

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

«До захисту допущено»
Завідувач кафедри
Едуард ЖАРИКОВ
«_____» _____ 2025 р.

Дипломний проєкт
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інженерія програмного забезпечення
інформаційних систем»
спеціальності «121 Інженерія програмного забезпечення»

на тему: Вебсервіс для пошуку партнерів з інтеграцією таск-менеджменту

Виконала студентка IV курсу, групи ІП-12

Доброхотова Катерина Олександрівна

(прізвище, ім'я, по батькові)

(підпис)

Керівник

доцент, к.т.н., доц., Лісовиченко Олег Іванович

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Рецензент

доц., к.т.н., доц. каф. ІСТ Ткач Михайло Мартинович

(посада, науковий ступінь, вчене звання, прізвище та ініціали)

(підпис)

Засвідчую, що у цій дипломній роботі
немає запозичень з праць інших
авторів без відповідних посилань.

Студент _____
(підпис)

Київ – 2025

Національний технічний університет України «Київський політехнічний інститут
імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 121 Інженерія програмного забезпечення Освітньо-професійна

програма – Інженерія програмного забезпечення інформаційних систем

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Едуард ЖАРІКОВ

_____ 2025 р.

ЗАВДАННЯ
на дипломний проєкт студенту

Доброхотовій Катерині Олександрівні

1. Тема проєкту Вебсервіс для пошуку партнерів з інтеграцією таск-менеджменту,
керівник роботи Лісовиченко Олег Іванович, к.т.н, доцент,

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 19 » травня 2025 р. № 1637-с

2. Термін подання студентом проєкту 11.06.2025

3. Вихідні дані до проєкту: технічне завдання _____

4. Зміст пояснювальної записки:

- 1) Аналіз предметної області
- 2) Обґрунтування вибору технологій
- 3) Проектування архітектури програмного забезпечення
- 4) Реалізація основного функціоналу платформи

- 5) Створення сторінок пошуку, профілів користувачів, списку проєктів, презентації стартапів, організація навігації та взаємозв'язків між ними.
- 6) Аналіз безпеки та перспективи розширення
- 7) Розгляд питань захисту персональних даних, можливостей інтеграції авторизації, модерації контенту, впровадження аналітики та SEO.

5. Перелік графічного матеріалу:

- Діаграма станів
- Діаграма послідовності
- Діаграма компонентів
- Діаграма патерна Mediator

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «15» березня 2025 року _____

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів роботи	Примітка
1	Вивчення рекомендованої літератури	16.04.2025	
2	Аналіз існуючих методів розв'язання	20.04.2025	

	задачі		
3	Постановка та формалізація задачі	22.04.2025	
4	Розробка інформаційного забезпечення	23.04.2025	
5	Алгоритмізація задачі	29.04.2025	
6	Обґрунтування вибору використаних технічних засобів	02.05.2025	
7	Розробка програмного забезпечення	10.05.2025	
8	Налагодження програми	15.05.2025	
9	Виконання графічних документів	16.05.2025	
10	Оформлення пояснювальної записки	31.05.2025	
11	Подання ДП на попередній захист	02.06.2025	
12	Подання ДП рецензенту	10.06.2025	
13	Подання ДП на основний захист	16.06.2025	

Студентка

(підпис)

Катерина ДОБРОХОТОВА

(ініціали, прізвище)

Керівник роботи

(підпис)

Олег ЛІСОВИЧЕНКО

(ініціали, прізвище)

АНОТАЦІЯ

Доброхотова К.О. Вебплатформа для пошуку бізнес-партнерів "Бізнес Матч".
КПІ ім. Ігоря Сікорського, Київ, 2025.

Проект містить 205 сторінок тексту, 10 рисунків, 5 таблиць, 4 UML-діаграми, 1 ER-діаграму, посилання на 26 джерел, 3 додатки.

Ключові слова: бізнес-платформа, пошук партнерів, GitHub Pages, HTML, CSS, TypeScript, UX/UI, авторизація, інтерактивність, React, масштабування.

Об'єктом дослідження є процес розробки платформи для налагодження ділових зв'язків між підприємцями, інвесторами та стартапами.

Мета проекту: розробити вебзастосунок, що дозволяє здійснювати пошук бізнес-партнерів, перегляд анкет та стартапів, організовувати ділову взаємодію через інтуїтивний вебінтерфейс.

У дипломному проекті реалізовано клієнтську частину застосунку на основі HTML, CSS та TypeScript із використанням компонентного підходу та подальшою можливістю інтеграції React. Платформу розгорнуто на GitHub Pages, що дало змогу швидко створити MVP. Передбачено базову структуру для реалізації реєстрації, авторизації, пошуку за фільтрами та динамічного перегляду даних.

У проекті також розглянуто перспективу впровадження серверної частини (наприклад, за допомогою Node.js або Firebase), системи аналітики, захисту персональних даних, адаптивного дизайну та SEO-оптимізації.

Результати тестування прототипу показали стабільність роботи, відповідність логіки користувацькому сценарію, готовність до інтеграції додаткового функціоналу та позитивну оцінку з боку тестової цільової аудиторії.

Платформа може бути використана як основа для запуску стартапу, спрямованого на підтримку ділового нетворкінгу серед підприємців, представників малого та середнього бізнесу, інвесторів і фахівців у галузі інновацій.

КЛЮЧОВІ СЛОВА: JWT, KAFKA, UUPS PROXY, ВЕРИФІКАЦІЯ ДОКУМЕНТІВ,
ВЕБ-ДОДАТОК, NODEJS, REACT, WEB3J, POSTGRESQL

ABSTRACT

Dobrokhotova K.O. *Web Platform for Business Partner Matching “Business Match”*.
Igor Sikorsky Kyiv Polytechnic Institute, Kyiv, 2025.

The project includes 64 pages of text, 10 figures, 5 tables, 4 UML diagrams, 1 ER diagram, references to 26 sources, and 3 appendices.

Keywords: business networking, partner search, GitHub Pages, HTML, CSS, TypeScript, UX/UI, web interface, React, scalability.

The object of this development is the creation of a web-based platform aimed at facilitating business networking and collaboration between entrepreneurs, investors, and startup teams.

The goal of the project: to develop a web application that allows you to search for business partners, view profiles and startups, and organize business interaction through an intuitive web interface.

The platform is developed using HTML, CSS, and TypeScript and is currently deployed on GitHub Pages as a minimum viable product (MVP).

Its structure is designed for easy integration of dynamic features, including user authentication, profile creation, and intelligent search filtering.

The project architecture supports a transition to component-based development using frameworks such as React.

Further enhancements may include integration with backend services (e.g., Node.js or Firebase), secure data handling, responsive design, and SEO optimization.

Testing of the MVP demonstrated consistent performance, logical user flow, and positive feedback from the target audience.

KEYWORDS: JWT, KAFKA, UUPS PROXY, DOCUMENT VERIFICATION, WEB APPLICATION, NODEJS, REACT, WEB3J, POSTGRESQL

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

Технічне завдання
КПІ.ІП-1208.045440.01.91.

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

Київ 2025

ЗМІСТ

1 НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ.....	10
2 ПІДСТАВА ДЛЯ РОЗРОБКИ.....	11
3 ПРИЗНАЧЕННЯ РОЗРОБКИ.....	12
4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	13
5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ.....	15
6 СТАДІЇ І ЕТАПИ РОЗРОБКИ.....	16
7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ.....	17

1. НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Найменування:

Вебсервіс "TaskPartner" для пошуку партнерів з інтеграцією систем таск-менеджменту. "TaskPartner" — це вебплатформа, що поєднує функціонал пошуку партнерів для спільної роботи над завданнями з інструментами управління проектами. Сервіс орієнтований на користувачів, які потребують швидкої та ефективної кооперації з іншими фахівцями або студентами для реалізації спільних цілей.

Галузь застосування:

- **Фріланс та стартапи** — організація команд для короткострокових проектів або MVP-розробок із подальшою масштабованістю.
- **Освіта** — підтримка студентів і викладачів у виконанні групових завдань, курсових та дипломних проектів.
- **Командна робота у МСБ** — забезпечення прозорості взаємодії в межах малих і середніх бізнес-команд із можливістю підключення таск-менеджмент систем.
- **Проектний менеджмент** — як інструмент для формування команд, постановки задач, контролю дедлайнів та відстеження прогресу.

2. ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки вебсервісу для створення відеопривітань є завдання на дипломне проектування, затверджене кафедрою інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

3. ПРИЗНАЧЕННЯ РОЗРОБКИ

Основні цілі:

- Надання зручної платформи для пошуку учасників за професійною спеціалізацією, досвідом, інтересами та іншими критеріями.
- Автоматизація створення завдань у таких популярних таск-менеджерах, як **Trello**, **Asana** та **Jira** після формування команди.
- Забезпечення централізованої комунікації та обміну інформацією між учасниками команд.

4. ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1.Вимоги до функціональних характеристик

4.1.1 Користувацький інтерфейс:

- Адаптивний дизайн (десктоп, мобільні пристрої).
- Інтуїтивне меню: головна сторінка, пошук, профіль, повідомлення.

4.1.2 Облік користувачів:

- Реєстрація через email/соцмережі (Google, GitHub).
- Ролі: звичайний користувач, адміністратор.
- Редагування профілю (навички, досвід, інтереси).

4.2 Вимоги до надійності

- Захист даних (HTTPS, шифрування паролів).
- Резервне копіювання бази даних щоденно.

4.3 Умови експлуатації

4.3.1 Вид обслуговування:

Онлайн-доступ через браузер.

4.3.2 Обслуговуючий персонал:

- 1 адміністратор для модерації контенту.

4.4 Вимоги до складу і параметрів технічних засобів

- **Сервер:** CPU 2+ ядра, RAM 4+ GB, SSD 50+ GB.
- **Клієнт:** Браузери Chrome, Firefox, Safari останніх версій.

4.5 Вимоги до інформаційної та програмної сумісності

4.5.1 Вхідні дані:

- Дані профілю (текст, фото).

- API-запити до Trello/Asana.

4.5.2 Вихідні дані:

- Список партнерів у форматі JSON.
- Експорт завдань у підключений таск-менеджер.

4.6 Мова розробки:

- Backend: JavaScript (Node.js) / Python.
- Frontend: TypeScript (React).

4.7 Середовище розробки:

VS Code, Docker, PostgreSQL.

4.8 Вихідні коди:

Документовані коментарями. Ліцензія: MIT.

5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

5.1 Попередній склад програмної документації

У склад супроводжувальної документації повинні входити наступні документи на аркушах формату А4:

- пояснювальна записка;
- технічне завдання;
- текст програми;
- програма та методика тестування;
- керівництво користувача.

Графічна частина повинна бути виконана на аркушах формату А3 та містити наступні документи:

- схема структурна варіантів використання;
- архітектура програмного забезпечення;
- креслення вигляду екранних форм.

5.2 Спеціальні вимоги до програмної документації

Програмні модулі, котрі розробляються, повинні бути задокументовані, тобто тексти програм повинні містити всі необхідні коментарі.

6 СТАДІЇ І ЕТАПИ РОЗРОБКИ

№	Назва етапу	Строк	Звітність
1.	Вивчення літератури за тематикою проєкту	15.03	
2.	Розробка технічного завдання	19.04	Технічне завдання
3.	Аналіз вимог та уточнення специфікацій	21.04	Специфікації програмного забезпечення
4.	Проектування структури програмного забезпечення, проектування компонентів	22.04	Схема структурна програмного забезпечення та специфікація компонентів (діаграма класів, схема алгоритму)
5.	Програмна реалізація програмного забезпечення	09.05	Тексти програмного забезпечення
6.	Тестування програмного забезпечення	14.05	Тести, результати тестування
7.	Розробка матеріалів текстової частини проєкту	20.05	Пояснювальна записка
8.	Розробка матеріалів графічної частини проєкту	22.05	Графічний матеріал проєкту
9.	Оформлення технічної документації проєкту	31.05	Технічна документація

7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ

Тестування розробленого програмного продукту виконується відповідно до “Програми та методики тестування”.

Пояснювальна записка

до дипломної роботи

на тему: **Вебсервіс для пошуку партнерів з інтеграцією
таск-менеджменту**

КПІ.ІП-1208.045440.02.81

Київ – 2025

ЗМІСТ

ВСТУП.....	23
Актуальність теми роботи.....	23
Мета та завдання дослідження.....	23
Об'єкт і предмет дослідження.....	24
Методи дослідження.....	24
Структура роботи.....	25
1. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ У СФЕРІ ПРОФЕСІЙНОГО НЕТВОРКІНГУ ТА TASK-МЕНЕДЖМЕНТУ.....	27
1.1. Огляд популярних вебсервісів для пошуку партнерів (LinkedIn, Upwork, AngellList, тощо).....	27
1.2. Огляд сучасних рішень в області таск-менеджменту.....	28
1.3. Інтеграція пошуку партнерів з таск-менеджментом.....	30
1.4. Переваги та недоліки існуючих рішень.....	32
Висновки до розділу.....	33
2. АНАЛІЗ ВИМОГ ДО ВЕБЗАСТОСУНКУ ДЛЯ ПОШУКУ ПАРТНЕРІВ І УПРАВЛІННЯ ЗАВДАННЯМИ.....	34
2.1. Функціональні вимоги до вебсервісу.....	34
2.2. Розроблення нефункціональних вимог.....	38
2.3. Специфікація інтерфейсу користувача.....	40
2.4. Визначення цільової аудиторії.....	43
2.5. Визначення варіантів використання та користувацьких сценаріїв.....	44
Висновки до розділу.....	48
3. ПРОЦЕС РОЗРОБКИ ВЕБЗАСТОСУНКУ.....	49
3.1. Вступ до процесу розробки вебзастосунку.....	49
3.2. Формування вимог.....	49
3.3. Проєктування архітектури.....	49
3.4. Вибір технологій.....	51
3.5. Розробка MVP.....	51
3.6. Тестування.....	52
3.7. Оптимізація та безпека.....	52
3.8. Підготовка до розгортання.....	53
3.9. Планування та аналіз вимог.....	54
3.10. Архітектурне проєктування.....	54
3.11. Реалізація функціональності.....	54
3.12. Забезпечення безпеки.....	55
3.13. Тестування.....	55

3.14. Оптимізація та підготовка до розгортання.....	55
3.15. Основні принципи, які лягли в основу розробки:.....	55
3.16. Вибір архітектури вебзастосунку.....	56
3.17. Технології та інструменти для розробки.....	59
3.18. Процес розробки та тестування.....	60
3.19. Безпека та конфіденційність даних.....	61
Висновки до розділу.....	61
4. ІМПЛЕМЕНТАЦІЯ ФУНКЦІОНАЛЬНОСТІ ПОШУКУ ПАРТНЕРІВ ТА ПІДБОРУ ФАХІВЦІВ.....	62
4.1. Загальна характеристика функціональності пошуку партнерів.....	62
4.2. Алгоритми для пошуку партнерів і фахівців.....	63
4.3. Інтерфейс пошуку та фільтрація результатів.....	66
4.4. Система рекомендацій для підбору партнерів.....	68
4.5. Інтеграція пошуку партнерів з профілями користувачів.....	71
Висновки до розділу.....	75
5. РОЗРОБКА ТА ВПРОВАДЖЕННЯ ІНСТРУМЕНТІВ ДЛЯ УПРАВЛІННЯ ЗАВДАННЯМИ.....	76
5.1. Вступ до реалізації таск-менеджменту.....	76
5.2. Основи таск-менеджменту.....	76
5.3. Розробка інтерфейсу для управління завданнями.....	78
5.4. Інтеграція завдань із профілем користувача та проектом.....	80
5.5. Визначення ролей та прав користувачів.....	82
5.6. Рішення для спільної роботи та обміну документами.....	84
Висновки до розділу.....	86
6. ІНТЕЛЕКТУАЛЬНА СИСТЕМА РЕКОМЕНДАЦІЙ ДЛЯ ПІДБОРУ ПАРТНЕРІВ І ВИКОНАВЦІВ.....	87
6.1. Розробка системи рекомендацій.....	88
6.2. Алгоритми для системи рекомендацій.....	88
6.3. Моделі машинного навчання для оптимізації підбору партнерів.....	89
6.4. Персоналізація рекомендацій на основі історії взаємодії.....	91
6.5. Врахування зворотного зв'язку від користувачів.....	93
Висновки до розділу.....	93
7. ОЦІНКА ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ ВЕБЗАСТОСУНКУ.....	94
7.1. Вступ.....	94
7.2. Методи оцінки ефективності вебзастосунку.....	95
7.3. Оцінка зручності та функціональності через опитування користувачів.....	98
7.4. Оцінка впливу на ефективність роботи команд і партнерів.....	100
7.5. Порівняння результатів до та після використання вебсервісу.....	101
Висновки до розділу.....	102
ВИСНОВКИ.....	103

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	104
ДОДАТКИ.....	105

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IDE - Integrated Development Environment – інтегроване середовище розробки

API - Application Programming Interface – прикладний програмний інтерфейс

SDK - Software Development Kit – набір інструментів розробника

IT - Information Technology – інформаційні технології

ER - Entity-Relationship Diagram – діаграма зв'язків між сутностями

ОС - Операційна система

БД - База даних

JWT - JSON Web Token – формат токена для аутентифікації

UUPS - Universal Upgradeable Proxy Standard – проксі-патерн для оновлюваних контрактів

EVM - Ethereum Virtual Machine – віртуальна машина Ethereum

VS Code - Visual Studio Code – текстовий редактор та IDE від Microsoft

DoS - Denial of Service – атака на відмову в обслуговуванні

MITM - Man-in-the-Middle – атака "людина посередині"

Kafka - Apache Kafka – система обміну повідомленнями

HTTP - Hypertext Transfer Protocol – протокол передачі гіпертексту

ICMP - Internet Control Message Protocol – протокол перевірки доступності

Multisig - Multi-signature wallet – гаманець з мультипідписом

DAO - Decentralized Autonomous Organization – децентралізована автономна організація

UUID - Universally Unique Identifier – універсальний унікальний ідентифікатор

ВСТУП

Актуальність теми роботи

У сучасному світі цифрова трансформація стала одним із основних чинників, що впливають на розвиток бізнесу та формування нових моделей співпраці. Сьогодні пошук надійних партнерів, клієнтів або спеціалістів не обмежується географічними рамками. Онлайн-платформи, такі як **LinkedIn**, **AngellList**, **Upwork** та інші, дозволяють знаходити потрібних фахівців у будь-якому куточку світу.

Метою дослідження є підвищення ефективності командної взаємодії через автоматизований підбір партнерів та єдину платформу для спільної роботи. Вона буде об'єднувати можливості пошуку партнерів для професійної співпраці з інструментами управління завданнями та проєктами.

Для досягнення поставленої мети необхідно виконати низку **конкретних дослідницьких та практичних завдань**, серед яких:

1. **Проаналізувати існуючі рішення** у сфері професійного нетворкінгу (наприклад, LinkedIn, AngellList) та таск-менеджменту (Trello, Asana, Jira) з метою виявлення їх функціональних особливостей, переваг і недоліків, а також встановлення причин відсутності комплексного підходу до інтеграції цих напрямів.
2. **Сформулювати концептуальну модель вебсервісу**, яка дозволяє об'єднати пошук партнерів і систему управління проєктами в єдиному інтерфейсі, враховуючи особливості взаємодії користувачів і їх ролі в спільній роботі.
3. **Визначити функціональні вимоги** до вебсервісу (ключові модулі, основні сценарії використання, логіка підбору партнерів, створення та контроль завдань), а також **нефункціональні вимоги**, що охоплюють продуктивність, масштабованість, безпеку, зручність інтерфейсу тощо.
4. **Розробити архітектуру вебзастосунку**, визначити основні компоненти системи, засоби їх взаємодії, обрати відповідний технологічний стек для реалізації проєкту.

5. **Інтерфейсна реалізація:** створити прототип та реалізувати інтерфейс користувача з урахуванням принципів UX/UI дизайну для забезпечення інтуїтивної та зручної навігації.

Об'єкт і предмет дослідження

Об'єктом дослідження є процес розробки вебсервісу для пошуку партнерів і управління завданнями, що охоплює повний життєвий цикл створення вебзастосунку: від формування вимог та вибору технологічного стеку до розробки архітектури, реалізації функціоналу, інтеграції окремих модулів та проведення тестування. **Предметом дослідження** виступає розробка та інтеграція функціональних модулів для пошуку партнерів і управління завданнями в єдиному вебзастосунку. Зокрема, досліджуються алгоритми підбору партнерів, методи рекомендацій на основі профільних даних, принципи організації співпраці в онлайн-середовищі, а також засоби синхронізації інформації між модулями нетворкінгу та таск-менеджменту..

Методи дослідження

Для досягнення поставленої мети дослідження та виконання визначених завдань було використано комплекс наукових та прикладних методів, які дозволили здійснити повноцінний аналіз проблеми, розробити ефективне рішення та оцінити його результативність. Зокрема, застосовувалися наступні методи:

- **Аналіз літературних джерел та огляд існуючих рішень** — використано для вивчення сучасних тенденцій у сфері професійного нетворкінгу та таск-менеджменту, аналізу функціональних можливостей популярних платформ (таких як LinkedIn, Upwork, Trello, Jira тощо), а також для визначення їхніх сильних і слабких сторін з метою формулювання вимог до нового вебсервісу.
- **Метод розробки програмного забезпечення** — реалізовано шляхом створення вебзастосунку з використанням сучасних технологій фронтенду та бекенду (зокрема, React для інтерфейсу, Node.js і Python для серверної логіки, SQL/NoSQL для роботи з базами даних). Цей метод охоплював як проєктування структури застосунку, так і реалізацію окремих його модулів.

- **Метод тестування та оцінки ефективності** — застосовувався на завершальному етапі роботи для перевірки працездатності системи, виявлення помилок, а також збору відгуків користувачів щодо зручності та корисності функціоналу. Застосування цих методів у комплексі забезпечило системний підхід до дослідження і дозволило створити конкурентоспроможне програмне рішення.

Структура роботи

Дипломна робота складається з восьми основних розділів, кожен з яких розкриває окремі аспекти дослідження, аналізу та реалізації вебсервісу для пошуку партнерів з інтегрованою системою управління завданнями.

- **Вступ** містить обґрунтування актуальності обраної теми, формулювання мети, завдань, об'єкта та предмета дослідження, а також короткий опис методів, що були використані у процесі роботи.
- **Огляд існуючих рішень у сфері професійного нетворкінгу та таск-менеджменту** присвячено аналізу популярних онлайн-платформ для пошуку партнерів (LinkedIn, AngelList, Upwork) та систем управління проєктами (Trello, Asana, Jira). У цьому розділі порівнюються їх функціональні можливості, сильні та слабкі сторони.
- **Аналіз вимог до вебзастосунку для пошуку партнерів і управління завданнями** охоплює визначення функціональних і нефункціональних вимог, які ставляться до розроблюваного сервісу, з урахуванням потреб користувачів та специфіки бізнес-взаємодії.
- **Процес розробки вебзастосунку** описує етапи розробки системи, вибір архітектурного підходу, структуру модулів, а також обґрунтування вибору технологій (React, Node.js, Python тощо).
- **Імплементація функціональності пошуку партнерів та підбору фахівців** розглядає реалізацію логіки пошуку партнерів, структуру зберігання профільної інформації та принципи взаємодії між користувачами.

- **Розробка та впровадження інструментів для управління завданнями** присвячена створенню інтерфейсу таск-менеджменту, логіці створення, призначення і контролю виконання завдань у межах проєктів.
- **Інтелектуальна система рекомендацій для підбору партнерів і виконавців** описує алгоритми машинного навчання або евристичні методи, які використовуються для персоналізованого підбору користувачів на основі аналізу їх профілів, навичок і попередньої активності.
- **Оцінка ефективності використання вебзастосунку** включає тестування вебсервісу, аналіз продуктивності, зручності користування, а також порівняння результатів із вже існуючими аналогами.

Завершує роботу висновок, у якому підбиваються підсумки дослідження, робиться оцінка досягнення поставленої мети, а також надаються пропозиції щодо подальшого розвитку системи.

1. ОГЛЯД ІСНУЮЧИХ РІШЕНЬ У СФЕРІ ПРОФЕСІЙНОГО НЕТВОРКІНГУ ТА TASK-МЕНЕДЖМЕНТУ

1.1. Огляд популярних вебсервісів для пошуку партнерів (LinkedIn, Upwork, AngellList, тощо)

У сучасному бізнес-середовищі наявність онлайн-платформ для пошуку партнерів є важливою складовою успішної діяльності компаній і окремих фахівців. Такі платформи дозволяють знаходити потенційних партнерів, клієнтів та співробітників, налагоджувати бізнес-зв'язки, що значно прискорює процеси розвитку та масштабування бізнесу.



