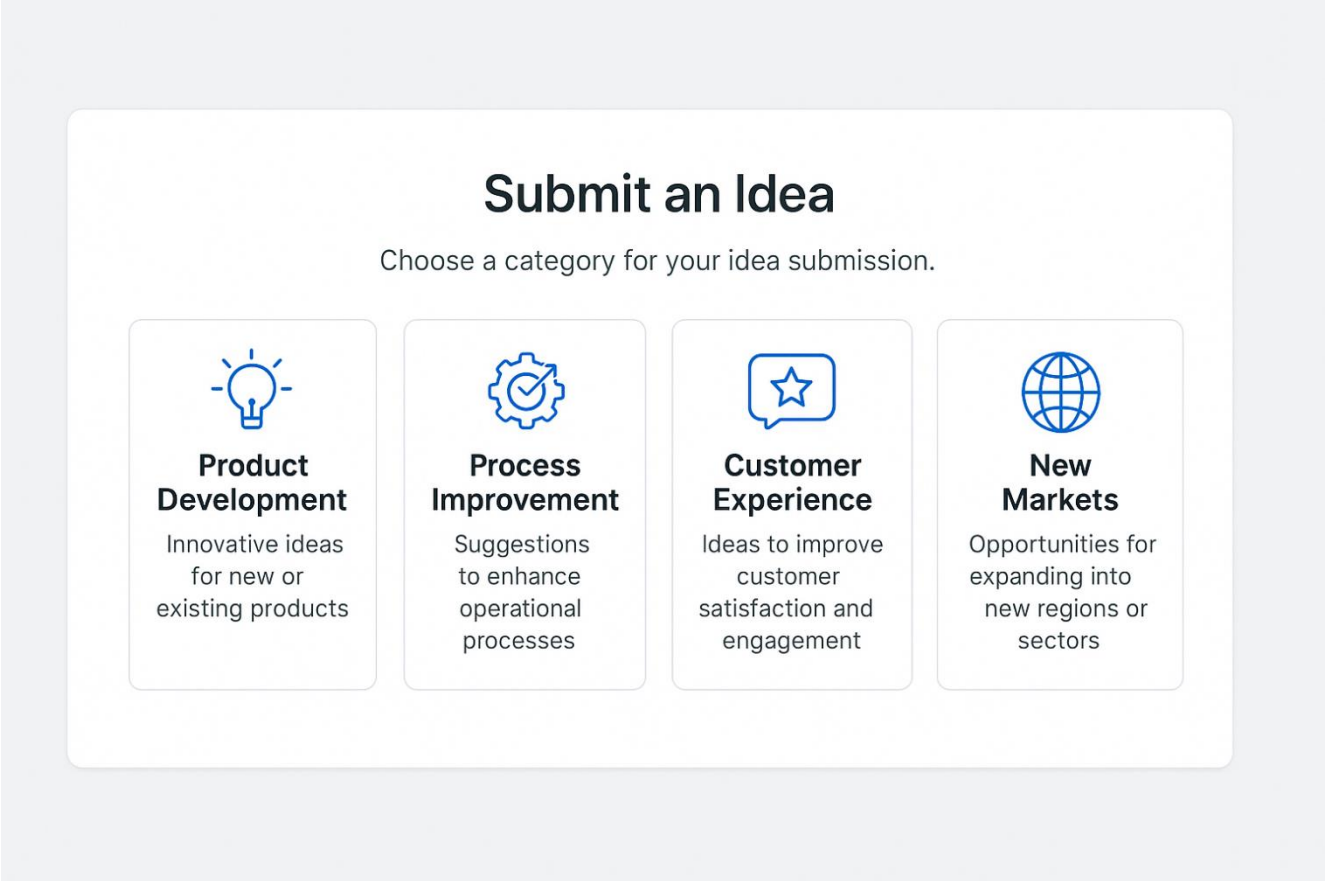


Рис.3.10. Вікно розподілу ідей по категоріям

