МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КІБЕРБЕЗПЕКИ, КОМП'ЮТЕРНОЇ ТА ПРОГРАМНОЇ ІНЖЕНЕРІЇ

Кафедра Комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
Аліна САВЧЕНКО
«_____» _____ 2022р.

КВАЛІФІКАЦІЙНА РОБОТА
(ДИПЛОМНА РОБОТА, ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ **"МАГІСТР"**
ЗА ОСВІТНЬО-ПРОФЕСІЙНОЮ ПРОГРАМОЮ
ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ

Тема: "Метод аналітичної обробки текстових матеріалів пошукової платформи"

Виконавець: студент групи УС-212М Коровін Дмитро Олегович

Керівник: професор Зіатдінов Юрій Кашафович

Нормоконтролер: Ігор РАЙЧЕВ

Київ – 2022

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *Кібербезпеки, комп'ютерної та програмної інженерії*

Кафедра *Комп'ютерних інформаційних* технологій

Галузь знань, спеціальність, освітньо-професійна програма: 12 "Інформаційні Технології", 122 "Комп'ютерні науки", "Інформаційні управляючі системи та технології"

ЗАТВЕРДЖУЮ
Завідувач випускової кафедри
_____Аліна САВЧЕНКО
«____»_____2022р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи студента

Коровіна Дмитра Олеговича
(прізвище, ім'я, по батькові)

1. Тема роботи: «Метод аналітичної обробки текстових матеріалів пошукової платформи» затверджена наказом ректора від 28 вересня 2022 р. за № 1774/ст.

2. Термін виконання роботи: 26.09.2022 – 21.11.2022

3. Вихідні данні до роботи: огляд категорій методів аналізу тексту ;аналіз методів відповідно до поставленої задачі; розробка веб платформи пошуку; інтеграція методів.

4. Зміст пояснювальної записки: вступ, методи пошуку та аналізу текстових даних, Засоби програмної розробки, проблематика пошукової платформи, метод аналітичної обробки текстових даних.

5. Перелік обов'язкового графічного матеріалу: структурна діаграма веб платформи, функціональна схема веб платформи, діаграма процесу авторизації, діаграма взаємодії програмних засобів розробки

6. Календарний план-графік

№ п/п	Завдання	Термін виконання	Підпис керівника
1.	Отримання завдання на дипломну роботу, створення плану дипломної роботи та побудова плану-графіку виконання робіт.	26.09.2022 – 28.09.2022	
2.	Огляд та аналіз наукової літератури по темі дипломної роботи та написання Розділу 1.	29.09.2022 – 09.10.2022	
3.	Написання Розділу 2 дипломної роботи.	10.10.2022 –20.10.2022	
4.	Написання Розділу 3 і Розділу 4 дипломної роботи. Завершення створення пояснювальної записки дипломної роботи.	21.10.2022 – 31.10.2022	
5.	Оформлення та друк пояснювальної записки.	01.11.2022 – 07.11.2022	
6.	Створення презентації, доповіді та підготовка до захисту дипломної роботи.	08.11.2022 – 15.11.2022	
7.	Підготовка матеріалів дипломної роботи для передачі секретарю ДЕК (папка, конверт, диск із файлом диплому, рецензія, відгук).	16.11.2022 – 18.11.2022	

7. Дата видачі завдання: «26» вересня 2022 р.

Керівник дипломної роботи _____ Юрій ЗІАТДІНОВ
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Дмитро КОРОВІН
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломної роботи «Метод аналітичної обробки текстових матеріалів пошукової платформи» складається зі вступу, чотирьох розділів, висновку, списку бібліографічних посилань і містить 88 сторінок, 33 рисунки. Список бібліографічних посилань складається з 13 найменувань.

Ключові слова: ІНТЕГРАЦІЯ МЕТОДІВ, WEB-ПЛАТФОРМА, WEB-СЕРВІС, РОЗРОБКА ІС, МОДЕЛЬ ДОДАТКУ СЕРВЕР-КЛІЖНТ, ТЕСТОВІ НАБОРИ ДАНИХ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ПРОГРАМНА СИСТЕМА.

Актуальність. Нині етап аналізу текстових даних та документів є обов'язковою частиною процесу взаємодії користувачів з великими системами колекцій даних, воно спрямоване на виявлення та усунення якомога більшої кількості помилок та поліпшення використання складних систем. Наслідком такої діяльності є підвищення якості властивостей ПЗ. Однак, незалежно від кількості методів порівняльного аналізу текстових даних, немає одного єдиного підходу, за допомогою якого можна було б проводити будь-який аналіз з постійною точністю, адже кожен метод націлений на виконання своєї специфічної задачі.

Метою дипломної роботи є дослідження методів тексту, задач які виконують аналітичні методи порівняння, способів створення і застосування власного методу. Як результат, на основі отриманих знань необхідно розробити проект пошукової системи для веб-платформи. Додатковою метою є підключення додаткових сервісів платформи.

Для досягнення поставленої мети необхідно вирішити такі **завдання**:

- виявити місце, яке займає аналіз текстових даних в процесі розробки ПС;
- детально ознайомитися з процесом інтеграції аналітичних методів;
- розглянути найвідоміші підходи аналізу тексту;
- ознайомитися з техніками створення інтерфейсів веб-платформ;
- виконати огляд ПЗ для розробки веб-платформи;
- обрати найкращий метод аналізу;
- оволодіти навиком налаштування взаємодії серверу та клієнта;

Об'єктом дослідження є процес аналізу текстових даних в ПЗ.

Предметом дослідження є методи аналізу текстових даних і можливість легкої інтеграції до різноманітних пошукових платформ.

Методи дослідження включають у себе:

- методи аналізу тексту;
- методи визначення якості продукту;
- методи верифікації програмного забезпечення;
- методи інтеграції.

Теоретичною основою дипломної роботи стали вітчизняні та зарубіжні дослідження щодо забезпечення якості програмного забезпечення та публікації на сайтах, присвячені питанням аналізу текстових даних програмних систем.

Теоретична і практична значимість роботи полягає в тому, що на основі отриманих знань:

1) можна в короткий термін ознайомитися з необхідною теорією методів аналізу текстових даних ПЗ, що може допомогти покращенню розуміння взаємодії з пошуковими системами;
2) по зібраному матеріалу можна сформувати методичний посібник для студентів і включити його в програму навчання в якості додаткового курсу до дисципліни "Методи та засоби обробки інформації в СК";
3) розроблено проект веб-платформи для публікації оголошень розшуку з системою пошуку .

На захист виносяться наступні положення:

1) процес аналізу текстових даних;
2) розробка веб платформ;
3) реалізація функціоналу системи пошуку.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

ПЗ	Програмне забезпечення
I/O	Введення/вихід – взаємодія між оброблювачем інформації (наприклад, комп'ютер) і зовнішнім світом, який може представляти як людина, так і будь-яка інша система обробки інформації. Введення – сигнал або дані, отримані системою, а вихід – сигнал або дані, надіслані нею (або з неї)
ООП	Об'єктно-орієнтоване програмування
FTP	File Transfer Protocol – стандартний протокол, призначений для передачі файлів по TCP/IP-мережі
СУБД	Система управління базами даних
GUI	Різновид призначеного для користувача інтерфейсу, в якому елементи інтерфейсу (меню, кнопки, значки, списки тощо), виконані у вигляді графічних зображень
ІС	Інформаційна система
ПС	Програмна система
ІТ	Інформаційні технології
Токенізація	процес захисту конфіденційних даних шляхом заміни алгоритмічно сформованого числа

ВСТУП

Швидкий розвиток інформаційних технологій, особливо розвиток Інтернету, привів людей в епоху обміну інформацією. Інтернет надає людям платформу для обміну інформацією та став невід'ємною частиною сучасних життєвих інструментів та інструментів роботи. Доступ до мобільного Інтернету став одним із найбільш часто використовуваних інтернет-каналів. З постійним збільшенням кількості користувачів Інтернету та безперервним зростанням онлайн-інформації люди зіткнулися з проблемою масової інформації, такої як пошук та керування, викликану розширенням кількості даних. Методи ефективної організації та керування цією інформацією стали сферами інформаційної науки. З безперервним розвитком технологій класифікація медіа файлів поступово змінилася від методу, заснованого на знаннях, до методу, заснованого на статистиці та машинному навчанні. Заходи подібності тексту відіграють все більш важливу роль у дослідженнях, пов'язаних з текстом, і додатках, таких як пошук інформації, класифікація тексту, кластеризація документів, виявлення теми, відстеження теми, створення питань, відповіді на питання, оцінка есе, оцінка коротких відповідей, машинний переклад, текст. підбиття підсумків та інші. Виявлення подібності між словами є фундаментальною частиною подібності тексту, яка потім використовується як основний етап для подібності речень, абзаців та документів.

Вимірювання подібності між словами, пропозиціями, абзацами та документами є важливим компонентом у різних завданнях, таких як пошук інформації, кластеризація документів, усунення неоднозначності слів, автоматична оцінка есе, оцінка коротких відповідей, машинний переклад та підсумовування тексту. У цьому огляді обговорюються існуючі роботи зі схожості текстів шляхом поділу їх на три підходи; Подібності на основі рядків, на основі корпусу та на основі знань. Крім того, представлені зразки поєднання цих подібностей.

Слова можуть бути схожі двояко лексично та семантично. Слова схожі лексично, якщо вони мають однакову послідовність символів. Слова подібні семантично, якщо вони мають одне й те саме, протилежні один одному, вживаються однаково,

вживаються в одному контексті і одне є типом іншого. Лексична схожість представлена в цьому огляді за допомогою різних алгоритмів на основі рядків, семантична схожість представлена за допомогою алгоритмів на основі корпусу та знань. Заходи на основі рядків працюють із послідовностями рядків та композицією символів. Метрика рядка - це метрика, яка вимірює подібність або відмінність між двома текстовими рядками для приблизного зіставлення або поріняння рядків. Подібність на основі корпусу - це міра семантичної подібності, яка визначає подібність між словами відповідно до інформації, отриманої з великих корпусів. Подібність на основі знань - це міра семантичної подібності, що визначає ступінь подібності між словами з використанням інформації, отриманої з семантичних мереж. Коротко будуть представлені найпопулярніші для кожного виду.

Пошук інформації, також відомий як запити, відноситься до чітко визначеного, цілеспрямованого пошуку інформації для чітко сформульованої інформаційної потреби, тобто коли ви маєте досить чітке уявлення про тип інформації, яка вам потрібна. Сценарій 1 вище з конкретними елементами належить до цієї категорії. Проте пошук інформації може також охоплювати пошук, коли ви ще не маєте конкретних елементів, які потрібно знайти, але чітко визначили свої інформаційні потреби та відносно впевнені щодо типу інформації, яка вам потрібна:

- пошук відомого елемента: коли у вас є достатньо деталей про елемент, щоб можна було його ідентифікувати та знайти, наприклад, ім’я автора, назва, ISBN, назва журналу, том і номер випуску;
- фактичний пошук: коли вам потрібна інформація про конкретні факти, наприклад, чисельність населення Ісландії або рік, коли було побудовано Емпайр-Стейт-Білдінг;
- тематичний пошук: це передбачає пошук інформації на тему, яку ви не можете повністю визначити. Це найскладніший тип пошуку, оскільки ви не можете точно вказати, що вам потрібно, а від чого можете сміливо відмовитися.. Більшість завдань, які ви отримуєте в коледжі, вимагатимуть пошуку предметів; тому ми витратимо деякий час на це обговорення.

На відміну від пошуку, перегляд - це нецільовий пошук, коли ваша потреба в

інформації є невизначеною або дуже загальною, або ви не знайомі з темою, яку досліджуєте. Перегляд дає змогу відчути тематику, яка, в ідеалі, через деякий час перетвориться на більш цілеспрямовану та точну форму пошуку. Метою перегляду є відкриття. Ви швидко переглядаєте та гортаєте інформацію в надії знайти інформацію, яка допоможе вам у виконанні вашого завдання. По суті, ви шукаєте інформаційні ресурси, про існування яких ви ще не підозрюєте. Для порівняння пошук інформації передбачає пошук ресурсів, про які ви точно знаєте або принаймні сильно підозрюєте, що вони десь є. Перегляд веб-сторінок уможливлює інтуїцію, «випадковість і розвиток подій щасливим або корисним способом». Ми переглядаємо в надії отримати натхнення або несподівано натрапити на ідеальне джерело. Перехід за посиланнями в Інтернеті, сканування полиць у бібліотеці, читання змісту книг, прокручування меню на веб-сайті - усе це приклади поведінки під час перегляду. Іноді нам потрібно починати широкий і звужувати пошук, поки ми не знайдемо ідеальне джерело. Більшість електронних пошукових систем тепер підтримують як пошук інформації, так і поведінку перегляду. Подумайте про такий веб-сайт, як Amazon: ви можете, наприклад, вибрати або шукати елемент безпосередньо, ввівши ключові слова чи імена авторів у вікно пошуку, або, якщо ви не впевнені, що хочете знайти, переглянути різні відділів або жанрів. такі веб-сайти, як Amazon, визнають важливість перегляду веб-сторінок і випадковості, надаючи списки товарів, схожих на той, який ви щойно клацнули під заголовком «Клієнти, які купили цей товар, також купили». Те саме стосується систем, які ви будете використовувати для роботи в коледжі. Залежно від того, чи є ваша потреба в інформації чітко визначеною чи нечіткою та загальною, як цифрові детективи ви повинні знати, як вибрати відповідні ресурси для пошуку та як розробити ефективні стратегії пошуку, які дозволять вам виконувати високоякісні завдання.

Найбільш поширеними типами медіа є текстові документи та графічні зображення. Кількість статей, що публікуються кожен день неймовірна та досягає сотні тисяч як унікальних так і комбінованих текстів. У кожному документі описано та чи інша проблема специфіки якогось ремесла і ці знання можуть бути корисні іншим людям, які ведуть свою діяльність у цій сфері. Технологія автоматичної категоризації тексту полягає у вивченні того, як дозволити машині класифікувати невідомий текст

шляхом самонавчання, вирішуючи тим самим труднощі, що виникають при ручній класифікації. Оскільки гранульовані обчислення можуть зменшити знання при вирішенні складних задач, зручніше узагальнювати та здобувати знання. Останніми роками він став гарячою точкою, а також надає нові ідеї для дослідження класифікації текстів. Приблизна модель детальних обчислень може отримати знання за допомогою правил прийняття рішень. Процес прийняття рішення є більш прозорим і легким для розуміння. На це приділено увагу та застосовано в дослідженні класифікації текстів. Ідентифікація інформації, а також її пошук значною мірою залежить від відповідного підходу до пошуку та пошуку інформації. Пошук інформації є явищем ширшим, ніж пошук інформації, і являє собою пошук інформації. З іншого боку, пошук інформації включає розробку пошукових термінів і подальше введення в пошукову систему — процес, який генерує інформацію про розташування інформаційних ресурсів. Потім ці ресурси можна отримати та оцінити на відповідність інформаційній потребі. Це вимагає знання пошукових термінів, а також стратегій, які використовуються для розробки таких термінів: усе це важливо для використання спеціалізованої служби бази даних.

Подібність — це відстань між двома векторами, де розміри векторів є характеристиками двох об'єктів. Простіше кажучи, подібність — це міра того, наскільки різні чи схожі два об'єкти даних. Якщо відстань замало, кажуть, що об'єкти мають високий ступінь подібності, і навпаки. Як правило, він вимірюється в діапазоні від 0 до 1. Цей показник у діапазоні [0, 1] називається показником подібності. Важливий момент, який слід пам'ятати про подібність, полягає в тому, що вона суб'єктивна і залежить від предметної області і варіанта використання. Наприклад, два автомобілі можуть бути схожі через такі прості речі, як компанія-виробник, колір, ціновий діапазон або технічні деталі, такі як тип палива, колісна база, потужність. Таким чином, слід виявляти особливу обережність при розрахунку подібності між ознаками, які не пов'язані один з одним або не стосуються проблеми. Якою б простою була ідея, подібність лежить в основі багатьох методів машинного навчання. Наприклад, класифікатор найближчих сусідів використовує подібність для класифікації нових об'єктів даних, аналогічним чином кластеризація середніх використовує подібні заходи для призначення точок даних відповідним кластерам. Навіть механізми

рекомендацій використовують методи спільної фільтрації на основі сусідства, які використовують подібність для ідентифікації сусідів користувача. Використання подібних заходів дуже помітно в області обробки природної мови. Все, від систем пошуку інформації, пошукових систем, виявлення перефразування до класифікації текстів, автоматичного зв'язування документів та виправлення орфографії, використовує подібні заходи.

РОЗДІЛ 1. МЕТОДИ ПОШУКУ ТА АНАЛІЗУ ТЕКСТОВИХ ДАНИХ

1.1. Огляд проблеми порівняльного аналізу текстових даних

Недавньою інформаційною проблематикою є швидке зростання даних. Вимірювання схожості тексту - це метод дослідження тексту, який може вирі-шити цю серйозну проблему. Виявлення подібності між словами є первинним етапом подібності речень, абзаців та документів. Підхід схожості тексту може полегшити пошук релевантної інформації. Це основа успішних операцій аналізу тексту, таких як пошук та вилучення інформації, класифікація тексту, вилучення інформації, кластеризація документів, аналіз тональності, машинний переклад, підсумовування тексту та обробка природної мови (NLP).

За допомогою аналізу тексту ви можете швидше отримати точну інформацію з джерел. Процес повністю автоматизований і послідовний, і він відображає дані, на основі яких можна діяти. Наприклад, використання програмного забезпечення для аналізу тексту дозволяє негайно виявляти негативні настрої в публікаціях у соціальних мережах, щоб ви могли працювати над вирішенням проблеми

Компонент Compare порівнює дані з одного набору вихідних таблиць з даними з іншого. Compare простий у використанні, простий за концепцією, але потужний у підтримці складних структур бази даних. Програмісти та адміністратори баз даних можуть легко перевіряти та порівнювати набори пов’язаних даних. Порівняння усуває трудомісткі зусилля вручну «збирати» дані з різних таблиць і систем керування базами даних. Інтуїтивно зрозумілі діалоги спрощують завдання введення даних і надають параметри для порівняння наборів реляційно незайманих даних.

- аналізувати дані, які використовуються для тестування програми, порівнюючи результати до та після виконання програми;
- порівняння архівні дані з поточними даними;
- визначення подібності та відмінності в окремих базах даних;
- відстежування зміни бази даних.

1.2. Категорії алгоритмів порівняння

Було запропоновано різні підходи для вимірювання подібності одного тексту до іншого. Метод поділено на чотири основні групи: подібності: на основі рядків чи строк, на основі наборів документів, на основі знань та гібридного тексту.

Подібність на основі рядків найстаріший, найпростіший, але найпопулярніший підхід до вимірювання. Цей алгоритм працює з послідовностями рядків та композицією символів. Два основних типи функцій подібності рядків – це функції подібності на основі *символів* та функції подібності на основі *токенів*. Подібність на основі символів також називається виміром на основі послідовності або відстанню редагування. На вході маємо два рядки символів, а потім обчислюємо відстань редагування (включаючи вставку, видалення та заміну) між ними.

На основі символів кількісно визначається схожість символів між двома рядками для кількісної оцінки подібності, наприклад, відстань редагування, яка є мінімальною кількістю операцій редагування одного символу, необхідних для перетворення одного рядка в інший [1]. Іншими словами, два рядки подібні, якщо номер операції мінімальної відстані редагування менший за заданий поріг. Деякими прикладами цього підходу є *відстань Хеммінга*, *відстань Левенштейна. Дамеро-Левенштейна,. найдовша загальна підпослідовність, метод Сміт-Ватермана, Джаро, Джаро-Вінклера та N-грам*. Символьна міра корисна для розпізнавання друкарських помилок, але марна при розпізнаванні переставлених термінів (наприклад, витік інформації – інформації витік). Відстань редагування широко використовується для апроксимації зіставлення рядків, щоб упоратися з наявною невідповідністю даних. Подібність на

основі термінів також відома як заснована на токенах, оскільки вона моделює кожен рядок як набір токенів. Подібність між рядками можна оцінити, маніпулюючи наборами токенів, наприклад словами. Основна ідея цього підходу полягає в тому, щоб виміряти подібність двох рядків на основі загальних токенів, що відповідають його наборам токенів. Якщо зазначено подібність, пара рядків позначається як схожа або повторювана. Подібності на основі термінів є більш ефективними у порівнянні з символьним, коли необхідно опрацювати величезну кількість рядків.