Рисунок 1.1 — Логотип компанії LinkedIn

LinkedIn — одна з найбільших соціальних мереж для професіоналів, яка активно використовується для побудови бізнес-зв'язків, пошуку партнерів і фахівців. Це найбільша у світі платформа для професійного нетворкінгу, яка налічує понад 700 мільйонів користувачів по всьому світу.

Основні можливості LinkedIn:

- Профіль користувача як резюме, що дозволяє потенційним партнерам оцінити професійний досвід.
- Можливість встановлювати ділові зв'язки з людьми в різних галузях через запити на додавання в контакти.

Переваги:

- Величезна кількість користувачів, що забезпечує великий потенціал для знаходження партнерів у різних сферах.
- Високий рівень довіри, оскільки користувачі часто публікують свої професійні досягнення, проекти та рекомендації.

Недоліки:

- Відсутність інструментів для прямого управління проектами та завданнями.
- Хоча можна шукати партнерів за різними критеріями, платформа не надає спеціалізованих інструментів для ефективної співпраці між партнерами.

1.2. Огляд сучасних рішень в області таск-менеджменту

Управління завданнями та проектами є критично важливим етапом у забезпеченні ефективної роботи команд і досягненні поставлених цілей у бізнесі, особливо в умовах сучасного стрімкого темпу розвитку технологій та глобалізації.

Системи таск-менеджменту стали незамінними інструментами для організації роботи, планування завдань, відстеження прогресу, комунікації та забезпечення зворотного зв'язку. Ці інструменти дозволяють командам бути не тільки більш продуктивними, а й організованими, знижуючи ймовірність помилок і затримок.

Перша платформа, яку я пропоную розглянути це Trello.



Рисунок 1.4 — Логотип компанії Trello

Trello — це одна з найбільш популярних і простих у використанні онлайн-платформ для управління завданнями та проектами. Trello базується на методології Kanban і дозволяє користувачам створювати дошки, списки та картки для організації робочих процесів.

Основні можливості Trello:

- Дошки та картки: Кожен проєкт можна організувати в окрему дошку, що містить списки і картки. Картки містять завдання, які можна пересувати між списками в залежності від їхнього статусу.
- ***Trello має наступні переваги:***
- Можливість гнучко налаштовувати робочі процеси.

Також хочу зазначити недоліки даної платформи:

- По-перше, відсутність розширених функцій для комплексного управління проєктами, таких як тайм-менеджмент або фінансовий облік.

Далі хочу розглянути таку платформу, як Asana.



Рисунок 1.5 — Логотип компанії Asana

Asana — потужний інструмент для управління проєктами та завданнями, орієнтований на більші команди та організації. Asana дозволяє не тільки відслідковувати індивідуальні завдання, а й управляти комплексними проєктами з чітким визначенням етапів і термінів.

Основні можливості Asana:

- Проєкти та завдання: Asana дозволяє створювати проєкти та завдання з визначеними дедлайнами, описами, відповідальними особами та пріоритетами.

Також хочу зазначити переваги та недоліки Asana.

Переваги:

- Це можливість гнучко налаштовувати завдання та проєкти під потреби команди.

Недоліки:

- Платна версія є доволі дорогою для малих команд.
- Важче освоїти новим користувачам, ніж більш прості інструменти.

1.3. Інтеграція пошуку партнерів з таск-менеджментом

Система управління задачами або таск-менеджер — сервіс, який дозволяє зібрати всі робочі завдання команди в єдиному просторі. Це свого роду “кардіограма” роботи компанії, на якій видно, чи все йде так, як треба, чи є збої, які треба ліквідувати.

Системи для управління задачами однозначно роблять роботу команди **більш злагодженою**. А ще таск-менеджери прискорюють вирішення багатьох питань, наприклад: пошук потрібних документів, прогнозування часу, потрібного для виконання типових завдань, розрахунок заробітку тим, хто працює на погодинній оплаті.

Сервіси з управління проектами зводять до мінімуму можливість щось забути або втратити, особливо, в режимі багатозадачної роботи.

Виклики інтеграції пошуку партнерів з таск-менеджментом

- Складність обробки великих обсягів даних

Вебсервіси для пошуку партнерів і таск-менеджменту часто працюють з великими обсягами даних, такими як профілі користувачів, деталі про бізнеси, інформація про виконані завдання тощо. Інтеграція таких двох систем потребує ретельного опрацювання даних, ефективних алгоритмів пошуку і фільтрації, а також надійної інфраструктури для їх зберігання і обробки.

- Забезпечення безпеки та конфіденційності

Інтеграція двох систем, одна з яких працює з особистими даними користувачів (пошук партнерів), а інша — з інформацією про виконання завдань і проекти, ставить підвищені вимоги до безпеки. Потрібно розробити ефективні механізми захисту даних, запобігання несанкціонованому доступу та забезпечення конфіденційності як бізнесових, так і особистих даних.

- Потреба в адаптивних інтерфейсах

Для забезпечення комфортного користування інтегрованими інструментами потрібно розробити адаптивний інтерфейс, який дозволяє легко перемикатися між функціями пошуку партнерів і управління завданнями. Користувач повинен мати змогу без проблем переміщатися між двома основними функціями, не відчуючи складнощів при використанні.

- Підтримка різних сценаріїв використання

Платформи для пошуку партнерів та таск-менеджменту можуть використовуватися в різних сферах і для різних типів співпраці (від фріланс-проектів до великих корпоративних партнерств). Інтеграція цих інструментів повинна підтримувати різні варіанти використання, що може бути складним завданням. Це вимагає гнучкої архітектури, яка дозволяє адаптувати функціональність під конкретні потреби користувачів.

Шляхи інтеграції пошуку партнерів і таск-менеджменту

- Інтеграція через API та вебсервіси

Одним з основних способів інтеграції таких двох систем є використання API. Вебсервіси для пошуку партнерів (наприклад, LinkedIn, AngelList) можуть бути інтегровані з платформами для управління завданнями (Asana, Trello, Jira) за допомогою API, що дозволяє передавати дані з одного інструменту в інший без необхідності вручну переносити.

Приклад інтеграції в реальному житті

Upwork + Asana: На платформі Upwork можна знайти фрілансерів, що працюють у різних сферах, таких як веброзробка, дизайн, маркетинг тощо. Після того як партнер знайдений, всі завдання можуть бути призначені через Asana для ефективного управління проектом. Платформа Asana дозволяє створювати проекти та завдання, визначати дедлайни, а також вести спільну комунікацію з партнером, що дозволяє безперешкодно контролювати хід роботи.

Тому *інтеграція пошуку партнерів з таск-менеджментом* має великий потенціал у підвищенні ефективності бізнес-процесів. Вона дозволяє знизити витрати часу на організацію роботи та пошук потрібних фахівців, а також забезпечує зручне управління проєктами, гарантуючи прозорість, контроль і результативність.

Однак для її успішного впровадження необхідно враховувати ряд технічних та організаційних викликів, таких як безпека даних, ефективна обробка великих обсягів інформації та забезпечення зручності користувачів.

1.4. Переваги та недоліки існуючих рішень

У цій частині пропоную розглянути переваги та недоліки існуючих рішень на основі огляду існуючих рішень у сфері професійного нетворкінгу та таск-менеджменту.

Переваги:

- Широкий вибір інструментів. Сьогодні на ринку існує великий вибір платформ для пошуку партнерів і управління завданнями, що дає можливість кожному користувачеві обирати оптимальний варіант залежно від потреб і бюджету.
- Багато рішень, зокрема Trello, Asana та Monday.com, мають високу гнучкість у налаштуваннях, що дозволяє адаптувати їх під специфічні потреби користувачів.
- Всі платформи для таск-менеджменту підтримують інтеграцію з іншими інструментами, що дозволяє створювати цілісні робочі процеси.

Недоліки:

- Більшість платформ для пошуку партнерів не мають вбудованих функцій для управління завданнями, що потребує додаткових зусиль для інтеграції цих інструментів.
- Деякі платформи, такі як LinkedIn або AngelList, спеціалізуються на пошуку партнерів, але не на управлінні завданнями або проєктами, що робить їх малоефективними для комплексного управління проєктами.

- Такі платформи, як-от Jira, можуть бути складними для освоєння новими користувачами через свою багатофункціональність та орієнтацію на технічні команди.

Таким чином, хоч існуючі рішення і мають значні переваги, інтеграція пошуку партнерів з таск-менеджментом залишається актуальною задачею для покращення ефективності роботи користувачів.

Висновки до розділу

Проведений огляд показує, що на ринку існує чимало ефективних інструментів як для професійного нетворкінгу, так і для управління завданнями. Вони забезпечують високу гнучкість, інтеграцію з іншими сервісами та широкий вибір функціональностей для користувачів із різними потребами. Водночас відсутність універсального рішення, яке б поєднувало можливості пошуку партнерів із повноцінним таск-менеджментом, створює виклики для організації цілісного та ефективного процесу управління проектами. Це підкреслює актуальність розробки нових рішень або удосконалення існуючих платформ шляхом об'єднання функцій нетворкінгу та управління завданнями в єдиному інтерфейсі.

2. АНАЛІЗ ВИМОГ ДО ВЕБЗАСТОСУНКУ ДЛЯ ПОШУКУ ПАРТНЕРІВ І УПРАВЛІННЯ ЗАВДАННЯМИ

2.1. Функціональні вимоги до вебсервісу

Функціональні вимоги до вебзастосунок для пошуку партнерів і управління завданнями визначають основні можливості та функції, які повинні бути реалізовані для забезпечення ефективного користування системою. Вебсервіс має виконувати низку задач, що охоплюють пошук партнерів, управління проєктами та завданнями, а також взаємодію між користувачами.

Нижче хочу описати 10 основних функціональних вимог до даного вебзастосунок.

а) Реєстрація та аутентифікація користувачів

Користувачі повинні мати можливість зареєструватися в системі, надаючи основні дані: ім'я, електронну адресу, пароль. Можна передбачити можливість реєстрації через соціальні мережі або корпоративні акаунти.

Також після реєстрації користувач повинен мати можливість увійти в систему за допомогою наданих облікових даних (електронна адреса та пароль), а також змінювати ці дані через особистий кабінет. Ще має бути можливість відновлення паролю через електронну пошту або за допомогою SMS-коду.

б) Профіль користувача

Користувачі повинні мати персональні профілі, які містять наступну інформацію:

- Ім'я, прізвище, контактні дані (телефон, електронна адреса).
- Інформація про діяльність (професійна діяльність, спеціалізація, інтереси).
- Опис компанії або організації (для бізнес-користувачів).
- Можливість завантаження аватару.
- Історія активності: участь у проєктах, виконані завдання, відгуки тощо.

в) Пошук партнерів

Вебзастосунок повинен дозволяти користувачам шукати партнерів за різними критеріями, наприклад:

- Географічне розташування.
- Тип партнерства (бізнес, проєкт, технології).
- Рейтинг користувача або компанії.
- Спеціалізація чи сфера діяльності.
- Мова спілкування.
- Рекомендації партнерів: На основі історії взаємодії користувачів та їх інтересів, система повинна пропонувати потенційних партнерів.
- Перегляд профілю партнера: Користувачі повинні мати можливість переглядати детальний профіль потенційного партнера, включаючи його активність, відгуки інших користувачів та інші відповідні дані.

г) Управління проєктами та завданнями

Користувачі повинні мати можливість створювати проєкти, визначати їх мету, опис, часові рамки та ресурси. Відповідно до цього в межах кожного проєкту повинна бути можливість створення завдань з детальним описом, строками виконання, пріоритетами та відповідальними особами. І звичайно ж користувачі повинні мати можливість призначати завдання на конкретних учасників проєкту, а також вказувати пріоритети завдань та їх терміни виконання.

д) Сповіщення та повідомлення

Користувачі повинні отримувати сповіщення про нові завдання, зміни статусу завдань, нові коментарі та інші важливі події, пов'язані з їхніми проєктами. Також крім внутрішніх повідомлень на платформі, система повинна надсилати сповіщення на електронну пошту або через SMS, якщо це вказано в налаштуваннях користувача. І відповідно до цього користувачі повинні мати можливість налаштовувати частоту та тип сповіщень (через електронну пошту, SMS, push-сповіщення).

е) Інтеграція з іншими системами

Вебзастосунок повинен підтримувати можливість імпорту та експорту даних у популярних форматах (CSV, Excel), щоб користувачі могли переносити свої дані з інших систем. А для планування завдань та подій у проєктах вебзастосунок відповідно повинен мати можливість інтеграції з популярними календарями (Google Calendar, Microsoft Outlook).

Також вебсервіс може інтегруватися з такими платформами, як Slack або Microsoft Teams, для покращення комунікації між учасниками проєкту.

є) Управління ролями та правами доступу

У системі повинні бути визначені різні ролі користувачів (адміністратор, менеджер проєкту, учасник, гість), кожна з яких має різні рівні доступу до функцій та даних. А от адміністратори повинні мати можливість налаштовувати права доступу для кожного користувача, охоплюючи можливість редагувати проєкти, завдання, додавати нових учасників тощо.

ж) Аналітика та звітність

Вебзастосунок має надавати можливість генерувати звіти про виконання завдань та проєктів, що містять статистику по термінам виконання, кількості завершених завдань, прогресу тощо. Також система повинна надавати інструменти для аналізу ефективності роботи команд, часу виконання завдань, наявності затримок та інших показників.

з) Підтримка мобільних пристроїв

Вебзастосунок має бути адаптованим для мобільних пристроїв або підтримувати розробку окремого мобільного застосунку для iOS та Android, щоб користувачі могли зручно працювати з платформою на ходу.

и) Безпека

Усі персональні дані користувачів та інформація про проєкти повинні бути зашифровані для забезпечення конфіденційності. Також платформа повинна мати

механізми резервного копіювання даних для захисту від втрати інформації. До того ж сама система повинна реалізувати механізми захисту від несанкціонованого доступу, тобто мати двофакторну аутентифікацію.

Для гарантування цілісності, конфіденційності та доступності даних реалізуються такі ключові заходи:

Шифрування даних

- Усі **персональні дані** (імена, email-адреси, телефони, біографічна інформація) зберігаються в зашифрованому вигляді з використанням сучасних симетричних та асиметричних алгоритмів (AES-256, RSA).
- **Паролі користувачів** шифруються з використанням **bcrypt** з високим рівнем складності хешування.
- Передача даних між клієнтом і сервером здійснюється **виключно через HTTPS**, що гарантує захист від перехоплення та атак типу "людина посередині" (MITM).

Резервне копіювання даних

- Регулярне **автоматичне створення резервних копій** бази даних та файлів проєктів (щоденно, щотижнево).
- Зберігання резервних копій у **захищених хмарних сховищах**, із географічно рознесеним дублюванням.
- Впровадження механізмів **швидкого відновлення** системи після збоїв (disaster recovery plan).

Захист від несанкціонованого доступу

- Реалізована **двофакторна аутентифікація (2FA)** для захисту облікових записів. Підтвердження входу відбувається через SMS, email або мобільний додаток-генератор кодів (TOTP).
- Система автентифікації побудована на базі **JWT-токенів** з обмеженим терміном дії та механізмом оновлення (refresh tokens).

- Підтримка **рольової моделі доступу (RBAC)**, що дозволяє контролювати рівень прав кожного користувача (адміністратор, менеджер, учасник проєкту тощо).
- Усі критичні дії (зміни в проєктах, доступ до аналітики, експортування даних) **логуються** з можливістю подальшого аудиту.

Додаткові заходи безпеки

- Захист від атак типу **Brute Force** через обмеження кількості спроб входу та CAPTCHA.
- Використання політик **CORS**, **CSRF-токенів** і фільтрації вхідних даних для запобігання поширеним вебзагрозам (XSS, SQL-ін'єкції).
- Проведення періодичного **пентестування** (penetration testing) та сканування на вразливості.

Завдяки цим заходам платформа забезпечує **високий рівень інформаційної безпеки**, що є необхідним для довіри з боку користувачів, а також для відповідності нормативним вимогам у сфері захисту даних, зокрема GDPR. Усі безпекові механізми реалізовано з можливістю подальшого масштабування та адаптації до зростаючих вимог і нових викликів у сфері кіберзахисту.

2.2. Розроблення нефункціональних вимог

Нефункціональні вимоги визначають характеристики та параметри системи, які не стосуються безпосередньо функціональних можливостей вебзастосунку, але є критичними для забезпечення ефективної роботи системи, її стабільності, зручності використання та відповідності сучасним вимогам.

Нижче хочу розповісти про основні нефункціональні вимоги до вебсервісу для пошуку партнерів і управління завданнями.

Продуктивність

Вебзастосунок має забезпечувати швидку реакцію на запити користувачів. Час відгуку при запитах до сервера не повинен перевищувати 2 секунд для найбільш поширених операцій (завантаження сторінки, пошук партнерів, оновлення статусу завдання).

До того ж система повинна мати можливість масштабування для підтримки зростаючої кількості користувачів та великих обсягів даних. Архітектура вебзастосунок повинна підтримувати горизонтальне та вертикальне масштабування. Також хочу додати, що вебзастосунок повинен ефективно обробляти великі обсяги даних (наприклад, кілька тисяч профілів партнерів, численні завдання та проекти).

Надійність

Система повинна бути доступною 24/7, із забезпеченням мінімальних перерв у роботі. Для цього необхідне використання механізмів резервування серверів, автоматичного відновлення та балансування навантаження. Також їй необхідно витримувати незначні відмови окремих компонентів без значного впливу на роботу користувачів. Для цього передбачається використання кластеризації та високодоступних баз даних. І звичайно ж потрібно реалізувати механізми моніторингу та логування для виявлення та швидкого реагування на проблеми системи.

Безпека

Вебзастосунок має підтримувати надійні механізми аутентифікації та управління доступом, щоб захистити дані користувачів від несанкціонованого доступу. А всі чутливі дані (особисті дані користувачів, паролі, інформація про проекти та завдання) повинні зберігатися в зашифрованому вигляді, а передача даних між клієнтом і сервером повинна здійснюватися через зашифровані канали (SSL/TLS).

До того ж вебзастосунок повинен бути захищений від атак типу SQL-ін'єкцій, XSS (міжсайтових скриптів) і CSRF (міжсайтових підробок запитів). Для цього необхідно використовувати сучасні методи обробки даних і безпечні бібліотеки.

Зручність користування

Вебзастосунок повинен мати інтуїтивно зрозумілий інтерфейс, що забезпечує зручну навігацію, мінімізацію кількості необхідних дій для виконання основних функцій, а також логічну та послідовну структуру елементів інтерфейсу. І звичайно має бути адаптованим до різних розмірів екрану та пристроїв (ПК, планшети,

смартфони), що дозволить користувачам комфортно працювати з системою в будь-якому середовищі.

Локалізація та підтримка мов

Вебзастосунок має підтримувати багато мов, зокрема, українську та англійську, для забезпечення комфортного використання як для локальних, так і для міжнародних користувачів. І повинен містити можливість зміни мови інтерфейсу без необхідності перезавантаження сторінки.

Сумісність

Вебзастосунок має бути сумісним з основними веббраузерами (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge) та різними операційними системами (Windows, macOS, Linux). При цьому ще йому необхідно коректно працювати з популярними мобільними операційними системами (iOS, Android) при використанні мобільних браузерів або відповідних застосунків.

Документація

Вебзастосунок має супроводжуватися докладною документацією для кінцевих користувачів, що пояснює основні функції, процеси та способи вирішення типових проблем. Також необхідно мати технічну документацію для розробників, яка охоплює архітектуру системи, інтерфейси програмування (API) та інші важливі аспекти.

Інтернаціоналізація та підтримка різних валют

Тут хочу зазначити, що для міжнародних користувачів потрібно забезпечити підтримку різних валют та можливість вибору валютних одиниць у проєктах, які мають фінансовий аспект.

2.3. Специфікація інтерфейсу користувача

Інтерфейс користувача є важливим компонентом вебзастосунку, оскільки він забезпечує взаємодію користувачів із системою. Важливо створити інтерфейс, який буде не лише зручним, але й інтуїтивно зрозумілим та функціональним. Ось основні елементи інтерфейсу:

Головна сторінка

- **Панель навігації:** Охоплює такі основні розділи, як "Головна", "Мій профіль", "Пошук партнерів", "Мої проєкти", "Сповіщення", "Налаштування". Панель навігації повинна бути легко доступною та мати чітке розташування елементів.
- **Кнопки входу та реєстрації:** Зрозумілі кнопки для входу в систему або реєстрації нового користувача. Вони повинні бути видимими на всіх сторінках, щоб користувачі могли легко увійти чи зареєструватися.
- **Основний блок:** Відображення новин, пропозицій або останніх оновлень на платформі. Це може бути стрічка новин, що постійно оновлюється.
- **Пошукова панель:** Пошук партнерів із фільтрами, що дозволяють знайти потрібного користувача за професійною діяльністю, навичками, місцем знаходження тощо.

Профіль користувача

- **Особиста інформація:** Можливість редагування особистих даних, таких як ім'я, контактні дані, професійна діяльність. Важливо передбачити форму з підтвердженням змін.
- **Інформація про проєкти:** Перелік активних і завершених проєктів, з можливістю їх перегляду та редагування. Кожен проєкт має бути пов'язаний із завданнями, які виконуються в межах цього проєкту.
- **Історія взаємодій:** Список партнерів, з якими користувач співпрацював. Тут можна побачити відгуки про співпрацю, додаткові контактні дані партнерів, а також мати можливість надіслати запит на співпрацю знову.

Сторінка пошуку партнерів

- **Фільтри пошуку:** Пошук потенційних партнерів із можливістю фільтрування за критеріями (професійна діяльність, місцезнаходження, навички, доступність тощо).

- **Результати пошуку:** Список потенційних партнерів, який містить базову інформацію (ім'я, професія, кількість завершених проєктів) та кнопки для перегляду повного профілю.
- **Картка партнера:** При натисканні на профіль партнера відкривається детальна сторінка, що містить більше інформації, контакти та відгуки від інших користувачів.

Сторінка проєкту

- **Основна інформація:** Назва проєкту, його опис, терміни виконання та список учасників.
- **Перелік завдань:** Завдання, що входять до складу проєкту. Можливість створювати нові завдання, редагувати існуючі, призначати відповідальних та змінювати статуси завдань.
- **Комунікація:** Вбудований чат або система коментарів для обміну ідеями та обговорення деталей проєкту між учасниками.

Керування завданнями

- **Список завдань:** Повний перелік завдань з можливістю фільтрації за статусом, пріоритетом, датою виконання. Завдання мають бути структуровані за категоріями (наприклад, "активні", "завершені", "відкладені").
- **Редагування завдання:** Можливість змінювати опис, додавати чи редагувати пріоритет, терміни виконання, додавати коментарі та прикріплювати файли.
- **Сповіщення:** Автоматичні сповіщення для користувачів про нові завдання, зміни статусів, наближення терміну виконання завдань.

Налаштування користувача

- **Редагування контактних даних:** Користувач може змінювати свій номер телефону, email, пароль.
- **Налаштування безпеки:** Можливість змінити пароль, налаштувати двофакторну аутентифікацію для підвищення безпеки облікового запису.

- **Сповіщення:** Користувач має можливість налаштувати, як отримувати сповіщення (електронна пошта, push-сповіщення).
- **Мова інтерфейсу:** Перемикач для вибору мови інтерфейсу, який дозволяє адаптувати платформу до різних користувачів по всьому світу.

Інтерфейс мобільної версії

- **Адаптивність:** Інтерфейс має бути адаптований до різних розмірів екранів, зокрема для смартфонів та планшетів. Елементи інтерфейсу повинні бути зручними для використання на невеликих екранах.
- **Спрощений вигляд:** Для мобільних пристроїв можна передбачити згорнуті меню, спрощене відображення профілю та проєктів для зручності навігації.
- **Пошук та завдання:** Пошукова панель, доступ до завдань і проєктів мають бути легко доступними через спливаючі меню або вертикальне меню для мобільної версії.

2.4. Визначення цільової аудиторії

Цільова аудиторія вебзастосунку для пошуку партнерів і управління завданнями визначається на основі потреб користувачів та специфіки функціонування платформи. Задовольнити потреби цієї аудиторії означає сформувати основу для проєктування інтерфейсу, функціональності, а також для оптимізації взаємодії з платформою.

Як відомо, визначення цільової аудиторії дозволяє точніше налаштувати маркетингові стратегії, вдосконалювати зручність користування та підвищити ефективність роботи вебзастосунку.

