

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Розробка високоефективного веб-сервісу для автоматизованого
пошуку вакансій із застосуванням методів штучного інтелекту»**

Виконав студент групи ІПЗ-21-5
СЕМЕРИКОВ Нікіта Олексійович

Керівник роботи:
ГРОМСЬКИЙ Олександр Олександрович

Рецензент:
ВЛАСЕНКО Олег Васильович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ЗАТВЕРДЖУЮ
Зав. кафедри інженерії
програмного забезпечення
Тетяна Вакалюк
«26» лютого 2025 р.

ЗАВДАННЯ

на кваліфікаційну роботу

Здобувач вищої освіти: Семериков Нікіта Олексійович

Керівник роботи: Громський Олександр Олександрович

Тема роботи: «Розробка високоефективного веб-сервісу для автоматизованого пошуку вакансій із застосуванням методів штучного інтелекту»,
затверджена Наказом закладу вищої освіти від 14.02.2025р., №61/с

Вихідні дані для роботи: процес автоматизованого пошуку вакансій і підбору персоналу із використанням інформаційних технологій, зокрема веб-сервісів, що інтегрують методи штучного інтелекту для персоналізованого аналізу резюме, рекомендацій і взаємодії між кандидатами та роботодавцями.

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Олександр ГРОМСЬКИЙ	02.03.2025	02.03.2025
2	Олександр ГРОМСЬКИЙ	02.03.2025	02.03.2025
3	Олександр ГРОМСЬКИЙ	02.03.2025	02.03.2025

РЕФЕРАТ

Кваліфікаційна робота освітнього ступеня «бакалавр» присвячена розробці високоефективного веб-сервісу для автоматизованого пошуку вакансій із застосуванням методів штучного інтелекту. Робота складається з програмного продукту та пояснювальної записки. Пояснювальна записка до кваліфікаційної роботи містить 87 сторінки, 36 ілюстрацій та 10 таблиць.

Метою роботи є розробка високоефективного веб-сервісу для автоматизованого пошуку вакансій із застосуванням методів штучного інтелекту, що дозволить підвищити точність і швидкість підбору відповідних пропозицій для користувачів, а також оптимізувати взаємодію між роботодавцями та кандидатами.

У роботі поставлені основні завдання на розробку системи. Також було проведено аналіз аналогічних систем з метою виявлення їх переваг і недоліків. Було обрано клієнт-серверну архітектуру на основі фреймворку Next.js. Приведені основні сценарії використання системи, логіка взаємодії між модулями системи, діаграми потоків даних, структура бази даних та алгоритми роботи ключових модулів. База даних спроектована з використанням MongoDB для забезпечення гнучкості та масштабованості. Для інтеграції функціоналу штучного інтелекту використовувалося API Gemini AI, яке забезпечує автоматизацію аналізу текстових даних резюме та вакансій. Клієнтська і серверна частини реалізовані за допомогою Next.js, що дозволяє об'єднати серверний рендеринг та швидкодію інтерфейсу користувача.

Ключові слова: ПОШУК РОБОТИ, АВТОМАТИЗАЦІЯ, ШТУЧНИЙ ІНТЕЛЕКТ, NEXT.JS, GEMINI AI, MONGODB.

					ДУ «Житомирська політехніка».25.121.21.000 - ПЗ				
Змн.	Арк.	№ докум.	Підпис	Дата					
Розроб.		Семериков Н.О.			Пояснювальна записка	Лім.		Арк.	Аркушів
Керівник		Громський О.О.						3	87
						ФІКТ Гр. ІПЗ-21-5			
Н. контр.									
Затвердив									

ABSTRACT

The qualification work for the bachelor's degree is dedicated to the development of a job search website utilizing artificial intelligence. The work consists of a software product and an explanatory note. The explanatory note includes 71 pages, 36 illustrations, and 10 tables, with a total volume of 87 pages.

The aim of this work is to develop a high-performance web service for automated job search using artificial intelligence methods, which will enhance the accuracy and speed of matching relevant opportunities for users, as well as optimize the interaction between employers and candidates.

The work sets the main tasks for the system's development. Additionally, an analysis of similar systems was conducted to identify their advantages and disadvantages. A client-server architecture based on the Next.js framework was chosen. The main usage scenarios of the system, interaction logic between system modules, data flow diagrams, database structure, and algorithms for the operation of key modules are presented.

The database was designed using MongoDB to ensure flexibility and scalability. To integrate artificial intelligence functionality, the Gemini AI API was used, providing automation for analyzing text data from resumes and job postings. Both the client and server sides were implemented using Next.js, which combines server-side rendering with high interface performance.

Keywords: JOB SEARCH, AUTOMATION, ARTIFICIAL INTELLIGENCE, NEXT.JS, GEMINI AI, MONGODB.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	0
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ.....	9
1.1 Постановка задачі	9
1.2 Аналіз аналогів програмного продукту	11
1.3 Вибір архітектури веб-сервісу для автоматизованого пошуку вакансій із застосуванням штучного інтелекту	15
1.4 Обґрунтування вибору інструментальних засобів	17
Висновки до першого розділу	21
РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБ-СЕРВІСУ ДЛЯ АВТОМАТИЗОВАНОГО ПОШУКУ РОБОТИ ІЗ ЗАСТОСУВАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ	22
2.1 Визначення варіантів використання та структури системи	22
2.2 Розробка бази даних системи	29
2.3 Проєктування та реалізація алгоритмів роботи системи	34
2.4 Реалізація функціоналу платформи сайту з пошуку роботи з використанням штучного інтелекту	40
Висновки до другого розділу	47
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ	48
3.1 Порядок встановлення та налаштування системи.....	48
3.2 Структура інтерфейсу та порядок взаємодії з платформою з пошуку роботи	49
3.3. Тестування роботи програмного продукту	67
Висновки до третього розділу	70
ВИСНОВКИ	71
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	72
ДОДАТКИ	75

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – база даних

ІІІ – штучний інтелект

SPA – single page application

		Семериків Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

У сучасному світі швидкий розвиток інформаційних технологій та штучного інтелекту (ШІ) істотно впливає на всі аспекти суспільного життя, у тому числі й на ринок праці. Однією з ключових проблем сучасного ринку праці є ефективний пошук роботи та підбір персоналу. Традиційні методи, що передбачають ручний аналіз резюме, зазвичай займають багато часу, потребують значних зусиль і ресурсів, а результати не завжди є достатньо ефективними. Натомість застосування технологій штучного інтелекту відкриває нові можливості для автоматизації та підвищення продуктивності цих процесів.

Актуальність теми полягає у необхідності створення інноваційних рішень для ринку праці, здатних забезпечити ефективний пошук роботи для кандидатів та підбір персоналу для роботодавців. Використання технологій штучного інтелекту дозволяє автоматизувати аналіз резюме, забезпечуючи персоналізовані рекомендації для кандидатів щодо покращення їх професійних характеристик, а також допомагає роботодавцям знаходити найбільш підходящих кандидатів на основі їх кваліфікацій, досвіду та навичок. Це сприятиме скороченню часу на пошук роботи та підбір персоналу, а також підвищенню якості цих процесів.

Метою кваліфікаційної роботи є розробка сучасного веб-сервісу для пошуку роботи з використанням технологій штучного інтелекту, який буде надавати роботодавцям інструменти для автоматизованого підбору персоналу, а користувачам — персоналізовані рекомендації щодо покращення їх резюме та підвищення шансів на працевлаштування. Основною метою є підвищення ефективності взаємодії між роботодавцями та кандидатами.

Практичними завданнями кваліфікаційної роботи є:

1. Провести аналіз сучасних веб-сервісів для пошуку роботи та вивчити підходи до інтеграції штучного інтелекту в системи підбору персоналу.
2. Обґрунтувати вибір технологічного стеку для реалізації клієнтської та серверної частини веб-застосунку, а також методів роботи з модулями ШІ.

		Семериков Н.О.			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

3. Спроекувати архітектуру веб-сервісу, базу даних і інтерфейс користувача з урахуванням принципів зручності, масштабованості та безпеки.

4. Реалізувати функціональність пошуку роботи, автоматизованого підбору персоналу, генерації рекомендацій та авторизації користувачів.

5. Провести тестування розробленого сервісу для перевірки його працездатності, відповідності вимогам і зручності використання.

Об'єктом дослідження є процеси пошуку роботи та підбору персоналу на ринку праці за допомогою цифрових технологій.

Предметом дослідження є веб-сервіс для пошуку роботи з інтеграцією штучного інтелекту, що забезпечує автоматизацію підбору персоналу та надає індивідуальні рекомендації для користувачів.

		Семериків Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1 Постановка задачі.

Метою цієї роботи є розробка інноваційної веб-платформи для пошуку роботи, що використовує алгоритми штучного інтелекту (ШІ) для оптимізації процесів підбору персоналу та надання персоналізованих рекомендацій кандидатам.

Одним із ключових елементів функціоналу платформи є створення інтерактивного середовища для роботодавців, яке полегшує аналіз відгуків на вакансії. Зокрема, система забезпечує доступ до віртуального HR-асистента у вигляді інтерактивного чат-помічника на основі штучного інтелекту. Завдяки інтеграції чат-бота, подібного до ChatGPT, роботодавець може ставити запитання щодо кандидатів і отримувати розгорнуту інформацію про їхній досвід, професійні навички та відповідність конкретним вимогам вакансії. Крім того, платформа надає зручний функціонал для пошуку як вакансій, так і кандидатів, дозволяючи застосовувати фільтри за місцем роботи, рівнем заробітної плати, досвідом та іншими критеріями, що значно підвищує ефективність навігації.

Користувачі можуть зберігати обрані вакансії у своєму особистому кабінеті, що дозволяє повернутися до них у будь-який зручний момент і зосередитися на тих пропозиціях, які їх дійсно зацікавили. Для кандидатів платформа формує персоналізовані рекомендації, спрямовані на покращення резюме та розвиток професійних навичок. Такі рекомендації генеруються на основі аналізу змісту резюме, відповідних вакансій та актуальних тенденцій на ринку праці, що дає змогу кожному користувачеві отримати індивідуальні поради щодо кар'єрного зростання.

Окремий модуль аналітики, призначений для адміністраторів системи, надає потужні інструменти для моніторингу активності користувачів, аналізу взаємодії між роботодавцями та кандидатами, а також оцінювання загальної ефективності функціонування платформи. Такий комплексний підхід до

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

реалізації функціоналу дозволяє створити сучасну та ефективну веб-платформу для пошуку і пропонування роботи.

Основні етапи розробки сайту для пошуку роботи з використанням штучного інтелекту:

1. Проектування платформи:

- Розробка архітектури системи, що включає серверну, клієнтську частини та базу даних.
- Проектування користувацького інтерфейсу з урахуванням потреб різних груп користувачів (роботодавців і кандидатів).

2. Реалізація функціоналу веб-сервіс:

- Інтеграція моделі штучного інтелекту Gemini для аналізу резюме, автоматизації підбору персоналу та формування персоналізованих рекомендацій.
- Налаштування параметрів і адаптація готової моделі під специфічні завдання платформи, такі як ранжування кандидатів, оцінка відповідності вакансій і надання рекомендацій.
- Реалізація механізмів пошуку та фільтрації вакансій і кандидатів.
- Впровадження системи збереження вакансій і профілів.
- Створення системи авторизації з підтримкою різних ролей користувачів.

3. Тестування веб-сайту:

- Функціональне тестування усіх компонентів.
- Перевірка точності роботи алгоритмів ШІ.
- Тестування інтерфейсу для забезпечення зручності використання.

4. Розробка дизайну веб-сайту:

- Створення загальної концепції дизайну сайту.
- Розробка зручного інтерфейсу для користувачів різних ролей (роботодавців, кандидатів, адміністраторів).
- Оптимізація UI/UX для забезпечення інтуїтивної навігації.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		10

Результатом розробки стане веб-сервіс, який полегшить процес підбору персоналу для роботодавців і підвищить ефективність пошуку роботи для кандидатів, забезпечуючи точний аналіз даних, персоналізовані рекомендації та можливості аналітики для адміністраторів.

1.2 Аналіз аналогів програмного продукту.

Для розробки сучасного веб-сервісу з пошуку роботи із застосуванням штучного інтелекту важливо проаналізувати наявні рішення, їх функціональні можливості та особливості реалізації. У цьому розділі представлено огляд трьох провідних онлайн-платформ в Україні: robota.ua, work.ua та djinni.co.

Robota.ua — одна з найстаріших платформ для пошуку роботи в Україні, яка охоплює широкий спектр вакансій і забезпечує взаємодію між роботодавцями та кандидатами в межах єдиної цифрової екосистеми. Серед основних переваг ресурсу — зручна система фільтрів, яка дозволяє шукати вакансії за різними критеріями, такими як категорія, місце роботи, рівень заробітної плати чи досвід. Кандидати можуть створювати резюме безпосередньо на сайті, використовуючи шаблони, що спрощує процес подання заявки. Крім того, платформа надсилає автоматичні сповіщення про нові вакансії відповідно до заданих параметрів пошуку. Для роботодавців доступна базова аналітика, яка відображає кількість переглядів та відгуків на опубліковані вакансії.

Разом з тим, платформа має і певні недоліки. Вона не пропонує персоналізованих рекомендацій для кандидатів, що могло б підвищити релевантність запропонованих вакансій. Засобів для прямої комунікації між роботодавцями та пошукачами також недостатньо, що обмежує гнучкість взаємодії. Аналітичний модуль надає лише базові показники, що ускладнює глибший аналіз ефективності публікацій для рекрутерів.

		Семериків Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		11

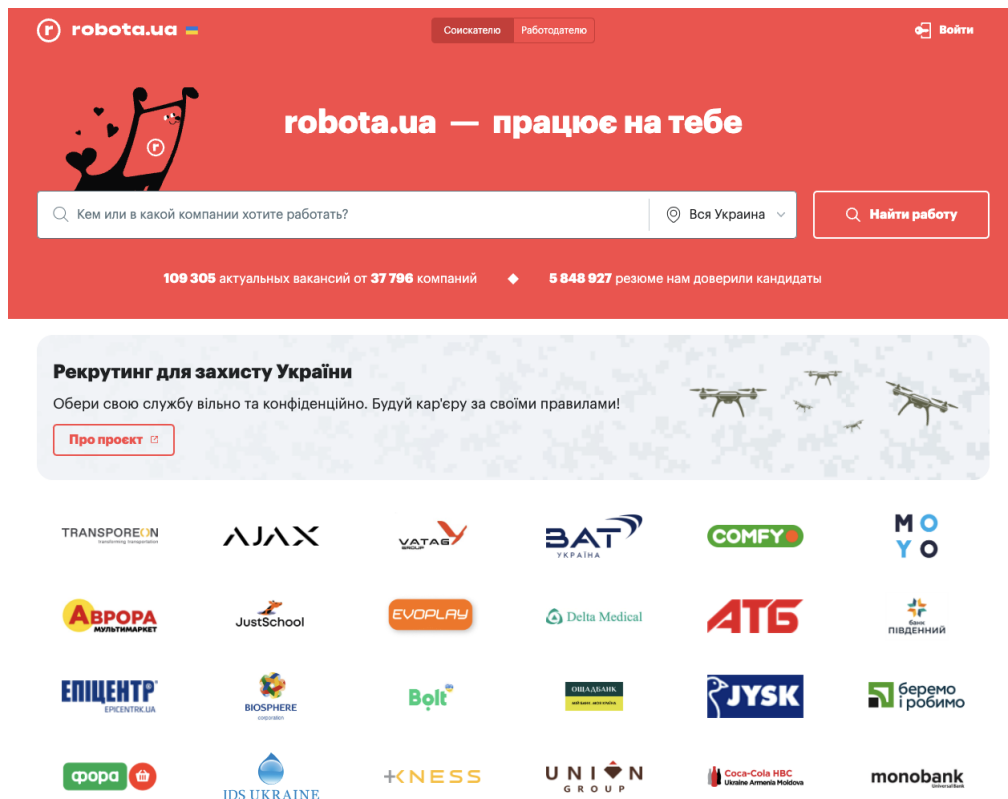


Рис. 1.1. Інтерфейс сайту Robota.ua

Work.ua — це одна з провідних платформ для пошуку роботи в Україні, яка виступає головним конкурентом Robota.ua та пропонує схожий набір функцій. Вона забезпечує користувачам можливість швидкого доступу до актуальних вакансій, публікації оголошень про роботу, а також надає багато додаткових матеріалів, зокрема порад щодо складання резюме, проходження співбесід і ефективного працевлаштування загалом. Платформа підтримує функціональний пошук із широким набором фільтрів, що дозволяє максимально точно підібрати вакансію за заданими параметрами. Користувачі можуть оперативно відгукуватися на пропозиції, надсилаючи резюме безпосередньо з сайту. Важливою особливістю є наявність розділу з відгуками про компанії, що допомагає оцінити роботодавця ще до подання заявки.

Водночас Work.ua має і свої обмеження. На відміну від деяких сучасних платформ, вона не пропонує інтерактивного помічника на основі штучного інтелекту, який міг би покращити досвід користування. Рівень персоналізації як для пошукачів, так і для роботодавців залишається недостатнім: платформа не враховує повною мірою індивідуальні переваги чи професійні інтереси

		Семериков Н.О			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		12

користувачів. Також варто зазначити, що аналітичні інструменти для роботодавців доволі обмежені і не дозволяють здійснювати глибоку оцінку ефективності розміщених вакансій.

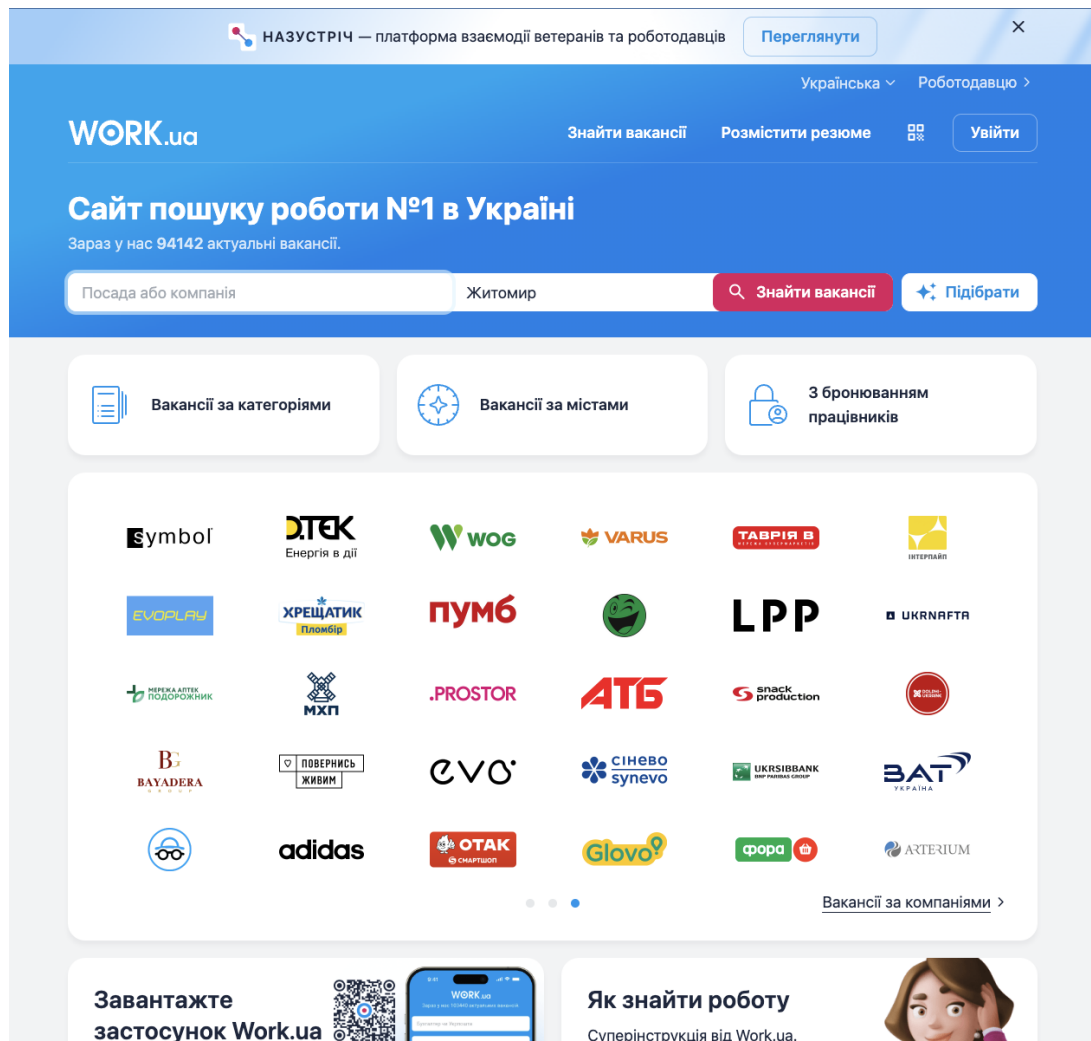


Рис. 1.2. Інтерфейс сайту Work.ua

Djinni.co — це спеціалізована платформа, орієнтована насамперед на IT-фахівців, яка вирізняється особливою увагою до конфіденційності кандидатів та використанням автоматизованого підбору вакансій. Унікальністю сервісу є можливість залишатися анонімним: кандидати можуть створювати профілі та подаватися на вакансії, не розкриваючи свої персональні дані до моменту, коли самі вирішать встановити контакт із роботодавцем. Завдяки інтелектуальним алгоритмам система пропонує користувачам лише ті вакансії, що максимально відповідають їхньому досвіду, навичкам і очікуванням. Комунікація між

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

роботодавцем і пошукачем відбувається безпосередньо через платформу, що забезпечує швидкий і зручний процес обміну інформацією.

Однак функціональність Djinni.co має певні обмеження. Платформа орієнтована майже виключно на ІТ-сферу, що значно звужує коло потенційних користувачів поза межами цієї галузі. Вона не пропонує універсального функціоналу для представників інших професій або масового ринку праці. Крім того, інструменти аналітики для роботодавців є досить базовими, що ускладнює проведення повного аналізу ефективності найму або оцінки поведінки кандидатів.