Насправді, символьні пріоритети стають надто дорогими в розрахунковому відношенні та менш точними для широких рядків, таких як текстові документи. Основною характеристикою подібності на основі токенів є використання перекриття двох наборів токенів як кількісну оцінку подібності. Перекриття розраховується на основі пар токенів, що точно збігаються, без урахування інших схожих токенів. Підхід подібності на основі токенів корисний для великого терміна "перестановка" шляху розбиття рядків на підрядки. *Подібність Жаккара, коефіцієнт Дайса, подібність косинуса, манхеттенська відстань та евклідова відстань* є деякими прикладами цих методів.

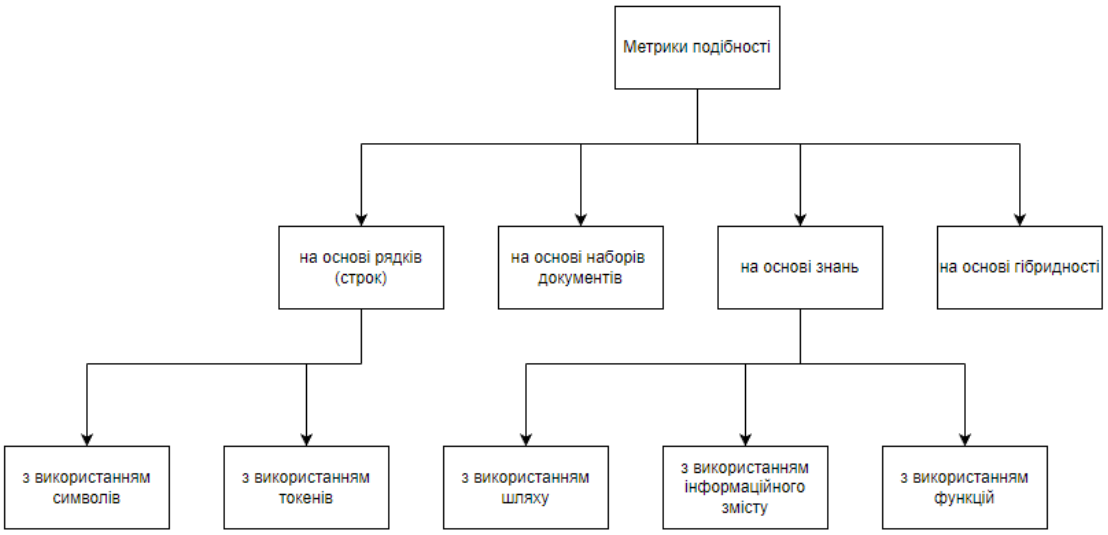

Рис. 1.1. Категорії алгоритмів

Подібність на основі документа використовує семантичний підхід. Цей підхід до подібності визначає схожість між двома поняттями на основі інформації, витягнутої

з великих документів. Документ чи множина документів є великою колекцією електронних письмових чи усних текстів. Документ містить певний набір речень та їх переклад іншою мовою. Мета полягає в тому, щоб зіставити вхідний текст з текстом у документі та домогтися перекладу [2]. Багато заходів подібності чи спорідненості з урахуванням корпусів засновані на концептуальних ресурсах, як-от *Вікіпедія*.

Подібність на основі знань - заходи семантичної подібності, які використовують інформацію з семантичних мереж визначення ступеня подібності слів [3]. Подібність, заснована на знаннях, складається з семантичної подібності та семантичної спорідненості. Подібність визначає два взаємозамінні поняття, тоді як пов'язаність семантично пов'язує поняття [4]. Семантичний підхід використовує явне уявлення знань, таких як взаємозв'язок фактів, значень слів та правил для опису висновків щодо конкретних областей. Схема подання знань зазвичай включає правила висновків, логічні твердження та мережеву семантику, таку як таксономія та онтологія. Деякі доступні онтології: WordNet, SENSUS1, Cyc2, UMLS3, SNOMED4, MeSH, GO5 та STDS6 [5]. WordNet є найцікавішим онтологічним ресурсом і широко використовується для вимірювання подібності на основі знань. WordNet - це велика англомовна лексична база даних дослідницького проекту, розробленого Прінстонським університетом. WordNet реалізує іменники, дієслова, прислівники і використовує одноразові семантичні відносини, звані наборами синонімів (синсетами), які застосовуються одноразово. Як понятійно-семантичні, і лексичні зв'язки пов'язують системні мережі. Слова WordNet структуровані ієрархічно з використанням гіпонімії та гіперонімії, і слова можна легко підібрати як поняття. Таким чином WordNet можна інтерпретувати як таксономію. Підхід подібності, що базується на знаннях, який використовує онтологію WordNet, можна розділити на чотири типи: заснований на шляху, що базується на інформаційному змісті, заснований на інших типах [6].

- **заснований на шляху:** основне поняття (також відоме як міри підрахунку ребер) це довжина шляху та його положення в таксономії, представлене функцією подібності між двома поняттями. Цей захід використовує найкоротший шлях між поняттями, як, наприклад, у новаторській роботі Rada et al., а деякі заходи, що стосуються цього підходу, описані у Ластра-Діаса та Гарсіа-Серрано [7].

- **заснований на інформаційному змісті (ІЗ):** включає певні концепції в розрахунку подібності. Основна ідея подібних заходів на основі ІЗ застосовується в моделі інформаційного контексту. Розрахунок залежить від кожного концепту та нащадка частот у текстовому документі. Фундаментальна гіпотеза повинна належати до абстрактнішого поняття з меншою інформативністю, ніж до конкретного змісту. Підхід, заснований на ІЗ, розглядається як дуже перспективний і стає одним із основних напрямків досліджень у цій галузі [6].
- **заснований на функціях:** полягає у використанні теоретико-множинної операції між наборами ознак концептів. Міра на основі ознак описує набір передбачуваних термінів як властивостей або ознак. Кількість загальних характеристик вище, ніж менш загальних характеристик двох термінів означає, що ці елементи схожі [8].

На основі гібридності: на додаток до трьох категорій, описаних раніше, є ще кілька заходів подібності, які не можна віднести до будь-якого попереднього сімейства. Ідея цього підходу полягає в тому, щоб об'єднати раніше описані підходи, включаючи подібність на основі рядків, на основі корпусу та на основі знань, щоб досягти кращої метрики за рахунок використання їх переваг.

1.3. Підхід “торбина слів” або індекс подібності Жаккара

Найпростішим способом порівняння двох документів було б просто взяти слова, які присутні в обох, і обчислити ступінь перекриття. Ми можемо відкинути інформацію про те, яке слово зустрічається з яким іншим словом у реченні. Оскільки таке уявлення про речення схоже на те, щоб покласти набір слів у пакет і змішати їх перед порівнянням, цей прийом називається моделлю «торбина слів». Наприклад є два речення:

1. Перевагу серед студентів бере онлайн навчання.
2. Навчання онлайн є сучасною тенденцією.

Тоді один дуже простий спосіб виміряти подібність — видалити стоп-слова (і, та, або т.д.), а потім обчислити кількість слів в обох документах, поділену на кількість слів у будь-якому документі.

Це число називається індексом Жаккара і було розроблено понад століття тому швейцарським ботаніком Полем Жаккаром.

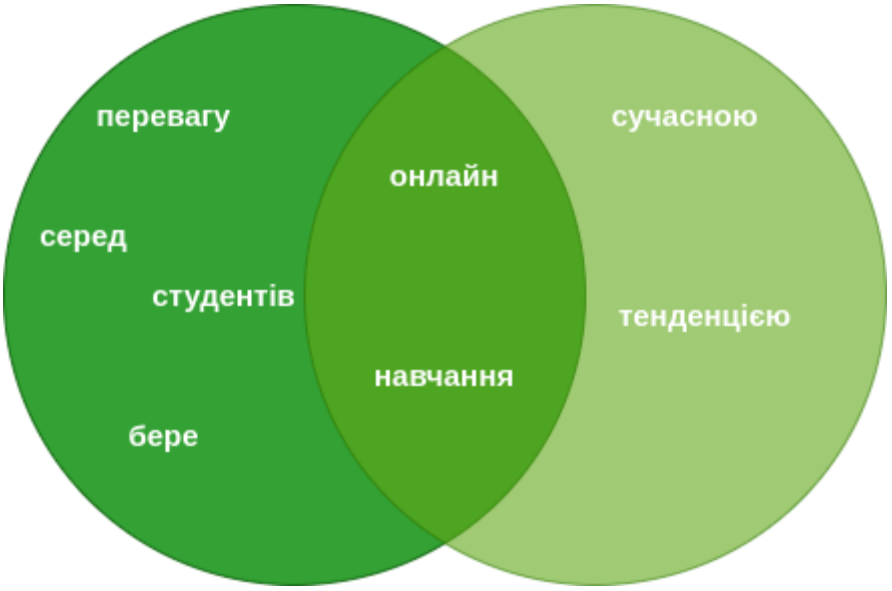

Рис. 1.2. Перетин текстових даних документів

Не стоп-слова у першому реченні:
{ “Перевагу”, “серед”, “студентів”, “бере”, “онлайн”, “навчання” }

Не стоп-слова другого речення:
{ “Навчання”, “онлайн”, “сучасною”, “тенденцією” }

В наборі 2 загальних слова та 10 слів в цілому.
Отже, індекс подібності Жаккара двох документів дорівнює: 2 / 10 = 20%
“Торбина слів”, ймовірно, буде найкращим вибором для дуже коротких типів документів, таких як заголовки статей або пошукові запити, де контекст навряд чи має значення. Наприклад, якщо у вас є дошка пошуку роботи, і ви хотіли б рекомендувати назви вакансії користувачам, які шукали подібний термін, то підхід “торбина слів”,

ймовірно, був би найкращою моделлю.

1.4. Метод Евклідової відстані

В NLP(natural language processing) тексти представлені у вигляді векторів. Отже, з математичної точки зору, який спосіб порівняння векторів буде найбільш підходящим? Інтуїтивно, порівняння відстані між двома векторами здається найбільш логічним підходом. Метрикою, яка вимірює відстань між двома векторами, є евклідова відстань.

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Рис. 1.3. Формула евклідової відстані

Відстань між векторами x і y залежить від величини цих векторів. У цьому випадку збільшення величини вектора x або y призведе до збільшення евклідової відстані. З точки зору НЛП, це означає, що більша частина тексту, яка містить більше слів з точки зору різноманітності та частоти, матиме набагато більшу величину, ніж менший основний текст, навіть якщо вони мають однакову тему.

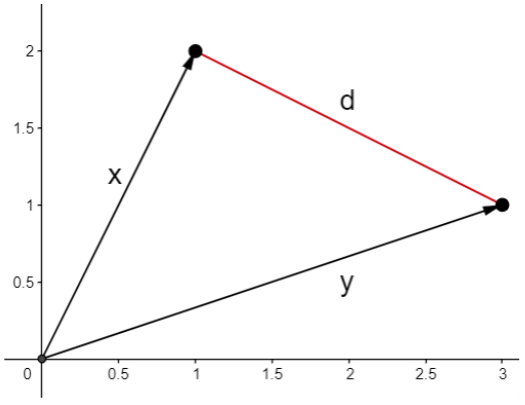

Рис. 1.4. Графічна ілюстрація евклідової відстані

Це може бути проблемою, оскільки людей, які порівнюють документи, більше цікавить тема документів, ніж кількість тексту. Розглянемо пошукові системи, на які ми так покладаємося. Якби ми шукали «повітряні кулі» в Google, ми б хотіли, щоб нам показували статті, найбільш релевантні для повітряних куль. Ми не хотіли б обмежуватися результатами, які мають стільки ж, скільки й сам запит. З цієї причини евклідова відстань може бути не найкращим показником подібності документів.

1.5. Метод порівняння Word2Vec та Doc2Vec

У 2013 році чеський вчений-комп'ютерник Томаш Міколов мав ідею представити слова в «семантичному просторі», де кожне слово має набір координат у просторі, а слова, близькі один до одного за значенням, мають коротку відстань між ними. Цей алгоритм називається word2vec і походить від ідеї, що слова, які зустрічаються в подібних контекстах, схожі семантично. Вхідні на виході векторні уявлення слів дозволяють обчислювати «семантичну відстань» між словами.

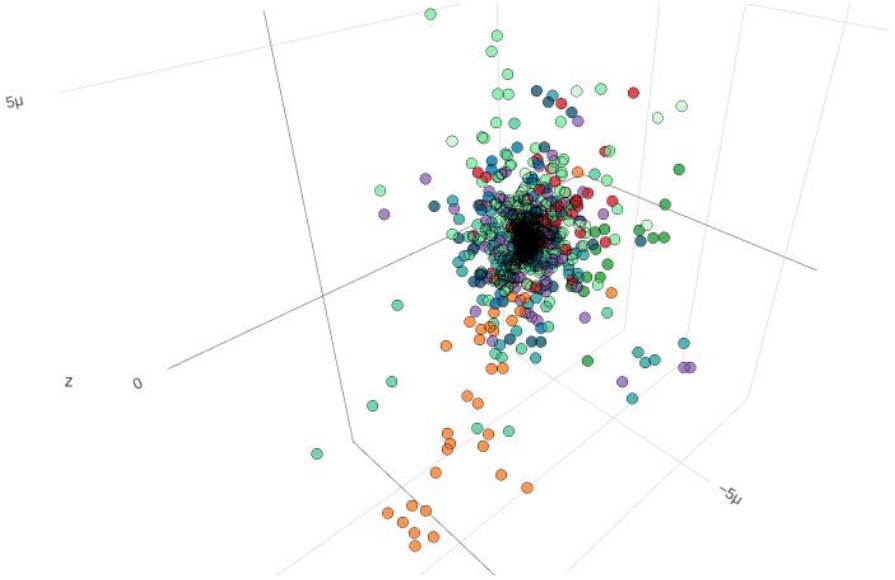

Рис. 1.5. Семантичний простір векторизованих слів

Так можна знаходити схожі за значенням слова. Зазвичай наводять приклад із королем

і королевою: король ставиться до чоловіка так само, як королева до жінки. Word2vec виконує прогнозування на основі контекстної близькості цих слів. Так як інструмент word2vec заснований на навчанні простої нейронної мережі, щоб досягти його найбільш ефективної роботи, необхідно використовувати великі корпуси для навчання. Це дозволяє підвищити якість передбачень.

Концептуально подібні слова розташовані близько один до одного на графіку. Пізніше Міколов розширив word2vec на документи, створивши алгоритм, який може представляти будь-який документ як, наприклад. 500-вимірний вектор. Про схожість двох документів свідчить близькість їх векторів.

1.6. Метод косинусної подібності

Один з методів підходу вирішення проблеми пропонує розглядати кут між векторами замість відстані. Документи, представлені у вигляді векторів, можна порівнювати, оцінюючи кут між двома векторами. Метрикою, яка враховує кут двох векторів, є метрика подібності косинуса.

$$Cosine\ Similarity = cos(\theta) = \frac{x \cdot y}{||x||\ ||y||}$$

Рис. 1.6. Формула косинусної подібності

Значення подібності косинуса може варіюватися від 0 до 1, причому 0 пред ставляє найменшу подібність, а 1 - найвищу схожість.

Кут між векторами x і y залишиться незмінним незалежно від того, наскільки вектори x і y зміняться за величиною.

Наприклад, під час пошуку інформації та аналізу тексту кожному слову присвоюється окрема координата, а документ представляється вектором числа входжень кожного слова в документ. Таким чином, косинусна подібність дає корисну міру того, наскільки схожими можуть бути два документи з точки зору їхньої тематики

та незалежно від довжини документів. Техніка також використовується для вимірювання згуртованості всередині кластерів у сфері інтелектуального аналізу даних. Однією з переваг косинусної подібності є її низька складність, особливо для розріджених векторів: потрібно враховувати лише ненульові координати.

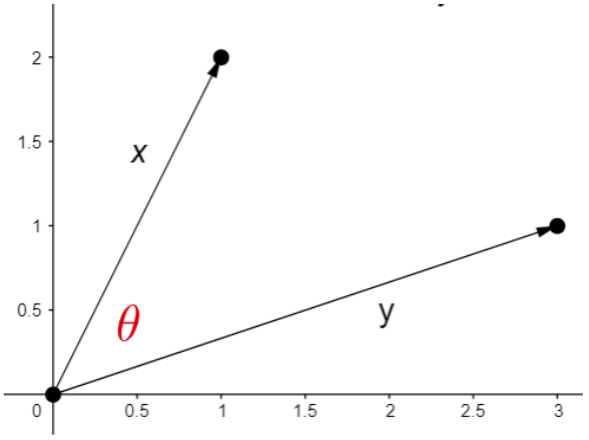

Рис. 1.7. Графічна ілюстрація косинусової подібності

Це означає, що косинусна подібність враховує кут між двома векторами, не досліджуючи величину векторів. Це усуває проблему, яку створює метрика евклідової відстані.

1.7. Метод Сьоренсена-Дайса

Коефіцієнт Сьоренсена-Дайса (**SDI**), також відомий як індекс Сьоренсена-Дайса (або **Dice**) - це статистичний показник, який використовується для вимірювання подібності двох вибірок сукупності.

$$SDI = 2 \times (A \cap B) / (A \uplus B)$$

Рис. 1.8. Формула розрахунку коефіцієнт Сьоренсена-Дайса

Спочатку використовувався в ботаніці, індексуючи подібність між популяціями флори

та фауни в різних областях, але він також використовується в інших областях. Її можна використовувати як функцію схожості тексту, дещо подібну до функції відстані редагування Левенштейна, хоча її перевага полягає в іншій області. Левенштейн хороший для пошуку орфографічних помилок, але покладається на те, що перевірене слово/фраза є досить схожою на бажану, і може бути дуже повільним для довгих слів/фраз. Sørensen-Dice більш корисний для «нечіткого» збігу часткових і погано написаних слів/фраз, можливо, у неправильному порядку. Існує кілька різних методів токенізації об'єктів для порівняння Сьоренсена-Дайс. Найбільш типовою схемою токенізації для тексту є розбиття слів на біграми: групи з двох послідовних літер.

Висновки до розділу 1

Виконана робота демонструє переваги та недоліки існуючих методів та алгоритмів для аналізу текстових даних.

Також результати дослідження показали, що для мети порівняння тексту, який підкреслює лексичну схожість, ігноруючи субстанцію значення, підходить підхід лексичної подібності. Ці вимірювання можна використовувати для виявлення дублювання або плагіату, не переймаючись контекстом документа. Підходи до подібності рядків принципово не залежать від мови, тому добре працюють для мов різних країн.

Семантичний підхід, здається, пропонує інтелектуальний вимір подібності. Цей вимір дуже підходить для пошуку тексту або документів, які дійсно схожі та відповідають змісту контексту. Однак семантична схожість зазвичай залежить від мови та предметної області, тому вона застосовується не до всіх мов. Інакше кажучи, якщо онтологія мови ще не доступна, її потрібно спочатку побудувати. Посилаючись на підходи до подібності тексту, видно, що семантична подібність дуже раціональна для пошуку подібності документів. Стосовно обробки текстових даних існує ряд моделей подібності документів. Відповідно до проведених досліджень характеристик методів текстового аналізу можна зробити рекомендацію підходу до проблеми подібності

документів, шляхом визначення завдання та золотого стандарту і вибравши метрику оцінки, а потім навчати низку моделей від простіших варіантів до більш складних альтернатив, поки не знайдеться модель. Для покриття потреб пошукової платформи метод Сьоренсена-Дайса є найбільш оптимальним та раціональним у використанні.