Основні групи користувачів

Вік користувачів може коливатися від 18 до 60 років, більшість користувачів – це люди у віці від 25 до 45 років, які активно займаються професійною діяльністю або ведуть бізнес. Ці користувачі орієнтовані на ефективне використання часу, інноваційні технології, зручні інструменти для роботи в команді і прагнуть до росту та розвитку у своїй професійній сфері. Зазвичай у них є потреба в швидкому пошуку надійних

партнерів, організації роботи в команді, створенні і виконанні завдань та в забезпеченні ефективної комунікації.

Малі та середні підприємства (МСП)

Підприємства, які шукають партнерів для розширення бізнесу, співпраці над проектами, розробки нових продуктів або виходу на нові ринки. Вони часто потребують платформи для пошуку надійних партнерів, замовників або постачальників.

Тобто їм необхідні прості та доступні інструменти для організації командної роботи, управління завданнями та проектами. Важливою є, як вже зазначалося в попередньому розділі, можливість налаштування прав доступу для різних ролей у команді та ефективного обміну даними.

Фрілансери та індивідуальні спеціалісти

Тобто це особи, які працюють самостійно, але потребують знаходження партнерів для спільної роботи над проектами або виконання завдань. Це можуть бути дизайнери, розробники, маркетологи, консультанти та інші фахівці.

Для фрілансерів важливі прості інтерфейси для керування завданнями, інструменти для комунікації з партнерами та замовниками, а також можливість переглядати та оновлювати статуси своїх завдань.

Компанії, що займаються проєктним управлінням

В даному випадку це організації, які мають велику кількість проєктів і завдань, відповідно до цього, вони потребують надійних інструментів для управління командами, моніторингу прогресу проєктів та планування ресурсів. Їм необхідне розширене управління проєктами, розподіл завдань серед команди, контролювання виконання завдань у реальному часі, спільна робота над завданнями з іншими учасниками.

Тобто тут ми бачимо високу потребу у функціях для гнучкого управління проєктами та командною роботою, інтеграції з іншими інструментами для управління бізнес-процесами, а також візуалізації прогресу (діаграми Ганта, канбан-дошки тощо).

Інвестори та стартапи

Ці категорії шукають потенційних партнерів для розширення свого бізнесу, розвитку нових ідей, залучення команди для реалізації стартап-проєктів. Інвестори потребують зручних інструментів для пошуку надійних партнерів, аналізу їхніх профілів, перегляду статусу проєктів та відгуків. Стартапи, в свою чергу, повинні мати доступ до потенційних співробітників та партнерів для формування команд та запуску нових ініціатив.

Освітні та консалтингові організації

Освітні установи та консалтингові фірми, потребують ефективних платформ для пошуку партнерів для спільних проєктів, курсових робіт, наукових досліджень або навчальних програм. Також їм потрібні прості та ефективні інструменти для керування групами студентів, викладачів, консультантів тощо. Та можливість залучення зовнішніх партнерів для виконання завдань або розробки спільних програм.

2.5. Визначення варіантів використання та користувацьких сценаріїв

Варіанти використання

Визначення варіантів використання є важливим етапом для опису того, як користувачі будуть взаємодіяти з вебзастосунком. Варіанти використання описують сценарії, у яких користувачі досягають своїх цілей за допомогою системи.

Для вебзастосунку для пошуку партнерів і управління завданнями **можна виділити кілька основних варіантів використання:**

Реєстрація та вхід в систему

Автор: Новий користувач

Користувач створює обліковий запис, вводячи необхідні дані (ім'я, електронну адресу, пароль). Після цього він отримує доступ до функціоналу системи.

Основний потік: користувач натискає на кнопку "Зареєструватися". Вводить необхідні дані. І після підтвердження реєстрації система надає користувачеві доступ до платформи.

Альтернативи: якщо користувач вже зареєстрований, він може здійснити вхід, використовуючи свої облікові дані.

Пошук партнера

Автор: Користувач (підприємець, фрілансер, інвестор)

Користувач використовує фільтри для пошуку партнерів за різними критеріями (географія, спеціалізація, рейтинг) та переглядає їхні профілі.

Основний потік: користувач заходить на сторінку пошуку партнерів. Вибирає фільтри для пошуку та переглядає результати пошуку (вибирає цікавого партнера).

Альтернативи: користувач може звернутися до системи рекомендацій, щоб отримати пропозиції на основі попередніх пошуків.

Створення проєкту

Автор: Користувач (підприємець, менеджер проєкту)

Користувач створює новий проєкт, визначає завдання, термін виконання та призначає відповідальних осіб.

Основний потік: користувач натискає "Створити проєкт". Вводить інформацію про проєкт (назва, опис, терміни). Та додає завдання та учасників.

Альтернативи: якщо проєкт потребує додаткових ресурсів, користувач може запросити партнерів через платформу.

Управління завданнями

Автор: Користувач (менеджер проєкту, учасник проєкту)

Користувач переглядає список завдань, призначає відповідальних осіб, контролює їх виконання, змінює статуси завдань.

Основний потік: користувач заходить на сторінку завдань проєкту. Переглядає завдання, змінює їх статус, додає коментарі. Та повідомляє учасників про зміни.

Альтернативи: користувач може використовувати інтеграцію з іншими інструментами управління проєктами.

Користувацькі сценарії

Користувацькі сценарії — це детальні описи того, як користувачі виконують свої задачі в рамках варіантів використання.

Розглянемо кілька ключових сценаріїв для цього вебзастосунку:

Сценарій 1: Пошук партнера для проєкту

Наприклад: користувач (підприємець) хоче знайти партнера для запуску нового проєкту в сфері розробки програмного забезпечення.

Кроки: користувач заходить на головну сторінку і переходить до розділу «Пошук партнерів». Далі вводить критерії пошуку (спеціалізація — «програмування», локація — «Україна», рейтинг — «4 з 5»). Згодом переглядає список партнерів, вибирає кілька профілів і детально їх переглядає. І у підсумку відправляє запит на співпрацю декільком потенційним партнерам через платформу.

Сценарій 2: Створення проєкту та завдань

Наприклад: менеджер проєкту створює новий проєкт для команди розробників.

Кроки: менеджер переходить до розділу «Мої проєкти» і натискає на «Створити проєкт». Вводить дані про проєкт: назва, опис, терміни. Далі додає завдання для учасників проєкту (розробка інтерфейсу, програмування бази даних тощо). Призначає відповідальних осіб та встановлює терміни виконання завдань. І вже після створення проєкту учасники отримують повідомлення про нові завдання та можуть почати роботу.

Висновки до розділу

У цьому підрозділі було розглянуто основні функціональні можливості вебзастосунку, а також типові сценарії використання з погляду різних ролей користувачів — підприємців, менеджерів проєктів, фрілансерів та інвесторів. Представлені варіанти використання, такі як реєстрація, пошук партнерів, створення проєктів та управління завданнями, демонструють цілісність і логічну структуру взаємодії користувача з платформою. Зазначені сценарії підтверджують актуальність запропонованого функціоналу для ефективної організації командної роботи та пошуку ділових контактів. Таким чином, вебзастосунок має потенціал стати універсальним інструментом для професійного нетворкінгу та управління проєктами.

3. КОНСТРУЮВАННЯ ТА РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1. Вступ до процесу розробки вебзастосунку

Процес розробки вебзастосунку для пошуку партнерів з інтеграцією task-менеджменту є комплексним та багатогранним завданням, що охоплює повний цикл створення програмного забезпечення — від постановки цілей і формування вимог до тестування, оптимізації та підготовки до розгортання системи. Розробка була спрямована на вирішення конкретних завдань, які виникають у сучасних користувачів у процесі професійної співпраці: ефективний пошук партнерів, організація командної роботи, управління завданнями, контроль за їх виконанням та збереження історії взаємодії.

Нижче детально описано ключові етапи розробки.

3.2. Формування вимог

- Проведено аналіз ринку, потреб цільової аудиторії та наявних аналогів;
- Сформульовано функціональні та нефункціональні вимоги (пошук партнерів, проєктна співпраця, безпека, масштабованість);
- Створено користувацькі сценарії (use cases) та специфікації до кожного модуля.

3.3. Проєктування архітектури

- Побудовано архітектуру за принципом клієнт-серверної моделі з поділом на фронтенд, бекенд та ML-сервіс;
- Використано C4-модель для створення багаторівневих діаграм структури;
- Передбачено підтримку майбутньої модульності та інтеграцій (через API, Webhooks).

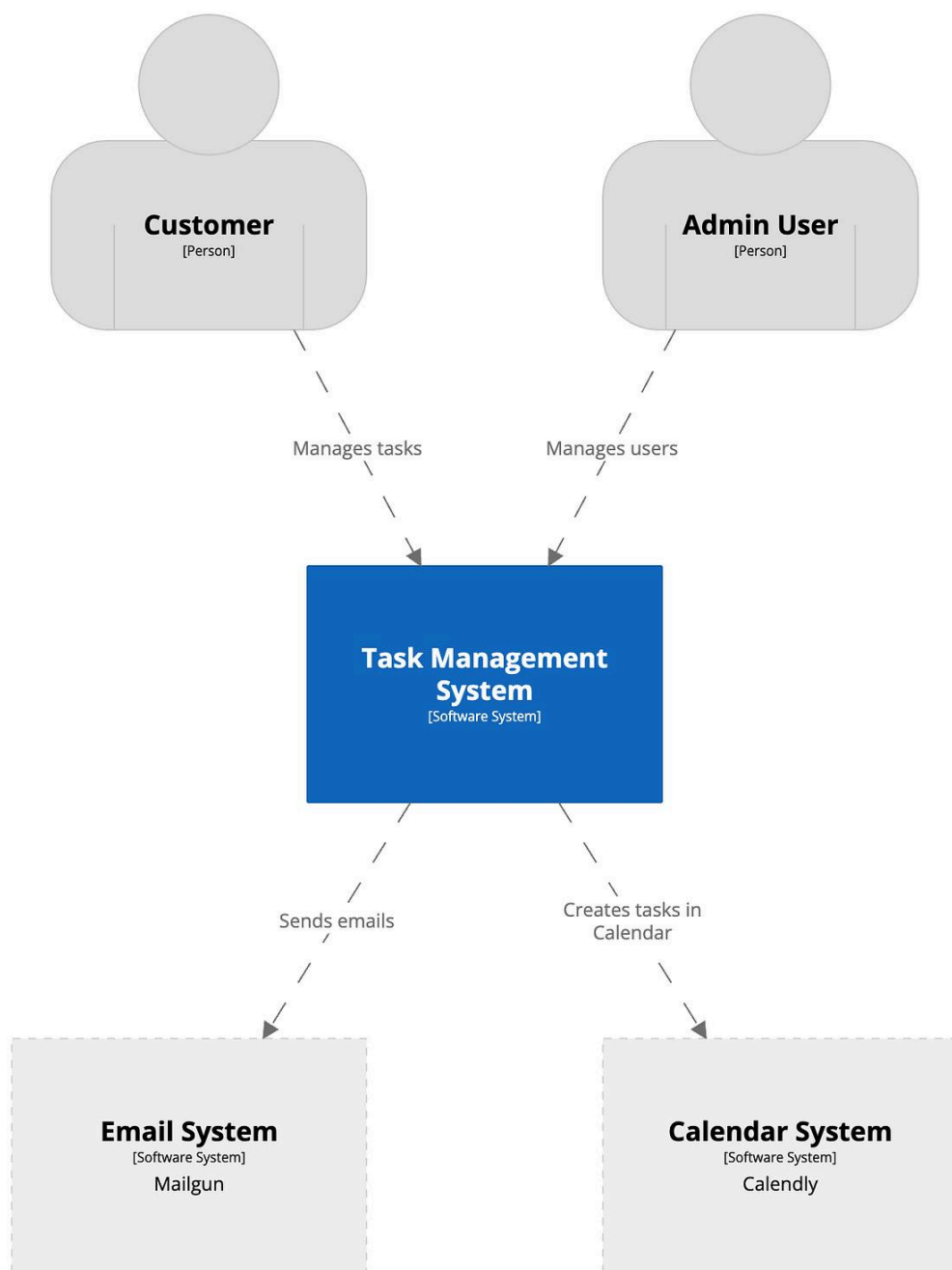


Рисунок 3.1 - Взаємодія користувачів та зовнішніх сервісів із системою керування завданнями

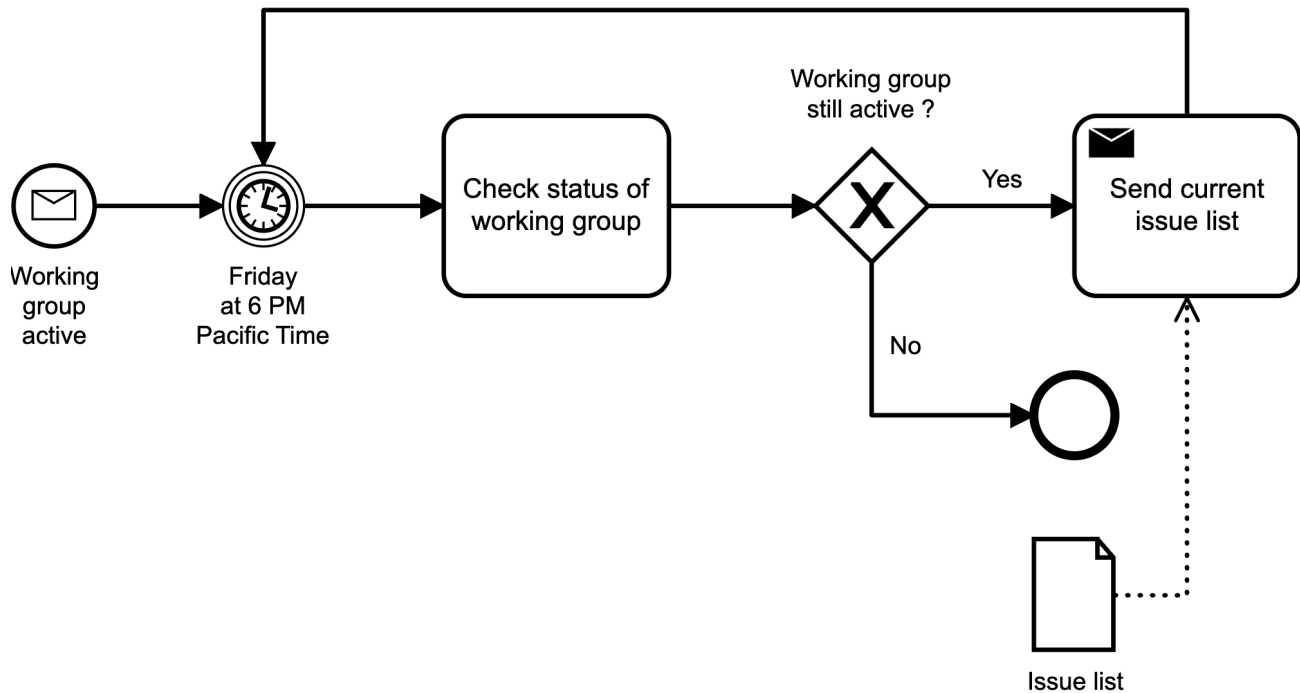


Рисунок 3.2 - Автоматизований процес надсилання списку проблем для активної робочої групи

3.4. Вибір технологій

- **Frontend:** React, Tailwind CSS — для створення інтуїтивного, адаптивного інтерфейсу;
- **Backend:** Node.js (Express), PostgreSQL — як основа логіки й зберігання даних;
- **ML-сервіс:** Python (scikit-learn) — для реалізації гібридної системи рекомендацій;
- **Docker** — для контейнеризації всіх сервісів і забезпечення гнучкого розгортання.

3.5. Розробка MVP

- Реалізовано базову функціональність:
 - реєстрація/авторизація користувача;

- створення профілю компанії та пошук партнерів;
- створення, делегування та відстеження завдань у межах спільного проєкту;
- Застосовано Git та GitHub для контролю версій, командної роботи та супроводу коду.

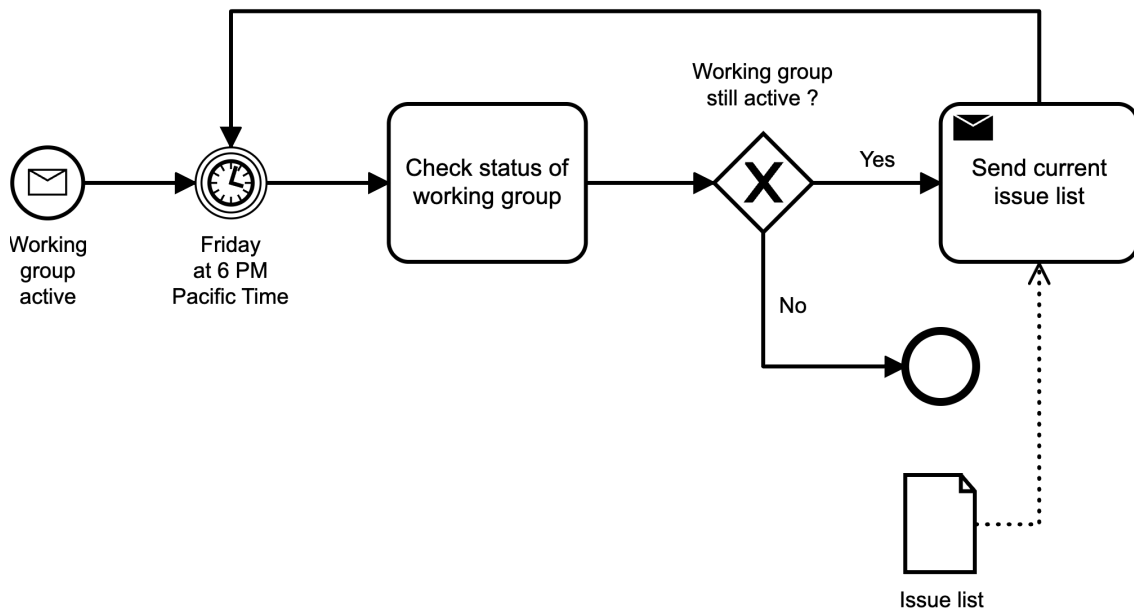


Рисунок 3.3.1 - Автоматизований процес надсилання email для активної робочої групи

3.6. Тестування

- Впроваджено юніт- та інтеграційне тестування ключових модулів;
- Автоматизоване тестове покриття забезпечує стабільність при змінах у коді;
- Виконано мануальне тестування інтерфейсу на різних пристроях.

3.7. Оптимізація та безпека

- Проведено оптимізацію запитів до бази даних, кешування результатів пошуку;

- Реалізовано захист від XSS, CSRF, SQL-ін'єкцій, впроваджено JWT-аутентифікацію та шифрування даних.

3.8. Підготовка до розгортання

- Налаштовано середовище staging для демонстрацій і попереднього тестування;
- Підготовлено CI/CD-процеси для автоматичного розгортання та оновлення.

Таким чином, процес розробки був побудований відповідно до принципів гнучкої методології (Agile), що дозволяло швидко реагувати на зміну вимог, поступово розширювати функціональність та інтегрувати зворотний зв'язок від перших користувачів. Як показано на рисунку 3.1. такий підхід забезпечив створення надійного, масштабованого та зручного вебзастосунку, здатного ефективно вирішувати задачі сучасного бізнес-співробітництва.

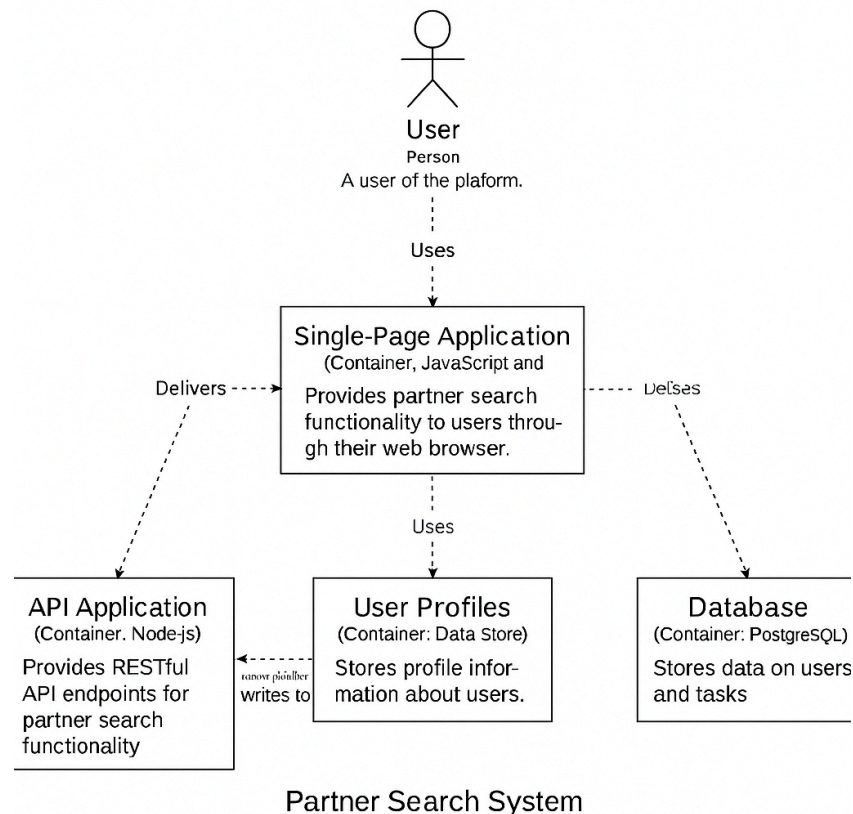


Рисунок 3.3.2 — Схема організації спільної роботи

Кожен етап проєкту реалізовувався з урахуванням ключових вимог до зручності використання, продуктивності, безпеки та можливості масштабування. Нижче подано основні принципи розробки та структурні етапи:

3.9. Планування та аналіз вимог

На цьому етапі було здійснено глибокий аналіз потреб цільової аудиторії та сформульовано функціональні та нефункціональні вимоги до системи. Основна увага приділялася аналізу існуючих рішень та виявленню функціональних прогалин, які новий вебзастосунок має покрити. Також проводився вибір стеку технологій, виходячи з цілей проєкту та очікуваних навантажень.

3.10. Архітектурне проєктування

Розробка архітектури застосунку відбувалась відповідно до принципів **модульності, розширюваності та безпеки**. Було обрано **трирівневу архітектуру**: фронтенд, бекенд та база даних.

- **Фронтенд:** React.js із використанням бібліотек Tailwind CSS або Material UI для побудови адаптивного й естетичного інтерфейсу.
- **Бекенд:** Node.js (або Python з Flask/Django) для обробки запитів, автентифікації користувачів, реалізації логіки таск-менеджменту та пошуку партнерів.
- **База даних:** PostgreSQL або MongoDB (залежно від потреб у структурованості/гнучкості даних).
- **API:** REST або GraphQL для забезпечення зручної взаємодії між фронтендом і сервером.

3.11. Реалізація функціональності

Реалізація функціональних модулів здійснювалася поетапно, з поступовим додаванням основних можливостей:

- Модуль реєстрації та автентифікації (з JWT-токенами, захистом сесій, шифруванням паролів).
- Створення профілів користувачів із редагуванням особистої інформації.

- Розробка інструменту пошуку партнерів з використанням фільтрів та системою рекомендацій.
- Побудова таск-менеджменту з можливістю створення проєктів, додавання завдань, контролю їх статусу, призначення виконавців та взаємодії у межах проєкту.
- Вбудований чат та система сповіщень.

3.12. Забезпечення безпеки

Особливу увагу було приділено захисту персональних даних користувачів. Реалізовано такі механізми:

- Гешування паролів (bcrypt).
- Захист від XSS-атак, CSRF та SQL-ін'єкцій.
- Розмежування прав доступу (рольова модель користувачів).
- Безпечна обробка файлів і обмеження розміру завантажуваних даних.

3.13. Тестування

На цьому етапі здійснювалося:

- Юніт-тестування основних функцій.
- Інтеграційне тестування модулів взаємодії.
- UI/UX-тестування для виявлення помилок інтерфейсу та покращення взаємодії з користувачем.
- Тестування продуктивності при збільшенні навантаження (наприклад, імітація великої кількості одночасних користувачів).

3.14. Оптимізація та підготовка до розгортання

Завершальним етапом стала оптимізація коду, зменшення розміру фронтенд-застосунку, налаштування серверів та бази даних для реального розгортання (на платформі типу Heroku, Vercel, AWS тощо).