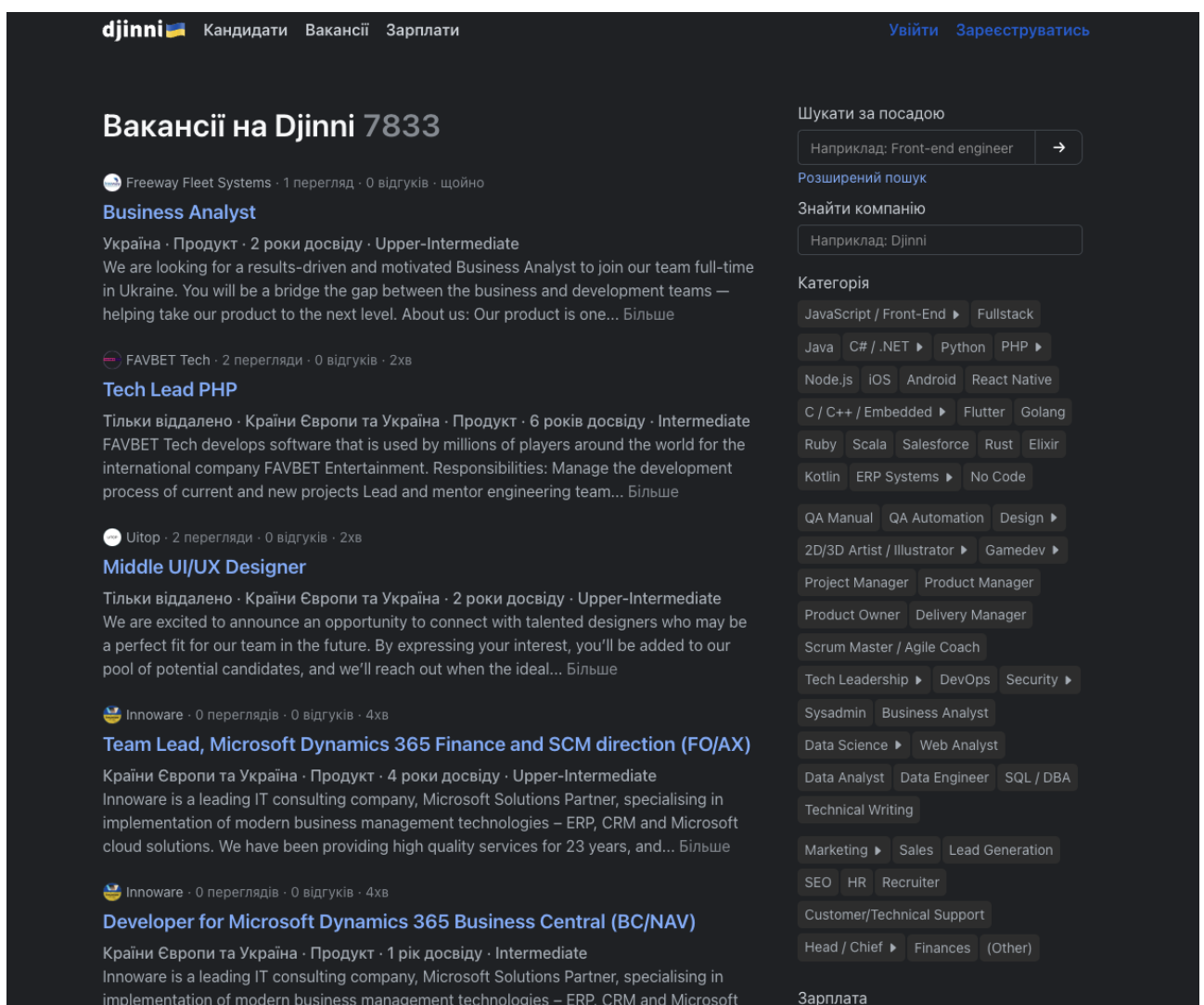


Рис. 1.3. Інтерфейс сайту Djinni.co

Проведений аналіз показав, що такі популярні платформи, як robota.ua, work.ua та djinni.co, мають розвинений базовий функціонал для публікації вакансій, створення резюме, пошуку роботи за фільтрами, проте їх можливості в

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		14

аспектах персоналізації, інтелектуального аналізу та автоматизації залишаються обмеженими. Наприклад, robota.ua і work.ua не забезпечують автоматичного добору вакансій на основі глибокого аналізу навичок кандидата, а djinni.co орієнтований переважно на IT-сферу й не пропонує універсального рішення для всіх галузей.

Натомість розроблюваний веб-сервіс вигідно вирізняється впровадженням інноваційних функцій на базі штучного інтелекту, які значно розширюють можливості як для пошукачів, так і для роботодавців. Зокрема, для кандидатів реалізовано інструмент персоналізованих порад, що аналізує завантажене резюме, виділяє ключові навички та автоматично пропонує найбільш релевантні вакансії, а також навички, яких бракує для омріяної посади. Це дозволяє користувачу не лише швидше знаходити відповідну роботу, але й цілеспрямовано розвивати кар'єру.

Для роботодавців платформа надає чат-бота-помічника, що виконує функції аналітики відгуків на вакансію. На основі зібраних даних він формує підсумки щодо кількості, якості та відповідності кандидатів, що дозволяє швидко оцінити ефективність оголошення. Та більше, система надає рекомендації щодо покращення вакансії: наприклад, пропонує уточнити вимоги або змінити формулювання для підвищення привабливості серед цільової аудиторії.

Завдяки таким можливостям запропонований веб-сервіс реалізує новий рівень автоматизації та персоналізації процесу підбору кадрів, якого не вистачає у поточних аналогах. Інтелектуальні функції платформи не тільки знижують навантаження на користувачів, а й підвищують точність підбору, скорочуючи час на пошук або найм. Таким чином, розробка має високий потенціал для вдосконалення сфери онлайн-рекрутингу в Україні.

1.3 Вибір архітектури веб-сервісу для автоматизованого пошуку вакансій із застосуванням штучного інтелекту.

Для створення веб-сервісу роботи, що використовує штучний інтелект, було обрано клієнт-серверну архітектуру, в якій клієнтська та серверна частини

		Семериків Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		15

реалізовані за допомогою Next.js[1]. Ця архітектура об'єднує фронтенд і бекенд в єдиному середовищі, забезпечуючи простоту обслуговування, швидкість роботи та зручність взаємодії користувачів із системою.

Унікальність підходу полягає у використанні функціоналу Server Actions [2], який дозволяє виконувати серверні операції безпосередньо в рамках компонентів Next.js. Це сприяє зменшенню складності розробки та забезпечує високу продуктивність сайту.

Клієнтська частина сайту, розроблена на базі Next.js, забезпечує швидкий доступ користувачів до функцій, таких як перегляд вакансій, отримання рекомендацій і взаємодія з іншими можливостями. Вона відповідає за збір даних від користувачів (наприклад, резюме чи налаштувань пошуку), надсилання запитів до серверної частини та відображення результатів обробки. Використання підходу SPA (Single Page Application) забезпечує плавний досвід роботи без необхідності перезавантаження сторінок. Завдяки адаптивному дизайну сайт коректно працює на різних пристроях: мобільних телефонах, планшетах і комп'ютерах. Основна задача клієнтської частини полягає у забезпеченні зручного інтерфейсу та взаємодії з серверною частиною для обробки складних операцій.

Серверна частина сайту, також реалізована з використанням Next.js, виконує обробку запитів і реалізацію бізнес-логіки. За допомогою технології Server Actions сервер аналізує резюме користувачів, співставляє їх із вакансіями та формує персоналізовані рекомендації на основі алгоритмів штучного інтелекту. Сервер також зберігає дані, включаючи профілі користувачів, історію взаємодій і вакансії, забезпечуючи надійність та швидкий доступ до інформації. Він управляє правами доступу до функцій сайту, включаючи преміум-опції, такі як розширена аналітика чи рекомендації. Серверна частина виступає центральним елементом обробки даних, забезпечуючи стабільну та ефективну роботу всього сайту.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

1.4 Обґрунтування вибору інструментальних засобів.

Для реалізації сайту пошуку роботи з використанням штучного інтелекту було обрано сучасний і продуктивний стек технологій, який забезпечує ефективну розробку, масштабованість та інтеграцію з інструментами ШІ. Обрані засоби дозволяють досягти поставлених цілей, забезпечуючи високу продуктивність, зручність розробки та якісний користувацький досвід.

Основу системи становить фреймворк Next.js, який було обрано для побудови як клієнтської, так і серверної частини сайту. Завдяки підтримці таких сучасних підходів, як Server-side Rendering (SSR), Static Site Generation (SSG) та Server Actions, він дозволяє поєднувати серверну логіку з фронтендом, що значно підвищує продуктивність і зручність використання. Для реалізації системи аутентифікація було інтегровано NextAuth.js [3] — рішення, яке ідеально підходить для Next.js і забезпечує підтримку різних методів авторизації, таких як Google, GitHub або email. Це дозволяє ефективно обробляти користувацькі сесії та гарантувати безпечний доступ до ресурсів платформи.

Уся розробка ведеться мовою TypeScript, що додає до JavaScript статичну типізацію та суворий контроль типів, що дозволяє виявляти помилки ще на етапі компіляції. Це знижує кількість помилок, спрощує підтримку коду та забезпечує типізацію всіх компонентів, моделей бази даних і бізнес-логіки. Для роботи з базою даних MongoDB[4] використовується ORM-бібліотека Mongoose [5], яка дозволяє ефективно моделювати дані, здійснювати їх валідацію та виконувати запити. Це особливо важливо для обробки даних користувачів, історії взаємодій, резюме та відгуків.

Інтеграція штучного інтелекту здійснюється за допомогою Vercel AI SDK — інструменту, що полегшує роботу з мовними моделями та ШІ-функціоналом [6]. Він дозволяє реалізувати персоналізовані рекомендації для користувачів, автоматизований аналіз резюме та генерування відповідей, підвищуючи рівень взаємодії з платформою. Робота з файлами, зокрема резюме у форматі PDF, реалізована через Vercel/blob, що забезпечує зручне завантаження, зберігання й

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

обробку документів у хмарному середовищі. Це сприяє ефективному управлінню файлами в межах усієї системи [7].

Для створення інтерфейсу з використанням сучасних іконок було обрано бібліотеку Lucide-react, яка вирізняється стильним дизайном, гнучкістю в налаштуванні та простотою інтеграції з компонентами React. Вона сприяє створенню привабливого, інтуїтивно зрозумілого користувацького інтерфейсу.

Загалом обраний стек технологій, що включає Next.js, TypeScript, Mongoose, Vercel AI SDK, Vercel/blob і Lucide-react, забезпечує надійну, масштабовану та інноваційну основу для створення веб-сайту пошуку роботи з інтегрованим штучним інтелектом.

Для реалізації веб-сервісу з пошуку роботи з використанням штучного інтелекту було обрано базу даних MongoDB у поєднанні з бібліотекою Mongoose. MongoDB забезпечує зручне зберігання даних у форматі JSON/BSON, що ідеально підходить для роботи з динамічними об'єктами, такими як резюме, профілі користувачів і вакансії. Щоб обґрунтувати вибір, розглянемо порівняння MongoDB з іншими популярними системами управління базами даних: PostgreSQL і MySQL.

Для оцінки основних характеристик СУБД важливо враховувати тип даних, що підтримується, можливості масштабування, продуктивність і гнучкість у роботі з даними. У таблиці 1.1 наведено порівняння ключових характеристик MongoDB, PostgreSQL і MySQL.

Таблиця 1.1

Порівняльна характеристика MongoDB, PostgreSQL і MySQL за загальними параметрами

Характеристика	MongoDB	PostgreSQL	MySQL
Тип даних	Документно-орієнтована (JSON/BSON)	Реляційна	Реляційна
Масштабованість	Горизонтальна	Горизонтальна та вертикальна	Переважно вертикальна

Продовження таблиці 1.1

Гнучкість схеми	Динамічна, без чітко визначеної схеми	Жорстко визначена схема	Жорстко визначена схема
Продуктивність	Висока для обробки нереляційних даних	Ефективна для складних реляційних зв'язків	Висока для простих реляційних запитів
Підтримка JSON	Природна, основний формат зберігання	JSONB, менш продуктивний	Обмежена підтримка
Простота інтеграції	Висока завдяки Mongoose	Стандартна для SQL	Стандартна для SQL

Для визначення відповідності СУБД до вимог проєкту, таких як робота з великою кількістю користувачів або обробка динамічних даних, було проаналізовано їхні можливості. У таблиці 1.2 наведено відповідність MongoDB, PostgreSQL та MySQL ключовим вимогам проєкту.

Таблиця 1.2.

Відповідність MongoDB, PostgreSQL і MySQL основним вимогам проєкту

Вимога	MongoDB	PostgreSQL	MySQL
Гнучкість у зміні структури даних	Висока	Обмежена	Низька
Масштабування для великої кількості користувачів	Горизонтальне	Горизонтальне	Переважно вертикальне
Обробка нереляційних даних	Дуже ефективна	Обмежена	Складно реалізувати

Продовження таблиці 1.2

Продуктивність із великим обсягом даних	Відмінна	Висока	Висока
Легка інтеграція з JavaScript	Ідеальна	Складна	Складна
Підтримка складних зв'язків	Обмежена	Повна	Повна

Для оцінки характеристик баз даних важливо врахувати такі параметри, як підтримуваний обсяг даних і можливості захисту. У таблиці 1.3 наведено порівняння MongoDB, PostgreSQL і MySQL за цими критеріями.

Таблиця 1.3.

Порівняння MongoDB, PostgreSQL і MySQL за характеристиками обсягу, масштабованості та безпеки

Характеристика	MongoDB	PostgreSQL	MySQL
Максимальний обсяг бази даних	Необмежений; обмеження залежать від файлової системи та ресурсів сервера	Необмежений; обмеження визначаються файловою системою та ресурсами сервера	Необмежений; обмеження залежать від файлової системи та конфігурації сервера
Розмір документа/рядка	До 16 МБ на документ; для більших об'єктів використовується GridFS	До 1,6 ТБ на рядок; обмеження залежать від типів даних та конфігурації таблиці	До 65 535 байт на рядок (без урахування BLOB/TEXT); BLOB/TEXT зберігаються окремо
Масштабованість	Горизонтальна (sharding)	Горизонтальна та вертикальна	Переважно вертикальна

Продовження таблиці 1.3

Захист даних	Шифрування на рівні поля і TLS	Шифрування на рівні поля і TLS	Шифрування через сторонні засоби
Резервне копіювання	Інтегрована підтримка	Інтегрована підтримка	Залежить від реалізації

На основі аналізу MongoDB було обрано як оптимальну базу даних для цього проєкту завдяки її гнучкості, підтримці великих обсягів даних, надійності та засобам захисту. Як показано у таблицях 1.1–1.3, MongoDB перевершує PostgreSQL і MySQL у роботі з динамічними структурами даних, масштабованістю та інтеграцією із сучасними технологіями.

Висновки до першого розділу.

У цьому розділі визначено мету роботи, сформульовано основні функціональні вимоги до платформи та проведено аналіз аналогів існуючих рішень. Основною перевагою розроблюваної системи стане інтеграція штучного інтелекту для автоматизації процесів і персоналізації рекомендацій. Обґрунтовано вибір архітектури та інструментальних засобів, що забезпечують ефективність, масштабованість і зручність використання платформи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБ-СЕРВІСУ ДЛЯ АВТОМАТИЗОВАНОГО ПОШУКУ РОБОТИ ІЗ ЗАСТОСУВАННЯМ ШТУЧНОГО ІНТЕЛЕКТУ

2.1 Визначення варіантів використання та структури системи.

Для забезпечення функціональності веб-сервісу з пошуку роботи, були реалізовані модулі для управління вакансіями, обробки резюме, а також комунікації між роботодавцями та шукачами роботи. Сайт надає можливість зберігати дані про вакансії, кандидатів, роботодавців, що значно спрощує процес пошуку роботи та підбору персоналу. Основний функціонал включає перегляд, публікація та редагування вакансій, перегляд резюме а також автоматизовану обробку текстових даних за допомогою штучного інтелекту, реалізованого через Gemini AI.

На сайті передбачені різні ролі користувачів із відповідними правами доступу (табл. 2.1).

Таблиця 2.1

Ролі користувачів платформи

Тип користувача	Короткий опис
Адміністратор	Керує сайтом, включаючи управління користувачами, модерацію контенту та перевірку вакансій.
Роботодавець	Має можливість публікувати вакансії, переглядати резюме та взаємодіяти з кандидатами.
Кандидат	Може, переглядати вакансії, подавати відгуки на вакансії, отримувати поради по ваканції, щоб мати більше шансів отримати бажану вакансію та завантажувати своє резюме.

На (рис. 2.1) зображено побудовану діаграму базових варіантів використання для кожного типу користувачів.

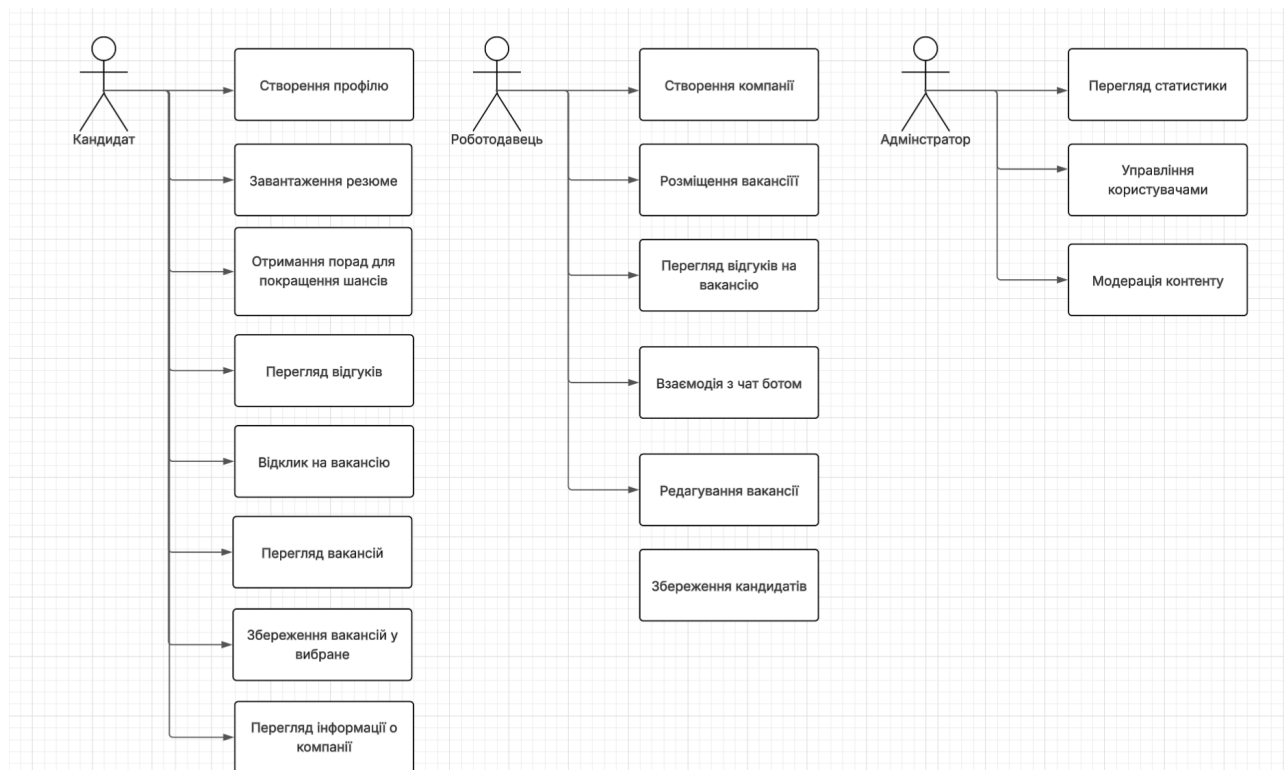


Рис 2.1. Варіанти використання веб-сервісу для пошуку роботи із застосуванням штучного інтелекту

Основними функціями користувачів сайту, які мають роль кандидат, є такі:

1. Можливість створення та редагування профілю користувача для демонстрації своїх професійних навичок та досвіду.
2. Зручний пошук вакансій за ключовими словами, містом та категорією, що дозволяє швидко знайти потрібну роботу.
3. Подача заявок на вакансії через простий та інтуїтивний інтерфейс.
4. Перегляд історії поданих заявок для відстеження прогресу у процесі працевлаштування.
5. Завантаження резюме у систему для його використання при подачі заявок.
6. Можливість переглядати інформацію про роботодавців, включаючи їх відгуки, рейтинг, та розмір компанії, для прийняття об'ґрунтованого рішення.
7. Збереження цікавих вакансій у список "Вибране" для зручного доступу в майбутньому.

8. Відстеження статусу поданих заявок, включаючи етапи розгляду заявки роботодавцем.

9. Отримання персоналізованих порад щодо підвищення шансів на працевлаштування, рекомендацій з покращення резюме або підготовки до співбесіди на основі аналізу даних за допомогою штучного інтелекту.

Основними функціями користувачів сайту, які мають ролі роботодавців, є такі:

1. Створення та редагування вакансій, включаючи опис роботи, вимоги до кандидатів та умови праці.

2. Перегляд кандидатів, які подали заявки на вакансії, та можливість сортування їх за статусом або іншими критеріями.

3. Використання штучного інтелекту для попереднього відбору резюме, щоб скоротити час на ручний перегляд.

4. Перегляд аналітики щодо вакансій, таких як кількість переглядів, поданих заявок та ефективність публікацій.

5. Створення облікового запису компанії з додаванням логотипу, опису та іншої інформації для залучення кандидатів.

6. Публікація вакансій з можливістю налаштування рівня приватності, включаючи приховування компанії, якщо це потрібно.

7. Аналіз відгуків кандидатів та їх резюме за допомогою інтегрованого чат-бота на базі штучного інтелекту, який надає узагальнені висновки, рекомендації та попередню оцінку відповідності кандидата вимогам вакансії.

Основними функціями користувачів сайту, які мають роль адміністратора, є такі:

1. Модерація контенту, включаючи вакансії та компанії, для забезпечення дотримання правил сайту.

2. Відстеження скарг користувачів, їх перевірка та прийняття відповідних заходів для вирішення проблем.