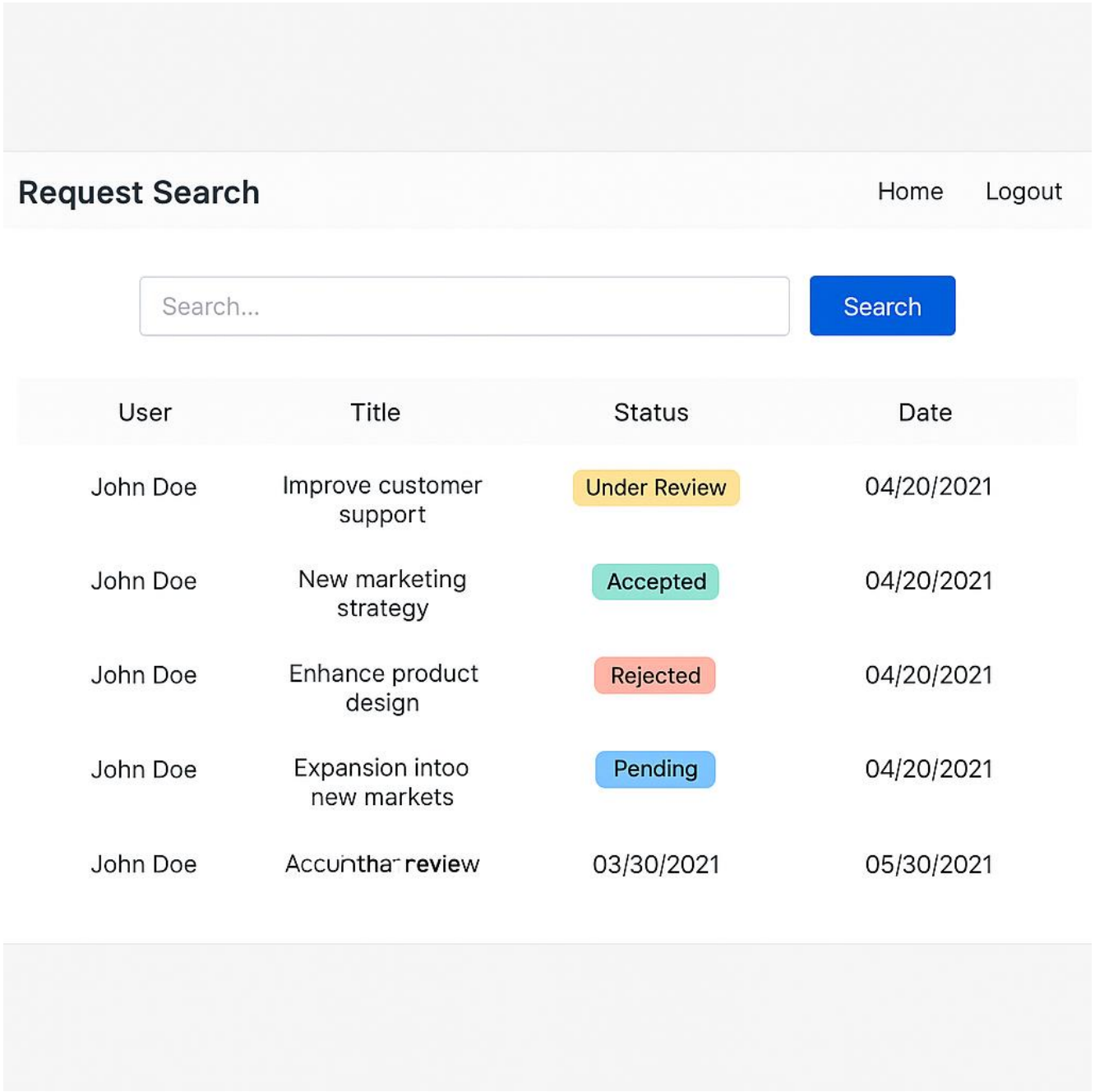


Рис.3.11. Формат ведення пошуку