РОЗДІЛ 2. ЗАСОБИ ПРОГРАМНОЇ РОЗРОБКИ

Програмне забезпечення або інструмент програмування - це набір комп'ютерних програм, які використовуються розробниками для створення, обслуговування, налагодження або підтримки інших програм та програм.

Інструменти розробки програмного забезпечення - це просто інструменти (як правило, саме програмне забезпечення), які програмісти використовують для створення іншого програмного забезпечення. Наприклад, мовні бібліотеки, редактори коду, налагоджувачі тощо. У цю категорію міститься будь-який інструмент розгортання програмного забезпечення, який дозволяє програмісту створювати стабільне програмне забезпечення, що відповідає потребам або цілям клієнта. Інструменти гнучкої розробки можуть бути різних типів, таких як компонувальники, компілятори, редактори коду, дизайнери графічного інтерфейсу, асемблери, налагоджувачі, інструменти аналізу продуктивності та багато інших. Існують деякі фактори, які необхідно враховувати при виборі відповідного інструменту розробки в залежності від типу дизайну. Всі професіонали потребують інструментів розробки програмного забезпечення для виконання своєї роботи. Теслярі потрібні молотки, пилки, рубанки, рулетки тощо. Автомеханіку потрібні гайкові ключі та розетки, тріскачки та ударні інструменти. Сантехніку потрібні трубні ключі, інструменти для паяння, пилки і т. д.

Так само розробникам програмного забезпечення потрібні правильні інструменти планування програмного забезпечення для виконання відповідних завдань. Інструменти розробки програмного забезпечення відіграють дуже важливу роль у сфері ІТ, хоча вони менш істотні, ніж інструменти, які використовуються іншими фахівцями.

2.1. Серверна частина

Сервери відіграють важливу роль у здатності компаній до мережі та спільної роботи. Вони надають загальний ресурс, який можуть покластися всі сторони в мережі.

Визначення сервера це тип комп'ютера, який обмінюється інформацією з іншими комп'ютерами. Існують різні типи серверів, які пропонують різні послуги мереж різного розміру. Іншими словами, сервер може робити і виглядати по-різному. Це можуть бути комп'ютери, програми або жорсткі диски. Вони бувають різних форм, але всі виконують ту саму роботу. Кінцевою функцією сервера є отримання, зберігання та обмін даними; його можна порівняти з електронною картотекою. Однак сервери динамічніші, ніж картотечна шафа, тому що вони можуть миттєво обмінюватися даними по всій країні або всередині домогосподарства; це називається мережею сервера. Мережа сервера складається з комп'ютерів, з якими обмінюється інформацією.

Різні сервери надають різні послуги. Сервери виділені, тобто створені для однієї мети і не можуть змінюватися. Іншими словами, сервер друку не може стати веб-сервером. До найпоширеніших типів серверів належать поштові, друковані, веб-сервери, файлові та сервери додатків.

- Поштовий сервер — основний сервер надсилає та зберігає електронні листи в мережі. Багато компаній мають приватні поштові сервери, тому конфіденційність повідомлень, якими обмінюються всередині компанії, захищена.
- Сервер друку — сервер друку підключено до принтера і дозволяє всім клієнтам у мережі використовувати принтер. З сервером друку хтось на 5 поверсі може роздрукувати документ на 1 поверсі по бездротовій мережі.
- Веб сервер — використовує HTTP (протокол передачі гіпертексту) для зв'язку між браузером та сервером. Це ефективно витягує дані з сервера та відображає їх для клієнта. Якщо хтось шукає зображення гори Еверест, він запросить його у браузері, який працює із сервером для отримання інформації.

- Файловий сервер — це сервер, який зберігає та розповсюджує файли в мережі. Вони призначені для захисту, зберігання та розповсюдження даних серед призначених клієнтів.
- Сервер додатків — керує всіма програмами, що використовуються в організації або бізнесі. Їх іноді називають «проміжним ПЗ», тому що вони є посередниками між серверами баз даних та кінцевими користувачами.

2.1.1. Серверне середовище NodeJS

Як асинхронне подієве JavaScript - оточення, **Node.js** спроектований для побудови масштабованих мережевих додатків. У нижче наведений приклад "hello world", який може одночасно обробляти багато з'єднань. Для кожного з'єднання викликається функція зворотнього виклику, проте коли з'єднань немає Node.js засинає.

Рис. 2.1. Виконавче середовище «NodeJs»

Це контрастує з більш загальною моделлю в якій використовуються паралельні OS потоки. Такий підхід є відносно неефективним та дуже важким у використанні. Більше того, користувачі Node.js можуть не турбуватись про блокування процесів, оскільки немає жодних блокувань. Майже жодна з функцій у Node.js не працює

напряму з I/O, тому процес не блокується ніколи. Оскільки нічого не блокується на Node.js легко розробляти масштабовані системи [9].

Зазвичай поведінка визначається через функції зворотнього виклику на початку скрипта і в кінці запускає сервер через блокуючий виклик, як от EventMachine::run(). В Node.js немає нічого подібного на виклик початку циклу подій. Node.js просто входить в подієвий цикл після запуску скрипта на виконання. Node.js виходить з подієвого циклу тоді, коли не залишається зареєстрованих функцій зворотнього виклику. Така поведінка схожа на поведінку браузерного JavaScript: подієвий цикл прихований від користувача.

HTTP є об'єктом першого роду в Node.js, розробленим з потоковістю та малою затримкою. Це робить Node.js хорошою основою для веб–бібліотеки або фреймворку. Те що Node.js спроектований без багатопоточності, не означає, що ви не можете використовувати можливості кількох ядер у вашому середовищі. Ви можете створювати дочірні процеси, якими легко керувати з допомогою API child_process.fork(). Модуль cluster побудований на цьому інтерфейсі і дозволяє вам ділитись сокетами між процесами та розподіляти навантаження між ядрами.

Node.js дозволяє розробникам використовувати JavaScript для написання внутрішнього коду, хоча традиційно він використовувався у браузері для написання зовнішнього коду. Об'єднання фронтенду та бекенду зменшує зусилля, необхідні для створення веб-сервера, що є основною причиною того, чому Node.js є популярним вибором для написання внутрішнього коду. У цьому посібнику ви дізнаєтесь, як створювати веб-сервери за допомогою модуля http, включеного в Node.js. Ви створите веб-сервери, які можуть повертати дані JSON, файли CSV та веб-сторінки HTML. З точки зору веб-серверної розробки Node має ряд переваг:

- продуктивність — Node був розроблений для оптимізації пропускної спроможності та масштабованості у веб-додатках і дуже добре справляється з багатьма поширеними проблемами веб-розробки (наприклад, веб-програми реального часу);
- код написаний на "звичайному старому JavaScript", а це означає, що витрачається менше часу при написанні коду для браузера та веб-сервера

пов'язане з "перемиканням технологій" між мовами.

- JavaScript є відносно новою мовою програмування та має переваги від покращення дизайну мови порівняно з іншими традиційними мовами для веб-серверів (наприклад, Python, PHP тощо). Багато інших нових та популярних мов компілюються/конвертуються в JavaScript, тому ви можете також використовувати CoffeeScript, ClosureScript, Scala, LiveScript, etc.
- менеджер пакетів Node (NPM) забезпечує доступ до сотень тисяч багаторазових пакетів. Він також має кращий у своєму класі дозвіл залежностей і може використовуватися для автоматизації більшості інструментів побудови;
- портативний, має версії для Microsoft Windows, OS X, Linux, Solaris, FreeBSD, OpenBSD, WebOS і NonStop OS. Крім того, він має гарну підтримку серед багатьох хостинг-провайдерів, які часто надають конкретну інфраструктуру та документацію для розміщення сайтів, що працюють на Node;
- він має дуже активну сторонню екосистему та спільноту розробників, які завжди готові допомогти.

2.1.2. Фреймворк взаємодії з серверним середовищем ExpressJS

Для більш ефективної взаємодіє з середовищем виконання можна використовувати спеціальні фреймворки. **Express** є популярним веб-фреймворком, написаним на JavaScript і працюючим усередині середовища виконання node.js. Цей модуль висвітлює деякі ключові переваги цього фреймворку, встановлення середовища розробки та виконання основних завдань веб-розроблення та розгортання.

Express – найпопулярніший веб-фреймворк для Node. Він є базовою бібліотекою для інших популярних веб-фреймворків Node. Він надає такі механізми:

- написання обробників для запитів з різними HTTP-методами у різних URL-адресах (маршрутах);

- інтеграцію з механізмами рендерингу «view», для створення відповідей, вставляючи дані в шаблони;
- встановлення загальних параметрів веб-програми, таких як порт для підключення, та розташування шаблонів, які використовуються для відображення відповіді;
- "проміжне ПЗ" для додаткової обробки запиту в будь-який момент у конвеєрі обробки запитів;
- легка інтеграція з додатковими логічними пакетами;
- стандартизація найкращої практики реалізації логічних функцій.

Рис. 2.2. Фреймворк для серверного середовища ExpressJS

У той час, як сам express досить мінімальний, розробники створили сумісні пакети проміжного програмного забезпечення для вирішення практично будь-якої проблеми з веб-розробкою. Існують бібліотеки для роботи з файлами cookie, сеансами, входами користувачів, параметрами URL, даними POST, заголовками безпеки та багатьма іншими. Ви можете знайти список пакетів проміжного програмного забезпечення, підтримуваних командою Express у Express Middleware (поряд зі списком деяких популярних пакетів сторонніх виробників) [10].

Термін сесія або сеанс користувача відноситься до серії взаємодій користувача з програмою, яка відстежується сервером. Сеанси використовуються для підтримки певного стану користувача, включаючи постійні об'єкти (такі як дескриптори

компонентів EJB або набори результатів бази даних) та автентифіковані ідентифікатори користувачів серед багатьох взаємодій. Наприклад, сеанс можна використовувати для відстеження підтвердженого входу користувача в систему, за яким слідує ряд спрямованих дій для конкретного користувача.

Сам сеанс знаходиться на сервері. Для кожного запиту клієнт передає ідентифікатор сеансу у файлі cookie або якщо браузер не дозволяє файли cookie, сервер автоматично записує ідентифікатор сеансу в URL-адресу. Сервер програм Sun ONE підтримує стандартний інтерфейс сеансу сервлета, званий HttpSession, для всіх дій сеансу. Цей інтерфейс дозволяє вам писати безпечні сервлети, що переносяться. Файл cookie - це невеликий набір інформації, яка може бути передана браузеру, що викликає, який витягує її при кожному наступному виклику з браузера, щоб сервер міг розпізнавати виклики від того ж клієнта. Файл cookie повертається при кожному зверненні до сайту, на якому він був створений, доки не закінчиться термін його дії. Сеанси автоматично підтримуються файлом cookie сеансу, який надсилається клієнту під час першого створення сеансу. Файл cookie сеансу містить ідентифікатор сеансу, який ідентифікує клієнта для браузера при кожній подальшій взаємодії. Якщо клієнт не підтримує або не дозволяє файли cookie, сервер перезаписує URL-адреси, в яких ідентифікатор сеансу з'являється в URL-адресах від цього клієнта.

2.1.3. База даних MongoDB

Дані користувачів необхідно зберігати в базі даних і для цієї мети разом з обраним серверним середовищем виконання чудовим вибором буде **MongoDB**. MongoDB - це документно-орієнтована база даних NoSQL, що використовується для зберігання великих обсягів даних. Замість використання таблиць та рядків, як у традиційних реляційних базах даних, MongoDB використовує колекції та документи. Документи складаються з пар ключ-значення, які є основною одиницею даних MongoDB. Колекції містять набори документів та функцій, які еквівалентні таблицям

реляційних баз даних. MongoDB - це база даних, що з'явилася приблизно в середині 2000-х років [11].

Рис. 2.3. База даних «MongoDB»

Структура документа більше відповідає тому, як розробники створюють свої класи та об'єкти у відповідних мовах програмування. Розробники часто кажуть, що їхні класи – це не рядки та стовпці, а чітка структура з парами ключ-значення. Особливостями якої є:

- кожна база даних містить колекції, які містять документи. Кожен документ може бути різним із різною кількістю полів;
- розмір та зміст кожного документа можуть відрізнятися один від одного;
- для рядків (або документів, як вони називаються MongoDB) не потрібна заздалегідь визначена схема. Замість цього поля можна створювати на льоту;
- модель даних, доступна в MongoDB, дозволяє вам простіше представляти ієрархічні відносини, зберігати масиви та інші складніші структури;
- масштабованість. Середовище MongoDB дуже масштабується. Компанії по всьому світу створили кластери, деякі з яких містять понад 100 вузлів із мільйонами документів у базі даних;
- балансування навантаження — MongoDB використовує концепцію сегментування для горизонтального масштабування шляхом поділу даних між

кількома екземплярами MongoDB. MongoDB може працювати на декількох серверах, балансуючи навантаження та/або дублюючи дані, щоб підтримувати працездатність системи у разі збою обладнання.

2.1.4. Моделювання об’єктних даних Mongoose

Mongoose – це бібліотека моделювання об'єктних даних (ODM) для MongoDB та Node.js. Він керує відносинами між даними, забезпечує перевірку схеми та використовується для переведення між об'єктами в коді та поданням цих об'єктів у MongoDB.

MongoDB – це база даних документів NoSQL без схеми. Це означає, що можна зберігати в ньому документи JSON, і структура цих документів може змінюватись, оскільки вона не застосовується, як бази даних SQL. Це одна з переваг використання NoSQL, оскільки вона прискорює розробку програм та знижує складність розгортання. Mongoose використовує наступні технології:

- колекції — в MongoDB еквівалентні таблицям в реляційних базах даних. Вони можуть містити декілька документів JSON;
- документи — еквівалентні записам або рядкам даних у SQL. У той час як рядок SQL може посилатися на дані в інших таблицях, документи Mongo зазвичай поєднують їх у документі;
- поля - або атрибути аналогічні стовпцям у таблиці SQL;
- схеми — у той час, як MongoDB не має схеми, SQL визначає схему через визначення таблиці. "Схема" Mongoose - це структура даних документа (або форма документа), яка застосовується на рівні програми;
- моделі — це конструктори вищого порядку, які беруть схему та створюють екземпляр документа, еквівалентний записам у реляційній базі даних.

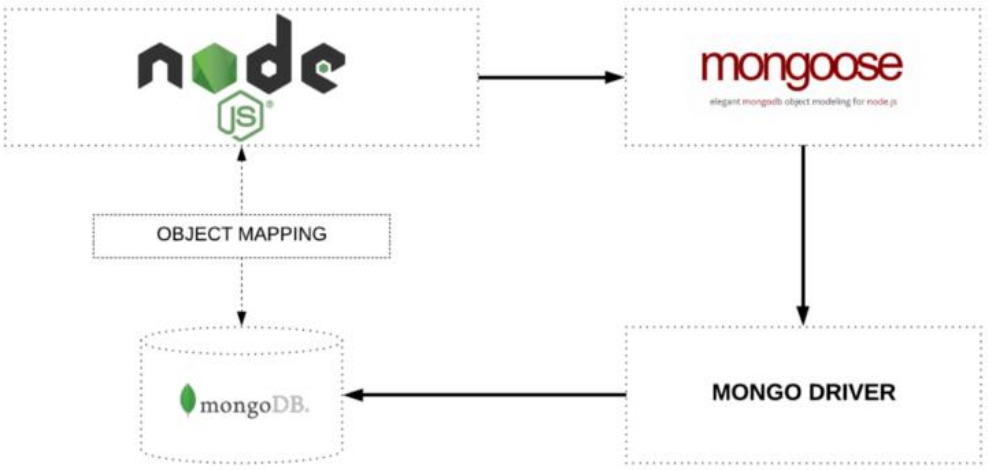

Рис. 2.4. Зіставлення об'єктів між Node та MongoDB, кероване через Mongoose

Mongoose має гнучкий API та надає безліч способів виконання завдання. Ми не фокусуватимемося на варіантах, тому що це виходить за рамки цієї статті, але пам'ятайте, що більшість операцій можна виконувати більш ніж одним способом або синтаксично, або через архітектуру програми.

Хоча ми можемо безпосередньо взаємодіяти з Mongo за допомогою драйвера Mongo, Mongoose спростить цю взаємодію, дозволяючи моделювати відносини між даними та легко їх перевіряти.

2.1.5. Утиліта шифрування даних BCrypt

Збереження паролів у вигляді звичайного тексту ніколи не повинно бути варіантом. Натомість нам потрібно надати безпеку з одностороннім рухом, хешуючи паролі. Раніше ми з'ясували, що лише хешування недостатньо для пом'якшення більш складних атак, таких як райдужні таблиці. Найкращою практикою зберігання паролів є включення солі в дескриптор хешування. Іншими словами, додайте додаткову довільну інформацію у вхідні дані хешування, які створюють секретне слово, щоб зробити хеш єдиним у своєму роді. Ідеальний етап перевірки має координувати ці дві

форми: послідовне хешування та соління.

Існує безліч криптографічних функцій на вибір, таких як сімейство SHA2 та сімейство SHA3. Одна з проблем проектування сімейств алгоритмів безпечного хешування (SHA) полягає в тому, що вони мають бути швидкими в обчислювальному плані. Від того, наскільки швидко криптографічний метод може згенерувати хеш, залежить, наскільки надійним та безпечним є пароль.

В даний час апаратне забезпечення, поряд з процесором та графічним процесором, дуже здатне. Він може обчислювати мільйони або, звичайно, мільярди хешів SHA-256 за хвилину проти вкраденої бази даних, що робить відмову в обслуговуванні (DoS), розподілену відмову в обслуговуванні (DDoS) або атаки грубої сили, що повторюються. Нам потрібна проміжна або помірна спроба хешування, тобто злому паролів, щоб практично зупинити зловмисників. Більш того, нам потрібно, щоб ця робота була універсальною, щоб ми були готові компенсувати майбутнє швидше обладнання, змушуючи функцію працювати все повільніше та повільніше з часом.

Цілісність та безпека даних завжди є найвищим пріоритетом. Алгоритм BCrypt використовується для безпечного хешування та солі паролів. BCrypt дозволяє створити рівень безпеки паролів, який може просувати прилеглі апаратні інновації для захисту від небезпек або загроз у довгостроковій перспективі, таких як зловмисники, які мають обчислювальну потужність, щоб подвоювати паролі швидше. Давайте заглибимося в специфікації та дизайн, які роблять BCrypt стандартом криптографічної безпеки.

Технології швидко змінюються. Розширення швидкості, потужності та можливостей управління комп'ютерами може надати перевагу як інженерам, які намагаються створити програмні системи, так і зловмисникам, які намагаються використовувати їх не за призначенням. Деякі криптографічні програми не призначені для масштабування з обчислювальним контролем. Як пояснювалося раніше, безпека пароля залежить від того, як швидко вибраний метод криптографічного хешування може обчислити хеш пароля. Швидкий метод буде виконуватися швидше при роботі на більш потужному устаткуванні.