Функціональні вимоги.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

Система повинна підтримувати реєстрацію користувачів із можливістю призначення ролі — кандидата, роботодавця або адміністратора. Після реєстрації користувачі матимуть змогу входити до системи за допомогою свого акаунта. У межах особистого кабінету кожен користувач зможе створювати, редагувати або видаляти профільну інформацію, включаючи резюме та перелік своїх навичок.

Сервіс забезпечує зручний пошук вакансій, що дозволяє використовувати ключові слова, категорії, місто чи рівень заробітної плати як фільтри. Роботодавці мають можливість публікувати власні вакансії, вказуючи вимоги, умови праці, заробітну плату та можливі переваги. Шукачі роботи, у свою чергу, можуть подавати заявки на цікаві вакансії, додаючи супровідний лист.

Після подання заявок роботодавці отримують доступ до інструментів управління ними: перегляд, затвердження, відхилення, коментування чи зміна статусу (наприклад, «розглянуто», «відібрано», «відхилено»). Для зручності користувачів реалізовано функції фільтрації та сортування — як вакансій, так і кандидатів — за різними критеріями, включно з рівнем зарплати, досвідом або місцем роботи. Шукачі роботи мають можливість відстежувати статус поданих заявок у своєму профілі.

Для роботодавців передбачено блок аналітики, який надає інформацію про перегляди вакансій, кількість заявок, рейтинг компанії та інші статистичні показники. Окрім того, усі користувачі можуть зберігати вакансії або профілі в обране для подальшого перегляду.

Адміністративна частина системи дозволяє модерувати контент, зокрема вакансії, резюме та коментарі користувачів. Додатково, для пошукачів роботи передбачено систему рекомендацій, яка за допомогою штучного інтелекту аналізує профіль користувача та надає персоналізовані поради щодо вдосконалення резюме й підготовки до співбесіди. Роботодавці також можуть користуватися чат-ботом на основі штучного інтелекту, який допомагає аналізувати резюме кандидатів, дає рекомендації щодо вибору найбільш відповідних претендентів та оптимізує процес підбору персоналу.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

Нефункціональні вимоги.

1. Сприйняття

- Час, необхідний для навчання звичайних користувачів – 2-3 години, а для досвідчених користувачів – до 1 години.
- Час відгуку для виконання типових задач (пошук вакансій, перегляд резюме) – не більше 10 секунд; для складних завдань (генерація рекомендацій, аналітика) – не більше 20 секунд.
- Інтерфейс повинен бути інтуїтивно зрозумілим, із детальними підказками та прикладами використання.

2. Надійність

- Доступність системи повинна становити не менше 98%, з максимально допустимим часом обслуговування – до 2% загального часу роботи.
- Середній час безвідмовної роботи системи повинен бути не менше 30 робочих днів.
- Максимально допустима норма помилок – 1 помилка на 20 000 рядків коду.
- Продуктивність
- Система повинна підтримувати одночасну роботу мінімум 1000 активних користувачів.
- Обробка запитів до бази даних повинна виконуватися за час, що не перевищує 2 секунд для простих запитів та 3 секунд для складних.
- Максимальний час завантаження сторінки – до 3 секунд при стандартній швидкості з'єднання.

4. Можливість експлуатації

- Масштабованість:

Система повинна підтримувати збільшення кількості користувачів без зниження продуктивності шляхом додавання серверних потужностей.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

- Адаптивність:

Інтерфейс повинен коректно працювати на різних пристроях (десктопи, смартфони, планшети) та підтримувати всі сучасні браузер.

5. Безпека

- Паролі користувачів повинні бути надійно зашифровані під час зберігання та передачі, щоб запобігти несанкціонованому доступу.
- Необхідно впровадити багаторівневу систему аутентифікації з урахуванням ролей користувачів, яка передбачає розмежування доступу до функціоналу залежно від ролі
- Регулярне резервне копіювання даних з мінімальною частотою раз на добу.

Для розробки сайту були використані такі технології — фреймворк Next.js, що забезпечує побудову як клієнтської, так і серверної частини; база даних MongoDB, яка використовується для зберігання даних; API Gemini AI та Ai SDK від Vercel, за допомогою яких реалізовано функції автоматичного аналізу текстових даних резюме та вакансій. Архітектура сайту побудована на основі функціонального підходу, що дозволяє організовувати бізнес-логіку у вигляді окремих функцій без необхідності використання об'єктно-орієнтованих структур.

Діаграма структури компонентів сайту (рис. 2.2) відображає зв'язки між основними функціональними блоками та потоками даних, які забезпечують реалізацію функцій пошуку роботи, управління вакансіями, обробки резюме та взаємодії між кандидатами і роботодавцями.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

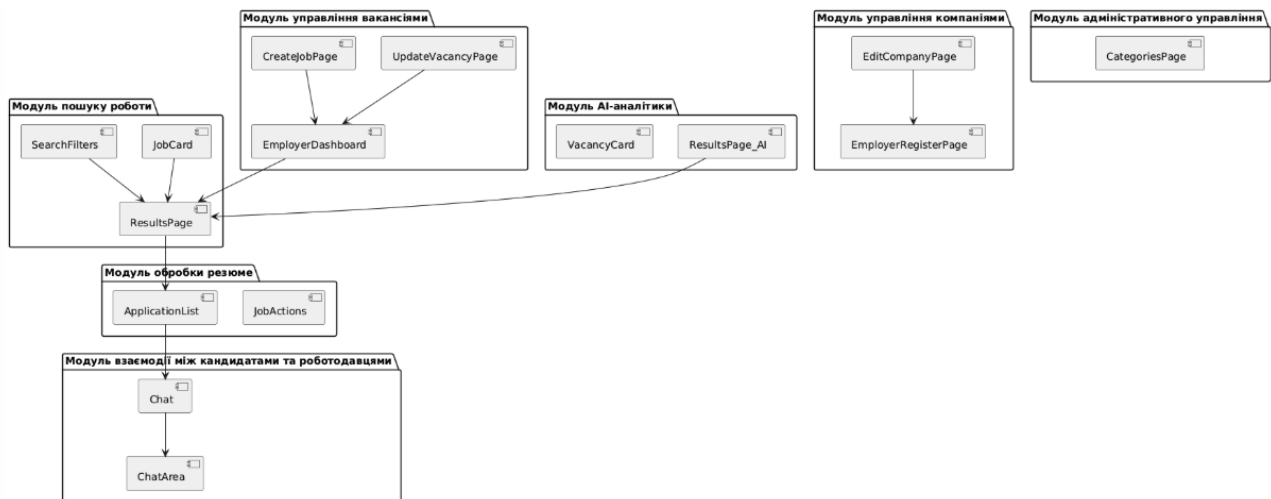


Рис. 2.2 Діаграма структури компонентів сайту

Функціональність платформи охоплює кілька ключових напрямів, які забезпечують повноцінну роботу веб-сервісу з пошуку роботи.

Система пошуку вакансій відповідає за фільтрацію, пошук і відображення актуальних пропозицій. Основними елементами цього напрямку є фільтри (за містом, зарплатою, категорією), картки вакансій і сторінка результатів, яка дозволяє кандидатам швидко знаходити релевантні варіанти.

Функціонал для роботодавців включає можливість створення, редагування та деактивації вакансій через спеціальну панель керування. Через неї роботодавці можуть повністю контролювати всі свої публікації, стежити за відгуками та оновлювати інформацію в реальному часі.

Система обробки резюме дозволяє кандидатам завантажувати своє резюме, подавати заявки на вакансії та переглядати статуси своїх відгуків. Водночас роботодавці мають доступ до списку отриманих заявок та можуть здійснювати первинний аналіз кандидатів.

Для комунікації між кандидатами та роботодавцями реалізовано чат, який стає доступним після подання заявки. Через цей чат обидві сторони можуть обговорити деталі вакансії, організувати співбесіду або поставити уточнювальні питання.

Профілі компаній дають змогу роботодавцям представляти свою організацію, надаючи можливість додавати логотип, опис, контактну

інформацію та посилання на офіційні ресурси. Такий підхід сприяє формуванню довіри з боку кандидатів і підвищує загальну привабливість опублікованих вакансій.

Адміністративна частина системи надає інструменти для керування категоріями вакансій, контролю контенту та підтримки структури платформи. Доступ до цього функціоналу мають лише адміністратори.

Штучний інтелект інтегровано у ключові частини системи такі як аналіз резюме та обробку вакансій. Він забезпечує автоматичне оцінювання відповідності кандидатів вимогам, формує персоналізовані рекомендації та сприяє підвищенню точності добору.

Усі елементи платформи тісно взаємодіють між собою, формуючи цілісну систему для ефективного пошуку роботи та підбору персоналу.

2.2 Розробка бази даних системи.

Дані в системі зберігаються у базі даних MongoDB, яка забезпечує гнучке зберігання та ефективний доступ до інформації. Основними колекціями бази даних є 8 колекцій, які реалізують основний функціонал сайту з пошуку роботи (табл. 2.2).

Таблиця 2.2

Основні колекції бази даних

Назва колекції	Призначення
User	Колекція користувачів
Company	Колекція компаній
JobVacancy	Колекція вакансій
Application	Колекція заявок на вакансії
Category	Колекція категорій вакансій
Favorite	Колекція обраних вакансій
Message	Колекція повідомлень чату з ІІІ
Resume	Зберігання резюме користувачів (інкапсульовано у User)

Структура бази даних наочно представлена на (рис. 2.3).

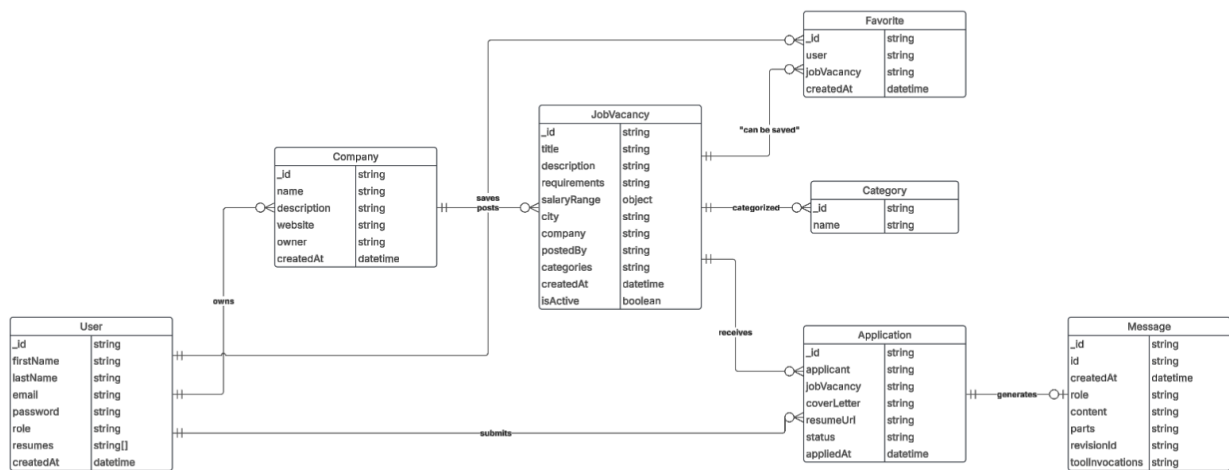


Рис 2.3 Структура бази даних

Колекція User слугує для збереження даних користувачів системи. Опис основних полів наведено у таблиці 2.3.

Таблиця 2.3

Структура колекції User

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор користувача
firstName	String	-	-	Ім'я користувача
lastName	String	-	-	Прізвище користувача
email	String	-	-	Електронна пошта
password	String	-	-	Пароль користувача (захищений)
role	String	-	-	Роль користувача (шукач роботи, роботодавець, адміністратор)
resumes	Array	-	-	Масив резюме користувача
createdAt	Date	-	-	Дата реєстрації

Колекція «Company» використовується для збереження даних про компанії, які реєструються на платформі для розміщення своїх вакансій. Опис полів колекції наведено нижче (табл. 2.4).

Таблиця 2.4

Структура колекції Company

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор компанії
name	String	-	-	Назва компанії
description	String	-	-	Опис діяльності компанії
website	String	-	-	Офіційний вебсайт компанії
owner	ObjectId	-	User._id	Ідентифікатор власника компанії
createdAt	Date	-	-	Дата створення профілю компанії

Колекція «JobVacancy» зберігає дані про вакансії, які створюють роботодавці. Вона є однією з ключових у системі, оскільки на пряму взаємодіє з користувачами та кандидатами. Опис структури наведено в табл. 2.5.

Таблиця 2.5

Структура колекції JobVacancy

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор вакансії
title	String	-	-	Назва вакансії
description	String	-	-	Опис обов'язків та завдань
requirements	Array	-	-	Перелік вимог до кандидата
salaryRange	Object	-	-	Діапазон зарплати (мінімальна та максимальна)
city	String	-	-	Місто розміщення вакансії
company	ObjectId	-	Company._id	Компанія, яка створила вакансію
postedBy	ObjectId	-	User._id	Користувач-роботодавець
categories	Array	-	Category._id	Категорії вакансії
createdAt	Date	-	-	Дата публікації
isActive	Boolean	-	-	Статус активності вакансії

Колекція «Application» зберігає дані про подані заявки кандидатів на відкриті вакансії. Вона фіксує зв'язок між шукачем роботи та вакансією. Детальний опис полів подано в табл. 2.6.

Таблиця 2.6

Структура колекції Application

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор заявки
applicant	ObjectId	-	User._id	Кандидат, який подав заявку
jobVacancy	ObjectId	-	JobVacancy._id	Вакансія, на яку подано заявку
coverLetter	String	-	-	Супровідний лист
resumeUrl	String	-	-	Посилання на резюме кандидата
status	String	-	-	Статус розгляду заявки (очікує, прийнято, відхилено)
appliedAt	Date	-	-	Дата подачі заявки

Колекція «Category» використовується для класифікації вакансій за професійними напрямками. Це допомагає покращити фільтрацію при пошуку. Опис полів — у табл. 2.7.

Таблиця 2.7

Структура колекції Category

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор категорії
name	String	-	-	Назва категорії вакансій

Колекція «Favorite» містить інформацію про вакансії, які шукачі роботи зберегли у "вибране" для подальшого перегляду. Її структура подана в табл. 2.8.

Таблиця 2.8

Структура колекції Favorite

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор збереження

Продовження таблиці 2.8

user	ObjectId	-	User._id	Користувач, який додав у вибране
jobVacancy	ObjectId	-	JobVacancy._id	Вакансія, додана у вибране
createdAt	Date	-	-	Дата додавання

Колекція «Message» відповідає за зберігання повідомлень між користувачами та штучним інтелектом у чаті. Її використання дозволяє проводити аналіз резюме та надавати рекомендації. Опис полів — у табл. 2.9.

Таблиця 2.9

Структура колекції Message

Назва поля	Тип даних	PK	FK	Опис
_id	ObjectId	+	-	Унікальний ідентифікатор повідомлення
id	String	-	-	Внутрішній ідентифікатор повідомлення
createdAt	Date	-	-	Дата створення повідомлення
role	String	-	-	Роль автора повідомлення (користувач або ІІІ)
content	String	-	-	Основний текст повідомлення
parts	Array	-	-	Частини повідомлення (наприклад, текстові блоки)
revisionId	String	-	-	Ідентифікатор ревізії
toolInvocations	Array	-	-	Викликані інструменти у процесі обробки

Використовуючи такі колекції, як User, Company та JobVacancy, було побудовано логіку для зберігання основної інформації про користувачів, роботодавців та їхні вакансії, із розподіленням доступів відповідно до ролей користувачів. Завдяки колекціям Application та Favorite реалізовано функціонал подання заявок на вакансії та можливість збереження цікавих пропозицій у "вибране". Колекції Category та Message використовуються для забезпечення зручної класифікації вакансій за напрямками діяльності та для організації чату зі

штучним інтелектом, який допомагає у аналізі резюме й наданні рекомендацій шукачам роботи та роботодавцям.

2.3 Проєктування та реалізація алгоритмів роботи системи.

Для реалізації веб-сервісу з пошуку роботи було обрано клієнт-серверну архітектуру з використанням сучасного фреймворку Next.js, який забезпечує ефективний розподіл навантаження між клієнтською та серверною частинами застосунку. Такий підхід дозволяє створювати динамічні веб-сторінки з високою продуктивністю, а також забезпечує підтримку серверного рендерингу (SSR) та статичної генерації сторінок (SSG), що суттєво покращує швидкодію і сприяє кращій індексації ресурсу пошуковими системами.

Ключовою особливістю реалізації архітектури є використання Server Actions — новітнього механізму, запропонованого в Next.js, який дозволяє викликати серверні функції безпосередньо з клієнтських компонентів, міняючи потребу у створенні окремих API-інтерфейсів або налаштуванні RESTful-запитів. Це інноваційне рішення значно спрощує взаємодію між фронтендом і бекендом.

Застосування Server Actions забезпечує низку важливих переваг. Уся логіка обробки даних виконується безпосередньо на сервері, що спрощує архітектуру системи та зменшує ймовірність виникнення помилок. Такий підхід також підвищує рівень безпеки, оскільки виконання коду відбувається в ізольованому серверному середовищі, мінімізуючи ризики, пов'язані з вразливістю клієнтської частини. Крім того, Server Actions значно спрощують розробку і підтримку застосунку, дозволяючи обійтись без великої кількості вручну створених API-маршрутів — це особливо актуально для масштабованих проєктів. Водночас зменшується кількість проміжних рівнів обробки запитів, що позитивно впливає на продуктивність і прискорює взаємодію між користувачем та системою. Таким чином, клієнт-серверна архітектура на основі Next.js із використанням Server Actions є ефективним рішенням для побудови сучасного високопродуктивного вебсервісу з пошуку роботи. На (рис. 2.4) представлено загальну схему взаємодії клієнта з сервером за допомогою Server Actions у Next.js.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

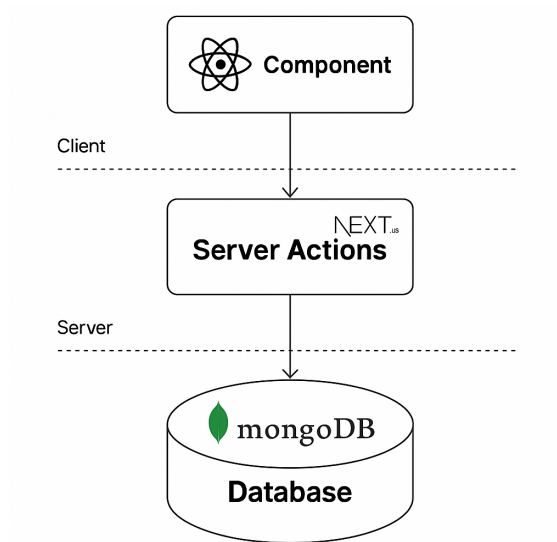


Рисунок 2.4 – Загальна схема взаємодії клієнта, серверних дій та бази даних у системі на основі Next.js

Також для ілюстрації алгоритму роботи сайту побудовано структурну діаграму (рис. 2.5), на якій зображені основні потоки виконання логіки для користувачів із різними ролями.

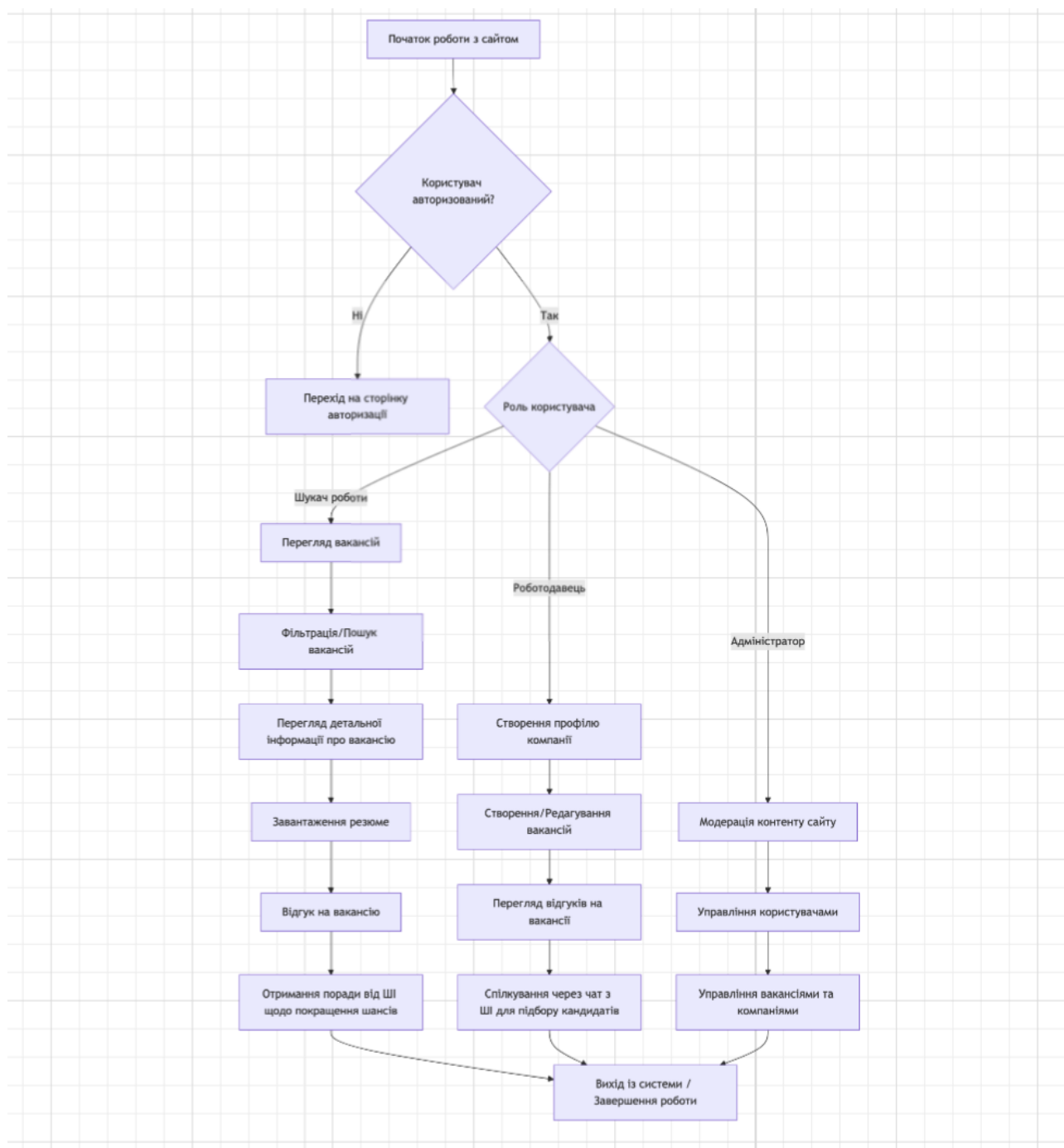


Рис 2.5 Структурна діаграма системи

Отримана діаграма ілюструє основну логіку взаємодії користувача із системою пошуку роботи, починаючи з моменту входу до завершення сеансу. На початковому етапі відбувається перевірка наявності авторизації: якщо користувач не авторизований, система автоматично перенаправляє його на сторінку входу. У разі успішної авторизації система ідентифікує роль користувача — кандидат, роботодавець або адміністратор — і відповідно до неї

налаштовує рівень доступу до функціоналу системи. Залежно від визначеної ролі користувачеві відкривається відповідний інтерфейс і набір дій, що забезпечує персоналізовану взаємодію з платформою.