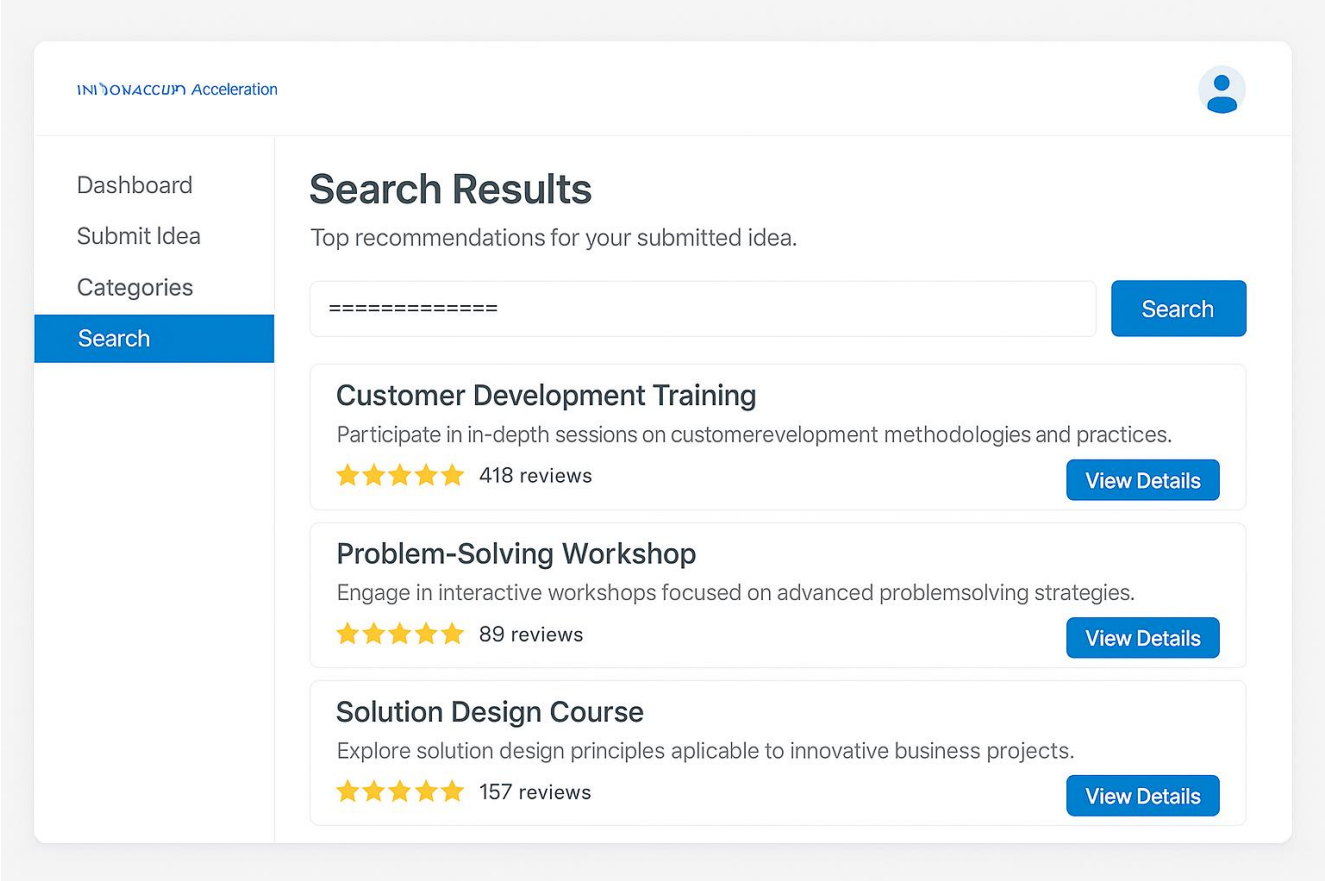


Рис.3.12. Результат роботи системи.