Щоб пом'якшити цей вектор атаки, ми можемо створити метод криптографічного хешування, який можна налаштувати так, щоб він працював повільніше на недавно доступному устаткуванні, тобто метод масштабується з обчислювальною потужністю управління. Таким чином, у плані криптографічного результату для цієї проблеми ми повинні враховувати обладнання, що швидко розвивається, і постійну довжину пароля.

```
1  const bcrypt = require("bcrypt");
2  const saltRounds = 10;
3  const plainText = "EDYu9943^%*_79";
4  bcrypt
5    .genSalt(saltRounds)
6    .then(salt => {
7      console.log(`salt = {salt}`);
8      return bcrypt.hash(plainText, salt);
9    })
10   .then(hash => {
11     console.log(`hash = {hash}`);
12   })
13   .catch(err => console.error(err.message));
```

Рис. 2.5. Приклад використання утиліти BCrypt

На початку необхідно визначити три змінні. Перший для імпорту модуля BCrypt та захоплення його псевдонімом з ім'ям bcrypt, другий для визначення кількості раундів солі, необхідних для хешування. Тут ми відзначаємо, що чим більше сольових раундів, тим більше пароль буде хешуватися, і тим надійнішим буде наш пароль. Тут ми проводимо десять сольових раундів, а третій це текст, який ми хочемо хешувати. Потім ми повинні викликати вбудовану функцію genSalt() модуля BCrypt і передати кілька раундів солі як аргумент. У разі успіху в блоці "тоді" ми повертаємо хеш тексту, знову викликаючи хеш вбудованої функції модуля BCrypt.

2.2. Клієнтська частина

Веб-клієнт - це зовнішній інтерфейс або сторона користувача веб-архітектури. Це може бути веб-браузер або веб-додаток, який обмінюється даними через протокол передачі гіпертексту (HTTP) для форматування та передачі даних, таких як документи, зображення, відео та аудіофайли, з веб-сервера кінцевому користувачеві.

Веб-клієнт підключається до веб-серверів через Інтернет та надає кінцевим користувачам інтерфейс для взаємодії з веб-серверами. Він запитує дані або веб-контент через HTTP, і веб-сервер відповідає веб-клієнту, використовуючи той же протокол. Деякі з його ключових функцій – доступність для всіх користувачів, швидке завантаження контенту, сумісність із мобільними пристроями, ефективна обробка помилок та ефективна навігація.

2.2.1. Бібліотека інтерфейсів користувача React

React - це бібліотека для розробки інтерфейсу користувача на основі JavaScript. Facebook та спільнота розробників з відкритим вихідним кодом управляють ним. Хоча React це скоріше бібліотека, ніж мова, він широко використовується у веб-розробці. Бібліотека вперше з'явилася в травні 2013 року і зараз є однією з найпопулярніших інтерфейсних бібліотек для веб-розробки. React пропонує різні розширення для підтримки всієї архітектури програми, такі як Flux і React Native, крім простого інтерфейсу користувача.

Популярність React сьогодні перевершила популярність всіх інших фреймворків для фронтенд-розробки. Ось чому:

- просте створення динамічних програм: React спрощує створення динамічних веб-програм, тому що вимагає менше коду і пропонує більше функцій, на відміну від JavaScript, де код часто дуже швидко ускладнюється;
- покращена продуктивність: React використовує Virtual DOM, тим самим швидше створюючи веб-програми. Віртуальний DOM порівнює попередні

стани компонентів і оновлює лише ті елементи в реальному DOM, які були змінені замість повторного оновлення всіх компонентів, як це роблять звичайні веб-додатки;

- компоненти багаторазового використання: Компоненти з будівельними блоками в 1-й реактивній версії, і єдине додаток зазвичай складають багато компонентів. Ці компоненти можуть мати свою логіку та управління, і вони можуть бути збережені для зручної роботи, що, у свою чергу, значно скорочує час розробки програми;
- однонаправлений потік даних: React слідує односпрямованому потоку даних. Це означає, що при розробці React розробники часто вкладають дочірні компоненти в батьківські компоненти. Оскільки дані передаються в одному напрямку, стає простіше налагоджувати помилки та дізнаватися, де в додатку виникає проблема на даний момент;
- невелика крива навчання: React легко освоїти, оскільки він в основному поєднує базові концепції HTML і JavaScript з деякими корисними доповненнями. Тим не менш, як і у випадку з іншими інструментами та фреймворками, вам потрібно витратити деякий час, щоб отримати правильне уявлення про бібліотеку React;
- його можна використовувати для розробки як веб-застосунків, так і мобільних: ми вже знаємо, що React використовується для розробки веб-застосунків, але це ще не все, на що він здатний. Існує фреймворк під назвою React Native, похідний від самого React, надзвичайно популярний і використовується для створення красивих мобільних додатків. Так що насправді React можна використовувати для створення веб-додатків, так і мобільних;
- спеціальні інструменти для легкого налагодження: Facebook випустив розширення для Chrome, яке можна використовувати для налагодження.

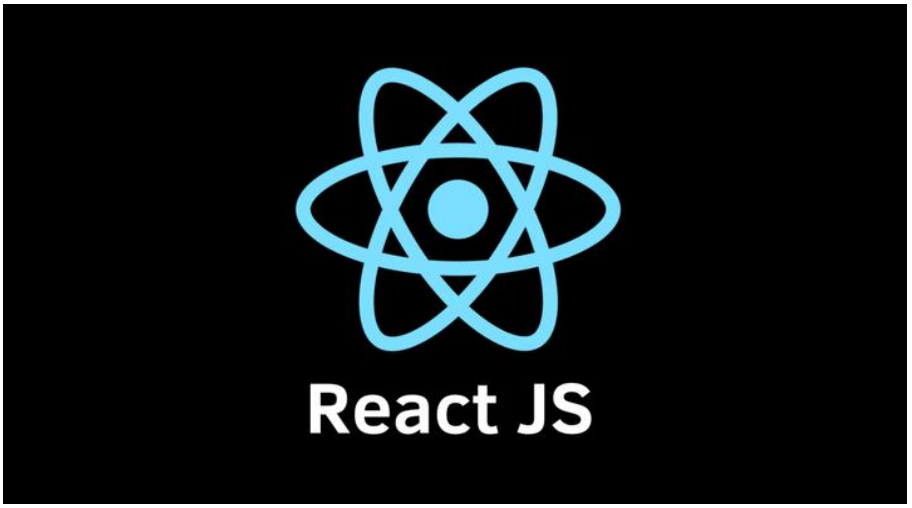

Рис. 2.6. Бібліотека інтерфейсів користувача React

В архітектурі Model View Controller (MVC) React є «поданням», що відповідає за те, як додаток виглядає та працює. MVC – це архітектурний шаблон, який поділяє рівень програми на модель, уявлення та контролер. Модель відноситься до всієї логіки, пов'язаної з даними; подання використовується для логіки інтерфейсу користувача, а контролер є інтерфейсом між моделлю і поданням. Оскільки існує широка спільнота розробників, програми React прості та легко тестуються. Facebook надає розширення для браузера, яке спрощує та прискорює налагодження React. Це розширення, наприклад, додає вкладку React до опцій інструментів розробника у веб-браузері Chrome. Вкладка дозволяє легко перевіряти компоненти React безпосередньо. Тепер, коли ви знаєте ключові особливості React, перейдемо до розуміння стовпів React.

2.2.2. Фреймворк Next.js для роботи з React бібліотекою

Next.js - це платформа JavaScript, яка дає змогу створювати надшвидкі та надзвичайно зручні статичні веб-сайти, а також веб-додатки за допомогою React. Насправді, завдяки автоматичній статичній оптимізації «статичний» і «динамічний» стали одним цілим. Ця функція дозволяє Next.js створювати гібридні програми, які містять як відтворені на стороні сервера, так і статично згенеровані сторінки. Next.js

широко використовується найбільшими та найпопулярнішими компаніями в усьому світі, такими як Netflix, Uber, Starbucks або Twitch. Він також вважається одним із найшвидше зростаючих фреймворків React, ідеально підходить для роботи зі статичними сайтами – що було найгарячішою темою у світі веб-розробки останнім часом.

Next.js зараз є одним із найпопулярніших фреймворків React для створення надшвидких та суперзручних для SEO веб-сайтів Jamstack. Його можна ідеально поєднувати з безголовими системами керування контентом або платформами електронної комерції для досягнення надзвичайної продуктивності та результатів SEO. Дизайн також важливий – якщо ви використовуєте теми чи шаблони, швидше за все, хтось має схожий на вигляд макет. Це також означає, що не можна створити унікальний клієнтський досвід і покращити його з часом. Навіть якщо це означає змінити одну просту річ, наприклад додати кнопку на сторінку продукту або видалити її. На щастя, завдяки Next.js ви можете створити повністю налаштований користувацький досвід. Давайте подивимося, що це насправді означає.

- свобода UX — вам не потрібно обмежувати себе будь-якими плагінами, шаблонами чи будь-якими іншими обмеженнями, продиктованими платформами електронної комерції чи CMS. Це дає вам повну свободу налаштовувати інтерфейс так, як вам потрібно або забажаєте. Це також дозволяє вносити творчі зміни без будь-яких обмежень;
- адаптивність і швидкість реагування — веб-сайти та веб-додатки, створені за допомогою Next.js, працюють на будь-якому пристрої та адаптуються до будь-якого розміру екрана чи роздільної здатності. Таким чином, користувачі можуть отримати доступ до вашого веб-сайту або веб-додатку зі свого улюбленого пристрою;
- короткий час завантаження сторінки – веб-сайти Next.js надзвичайно швидкі, оскільки вони статичні, тому відвідувачі будуть більш ніж задоволені продуктивністю;
- безпека даних — у випадку статичних веб-сайтів немає прямого зв’язку з базою даних, залежностями, даними користувачів чи іншою конфіденційною

інформацією, що робить їх абсолютно безпечними.

Усі ці речі, згадані вище, роблять користувальницький досвід настільки чудовим, наскільки це можливо. Але на цьому переваги використання Next.js не закінчуються. Ще одна вагома причина вибрати Next.js - це його ефективність SEO. Він використовує відтворення на стороні сервера (SSR) і водночас може бути чудовим генератором статичного сайту (SSG). Веб-сайти Next.js надшвидкі, їх легко сканувати та забезпечують чудову взаємодію з користувачем, тому Google віддасть їм перевагу над іншими та поставить їх вище [13].

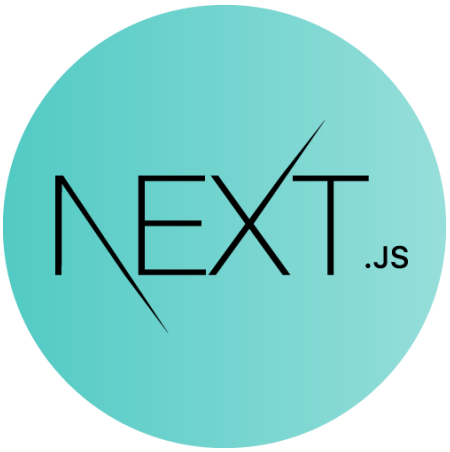

Рис. 2.6. Фреймворк Next.js

Незалежно від того, чи шукаєте ви переваги з точки зору бізнесу чи технічної, ви знайдете кілька причин, щоб серйозно розглянути вибір Next.js. Якщо ви хочете створити складний і вимогливий додаток, природа розробки React Next.js дозволяє заощадити багато часу. Розробники особливо віддають перевагу таким функціям, як:

- Zero Config — Next дозволяє вам зосередитися на бізнес-логіці вашої програми замість логіки програми. І щоб допомогти вам, він забезпечує автоматичне компілювання та групування. Іншими словами, Next оптимізовано для виробництва з самого початку;
- інкрементна статична регенерація — дозволяє оновлювати сторінки, повторно відтворюючи їх у фоновому режимі, коли надходить трафік. Іншими словами, статичний вміст може стати динамічним;

- гібрид візуалізації на стороні сервера SSR і статичної генерації сайту SSG – попередня візуалізація сторінок під час створення або запиту в одному проекті;
- підтримка TypeScript — автоматичне налаштування та компіляція TypeScript;
- швидке оновлення — швидке редагування в реальному часі – зміни, внесені в компоненти React, стають оперативними за лічені секунди. Він працює аналогічно гарячій заміні модулів (HMR);
- парсери CSS — можливість імпортувати файли CSS з файлу JavaScript. Нові аналізи покращили обробку CSS;
- вбудований компонент зображення та автоматична оптимізація зображення – ця функція автоматично оптимізує зображення;
- автоматичне поділ коду — автоматично зменшує розмір сторінки шляхом поділу коду та обслуговування компонентів лише за потреби. Модулі також можна автоматично імпортувати завдяки опції динамічного імпорту;
- отримання даних — ця опція дозволяє рендерити вміст різними способами відповідно до сценарію використання програми. Це можна зробити шляхом попереднього рендерингу за допомогою рендеринга на стороні сервера SSR або створення статичного сайту, а також шляхом оновлення або створення вмісту за допомогою ISR.

2.2.3. Бібліотека компонентів інтерфейсу MUI

MUI - це величезна бібліотека компонентів інтерфейсу користувача, які дизайнери та розробники можуть використовувати для створення програм React. Проект із відкритим вихідним кодом відповідає інструкціям Google щодо створення компонентів, надаючи вам налаштовану бібліотеку базових і розширених елементів інтерфейсу користувача. MUI також продає колекцію шаблонів і інструментів React, надаючи вам готові користувацькі інтерфейси для налаштування для вашого проекту.

Дизайнери часто використовують набори інтерфейсу користувача для створення нових продуктів або доповнень для існуючих проектів. Ці бібліотеки дозволяють дизайнерам перетягувати компоненти, необхідні для швидкого проектування інтерфейсів.

- швидший час виходу на ринок — у сучасному висококонкурентному технологічному середовищі час виходу на ринок є показником, який організації завжди прагнуть оптимізувати. Бібліотека компонентів дає дизайнерам і розробникам величезну перевагу з ретельно перевіреними елементами інтерфейсу користувача, готовими до роботи. Дизайнери можуть перетягувати елементи, щоб створювати користувацькі інтерфейси та налаштовувати компоненти відповідно до вимог продукту та брендингу. Команди дизайнерів можуть витрачати більше часу на розробку чудового досвіду для клієнтів, а не загрузнути в створенні та тестуванні компонентів інтерфейсу користувача з нуля - процес, який значно збільшує час виходу на ринок! Тестування зручності використання відбувається набагато швидше, оскільки дизайнери можуть швидко створювати прототипи, тестувати та повторювати. Якщо інтерфейс користувача не працює під час тестування, вони можуть миттєво вносити зміни, користуючись величезною бібліотекою, щоб отримати миттєвий відгук від учасників і зацікавлених сторін;
- єдине джерело істини — однією з найбільших проблем управління системою розробки є збереження єдиного джерела правди. Це не рідкість, коли команди продуктів, дизайнери UX і розробники мають несинхронізовані системи дизайну, що призводить до помилок, переробок і величезних головних болів і проблем для DesignOps. Використання бібліотеки компонентів MUI може значно зменшити ці проблеми, створюючи єдине джерело правди між проектуванням і розробкою. Дизайнери та інженери все ще матимуть окремі системи проектування (на основі зображень для дизайнерів і коду для інженерів), але MUI дає їм однакові стартові блоки. Використовуючи Merge з редактором на основі коду UXPin, дизайнери та інженери використовують ті самі компоненти системи проектування, синхронізовані через єдине сховище. Будь-які оновлення сховища синхронізуються з UXPin, повідомляючи

дизайнерів про зміни. Ви можете підключити Merge за допомогою Git для бібліотек компонентів React або Storybook для інших популярних технологій;

- послідовність дизайну — узгодженість є життєво важливою для взаємодії з користувачем, формування довіри та лояльності до бренду. Використання однакових компонентів інтерфейсу дозволяє дизайнерам підвищити узгодженість, мінімізуючи помилки та переробку;
- масштабованість — є ще одним важливим фактором дизайну продукту. Якщо ви створюєте систему проектування з нуля, дизайнери повинні спроектувати, прототипувати та протестувати нові компоненти перед масштабуванням продукту.
- завдяки повній бібліотеці інтерфейсу користувача MUI дизайнери можуть шукати компоненти, які їм потрібні для створення прототипу та негайного масштабування. Інженери можуть копіювати/вставляти ідентичні компоненти React з MUI і налаштовувати їх відповідно до специфікацій дизайнера.

MUI X містить бібліотеку розширених компонентів React, які команди можуть використовувати для ще більшого масштабування складних продуктів, включаючи сітки даних, засоби вибору дати, діаграми, розбивку на сторінки, фільтрацію тощо;

- легке обслуговування — бібліотека компонентів, як-от MUI, постачається з детальною документацією щодо встановлення, використання, оновлення та налаштування компонентів. Дизайнери та інженери можуть використовувати цю структуру для підтримки системи проектування організації, полегшуючи створення систем управління та протоколів. MUI також містить інструкції з переходу з однієї версії на іншу. Таким чином, організації можуть скористатися перевагами найновіших стилів інтерфейсу користувача, технологій і тенденцій кожного разу, коли MUI випускає оновлення;
- доступність — хто має досвід налагодження системи проектування, знатимуть, скільки часу та грошей потрібно, щоб переконатися, що кожен компонент відповідає стандартам доступності. Розробники MUI дуже ретельно розробляли компоненти, щоб вони відповідали вимогам доступності WCAD 2.0, зменшуючи роботу дослідників і дизайнерів. Важливо зауважити, що навіть

коли ви розробляєте інтерфейси з використанням доступних компонентів, ви все одно повинні тестувати навігацію та потоки користувачів, щоб переконатися, що продукт у цілому відповідає стандартам доступності;

- розширення навичок — бібліотека інтерфейсу користувача з відкритим кодом компонентів MUI дає змогу стартапам і молодим підприємцям створювати нові продукти, особливо в країнах, що розвиваються, де вони не мають однакового доступу до освіти, наставництва та передачі навичок. Бібліотека також надзвичайно корисна для благодійних, некомерційних, НУО та подібних організацій, які хочуть розробляти продукти та інструменти, але не мають бюджету, щоб інвестувати в систему дизайну. Будь-хто може використати навички талановитих дизайнерів і розробників MUI, використовуючи ту саму бібліотеку компонентів, яку використовують компанії зі списку Fortune 500, для розробки складних цифрових продуктів і конкуренції на глобальному ринку.

Рис. 2.7. Бібліотека MUI

Google Material Design UI, можливо, є однією з найкращих і найповніших бібліотек дизайну в світі. Спираючись на Material Design, MUI надає відповідну

бібліотеку компонентів React. Можливість легкого налаштування MUI за допомогою функції Theming і чудової документації бібліотек роблять його доступним для створення продуктів для транснаціональних корпорацій або одного розробника з ідеєю продукту. Оскільки MUI настільки широко використовується, існує величезна глобальна спільнота дизайнерів, дослідників і розробників, до яких можна звернутися за порадами та підтримкою. Крім того, що React є одним із найпопулярніших інтерфейсних фреймворків, MUI стає привабливою бібліотекою компонентів.

2.2.4. Сервіс геолокації Leaflet

API геолокації дозволяє користувачеві надавати своє місцезнаходження веб-додаткам, якщо вони того бажають. З міркувань конфіденційності користувач запитує дозвіл повідомляти інформацію про місцезнаходження. WebExtensions, які бажають використовувати об’єкт Geolocation, повинні додати дозвіл «geolocation» до свого маніфесту. Операційна система користувача запропонує користувачу надати доступ до місцезнаходження під час першого запиту.

Доступ до API геолокації здійснюється за допомогою виклику navigator.geolocation; це призведе до того, що браузер користувача запитуватиме дозвіл на доступ до даних про місцезнаходження. Якщо вони приймуть, то браузер використовуватиме найкращі доступні функції пристрою для доступу до цієї інформації (наприклад, GPS). Тепер розробник може отримати доступ до цієї інформації про місцезнаходження кількома способами:

- Geolocation.getCurrentPosition(): отримує поточне місцезнаходження пристрою.
- Geolocation.watchPosition(): реєструє функцію обробки, яка автоматично викликається щоразу, коли змінюється положення пристрою, повертаючи оновлене розташування.