3.15. Основні принципи, які лягли в основу розробки:

- **Функціональність:** Забезпечення повного набору інструментів для нетворкінгу та управління проектами.
- **Зручність використання:** Сучасний, інтуїтивно зрозумілий інтерфейс із адаптацією до різних пристроїв.
- **Безпека:** Захист персональних даних та безпечна аутентифікація.
- **Масштабованість:** Можливість подальшого розширення функціональності без повної перебудови архітектури.

Такий підхід до розробки дозволив створити вебзастосунок, що відповідає сучасним стандартам і може ефективно використовуватись як окремими фахівцями, так і командами для співпраці над спільними проектами.

3.16. Вибір архітектури вебзастосунку

Архітектура вебзастосунку відіграє ключову роль у забезпеченні його стабільності, масштабованості, безпеки та ефективної взаємодії з кінцевим користувачем. У контексті створення вебсервісу для пошуку партнерів з інтеграцією функціоналу таск-менеджменту було прийнято рішення реалізувати **багатошарову архітектуру з використанням мікросервісного підходу**.

Таке рішення дозволяє досягти високої гнучкості, легко масштабувати окремі компоненти, забезпечити незалежну розробку модулів і підтримку складних бізнес-процесів.

Основні компоненти архітектури

Клієнтська частина (фронтенд)

- Розроблена за допомогою **React.js**, що забезпечує динамічне оновлення інтерфейсу без необхідності перезавантаження сторінки.
- **Redux** використовується для централізованого управління станом застосунку, що забезпечує узгодженість даних між різними компонентами інтерфейсу.
- **Axios** — для асинхронної обробки HTTP-запитів до бекенду.

- Реалізовано **адаптивний дизайн**, що забезпечує зручність користування на десктопах і мобільних пристроях.

Серверна частина (бекенд)

- Реалізована з використанням **Node.js** і **Express.js**, що забезпечує високу продуктивність, масштабованість і просту обробку великої кількості одночасних запитів.
- Бекенд реалізує **RESTful API**, яке забезпечує зв'язок із фронтом та іншими сервісами.
- Авторизація користувачів відбувається за допомогою **JWT (JSON Web Tokens)**.
- Реалізовано **шар бізнес-логіки**, який відповідає за перевірку прав доступу, обробку даних, реалізацію логіки підбору партнерів, управління завданнями тощо.

База даних

- Використовується **PostgreSQL** як основна СУБД, що забезпечує:
 - підтримку складних SQL-запитів;
 - цілісність та узгодженість даних;
 - можливість використання зв'язків між таблицями для реалізації складної структури проєктів, партнерів, завдань тощо.
- У разі потреби масштабування читання — можлива інтеграція з **репліками** бази даних або кешування за допомогою **Redis**.

Інтеграція сторонніх сервісів

- Застосунок передбачає можливість підключення зовнішніх таск-менеджерів, таких як **Trello**, через відповідні **REST API**.
- Для цього реалізовані окремі мікросервіси, які відповідають за:
 - підключення/авторизацію в сторонньому сервісі;
 - синхронізацію списків завдань;

- збереження стану та змін у базі даних.

Система мікросервісів

- Застосунок розділено на логічно незалежні **мікросервіси**, серед яких:
 - сервіс автентифікації;
 - сервіс пошуку партнерів;
 - сервіс управління завданнями;
 - сервіс рекомендацій;
 - API-агрегатор або gateway.
- Кожен мікросервіс має власну бізнес-логіку і може бути масштабований окремо відповідно до навантаження.
- Комунікація між сервісами може відбуватись за допомогою **HTTP-запитів** або **черг повідомлень** (наприклад, **RabbitMQ**).

Переваги обраної архітектури:

- Гнучкість: можна змінювати або оновлювати окремі частини без впливу на всю систему.
- Масштабованість: легко адаптувати систему до великої кількості користувачів.
- Стійкість до збоїв: відмова одного сервісу не призводить до відмови всієї системи.
- Придатність до розширення: можливість легко додавати нові модулі (наприклад, календар, відеозустрічі, інтеграцію з іншими CRM).

Обрана багаторівнева мікросервісна архітектура ідеально підходить для сучасного вебзастосунку з високими вимогами до продуктивності, модульності та розширюваності. Вона дозволяє створити надійний, масштабований і зручний сервіс, який забезпечує повний цикл співпраці: від пошуку партнерів до управління завданнями в рамках спільних проєктів.

3.17. Технології та інструменти для розробки

Для створення вебзастосунку було обрано потужні та популярні технології, що гарантують високу продуктивність, безпеку та масштабованість:

- **Фронтенд:**

- **React.js** – для побудови інтерактивного інтерфейсу. Завдяки використанню компонентної архітектури, розробники змогли створити UI, який швидко реагує на зміни даних без перезавантаження сторінки.
- **Redux** – для управління станом застосунку. Ця технологія дозволяє зберігати та централізовано управляти даними всіх компонентів, що важливо для відображення актуальної інформації про партнерів і задачі.
- **React Router** – для реалізації маршрутизації в односторінковому застосунку.

- **Бекенд:**

- **Node.js** – вибраний через високу продуктивність при обробці асинхронних запитів та можливість працювати з великою кількістю одночасних з'єднань.
- **Express.js** – фреймворк для Node.js, що забезпечує простоту створення RESTful API для взаємодії між клієнтом і сервером.
- **JWT (JSON Web Tokens)** – для реалізації авторизації та автентифікації користувачів через токени, що гарантує надійний доступ до захищених ресурсів.

- **База даних:**

- **PostgreSQL** – реляційна база даних, що забезпечує цілісність даних, а також підтримує складні запити та транзакції. Вона ідеально підходить для зберігання інформації про користувачів, партнерів та задачі.

- **Інтеграція з Trello:**

- **Trello API** – інтеграція з цим сервісом дозволяє користувачам управляти задачами через сторонній таск-менеджмент. Вебсервіс отримує дані через API-запити та надає користувачам можливість синхронізувати свої задачі з Trello.

- **Безпека:**

- **OAuth2.0** – використовується для автентифікації та авторизації користувачів. Користувач може увійти через популярні сервіси (Google, Facebook), що знижує ризик несанкціонованого доступу.
- **HTTPS** – весь трафік шифрується через HTTPS для забезпечення конфіденційності даних.
- **Інструменти для тестування:**
 - **Jest** – для юніт-тестування компонентів фронкенду та логіки бекенду.
 - **Cypress** – для енд-ту-енд тестування, що дозволяє перевіряти взаємодію між користувачем і системою в реальних умовах.

3.18. Процес розробки та тестування

Процес розробки вебзастосунку був поділений на кілька етапів:

Аналіз вимог та проєктування: На початку проєкту був проведений аналіз вимог до функціональності вебзастосунку. Визначено основні функції: реєстрація та аутентифікація користувачів, пошук партнерів, управління завданнями та інтеграція з Trello. Створено прототипи інтерфейсу в **Figma**, де були чітко визначені всі елементи взаємодії.

Розробка бекенду: Розробка почалася з налаштування сервера, створення бази даних і написання API для взаємодії з фронкендом. Також реалізована логіка інтеграції з Trello через API, а також система обробки запитів користувачів для пошуку партнерів.

Розробка фронкенду: Водночас розроблялася частина клієнтського інтерфейсу. Компоненти для реєстрації, пошуку партнерів, створення задач було реалізовано за допомогою **React** та **Redux** для управління станом.

Тестування: Після завершення розробки кожного модуля проводилося тестування:

- **Юніт-тести** для перевірки логіки окремих компонентів (наприклад, валідація даних, правильність відображення даних).

- **Інтеграційні тести** для перевірки взаємодії між компонентами, таких як запити до бази даних та API.
- **Енд-ту-енд тести** для перевірки взаємодії з користувачем та зовнішніми системами (наприклад, перевірка інтеграції з Trello).

Реліз та підтримка: Після завершення тестування вебзастосунок був розгорнутий на сервері, і розпочалося бета-тестування. Під час бета-тестування збиралися відгуки користувачів, після чого система була оптимізована та виправлені виявлені баги.

3.19. Безпека та конфіденційність даних

Безпека даних є критично важливою складовою будь-якого сучасного вебзастосунку, особливо коли йдеться про сервіси, що працюють з особистою інформацією користувачів, такими як платформи для пошуку партнерів та спільної роботи. У вебсервісі, що розробляється в межах даної дипломної роботи, питання безпеки та конфіденційності були враховані з перших етапів проектування, що дозволяє забезпечити захист даних як на рівні зберігання, так і під час обміну між клієнтом і сервером.

Висновки до розділу

У цьому підрозділі було детально описано етапи розробки, тестування та забезпечення безпеки вебзастосунку. Послідовне проходження всіх фаз — від аналізу вимог і проектування до реалізації фронтенду й бекенду, а також проведення юніт-, інтеграційного та енд-ту-енд тестування — дозволило створити стабільний та функціональний продукт. Особлива увага була приділена питанням безпеки й конфіденційності даних, що є необхідною умовою для сервісу, який працює з особистою інформацією користувачів. У результаті реалізовано вебзастосунок, який не лише відповідає функціональним вимогам, а й забезпечує надійність, зручність та захищеність для кінцевих користувачів.

4. ІМПЛЕМЕНТАЦІЯ ФУНКЦІОНАЛЬНОСТІ ПОШУКУ ПАРТНЕРІВ ТА ПІДБОРУ ФАХІВЦІВ

4.1. Загальна характеристика функціональності пошуку партнерів

Однією з ключових функцій вебзастосунку є можливість **ефективного пошуку партнерів і фахівців** для участі у спільних проєктах, вирішення завдань та формування команд. У сучасному цифровому середовищі, де співпраця часто відбувається в режимі онлайн, можливість швидко знаходити відповідних людей за професійними, географічними або особистісними критеріями набуває особливого значення.

Розроблена система пошуку дозволяє:

- знаходити фахівців із потрібними навичками для конкретних завдань;
- обирати партнерів для довгострокових проєктів;
- формувати збалансовані команди з урахуванням компетенцій, досвіду, рейтингу та сумісності;
- автоматично отримувати рекомендації на основі активності користувача та його інтересів.

У процесі реалізації даної функціональності було враховано вимоги до **масштабованості, точності, швидкодії та персоналізації**. Система побудована з використанням **модульного підходу**: окремо реалізовані логіка пошуку, інтерфейс користувача, система рекомендацій і модуль обробки профілів. Це дозволяє забезпечити гнучкість та легкість у подальшому розширенні або оптимізації окремих компонентів.

Функціональність пошуку реалізовано з урахуванням наступних принципів:

- **Інтуїтивний та адаптивний інтерфейс**, який дозволяє навіть недосвідченим користувачам швидко орієнтуватися в результатах;
- **Гнучка система фільтрів**, яка дозволяє налаштувати критерії пошуку залежно від конкретних потреб;

- **Використання алгоритмів обчислення релевантності**, які враховують не лише введені параметри, але й контекст використання платформи;
- **Інтеграція з профілями користувачів**, що дозволяє переглядати детальну інформацію про кандидатів безпосередньо з результатів пошуку;
- **Рекомендаційний модуль**, що підвищує ймовірність ефективної взаємодії, базуючись на історії активності та подібності профілів.

Для реалізації пошуку використовуються сучасні технології, серед яких:

- **Elasticsearch** для побудови індексованих пошукових запитів;
- **PostgreSQL** з підтримкою повнотекстового пошуку для гнучкої фільтрації в базі даних;
- **сервіси обробки природної мови (NLP)** для семантичного аналізу текстового опису профілів;
- **машинне навчання** для класифікації профілів, побудови рекомендацій та оцінки схожості між користувачами.

Завдяки продуманій архітектурі та застосуванню перевірених рішень, система дозволяє не лише економити час при пошуку фахівців, але й сприяє створенню більш ефективних команд, орієнтованих на результат.

4.2. Алгоритми для пошуку партнерів і фахівців

Для ефективного функціонування пошуку у вебзастосунку було реалізовано комбінацію традиційного фільтраційного пошуку, семантичного аналізу текстових даних і алгоритмів машинного навчання. Такий підхід дозволяє забезпечити гнучкість у пошуку, персоналізацію результатів і підвищену релевантність відповідей для кожного користувача.

Фільтраційний пошук

Базовий рівень пошуку реалізований через систему фільтрів, які дозволяють користувачу вручну налаштувати критерії відбору партнерів. Зокрема, підтримується фільтрація за такими параметрами:

- **галузь діяльності** (ІТ, маркетинг, дизайн тощо);

- **спеціалізація** (наприклад, front-end розробник, UI/UX дизайнер);
- **рівень досвіду** (junior, middle, senior);
- **місцезнаходження або часова зона**;
- **рейтинг користувача** (середній бал відгуків);
- **наявність портфоліо або рекомендацій**.

Семантичний пошук

Для підвищення гнучкості та зручності користування, у пошукову систему інтегровано семантичний пошук, який аналізує **ключові слова** у полях профілю: описі досвіду, переліку навичок, портфоліо тощо.

Технологічна реалізація базується на:

- **Elasticsearch** — високошвидкісний пошуковий рушій, що підтримує fuzzy search (пошук із нечітким співпадінням), синоніми та скоординований аналіз тексту;
- **NLP-бібліотеках** (наприклад, spaCy, NLTK або HuggingFace Transformers) для попередньої обробки текстових полів: видалення стоп-слів, нормалізації, лематизації.

Завдяки цьому користувач може знайти релевантних кандидатів навіть при неідеальному введенні запиту (наприклад, «UX дизанер» → «UX дизайнер»).

Алгоритми схожості профілів

Щоб забезпечити інтелектуальне ранжування результатів пошуку, система обчислює ступінь схожості між профілем користувача та іншими профілями у базі. Для цього використовується **метрика косинусної схожості (cosine similarity)**.

Формула:

$$\text{similarity}(A, B) = \frac{A \cdot B}{||A|| \cdot ||B||}$$

де **A** і **B** — вектори ознак (наприклад, скіли, інтереси, ключові слова профілю).

Це дозволяє:

- Ранжувати результати за ступенем відповідності;
- Створювати персоналізовані пропозиції;
- Уникати дублювання або надмірно загальних результатів.

Вектори профілів формуються з урахуванням вказаних навичок, категорій, минулого досвіду, а також тегів, що генеруються автоматично при створенні або оновленні профілю.

Методи машинного навчання

Для вдосконалення результатів пошуку та автоматичної побудови зв'язків між користувачами застосовано **наступні моделі та алгоритми машинного навчання**:

Кластеризація за інтересами (K-Means)

Метою є об'єднання користувачів у **класи за схожістю** інтересів, скілів або типів проєктів, у яких вони беруть участь. Це дає змогу:

- створити сегменти користувачів (наприклад, «технічні фахівці», «креативні спеціалісти», «керівники проєктів»);
- проводити пошук і рекомендації в межах релевантного кластеру, що підвищує точність;

Класифікація профілів за категоріями

На основі ключових слів, тегів і опису профілю проводиться автоматичне визначення ролі користувача. Для цього застосовуються моделі:

- **Naïve Bayes** — для текстової класифікації;
- **Logistic Regression** або **Random Forest** — для побудови моделі на основі векторизованих скілів.

Це дозволяє системі автоматично відносити нові профілі до певних категорій (наприклад, «розробник», «аналітик», «копірайтер»), навіть якщо користувач не вказав їх явно.

Обчислення рейтингу профілю

Для формування остаточного рейтингу профілю враховуються такі фактори:

- **кількість і якість відгуків;**
- **рівень активності у проєктах** (участь у тасках, коментарях, завершених завданнях);
- **тривалість часу, проведеного у системі;**
- **кількість запитів на співпрацю від інших користувачів.**

Рейтинг оновлюється динамічно на основі нових даних і впливає як на позицію у результатах пошуку, так і на рекомендаційний модуль.

У підсумку, реалізована система поєднує **традиційні методи пошуку з інтелектуальними алгоритмами**, що дозволяє досягти високої якості підбору партнерів і фахівців. Такий підхід не лише оптимізує процес формування команд, а й створює індивідуалізований користувацький досвід.

4.3. Інтерфейс пошуку та фільтрація результатів

Інтерфейс пошуку у вебзастосунку відіграє важливу роль у забезпеченні зручної та ефективної взаємодії користувача з системою. Його структура розроблена таким чином, щоб надати швидкий доступ до розширених можливостей пошуку фахівців або партнерів, забезпечити гнучкість налаштувань, а також візуальну привабливість і зручність навігації.

Основні компоненти інтерфейсу пошуку

Інтерфейс пошуку є **інтегрованою частиною головної панелі користувача** та складається з наступних ключових елементів:

Панель фільтрів

Панель фільтрів дозволяє швидко налаштувати параметри пошуку згідно з потребами користувача. Вона розміщується зліва або у вигляді динамічного випадаючого меню (на мобільних пристроях) і містить такі опції:

- **Тип спеціаліста** (наприклад, програміст, дизайнер, маркетолог);
- **Галузь або напрямок діяльності** (ІТ, освіта, фінанси, e-commerce тощо);
- **Рівень досвіду:** Junior, Middle, Senior;
- **Формат співпраці:** Онлайн, Офлайн, Гібрид;
- **Наявність відгуків або рекомендацій** (перемикачі або чекбокси).

Фільтри реалізовані за допомогою динамічних форм з підтримкою багатовибору, скидання значень, збереження останніх параметрів (наприклад, у localStorage або через бекенд).

Пошукове поле з автозаповненням

Центральним елементом є **інтелектуальне пошукове поле**, що дозволяє користувачу вводити ключові слова, назви навичок або імена користувачів. Реалізовано такі функціональні можливості:

- **Автозаповнення (autocomplete)** на основі існуючих тегів, навичок, категорій та імен;
- **Debounce-метод** — затримка введення (наприклад, 300 мс) перед виконанням запиту, що зменшує навантаження на сервер;
- Інтеграція з бекендом з підтримкою фільтрації запитів у режимі реального часу.

Список результатів пошуку

Результати пошуку відображаються у вигляді **карток профілів**, кожна з яких містить:

- **Фото профілю** (або аватар за замовчуванням);
- **Ім'я та коротку інформацію про фахівця**;
- **Ключові навички та спеціалізацію**;

- **Кнопки швидких дій:**

- «Переглянути профіль» (перехід до розгорнутого профілю користувача);
- «Запросити до проєкту» (відкриття діалогу запрошення);
- «Додати до обраного» (збереження користувача для подальшого перегляду).

Список підтримує **динамічне сортування** (наприклад, за релевантністю, за рейтингом, за новизною) та **пагінацію / нескінченну прокрутку (infinite scroll)**.

UX/UI-особливості та адаптивність інтерфейсу

При розробці інтерфейсу пошуку були враховані принципи **user-centered design** та адаптивності. Основні особливості:

- **Мобільна адаптація:** інтерфейс коректно масштабується для екранів смартфонів і планшетів; основні елементи автоматично змінюють розташування (наприклад, фільтри перетворюються на випадаючі меню).
- **Підтримка темної/світлої теми:** користувачі можуть перемикаати вигляд залежно від умов освітлення або особистих уподобань.
- **Інтерактивні елементи:** кнопки з анімацією, підказки при наведенні, реакції на дії користувача (наприклад, підтвердження додавання в обрані).
- **Доступність (Accessibility):** усі елементи мають підтримку навігації з клавіатури, aria-label атрибути та семантичну розмітку, що полегшує доступ для людей з інвалідністю.

Технічна реалізація

Інтерфейс реалізовано з використанням сучасного технологічного стеку.

У результаті реалізовано зручний, гнучкий та функціональний пошуковий інтерфейс, який дозволяє швидко і точно знаходити релевантних кандидатів, підвищуючи загальну ефективність використання платформи.

4.4. Система рекомендацій для підбору партнерів

Для покращення користувацького досвіду та автоматизації процесу формування команд у вебзастосунку реалізовано **рекомендаційний модуль**, який дозволяє знаходити потенційних партнерів навіть без ручного введення параметрів пошуку. Така система дає змогу враховувати глибший контекст, ніж звичайний пошуковий запит, і допомагає формувати якісні професійні зв'язки на основі реальної взаємодії користувачів з платформою.

Основні фактори, що впливають на рекомендації

Формування списку рекомендованих партнерів здійснюється на основі комбінації **контентних, поведінкових і соціальних характеристик**, серед яких:

- **Історія попередніх співпраць:** участь у спільних проєктах, завданнях або тасках, відгуки одне про одного.
- **Схожість інтересів і навичок:** зіставлення ключових слів, тегів та класифікованих категорій профілю.
- **Взаємна активність:** двосторонній інтерес між користувачами — перегляди профілів, додавання до обраного, позитивна взаємодія (вподобання, оцінки).
- **Поведінкові патерни:** частота входу на платформу, участь у нових тасках, ініціативність у створенні проєктів або команд.

Застосовані алгоритми рекомендацій

У системі реалізовано **гібридну модель рекомендацій**, яка поєднує дві основні стратегії:

Контентно-орієнтована фільтрація (Content-Based Filtering)

Цей підхід використовує інформацію з профілю користувача (набір навичок, інтересів, типові категорії завдань) для формування індивідуальних рекомендацій. Система створює **вектор профілю** користувача та порівнює його з векторами інших користувачів для виявлення найбільш релевантних кандидатів.

Алгоритми:

- TF-IDF / Bag of Words для текстового опису скілів і портфолію;

- Cosine similarity або Euclidean distance для порівняння профілів;
- Вага окремих характеристик може налаштовуватись залежно від важливості (наприклад, збільшення ваги технічних скілів для IT-напрямку).

Колаборативна фільтрація (Collaborative Filtering)

Цей метод базується на **аналізі взаємодії користувачів** між собою. Якщо користувач А має схожі вподобання з користувачем В, і В працював з користувачем С — то С може бути рекомендований А. Це дозволяє:

- виявляти **неочевидні зв'язки**;
- **підвищити персоналізацію** рекомендацій навіть за відсутності детального профілю.

Застосовуються алгоритми:

- User-User Similarity та Item-Item Similarity;
- Матрична факторизація (Matrix Factorization), наприклад, SVD (Singular Value Decomposition);
- ALS (Alternating Least Squares) — ефективна реалізація для великих масивів даних.

Використання графів зв'язків між користувачами

Окремим інноваційним підходом у системі рекомендацій є побудова **графа взаємозв'язків** між користувачами. Кожен вузол графа — це користувач, а ребра — взаємодії (перегляди профілів, спільні проекти, лайки, запрошення тощо).

Для обробки таких структур використано:

- **Алгоритми графового аналізу:**
 - *PageRank* — для визначення «авторитетності» профілю;
 - *Community Detection (Louvain method)* — для виділення кластерів користувачів;
 - *Shortest Path* — для виявлення найкоротшого шляху між потенційними партнерами (наприклад, через спільних знайомих).

Цей підхід дозволяє не лише рекомендувати фахівців, а й будувати **соціальну карту професійної взаємодії** в межах платформи.

Технологічна реалізація

- **Мова реалізації логіки:** Python (з використанням бібліотек **scikit-learn**, **pandas**, **surprise**, **networkx**);
- **База даних:** PostgreSQL (для основних даних) + Neo4j (графова структура);
- **API взаємодії:** REST або GraphQL (в залежності від архітектури застосунку);
- **Кешування результатів:** Redis — для прискорення видачі часто повторюваних рекомендацій;
- **Оновлення моделей:** за розкладом (наприклад, щоденно), або в реальному часі (наприклад, при завершенні нового проєкту).

Переваги реалізованої системи

- Забезпечує **високу релевантність рекомендацій** навіть при обмеженій кількості введених даних.
- Зменшує навантаження на користувача за рахунок автоматичного підбору потенційних партнерів.
- Підвищує **ймовірність успішної співпраці**, використовуючи історичні дані та взаємодію.
- Створює **динамічну екосистему** на платформі, що стимулює зростання активності та кількість завершених проєктів.

4.5. Інтеграція пошуку партнерів з профілями користувачів

Інтеграція механізму пошуку з модулем користувацьких профілів є ключовим аспектом реалізації логіки платформи, оскільки забезпечує **безшовний перехід від пошуку до взаємодії**.

межах даної системи **профіль користувача** виступає не лише як візитна картка, але і як **інформаційний вузол**, що динамічно формується з урахуванням дій, уподобань і взаємодій у межах платформи.

Основні характеристики інтеграції:

- **Розширена структура профілю**

- Профіль містить загальну інформацію про особу або компанію: назву, локацію, галузь, контактні дані;
- Додаткові розділи: досвід, реалізовані проєкти, ключові компетенції, інтереси щодо співпраці, цілі;
- Інтеграція з зовнішніми джерелами (LinkedIn, OpenCorporates, реєстри компаній) для автоматичного збагачення профілю.