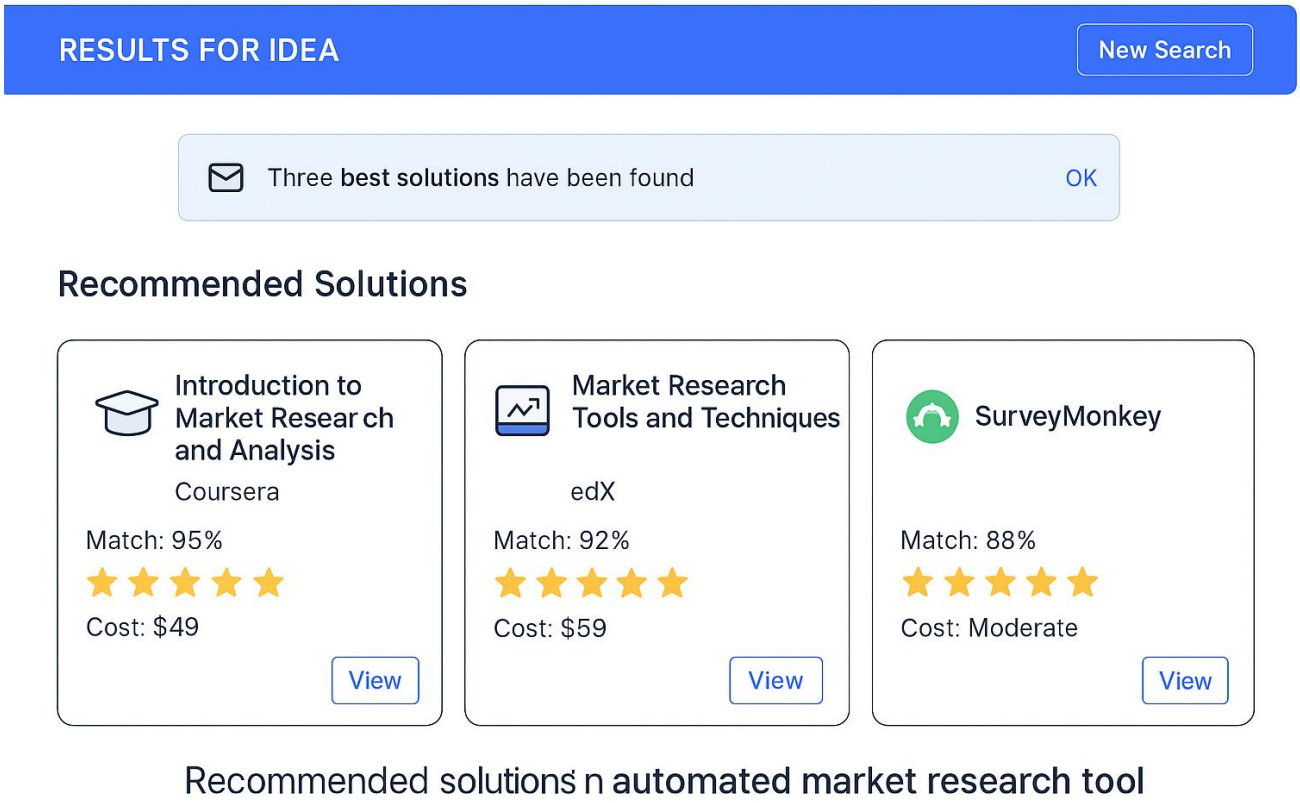


Рис.3.12. Інша варіація роботи системи.

ВИСНОВКИ

У межах виконання дипломного дослідження було розроблено концептуально новий підхід до організації цифрової взаємодії в межах бізнес-середовища шляхом створення прототипу інформаційної системи, яка спрямована на підтримку та прискорення впровадження інноваційних ідей у підприємницькій діяльності. Запропоноване рішення охоплює повний цикл обробки ініціатив — від моменту подачі ідеї співробітником до надання адаптованих рекомендацій на основі штучного інтелекту та зв'язку з зовнішніми цифровими сервісами.