Для шукача роботи відкривається функціонал перегляду та фільтрації вакансій, доступ до детальної інформації про вакансії, можливість завантажити резюме, подати заявку на обрану вакансію, а також отримати персоналізовані рекомендації й поради від системи штучного інтелекту. Роботодавець, у свою чергу, має змогу створювати профіль компанії, розміщувати, редагувати або деактивувати вакансії, переглядати заявки кандидатів і користуватися інтегрованим чат-помічником на основі ШІ для допомоги у відборі потенційних співробітників. Адміністратор системи має розширені повноваження, включаючи модерацію контенту сайту, управління обліковими записами користувачів, вакансіями та інформацією про компанії.

Після завершення взаємодії із системою кожен користувач може вийти з облікового запису або автоматично завершити сеанс, що закриває доступ до функціоналу до наступної авторизації. Така логіка забезпечує структуровану, рольову взаємодію з системою та високий рівень безпеки.

Також для відображення архітектурної побудови системи було побудовано діаграму компонентів (рис. 2.6), на якій зображено основні модулі клієнтської частини, серверної частини та взаємодію з базою даних.

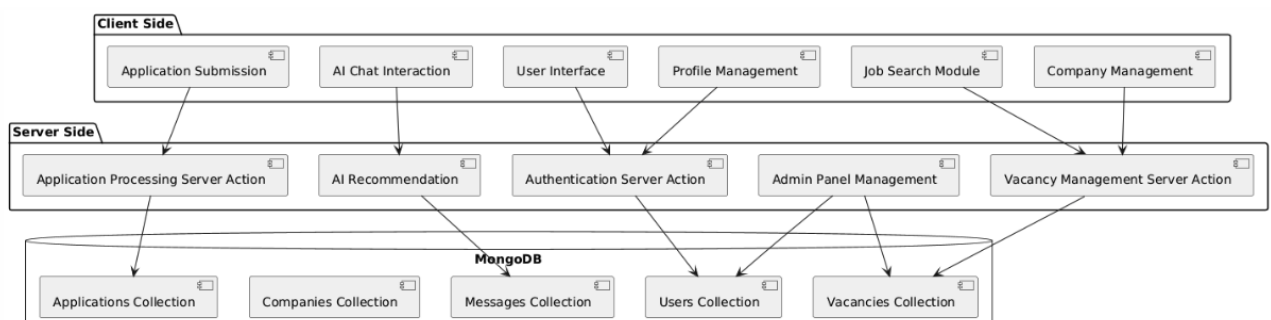


Рисунок 2.6 – Діаграма компонентів сайту

На діаграмі представлено структуру основних компонентів системи сайту з пошуку роботи, які взаємодіють між собою для забезпечення повного

функціоналу платформи. Центральною частиною є User Interface, що відповідає за відображення інтерфейсу та забезпечує взаємодію користувача з усіма можливостями сайту. Користувач може шукати вакансії через Job Search Module, який дозволяє здійснювати фільтрацію за ключовими словами, містом, категорією чи рівнем зарплати.

Функціонал управління профілем реалізовано у Profile Management, який забезпечує створення, редагування, видалення облікового запису та завантаження резюме. Після вибору вакансії користувач має можливість надіслати своє резюме через Application Submission, який відповідає за процес подання заявок та супровідних документів.

Роботодавці мають змогу управляти профілем компанії за допомогою Company Management, включаючи створення, редагування та публікацію вакансій. Інтегрований AI Chat Interaction дозволяє кандидатам і роботодавцям отримувати рекомендації, аналіз резюме та поради щодо відповідності вакансіям завдяки взаємодії з чат-помічником на основі штучного інтелекту.

Для забезпечення безпеки та контролю доступу використовується Authentication Server Action, який відповідає за авторизацію, реєстрацію користувачів і перевірку прав доступу. Окремий Vacancy Management Server Action реалізує всі дії, пов'язані з управлінням вакансіями — їх створення, редагування та деактивацію. Обробка заявок відбувається через Application Processing Server Action, який зберігає подані документи, змінює статуси заявок та веде облік активності кандидатів.

Інтелектуальні поради формуються на рівні AI Recommendation, де система аналізує дані користувачів та вакансій і надає персоналізовані рекомендації. Для адміністраторів доступний Admin Panel Management, який забезпечує модерацію контенту, управління користувачами, вакансіями та компаніями.

На рівні бази даних кожен тип об'єкта зберігається у відповідній колекції. Users Collection зберігає інформацію про користувачів, їхні профілі та резюме. Companies Collection містить відомості про компанії, які розміщують вакансії.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Vacancies Collection акумулює дані про відкриті позиції, включаючи опис, вимоги та умови праці. Applications Collection містить усі заявки, подані кандидатами. Messages Collection зберігає історію взаємодії користувачів із ШІ-системою, що забезпечує прозорість комунікації та можливість аналізу запитів.

У сукупності ці компоненти створюють цілісну, функціональну та масштабовану архітектуру системи пошуку роботи з інтеграцією штучного інтелекту.

Далі розглянемо алгоритм роботи модуля подання відгуку на вакансію. Для наочного зображення послідовності взаємодії між основними компонентами системи було побудовано діаграму послідовності (рис. 2.7).

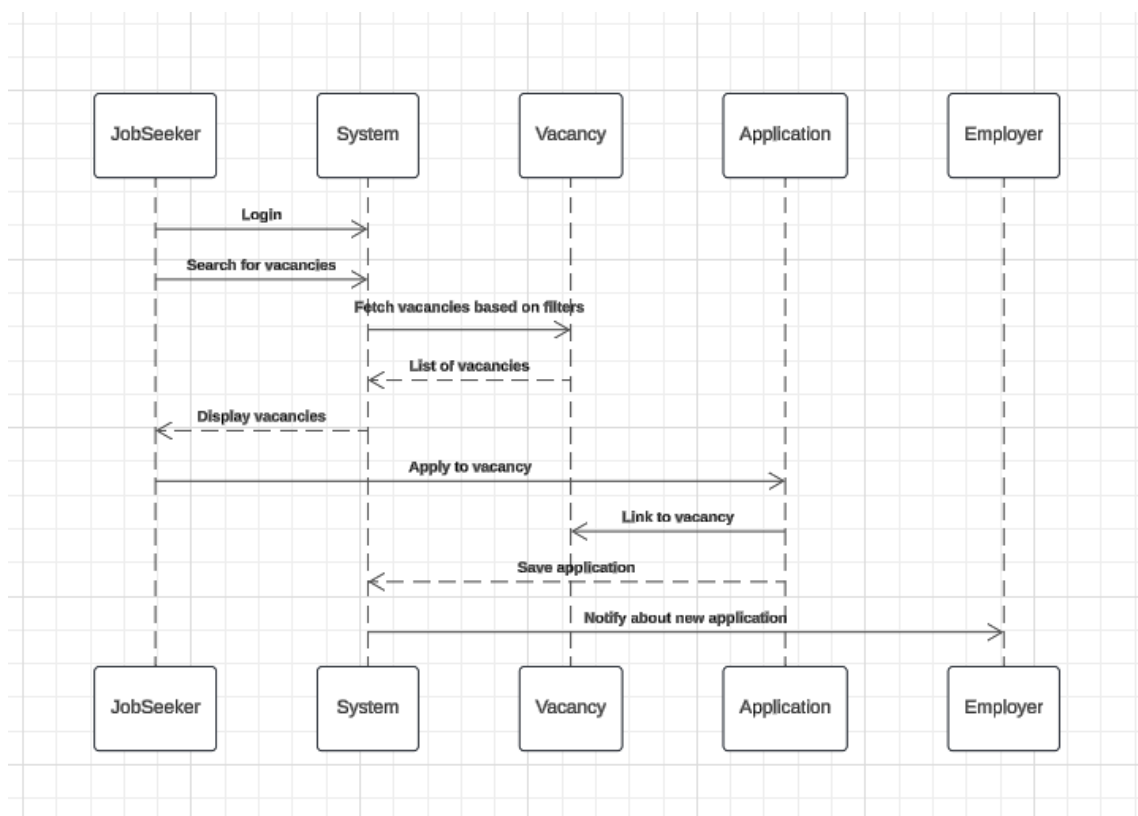


Рисунок 2.7 – Діаграма послідовності процесу подання заявки на вакансію

На даній діаграмі приведено основний потік виконання при роботі з модулем подання заявки на вакансію. Спочатку шукач роботи авторизується у системі, після чого має можливість перейти на сторінку пошуку вакансій. Користувач вводить критерії пошуку (наприклад, місто, ключові слова або

категорію), після чого система здійснює пошук відповідних вакансій на основі заданих фільтрів.

Отриманий список вакансій відображається користувачу для перегляду. Після вибору вакансії, користувач має можливість подати заявку, додавши своє резюме та супровідний лист. Система зберігає інформацію про подану заявку та встановлює зв'язок між заявкою і відповідною вакансією. Після успішного створення заявки роботодавець отримує сповіщення про надходження нового відгуку для подальшого розгляду кандидата.

2.4 Реалізація функціоналу платформи сайту з пошуку роботи з використанням штучного інтелекту.

Додаток складається з клієнтської частини, серверних функцій та панелі адміністратора. Усі частини реалізовані в одному проєкті на основі фреймворку Next.js. Серверна логіка побудована за допомогою Server Actions, що дозволяє викликати серверні функції напряму з клієнтських компонентів. Вхідною точкою додатку є компоненти маршрутизації layout.tsx та серверні функції з директивою "use server". База даних реалізована на основі MongoDB із використанням Mongoose для роботи з основними колекціями.

У даній функції реалізується підключення до бази даних MongoDB за допомогою бібліотеки mongoose. При першому виклику створюється нове з'єднання, яке кешується для подальшого використання без повторного встановлення з'єднання.

```
import mongoose, { Connection } from "mongoose";

let cachedConnection: Connection | null = null;

export async function dbConnect() {
  if (cachedConnection) {
    console.log("Using cached db connection");
    return cachedConnection;
  }
  try {
    const cnx = await mongoose.connect(process.env.MONGODB_URL!);

    cachedConnection = cnx.connection;
    console.log("New mongodb connection established");
    return cachedConnection;
  } catch (error) {
    console.log(error);
    throw error;
  }
}
```

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				40
Змн.	Арк.	№ докум.	Підпис	Дата		

```
}
```

Також у системі реалізовано middleware для контролю доступу користувачів до захищених маршрутів відповідно до їх ролі. Middleware перевіряє наявність токена авторизації, отриманого через next-auth, і звіряє роль користувача з вимогами доступу до маршруту. Якщо роль користувача не відповідає потрібній для маршруту, система виконує перенаправлення на відповідну сторінку авторизації. Повний код middleware наведено в додатку А.

```
export async function middleware(req: NextRequest) {
  const token = await getToken({ req, secret: process.env.NEXTAUTH_SECRET });

  const protectedRoutes: { [key: string]: string } = {
    "/admin": "admin",
    "/employer": "employer",
    "/jobseeker": "job_seeker",
  };

  const pathname = req.nextUrl.pathname;
  const requiredRole = Object.keys(protectedRoutes).find((route) =>
    pathname.startsWith(route)
  );

  if (
    pathname.startsWith("/admin/signin") ||
    pathname.startsWith("/employer/signin") ||
    pathname.startsWith("/jobseeker/signin") ||
    pathname.startsWith("/employer/register") ||
    pathname.startsWith("/jobseeker/register")
  ) {
    return NextResponse.next();
  }

  if (requiredRole) {
    const userRole = protectedRoutes[requiredRole];

    if (!token || token.role !== userRole) {
      const redirectUrl = new URL(`/${userRole}/signin`, req.url);
      return NextResponse.redirect(redirectUrl);
    }
  }

  return NextResponse.next();
}
```

Далі розглянемо функціонал подання та обробки заявок шукачів роботи на вакансії, було створено окремий серверний модуль, що використовує можливості Server Actions у Next.js.

```
"use server";
```

```
import { dbConnect } from "@config/db";
```

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

```

import ApplicationModel, { IApplication } from "@models/Application";
import { authOptions } from "@app/api/auth/[...nextauth]/route";
import { getServerSession } from "next-auth";
import { IApplicationCreate } from "@types/application";

export async function createApplication(data: IApplicationCreate) {
  try {
    await dbConnect();

    const session = await getServerSession(authOptions);

    if (!session || !session.user) {
      throw new Error("User is not authenticated");
    }

    const newApplication = new ApplicationModel({
      ...data,
      applicant: session.user.id,
    });

    await newApplication.save();

    return {
      success: true,
      message: "Application submitted successfully",
    };
  } catch (error) {
    return {
      success: false,
      error: error instanceof Error ? error.message : "Unknown error",
    };
  }
}

export async function getApplicationById(applicationId: string) {
  await dbConnect();
  return await ApplicationModel.findById(applicationId);
}

export async function updateApplication(
  applicationId: string,
  data: Partial<Omit<IApplication, "applicant">>
) {
  await dbConnect();

  const session = await getServerSession(authOptions);

  if (!session || !session.user) {
    throw new Error("User is not authenticated.");
  }

  const application = await ApplicationModel.findById(applicationId);
  if (application?.applicant.toString() !== session.user.id) {
    throw new Error("You are not authorized to update this application.");
  }

  return await ApplicationModel.findByIdAndUpdate(applicationId, data, {
    new: true,
  });
}

```

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

```

export async function deleteApplication(applicationId: string) {
    await dbConnect();

    const session = await getServerSession(authOptions);

    if (!session || !session.user) {
        throw new Error("User is not authenticated.");
    }

    const application = await ApplicationModel.findById(applicationId);
    if (application?.applicant.toString() !== session.user.id) {
        throw new Error("You are not authorized to delete this application.");
    }

    return await ApplicationModel.findByIdAndDelete(applicationId);
}

export async function getApplicationsByJobVacancyId(vacancyId: string) {
    try {
        await dbConnect();

        const applications = await ApplicationModel.find({
            jobVacancy: vacancyId,
        })
        .populate("applicant", "firstName lastName email")
        .populate("jobVacancy", "title")
        .lean();

        if (!applications || applications.length === 0) {
            return {
                success: false,
                error: "No applications found for this job vacancy.",
            };
        }

        const plainApplications = applications.map((application) => ({
            ...application,
            _id: application._id.toString(),
            applicant: {
                ...application.applicant,
                _id: application.applicant._id.toString(),
            },
            jobVacancy: {
                ...application.jobVacancy,
                _id: application.jobVacancy._id.toString(),
            },
            appliedAt: application.appliedAt.toISOString(),
        }));

        return {
            success: true,
            data: plainApplications,
        };
    } catch (error) {
        return {
            success: false,
            error:
                error instanceof Error ? error.message : "An unknown error occurred",
        };
    }
}

```

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

```
}  
}
```

Для реалізації функціоналу подання та обробки заявок шукачів роботи на вакансії було створено окремий серверний модуль, що використовує можливості Server Actions у Next.js.

У цьому файлі описані всі основні функції для роботи з заявками, а саме `createApplication`, `getApplicationById`, `updateApplication`, `deleteApplication` та `getApplicationsByJobVacancyId`. Функція `createApplication` використовується для створення нової заявки, де автоматично прив'язується ідентифікатор авторизованого користувача та зберігається інформація у колекції бази даних. `getApplicationById` дозволяє отримати детальну інформацію про конкретну заявку за її унікальним ідентифікатором, тоді як `updateApplication` забезпечує можливість редагування заявки лише її автором після проходження перевірки прав доступу. Видалення заявки реалізовано у функції `deleteApplication`, що також передбачає перевірку авторизації користувача. Функція `getApplicationsByJobVacancyId` виконує пошук усіх заявок на певну вакансію та збагачує їх даними про заявника і вакансію через механізм `populate`, повертаючи їх у зручному для обробки форматі. Усі функції модуля працюють із попереднім встановленням з'єднання з базою даних за допомогою `dbConnect`, що забезпечує стабільну роботу із збереженням кешованого з'єднання.

Для реалізації механізму аутентифікації користувачів у системі використано бібліотеку `NextAuth` з провайдером `CredentialsProvider`. У цьому модулі описана логіка обробки даних для входу в систему через електронну пошту, пароль та роль користувача. Функція `authorize` виконує підключення до бази даних `MongoDB`, знаходить користувача за введеною електронною поштою, перевіряє правильність пароля за допомогою бібліотеки `bcrypt`, а також звіряє роль користувача для запобігання несанкціонованому доступу. При успішній аутентифікації створюється JWT-токен, який містить основну інформацію про користувача (ID, ім'я, прізвище, email та роль) і використовується для управління сесією під час подальшої роботи з сайтом. Повний код реалізації механізму аутентифікації наведено в додатку А.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

```

export const authOptions: NextAuthOptions = {
  providers: [
    CredentialsProvider({
      name: "Credentials",
      credentials: {
        email: { label: "Email", type: "email" },
        password: { label: "Password", type: "password" },
        role: { label: "Role", type: "text" },
      },
      async authorize(credentials) {
        if (!credentials) {
          return null;
        }

        await dbConnect();

        const user = await UserModel.findOne({ email: credentials.email });

        if (user) {
          const isPasswordValid = await bcrypt.compare(
            credentials.password,
            user.password
          );
          if (isPasswordValid) {
            if (user.role !== credentials.role) {
              throw new Error("Role mismatch");
            }

            return {
              id: user._id.toString(),
              email: user.email,
              firstName: user.firstName,
              lastName: user.lastName,
              role: user.role,
            };
          }
        }
        return null;
      },
    }),
  ],
  ...

```

Для реалізації функціоналу рекомендацій та аналізу резюме з боку роботодавців у системі були створені окремі серверні кінцеві точки, побудовані на основі Next.js із використанням Google Gemini 2.0 та AI SDK. У структурі проєкту, а саме в папці `app/api`, реалізовано два основні маршрути. Один із них відповідає за аналіз відповідності конкретного резюме вимогам окремої вакансії, тоді як другий дозволяє порівнювати кілька резюме з метою визначення найвідповіднішого кандидата. Всі запити обробляються виключно на сервері, що гарантує безпечну роботу з персональними даними користувачів.

		Семериков Н.О			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Резюме передаються у форматі PDF, після чого модель штучного інтелекту аналізує їх зміст, визначає відповідність зазначеним критеріям і формує рекомендації. Ці рекомендації можуть бути використані як для покращення шансів пошукачів на успішне працевлаштування, так і для підтримки роботодавців у процесі ухвалення рішення щодо кандидатів. Повна реалізація серверної інтеграції з моделлю штучного інтелекту наведена в додатку А.

```
let { messages, id, jobVacancyId } = await req.json();

if (!jobVacancyId) {
  const chat = await getChatById(id);

  if (!chat) {
    throw new Error("Chat not found");
  }
  jobVacancyId = chat.jobVacancy;
}

const [vacancy, jobVacancyApplications] = await Promise.all([
  getJobVacancyById(jobVacancyId),
  getApplicationsByJobVacancyId(jobVacancyId),
]);

if (!vacancy.success || !vacancy.data) {
  throw new Error("Vacancy not found");
}
if (
  !jobVacancyApplications.success ||
  !jobVacancyApplications.data?.length
) {
  throw new Error("No resumes found");
}

const resumeFiles = jobVacancyApplications.data
  .filter((application) => application.resumeUrl)
  .map((application) => ({
    type: "file",
    data: new URL(application.resumeUrl),
    mimeType: "application/pdf",
  }));

const requestMessages = [
  {
    role: "system",
    content:
      "Ти фахівець з управління персоналом. Проаналізуй надані резюме для цієї вакансії і вкажи, який кандидат найкраще підходить.",
  },
  {
    role: "user",
    content: `Назва вакансії: "${vacancy.data.title}".
      Опис: "${vacancy.data.description}".
      Вимоги: "${vacancy.data.requirements.join(", ")}".`,
  },
]
```

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

```

    role: "user",
    content: [
      { type: "text", text: "These are the submitted resumes:" },
      ...resumeFiles,
    ],
  },
  ...messages,
];

const result = streamText({
  model: google("gemini-1.5-pro-latest"),
  messages: requestMessages,
  experimental_generateMessageId: createIdGenerator({
    prefix: "msgs",
    size: 16,
  }),
  async onFinish({ response }) {
    await saveChat({
      id,
      messages: appendResponseMessages({
        messages: messages,
        responseMessages: response.messages,
      }),
    });
  },
});

```

Висновки до другого розділу.

У другому розділі було здійснено проектування та реалізацію основних компонентів сайту з пошуку роботи з використанням штучного інтелекту. Було визначено варіанти використання системи для різних ролей користувачів, розроблено структуру бази даних на основі MongoDB та побудовано основні модулі клієнтської та серверної частин за допомогою Next.js і Server Actions. Реалізовано ключовий функціонал для реєстрації, пошуку вакансій, подання заявок, управління вакансіями та інтеграції модуля штучного інтелекту для рекомендацій і аналізу резюме. Створена архітектура системи забезпечує гнучкість, масштабованість та високу якість користувацького досвіду, що є основою для подальшого розвитку та удосконалення платформи.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ ДОДАТКУ

1.1 Порядок встановлення та налаштування системи.