- **Пошукова система**

- Пошук здійснюється на основі фільтрів (галузь, тип компанії, регіон, мова, інтереси);
- Передбачена **гібридна модель пошуку**: поєднання ключових слів, структурованих фільтрів і рекомендацій на основі ML-модуля;
- Пошукова видача напряду зв'язана з профілями: користувач може з одного кліку перейти до перегляду детальної інформації або ініціювати контакт.

- **Оцінка релевантності профілю**

- Кожен профіль має **індикатор сумісності**, який формується на основі збігу ключових параметрів з поточними потребами користувача;
- Система відображає **історію активності** (наприклад, участь у проєктах, попередні контакти), що допомагає у прийнятті рішень.

- **Можливості взаємодії**

- З профілю можна одразу надіслати запит на співпрацю, запропонувати створення спільного проєкту або призначити першу зустріч (онлайн/офлайн);

- Передбачено систему повідомлень, коментарів та позначок "обране", що стимулює подальший розвиток контактів.
- **Динамічне оновлення профілю**
 - Усі зміни в профілі користувача автоматично враховуються в пошукових запитах;
 - Рекомендаційна система враховує історію змін для точнішого підбору нових партнерів.

Інтеграція пошуку з профілями створює **логічно завершений цикл взаємодії**: від пошуку — до перегляду — до дії. Такий підхід підвищує ефективність бізнес-матчмейкінгу, дозволяє глибше аналізувати потенційних партнерів і значно спрощує старт співпраці. У результаті це формує **централізовану бізнес-екосистему**, де кожен учасник має інструменти для продуктивного нетворкінгу, з урахуванням не лише поверхневих даних, а й контексту співпраці.

Структура профілю користувача

Кожен користувач після реєстрації отримує персоналізований профіль, який формується на основі введених даних, автоматичного аналізу активності та відгуків. Структура профілю містить наступні компоненти:

- **Контактна інформація**: ім'я, фото, місцезнаходження, посилання на соціальні мережі або портфоліо;
- **Професійний досвід**: опис попередніх місць роботи, позицій, участі в проєктах;
- **Навички та сертифікати**: зазначені вручну або імпортовані з професійних платформ (наприклад, LinkedIn);
- **Портфоліо**: візуальні або текстові приклади виконаних робіт;
- **Поточна зайнятість і графік доступності**: фіксація вільного часу, бажаного формату роботи (онлайн/офлайн/гібрид);
- **Активні та завершені проєкти**: інтеграція з таск-менеджером дозволяє бачити статуси проєктів і завдань у реальному часі;
- **Відгуки та рейтинги**: отримані від колег по проєктах або замовників.

Інтеграція пошуку з профілями

Кожен результат пошуку тісно пов'язаний із профілем користувача. Візуально — це реалізовано у вигляді **картки профілю**, при натисканні на яку відкривається розгорнута інформація про кандидата. Це дозволяє:

- Миттєво перевірити **релевантність профілю до запиту**;
- Отримати повну інформацію без потреби у додаткових переходах між розділами платформи;
- Прийняти рішення про співпрацю на основі повного контексту (від скілів до відгуків).

Також реалізовано **динамічне виділення збігів** із критеріями пошуку: наприклад, якщо користувач шукає спеціаліста з Python, це слово буде підсвічено у профілях, де воно згадується (у скілах або портфоліо).

Функціональність взаємодії з профілем

Крім перегляду, профіль надає можливість миттєвої взаємодії з потенційним партнером. Зокрема, користувач може:

- **Надіслати запит на співпрацю** — ініціація створення зв'язку або спільного проєкту;
- **Запросити до конкретного завдання або проєкту** — через інтеграцію з таск-менеджером (з вибором ролі або дедлайну);
- **Розпочати діалог у чаті або відеозв'язок** — реалізовано через інтеграцію з месенджерами (наприклад, Telegram API, Slack) або технологіями WebRTC для прямого відеозв'язку.

Переваги інтеграції

Тісний зв'язок між пошуком і профілем забезпечує **інтуїтивний та швидкий користувацький досвід**, що особливо критично для середовищ з високою динамікою командної роботи. Основні переваги:

- **Зменшення кількості кліків** від пошуку до дії (співпраці);

- **Покращення конверсії** переглядів профілю у реальні контакти;
- **Оптимізація прийняття рішень** завдяки наочному і повному представленню кандидатів;
- **Підвищення довіри** між користувачами завдяки відкритим відгукам, портфолію та прозорій історії проєктів.

Технічна реалізація

У підсумку, інтеграція пошуку з профілями користувачів забезпечує **єдину точку доступу до всієї релевантної інформації** про потенційного партнера, спрощує прийняття рішень та сприяє ефективному формуванню команд в рамках спільної діяльності на платформі.

Висновки до розділу

Інтеграція пошуку з профілями користувачів значно підвищує ефективність та зручність використання платформи. Завдяки наочному представленню результатів у вигляді інтерактивних карток із динамічним виділенням релевантної інформації, користувачі отримують повний контекст для швидкого прийняття рішень. Можливість миттєвої взаємодії з потенційними партнерами через профіль — від запиту на співпрацю до запуску відеозв'язку — створює безперервний робочий потік без необхідності перемикання між розділами платформи. Такий підхід не лише покращує користувацький досвід, а й сприяє підвищенню конверсії взаємодій у реальну співпрацю, що є ключовим фактором успішності вебзастосунку у сфері професійного нетворкінгу.

5. РОЗРОБКА ТА ВПРОВАДЖЕННЯ ІНСТРУМЕНТІВ ДЛЯ УПРАВЛІННЯ ЗАВДАННЯМИ

5.1. Вступ до реалізації таск-менеджменту

У сучасних умовах динамічного розвитку цифрових технологій ефективно управління завданнями стало невід’ємною складовою успішної реалізації будь-яких програмних продуктів чи бізнес-процесів. Незалежно від розміру команди чи типу проєкту, виникає потреба в системному підході до організації роботи, який дозволяє відстежувати хід виконання завдань, координувати дії між учасниками, оптимізувати розподіл ресурсів та забезпечити прозорість взаємодії.

Розробка функціоналу для управління завданнями є одним із ключових компонентів інформаційної системи, яка впроваджується в рамках цього дипломного проєкту. Такий інструмент покликаний автоматизувати рутинні процеси, підвищити продуктивність командної роботи та мінімізувати ризики втрати інформації або порушення термінів.

У цьому розділі детально розглядаються:

- концептуальні засади таск-менеджменту;
- етапи проєктування та реалізації користувацького інтерфейсу;
- механізми інтеграції завдань з профілями користувачів і проєктами;
- система ролей і прав доступу;
- функціональні можливості для спільної роботи, охоплюючи обмін файлами та коментування.

Описані рішення базуються на сучасних принципах UI/UX дизайну, модульної архітектури та орієнтовані на зручність і масштабованість системи. Це дозволяє інтегрувати її в ширший контекст цифрового середовища організації або стартапу.

5.2. Основи таск-менеджменту

Таск-менеджмент, або управління завданнями, є ключовим елементом у забезпеченні ефективного виконання проєктів, особливо в умовах командної роботи та багаторівневих процесів. Це не лише технічна реалізація інструментів для створення

та контролю завдань, але й загальна методологія організації роботи, яка забезпечує прозорість, відповідальність і своєчасне досягнення цілей.

Суть таск-менеджменту полягає у створенні чіткої структури, яка дозволяє:

- розбивати великі проєкти на менші, контрольовані одиниці (завдання);
- розподіляти ресурси (час, людей, інструменти) відповідно до пріоритетів;
- забезпечити контроль за виконанням завдань у режимі реального часу;
- оперативно реагувати на зміни у проєктному середовищі.

Система управління завданнями повинна підтримувати такі базові функції:

- **Створення та редагування завдань**

Користувачі мають мати змогу створювати нові завдання з вказанням назви, опису, термінів, пріоритету та інших параметрів. У процесі роботи можливість редагування є критичною для врахування змін у вимогах або контексті виконання.

- **Призначення виконавців**

Кожне завдання повинне мати чітко визначеного відповідального користувача або групу виконавців. Це дозволяє не лише розподіляти робоче навантаження, але й підвищити рівень персональної відповідальності в команді.

- **Встановлення термінів виконання (дедлайнів)**

Чітко окреслені часові рамки допомагають планувати навантаження та уникати затримок. Система також повинна забезпечувати нагадування про наближення дедлайнів.

- **Пріоритезація завдань**

Завдання можуть мати різний ступінь важливості та терміновості. Можливість визначення пріоритетів (наприклад, високий, середній, низький) допомагає ефективніше розподіляти ресурси й фокусувати увагу на ключових задачах.

- **Відслідковування статусу виконання**

Система має підтримувати зміну статусу завдання (наприклад: «нове», «в процесі», «на перевірці», «виконано», «скасовано»). Це дозволяє візуально відображати динаміку виконання роботи і створює ефективний канал зворотного зв'язку для керівників проєктів.

- **Сповіщення про зміни**

Оперативні сповіщення (email, push, внутрішні повідомлення) про зміни у завданні, призначення або оновлення дедлайну підвищують залученість користувачів і дозволяють швидше реагувати на нові умови.

Крім перелічених базових функцій, сучасні таск-менеджери часто охоплюють додаткові можливості, як-от аналітика, візуалізація (канбан-дошки, діаграми Ганта), інтеграція з календарями та сторонніми сервісами (Slack, Google Drive, Jira тощо). Однак фундаментальною залишається простота використання, швидкість доступу до інформації та надійність збереження даних.

Таким чином, у межах проєкту особливу увагу було приділено не лише реалізації технічного функціоналу, але й розумінню концепцій таск-менеджменту як методології, яка сприяє підвищенню продуктивності та ефективності командної роботи.

5.3. Розробка інтерфейсу для управління завданнями

Інтерфейс користувача відіграє вирішальну роль у забезпеченні ефективної взаємодії з системою управління завданнями. Його зручність, логічна структура та адаптивність безпосередньо впливають на продуктивність користувачів, особливо в умовах багатозадачності та командної роботи. Тому при розробці інтерфейсу особливу увагу було приділено принципам **UX/UI-дизайну**, серед яких — мінімалізм, зрозуміла ієрархія елементів, швидкий доступ до функціоналу, а також адаптивність до різних типів пристроїв.

Основні елементи інтерфейсу були реалізовані таким чином:

- **Список завдань**

Головний робочий простір інтерфейсу — це таблиця або блоковий список завдань, у якому користувач бачить актуальні задачі відповідно до обраного фільтру. Реалізовано можливість сортування та фільтрації за наступними критеріями:

- статус виконання (нове, в роботі, завершене);
- виконавець;

- проєкт або категорія;
- рівень пріоритету;
- наближення дедлайну.

Завдяки цьому користувачі можуть швидко знаходити релевантну інформацію без необхідності перегортати великі списки.

● **Форма створення/редагування завдань**

Для зручності користувачів реалізовано окрему модальну або сторінкову форму, яка відкривається під час створення нового завдання або редагування наявного.

У ній доступні такі поля:

- назва завдання;
 - опис або технічне завдання;
 - термін виконання;
 - вибір виконавця з-поміж членів проєкту;
 - прикріплення файлів або посилань;
 - встановлення пріоритету;
 - тегування або вибір категорії.
- Також додано можливість миттєвого збереження змін або скасування операції без втрати даних.

● **Панель навігації**

Для швидкого перемикання між розділами реалізована зручна бічна або верхня панель навігації. Основні пункти містять:

- перегляд усіх проєктів;
 - фільтр «мої завдання»;
 - розділ «архів» або «виконані завдання»;
 - адміністративні функції (для користувачів із розширеними правами);
 - доступ до налаштувань профілю та вихід з системи.
- **Візуальні індикатори прогресу**

Для полегшення сприйняття інформації реалізовано кольорове кодування та піктограми:

- статуси задач позначаються кольором (наприклад, червоний — «прострочено», жовтий — «в процесі», зелений — «виконано»);

- використано прогрес-бари для відображення загального стану проєкту;
- відображення часових обмежень (наприклад, іконка годинника або відлік до дедлайну).

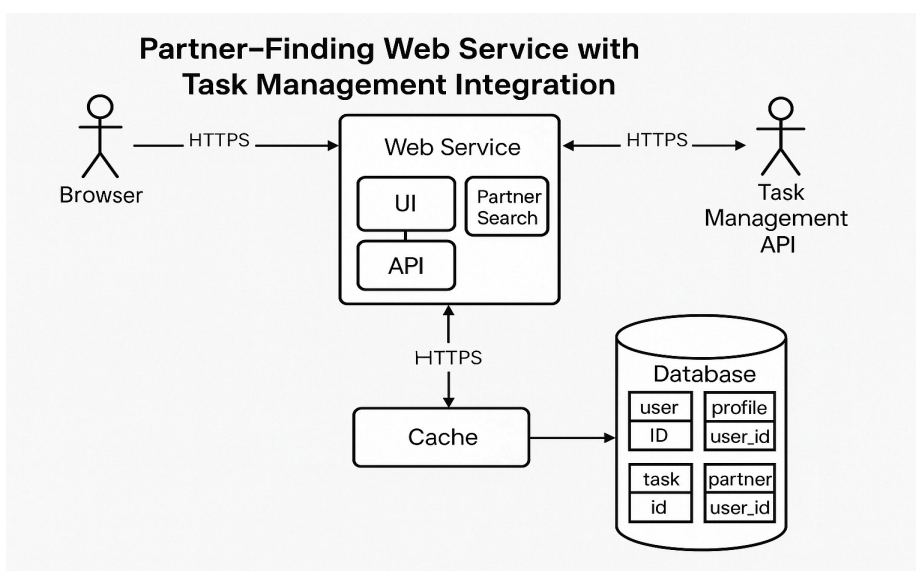
● Адаптивний дизайн

Інтерфейс був протестований і адаптований для мобільних пристроїв (телефони, планшети) з урахуванням сучасних вимог до responsive-дизайну. Це дозволяє користувачам працювати із завданнями на ходу, переглядати статуси, редагувати записи та отримувати сповіщення з будь-якого пристрою, незалежно від платформи.

Таким чином, інтерфейс не лише виконує функцію візуалізації даних, але й значно підвищує продуктивність роботи користувачів завдяки зручній архітектурі, інтуїтивній навігації та гнучким інструментам взаємодії з системою.

5.4. Інтеграція завдань із профілем користувача та проєктом

Однією з ключових функцій системи управління завданнями є її здатність забезпечувати логічну й структуровану прив'язку кожного завдання до відповідного **проєкту** та **профілю користувача**. Це дає змогу досягти високого рівня організації робочих процесів, сприяє персоналізації та забезпечує гнучкість під час формування звітності й аналітики. На рисунку 5.3 продемонстровано вебсервіс пошуку партнерів з інтеграцією керування завданнями



Прив'язка до проєкту

Кожне завдання автоматично належить до певного проєкту або робочої області (work space), що дозволяє:

- об'єднувати завдання за напрямками діяльності;
- розділяти доступ до задач між різними командами;
- формувати звіти та статистику в межах одного проєкту;
- ефективно планувати етапи реалізації на рівні проєктного менеджменту.

Проєкти в системі можуть мати власні налаштування, такі як список учасників, дедлайни, пріоритети або етапи (спринти), що впливає на параметри задач, які в них створюються.

Прив'язка до профілю користувача

Кожне завдання містить інформацію про:

- **автора** (користувача, який створив завдання);
- **виконавця** (користувача, який відповідає за виконання);
- **співвиконавців** (за потреби — при спільному виконанні завдання).

Ця модель дозволяє налаштовувати персональні робочі простори, у яких користувач бачить тільки ті завдання, що стосуються його безпосередньо. Крім того, це спрощує фільтрацію й навігацію в системі, дає змогу керівникам ефективно контролювати навантаження окремих співробітників.

Додаткові функціональні зв'язки

З метою підвищення прозорості процесу та ефективної командної взаємодії, були реалізовані такі механізми:

- **Коментування завдань** — дозволяє учасникам проєкту обговорювати деталі задачі безпосередньо в її картці. Це усуває потребу в зовнішній комунікації та фіксує весь контекст у межах системи.

- **Історія змін** — фіксується кожна зміна в задачі (зміна статусу, виконавця, терміну виконання тощо). Це забезпечує повну **прозорість** і дає змогу легко відстежити дії окремих користувачів.
- **Сповіщення** — у разі змін у завданні, що стосуються конкретного користувача, він автоматично отримує повідомлення (email або в системі), що допомагає уникати пропущених дедлайнів чи оновлень.

Аналітика та звітність

Завдяки інтеграції завдань із користувачами та проектами стало можливим формування таких звітів:

- виконані завдання по кожному користувачу;
- кількість задач у проекті за статусами;
- продуктивність команди за обраний період;
- статистика дотримання термінів виконання.

Таким чином, прив'язка завдань до користувача та проекту не лише підвищує організацію й логіку системи, але й є необхідною умовою для якісного управління ресурсами, контролю виконання та підвищення ефективності всієї команди.

5.5. Визначення ролей та прав користувачів

Одним із ключових аспектів проєктування системи управління завданнями є **реалізація моделі контролю доступу**, яка забезпечує безпеку, зручність використання та ефективне розмежування функціоналу для різних категорій користувачів. Це дозволяє уникнути несанкціонованих змін, гарантувати цілісність даних і забезпечити логічну структуру взаємодії в команді.

Базова модель ролей

У системі реалізована **рольова модель доступу**, де кожна роль має певний набір прав і обмежень. Основні ролі охоплюють:

- **Адміністратор**

Має повний контроль над усіма елементами системи:

- створення, редагування та видалення проєктів і завдань;
- керування користувачами (створення, редагування профілю, призначення ролей);
- доступ до аналітики, журналу дій та налаштувань системи;
- призначення менеджерів та перевизначення виконавців;
- контроль за всіма об'єктами незалежно від приналежності до проєкту.

● Менеджер проєкту

Відповідає за організацію та координацію роботи в межах окремого проєкту:

- створення та редагування завдань у межах свого проєкту;
- призначення виконавців і встановлення дедлайнів;
- контроль статусів задач та аналіз продуктивності команди;
- право коментування й управління внутрішньою документацією;
- обмежений доступ до користувачів (тільки в межах свого проєкту).

● Виконавець

Звичайний учасник команди, який отримує та виконує завдання:

- перегляд призначених завдань;
- зміна статусу завдань («нове», «в роботі», «завершено»);
- додавання коментарів, прикріплення файлів;
- перегляд загального статусу проєкту (без права редагування структури);
- відстеження власного прогресу в системі.

● Гість / Спостерігач

Роль для зовнішніх користувачів або тимчасових учасників:

- лише читання інформації в межах дозволених проєктів;
- немає прав на зміну задач, коментарів або створення об'єктів;
- зручна для клієнтів, інвесторів або зовнішніх аудиторів.

Переваги впровадженої моделі

Безпека та контроль

Рольова модель дозволяє звести до мінімуму ризик випадкового або навмисного порушення цілісності даних.

Простота адміністрування

Завдяки чіткому поділу функціоналу, адміністратор легко керує системою, не створюючи зайвих обмежень для інших користувачів.

Масштабованість

Архітектура системи дозволяє розширити набір ролей або додати індивідуальні права за необхідності.

Гнучкість у роботі над проєктами

Різні рівні доступу допомагають чітко структурувати процес роботи над задачами, зменшуючи кількість помилок і покращуючи командну ефективність.

Таким чином, правильне визначення ролей та їхніх прав у системі управління завданнями є критично важливою складовою функціональності, що забезпечує не лише ефективну організацію командної роботи, а й стабільну й безпечну роботу самої системи.

5.6. Рішення для спільної роботи та обміну документами

У сучасному цифровому середовищі **ефективна командна взаємодія** є ключовим фактором успішної реалізації проєктів. Особливо актуально це для команд, що працюють віддалено або мають розподілену структуру. Враховуючи це, до системи управління завданнями було інтегровано функціонал, який дозволяє не лише керувати задачами, а й організовувати зручне спілкування, обмін файлами та зберігання історії взаємодії.

Основні функціональні можливості:

- **Коментарі до завдань з підтримкою згадувань (@mention):**

Користувачі можуть залишати коментарі безпосередньо під завданнями, а також згадувати інших учасників команди за допомогою символу @. Це дозволяє

залучати відповідальних осіб до обговорення та оперативно вирішувати поточні питання.

- **Можливість прикріплення файлів:**

Для кожного завдання передбачена можливість завантаження та перегляду супровідних файлів (документації, макетів, технічних завдань, таблиць, тощо). Це дозволяє уникнути сторонніх сервісів і зберігати всі дані в єдиному робочому просторі.

- **Реальні сповіщення:**

Система надсилає миттєві повідомлення про:

- зміну статусу задачі;
- нові коментарі;
- призначення нових завдань;
- оновлення термінів або файлів.

Сповіщення реалізовані як через інтерфейс системи, так і за допомогою електронної пошти або мобільних push-повідомлень (у разі наявності мобільного застосунку).

- **Пошук за ключовими словами:**

Реалізовано механізм швидкого пошуку по назвах завдань, змісті коментарів, прикріплених файлах та тегах. Це значно пришвидшує навігацію по проєктах та дозволяє оперативно знаходити потрібну інформацію.

- **Журнал змін (історія дій):**

Для кожного завдання зберігається повна історія змін:

- хто і коли змінював статус;
- що саме було оновлено (наприклад, зміна терміну, опису, пріоритету);
- коли були додані або видалені файли чи коментарі.

Це забезпечує прозорість процесу та дозволяє в разі потреби відновити попередній стан задачі або відстежити дії конкретного користувача.

Переваги впровадження спільної роботи:

Покращення комунікації:

Усі обговорення прив'язані безпосередньо до завдань, що унеможливорює потребу у сторонніх месенджерах або листуванні по email.

Централізація даних:

Усі документи, обговорення та зміни зберігаються в одному місці, що знижує ризик втрати інформації.

Зменшення кількості помилок:

Швидке інформування учасників команди дозволяє своєчасно реагувати на зміни та уникати неправильного трактування вимог.

Підтримка асинхронної роботи:

Навіть у випадку роботи в різних часових зонах, учасники команди можуть залишати коментарі та ділитися файлами, що дозволяє продовжувати роботу над проєктом у зручний для кожного час.

Таким чином, реалізовані інструменти для спільної роботи не тільки підвищують ефективність команди, а й забезпечують **гнучкість, прозорість і зручність у роботі над завданнями**, що є критично важливим у сучасних умовах розробки програмного забезпечення.

Висновки до розділу

У цьому підрозділі було розглянуто ключові аспекти впровадження функціональності спільної роботи над завданнями. Завдяки можливості відстеження всіх змін, доданих файлів і коментарів, забезпечується повна прозорість та контроль за історією змін. Централізація комунікації в межах завдань усуває потребу в зовнішніх каналах зв'язку, мінімізуючи ризики втрати інформації та непорозумінь. Інструменти асинхронної взаємодії дозволяють підтримувати ефективну співпрацю незалежно від місця та часу. У сукупності ці рішення значно підвищують організованість, гнучкість і ефективність командної роботи в межах платформи, що є важливим фактором для успішного управління сучасними проєктами.

6. ІНТЕЛЕКТУАЛЬНА СИСТЕМА РЕКОМЕНДАЦІЙ ДЛЯ ПІДБОРУ ПАРТНЕРІВ І ВИКОНАВЦІВ

6.1. Розробка системи рекомендацій:

У сучасному бізнес-середовищі ефективний підбір партнерів та виконавців є критично важливим етапом реалізації будь-якого проєкту. Неправильне рішення на цьому етапі може призвести до затримок, перевитрати ресурсів або навіть провалу завдання. В умовах постійного зростання обсягів інформації, кількості учасників ринку та проєктної складності традиційні ручні методи пошуку вже не справляються з вимогами до швидкості та точності вибору.

Основні цілі впровадження інтелектуальної системи рекомендацій у вебзастосунку:

- **Автоматизація підбору:** зменшення навантаження на менеджерів проєктів при розподілі задач.
- **Підвищення якості вибору:** використання даних і моделей для визначення найбільш підходящих кандидатів.
- **Прозорість і персоналізація:** адаптація рекомендацій під конкретного користувача або команду з урахуванням їхніх уподобань, досвіду, історії взаємодії.
- **Оптимізація часу прийняття рішень:** миттєвий аналіз великих масивів даних для генерації актуальних пропозицій.