В обох випадках виклик методу приймає до трьох аргументів:

- Обов’язковий успішний зворотний виклик: якщо пошук місцезнаходження успішний, зворотний виклик виконується з об’єктом GeolocationPosition як єдиним параметром, надаючи доступ до даних про місцезнаходження.
- Необов’язковий зворотний виклик помилки: якщо пошук розташування невдалий, зворотний виклик виконується з об’єктом GeolocationPositionError як єдиним параметром, надаючи доступ до інформації про те, що пішло не так.
- Додатковий об’єкт, який надає параметри для отримання даних про місцезнаходження.

Рис. 2.8. Сервіс геолокації

Leaflet - бібліотека JavaScript для створення карт. React використовує Virtual DOM, який створює відмінності попередньої та фактичної структури DOM і оновлює DOM, коли це необхідно. Це означає, що React відповідає за оновлення DOM. Навпаки, Leaflet має імперативний API і безпосередньо маніпулює DOM. Через цю різницю інтеграція React і Leaflet спочатку може бути не інтуїтивно зрозумілою.

Висновки до розділу 2

Програмне забезпечення стало значною частиною нашого повсякденного життя. Ми дуже покладаємося на них для ефективного виконання наших операцій. Будь то мобільні програми чи веб-сайти, інновації, пов’язані з програмним забезпеченням, стрімко набирають темп. У результаті для компаній стало вкрай важливо не відставати від швидкоплинного світу. Однак нам потрібні найкращі інструменти розробки програмного забезпечення, щоб створити робоче програмне забезпечення ефективно

та за мінімальний час. Оскільки розробка програмного забезпечення набуває популярності в індустрії програмного забезпечення, розробники та організації постійно шукають способи полегшити собі життя. Правильні інструменти можуть допомогти вам швидко отримати максимум від кожного дня, але вибрати свій арсенал найкращих інструментів для розробки програмного забезпечення нелегко. Інструменти розробки програмного забезпечення - це рішення, які програмісти використовують для створення, редагування, обслуговування та налагодження програм або програм. Компонувальники, компілятори, редактори коду, асемблери, налагоджувачі, інструменти тестування програмного забезпечення тощо є кількома прикладами. Ці інструменти доступні в платній версії та корпоративній версії. Крім того, ви можете отримати безкоштовну пробну версію або придбати повну версію для кращих результатів.

РОЗДІЛ 3. ПРОБЛЕМАТИКА ПОШУКОВОЇ ПЛАТФОРМИ

Мільйони людей використовують різноманітні пошукові системи, щоб отримати необхідні набори даних та задовільнити власні потреби. З розвитком технологій, кількість пошукових систем збільшилася в рази. Відповіді на всі ці запити, зберігаються на безлічі різних ресурсів. Завдання пошукової системи, систематизувати для користувача видачу наборів даних, які можуть задовольнити його запит необхідною, максимально релевантною інформацією.

Відповідаючи на питання, що таке пошукова система, можна сказати, що це величезне сховище даних, віртуальна бібліотека з дуже розумним механізмом обробки, в якій кожен, може знайти ту інформацію, яка йому необхідна в конкретний момент часу. А спеціальні методи та алгоритми, відфільтрують і відсортують цю інформацію так, щоб люди не витрачали зайвого часу.

У пошукових систем, є дві основні функції. Перша - це зібрати інформацію з відповідних джерел і обробити її, помістивши в сховище даних вже в тому вигляді, який буде зручний для пошуку інформації за критеріями. Друга - це видати інформацію користувачеві за запитом, з максимально відповідним вмістом його запиту вигляді. Тобто, що б корисні дані були зверху, і так по спадаючій.

Для того, щоб зібрати інформацію, її контент, пошукові системи засилають реалізують спеціальні алгоритмі на основі математичних методів, для аналізу даних.

Алгоритм, аналізуючи набір даних, збирає інформацію про його текстовий контент, зображеннях, медіа файли. Зібравши інформацію і передавши її в сховище даних, пошукова система аналізує ці дані, структурує і визначає релевантність цієї інформації пошуковим запитам.

Релевантність і відповідно ранжування інформації по певних запитах, відбувається саме на етапі аналізу даних. Вводячи пошуковий запит в системі, наприклад, google, людина отримує вже заздалегідь створених список сайтів, ранжируваних відповідно до релевантності запиту. Періодично, пошукові алгоритми повертаються на сайти, і перевіряють, чи змінилася інформація на них. Якщо так, інформація знову обробляється, і процес визначення подібності і ранжування сторінок оновлюється.

3.1. Проблема розміщення оголошень розшуку

Більшість із нас у добре забезпечених ресурсами демократичних суспільствах живуть із само собою зрозумілими гарантіями у звичайному житті, в якому наші живі близькі майже завжди доступні для контакту або знають, що вони десь знаходяться. Для деяких, однак, це почуття безпеки виявляється під загрозою, коли зникає член сім'ї, друг або колега або втрачають цінні особисті речі.

Розглядається, які емоційні дії накопичуються у просторі відсутності для людей, що залишилися позаду пошуків зниклої людини. Як виявилося, пошук зниклої людини - це емоційний процес, також відзначений (часто конкуруючими) географічними знаннями і складними відносинами з поліцейськими, яким доручено знайти зниклого (цей процес може значно відрізнятися в різних місцях).

Нинішній підхід до розуміння цього питання має наслідки для повідомлень про зниклих безвісти, які розглядає поліція, яка часто є першою, до кого звертаються за пошуком зниклих безвісти, а також для розробки соціальної та державної політики. На додаток до цього, хоча виявлені фактори ризику, пов'язані з зникненням безвісти, можуть бути інформативними, в поточній літературі немає ясності та зв'язку між цими факторами, щоб встановити групи низького та високого ризику для керівництва оцінкою ризику та профілактичними заходами для поліції.

Основним, що є у процесі пошуку загубленої людини чи особистих речей є час. Чим швидше публіка та правоохоронні органи дізнаються про інцидент тим більше ймовірність вирішення проблеми. Недоліком підходу одностайної взаємодії з поліцією

у таких випадках є значні витрати часу, оскільки відповідно до більшості правоохоронних систем діють обмеження коли вони мають праву оголосити людину у розшук лише після періоду у 3 дні коли людина дійсно не вийшла на зв’язок. Також процедура оформлення подібних актів вимагає оформлення великої кількості формальних документів. А публікація подібних оголошень відбувається на інституційних платформах про які громадськість мало обізнана та не надають можливості особистого пошуку за критеріями серед колосальної кількості інших оголошень.

Та у випадку загублення чи викрадення особистих речей, з усією повагою до правоохоронних органів, мають менш важливий пріоритет для працівник поліції та відповідно до статистики у більшості випадках подібні розслідування заходять у глухий кут.

3.2. Мета пошукової платформи

Як було описано у попередньому підрозділі основною проблемою у розміщенні оголошень з розшуку є значні витрати часових ресурсів.

Саме тому основною метою пошукової платформи є максимізація спрощення публікації оголошення з моменту виникнення інциденту до моменту його виявлення шляхом реалізації аналітичних методів порівняння текстових даних та методів сповіщення користувача.

Підхід використання централізованої пошукової платформи дозволяє забезпечити єдине джерело правди та стати єдиною асоціацією для кожної людини, яка на особистому досвіді відчує подібні труднощі.

Джерело єдиної правди означає, що кожен буде посилатися саме на нього, що дає можливість концентрації усіх випадків в одному місці та надає спроможність скоротити період виконання операцій пошуку та аналізу даних, щоб отримати бажаний результат.

Також це означає, що усі операції будуть проводитися в межах та умовах одного

середовища, таким чином позбавляє необхідності обдумувати гібридні рішення для взаємодії різних систем та зменшує ризики виникнення конфліктів базованих різноманітністю середовищ виконання.

На додаток, платформа передбачає реалізацію функціоналу ефективного пошуку та простої взаємодії з його результатами, що покращує досвід користувача з програмним продуктом та зміцнює його довіру у використанні даної платформи.

3.3. Структура пошукової платформи

Платформа складається з чотирьох основних частин, які забезпечують нормальне та повноцінне функціонування системи в цілому.

А саме: сторінка авторизації користувача системі, сторінка перегляду оголошень за категоріями, сторінка створення власних оголошень та сторінка відображення сповіщень, які характеризують подібність з існуючими оголошеннями.

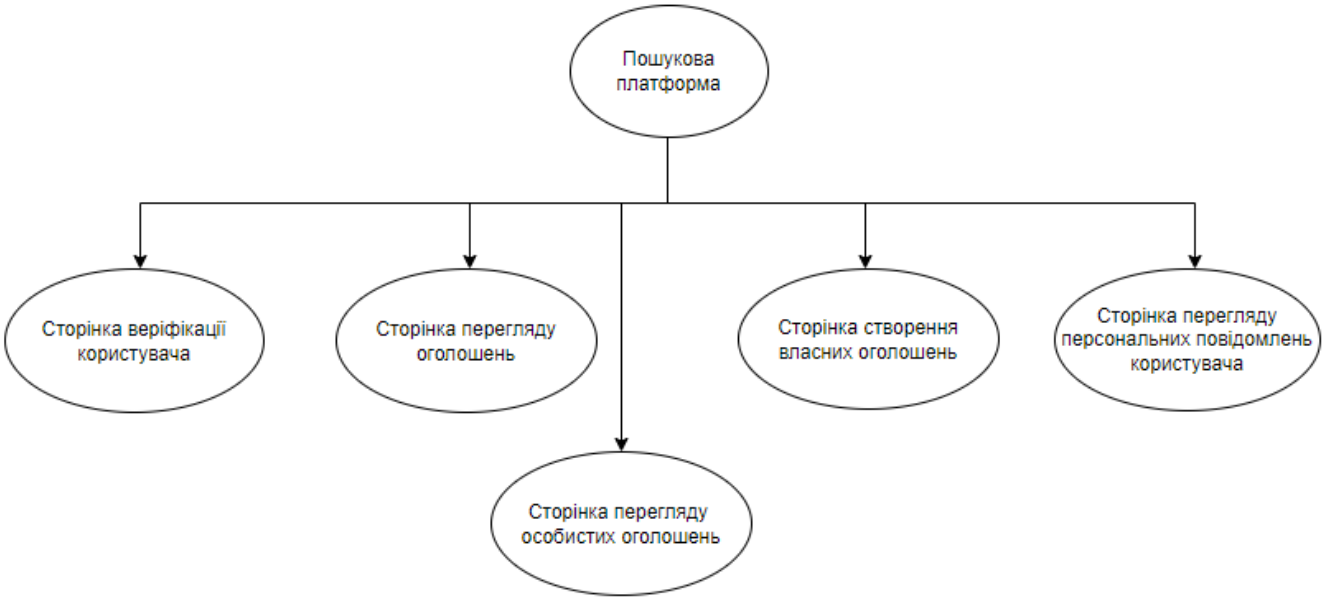

Рис. 3.1. Структурна схема платформи

Кожна з частин платформи надає у використання свої особливі функції та доступ до відповідної інформації, яка необхідна користувачеві.

Головними аспектами кожної системи, незалежно від проблематики якою вона оперує є наступні критерії:

- безпека особистих даних;
- інтуїтивне розташування компонентів інтерфейсу;
- інформативність виконання кожної функції;
- швидкодія обробки результатів необхідних користувачеві;
- обробка потенційних помилок;
- механізм сповіщення користувача;
- раціональне використання ресурсів.

Рис. 3.2. Функціональна схема платформи

3.3.1. Інформаційна сторінка

Кожна платформа має власну інформаційну сторінку або ж «домашню» сторінку де розміщується основна інформація з напрямку роботи цієї платформи, контактні дані компанії, яка займається обслуговуванням сервісів платформи та розуміння в цілому: які задачі вирішує даний програмний продукт.

Також інформаційна сторінка містить спеціальні дані для опису наповнення, яке необхідне для коректного відображення та появлення у пошукових запитах браузера в глобальній мережі Інтернет. Це допомагає просувати сервіси платформи та збільшувати аудиторію користувачів, що формує загальне ставлення суспільства до цих сервісів та бажання їх використовувати.

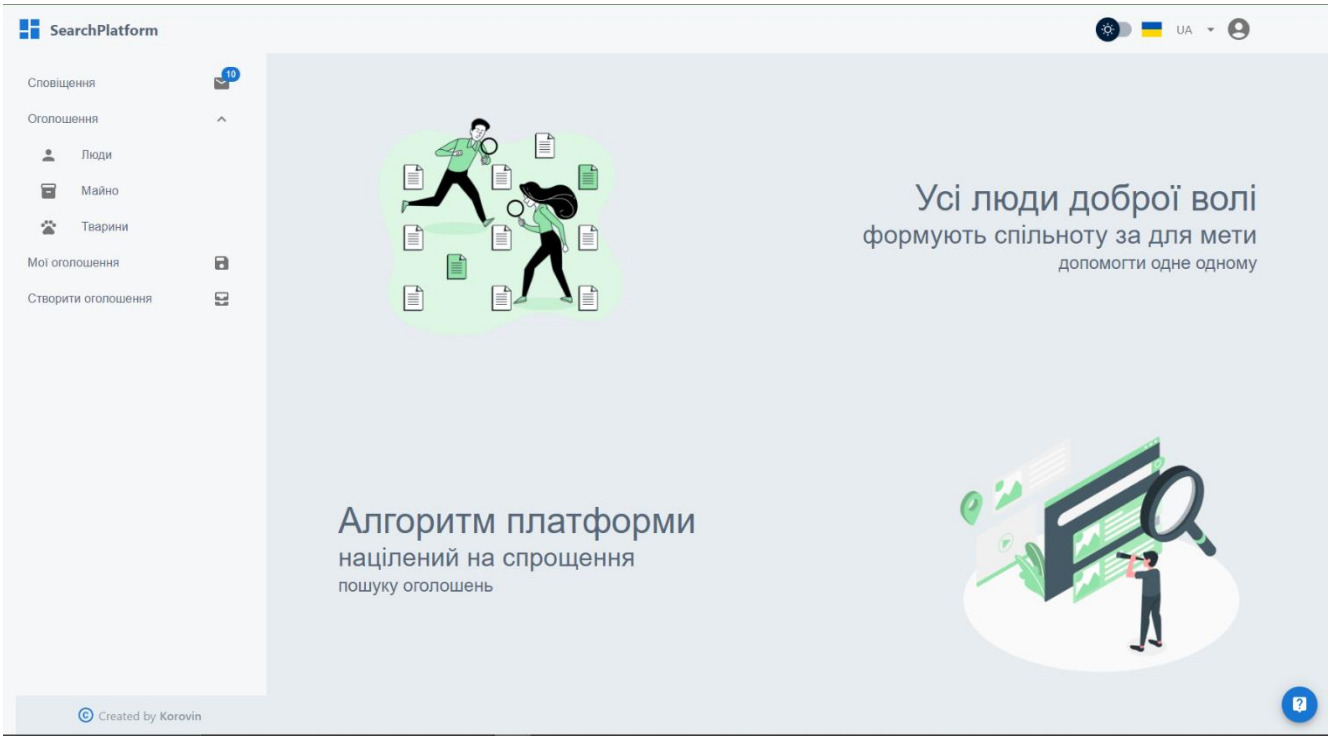

Рис. 3.3. Компонент інтерфейсу «Інформаційна сторінка»

Домашня сторінка - це головна сторінка сайту за умовчанням. Це перша сторінка, яку бачать відвідувачі, коли завантажують URL-адресу. Веб-менеджери можуть керувати домашньою сторінкою як спосіб керування користувальницьким досвідом. Домашні сторінки розташовані в кореневому каталозі веб-сайту. Багато домашніх сторінок

діють як віртуальний каталог для сайту - вони надають меню верхнього рівня, де відвідувачі можуть глибше перейти до різних розділів сайту. Наприклад, типовий веб-сайт має домашню сторінку з такими пунктами меню, як «про нас», «контакти», «продукти», «послуги», «преса» або «новини». Крім того, домашня сторінка часто служить для орієнтування відвідувачів, надаючи заголовки, заголовки, зображення та візуальні елементи, які показують, про що веб-сайт, а в деяких випадках, хто ним володіє та підтримує його. Одним із найкращих прикладів є звичайний веб-сайт компанії, на якому назва компанії розміщена на видному місці та часто містить логотип, а також показує зображення, пов’язані з цією компанією, наприклад, хто там працює, що компанія виробляє чи що це робиться в спільноті. Домашня сторінка є частиною природного шляху, яким Інтернет з'явився для орієнтованих користувачів Інтернету та допомагає їм орієнтуватися на багатьох сайтах у глобальній мережі. Немає стандартного макета домашньої сторінки, але більшість головних сторінок включають панель навігації, яка пропонує посилання на різні розділи веб-сайту. Іншими поширеними елементами, які можна знайти на домашній сторінці, є рядок пошуку, інформація про веб-сайт та останні новини чи оновлення. Деякі веб-сайти містять інформацію, яка змінюється щодня. Під час пошуку в Інтернеті зазвичай першим з’являється посилання на домашню сторінку, тому зазвичай увагу відвідувачів привертає саме цільова сторінка. Хороша домашня сторінка повинна містити посилання, щоб решту веб-сайту було зручно навігувати, бажано додати вікно пошуку, щоб пришвидшити взаємодію з користувачем. Чим більше користувачів матиме платформа, тим більше пошукові механізми браузера будуть рекомендувати її для нових потенційних учасників.

3.3.2. Керування статусом користувача

Платформи подібного типу повинні бути забезпечені механізмами контролю аутентифікації користувачів з метою посилення інформаційної безпеки та надання захисту персональних даних кожного учасника системи.

Аби уникнути усілякого роду вандалізму та неправдоподібного роду інформації реалізована система аутентифікації користувача, яка передбачає спрощення процесу інтеграції шляхом верифікації через головні сервіси, які вже мають данні користувача. Також система надає можливість зареєструватися традиційним шляхом через використання особистої поштової адреси, але за умови підтвердження права власності через поштове та мобільні повідомлення. Авторизація – це процес надання будь-кому можливості доступу до ресурсу. Звичайно, це визначення може здатися неясним, але багато ситуацій у реальному житті можуть допомогти проілюструвати, що означає авторизація, щоб ви могли застосувати ці концепції до комп'ютерних систем. Хорошим прикладом є домоволодіння. Власник має повні права на доступ до власності (ресурсу), але може надати право доступу іншим людям. Ви кажете, що власник дозволяє людям доступ до нього. Цей простий приклад дозволяє нам запровадити кілька понять у контексті авторизації. Наприклад, доступ до будинку – це дозвіл, тобто дію, яку ви можете виконувати над ресурсом. Іншими дозволами на будинок можуть бути його меблювання, прибирання, ремонт тощо. Дозвіл стає привілеєм (або правом), коли він призначається будь-кому. Отже, якщо ви даєте дозвіл на оздоблення вашого будинку вашому декоратору інтер'єру, ви надаєте йому цей привілей. З іншого боку, декоратор може попросити у вас дозволу обставити вашу оселю. У цьому випадку запитаний дозвіл є областю дії, тобто дією, яку декоратор хотів би виконати у вашому домі. Іноді авторизація дещо пов'язана з ідентифікацією. Подумайте про процес посадки у літак. Ви маєте посадковий талон, в якому зазначено, що ви маєте право літати на цьому літаку. Проте агенту на вході недостатньо пропустити вас на борт. Вам також знадобиться паспорт, що підтверджує вашу особу. У цьому випадку гейт-агент порівнює ім'я в паспорті з ім'ям на посадковому талоні та пропускає вас, якщо вони збігаються.

Підхід до реалізації авторизації реалізовано стандартним шляхом надання унікальної поштової адреси користувачем за допомогою якої буде знайдено обліковий запис та пароль для отримання доступу до облікового запису.

Метод авторизації здійснюється обробкою сесії користувача яка містить дані за якими надається доступ та його обмеження, а також ідентифікатор самого

користувача. Процес обробки даних наведено нижче.