Діаграма розгортання (рис. 3.1) демонструє основну інфраструктуру системи під час її встановлення, включаючи логічну взаємодію між апаратними вузлами та програмними компонентами. Такий тип діаграми дає змогу візуалізувати, як різні частини веб застосунку розміщуються та взаємодіють у реальному середовищі виконання. На наведеній схемі представлено три основні вузли — робоча станція користувача, веб-сервер та сервер бази даних. Користувач взаємодіє з вебзастосунком через браузер, встановлений на локальному пристрої.

Веб-сервер відповідає за обробку запитів і складається з кількох функціональних модулів — презентаційного шару, що відповідає за інтерфейс, обробника API-запитів, модуля інтеграції з системою штучного інтелекту Gemini AI, а також модуля ведення журналів подій. Програмна частина веб-сервера реалізована з використанням фреймворку Next.js, який підтримує серверний рендеринг та обробку API-запитів в одному середовищі.

Для зберігання даних використовується хмарна база даних MongoDB Atlas, яка з'єднується з сервером за допомогою TCP/IP-протоколу. Саме тут зберігаються основні сутності системи — облікові записи користувачів, вакансії, резюме, листування тощо.

Інтеграція з Gemini AI API дозволяє реалізувати інтелектуальні функції, такі як генерація порад, аналіз змісту резюме або оцінка релевантності вакансій. Зв'язок із цим сервісом також відбувається на стороні сервера через програмний інтерфейс.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

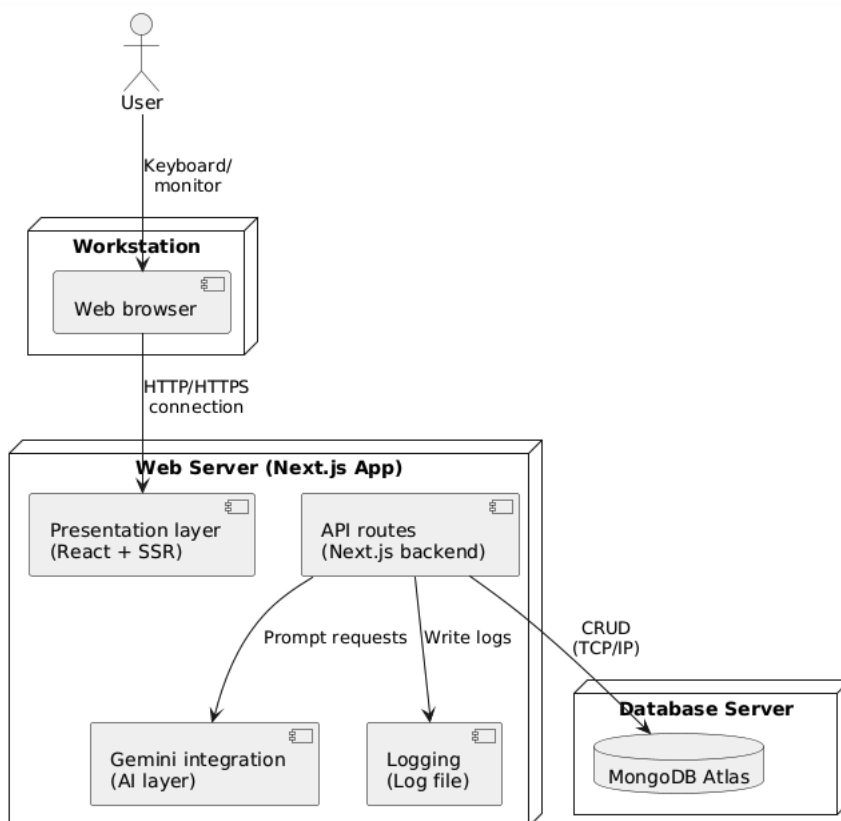


Рисунок 3.1 – Діаграма розгортання системи “Платформи для пошуку роботи з використанням ШІ”

3.2 Структура інтерфейсу та порядок взаємодії з платформою з пошуку роботи.

Для користування платформою необхідно мати пристрій з доступом до Інтернету та браузером. Інтерфейс адаптований під різні екрани, тому зручний як на комп’ютерах, так і на мобільних телефонах.

На головній сторінці відображається банер із заголовком та пошуком, де можна ввести назву посади або компанії, вибрати місто й натиснути кнопку «Знайти вакансії». У верхньому меню доступні переходи до розділів «Вакансії», «Компанії», «Додати вакансію», а також кнопки «Увійти» та «Зареєструватися». Натискання на логотип HireZone повертає користувача на головну сторінку. Нижче розміщено популярні категорії вакансій, наприклад, Технології, Фінанси, Маркетинг, з відповідною кількістю доступних пропозицій. У кінці сторінки відображається коротка статистика — кількість вакансій, компаній, пошукачів та працевлаштувань.

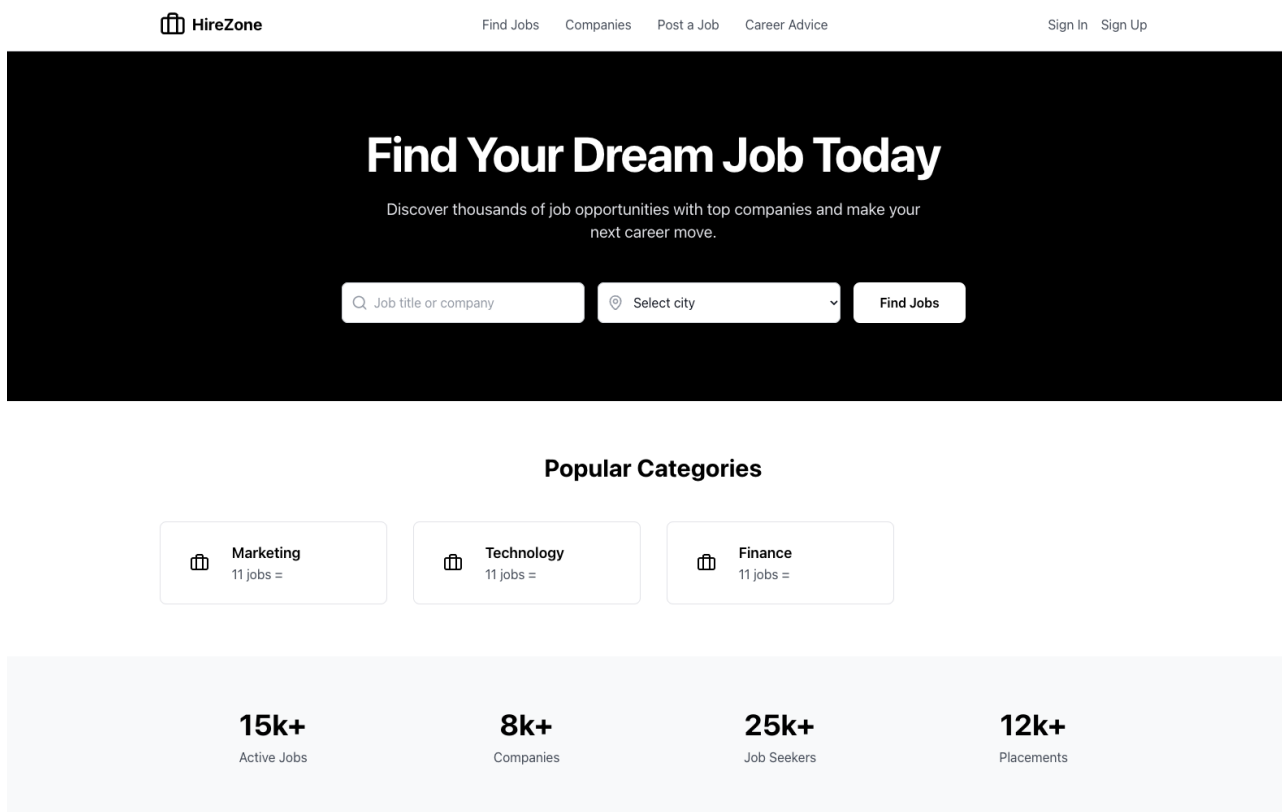


Рисунок 3.2 – Головна сторінка платформи HireZone

Користувачі платформи поділяються на три основні ролі — кандидат, роботодавець та адміністратор. Для кожної з них реалізовано окрему сторінку входу, що забезпечує зручність у використанні та дозволяє чітко розмежувати доступ до функціоналу. Шукачі роботи використовують стандартну сторінку авторизації, яка містить поля для введення електронної пошти та пароля, а також посилання на сторінку реєстрації у випадку відсутності облікового запису. На цій же сторінці доступне посилання для переходу до форми входу для роботодавців. Останні мають власну сторінку авторизації з додатковою опцією запам'ятовування користувача та можливістю створення нового облікового запису роботодавця. Адміністратори системи також мають окрему форму входу, яка містить базові поля авторизації, а також посилання на службу підтримки для вирішення можливих технічних питань. Така структура дозволяє ефективно організувати роботу платформи відповідно до потреб кожної категорії користувачів, забезпечуючи інтуїтивно зрозумілий інтерфейс і чіткий поділ прав доступу.

		Семериков Н.О.			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

 HireZone

Sign in to your account

Or create a new account

Email address

 you@example.com

Password



Don't have an account? [Register](#)

Sign in

Are you an employer? [Sign in here](#)

Рисунок 3.4 – Сторінка входу для шукача роботи

 HireZone

[Find Jobs](#) [Companies](#)

[Sign In](#) [Sign Up](#)

 HireZone

Employer Sign In

Email address

 admin@example.com

Password



☐ Remember me

Sign in

[Don't have an account? Register as Employer](#)

Рисунок 3.5 – Сторінка входу для роботодавця

Admin Sign In

Email address

 admin@example.com

Password



Sign in

[Need help? Contact support](#)

Рисунок 3.6 – Сторінка входу для адміністратора

У разі відсутності облікового запису, користувач може пройти реєстрацію на відповідній сторінці. Для шукача роботи передбачена форма, де потрібно вказати ім'я, прізвище, електронну пошту та пароль. Після реєстрації він отримує доступ до функцій пошуку вакансій і подачі заявок. Роботодавці реєструються окремо через іншу форму, яка також містить поля для особистих даних і корпоративної пошти. Такий поділ дозволяє відразу визначити тип користувача та надати йому доступ до відповідного функціоналу.

HireZone

Create your account

Or sign in to your account

First Name

Last Name

Email address

Password

Register

————— Already have an account? Sign in —————

Рисунок 3.7 – Реєстрація шукача роботи

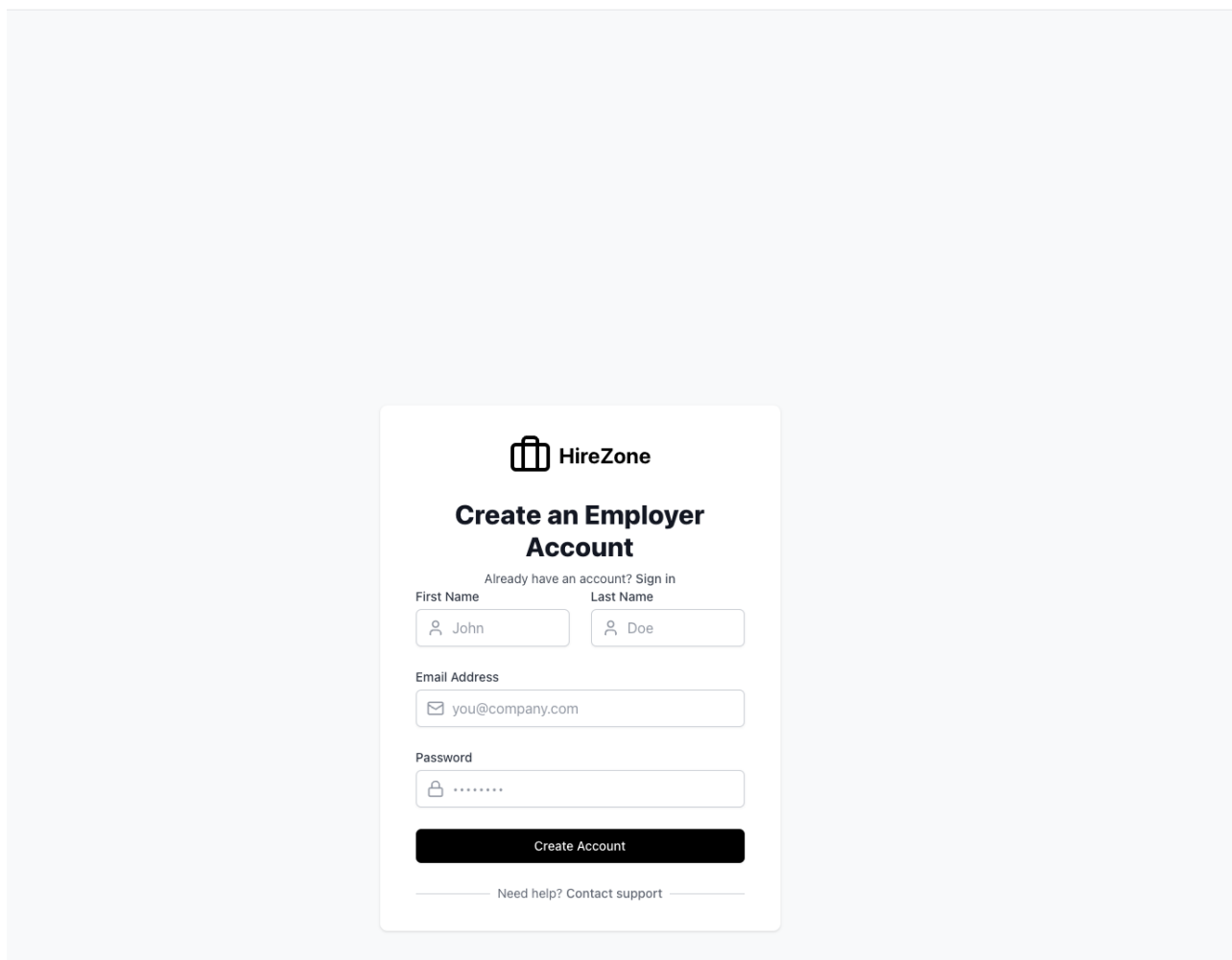


Рисунок 3.8 – Реєстрація роботодавця

Після реєстрації роботодавець автоматично перенаправляється на сторінку створення компанії. Для продовження роботи з платформою він повинен заповнити форму з основною інформацією про компанію, включаючи назву, опис діяльності та веб-сайт. Лише після створення профілю компанії відкривається доступ до публікації вакансій та інших можливостей кабінету роботодавця.

[Find Jobs](#)
[Companies](#)
[Post a Job](#)
[Dashboard](#)

[Welcome, Test!](#)
[Sign Out](#)

Create Your Company Profile

Set up your company profile to start posting job opportunities.

Company Name

Company Description

Tell us about your company...

Company Website

Create Company

Need help? Contact support

Рисунок 3.9 – Форма створення профілю компанії

Після створення компанії роботодавець автоматично потрапляє до дашборду, де може керувати вакансіями, оновлювати профіль компанії або повністю видалити її. Якщо вакансій ще не створено, відображається повідомлення з пропозицією додати першу пропозицію. Для цього необхідно натиснути кнопку «Створити вакансію», після чого відкривається форма заповнення — з полями для назви посади, опису, місцезнаходження, вимог і бажаного рівня заробітної плати. Кожна створена вакансія відображається у списку з можливістю перегляду, редагування або видалення. Усі дії супроводжуються діалоговими вікнами підтвердження — зокрема, під час спроби видалити вакансію чи саму компанію система запитує підтвердження, щоб уникнути випадкових втрат даних. Редагування профілю компанії також доступне — через кнопку «Оновити компанію», що дозволяє змінити назву, опис і веб-сайт організації. Таким чином, дашборд роботодавця забезпечує повний цикл управління компанією та її вакансіями в межах платформи, даючи змогу гнучко реагувати на зміну потреб у персоналі.

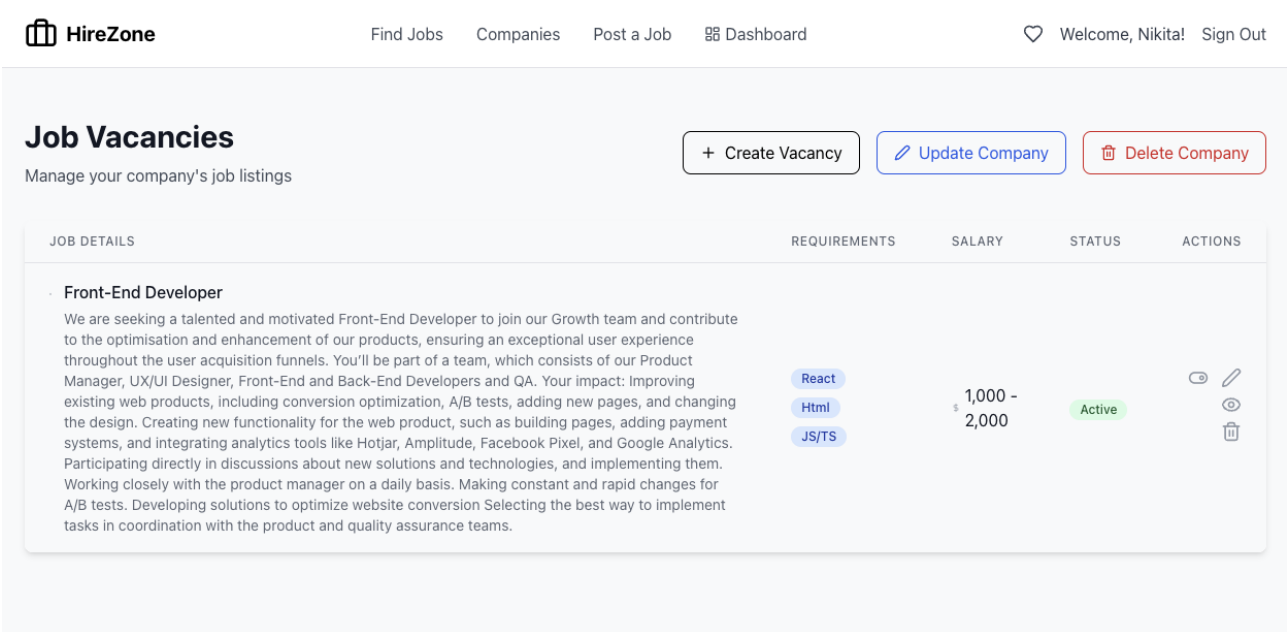


Рисунок 3.10 – Дашборд компанії після створення профілю

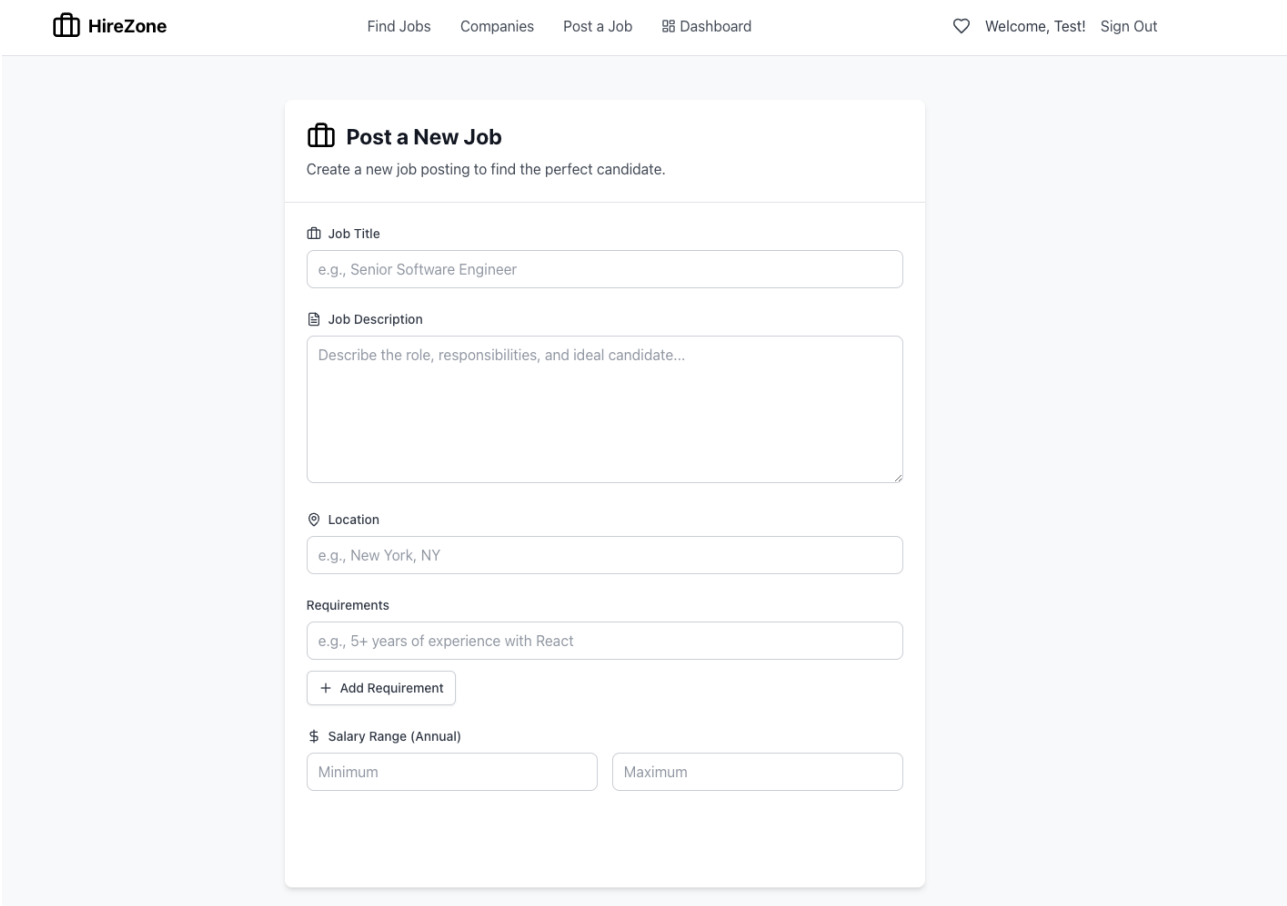


Рисунок 3.11 – Форма створення вакансії

Update Company Profile

Keep your company information up to date to attract the best candidates.