Ключові принципи роботи системи рекомендацій:

- **Збір і обробка даних про користувачів:** аналітика профілів, історії виконаних проєктів, оцінок, коментарів, навичок тощо.
- **Категоризація проєктів і виконавців:** визначення напрямків діяльності, рівня складності, необхідних компетенцій.
- **Визначення відповідностей:** застосування алгоритмів (наприклад, колаборативного фільтрування чи контентного аналізу) для виявлення найбільш релевантних варіантів.

- **Адаптація результатів:** навчання системи на основі зворотного зв'язку користувача (підтвердження або відхилення рекомендацій).

6.2. Алгоритми для системи рекомендацій

Рекомендаційна система є ядром інтелектуального механізму підбору партнерів і виконавців, і саме вибір відповідного алгоритму має ключове значення для якості рекомендацій. У цьому підрозділі розглянемо основні типи алгоритмів, що використовуються в сучасних системах, їх переваги, недоліки та особливості застосування в контексті вебплатформи для управління проєктами.

Колаборативна фільтрація (Collaborative Filtering)

Цей алгоритм ґрунтується на поведінці користувачів — виборах, оцінках, відгуках, частоті співпраці тощо. Основна ідея полягає в тому, що користувачі з подібними вподобаннями, як правило, надають перевагу подібним виконавцям або партнерам.

- **User-based CF:** шукає користувачів зі схожими діями й вподобаннями, та пропонує те, що сподобалося їм.
- **Item-based CF:** виявляє схожі об'єкти (наприклад, виконавців), ґрунтуючись на тому, як інші користувачі оцінювали їх.

Переваги:

- Автоматичне виявлення схожих інтересів без знання внутрішніх характеристик об'єкта.
- Добре масштабується у великих мережах користувачів.

Недоліки:

- Проблема холодного старту: важко рекомендувати нових користувачів або об'єкти.
- Залежність від великої кількості взаємодій і оцінок.

Контентно-орієнтовані підходи (Content-based Filtering)

Цей підхід аналізує атрибути об'єктів (наприклад, навички, досвід, типи проєктів) та шукає відповідність між цими атрибутами й профілем користувача. Профіль формується на основі попередніх виборів і активностей.

Приклад застосування:

- Користувач, який часто працював із партнерами, що мають досвід у сфері ІТ, отримуватиме рекомендації інших фахівців із цієї сфери.

Переваги:

- Працює навіть за відсутності активностей інших користувачів.
- Прозорість і пояснюваність результатів.

Недоліки:

- Обмежена здатність виявляти нові інтереси користувача.
- Залежність від повноти та якості опису об'єктів.

6.3. Моделі машинного навчання для оптимізації підбору партнерів

Впровадження моделей машинного навчання (ML) в систему рекомендацій дозволяє досягти значно вищого рівня точності у прогнозуванні відповідностей між користувачами, партнерами та виконавцями. На відміну від класичних алгоритмів, ML-моделі здатні виявляти складні, нелінійні залежності в даних, враховуючи багатовимірний контекст і динамічну природу взаємодій у системі.

Основні підходи, що застосовуються:

Глибока матрична факторизація (Deep Matrix Factorization)

Цей підхід є розширенням традиційної факторизації матриць і дозволяє враховувати не лише взаємодії між користувачами та об'єктами, але й додаткову інформацію (контекст, метадані, профілі користувачів).

- Кожен користувач і партнер представляється у вигляді векторів ознак.
- Нейронна мережа навчається прогнозувати рейтинг або ймовірність взаємодії на основі цих векторів.

Переваги:

- Висока точність навіть при наявності складних даних.
- Можливість додавати сторонні ознаки (наприклад, геолокацію, галузь, кількість завершених проєктів).

Нейронні мережі (Neural Networks)

Використовуються як багат шарові перцептрони або спеціалізовані архітектури (наприклад, Deep & Cross Network, Neural Collaborative Filtering).

- Мережі навчаються на основі історичних взаємодій користувачів.
- Можуть поєднувати ознаки з різних джерел — профіль, попередні оцінки, активність, поведінкові патерни.

Переваги:

- Потужне моделювання складних нелінійних залежностей.
- Підтримка багатофакторного аналізу (наприклад, одночасно враховуються досвід, відгуки, тематика проєктів тощо).

Decision Trees / Random Forest / Gradient Boosting

Методи на основі дерев рішень добре працюють з табличними даними та дають прозорі результати.

- Можуть бути використані як додатковий шар у гібридних моделях.
- Добре інтерпретуються, підходять для початкового етапу аналізу.

Навчання моделей

Для побудови ефективних моделей необхідно мати якісний набір даних:

- Профілі користувачів і партнерів (освіта, досвід, напрямки діяльності тощо).
- Історію співпраці (кількість проєктів, результати, оцінки).
- Поведінкові метрики (активність, відгуки, перегляди профілів).

Навчання може відбуватися як у режимі supervised (наприклад, на основі історичних оцінок), так і reinforcement learning (де система сама вчиться на основі відгуку користувачів після рекомендацій).

Результати впровадження

Застосування ML-моделей у системі дозволяє:

- Підвищити релевантність рекомендованих кандидатів.
- Зменшити середній час на пошук партнера або виконавця.
- Адаптувати рекомендації до змін у поведінці та потребах користувачів.

Отже, моделі машинного навчання є невід’ємним інструментом сучасних систем рекомендацій. Вони дозволяють досягти високого рівня персоналізації, передбачати приховані зв’язки між учасниками системи та забезпечують масштабованість рішення при зростанні бази користувачів. У контексті платформи для співпраці це відкриває нові можливості для ефективного підбору партнерів і підвищення результативності проєктної діяльності.

6.4. Персоналізація рекомендацій на основі історії взаємодії

Персоналізація — один із найважливіших напрямів розвитку інтелектуальних рекомендаційних систем, що спрямований на покращення користувацького досвіду шляхом адаптації контенту під конкретні потреби й інтереси кожного користувача. У контексті підбору партнерів і виконавців це дозволяє не лише зменшити час на пошук відповідних кандидатів, а й підвищити ймовірність ефективної співпраці.

Основні принципи персоналізації

Аналіз історії взаємодій

Система фіксує всі взаємодії користувача з платформою:

- попередні вибори партнерів або виконавців;
- оцінки після завершення проєктів;
- час, проведений на сторінках певних профілів;
- частота запитів і видів завдань, що створюються;

- історія відгуків та коментарів.

Ці дані слугують основою для побудови персонального профілю вподобань користувача.

Побудова профілю користувача

На основі накопиченої інформації система створює вектор характеристик, який відображає:

- галузеві уподобання (ІТ, дизайн, маркетинг тощо),
- типи проєктів, до яких виявляв інтерес,
- рівень досвіду бажаних виконавців або партнерів,
- переважні моделі взаємодії (довгострокова співпраця, короткі задачі).

Цей профіль оновлюється в режимі реального часу, відображаючи актуальні зміни в поведінці користувача.

Динамічна адаптація рекомендацій

Персоналізована система не є статичною — вона адаптується до нових дій користувача:

- якщо користувач починає проявляти інтерес до іншої галузі — пропозиції змінюються відповідно;
- якщо співпраця з певним типом виконавців призвела до негативного результату — система це враховує в майбутньому.

Такий підхід дозволяє забезпечити максимальну релевантність рекомендацій у змінюваному контексті.

Приклад персоналізації

Якщо користувач неодноразово обирав дизайнерів з досвідом у UI/UX, працював над завданнями з мобільних застосунків і залишав позитивні оцінки після співпраці з фрилансерами з України — система, враховуючи ці патерни, буде пропонувати подібні профілі в майбутньому, навіть у нових категоріях проєктів.

6.5. Врахування зворотного зв'язку від користувачів

Зворотний зв'язок користувачів відіграє критичну роль у вдосконаленні системи рекомендацій, оскільки він дозволяє оцінити якість та релевантність запропонованих результатів. У рамках інтелектуальної системи підбору партнерів та виконавців цей компонент дає змогу підвищити точність та персоналізацію рекомендацій завдяки прямим і непрямим сигналам від користувача.

Прямий зворотний зв'язок — це явні дії користувача, такі як виставлення оцінок (зірочки, бали), написання коментарів, залишення відгуків після співпраці або використання кнопок на кшталт «сподобалося/не сподобалося». Ці дії дозволяють системі швидко отримати інформацію про те, наскільки успішною була взаємодія з конкретним партнером або виконавцем.

Непрямий (неявний) зворотний зв'язок охоплює поведінкові дані: перегляди профілів, кількість кліків, тривалість взаємодії, кількість завершених проєктів, прийняття чи відхилення рекомендацій, тощо. Аналіз таких патернів дозволяє системі виявляти приховані вподобання користувачів, навіть коли вони прямо їх не висловлюють.

Висновки до розділу

Персоналізація та врахування зворотного зв'язку є ключовими компонентами ефективної системи рекомендацій у вебзастосунку. Завдяки аналізу як прямого, так і непрямого зворотного зв'язку, система здатна адаптувати свої пропозиції до індивідуальних потреб і вподобань користувача, підвищуючи релевантність результатів та рівень задоволеності. Такий підхід дозволяє не лише покращити досвід користування платформою, а й сприяє формуванню стійких, продуктивних партнерств. У підсумку, впровадження інтелектуальних механізмів персоналізації на основі зворотного зв'язку значно підсилює цінність платформи для кожного окремого користувача.

7. ОЦІНКА ЕФЕКТИВНОСТІ ВИКОРИСТАННЯ ВЕБЗАСТОСУНКУ

7.1. Вступ

Для обґрунтованого впровадження та подальшого вдосконалення вебзастосунку необхідно провести всебічну оцінку його ефективності. Цей процес охоплює як технічні, так і користувацькі аспекти роботи системи(рисунок 7.1, 7.2)

Зокрема, розглядаються показники продуктивності, зручності використання, вплив на бізнес-процеси та загальне покращення взаємодії користувачів із системою. Ефективна оцінка дозволяє приймати зважені рішення щодо подальшого розвитку, масштабування або зміни функціональності вебзастосунку.

Для отримання об'єктивної картини ефективності використовуються методи опитувань, аналітика користувацької поведінки, аналіз логів, А/В тестування, а також порівняльний аналіз продуктивності до та після впровадження системи. У наступних підрозділах розглядатимуться конкретні інструменти та підходи до оцінки кожного з цих аспектів.

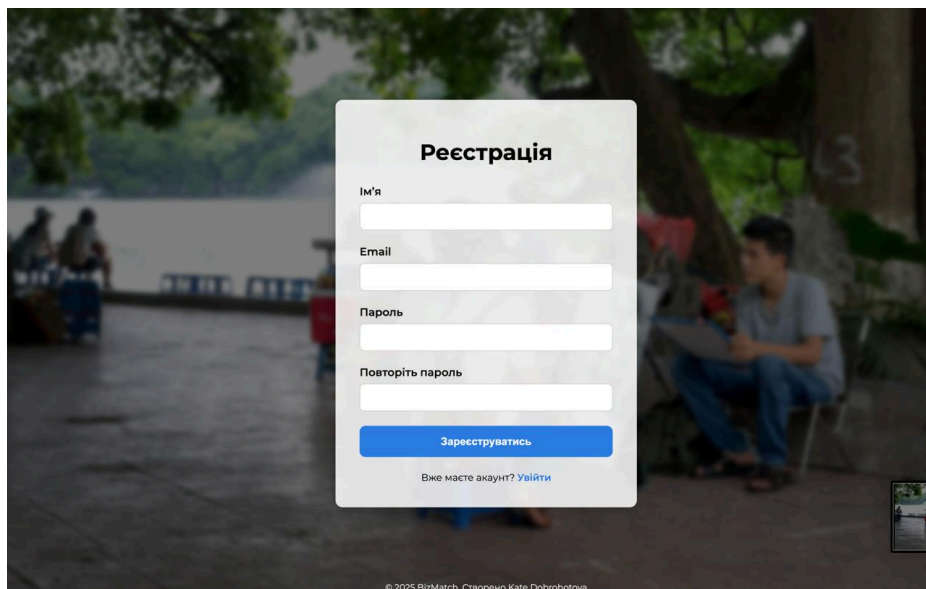


Рисунок 7.1 — Реєстрація

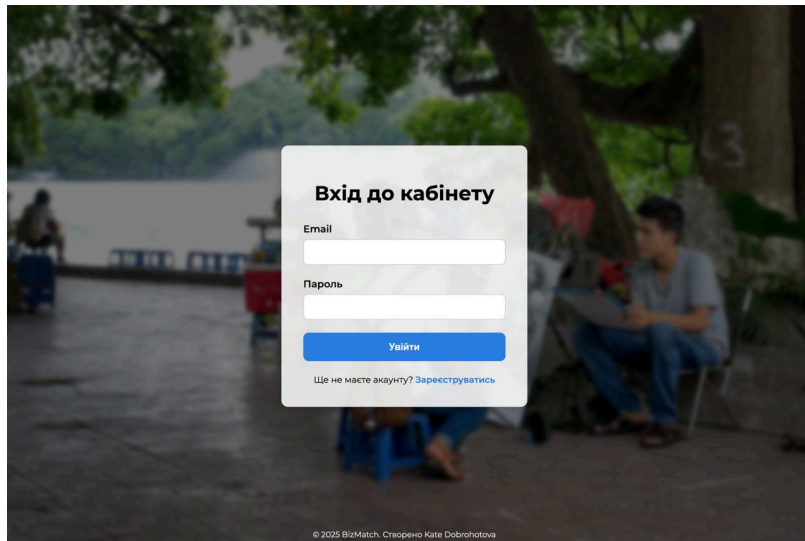


Рисунок 7.2 — Вхід до кабінету

7.2. Методи оцінки ефективності вебзастосунку

Методи оцінки ефективності вебзастосунку охоплюють як технічні аспекти його роботи, так і поведінкові показники користувачів, що дозволяє сформувати повну картину впливу системи на процеси в організації(рисунк 7.3).

Розглянемо ключові з них детальніше:

Технічні метрики

Ці показники дають змогу оцінити продуктивність та стабільність вебзастосунку. Сюди належать:

- *Час завантаження сторінок*: важливий для збереження уваги користувача — повільна робота знижує задоволеність і може призводити до відтоку аудиторії.
- *Кількість технічних помилок (HTTP-коди 4xx/5xx)*: свідчить про наявність критичних проблем у роботі системи.
- *Масштабованість*: як вебзастосунок реагує на збільшення кількості одночасних користувачів — важливо при зростанні навантаження.
- *Відповідність стандартам (наприклад, WCAG для доступності)*: демонструє, чи враховано потреби всіх груп користувачів.

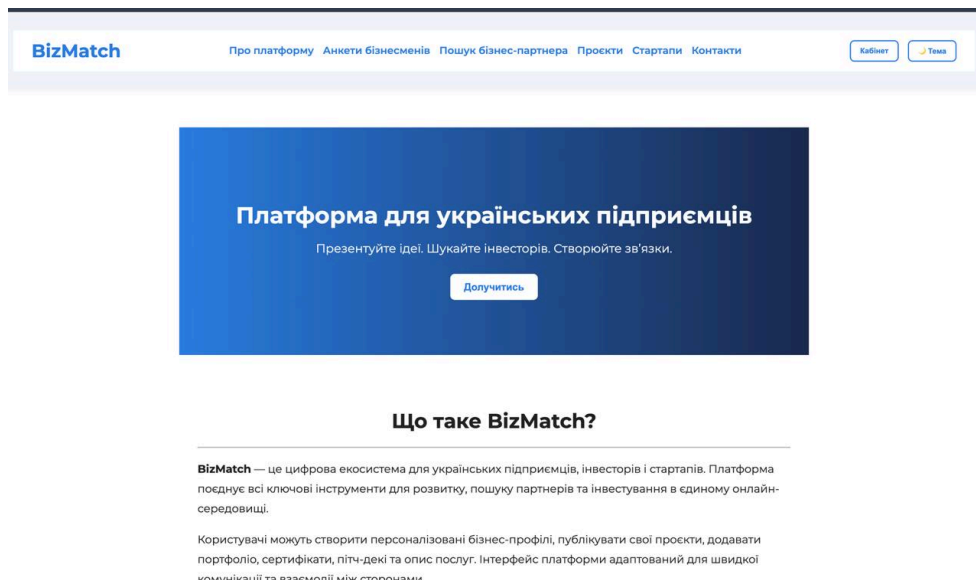


Рисунок 7.3 — Головний екран

Метрики користування

Ці показники відображають реальну взаємодію користувачів із системою:

- *Середня тривалість сесії*: дозволяє оцінити рівень залучення.
- *Кількість активних користувачів (DAU/WAU)*: свідчить про затребуваність застосунку.
- *Коефіцієнт повернення (retention rate)*: показує, чи повертаються користувачі після першого використання.
- *Кількість завершених дій*: наприклад, кількість створених, оновлених або завершених завдань, що демонструє практичну користь системи.

Аналіз поведінки користувачів

За допомогою інструментів вебаналітики:

- *Google Analytics* — дозволяє відслідковувати трафік, канали залучення, показники відмов тощо.
- *Hotjar або Clarity* — візуалізують кліки, рухи миші, теплові карти та записи сесій користувачів, що допомагає виявити UX-проблеми.

Оцінка досягнення бізнес-цілей

Вебзастосунок створюється не лише як технічний продукт, а як засіб досягнення конкретних бізнес-результатів:

- *Покращення внутрішньої комунікації* — через спільну роботу над завданнями(рисунок 7.5)
- *Зменшення кількості помилок або затримок* — завдяки прозорості процесів.(рисунок 7.7)
- *Зростання задоволеності клієнтів або партнерів* — через покращення якості виконання проєктів(рисунок 7.4)
- *Оптимізація робочих процесів* — наприклад, автоматизація рутинних операцій(рисунок 7.6)

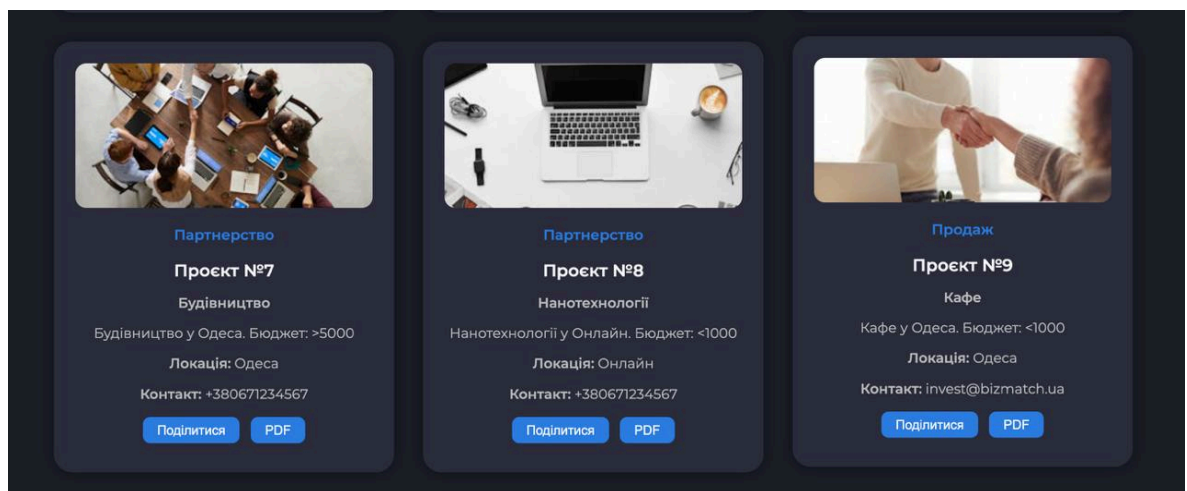


Рисунок 7.4 — Розділ проєктів

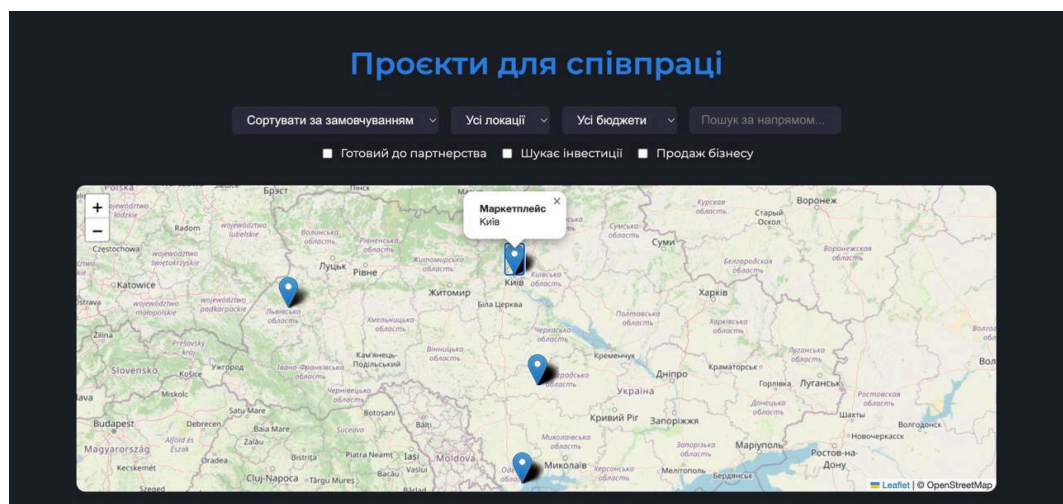


Рисунок 7.5 — Мапа проєктів для співпраці

Систематичне застосування вищезгаданих методів дозволяє здійснювати не лише оцінку поточної ефективності, а й формувати підґрунтя для прийняття рішень щодо розвитку вебзастосунку.

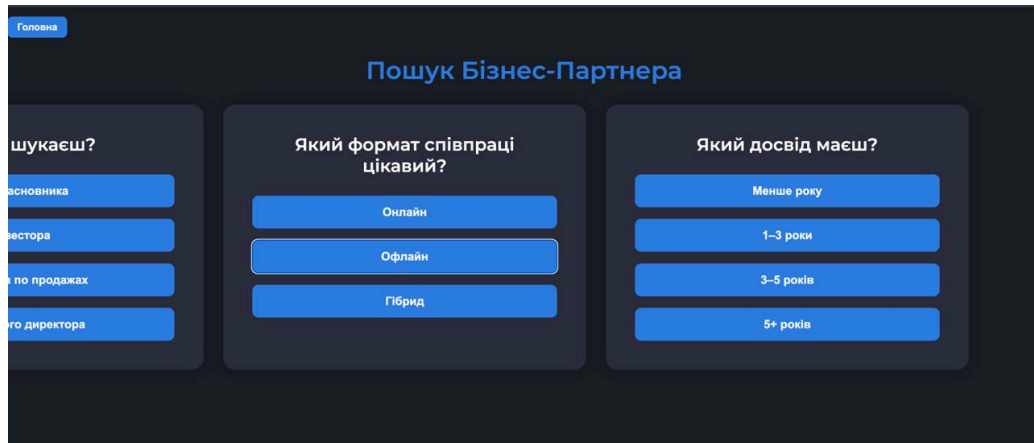


Рисунок 7.6 — Пошук бізнес-партнера

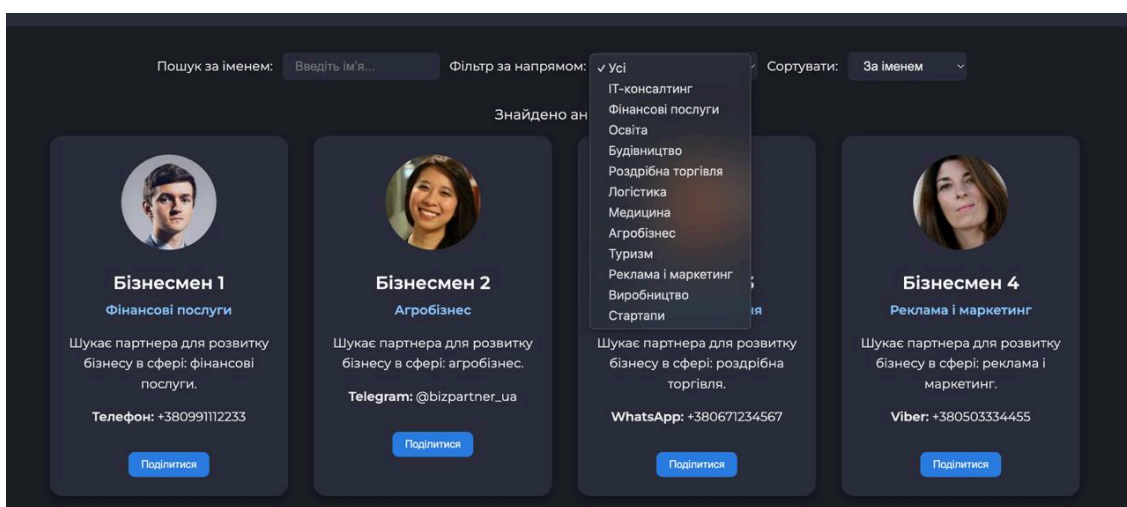


Рисунок 7.7 — Анкети бізнесменів

7.3. Оцінка зручності та функціональності через опитування користувачів

Оцінювання зручності використання та ефективності функціональності вебзастосунку є важливою складовою його розвитку. Для цього широко використовуються методи прямого зворотного зв'язку, які дозволяють виявити, наскільки застосунок відповідає очікуванням користувачів та які саме аспекти потребують поліпшення.