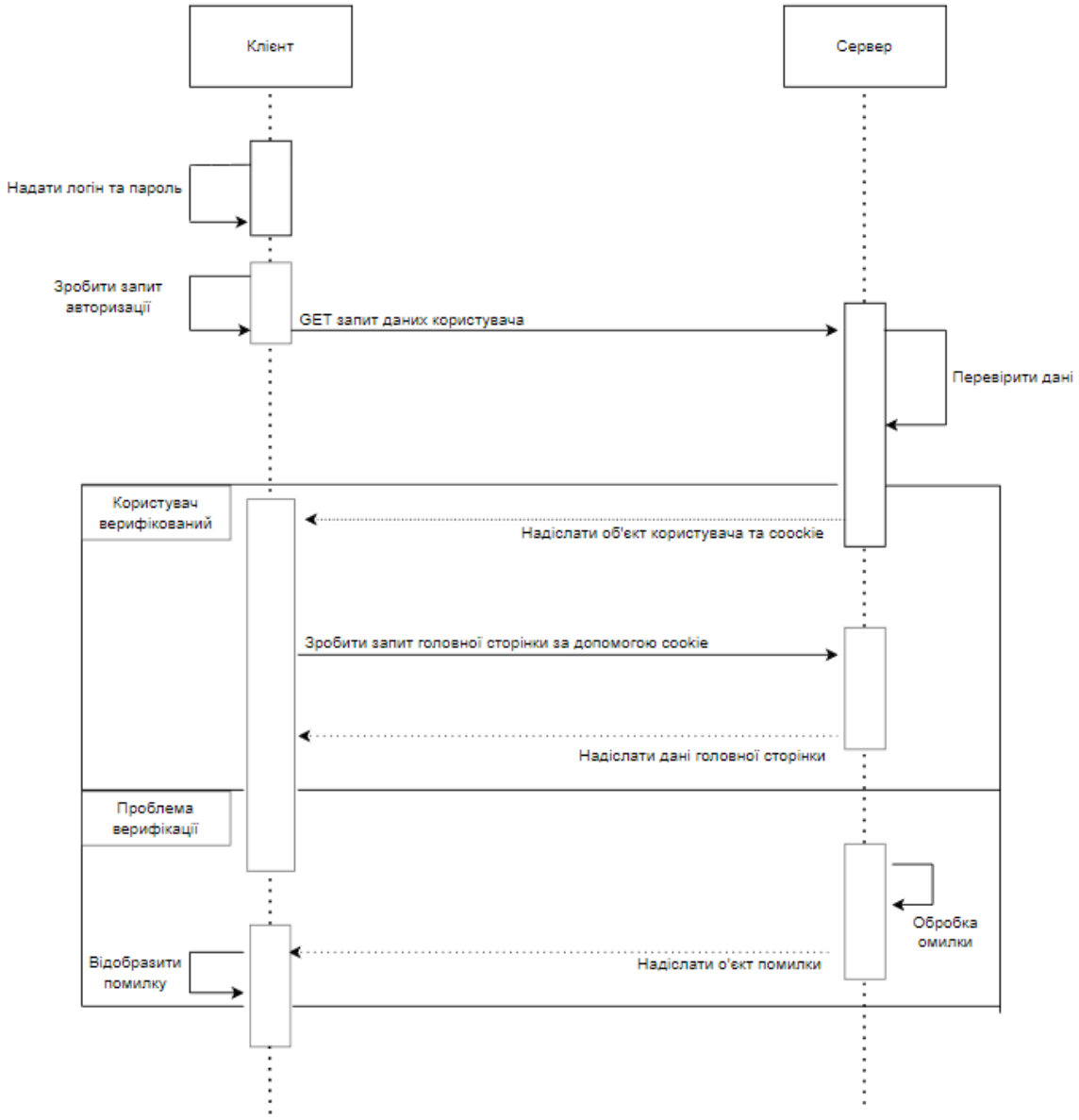

Рис. 3.4. Схема процесу авторизації користувача

Через браузер, користувач, за допомогою форми пошукової платформи надає особисту пошту, за якою пов’язаний запис профілю, та пароль, щоб отримати доступ до профілю. Далі відбувається запит на сервер. Сервер в свою чергу робить запит до бази даних, а саме до колекції бази даних, де зберігається інформація про усіх зареєстрованих користувачів. Їде пошук користувача за отриманою поштовою адресую, у випадку знаходження відповідної адреси, починається перевірка закодованого паролю. Якщо значення паролів збігаються, сервер створює новий запис

активної сесії до відповідної колекції користувача та відправляє назад до клієнта об’єкт сесії, аби той мав можливість перенаправити користувача на головну сторінку та забезпечити його усіма привілеями авторизованого користувача системи.

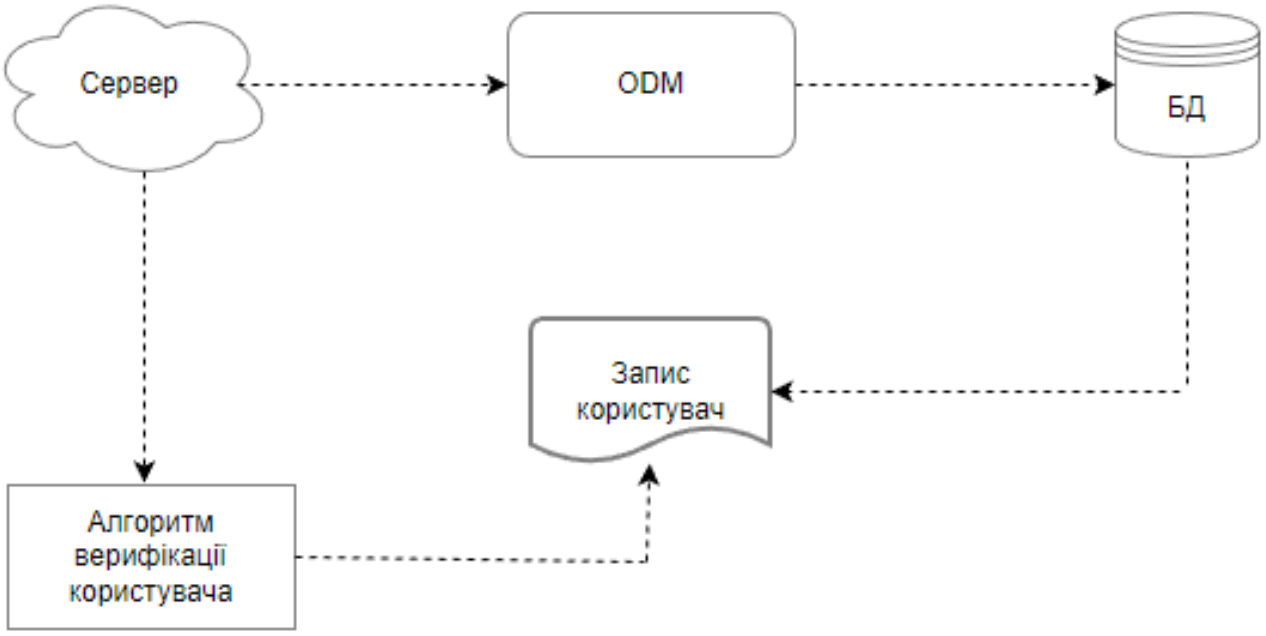

Рис. 3.5. Схема взіємодії сервера з БД

Кожна форма, що пов’язана з авторизацією користувача містить елементи, які надають інформацію щодо виконання процесів: реєстрації, авторизації або ж надіслання та обробку запиту на відновлення доступу до облікового запису. У комп'ютерних системах правила авторизації є частиною IT-дисципліни, яка називається Управління ідентифікацією та доступом (IAM). В рамках IAM авторизація та автентифікація допомагають системним адміністраторам контролювати, хто має доступ до системних ресурсів, та встановлювати привілеї клієнтів. Те, як IT-системи працюють із службами авторизації, дуже схоже на реальний процес управління доступом. снує кілька різних стратегій авторизації, які комп'ютерні системи використовують під час розгортання програм. Найбільш відомими є управління доступом на основі ролей (RBAC) і управління доступом на основі атрибутів (ABAC). Нещодавно Auth0 займався дослідженням та вирішенням проблеми управління

доступом на основі відносин (ReBAC). Існує безліч інших альтернатив, у тому числі керування доступом на основі графа (GBAC) та дискреційне керування доступом (DAC). Кожна з цих стратегій допоможе розробникам додатків упоратися з різними вимогами авторизації та службами авторизації. Також якщо користувач спробує власноруч змінити адресу сторінки з метою перейти на сторінку, яка доступна лише авторизованим користувачам, механізм авторизації відреагує на це.

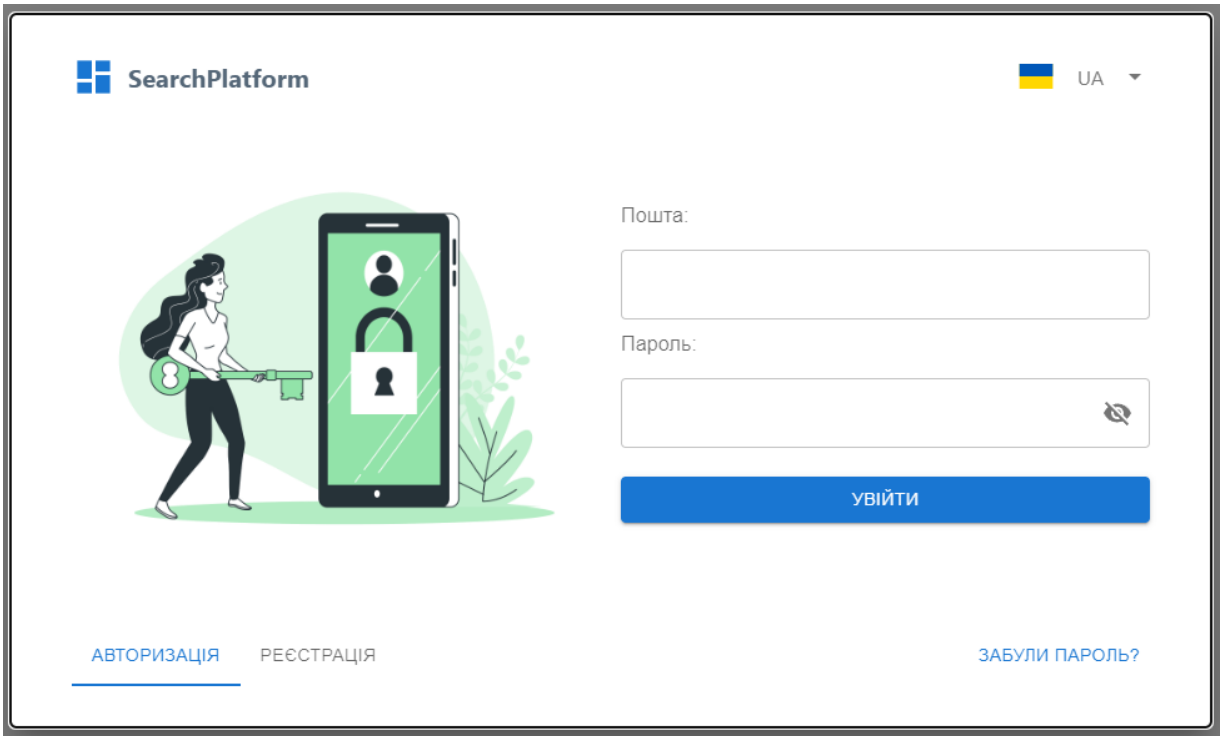

Рис. 3.7. Компонент інтерфейсу «Вхід в систему»

У випадку якщо користувач ще не має особового облікового запису, йому необхідно зареєструватися, шляхом надання власної поштової адреси та створення надійного паролю. Найкращий спосіб створення системи реєстрації залежить від того, як зараз налаштований ваш веб-сайт. Якщо ви використовуєте систему керування контентом, таку як WordPress або аналогічну платформу, ви можете використовувати плагін або розширення, щоб легко налаштувати сторінку реєстрації. Якщо ні, ви можете створити код самостійно або зв'язати свій веб-сайт з іншими популярними сайтами, такими як Facebook або Twitter, використовуючи процес реєстрації.

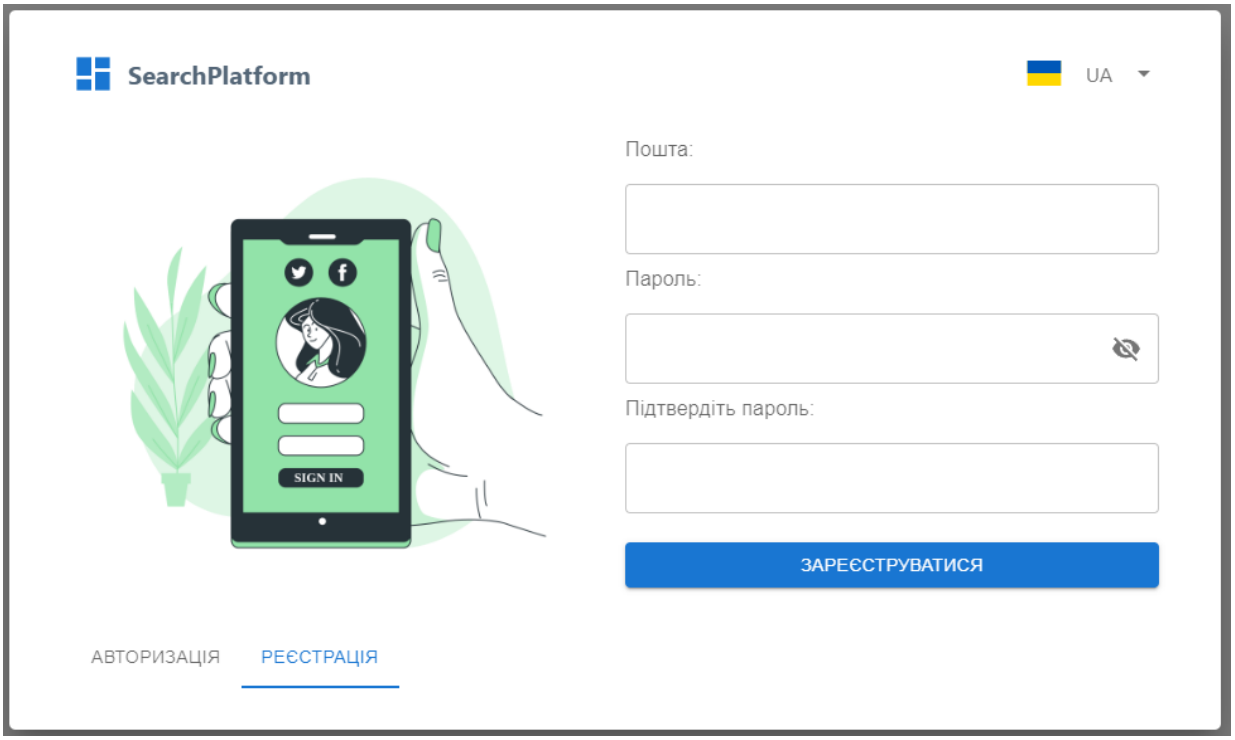

Рис. 3.6. Компонент інтерфейсу «Реєстрація нового користувача»

Так само, як використовувати код для створення реєстраційних форм, майте на увазі, що інші можуть використовувати код для автоматизації запитів на реєстрацію, що призводить до спаму, а іноді і злому вашого веб-сайту. Використання заходів безпеки CAPTCHA, які змушують відвідувачів веб-сайту вручну вводити літери та цифри, що відображаються на зображенні, може унеможливити автоматичні запити на реєстрацію. Ви можете налаштувати систему реєстрації таким чином, щоб відвідувачі реєструвалися автоматично або, як додатковий захід безпеки, запитували реєстрацію, а потім їх запит затверджувався адміністратором веб-сайту.

У випадку коли користувач втратив значення секретного паролю, система надає можливість надіслати запит на його відновлення. Сервер сформує тимчасове посилання з закодованим значенням доступу до профіля. За допомогою поштового сервісу користувач отримає сформоване посилання та детальну інструкцію з виконання наступних кроків по відновленню доступу.

Використавши посилання, користувач буде направлений на окрему сторінку де йому необхідно ввести значення нового паролю аби той не мав збігів з попереднім паролем та підтвердити свій вибір.

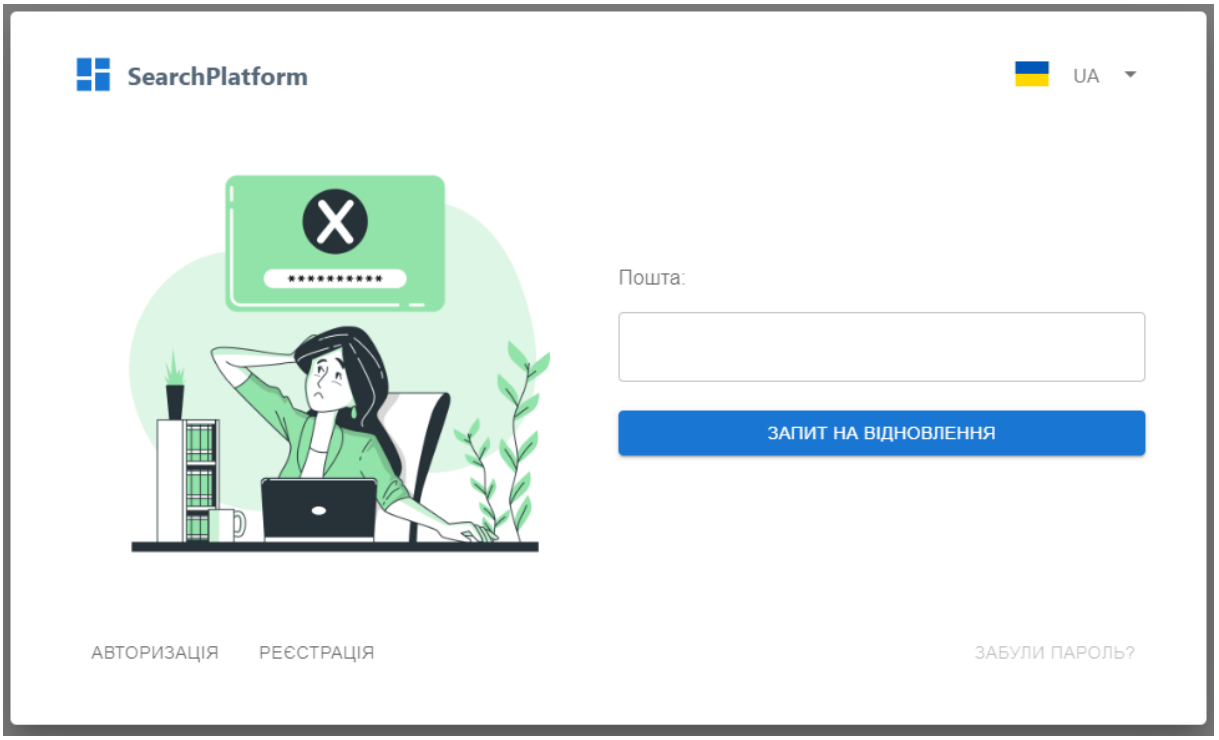

Рис. 3.8. Компонент інтерфейсу «Відновлення доступу до облікового запису»

Сервер в свою чергу проведе необхідну маніпуляцію з записом користувача, а саме оновить поле пароля на отримане нове значення. Після підтвердження паролю, та успішної обробки зі сторони серверу, користувач буде направлений назад на сторінку авторизації, щоб сформувати нову активну сесію та почати нормальний процес використання платформи.

Відновлення пароля - це метод, який використовується для відновлення пароля за допомогою різних рішень для розблокування. Метод відновлення пароля простий, оскільки в більшості випадків дані на пристрої не шифруються. Процес відновлення пароля зазвичай миттєвий і простий у порівнянні зі зломом пароля.

Організація контролю статусу користувача подібним чином є найбільш раціональним та зрозумілим с точки зору кінцевого користувача.