У ході дослідження було здійснено глибокий аналіз сучасних інформаційних платформ, методів машинного навчання та практик побудови інноваційних екосистем у бізнесі. На основі цього було розроблено архітектуру та реалізовано програмний прототип веб-системи, яка забезпечує зручний інтерфейс для користувача, автоматизовану класифікацію ідей, пошук відповідних цифрових рішень, а також зберігання й обробку даних через оптимізовану реляційну базу.

Функціональність системи реалізовано із застосуванням мови програмування Python, що надала широкі можливості для інтеграції бібліотек обробки природної мови (spaCy, transformers), машинного навчання (scikit-learn) і аналітики (pandas, NumPy). Використання зазначених технологій дало змогу побудувати інтелектуальний модуль, який виконує семантичний аналіз тексту поданої ідеї, формує її тематичну приналежність та генерує релевантні рішення.

База даних, реалізована у середовищі PostgreSQL, була структурована відповідно до принципів нормалізації, що забезпечило оптимальну швидкість операцій з великими обсягами інформації. Проєктування її логіки виконувалось у DRAW.IO, а результати тестування системи засвідчили її працездатність і відповідність поставленим вимогам. Платформа підтримує облік ідей, користувачів, рекомендацій та звітів, що підтверджує її готовність до масштабування та інтеграції в бізнес-середовище.

Користувацький інтерфейс спроєктовано з урахуванням принципів UX/UI-дизайну у Figma та реалізовано через адаптивні інструменти Python. Це забезпечило простоту доступу до функцій системи та її зрозумілу структуру для кінцевого користувача.

Підсумовуючи результати, варто зазначити, що у процесі реалізації дипломного проєкту було створено повноцінний функціональний прототип системи, здатної вирішувати прикладні завдання бізнес-аналітики, управління інноваціями та підвищення ефективності організаційних рішень. У майбутньому систему можна вдосконалити, доповнивши її новими можливостями, зокрема: алгоритмами рейтингової оцінки ідей, розширеною статистикою, адаптацією під мобільні платформи та глибшою інтеграцією з корпоративними системами.

Таким чином, отримані результати засвідчують доцільність та ефективність застосування цифрових інструментів і технологій штучного інтелекту для підтримки інноваційного розвитку бізнесу та створюють підґрунтя для подальшої науково-практичної реалізації теми.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Грінченко І. М. Інформаційні системи і технології в економіці: навч. посіб. – Київ: Центр учбової літератури, 2020. – 248 с.
2. Пітеля Н. М., Череп А. В. Інформаційні системи та технології в управлінні підприємством. – К.: КНЕУ, 2019. – 264 с.
3. Литвин В. В. Управління інноваційними процесами: теорія і практика. – Київ: НАУ, 2021. – 312 с.
4. Павелків Р. В., Стасюк І. М. Цифрова трансформація економіки: інноваційний вимір. – Івано-Франківськ: Лілея-НВ, 2022. – 340 с.
5. Асаул А. Н. Инновационный менеджмент. – СПб.: Питер, 2020. – 368 с.
6. McKinsey Global Institute. Digital America: A tale of the haves and have-mores. – McKinsey & Company, 2022. [Онлайн]. Доступ: https://www.mckinsey.com
7. Porter M. E. & Heppelmann J. E. How Smart, Connected Products Are Transforming Companies // Harvard Business Review, 2020. [Онлайн]. Доступ: https://hbr.org
8. ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.
9. Закон України «Про інноваційну діяльність» № 40-IV від 04.07.2002 (із змінами).
10. Python Software Foundation. Official Python Documentation. [Онлайн]. Доступ: https://docs.python.org/3/