Company Name

HireZone Development

Company Description

HireZone is a modern job search platform that connects job seekers with relevant opportunities through smart technology and user-friendly tools.

Our mission is to simplify and accelerate the job search process for everyone — whether you're just starting your career or looking for your next big opportunity.

Company Website

https://www.hire-zone.ua/

Update Company

Рисунок 3.12 – Редагування профілю компанії

Job Vacancies

Manage your company's job listings

+ Create Vacancy

Update Company

Delete Company

JOB DETAILS

REQUIREMENTS

SALARY

STATUS

ACTIONS

Front-End Developer

We are seeking a talented and motivated Front-End Developer to join our Growth team and contribute to the optimisation and enhancement of our products, ensuring an exceptional user experience throughout the user acquisition funnels. You'll be part of a team, which consists of our Product Manager, UX/UI Designer, Front-End and Back-End Developers and QA. Your impact: Improving existing web products, including conversion optimization, A/B tests, adding new pages, and changing the design. Creating new functionality for the web product, such as building pages, adding payment systems, and integrating analytics tools like Hotjar, Amplitude, Facebook Pixel, and Google Analytics. Participating directly in discussions about new solutions and technologies, and implementing them. Working closely with the product manager on a daily basis. Making constant and rapid changes for A/B tests. Developing solutions to optimize website conversion. Selecting the best way to implement tasks in coordination with the product and quality assurance teams.

React

Html

JS/TS

1,000 -
2,000

Active



Delete Company

Are you sure you want to delete your company? This will permanently remove all job vacancies and company data. This action cannot be undone.

Cancel

Delete Company

		Семериков Н.О.		
		Громський О.О.		
Змн.	Арк.	№ докум.	Підпис	Дата

Рисунок 3.13 – Підтвердження видалення компанії

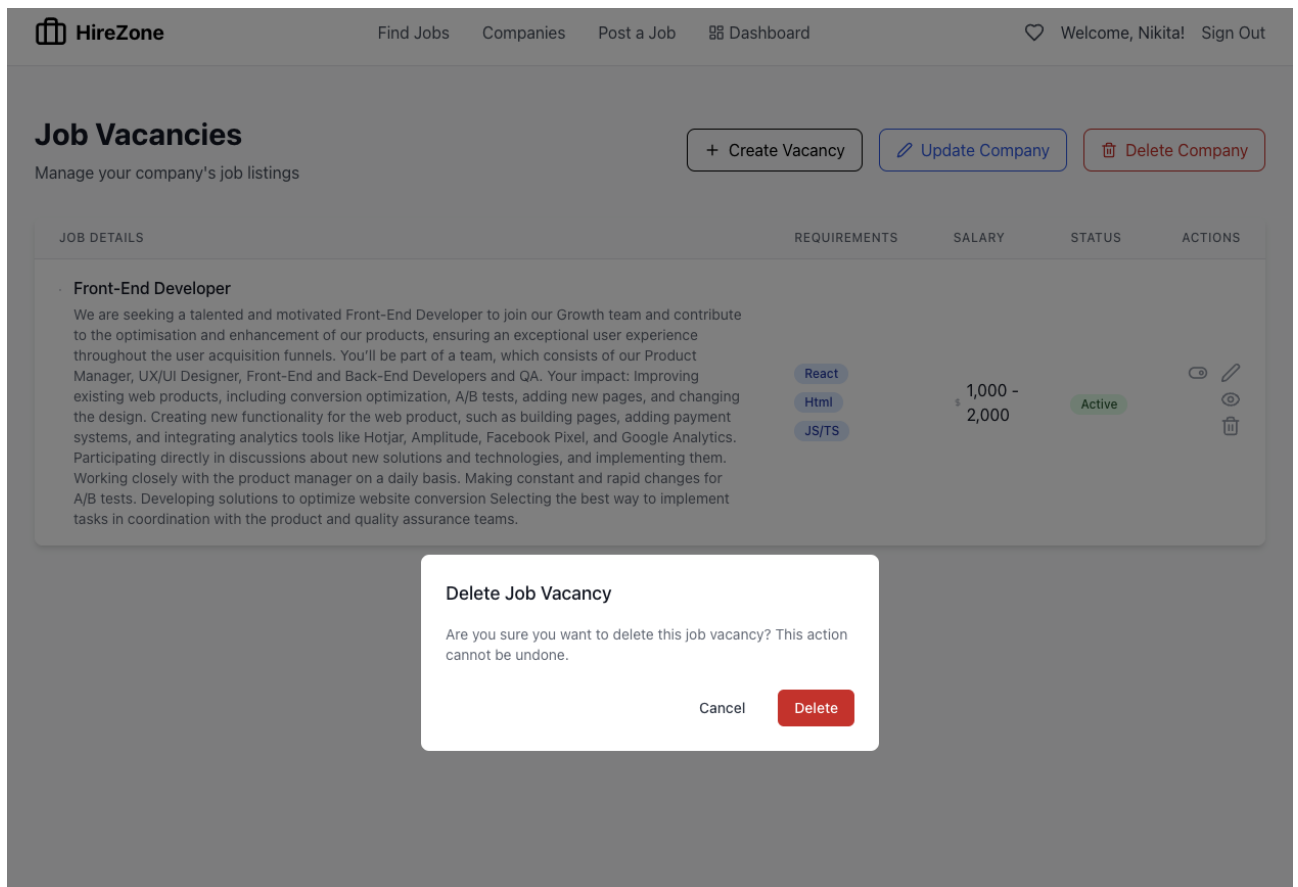


Рисунок 3.14 – Підтвердження видалення вакансії

Після входу в систему кандидат потрапляє до особистого кабінету, де відображається привітання з ім'ям користувача та його електронною поштою. Основним блоком є список відгуків на вакансії, поданих через платформу. Кожна заявка містить назву посади, дату подачі (у разі наявності) та поточний статус — наприклад, “Очікує розгляду”. Також доступна кнопка «Редагувати профіль», яка дозволяє оновити особисті дані. Цей кабінет надає користувачеві змогу відслідковувати свою активність і оперативно реагувати на зміни в статусі заявок.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

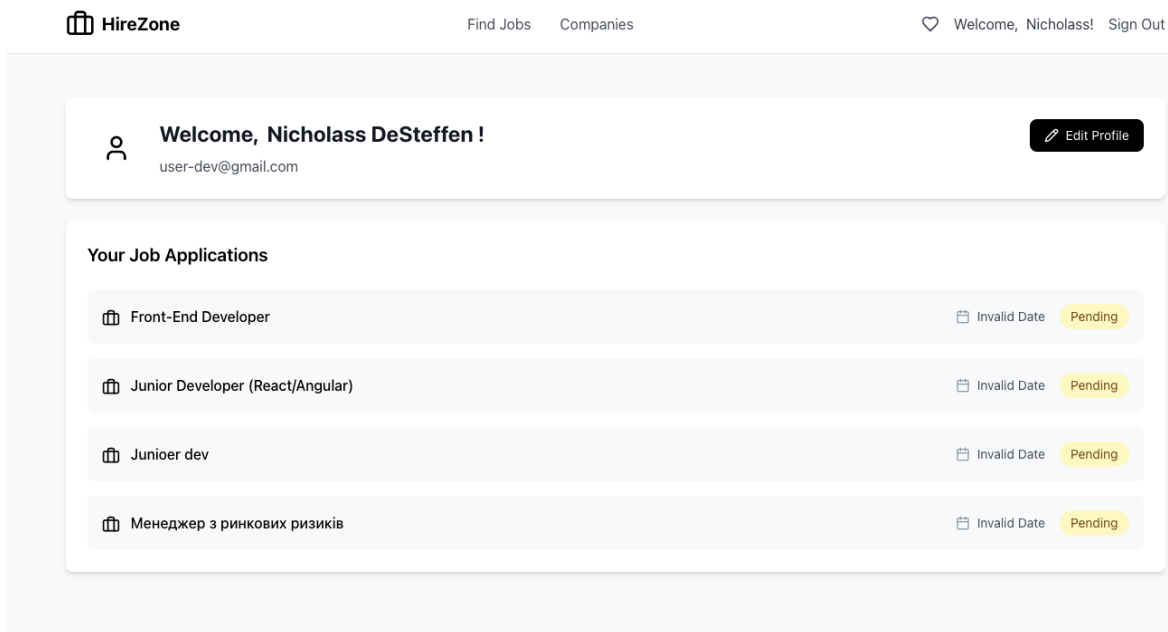


Рисунок 3.15 – Особистий кабінет шукача роботи

Для перегляду доступних вакансій користувач переходить до розділу «Вакансії» через головне меню. На сторінці відображається каталог з усіма активними пропозиціями. Доступна фільтрація за ключовими словами, містом, типом роботи та діапазоном заробітної плати. Кожна вакансія представлена у вигляді картки, яка містить назву посади, компанію, рівень заробітної плати, дату публікації, короткий опис і ключові вимоги. Для перегляду повної інформації про вакансію необхідно натиснути кнопку «Переглянути деталі», після чого відкривається розгорнута сторінка з можливістю подати заявку.

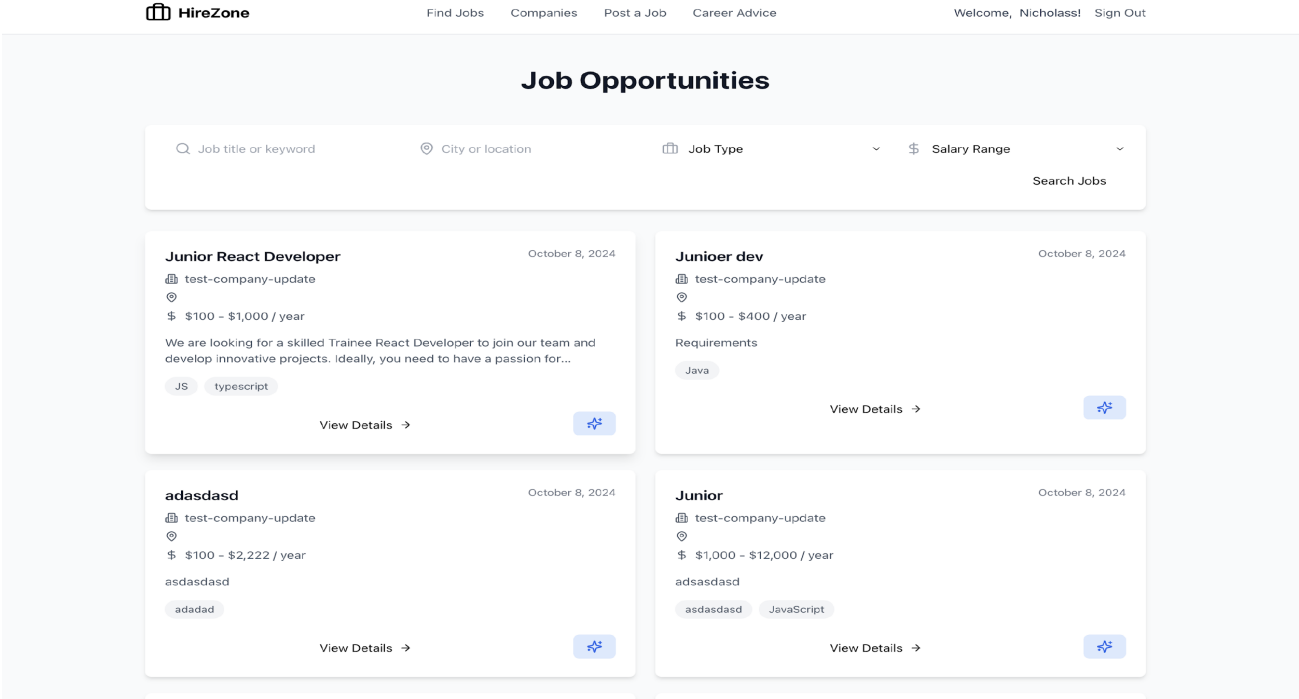


Рисунок 3.16 – Сторінка перегляду вакансій

Додатково на сторінці з вакансіями реалізована функція інтелектуального підбору. При натисканні на відповідну кнопку поруч із вакансією відкривається модальне вікно з порадами від ШІ, згенерованими на основі резюме користувача. Система аналізує досвід, навички та рівень кваліфікації кандидата й надає рекомендації щодо доцільності відгуку на конкретну вакансію. Якщо обрана посада не відповідає профілю користувача, ШІ пояснює чому та пропонує більш релевантні ролі. Це підвищує ефективність пошуку і допомагає уникнути подачі заявок на невідповідні позиції.

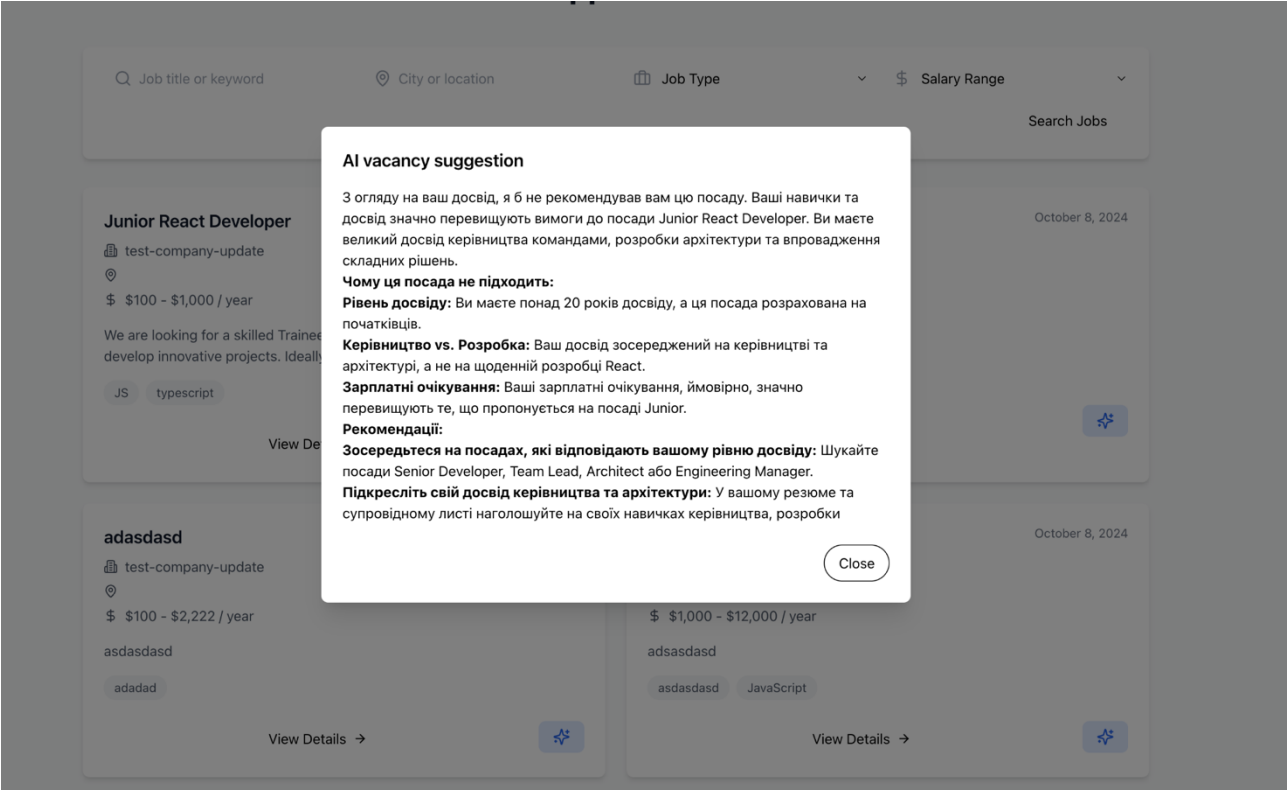


Рисунок 3.17 – Вікно рекомендацій ШІ для конкретної вакансії

На сторінці вакансії користувач може детально ознайомитися з умовами пропозиції — назвою посади, містом, заробітною платою, вимогами та описом роботи. В окремому блоці відображається інформація про компанію — її назва, сайт та короткий опис діяльності. Над описом вакансії розміщено кнопки «Подати заявку» та «Додати в обране» — натискання на першу відкриває процес подачі, а друга дозволяє зберегти вакансію для подальшого перегляду. Така структура забезпечує зручний доступ до всієї необхідної інформації та швидку взаємодію з платформою.

Front-End Developer

Submit Application

Add to Favorites

Zhytomyr

\$ \$1,000 - \$2,000 / year

Job Description

We are seeking a talented and motivated Front-End Developer to join our Growth team and contribute to the optimisation and enhancement of our products, ensuring an exceptional user experience throughout the user acquisition funnels. You'll be part of a team, which consists of our Product Manager, UX/UI Designer, Front-End and Back-End Developers and QA.

Your impact:

Improving existing web products, including conversion optimization, A/B tests, adding new pages, and changing the design.

Creating new functionality for the web product, such as building pages, adding payment systems, and integrating analytics tools like Hotjar, Amplitude, Facebook Pixel, and Google Analytics.

Participating directly in discussions about new solutions and technologies, and implementing them.

Working closely with the product manager on a daily basis.

Making constant and rapid changes for A/B tests.

Developing solutions to optimize website conversion

Selecting the best way to implement tasks in coordination with the product and quality assurance teams.

Company Details

Company Name
HireZone Development

Website
<https://www.hire-zone.ua/>

Description

HireZone is a modern job search platform that connects job seekers with relevant opportunities through smart technology and user-friendly tools. Our mission is to simplify and accelerate the job search process for everyone — whether you're just starting your career or looking for your next big opportunity. We've integrated AI-powered features that analyze your CV and match it with job listings that best fit your skills, experience, and career goals. With real-time feedback and personalized job suggestions, you'll always know where you stand and how to improve your chances.

Requirements

- React
- Html
- JS/TS

Рисунок 3.18 – Сторінка перегляду вакансії

Після ознайомлення з описом вакансії користувач може подати заявку, натиснувши кнопку «Подати заявку», що відкриває відповідну форму. У формі кандидат має можливість додати супровідний лист, обрати або завантажити резюме у форматі PDF, DOC чи DOCX (до 5 МБ) та натиснути «Надіслати заявку». Інтерфейс подачі простий і зрозумілий, що дозволяє користувачам швидко й зручно взаємодіяти з платформою без зайвих кроків.


Find Jobs
Companies
Welcome, Nicholas! Sign Out


Apply for Front-End Developer

HireZone Development

About the Position

We are seeking a talented and motivated Front-End Developer to join our Growth team and contribute to the optimisation and enhancement of our products, ensuring an exceptional user experience throughout the user acquisition funnels. You'll be part of a team, which consists of our Product Manager, UX/UI Designer, Front-End and Back-End Developers and QA. Your impact: Improving existing web products, including conversion optimization, A/B tests, adding new pages, and changing the design. Creating new functionality for the web product, such as building pages, adding payment systems, and integrating analytics tools like Hotjar, Amplitude, Facebook Pixel, and Google Analytics. Participating directly in discussions about new solutions and technologies, and implementing them. Working closely with the product manager on a daily basis. Making constant and rapid changes for A/B tests. Developing solutions to optimize website conversion. Selecting the best way to implement tasks in coordination with the product and quality assurance teams.


Cover Letter

Tell us why you're the perfect fit for this position...

Select Existing Resume

Upload new resume

Upload New Resume



 Click to upload or drag and drop
 PDF, DOC, or DOCX (Max 5MB)

Submit Application

Рисунок 3.19 – Форма подачі заявки на вакансію

Після заповнення форми та натискання кнопки «Надіслати заявку» система виводить повідомлення про успішну подачу. Користувач бачить підтвердження “Success!”, після чого може повернутися на головну сторінку. Це інформує пошукача про те, що заявка була прийнята й передана роботодавцю для розгляду.

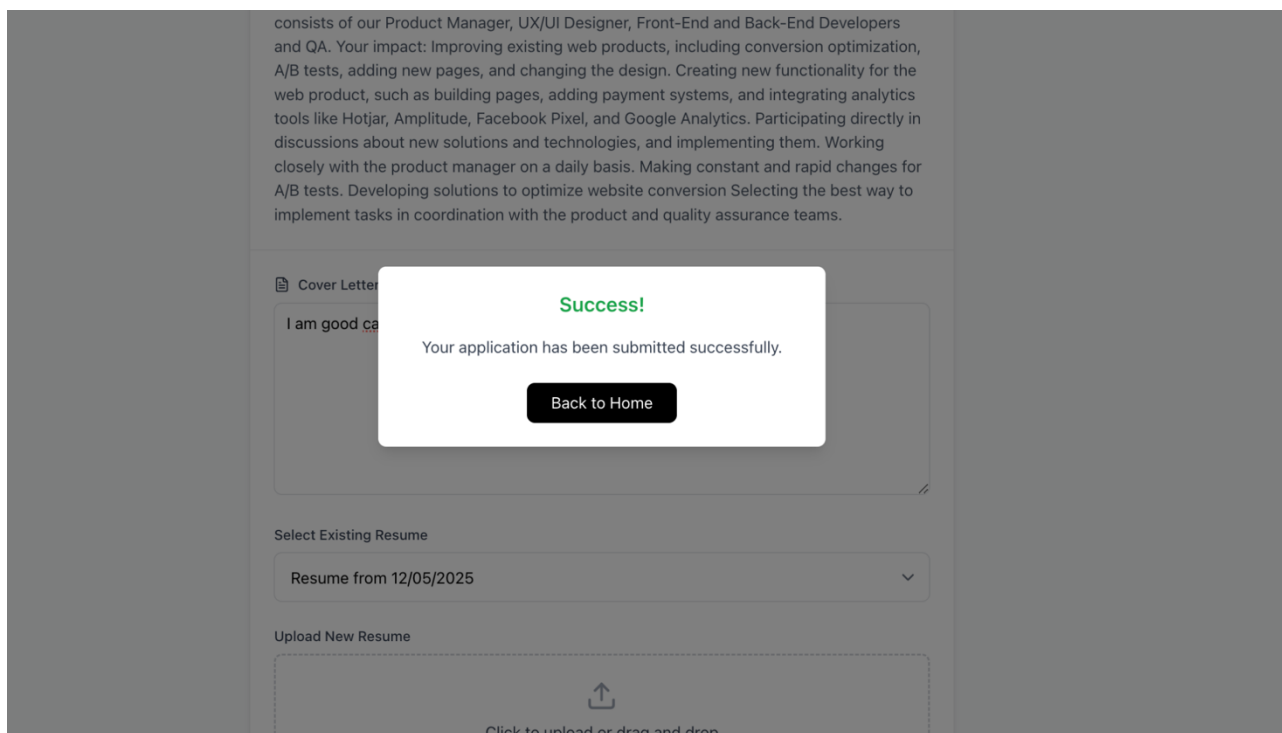


Рисунок 3.20 – Повідомлення про успішну подачу заявки

У свою чергу, роботодавець може переглянути всі відгуки кандидатів на розміщені вакансії. На сторінці перевірки заявок відображається ім'я кандидата, його електронна пошта, супровідний лист і посилання на перегляд резюме. Біля кожної заявки зазначається поточний статус — наприклад, “Очікує розгляду”. Такий формат дозволяє швидко ознайомитись з профілем кандидата та приймати подальші рішення щодо співбесіди чи відхилення.