3.3.3. Публікація оголошень

Після успішної авторизації користувач отримує привілеї у використанні основної функції– публікації оголошень.

З метою отримання найбільш якісного результату користувачеві необхідно заповнити поля спеціальної форми, які визначені наступним чином:

- категорію оголошення (загублено/знайдено);
- тип оголошення (людина/майно/тварина);
- локація;
- контактні дані;
- детальна інформація в залежності від типу оголошення;
- дата;
- медіа файли.

Платформа надає функціонал обирати адресу локації потенційного місцезнаходження бажаного об'єкта. Користувач має можливість вказати місце на мапі шляхом інтерактивної взаємодії, ручним введенням адреси до відповідної форми або ж у випадку коли інцидент трапився не за довго як користувач увійшов до облікового запису, система може автоматично визначити місцезнаходження користувача та зекономити цінний ресурс у вигляді часу. Більш того, користувач має можливість вказати радіус області де відбувся інцидент і тим самим звузити варіанти локацій для обшуку.

Метод запиту призначений для направлення запиту, при якому веб-сервер приймає дані, що містяться в тіло повідомлення, для зберігання. Він часто використовується для завантаження файлу або представлення заповненої веб-форми.
Надання детальної інформації стосовно місцезнаходження об’єкта забезпечить покращення якості та ефективності пошуку самого об’єкту в майбутньому. Саме тому цій частині функціоналу приділена особлива увага під час розробки та інтеграції аби забезпечити максимальний рівень точності виконання та поліпшити взаємодію з сервісами платформи.

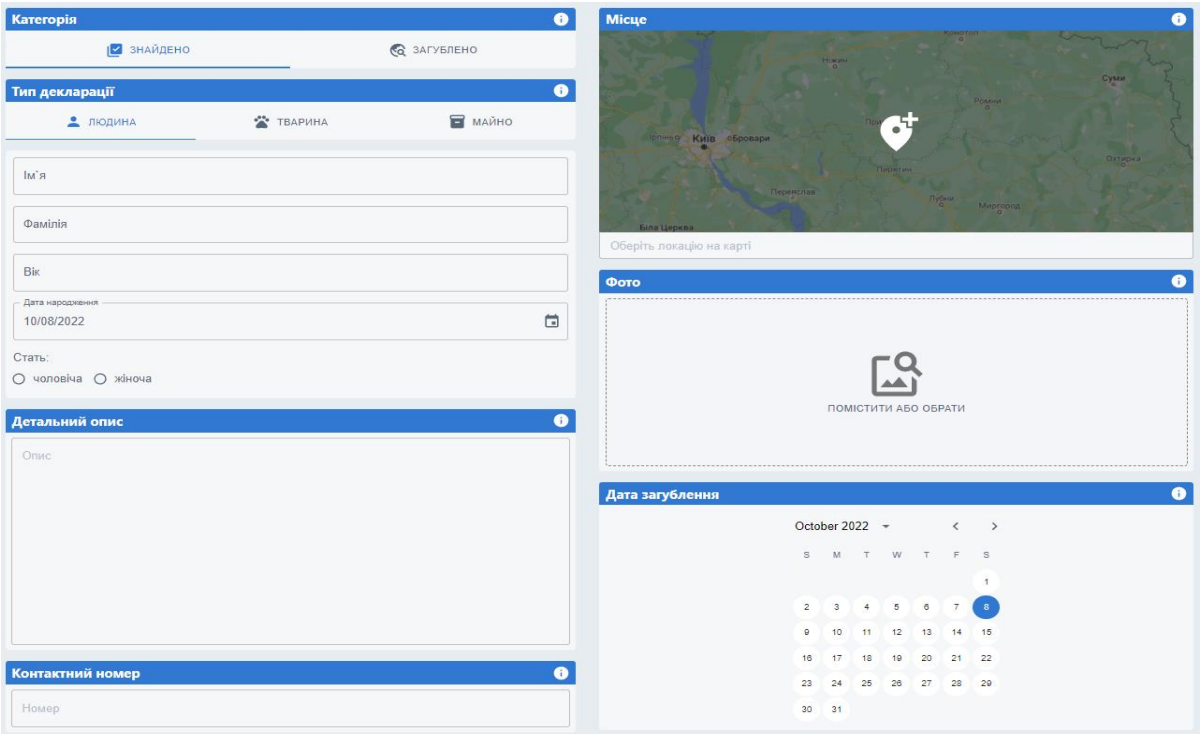

Рис. 3.9. Компонент інтерфейсу «Створення декларації»

Коли веб-браузер надсилає запит з елементами веб-форми, за замовчуванням інтернет-тип даних медіа - application/x-www-form-urlencoded. Це формат для кодування пар ключ-значення з можливістю дублювання ключів. Кожна пара ключ-значення відокремлюється символом & ключ відокремлений від значення символом =. У ключах і значеннях пробіли замінюються на знак +, а потім, використовуючи URL-кодування, замінюються всі буквено-цифрові символи. метод запиту має бути використаний для будь-якого контексту, в якому запит не ідемопотентний: тобто він викликає зміну стану сервера щоразу при виконанні, такі як відправлення коментаря до повідомлення у блозі або інтернет-голосування. На практиці, метод GET часто зарезервований не просто для ідемпотентних дій, але і для нульпотентних, тобто без побічних ефектів (на відміну від «без побічних ефектів при другому та наступних запитах» як з ідемпотентними операціями). З цієї причини сайти пошукових систем, таких як індексатори пошукових систем, зазвичай використовують виключно метод GET, для запобігання будь-яким діям при автоматизованих запитах.

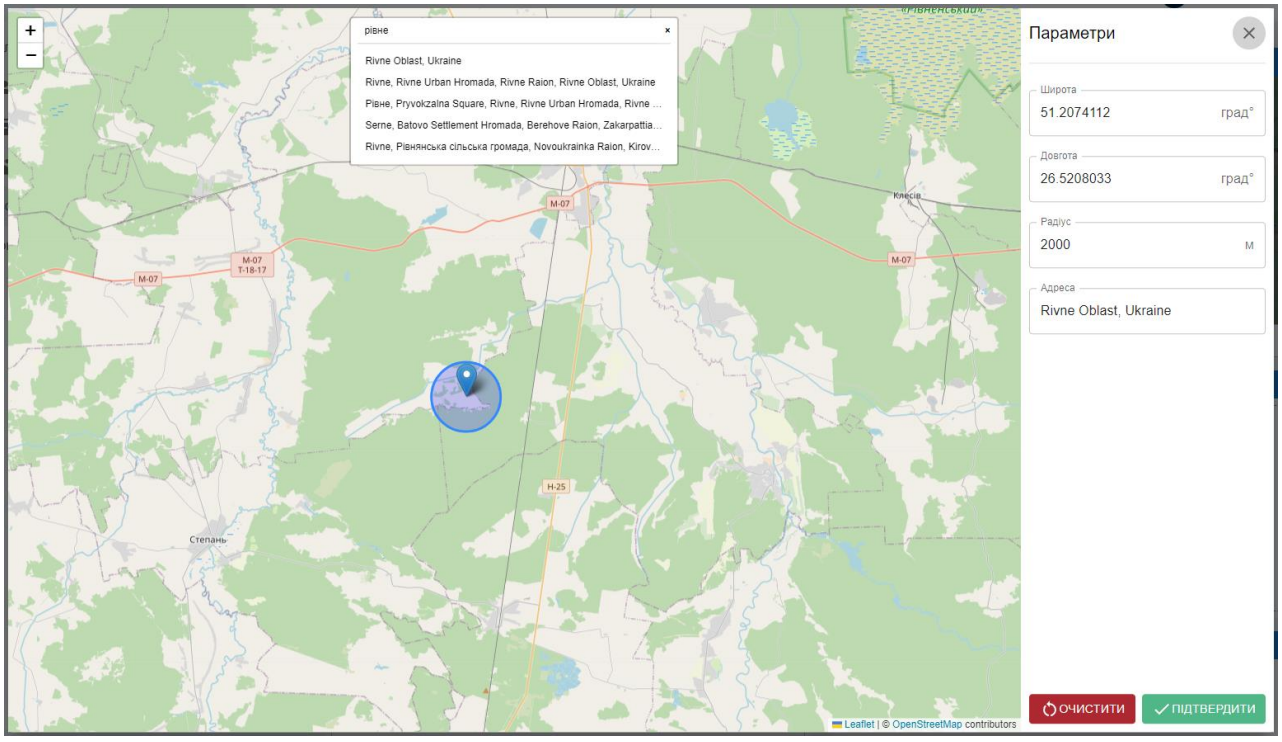

Рис. 3.9. Компонент інтерфейсу «Визначення місцезнаходження»

Також у довгостроковій перспективі платформа має можливість легкого інтегрування інших сервісів обробки місцезнаходження, які матимуть покращенні алгоритми отримання геоданих користувача та розширену базу даних локацій. Це забезпечить можливість розширити область застосування програмного продукту щоб допомогти ще більшій кількості людей.

3.3.4. Перегляд загального переліку публікацій

Користувач має можливість безпосередньо переглянути усі наявні публікації, що були створені в межах платформи та за допомогою визначення критеріїв зробити власну вибірку та провести огляд необхідних оголошень, які можливо йому необхідні.

Публікації поділені на три основні категорії: людина, майно та тварини. Категоризація необхідна як найбільший критерій для формування вибірки пошуку та зменшення кількості об’єктів в вибірці за для прискорення дії пошукового методу аналізу шляхом зменшення кількості ітерацій алгоритму та загальної кількості вхідних

даних, які необхідно обробити та на їх основі сформувати відповідну статистику щодо їх схожості та подібності.

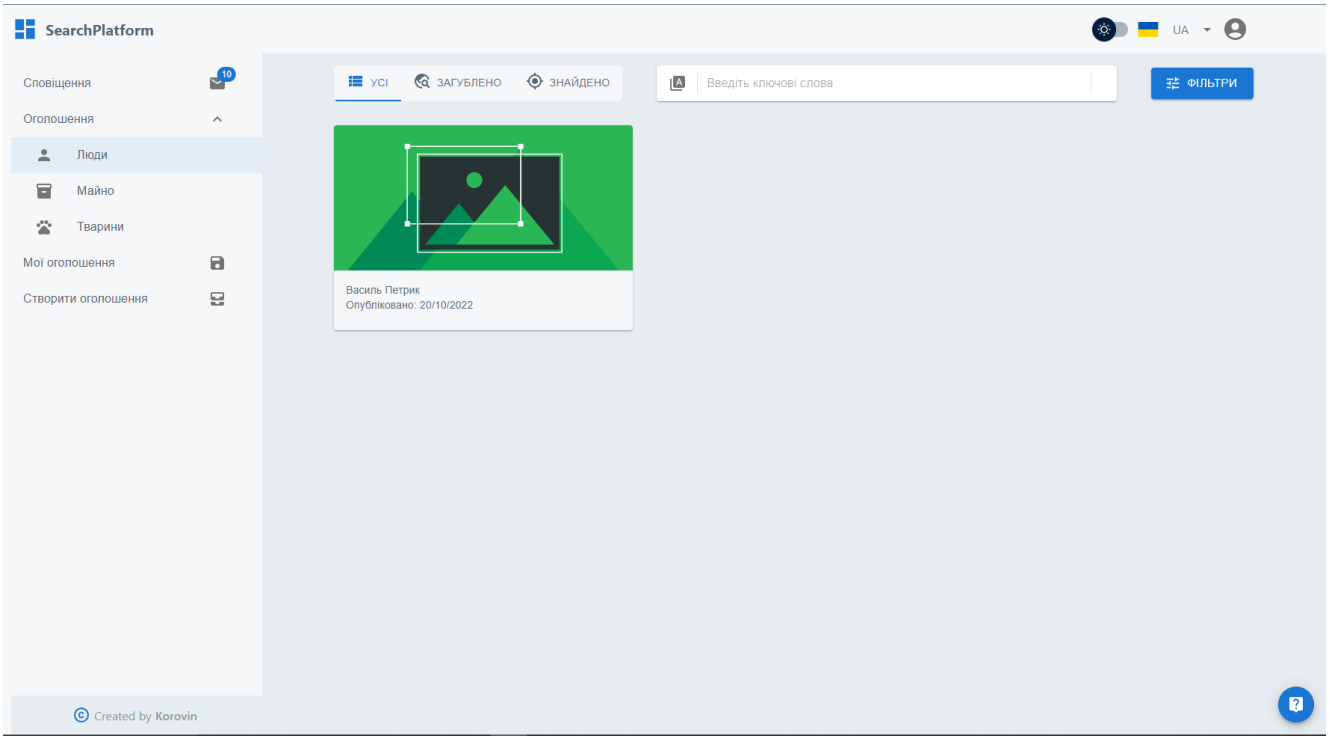

Рис. 3.10. Компонент інтерфейсу «Перелік публікацій»

Використання фільтрів та механізмів пошуку є загальною практикою у системах, які оперують великою кількістю даних, що сприйнятливо діє на досвід використання та загальне ставлення користувача у цілому до обраної системи, щоб задовольнити власні потреби. Забезпечення комфортного перегляду великої кількості інформації для користувача є одним із найбільших пріорітетів для розробників програмного забезпечення .

3.3.5. Перегляд деталей окремих публікацій

Кожна публікація, в залежності від категорії, яка була обрана під час її створення має специфічні поля, визначені користувачем, які її характеризують і ці поля не відображаються на сторінці перегляду загального списку публікацій.

За для економії простору на сторінці та зменшення навантаження на відображення великої кількості даних, сторінка перегляду загального переліку публікацій надає лише основну інформацію, що стосується оголошення. Цей компонент несе лише інформативний характер для покращення взаємодії з іншими компонентами платформи.

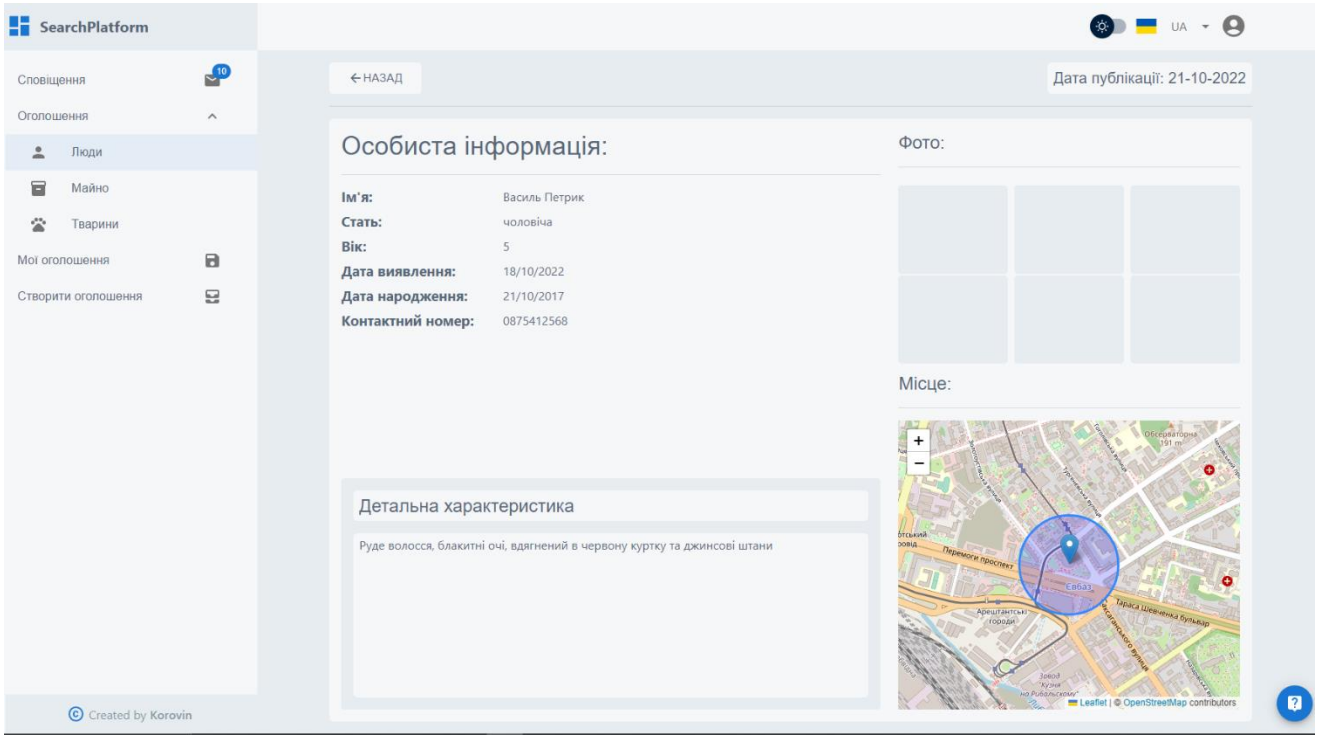

Рис. 3.11. Компонент інтерфейсу «Деталі окремої публікації»

З метою отримати більш детальну інформацію, як наприклад: номер контактної особи, яка створила це оголошення, або ж провести особистий аналіз на основі даних місцезнаходження так прикріплених фото медіа файлів до оголошення, з метою покращення візуалізації та як основа для подальшого розвитку платформи, а саме реалізація порівняльної характеристики на основі фотографій, платформа надає можливість окремо переглянути кожне оголошення.

3.3.6. Перегляд власних публікацій

Під час створення публікації користувачем їй надається унікальний ідентифікатор цього користувача та сама публікація додається до загального переліку платформи, а також до особистої колекції користувача. Метою такого розподілення є покращення менеджменту створених публікації користувачем.

До менеджменту публікації відносяться наступні маніпуляції, а саме: редагування даних, видалення неактивних публікацій.

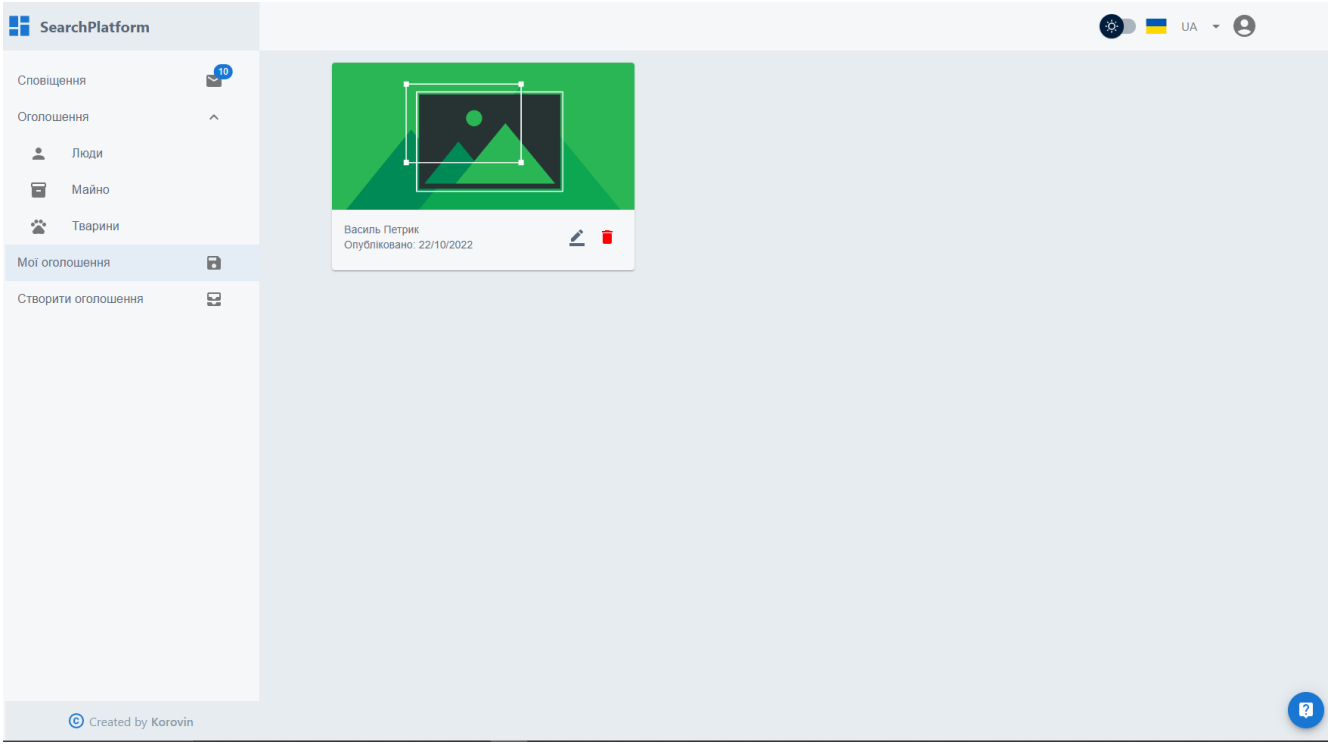

Рис. 3.12. Компонент інтерфейсу «Перегляд власних публікацій»

Оскільки кожна людина схильна до того, щоб робити помилки, функція редагування є край необхідною аби уникнути створення недостовірної інформації та погіршення процесу аналізу на фоні інших публікацій. Також, якщо помилка була допущена лише в одному місці, то немає необхідності видаляти усе оголошення повністю, замість цього надається можливість виправити саме ту частину, яка цього потребує. Мінімізація впливу помилок користувача на загальний досвід використання програм-

ного продукту є ключовою метою кожного розробника за для підтримки репутації компанії, яка організовує необхідні заходи для обслуговування програмного продукту з метою покращення сервісу.