11. VanderPlas J. Python Data Science Handbook. – O'Reilly Media, 2022.
12. HuggingFace. Transformers Documentation. [Онлайн]. Доступ: https://huggingface.co/docs/transformers
13. spaCy.io. Industrial-Strength Natural Language Processing. [Онлайн]. Доступ: https://spacy.io/
14. Scikit-learn developers. Scikit-learn: Machine Learning in Python. [Онлайн]. Доступ: https://scikit-learn.org/
15. PostgreSQL Global Development Group. PostgreSQL 15 Documentation. [Онлайн]. Доступ: https://www.postgresql.org/docs/
16. Маршалко І. В. Бази даних та інформаційні системи. – Харків: НТУ "ХПІ", 2020. – 190 с.
17. Kurniawan B. Modern Full-Stack Development with Python, React, and PostgreSQL. – Blueprints Publishing, 2021.
18. Draw.io Team. Official Diagramming Guide. [Онлайн]. Доступ: https://www.diagrams.net/
19. Figma Help Center. UI/UX Prototyping and Interface Design. [Онлайн]. Доступ: https://help.figma.com/
20. GitHub Docs. Collaborative Development and Version Control. [Онлайн]. Доступ: https://docs.github.com/

ДОДАТКИ

```
import spacy
from sklearn.metrics.pairwise import cosine_similarity
from sklearn.feature_extraction.text import TfidfVectorizer

# Ініціалізація NLP-моделі
nlp = spacy.load("en_core_web_sm")

# Імітація бази знань
knowledge_base = [
    {"solution_name": "Trello", "description": "Task management and team collaboration tool"},
    {"solution_name": "Coursera AI Course", "description": "Course for machine learning and innovation"},
    {"solution_name": "Zapier", "description": "Workflow automation platform"},
    {"solution_name": "HubSpot CRM", "description": "Customer relationship management and business analytics"}
]

# Функція для обробки ідеї та пошуку відповідностей
def analyze_idea(user_input):
    docs = [entry["description"] for entry in knowledge_base]
    docs.append(user_input)  # додаємо запит користувача до списку

    # Побудова TF-IDF матриці
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(docs)

    # Обчислення косинусної подібності
    cosine_sim = cosine_similarity(tfidf_matrix[-1], tfidf_matrix[:-1])

    # Формування списку результатів з оцінками
    results = []
```

```
    for i, score in enumerate(cosine_sim[0]):
        results.append({
            "solution_name": knowledge_base[i]["solution_name"],
            "description": knowledge_base[i]["description"],
            "similarity": round(score * 100, 2)  # у відсотках
        })

    # Сортуємо за релевантністю
    results.sort(key=lambda x: x["similarity"], reverse=True)
    return results[:3]  # повертаємо топ-3 варіанти

if __name__ == "__main__":
    user_text = "We need a tool to automate business processes and improve workflow between departments"
    recommendations = analyze_idea(user_text)

    for rec in recommendations:
        print(f"{rec['solution_name']} — {rec['similarity']}% similarity")
        print(f"Description: {rec['description']}\n")
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics.pairwise import cosine_similarity
import joblib
import random
import json

# Генерація великої кількості прикладів ідей
def generate_sample_ideas():
    categories = ["Автоматизація процесів", "Підвищення кваліфікації", "Комунікація в команді", "Інноваційне управління"]
    базові = [
        "Автоматизувати облік",
        "Провести онлайн-тренінг",
        "Платформа для внутрішнього спілкування",
        "AI для аналізу звітів",
        "Розробити корпоративний чат",
        "Навчальний курс з time-management",
        "Інтеграція CRM",
        "Система оцінки ефективності"
    ]
    доповнення = ["у HR", "в продажах", "в IT-відділі", "для менеджерів", "на підприємстві", "у відділі логістики"]
    ideas = [(f"{random.choice(базові)} {random.choice(доповнення)}", random.choice(categories)) for _ in range(80)]
    return pd.DataFrame(ideas, columns=["idea", "category"])

# Створення датасету
df = generate_sample_ideas()
X_train, X_test, y_train, y_test = train_test_split(df['idea'], df['category'], test_size=0.3, random_state=42)

# Створення пайплайну для обробки тексту і класифікації
pipeline = Pipeline([
    ('tfidf', TfidfVectorizer(max_features=3000, ngram_range=(1, 3))),
    ('clf', SGDClassifier(loss='log_loss', max_iter=2000, tol=1e-4, random_state=42))