Онлайн-опитування

За допомогою інструментів на зразок *Google Forms*, *Typeform*, *Survio* тощо, користувачам пропонують заповнити анкети з питаннями на оцінку:

- інтуїтивності навігації;

- простоти виконання стандартних дій (створення/редагування завдань, взаємодія з іншими учасниками);
- загальної швидкодії застосунку;
- доступності функцій на різних пристроях.

Такий підхід дозволяє зібрати кількісні дані (оцінки за шкалою) і якісні коментарі, що вказують на конкретні труднощі користувачів.

Інтерв'ю та фокус-групи

Проведення структурованих інтерв'ю або обговорень з представниками різних категорій користувачів (адміністратори, менеджери проєктів, виконавці) допомагає:

- глибше зрозуміти потреби кожної ролі;
- виявити неочевидні проблеми, які не фіксуються аналітикою;
- отримати пропозиції щодо нової функціональності.

Індекси CSAT та NPS

- *Customer Satisfaction Score (CSAT)* — визначає рівень задоволеності користувачів після виконання певної дії або взаємодії з системою.
- *Net Promoter Score (NPS)* — показує готовність користувача рекомендувати продукт іншим, що відображає загальну лояльність до сервісу.

Високі показники цих індексів свідчать про позитивний користувацький досвід, тоді як низькі — сигналізують про необхідність термінових покращень.

Практичне застосування результатів

Зібрані дані є основою для:

- перегляду дизайну інтерфейсу (UI/UX);
- змін у логіці користувацьких сценаріїв;
- вдосконалення мобільної адаптації;
- підвищення доступності для користувачів із різними рівнями технічної підготовки.

Таким чином, систематичне опитування користувачів не лише підвищує рівень залучення, а й дозволяє оперативно адаптувати систему до реальних потреб аудиторії.

7.4. Оцінка впливу на ефективність роботи команд і партнерів

Оцінка впливу вебзастосунку на роботу команд і зовнішніх партнерів дає змогу визначити, наскільки система сприяє досягненню ключових цілей у проєктному середовищі. Це дозволяє не лише виявити практичну користь від впровадження, а й обґрунтувати подальші інвестиції в її розвиток.

Порівняння продуктивності до та після впровадження

Першочергово аналізується статистика виконання завдань у командах:

- загальна кількість виконаних завдань за визначений період;
- середній час на закриття одного завдання;
- рівень дотримання встановлених дедлайнів.

Аналіз командної взаємодії

Вивчається якість і результативність комунікації в середині команд:

- наявність та ефективність коментарів у завданнях;
- частота використання функцій сповіщень і згадок;
- зменшення кількості конфліктів через дублювання зусиль або відсутність інформації;
- покращення узгодженості дій між учасниками проєкту.

Оцінка впливу на взаємодію з партнерами

Для зовнішніх співпрацівників і партнерських команд аналізуються такі аспекти:

- зменшення часу на погодження завдань і призначення виконавців;
- швидкість старту спільних робіт;
- зростання кількості завершених проєктів за участю партнерів;
- покращення загальної комунікації з зовнішніми учасниками.

У підсумку, регулярний аналіз зазначених показників дозволяє оцінити не лише внутрішню ефективність команд, а й здатність системи підтримувати зовнішні робочі процеси на високому рівні.

7.5. Порівняння результатів до та після використання вебсервісу

Проведення порівняльного аналізу до та після впровадження вебзастосунку дозволяє наочно оцінити ефективність його використання. Такий аналіз охоплює як кількісні, так і якісні показники, які безпосередньо впливають на продуктивність команд, рівень організації робочих процесів та загальне користувацьке задоволення.

До впровадження вебзастосунку:

- **Відсутність централізації:** робота з завданнями велася у розрізнених середовищах (таблиці, листування, месенджери), що ускладнювало контроль та управління.
- **Низька прозорість процесів:** важко було відстежити статус виконання завдань, відповідальних осіб та терміни.
- **Часті затримки:** дедлайни не дотримувалися через відсутність чіткої координації та нагадувань.
- **Слабка комунікація:** учасники команди не мали єдиного простору для обговорення завдань, що призводило до дублювання роботи або втрати важливої інформації.

Після впровадження вебзастосунку:

- **Поліпшене планування:** завдяки чіткій структурі завдань, календарю та сповіщенням вдалося краще організувати процеси.
- **Зниження кількості пропущених дедлайнів:** автоматичні нагадування, чітка відповідальність і моніторинг допомагають вчасно завершувати задачі.
- **Розмежування ролей та відповідальності:** чітка рольова модель забезпечила контроль доступу і відповідальність за окремі дії.
- **Підвищення задоволеності користувачів:** інтерфейс став зручним, інформація — доступною, а загальний досвід користування — позитивним.

Висновки до розділу

Порівняльний аналіз до та після впровадження вебзастосунку переконливо демонструє його позитивний вплив на організацію командної роботи. Зменшення хаотичності у виконанні завдань, покращення прозорості процесів, своєчасне завершення проєктів та зростання рівня користувацького задоволення свідчать про успішність розробленого рішення. Завдяки централізації інформації, зручному інтерфейсу та ефективним інструментам комунікації, вебзастосунок значно покращує якість управління проєктами. Аналіз результатів також дозволяє виявити перспективи подальшого вдосконалення платформи, зокрема в напрямку масштабування, адаптації до різних сценаріїв використання та розширення функціональних можливостей.

ВИСНОВКИ

У ході дослідження було поставлено мету — створення вебсервісу, який об'єднує функціонал пошуку партнерів для спільної роботи та інструменти управління завданнями з метою підвищення ефективності командної взаємодії.

На початковому етапі проведено детальний аналіз існуючих рішень, таких як LinkedIn, Upwork, Asana, Trello та Jira. Результати показали, що жодна з наявних платформ не надає комплексного підходу: сервіси для пошуку партнерів не включають засобів для керування завданнями, тоді як таск-менеджери не мають функцій для формування команд або знаходження нових учасників проєктів.

У межах реалізації проєкту було створено вебзастосунок, який об'єднує ці два напрями. Розроблено інтелектуальну систему персоналізованих рекомендацій, реалізовано повноцінний таск-менеджер, а також створено інтерфейс, що дозволяє здійснювати повний цикл командної співпраці — від пошуку партнерів до виконання конкретних завдань.

Результати впровадження сервісу засвідчили його ефективність: користувачі відзначили зручність використання, високу релевантність рекомендацій, покращення в комунікації та організації роботи. Об'єднання ключових функцій в одному середовищі дозволило зменшити фрагментацію робочих процесів, підвищити продуктивність команд та знизити витрати часу на координування дій.

Таким чином, мету дослідження досягнуто: створено інноваційне рішення, яке усуває наявний розрив між системами професійного нетворкінгу та інструментами управління проєктами, забезпечуючи ефективну, зручну та прозору співпрацю.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сабадаш В. В. Технології розробки програмного забезпечення / В. В. Сабадаш. – Київ : КНЕУ, 2020. – 288 с.
2. Федоренко І. О. Інженерія програмного забезпечення / І. О. Федоренко. – Львів : ЛНУ, 2019. – 260 с.
3. Коваленко С. Ю. Побудова SaaS-платформ: технології та архітектура / С. Ю. Коваленко. – Дніпро : НМетАУ, 2021. – 180 с.
4. Гресько Т. П. Веб-програмування: PHP, MySQL, JS / Т. П. Гресько. – Харків : ХНУРЕ, 2022. – 234 с.
5. Суханова О. В. Електронна комерція на ринку автомобілів / О. В. Суханова // Економіка і суспільство. – 2020. – № 22. – С. 98–103.
6. Жуковський Р. П. Платформи онлайн-продажу автомобілів: функціональність і виклики / Р. П. Жуковський // Бізнес-Інформ. – 2021. – № 6. – С. 150–156.
7. Чернишова І. В. Автомобільний бізнес в умовах цифровізації / І. В. Чернишова // Маркетинг і цифрова трансформація. – 2022. – № 3. – С. 74–80.
8. Білик А. М. Створення MVP для стартапу: підходи та інструменти / А. М. Білик // ІТ менеджмент. – 2021. – № 4. – С. 45–51

ДОДАТКИ

ДОДАТОК А ЗВІТ ПОДІБНОСТІ



Дата звіту 6/17/2025
Дата редагування 6/17/2025

Документ прийнятий

Звіт подібності

метадані

Назва організації
National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute
Заголовок
ІП-12_Доброхотова_ПЗ
Автор
Науковий керівник / Експерт
ІП-12_ДоброхотоваЛісовиченко О.І.
підрозділ
ФІОТ, К-а інформатики та програмної інженерії

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



10

Довжина фрази для коефіцієнта подібності 2

16122

Кількість слів

129592

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навісний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		0
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		9

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Копію тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Копію тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	https://worksection.com/ua/blog/task-managers-for-every-team.html	53	0.33 %
2	https://worksection.com/ua/blog/task-managers-for-every-team.html	38	0.24 %
3	https://ir.kneu.edu.ua/bitstreams/2cc6e594-789f-4e92-836f-e02652059005/download	16	0.10 %
4	https://openarchive.nure.ua/bitstreams/f509e0c7-b2e1-4d2c-abf5-01e53d9e70b2/download	13	0.08 %

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

Текст програми
КПІ.ІП-1208.045440.03.12

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

Київ – 2025

Посилання на репозиторій з повним текстом програмного коду

<https://github.com/Kaaate/BuisnessMatchDiploma>

Файл index.html

Реалізація функціональної задачі “Пошук партнера з інтеграцією тасок”

```
<!DOCTYPE html>
<html lang="uk">
<head>
<meta charset="UTF-8">
<title>Адмін-панель | BizMatch</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="https://cdn.jsdelivr.net/npm/chart.js"></script> <style>
body {
  background-color: #1c1e26;
  color: #f0f0f0;
  font-family: 'Montserrat', sans-serif;
  margin: 0;
  padding: 2rem;
}
.login-box, .dashboard {
  max-width: 1000px;
  margin: 2rem auto;
  background: #2a2d3c;
  padding: 2rem;
  border-radius: 16px;
  box-shadow: 0 0 20px rgba(0,0,0,0.4);
}
h2 {
  text-align: center;
  color: #2a7de1;
}
input[type="password"] {
  width: 100%;
  padding: 0.8rem;
  margin-top: 1rem;
  border-radius: 8px;
  border: none;
  background: #1c1e26;
  color: white;
}
</style>
</head>
<body>
<div class="login-box" id="loginBox">
  <h2>Увійдіть у Dashboard</h2>
  <input type="password" id="password" placeholder="Введіть пароль">
  <button onclick="checkLogin()">Увійти</button>
</div>
<div class="dashboard" id="dashboard" style="display: none;">
  <h2>Статистика сайту</h2>
```

```

<div class="stat">Кількість зареєстрованих стартапів: <strong id="startupCount">0</strong></div>
<div class="stat">Кількість переглядів за день: <strong id="views">0</strong></div>
<div class="stat">Кількість анкет бізнесменів: <strong id="profiles">0</strong></div>
<div class="stat">Загальна сума запитаних інвестицій: <strong id="investment">0</strong></div>
<button onclick="downloadPDF()">Експортувати в PDF</button>
<div class="charts">
  <canvas id="barChart"></canvas>
  <canvas id="pieChart"></canvas>
  <canvas id="lineChart"></canvas>
  <canvas id="radarChart"></canvas>
</div>
<div class="status">Стан сайту: Стабільний і працює нормально </div>
</div>
<script>
const stats = {
  startups: 48,
  views: 1540,
  profiles: 50,
  investment: 1200000
};

function checkLogin() {
  const pass = document.getElementById('password').value;
  if (pass === 'admin123') {
    document.getElementById('loginBox').style.display = 'none';
    document.getElementById('dashboard').style.display = 'block';
    loadStats();
    renderCharts();
  } else {
    alert('Невірний пароль');
  }
}

function loadStats() {
  document.getElementById('startupCount').textContent = stats.startups;
  document.getElementById('views').textContent = stats.views;
  document.getElementById('profiles').textContent = stats.profiles;
  document.getElementById('investment').textContent = '$' + stats.investment.toLocaleString();
}

function renderCharts() {
  new Chart(barChart, {
    type: 'bar',
    data: {
      labels: ['Стартапи', 'Перегляди', 'Анкети', 'Інвестиції ($K)'],
      datasets: [{
        label: 'Значення',
        data: [stats.startups, stats.views, stats.profiles, stats.investment / 1000],
        backgroundColor: '#2a7de1'
      }]
    }
  });

  new Chart(pieChart, {
    type: 'doughnut',
    data: {
      labels: ['ІТ', 'Фінтех', 'Освіта', 'Медицина'],

```

```

    datasets: [{
      data: [14, 12, 10, 12],
      backgroundColor: ['#4dc9f6', '#f67019', '#f53794', '#537bc4']
    }]
  }
});

new Chart(lineChart, {
  type: 'line',
  data: {
    labels: ['Пн', 'Вт', 'Ср', 'Чт', 'Пт', 'Сб', 'Нд'],
    datasets: [{
      label: 'Перегляди за тиждень',
      data: [200, 300, 450, 600, 700, 650, 500],
      fill: false,
      borderColor: '#2a7de1'
    }]
  }
});

new Chart(radarChart, {
  type: 'radar',
  data: {
    labels: ['UX', 'UI', 'Навантаження', 'SEO', 'Безпека', 'Доступність'],
    datasets: [{
      label: 'Оцінка системи',
      data: [90, 80, 85, 70, 95, 88],
      backgroundColor: 'rgba(42,125,225,0.2)',
      borderColor: '#2a7de1'
    }]
  }
});
}

function downloadPDF() {
  const { jsPDF } = window.jspdf;
  const doc = new jsPDF();
  doc.text("Звіт по сайту BizMatch", 10, 10);
  doc.text(`Стартів: ${stats.startups}`, 10, 20);
  doc.text(`Переглядів: ${stats.views}`, 10, 30);
  doc.text(`Анкет бізнесменів: ${stats.profiles}`, 10, 40);
  doc.text(`Інвестиції: $$${stats.investment}`, 10, 50);
  doc.text("Стан: Стабільний ", 10, 60);
  doc.save("bizmatch-dashboard.pdf");
}
</script>
</body>
</html>

```

Файл style.css

Реалізація “Стилів пошуку партнерів”

```
.NAVBAR {  
  BACKGROUND-COLOR: var(--PRIMARY);  
  PADDING: 1rem 2rem;  
  DISPLAY: flex;  
  JUSTIFY-CONTENT: space-between;  
  ALIGN-ITEMS: center;  
}
```

```
.NAVBAR A {  
  COLOR: white;  
  TEXT-DECORATION: none;  
  MARGIN-LEFT: 1.5rem;  
  FONT-WEIGHT: 500;  
}
```

```
.NAVBAR A:HOVER {  
  TEXT-DECORATION: underline;  
}
```

```
.ALERT {  
  PADDING: 1rem;  
  MARGIN-BOTTOM: 1rem;  
  BORDER-RADIUS: 4px;  
  FONT-WEIGHT: bold;  
}
```

```
.ALERT.SUCCESS {  
  BACKGROUND-COLOR: #d4edda;  
  COLOR: #155724;  
}
```

```
.ALERT.ERROR {  
  BACKGROUND-COLOR: #f8d7da;  
  COLOR: #721c24;  
}
```

```
.ALERT.INFO {  
  BACKGROUND-COLOR: #cce5ff;  
  COLOR: #004085;  
}
```

```
.PROGRESS {  
  BACKGROUND-COLOR: #eee;
```

```

BORDER-RADIUS: 20px;

OVERFLOW: HIDDEN;

HEIGHT: 20px;

MARGIN-BOTTOM: 1rem;
}

.PROGRESS-BAR {
  HEIGHT: 100%;

  BACKGROUND-COLOR: var(--PRIMARY);

  WIDTH: 0%;

  TRANSITION: WIDTH 0.5s EASE;
}

.TABLE {
  WIDTH: 100%;

  BORDER-COLLAPSE: COLLAPSE;

  MARGIN-TOP: 1rem;
}


  <p>Інтеграція з месенджерами та соціальними мережами.</p>
</div>
<div class="card">
<section class="fade-in" id="contact">
  <h2 class="section-title">Зв'язатися з нами</h2>
  <hr>
  <div style="max-width: 600px; margin: auto;">
    <form onsubmit="alert('Дякуємо за повідомлення!'); return false;">
      <label>Ім'я:<br><input type="text" required style="width:100%;padding:0.5rem;margin:0.5rem 0;"></label>
      <label>Email:<br><input type="email" required style="width:100%;padding:0.5rem;margin:0.5rem 0;"></label>
      <label>Повідомлення:<br><textarea rows="5" required style="width:100%;padding:0.5rem;margin:0.5rem 0;"></label>
      <button type="submit" style="padding: 0.7rem 1.5rem; background: #2a7de1; color: white; border: none; border-radius: 6px; font-weight:
600;">Надіслати</button>
    </form>
  </div>
</section>

```

—

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

Програма та методика тестування
КПІ.ІП-1208.045440.04.51

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

Київ – 2025

ЗМІСТ

1 ОБ'ЄКТ ВИПРОБУВАНЬ.....	106
2 МЕТА ТЕСТУВАННЯ.....	107
3 МЕТОДИ ТЕСТУВАННЯ.....	108
4 ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ.....	109

1 ОБ'ЄКТ ВИПРОБУВАНЬ

Об'єктом випробування є розроблене програмне забезпечення, призначене для збереження та валідації електронних документів з використанням смарт-контрактів у блокчейн-мережі.

Склад програмного забезпечення включає такі основні компоненти:

- вебклієнт, реалізований за допомогою фреймворку Angular, що забезпечує взаємодію користувача із системою;
- бекенд-сервіс, розроблений на основі платформи Spring Boot, який відповідає за бізнес-логіку та API;
- смарт-контракт, розгорнутий у тестовій мережі Ethereum (EVM-сумісній), що забезпечує фіксацію й перевірку даних у блокчейні;
- інфраструктурні компоненти, зокрема:
 - PostgreSQL — для зберігання метаданих електронних документів;
 - Apache Kafka — для обміну повідомленнями між мікросервісами.

Середовище тестування: програмне забезпечення перевірялося у середовищі Docker, з використанням операційної системи Ubuntu 22.04, а також у веббраузерах Google Chrome та Mozilla Firefox. Додатково було протестовано працездатність системи на платформі Windows 10 для перевірки кросплатформеності.

2 МЕТА ТЕСТУВАННЯ

Метою тестування є всебічна перевірка працездатності, стабільності, зручності та безпеки розробленого програмного забезпечення. Зокрема, тестування передбачає досягнення таких завдань:

- перевірка відповідності функціональності системи встановленим вимогам;
- оцінка коректності збереження та валідації електронних документів за допомогою смарт-контрактів;
- тестування взаємодії між усіма основними компонентами системи: інтерфейсом користувача (фронтендом), серверною частиною (бекендом), блокчейн-компонентом, базою даних і системою обміну повідомленнями Kafka;
- перевірка сумісності вебзастосунку з актуальними версіями популярних браузерів (Google Chrome, Mozilla Firefox, Opera);
- тестування роботи програмного забезпечення в різних операційних системах (Windows, Linux);
- виявлення логічних помилок, збоїв, візуальних недоліків або проблем користувацького досвіду (UX) з метою їх подальшого усунення;
- перевірка зручності, інтуїтивності та доступності графічного інтерфейсу користувача;
- перевірка стабільності роботи при одночасній взаємодії користувачів із різними ролями (звичайний користувач, модератор, адміністратор);
- тестування обробки помилкових ситуацій (наприклад, некоректне завантаження документа, відсутність відповіді з блокчейну тощо);

3 МЕТОДИ ТЕСТУВАННЯ

- Типи та методи тестування програмного забезпечення
- У відповідності до концепції піраміди тестування для перевірки якості розробленого програмного забезпечення було застосовано декілька рівнів тестування:
- Юніт-тестування (Unit testing) — базовий рівень, що охоплює тестування окремих функціональних блоків. Зокрема, тестувалися: логіка хешування документа, перевірка ролей користувачів, обробка вхідних даних на рівні бекенду.
- Інтеграційне тестування (Integration testing) — перевірка взаємодії між основними компонентами: серверною частиною, базою даних PostgreSQL, брокером повідомлень Apache Kafka та смарт-контрактом, розгорнутим у блокчейн-мережі.
- Системне тестування (System testing) — комплексна перевірка роботи всієї системи в середовищі Docker із повною інфраструктурою. Включало типові користувацькі сценарії.
- UI/мануальне тестування (Manual/UI testing) — ручне тестування інтерфейсу користувача з акцентом на зручність використання, доступність елементів і відповідність очікуваній поведінці.
- End-to-End тестування (E2E testing) — імітація повного ланцюжка дій користувача: авторизація, завантаження документа, перевірка автентичності через блокчейн та отримання підтвердження.
- Також застосовувалися наступні методи тестування:
- Статичне тестування — передбачало аналіз вихідного коду, конфігураційних файлів і документації на відповідність специфікації, правилам кодування та архітектурним стандартам.
- Динамічне тестування — здійснювалося під час виконання програми: моделювалися різні сценарії роботи з системою з метою виявлення збоїв та неочікуваної поведінки.
- Функціональне тестування — охоплювало перевірку ключових функцій системи: автентифікація, завантаження документа, створення та збереження хешу, валідація через смарт-контракт, управління користувацькими ролями.

- Системне тестування — проводилося в повністю змодельованому середовищі (Docker Compose) з усіма активними сервісами.
- Мануальне тестування — здійснювалося без автоматизованих фреймворків, із використанням ручних тест-кейсів для перевірки логіки бізнес-процесів.
- Тестування «чорної скриньки» — передбачало оцінку поведінки системи без доступу до її внутрішнього коду. Наприклад, перевірка відповіді API при завантаженні невалідного документа.
- Тестування «білої скриньки» — базувалося на знанні внутрішньої реалізації: проводився аналіз роботи Kafka-обробників, функцій валідації, механізмів контролю доступу.
- Тестування «сірої скриньки» — поєднувало знання логіки системи з перевіркою зовнішньої поведінки. Наприклад, тестування механізму запису хешу у смарт-контракт і подальшої перевірки достовірності документа.

4 ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ

У процесі тестування програмного забезпечення було використано широкий набір інструментів, що охоплюють як стороннє програмне забезпечення, так і спеціалізовані утиліти:

- Стороннє програмне забезпечення:
 - Postman — для ручного тестування REST API-запитів до бекенд-сервісу;
 - Metamask — для взаємодії користувача зі смарт-контрактами в мережі Ethereum;
 - Ganache / Hardhat — для розгортання та локального тестування смарт-контрактів;
 - Docker / Docker Compose — для контейнеризації та розгортання всіх компонентів у тестовому середовищі;
 - Browser Developer Tools (Chrome, Firefox) — для аналізу поведінки клієнтської частини та моніторингу HTTP-запитів;
 - PgAdmin / DBeaver — для перевірки стану та вмісту бази даних PostgreSQL;
 - Kafka CLI Tools — для моніторингу обміну повідомленнями між мікросервісами через Apache Kafka.

Порядок проведення тестування

Тестування здійснювалося за узгодженим планом, який охоплював наступні етапи:

1. Статичний аналіз коду
Огляд вихідного коду на наявність помилок, невикористаних змінних, порушень архітектурних принципів та стандартів.
2. Динамічне тестування функціональності
Перевірка відповідності системи функціональним вимогам:

- авторизація та розмежування прав доступу;

- завантаження документів, збереження та валідація;
- запис і зчитування хешів через смарт-контракт;
- обробка подій за допомогою Kafka.

3. Функціональне тестування окремих компонентів

Оцінка коректності роботи кожного модуля:

- клієнтського інтерфейсу;
- серверної логіки;
- API-ендпоінтів;
- бази даних;
- механізмів взаємодії з блокчейном.

4. Системне тестування

Повна перевірка інтегрованого середовища з усіма активними компонентами (frontend + backend + blockchain + Kafka + PostgreSQL) у межах середовища Docker Compose.