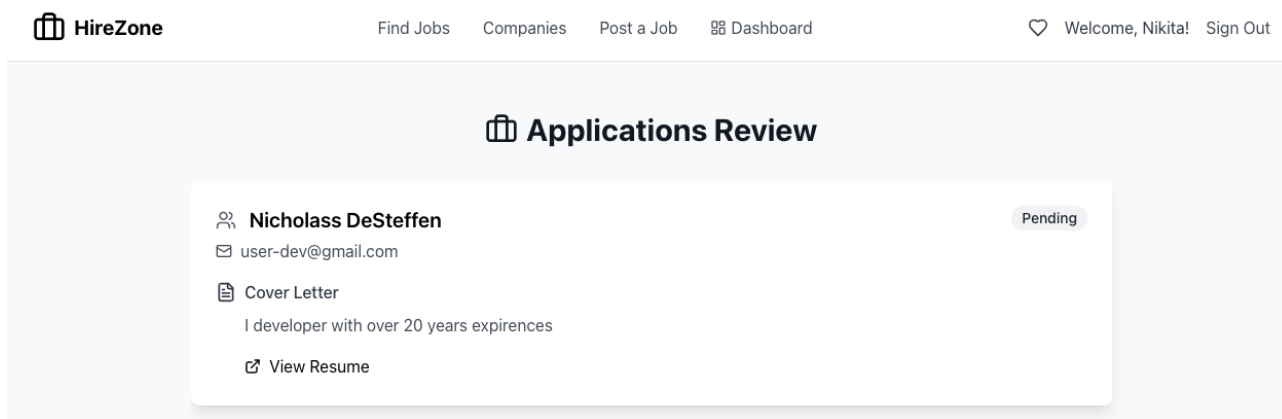


Рисунок 3.21 – Перегляд заявки кандидата роботодавцем

Додатково, у дашборді роботодавця реалізовано інтеграцію з ІІІ-асистентом, який дозволяє проводити попередній аналіз кандидатів. Після отримання відгуків на вакансію роботодавець може звернутися до чату,

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				64
Змн.	Арк.	№ докум.	Підпис	Дата		

надіслати резюме кандидата та поставити запитання, наприклад «Що ти можеш сказати про цього кандидата?». Система проводить автоматичний аналіз досвіду, навичок і відповідності вимогам вакансії, виокремлює сильні та слабкі сторони, дає загальний висновок і надає конкретні рекомендації щодо питань, які варто задати на співбесіді. Це допомагає швидше оцінити потенційного працівника без потреби в довгому ручному аналізі.

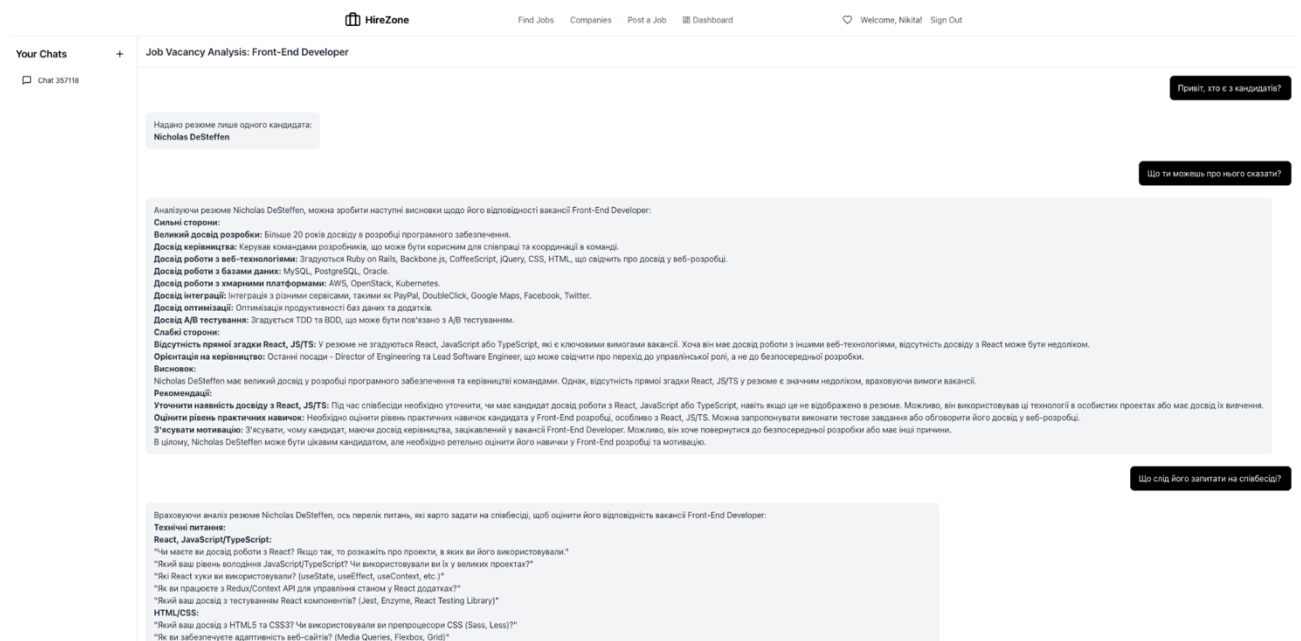


Рисунок 3.22 – Аналіз резюме кандидата у чаті з ШІ

Адміністратор має окремий доступ до панелі керування, яка забезпечує повний контроль над усіма ключовими елементами системи. У секції «Огляд» відображається актуальна статистика — кількість користувачів, компаній, опублікованих вакансій та поданих заявок. Панель дозволяє адміністратору переглядати список зареєстрованих користувачів, змінювати їхні ролі або видаляти облікові записи за потреби. У розділі «Компанії» адміністратор бачить усі створені компанії разом із вказаними сайтами та власниками, а також має можливість видаляти небажані записи. Окремий розділ «Вакансії» надає повний контроль над усіма активними публікаціями на платформі. Крім цього, через вкладку «Категорії» адміністратор може керувати переліком категорій вакансій — створювати нові, редагувати наявні або видаляти застарілі. Уся

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				65
Змн.	Арк.	№ докум.	Підпис	Дата		

адміністративна панель спроектована з акцентом на зручність керування та забезпечення швидкого доступу до основних функцій.

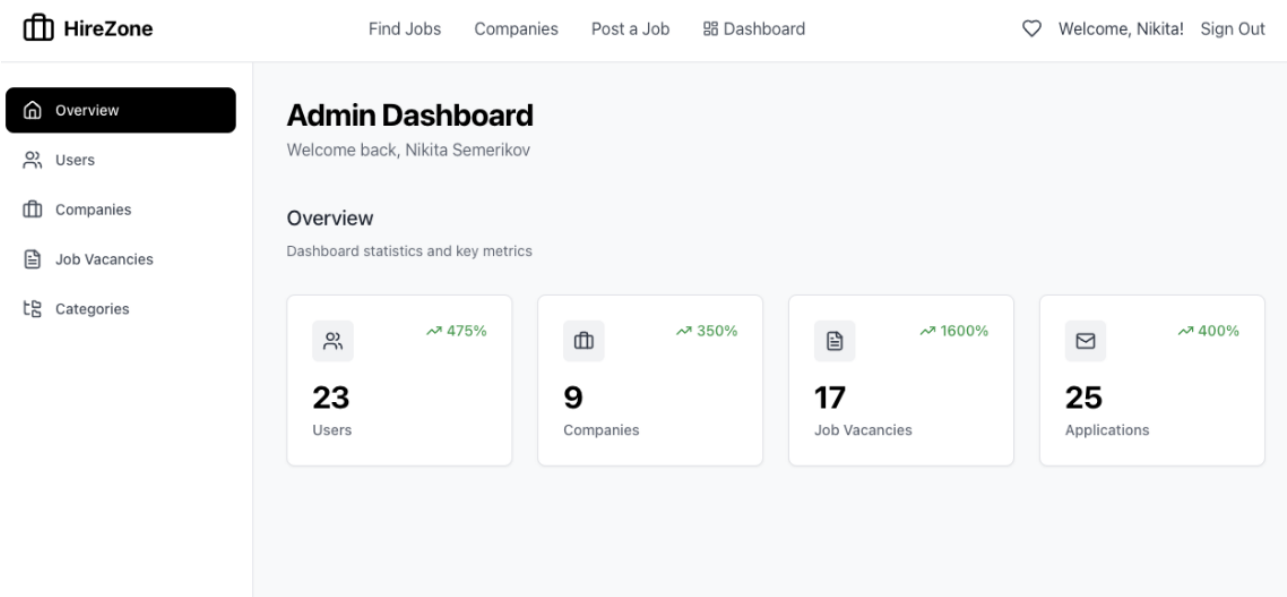


Рисунок 3.23 – Панель адміністратора: статистика та користувачі

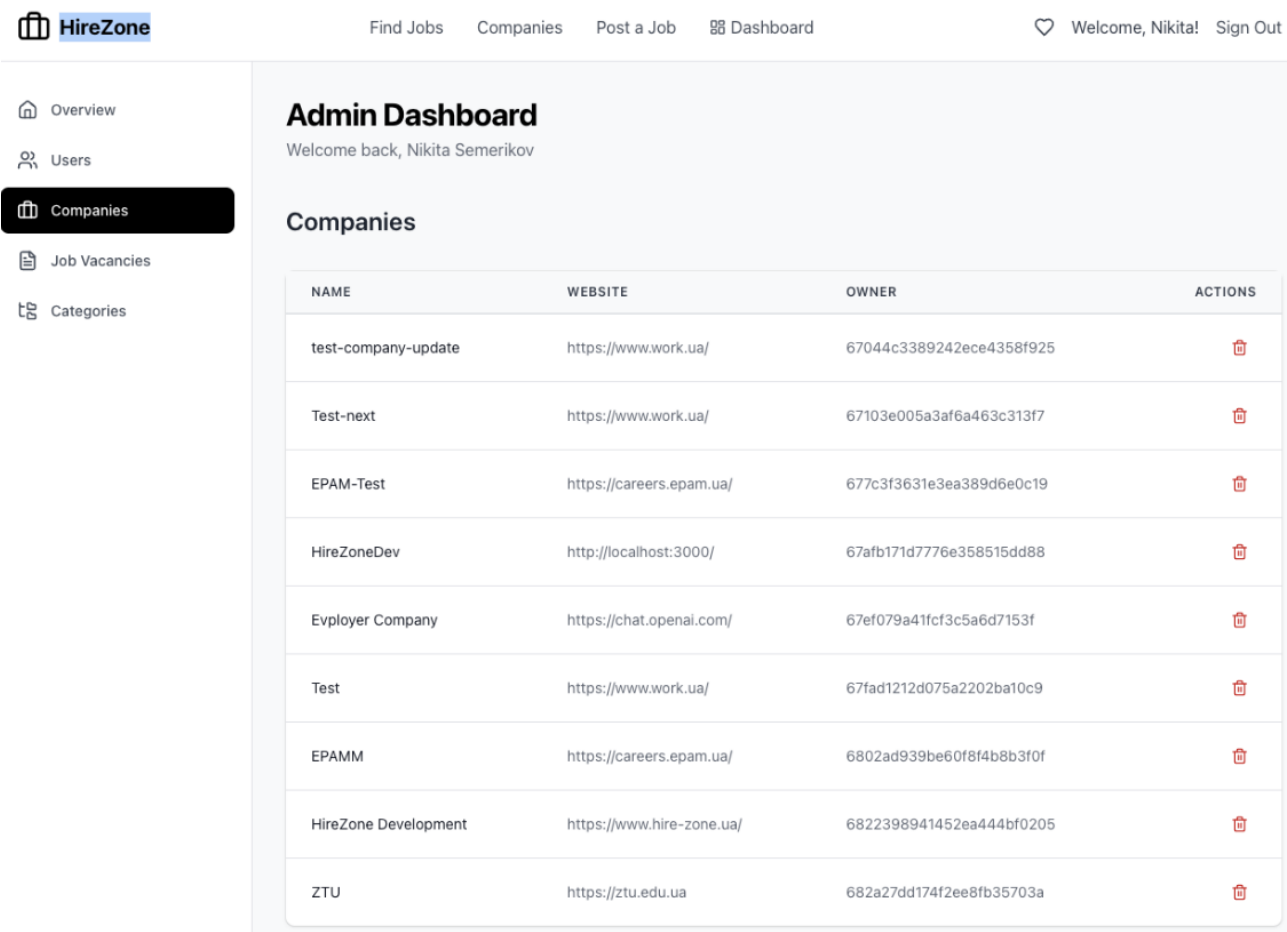


Рисунок 3.24 – Список компаній

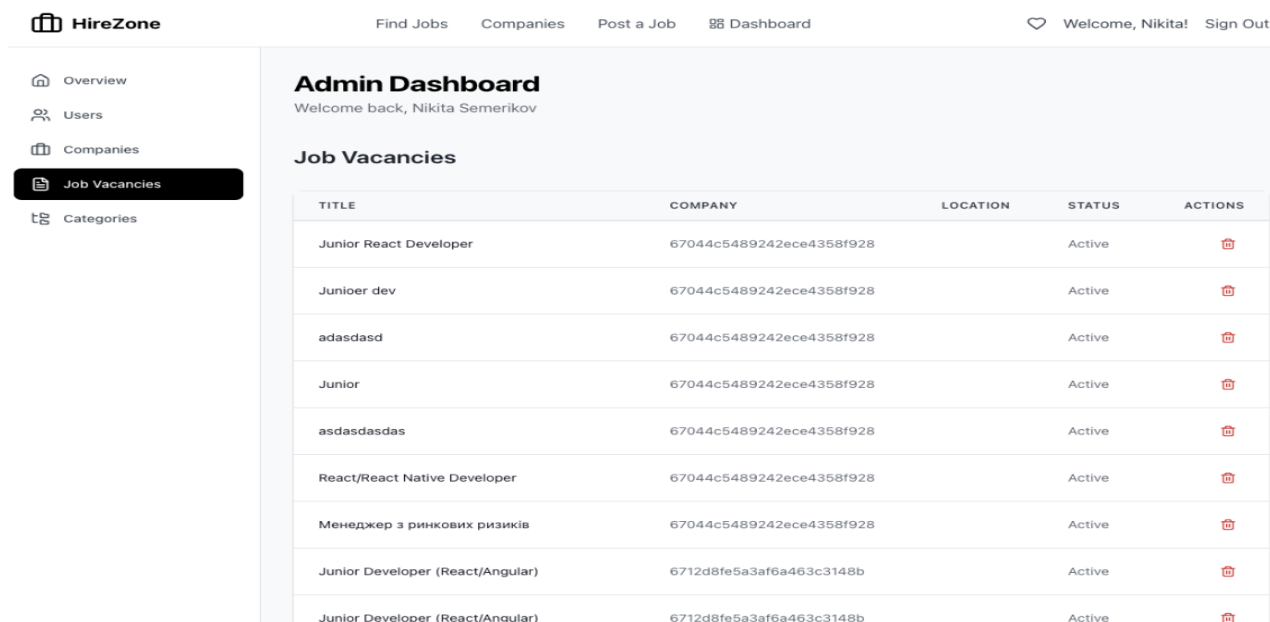


Рисунок 3.25 – Список вакансій

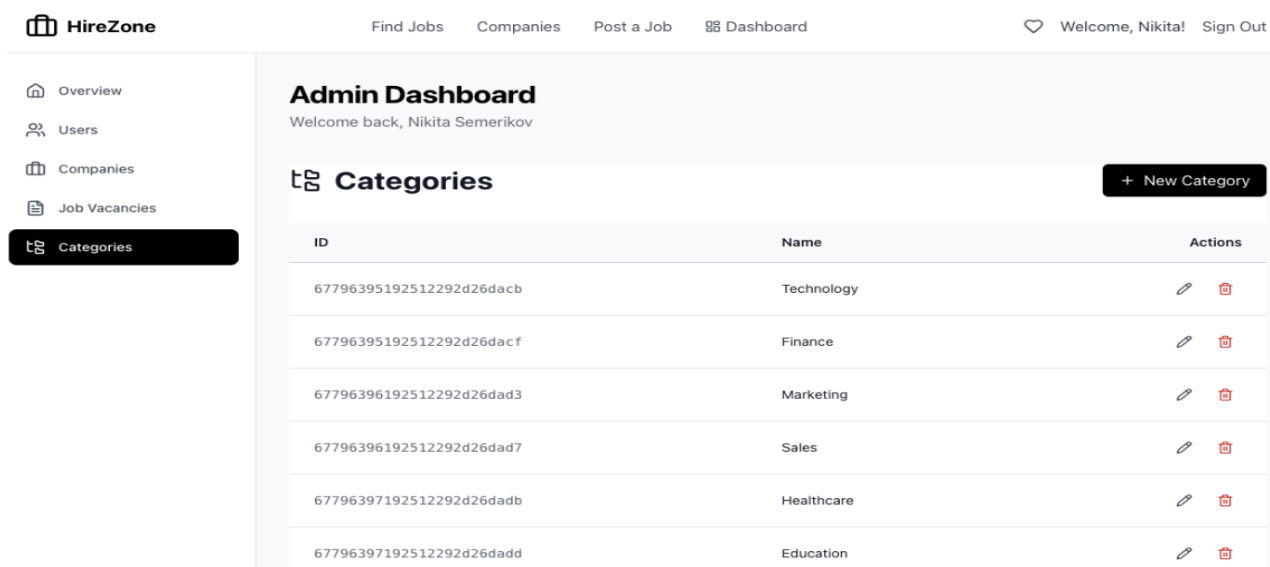


Рисунок 3.26 – Список категорій вакансій

3.3. Тестування роботи програмного продукту.

Після реалізації основної функціональності веб-сервісу для автоматизованого пошуку вакансій було проведено ручне тестування, спрямоване на перевірку її працездатності, зручності використання та відповідності функціональним вимогам. Тестування охоплювало як фронтенд, так і бекенд частини застосунку, включно з інтегрованими ШІ-функціями. Функціональне тестування охоплювало всі ключові сценарії взаємодії користувача з системою, включаючи реєстрацію та авторизацію для кожної з ролей — шукача, роботодавця та адміністратора. Було перевірено коректність

роботи функцій створення та редагування компаній, публікації вакансій, подачі заявок на них, а також перегляд і оновлення даних у особистому кабінеті користувача. Було перевірено доступність усіх форм, валідність введення даних та правильність відображення повідомлень про помилки. Особливу увагу було приділено тестуванню рекомендаційного модуля на базі штучного інтелекту, який надає поради щодо вибору вакансій. Було перевірено, чи враховуються дані з резюме користувача та чи генерується відповідна порада. Також протестовано функціонал ШІ для роботодавців — аналіз резюме кандидатів через інтегрований чат.

У процесі тестування було проведено оцінку зручності навігації, адаптивності інтерфейсу для мобільних пристроїв, зрозумілості форм і логічної структури інтерфейсів. Інтерфейс перевірено в популярних браузерях (Chrome, Firefox, Safari), що підтвердило його стабільну роботу. Адміністративна панель також пройшла тестування, під час якого було перевірено можливість зміни ролей користувачів, видалення компаній, контроль за актуальністю та змістом опублікованих вакансій, а також коректну роботу функцій керування категоріями — створення, редагування й видалення.

Усі тестові сценарії виконувалися вручну відповідно до заздалегідь підготовленого чекліста. У табл. 3.1 наведено приклади основних перевірених функцій, дій користувача, очікуваних результатів і фактичного статусу проходження.

Таблиця 3.1

Функціональне тестування системи

№ тесту	Функціональність	Кроки	Очікуваний результат	Статус
1	Реєстрація користувача	Перейти на сторінку реєстрації, ввести дані, натиснути "Зареєструватися"	Користувач успішно створений, редирект у кабінет	Пройдено

Продовження таблиці 3.1

2	Авторизація користувача	Ввести email/пароль та натиснути "Увійти"	Вхід в особистий кабінет відповідно до ролі	Пройдено
3	Створення компанії роботодавцем	Увійти як роботодавець, перейти до "Створити компанію", заповнити форму	Компанію створено, відображається в дашборді	Пройдено
4	Публікація вакансії	У дашборді компанії натиснути "Створити вакансію", ввести дані	Вакансія з'являється у списку публічних	Пройдено
5	Подання заявки на вакансію	Увійти як пошукач, переглянути вакансію, натиснути "Подати заявку"	Заявка з'являється в кабінеті роботодавця	Пройдено
6	Рекомендації від ШІ	Увійти як пошукач, натиснути на іконку рекомендацій на вакансії	Виводиться порада на основі резюме	Пройдено
7	Аналіз кандидатів через ШІ в чаті	Увійти як роботодавець, перейти до чату, поставити запитання про кандидата	Генерується відповідь з аналізом резюме	Пройдено
8	Зміна ролі користувача адміністратором	Увійти як адміністратор, відкрити список користувачів, змінити роль	Роль змінена, зміни збережені	Пройдено

За результатами тестування програмний продукт виявився стабільним і працездатним. Не було виявлено критичних помилок, які б заважали базовому функціонуванню платформи. Незначні помилки інтерфейсу були усунені під час ітерацій розробки. Таким чином, платформа готова до використання кінцевими користувачами.