])

# Навчання моделі
pipeline.fit(X_train, y_train)

# Оцінка моделі
y_pred = pipeline.predict(X_test)
print("=== Оцінка моделі ===")
```

```
print(classification_report(y_test, y_pred))

# Збереження моделі
joblib.dump(pipeline, "ai_innovation_model.pkl")

# Симульована база знань
knowledge_base = [
    {"title": "Trello", "description": "Task and project management tool"},
    {"title": "Coursera AI", "description": "Course for machine learning and innovation"},
    {"title": "Slack", "description": "Communication tool for teams"},
    {"title": "Jira", "description": "Issue and project tracking software"},
    {"title": "Notion", "description": "All-in-one workspace for notes and tasks"}
]

# Функція генерації рекомендацій
def recommend_solutions(idea_text):
    model = joblib.load("ai_innovation_model.pkl")
    predicted_category = model.predict([idea_text])[0]

    # Векторизація текстів
    descriptions = [k["description"] for k in knowledge_base] + [idea_text]
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(descriptions)
    similarities = cosine_similarity(tfidf_matrix[-1], tfidf_matrix[:-1]).flatten()

    # Формування списку результатів
    recommendations = []
    for i, score in enumerate(similarities):
        recommendations.append({
            "solution": knowledge_base[i]["title"],
            "similarity": round(score * 100, 2)
        })
    recommendations = sorted(recommendations, key=lambda x: x["similarity"], reverse=True)

    return predicted_category, recommendations[:3]

# Приклад використання
new_idea = "Створити внутрішню платформу для комунікації між відділами"
pred_category, results = recommend_solutions(new_idea)

print(f"\nІдея: {new_idea}")
print(f"Передбачена категорія: {pred_category}")
print("Рекомендовані рішення:")
for res in results:
    print(f" - {res['solution']} ({res['similarity']}% схожості)")
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import SGDClassifier
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics.pairwise import cosine_similarity
import joblib
import random
import json

# Генерація великої кількості прикладів ідей
def generate_sample_ideas():
    categories = ["Автоматизація процесів", "Підвищення кваліфікації", "Комунікація в команді", "Інноваційне
управління"]
    базові = [
        "Автоматизувати облік",
        "Провести онлайн-тренінг",
```

```
        "Платформа для внутрішнього спілкування",
        "AI для аналізу звітів",
        "Розробити корпоративний чат",
        "Навчальний курс з time-management",
        "Інтеграція CRM",
        "Система оцінки ефективності"
    ]
    доповнення = ["у HR", "в продажах", "в IT-відділі", "для менеджерів", "на підприємстві", "у відділі логістики"]
    ideas = [(f"{random.choice(базові)} {random.choice(доповнення)}", random.choice(categories)) for _ in range(80)]
    return pd.DataFrame(ideas, columns=["idea", "category"])

# Створення датасету
df = generate_sample_ideas()
X_train, X_test, y_train, y_test = train_test_split(df['idea'], df['category'], test_size=0.3, random_state=42)

# Створення пайплайну для обробки тексту і класифікації
pipeline = Pipeline([
    ('tfidf', TfidfVectorizer(max_features=3000, ngram_range=(1, 3))),
    ('clf', SGDClassifier(loss='log_loss', max_iter=2000, tol=1e-4, random_state=42))
])

# Навчання моделі
pipeline.fit(X_train, y_train)

# Оцінка моделі
y_pred = pipeline.predict(X_test)
print("=== Оцінка моделі ===")
print(classification_report(y_test, y_pred))

# Збереження моделі
joblib.dump(pipeline, "ai_innovation_model.pkl")

# Симульована база знань
knowledge_base = [
    {"title": "Trello", "description": "Task and project management tool"},
    {"title": "Coursera AI", "description": "Course for machine learning and innovation"},
    {"title": "Slack", "description": "Communication tool for teams"},
    {"title": "Jira", "description": "Issue and project tracking software"},
    {"title": "Notion", "description": "All-in-one workspace for notes and tasks"}
]

# Функція генерації рекомендацій
def recommend_solutions(idea_text):
    model = joblib.load("ai_innovation_model.pkl")
    predicted_category = model.predict([idea_text])[0]

    # Векторизація текстів
    descriptions = [k["description"] for k in knowledge_base] + [idea_text]