5. Кросбраузерне тестування

Перевірка відображення і функціональності інтерфейсу в браузерах Google Chrome, Mozilla Firefox, Opera.

6. Кросплатформне тестування

Тестування стабільності роботи застосунку в операційних системах Windows 10 та Ubuntu 22.04.

7. Оцінка зручності використання (usability testing)

Аналіз інтерфейсу з точки зору кінцевого користувача — інтуїтивність навігації, логічна структура, доступність функціоналу.

8. Тестування граничних і помилкових випадків

Моделювання сценаріїв з некоректними або атиповими вхідними даними:

- завантаження великого файлу;
- використання невалідного формату;
- спроба повторного завантаження того самого документа.

9. Тестування відновлення після збоїв

Перевірка поведінки системи після відключення сервісів, перезапуску контейнерів або втрати підключення до блокчейн-мережі.

10. Тестування безпеки

Оцінка надійності контролю доступу до функцій системи, перевірка дотримання обмежень згідно з роллю користувача та недопущення несанкціонованих дій.

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

Керівництво користувача
КПІ.ІП-1208.045440.05.34

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

Київ – 2025

Зміст

1. ПРИЗНАЧЕННЯ ПЛАТФОРМИ.....	113
2. КАТЕГОРІЇ КОРИСТУВАЧІВ.....	113
3. ПОЧАТОК РОБОТИ З ПЛАТФОРМОЮ.....	113
4. НАВІГАЦІЯ ІНТЕРФЕЙСОМ.....	114
5. ОСНОВНІ ФУНКЦІЇ.....	114
6. РЕКОМЕНДАЦІЇ З ВИКОРИСТАННЯ.....	114
7. ТЕХНІЧНА ПІДТРИМКА.....	114

1. ПРИЗНАЧЕННЯ ПЛАТФОРМИ

Платформа “Бізнес Матч” розроблена для пошуку бізнес-партнерів, перегляду анкет підприємців та стартапів, а також для налагодження ділових зв’язків між учасниками малого та середнього бізнесу, інвесторами та представниками інноваційних команд. Інтерфейс доступний у веббраузері без необхідності встановлення.

2. КАТЕГОРІЇ КОРИСТУВАЧІВ

- **Гість** — має доступ до загальнодоступної інформації (перегляд профілів, стартапів, пошук).
- **Зареєстрований користувач** (функціонал реалізується на наступному етапі розробки) — створює профіль, додає стартапи, надсилає заявки.
- **Адміністратор** — керує вмістом, модеруючи профілі та проекти (описано окремо в керівництві адміністратора).

3. ПОЧАТОК РОБОТИ З ПЛАТФОРМОЮ

1. Відкрийте платформу у браузері за адресою:
<https://kaaate.github.io/BuisnessMatchDiploma>
2. Оберіть бажаний розділ:
 - **Анкет**и бізнесменів – [profiles.html](#)
 - **Пошук** партнера – [find_partner.html](#)
 - **Стартапи** – [startup_page.html](#)
 - **Проекти** – [projects.html](#)
3. Навігація доступна через головне меню або за прямими посиланнями.

4. НАВІГАЦІЯ ІНТЕРФЕЙСОМ

- Меню розміщене у верхній частині сторінки.
- Кнопки перемикають між сторінками без втрати контексту.
- Інтерактивні елементи виділені кольором при наведенні.
- Кнопки фільтрації (у плані реалізації) передбачають динамічне сортування учасників за напрямками.

5. ОСНОВНІ ФУНКЦІЇ ПЛАТФОРМИ

- **Огляд профілів бізнесменів** – перегляд короткої та розширеної інформації про учасників.
- **Пошук бізнес-партнерів** – перегляд відповідних пропозицій на основі галузі.
- **Перегляд стартапів** – картки із ключовою інформацією про ідеї, команду, етапи реалізації.
- **Презентація проєктів** – сторінка для перегляду активних або завершених ініціатив.

6. РЕКОМЕНДАЦІЇ З ВИКОРИСТАННЯ

- Для коректної роботи використовуйте сучасні браузері: Google Chrome, Firefox, Microsoft Edge.
- Платформа адаптована під мобільні пристрої, однак рекомендується настільний перегляд.
- Рекомендується регулярно оновлювати сторінки (Ctrl + R), щоб бачити актуальні зміни.

7. ТЕХНІЧНА ПІДТРИМКА

На етапі MVP технічна підтримка реалізується через GitHub (розділ Issues у репозиторії).

У майбутньому, з розширенням користувацької бази та виходом за межі технічної спільноти, **планується впровадження повноцінної системи зворотного зв'язку**, яка включатиме:

- **Інтегровану контактну форму** безпосередньо у вебінтерфейсі для швидкого надсилання повідомлень технічній команді;
- **Систему категоризації звернень** (баг, питання, пропозиція, скарга), що дозволить автоматизувати їх обробку;
- **Підтримку через email** або чат-бот на основі Telegram/WhatsApp для оперативного реагування;
- **Базу знань (Knowledge Base)** з відповідями на типові запитання та інструкціями для самостійного вирішення проблем;
- **Панель адміністратора підтримки**, через яку модератори зможуть швидко обробляти звернення, призначати відповідальних осіб і вести статистику.

У перспективі можливе підключення **системи HelpDesk** (наприклад, через Zendesk або Freshdesk) для централізації обслуговування користувачів, накопичення аналітики та підвищення якості технічної підтримки.

Загалом, концепція технічної підтримки передбачає **масштабованість і еволюційність**, — від спрощеної системи для MVP до багатofункціонального сервісу з урахуванням зростання навантаження, очікувань користувачів та комерційного масштабу проєкту.

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

**Керівництво адміністратора і програміста
КПІ.ІП-1208.045440.06.33**

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

Київ – 2025

Зміст

1. ПРИЗНАЧЕННЯ.....	117
2. ОБОВ’ЯЗКИ АДМІНІСТРАТОРА.....	117
3. ВИМОГИ ДО СЕРЕДОВИЩА.....	118
4. ОПИС АДМІНІСТРАТИВНОГО ФУНКЦІОНАЛУ.....	118
5. ПІДГОТОВКА ТА РОЗГОРТАННЯ ПЛАТФОРМИ.....	118
6. ПІДТРИМКА ТА ОНОВЛЕННЯ.....	119
7. ЗАХОДИ БЕЗПЕКИ.....	119

1. ПРИЗНАЧЕННЯ

Цей документ призначений для системного адміністратора вебплатформи “Бізнес Матч” та містить інструкції щодо налаштування, розгортання, моніторингу та обслуговування сервісу. Керівництво також охоплює вимоги до середовища, з яким працює платформа, та заходи з забезпечення її безперервної роботи.

2. ОБОВ'ЯЗКИ АДМІНІСТРАТОРА

- Контроль за працездатністю вебплатформи та відповідністю її цільовому функціоналу.
- Публікація оновлень на GitHub Pages або інший хостинг при переході до серверного рішення.
- Управління файлами HTML, CSS, JS/TS.
- Контроль доступу до репозиторію (налаштування прав користувачів).
- Забезпечення резервного копіювання даних (у разі переходу до динамічної моделі).
- Забезпечення актуальності бібліотек та компонентів (шрифтів, скриптів тощо).

3. ВИМОГИ ДО СЕРЕДОВИЩА

- Підтримка сучасного браузера (Chrome, Firefox, Edge).
- Git-репозиторій для керування версіями.
- Платформа для хостингу (GitHub Pages або альтернативна інфраструктура, якщо запроваджується бекенд).
- Node.js та npm (для сценаріїв збірки або міграції до фреймворків типу React).

4. ОПИС АДМІНІСТРАТИВНОГО ФУНКЦІОНАЛУ

- Додавання та оновлення файлів сайту через Git.
- Внесення змін у структуру сторінок, стилів або логіки взаємодії.
- Створення резервних копій та збереження стабільних релізів.
- Заміна контенту (зображень, описів, профілів стартапів тощо).

5. ПІДГОТОВКА ТА РОЗГОРТАННЯ ПЛАТФОРМИ

1. Клонування репозиторію:

```
git clone https://github.com/username/repo.git
```

2. Додавання нових сторінок, перевірка локально.

3. Коміт змін:

```
git add .
```

```
git commit -m "Update content"
```

```
git push
```

4. Перевірка оновлень на публічній сторінці GitHub Pages.

6. ПІДТРИМКА ТА ОНОВЛЕННЯ

- Регулярна перевірка працездатності посилань, адаптивності сторінок.
- Відслідковування оновлень зовнішніх бібліотек (Tailwind, FontAwesome тощо).
- Моніторинг відгуків користувачів і вчасне внесення змін на основі їхніх потреб.

7. ЗАХОДИ БЕЗПЕКИ

- Використання SSH-ключів для доступу до репозиторію.

Такі заходи забезпечують цілісність коду, стабільність середовища розгортання та безпеку даних користувачів на всіх рівнях — від етапу розробки до продуктивного використання. Вони закладають основу для подальшого розширення системи відповідно до сучасних вимог інформаційної безпеки.

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”
Завідувач кафедри
Едуард ЖАРІКОВ
“ _____ ” _____ 2025 р.

**ВЕБСЕРВІС ДЛЯ ПОШУКУ ПАРТНЕРІВ З ІНТЕГРАЦІЄЮ
TASK-МЕНЕДЖМЕНТУ**

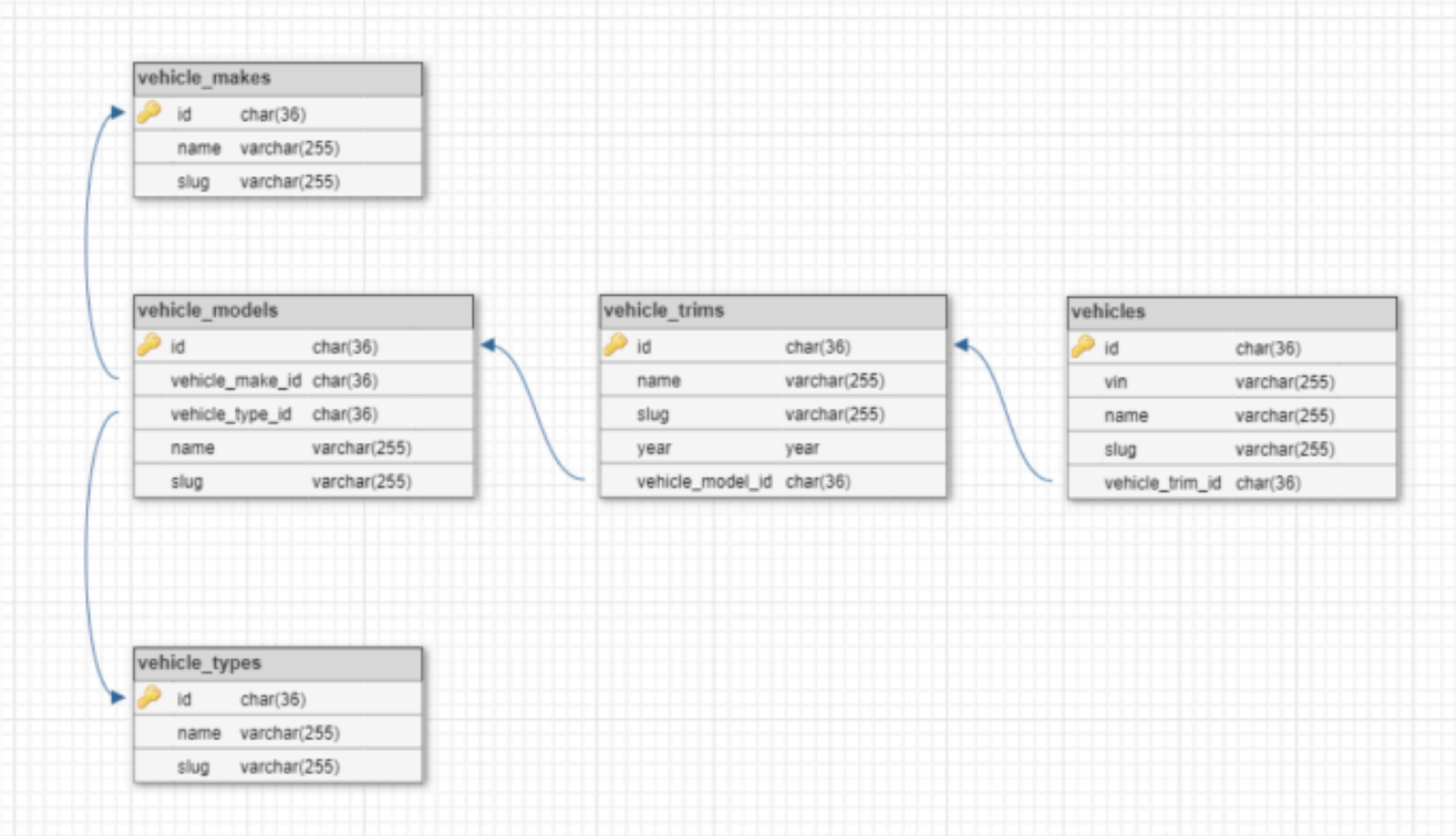
Графічні матеріали
КПІ.ІП-1208..045440.07.99

“ПОГОДЖЕНО”
_____ Олег ЛІСОВИЧЕНКО

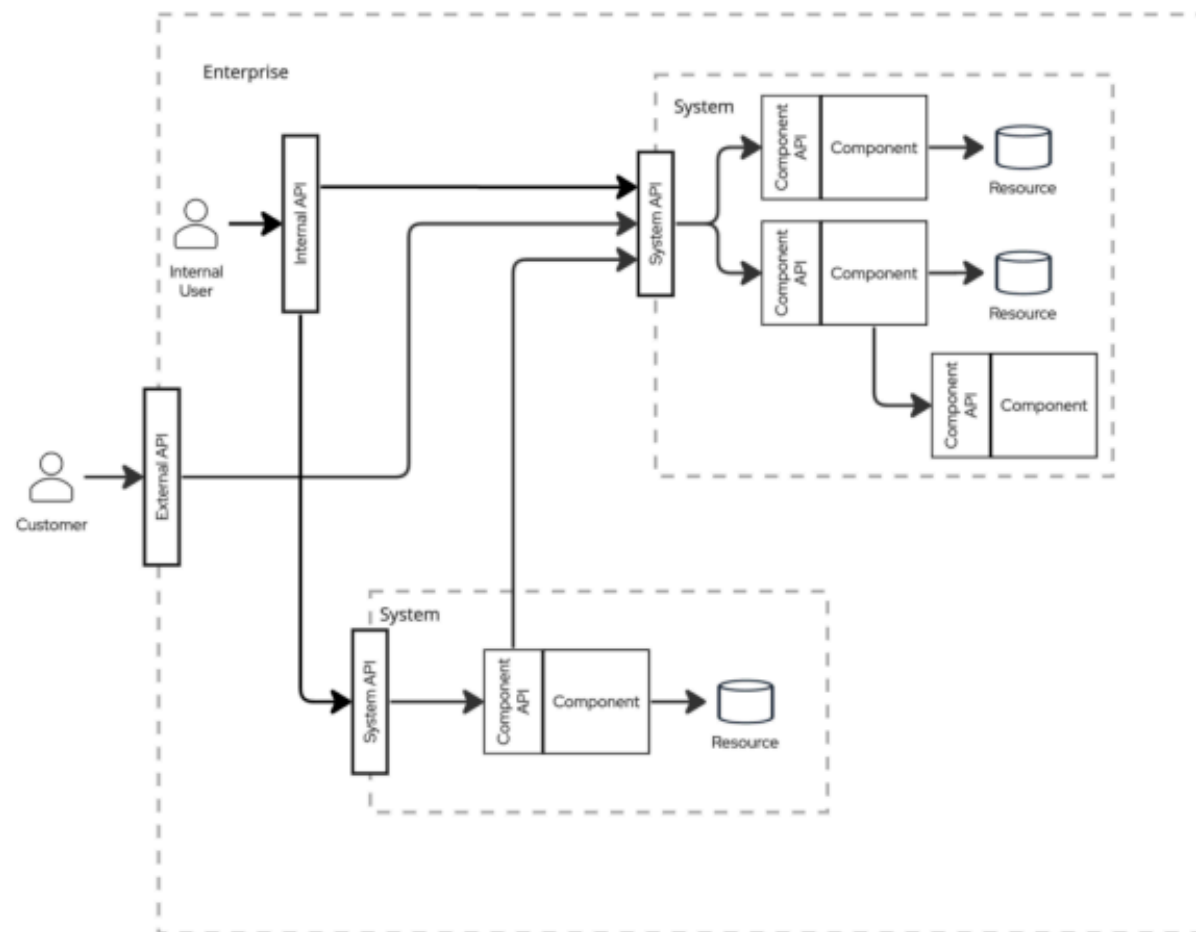
Нормоконтроль:
_____ Максим
ГОЛОВЧЕНКО

Виконавець:
_____ Катерина ДОБРОХОТОВА

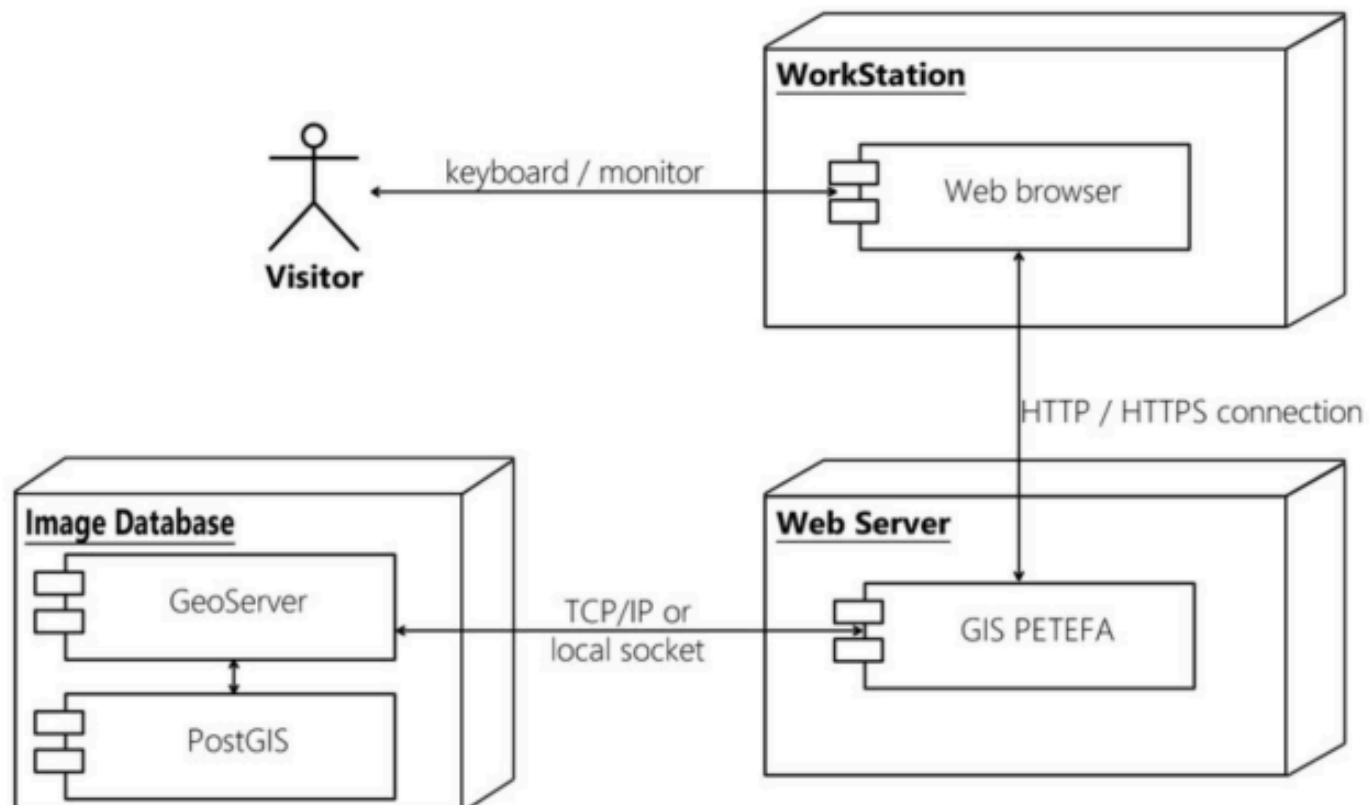
Київ – 2025



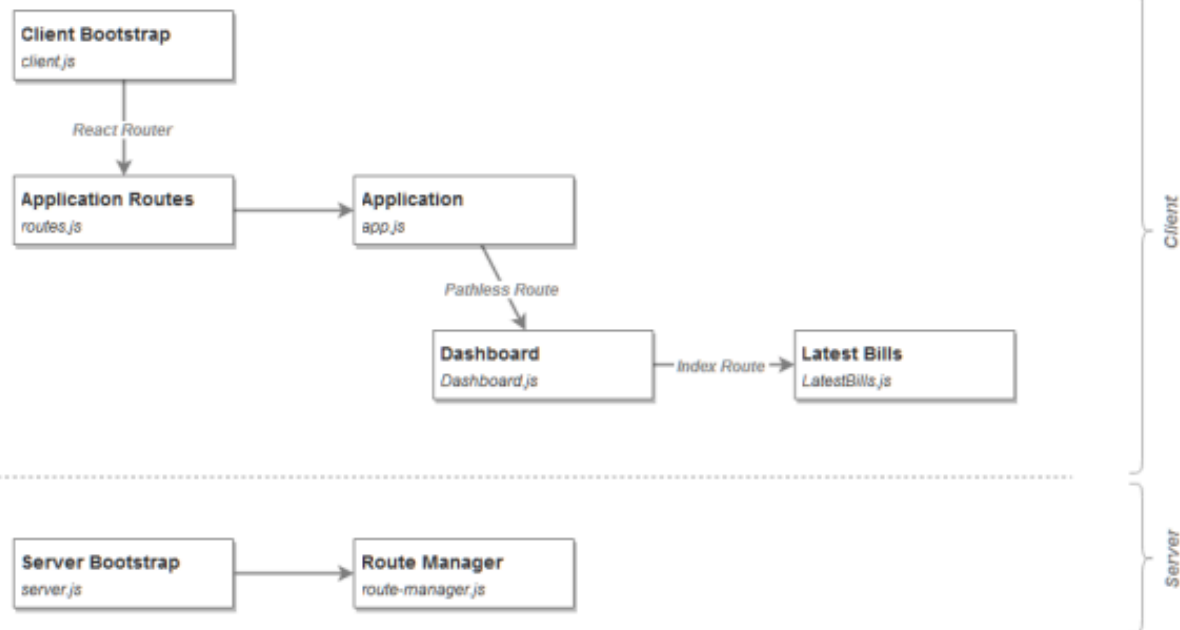
						КПІ.ІП-1208..045440.07.99.СБД			
						Вебсервіс для пошуку партнерів з інтеграцією таск- менеджменту	Літ.	Маса	Мірило
зм.	Лист	докум.	Підпис	Дата					
Розроб.	Добролюбова К.А.								
Перев.	Лисовиченко О.І.								
						Схема бази даних	Листів 1		
Н. контр.							КПІ ім. Ігоря Сікорського		
Затв.									



						КПІ.ІП-1208..045440.07.99.CCM					
						Вебсервіс для пошуку партнерів з інтеграцією таск- менеджменту	Літ.		Маса	Мірило	
Зм.	Лист	докум.	Підпис	Дата							
Розроб.	Добролюбова К.А.										
Перев.	Лисовиченко О.І.										
							Лист 1		Листів 1		
Н. контр.						Блок-схема «Діаграма API»	КПІ ім. Ігоря Сікорського				
Затв.											



						КП.ІП-1208..045440.07.99.АД						
						Вебсервіс для пошуку партнерів з інтеграцією таск- менеджменту			Лп.	Маса	Мірило	
Розроб.	Добриченко К.А.											
Перев.	Лисовиченко О.І.											
									Лист 1	Листів 1		
Н. контр						Блок-схема «Архітектура додатку»			КПІ ім. Ігоря Сікорського			
Затв.												



						КПІ.ІП-1208..045440.07.99.CCM								
						Вебсервіс для пошуку партнерів з інтеграцією task				Літ.	Маса	Мірило		
Зм.	Лист	докум		Підпис	Дата									
Розроб.	Тобіратомова К.А.													
Перев.	Височенська О.І.													
										Лист 1	Листів 1			
						Схема структурна компонентів				КПІ ім. Ігоря Сікорського				
Н. контр														
Затв.														