Висновки до третього розділу.

У третьому розділі було представлено структуру розгортання платформи та описано логіку взаємодії користувачів різних ролей із системою. Детально розглянуто інтерфейси, функціональні можливості та дії, які можуть виконувати шукачі роботи, роботодавці й адміністратори. Проведене ручне тестування підтвердило стабільність роботи платформи та відповідність її функціональності поставленим вимогам.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

ВИСНОВКИ

Під час виконання випускної кваліфікаційної роботи бакалавра було спроектовано та розроблено веб-сервіс для автоматизованого пошуку вакансій із використанням технологій штучного інтелекту. Проведено дослідження предметної області та ринку праці, а також виконано порівняльний аналіз існуючих онлайн-сервісів, що автоматизують підбір персоналу та взаємодію з кандидатами. На основі отриманих результатів були сформовані вимоги до функціональності системи, що розроблялася.

Проектування системи включало аналіз сценаріїв використання, побудову діаграми розгортання, розробку структури інтерфейсу та логіки для трьох типів користувачів: шукача роботи, роботодавця та адміністратора. Також було спроектовано базу даних для зберігання інформації про користувачів, компанії, вакансії та відгуки. Визначено та реалізовано основний потік взаємодії між користувачами платформи.

Розглянуто функціонал платформи, що охоплює реєстрацію, авторизацію, створення компаній, публікацію вакансій, подачу заявок, управління відгуками, а також використання штучного інтелекту для надання рекомендацій кандидатам та попереднього аналізу резюме для роботодавців. У роботі показано і описано інтерфейс системи, а також продемонстровано приклади основних дій користувача.

Інтерфейс платформи є інтуїтивно зрозумілим і адаптивним до різних типів пристроїв, що робить його зручним для широкого кола користувачів. Система забезпечує ефективну взаємодію між шукачами та роботодавцями й може бути використана як інструмент автоматизації процесів рекрутингу.

Результатом кваліфікаційної роботи є готовий до впровадження веб-сервіс, яка поєднує традиційні механізми пошуку роботи з інноваційними можливостями штучного інтелекту, сприяючи підвищенню ефективності працевлаштування та підбору персоналу.

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. NextJS [Електронний ресурс] – Режим доступу: <https://nextjs.org>
2. NextJS Server Actions [Електронний ресурс] – Режим доступу: <https://nextjs.org/docs/app/building-your-application/data-fetching/server-actions-and-mutations>
3. NextAuth.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://next-auth.js.org/getting-started/introduction>
4. MongoDB [Електронний ресурс] – Режим доступу: <https://www.mongodb.com>
5. Вступ до Mongoose для MongoDB та Node.js [Електронний ресурс].– Режим доступу до ресурсу: <https://code.tutsplus.com/uk/an-introduction-to-mongoose-for-mongodb-and-nodejs--cms-29527a>.
6. Vercel AI SDK [Електронний ресурс] – Режим доступу: <https://vercel.com/docs/ai/vercel-ai-sdk>
7. Vercel Blob [Електронний ресурс] – Режим доступу: <https://vercel.com/docs/storage/vercel-blob>
8. NextJS SSR [Електронний ресурс] – Режим доступу: <https://nextjs.org/docs/pages/building-your-application/rendering/server-side-rendering>
9. NextJS SSG [Електронний ресурс] – Режим доступу: <https://nextjs.org/docs/pages/building-your-application/rendering/static-site-generation>
10. NextJS CSR [Електронний ресурс] – Режим доступу: <https://nextjs.org/docs/pages/building-your-application/rendering/client-side-rendering>
11. TypeScript [Електронний ресурс] – Режим доступу: <https://www.typescriptlang.org>
12. ReactJS [Електронний ресурс] – Режим доступу: <https://react.dev/reference/react>

13. Tailwind CSS [Електронний ресурс] – Режим доступу: <https://tailwindcss.com/docs>
14. Роберт М. Чиста архітектура: Мистецтво створення програмного забезпечення. / Роберт М. [Видання друге] – Харків: Фабула, 2023 – 368 с
15. Learning React: Modern Patterns for Developing React Apps, 2nd edition / Alex Banks, Eve Porcello, 2020 – 307 p.
16. React Explained: Your Step-by-Step Guide to React, 1st edition / Zac Gordon, 2019 – 366 p.
17. Документація JavaScript [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide>
18. Документація HTML [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.mozilla.org/en-US/docs/Web/HTML>
19. Н. J. How HTML, CSS, and JavaScript work [Електронний ресурс] Н. J. Scrimba blog – Режим доступу до ресурсу: <https://scrimba.com/articles/html-css-javascript/>.
20. What is Client Server Architecture? - Differences, Types, Example [Електронний ресурс] Intellipaat Blog – Режим доступу до ресурсу: <https://intellipaat.com/blog/what-is-client-server-architecture/?US>.
21. Web Application Architecture: Principles, Protocols, and Practices / Leon Shklar and Rich Rosen, 2003. – 608 с.
22. MongoDB: The Definitive Guide: Powerful and Scalable Data Storage / Kristina Chodorow., 2013. – 430 с.
23. Вступ до Mongoose для MongoDB та Node.js [Електронний ресурс]. – 2017. – Режим доступу до ресурсу: <https://code.tutsplus.com/uk/an-introduction-to-mongoose-for-mongodb-and-nodejs--cms-29527a>.
24. JavaScript – JavaScript. Work with DOM elements [Електронний ресурс] – Режим доступу: <https://www.educative.io/answers/how-to-remove-a-dom-event-listener-using-javascript>

- 25.CSS Basic concepts of flexbox. How to use flexbox [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/enUS/docs/Web/CSS/CSS_flexible_box_layout/Basic_concepts_of_flexbox
- 26.Стаття на ресурсі Mongo DB про стек MERN та подібні до нього [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mongodb.com/blog/post/the-modern-application-stack-part-1-introducing-the-mean-stack>
- 27.NextAuth.js Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://next-auth.js.org/getting-started/introduction>
- 28.Веб-технології та веб-дизайн / О. Г. Трофименко, О. Б. Козін, О. В. Задерейко, О. Є. Плачінда., 2019. – 284 с.
- 29.Building your own hooks [Електронний ресурс] - Режим доступу до ресурсу: <https://legacy.reactjs.org/docs/hooks-custom.html>
30. Сесіл Мартін Р. Чиста архітектура / Роберт Сесіл Мартін., 2019. – 368 с.
31. Single-page application [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/Single-page_application.

ДОДАТКИ

		Семериків Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		75

Лістинг authOption.ts:

```

import NextAuth, { NextAuthOptions } from "next-auth";
import CredentialsProvider from "next-auth/providers/credentials";
import { dbConnect } from "@config/db";
import UserModel from "@models/User";
import bcrypt from "bcrypt";

export const authOptions: NextAuthOptions = {
  providers: [
    CredentialsProvider({
      name: "Credentials",
      credentials: {
        email: { label: "Email", type: "email" },
        password: { label: "Password", type: "password" },
        role: { label: "Role", type: "text" },
      },
      async authorize(credentials) {
        if (!credentials) {
          return null;
        }

        await dbConnect();

        const user = await UserModel.findOne({ email: credentials.email });

        if (user) {
          const isPasswordValid = await bcrypt.compare(
            credentials.password,
            user.password
          );
          if (isPasswordValid) {
            if (user.role !== credentials.role) {
              throw new Error("Role mismatch");
            }

            return {
              id: user._id.toString(),
              email: user.email,
              firstName: user.firstName,
              lastName: user.lastName,
              role: user.role,
            };
          }
        }
        return null;
      },
    }),
  ],
  session: {
    strategy: "jwt",
  },
  secret: process.env.NEXTAUTH_SECRET,

```

		Семериков Н.О.			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

```

callbacks: {
  async jwt({ token, user }) {
    if (user) {
      token.id = user.id;
      token.firstName = user.firstName;
      token.lastName = user.lastName;
      token.email = user.email;
      token.role = user.role;
    }
    return token;
  },
  async session({ session, token }) {
    if (token) {
      session.user.id = token.id as string;
      session.user.firstName = token.firstName as string;
      session.user.lastName = token.lastName as string;
      session.user.email = token.email as string;
      session.user.role = token.role as string;
    }
    return session;
  },
},
pages: {
  signIn: "/auth/signin",
},
};

```

```

const handler = NextAuth(authOptions);
export { handler as GET, handler as POST };

```

ЛІСТИНГ jobVacancyAction.ts

```

"use server";

import { authOptions } from "@app/api/auth/[...nextauth]/route";
import { dbConnect } from "@config/db";
import CompanyModel from "@models/Company";
import JobVacancyModel, { IJobVacancy } from "@models/JobVacancy";
import {
  IJobVacancyCreate,
  ICreateJobResponse,
  IGetJobVacancyResponse,
  IUpdateJobVacancyResponse,
  IDeleteJobVacancyResponse,
  IGetJobVacancyFullResponse,
} from "@types/job";
import mongoose from "mongoose";
import { getServerSession } from "next-auth";
import { revalidatePath } from "next/cache";

export async function createJobVacancy(
  data: IJobVacancyCreate
): Promise {

```

		Семериків Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

```

try {
  await dbConnect();

  const newJobVacancy = new JobVacancyModel(data);
  const savedJobVacancy = await newJobVacancy.save();
  const plainJobVacancyData = savedJobVacancy.toObject();

  plainJobVacancyData._id = plainJobVacancyData._id.toString();
  plainJobVacancyData.company = plainJobVacancyData.company.toString();
  plainJobVacancyData.postedBy = plainJobVacancyData.postedBy.toString();

  return {
    success: true,
    data: plainJobVacancyData,
  };
} catch (error) {
  return {
    success: false,
    error:
      error instanceof Error ? error.message : "An unknown error occurred",
  };
}

export async function getJobVacancyById(
  vacancyId: string
): Promise<IGetJobVacancyFullResponse> {
  try {
    await dbConnect();

    const jobVacancy = await JobVacancyModel.findById(vacancyId)
      .populate("company")
      .exec();

    if (!jobVacancy || !jobVacancy.company) {
      return {
        success: false,
        error: "Job vacancy or company not found.",
      };
    }

    const plainJobVacancy = {
      ...jobVacancy.toObject(),
      _id: jobVacancy._id.toString(),
      company: {
        ...jobVacancy.company.toObject(),
        _id: jobVacancy.company._id.toString(),
        owner: jobVacancy.company.owner.toString(),
        createdAt: jobVacancy.company.createdAt.toISOString(),
      },
      postedBy: jobVacancy.postedBy.toString(),
      createdAt: jobVacancy.createdAt.toISOString(),
    };
  }
}

```

```

    return {
      success: true,
      data: plainJobVacancy,
    };
  } catch (error) {
    return {
      success: false,
      error:
        error instanceof Error ? error.message : "An unknown error occurred",
    };
  }
}

export async function toggleJobVacancyStatus(vacancyId: string) {
  try {
    await dbConnect();
    const session = await getServerSession(authOptions);

    if (!session || !session.user) {
      return { success: false, error: "User not authenticated" };
    }

    const jobVacancy = await JobVacancyModel.findById(vacancyId);

    if (!jobVacancy) {
      return { success: false, error: "Job vacancy not found." };
    }

    if (jobVacancy.postedBy.toString() !== session.user.id) {
      return {
        success: false,
        error: "You are not authorized to modify this job vacancy.",
      };
    }

    jobVacancy.isActive = !jobVacancy.isActive;
    const updatedJobVacancy = await jobVacancy.save();

    const plainJobVacancy = updatedJobVacancy.toObject();
    plainJobVacancy._id = plainJobVacancy._id.toString();
    plainJobVacancy.company = plainJobVacancy.company.toString();
    plainJobVacancy.postedBy = plainJobVacancy.postedBy.toString();
    plainJobVacancy.createdAt = plainJobVacancy.createdAt.toISOString();

    return { success: true, data: plainJobVacancy };
  } catch (error) {
    return {
      success: false,
      error:
        error instanceof Error ? error.message : "An unknown error occurred",
    };
  }
}

```

		Семериков Н.О			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		79

```

export async function updateJobVacancy(
  vacancyId: string,
  data: Partial<IJobVacancyCreate>
): Promise<IUpdateJobVacancyResponse> {
  try {
    await dbConnect();
    const session = await getServerSession(authOptions);

    if (!session || !session.user) {
      throw new Error("User not authenticated");
    }

    const userId = session.user.id;

    const jobVacancy = await JobVacancyModel.findById(vacancyId)
      .populate("company")
      .lean();

    if (!jobVacancy) {
      return {
        success: false,
        error: "Job vacancy not found.",
      };
    }

    if (
      jobVacancy.postedBy.toString() !== userId &&
      jobVacancy.company.owner.toString() !== userId
    ) {
      return {
        success: false,
        error: "You do not have permission to update this job vacancy.",
      };
    }

    const updatedJobVacancy = await JobVacancyModel.findByIdAndUpdate(
      vacancyId,
      data,
      { new: true }
    );

    if (!updatedJobVacancy) {
      return {
        success: false,
        error: "Failed to update job vacancy.",
      };
    }

    const plainUpdatedVacancy = {
      ...updatedJobVacancy.toObject(),
      _id: updatedJobVacancy._id.toString(),
      company: updatedJobVacancy.company.toString(),
      postedBy: updatedJobVacancy.postedBy.toString(),
      createdAt: updatedJobVacancy.createdAt.toISOString(),
    };
  }
}

```

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

```

    };

    return {
        success: true,
        data: plainUpdatedVacancy,
    };
} catch (error) {
    return {
        success: false,
        error:
            error instanceof Error ? error.message : "An unknown error occurred",
    };
}
}

export async function getJobVacanciesByCompanyId(
    companyId: string
): Promise<IGetJobVacancyResponse> {
    try {
        await dbConnect();

        const jobVacancies = await JobVacancyModel.find({
            company: companyId,
        }).lean<IJobVacancy[]>();

        if (!jobVacancies || jobVacancies.length === 0) {
            return {
                success: false,
                error: "No job vacancies found for this company.",
            };
        }

        const plainJobVacancies = jobVacancies.map((vacancy) => ({
            ...vacancy,
            _id: (vacancy._id as mongoose.Types.ObjectId).toString(),
            company: (
                vacancy.company as unknown as mongoose.Types.ObjectId
            ).toString(),
            postedBy: (
                vacancy.postedBy as unknown as mongoose.Types.ObjectId
            ).toString(),
            createdAt: vacancy.createdAt.toISOString(),
        }));

        return {
            success: true,
            data: plainJobVacancies,
        };
    } catch (error) {
        return {
            success: false,
            error:
                error instanceof Error ? error.message : "An unknown error occurred",
        };
    }
}

```

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		81

```

    }
}

export async function deleteJobVacancy(vacancyId: string) {
  try {
    await dbConnect();
    const session = await getServerSession(authOptions);

    if (!session || !session.user) {
      return { success: false, error: "User not authenticated" };
    }

    const jobVacancy = await JobVacancyModel.findById(vacancyId);

    if (!jobVacancy) {
      return { success: false, error: "Job vacancy not found." };
    }

    if (jobVacancy.postedBy.toString() !== session.user.id) {
      return {
        success: false,
        error: "You are not authorized to delete this job vacancy.",
      };
    }

    await jobVacancy.deleteOne();

    return { success: true, message: "Job vacancy deleted successfully." };
  } catch (error) {
    return {
      success: false,
      error:
        error instanceof Error ? error.message : "An unknown error occurred",
    };
  }
}

export async function getJobVacanciesBySearch(
  title: string | null,
  city: string | null,
  page: number = 1,
  limit: number = 10
): Promise<IGetJobVacancyResponse> {
  try {
    await dbConnect();

    const query: any = {};

    if (city) {
      query.city = city;
    }

    if (title) {
      const companies = await CompanyModel.find({

```

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

    name: { $regex: title, $options: "i" },
  }).lean();

  const companyIds = companies.map((company) => company._id.toString());

  query.$or = [
    { title: { $regex: title, $options: "i" } },
    { description: { $regex: title, $options: "i" } },
    { company: { $in: companyIds } },
    { requirements: { $regex: title, $options: "i" } },
  ];
}

const total = await JobVacancyModel.countDocuments(query);
const jobVacancies = await JobVacancyModel.find(query)
  .populate("company")
  .skip((page - 1) * limit)
  .limit(limit)
  .lean();

if (!jobVacancies || jobVacancies.length === 0) {
  return {
    success: false,
    error: "No job vacancies found.",
  };
}

const plainJobVacancies = jobVacancies.map((vacancy) => ({
  ...vacancy,
  _id: (vacancy._id as mongoose.Types.ObjectId).toString(),
  company: vacancy.company ? vacancy.company.name : null,
  postedBy: (vacancy.postedBy as mongoose.Types.ObjectId).toString(),
  createdAt: vacancy.createdAt.toISOString(),
  title: vacancy.title,
  description: vacancy.description,
  requirements: vacancy.requirements,
  salaryRange: vacancy.salaryRange,
})));

return {
  success: true,
  data: plainJobVacancies,
  pagination: {
    total,
    page,
    limit,
  },
};
} catch (error) {
  return {
    success: false,
    error:
      error instanceof Error ? error.message : "An unknown error occurred",
  };
}

```

		Семериков Н.О			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

    }
  }

export async function getAllVacancies() {
  await dbConnect();
  const vacancies = await JobVacancyModel.find().lean();
  return vacancies.map((v) => ({
    id: v._id.toString(),
    title: v.title,
    company: v.company.toString(),
    location: v.location || "",
    isActive: v.isActive,
  }));
}

export async function deleteJobVacancyAction(formData: FormData) {
  const vacancyId = formData.get("vacancyId") as string;
  await dbConnect();
  await JobVacancyModel.findByIdAndDelete(vacancyId);
  revalidatePath("/admin/jobs");
}

```

Лістинг aiAction.ts

```

import { getJobVacancyById } from "../jobVacancyActions";
import { getUserResumes } from "../userActions";
import { openai } from "@ai-sdk/openai";
import { streamText } from "ai";

export async function analyzeJobEligibility(vacancyId: string) {
  try {
    const vacancy = await getJobVacancyById(vacancyId);
    if (!vacancy.success || !vacancy.data) {
      throw new Error("Vacancy not found");
    }

    const userResume = await getUserResumes();
    if (!userResume.success || !userResume.data.length) {
      throw new Error("No resumes found");
    }

    const resumeFiles = userResume.data.map(
      (resume: { url: string | URL }) => ({
        type: "file",
        data: new URL(resume.url),
        mimeType: "application/pdf",
      })
    );

    const { textStream } = streamText({
      model: openai("gpt-4o"),
      messages: [
        {
          role: "system",

```

		Семериков Н.О			ДУ «Житомирська політехніка». 25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

        content:
            "Ти фахівець з управління персоналом. Подумайте про те, щоб дати
            цьому кандидату шанс на роботу, якщо необхідні зміни незначні. Якщо ні, надайте
            чіткі поради щодо того, чого потрібно навчитися або вдосконалити, щоб
            відповідати вимогам. дай відповідь на укр",
        },
        {
            role: "user",
            content: `This is the vacancy title: "${
                vacancy.data.title
            }", description: "${
                vacancy.data.description
            }", requirements: "${vacancy.data.requirements.join(", ")}".`,
        },
        {
            role: "user",
            content: [
                { type: "text", text: "These are my resumes:" },
                ...resumeFiles,
            ],
        },
    ],
});

return { success: true, data: textStream };
} catch (error) {
    console.error("Error in analyzeJobEligibility:", error);
    return {
        success: false,
        error: error instanceof Error ? error.message : "Unknown error occurred",
    };
}
}

```

Лістинг chat/route.ts

```

import { getApplicationsByJobVacancyId } from "@actions/applicationActions";
import { getChatById } from "@actions/chatActions";
import { getJobVacancyById } from "@actions/jobVacancyActions";
import { saveChat } from "@tools/chat-store";
import { google } from "@ai-sdk/google";
import { appendResponseMessages, createIdGenerator, streamText } from "ai";

export async function POST(req: Request) {
    try {
        let { messages, id, jobVacancyId } = await req.json();

        if (!jobVacancyId) {
            const chat = await getChatById(id);

            if (!chat) {
                throw new Error("Chat not found");
            }
            jobVacancyId = chat.jobVacancy;
        }
    }
}

```

		Семериков Н.О.			ДУ «Житомирська політехніка».25.121.21.000 - ПЗ	Арк.
		Громський О.О.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

const [vacancy, jobVacancyApplications] = await Promise.all([
  getJobVacancyById(jobVacancyId),
  getApplicationsByJobVacancyId(jobVacancyId),
]);

if (!vacancy.success || !vacancy.data) {
  throw new Error("Vacancy not found");
}
if (
  !jobVacancyApplications.success ||
  !jobVacancyApplications.data?.length
) {
  throw new Error("No resumes found");
}

const resumeFiles = jobVacancyApplications.data
  .filter((application) => application.resumeUrl)
  .map((application) => ({
    type: "file",
    data: new URL(application.resumeUrl),
    mimeType: "application/pdf",
  }));

const requestMessages = [
  {
    role: "system",
    content:
      "Ти фахівець з управління персоналом. Проаналізуй надані резюме для  

      цієї вакансії і вкажи, який кандидат найкраще підходить.",
  },
  {
    role: "user",
    content: `Назва вакансії: "${vacancy.data.title}".
      Опис: "${vacancy.data.description}".
      Вимоги: "${vacancy.data.requirements.join(", ")}".`,
  },
  {
    role: "user",
    content: [
      { type: "text", text: "These are the submitted resumes:" },
      ...resumeFiles,
    ],
  },
  ...messages,
];

const result = streamText({
  model: google("models/gemini-2.0-flash-exp"),
  messages: requestMessages,
  experimental_generateMessageId: createIdGenerator({
    prefix: "msgs",
    size: 16,
  }),
});

```

```

    async onFinish({ response }) {
      await saveChat({
        id,
        messages: appendResponseMessages({
          messages: messages,
          responseMessages: response.messages,
        }),
      });
    },
  });

  return result.toDataStreamResponse();
} catch (error: any) {
  return new Response(error.message, { status: 400 });
}
}

```