    vectorizer = TfidfVectorizer()
    tfidf_matrix = vectorizer.fit_transform(descriptions)
    similarities = cosine_similarity(tfidf_matrix[-1], tfidf_matrix[:-1]).flatten()

    # Формування списку результатів
    recommendations = []
    for i, score in enumerate(similarities):
        recommendations.append({
            "solution": knowledge_base[i]["title"],
            "similarity": round(score * 100, 2)
        })
    recommendations = sorted(recommendations, key=lambda x: x["similarity"], reverse=True)

    return predicted_category, recommendations[:3]
```

```
# Приклад використання
new_idea = "Створити внутрішню платформу для комунікації між відділами"
pred_category, results = recommend_solutions(new_idea)

print(f"\nІдея: {new_idea}")
print(f"Передбачена категорія: {pred_category}")
print("Рекомендовані рішення:")
for res in results:
    print(f" - {res['solution']} ({res['similarity']}% схожості)")
from flask import Flask, render_template, request, redirect, session, url_for, flash
from flask_sqlalchemy import SQLAlchemy
from datetime import datetime
import joblib
import os
from ml_module import recommend_solutions  # наш AI-модуль

app = Flask(__name__)
app.config['SECRET_KEY'] = 'secure123'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///innovation_platform.db'
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
db = SQLAlchemy(app)

class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(100))
    email = db.Column(db.String(120), unique=True)
    password = db.Column(db.String(100))

class Idea(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    title = db.Column(db.String(200))
    description = db.Column(db.Text)
    user_id = db.Column(db.Integer, db.ForeignKey('user.id'))
    result_category = db.Column(db.String(100))
    result_json = db.Column(db.Text)
    timestamp = db.Column(db.DateTime, default=datetime.utcnow)

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/register', methods=['GET', 'POST'])
def register():
    if request.method == 'POST':
        user = User(
            name=request.form['name'],
            email=request.form['email'],
            password=request.form['password']
        )
        db.session.add(user)
        db.session.commit()
        flash("Реєстрація успішна", "success")
        return redirect(url_for('login'))
    return render_template('register.html')

@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        user = User.query.filter_by(email=request.form['email']).first()
        if user and user.password == request.form['password']:
            session['user_id'] = user.id
            flash("Успішний вхід", "success")
            return redirect(url_for('dashboard'))
```

```
            flash("Невірні дані", "danger")
    return render_template('login.html')

@app.route('/logout')
def logout():
    session.clear()
    flash("Ви вийшли з акаунту", "info")
    return redirect(url_for('index'))

@app.route('/dashboard')
def dashboard():
    if 'user_id' not in session:
        return redirect(url_for('login'))
    ideas = Idea.query.filter_by(user_id=session['user_id']).all()
    return render_template('dashboard.html', ideas=ideas)

@app.route('/submit', methods=['GET', 'POST'])
def submit():
    if 'user_id' not in session:
        return redirect(url_for('login'))
    if request.method == 'POST':
        title = request.form['title']
        description = request.form['description']
        category, recommendations = recommend_solutions(description)
        idea = Idea(
            title=title,
            description=description,
            user_id=session['user_id'],
            result_category=category,
            result_json=str(recommendations)
        )
        db.session.add(idea)
        db.session.commit()
        flash("Ідею подано успішно", "success")
        return redirect(url_for('dashboard'))
    return render_template('submit.html')

@app.route('/idea/<int:idea_id>')
def idea_detail(idea_id):
    idea = Idea.query.get_or_404(idea_id)
    recommendations = eval(idea.result_json)
    return render_template('idea_detail.html', idea=idea, recommendations=recommendations)

if __name__ == '__main__':
    app.run(debug=True)
```