3.3.7. Механізм сповіщення користувача

Під час створення нового оголошення, попри 2 основні функції, які були описані раніше: додання оголошення до загального переліку та переліку особистих оголошень користувача, є також третя не менш важлива функція, яка забезпечить економію часу на розшук, а саме - механізм сповіщення.

Коли користувач створює нове оголошення, сервер автоматично запускає алгоритм перевірки на схожість з уже існуючими оголошеннями і у випадку потенційного співпадіння, користувачеві буде надіслане повідомлення разом з об’єктом схожого оголошення для подальшої безпосередньої перевірки користувачем.

Висновки до розділу 3

Чітко визначена проблематика та структура пошукової платформи оголошень розшуку. Сформульована головна мета функціонування програмного продукту та визначені компоненти інтерфейсу для подальшої інтеграції з методом обробки текстових даних.

Описаний кожен компонент інтерфейсу та функціонал, що він реалізовує. Обрано найбільш раціональний та ефективний підхід розробки, що забезпечить комфортне та зручне користування платформою для будь-якого користувача. Приділена увага безпеці даних, що надає користувач системи, задля підвищення рівню довіри серед потенційних майбутніх користувачів. Реалізована оптимальна взаємодія сервісів платформи між собою з мірами запобігання затримок між роботою інтерфейсів серверу та клієнта.

Для реалізації кожного компоненту системи використовуються сучасні програмні засоби для розробки інтерфейсів користувача, щоб задовольнити усіх учасників, що взаємодіють з системою, а саме: розробники, менеджери та користувачі. Програмні засоби забезпечують ефективними інструментами розробки, що покращують, пришвидшують та збільшують якість коду програмного коду, що є ключовим фактором продуктивного робочого оточення та мінімізує непорозуміння серед розробників. Користувачі в свою чергу отримують продукт, який швидко реагує на їх потреби та ефективно вирішує поставлені проблеми. Менеджери мають можливість легко відстежувати усі процеси, що відбуваються в межах інформаційної системи, корегувати та покращувати їх результати.

РОЗДІЛ 4. МЕТОД АНАЛІТИЧНОЇ ОБРОБКИ ТЕКСТОВИХ ДАНИХ

Кожна інформаційна система має в своїй основі відповідний алгоритм або метод, який націлений на вирішення проблеми відповідно до тематики системи. Платформа розміщення оголошень не є винятком. В основі закладений метод з порівняння та аналізу текстових даних, який вирішує основну частину проблеми за для якої була створена платформа.

На основі виконаного аналізу та висновків з першого розділу роботи. Метод обробки текстових даних буде розроблений на основі підходу Сьоренсена-Дайса. Обраний підхід є найбільш оптимальним для вирішення задачі у відповідності до потреб платформи.

Сам по собі метод - математична формула, яка описує коефіцієнт подібності декількох множин. Щоб розробити метод, який в подальшому буде інтегрований до пошукової платформи необхідно розробити комплексний набір функції, щоб отримати максимальну точність результатів та уникнути зайвих обчислень. Набір складається з наступних функцій: валідація вхідних даних, отримання набору даних, фільтрація множини даних за визначеними критеріями окремих елементів набору даних, формування унікальної вибірки, формування статистичної інформації та підготовка її для подальшого використання в компонентах системи.

4.1. Валідація вхідних даних

Перевірка вхідних даних виконується для того, щоб гарантувати, що тільки правильно сформовані дані надходять у робочий процес в інформаційній системі, запобігаючи збереженню спотворених даних у базі даних та викликаючи збої в роботі різних компонентів нижче. Перевірка вхідних даних повинна відбуватися

якомога раніше в потоці даних, переважно відразу після отримання даних від зовнішньої сторони.

Цілісність даних стає все більш важливою, оскільки все більше фірм B2B використовують методи, що базуються на даних, для збільшення доходів і підвищення операційної ефективності. Нездатність довіряти бізнес-даним, зібраним із різних джерел, може саботувати зусилля організації з досягнення найважливіших бізнес-цілей. Величезний обсяг даних може бути приголомшливим для бізнесу. Стандарти даних, різнорідні системи даних, відсутність управління даними, ручні процеси тощо - це проблеми, з якими вони стикаються.

Підприємства отримують дані про своїх клієнтів за допомогою внутрішніх процесів, а також зовнішніх взаємодій, включаючи демографічну, технографічну, фірмографічну та фінансову інформацію. Однак зібрана інформація часто необроблена і рясніє помилками, що ускладнює висновки. Внаслідок цієї неможливості довіряти даним потрібна перевірка даних. Перевірка даних дозволяє підприємствам бути впевненішими у своїх даних.

Перевірка даних відноситься до процесу забезпечення точності та якості даних. Це реалізується шляхом вбудовування кількох перевірок у систему або звіт для забезпечення логічної узгодженості даних, що вводяться і зберігаються.

Перевірка введення повинна застосовуватись як на синтаксичному, так і на семантичному рівні .Синтаксична перевірка має забезпечувати правильний синтаксис структурованих полів (наприклад, дата, символ валюти). Семантична перевірка повинна забезпечувати правильність їх значень у конкретному бізнес-контексті (наприклад, дата початку передує даті закінчення, ціна знаходиться в очікуваному діапазоні).

Перевірка введення може бути реалізована з використанням будь-якого методу програмування, який дозволяє ефективно забезпечувати синтаксичну та семантичну правильність, наприклад:

- валідатори типів даних спочатку доступні у фреймворках веб-додатків (наприклад, Django Validators, Apache Commons Validators тощо);

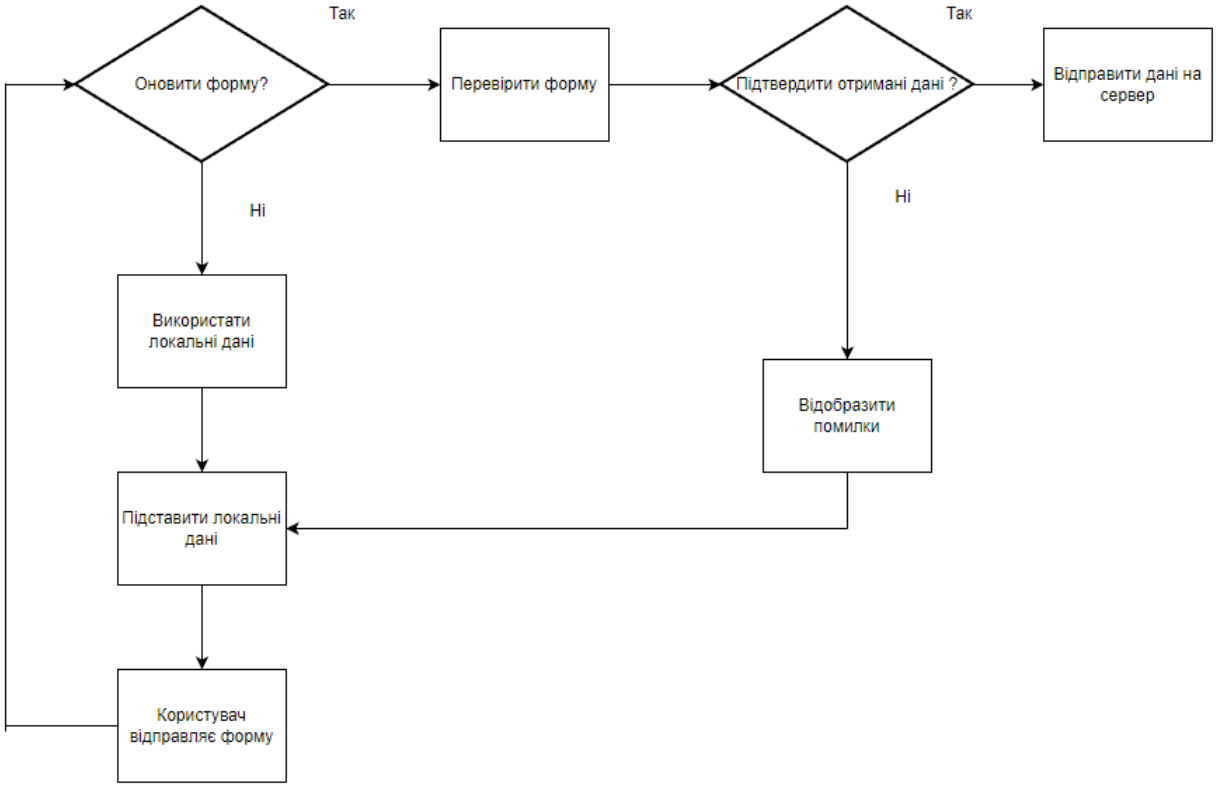

Рис. 4.1. Діаграма процесу перевірки даних на стороні клієнта

- перевірка за схемою JSON та схемою XML (XSD) для введення в цих форматах;
- перетворення типів (наприклад, Integer.parseInt() в Java, int() в Python) зі строгою обробкою винятків;
- перевірка мінімального та максимального діапазону значень для числових параметрів та дат, перевірка мінімальної та максимальної довжини для рядків;
- масив допустимих значень для невеликих наборів рядкових параметрів (наприклад, дні тижня);
- регулярні вирази для будь-яких інших структурованих даних, що охоплюють весь вхідний рядок (^...$) і не використовують підстановочний знак «будь-який символ» (наприклад, . або \S).

Основні вимоги до формату та типу даних з якими оперує метод порівняльного аналізу пошукової платформи розміщення оголошень розшуку виглядають наступним чином:

- відсутність специфічних символів, які не мають семантичного змісту;

- відповідність до мінімальної довжини набору текстових символів;
- перевірка типу даних, метод оперує лише з двома вхідними змінними, які мають визначені типи: текстовий рядок, як об’єкт порівняння та маси рядків, як цільові дані для порівняння;
- перевірка унікальності цільових даних для порівняння, аби уникнути зайві розрахунки та дублікати однакових результатів.

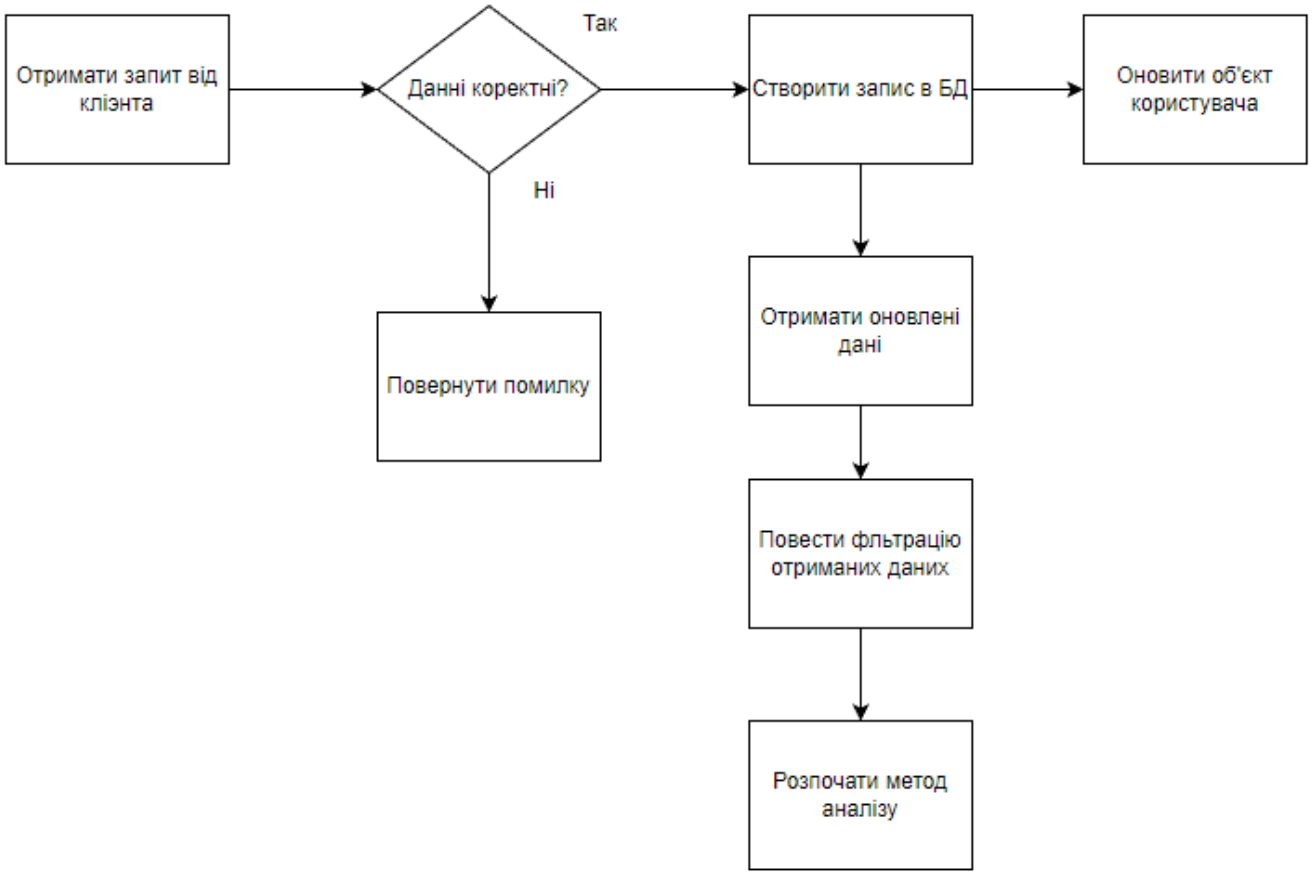

Рис. 4.2. Діаграма процесу перевірки даних на стороні сервера

Дані для методу отримуються через форму, яка наявна на стороні клієнта, тобто користувача платформи, саме тому початкові перевірки виконуються ще до відправлення даних на сервер. Перевірки типу: пусті форми, форми з недостатньо кількістю символів, форми, які містять лише відступи та форми, які мають повторюваність символів більше ніж два символи поспіль.

Якщо введені користувачем дані відповідають описаним вище критеріям, йому

дозволено відіслати запит на сервер. Сервер в свою чергу робить відповідні перевірки, аби покращити якість оцінки.

4.2. Отримання набору даних

Вибір даних визначається як процес визначення відповідного типу даних та джерела, а також відповідних інструментів для збирання даних. Вибір даних передує фактичної практики збору даних. Це визначення відрізняє вибір даних від вибіркового представлення (вибірковий виняток даних, що не підтримують дослідницьку гіпотезу) та інтерактивного вибору даних (використання зібраних даних для моніторингу подій або проведення аналізу вторинних даних). Процес вибору потрібних даних для дослідницького проекту може вплинути на цілісність даних.

Основною метою відбору даних є визначення відповідного типу даних, джерела та інструменту, які дозволяють дослідникам адекватно відповідати на питання дослідження. Проблеми з цілісністю можуть виникнути, коли рішення про вибір «відповідних» даних для збору засновані насамперед на міркуваннях вартості та зручності, а не на здатності даних адекватно відповідати на питання дослідження. Звичайно, вартість та зручність є важливими факторами у процесі прийняття рішень. Однак дослідники повинні оцінити, як ці фактори можуть поставити під загрозу цілісність дослідницької діяльності.

Увага до процесу відбору даних має вирішальне значення для підтримки наступних етапів дослідження. Незважаючи на зусилля щодо суворого дотримання протоколів збору даних, вибору відповідного статистичного аналізу, точної звітності даних та неупередженого опису, наукові результати матимуть сумнівну цінність, якщо процес відбору даних буде помилковим.

Після того, як оголошення було створено, сервер робить перевірку нового оголошення, чи схоже воно на інші оголошення які є в системі, аби одразу сповістити користувача про знахідку. Для цього виконується наступна процедура: створюється нове оголошення в базі даних, отримується новий перелік оголошень, з переліку

формується вибірка за визначеними критеріями, сформована вибірка передається у якості аргументу до функції метода порівняння текстових даних та починається виконання останнього етапу перетворення даних, зведення до одного вигляду.

Вибір даних визначається як процес визначення відповідного типу даних та джерела, а також відповідних інструментів для збирання даних. Вибір даних передує фактичної практики збору даних. Це визначення відрізняє вибір даних від вибіркового представлення даних (за винятком даних, які не підтримують дослідницьку гіпотезу) та інтерактивного/активного вибору даних (використання зібраних даних для моніторингу дій/подій або проведення аналізу вторинних даних). Процес вибору потрібних даних для дослідницького проекту може вплинути на цілісність даних. Основною метою відбору даних є визначення відповідного типу даних, джерела та інструменту, які дозволяють дослідникам адекватно відповідати на питання дослідження. Це визначення часто залежить від дисципліни і в першу чергу визначається характером дослідження, що існує літературою та доступністю до необхідних джерел даних. Проблеми з цілісністю можуть виникнути, коли рішення про вибір «відповідних» даних для збору засновані насамперед на міркуваннях вартості та зручності, а не на здатності даних адекватно відповідати на питання дослідження. Звичайно, вартість та зручність є важливими факторами у процесі прийняття рішень. Однак дослідники повинні оцінити, як ці фактори можуть поставити під загрозу цілісність дослідницької діяльності. Є деякі питання, про які слід пам'ятати дослідникам при відборі даних, наприклад:

- Відповідний тип та джерела даних дозволяють дослідникам адекватно відповісти на поставлені дослідницькі питання.
- Відповідні процедури для отримання репрезентативної вибірки.
- Належні інструменти для збору даних. Нелегко відокремити вибір типу/джерела даних від інструментів, що використовуються для збирання даних. Повинна бути сумісність між типом/джерелом даних та механізмами їх збору.

Вибір ознак був активною областю досліджень у спільнотах розпізнавання образів, статистики та інтелектуального аналізу даних. Основна ідея вибору ознак полягає в

тому, щоб вибрати підмножину вхідних змінних, виключивши ознаки з невеликою прогностичною інформацією або без неї. Вибір ознак може значно покращити зрозумілість результуючих моделей класифікатора і часто будувати модель, яка узагальнює невидимі точки. Крім того, часто буває так, що пошук правильної підмножини прогностичних ознак сам по собі є важливою проблемою. Наприклад, лікар може вирішити на основі вибраних ознак, чи необхідна небезпечна операція для лікування чи ні. Вибір ознак у навчанні з учителем добре вивчений, основна мета якого - знайти підмножина ознак, що забезпечує більш високу точність класифікації.

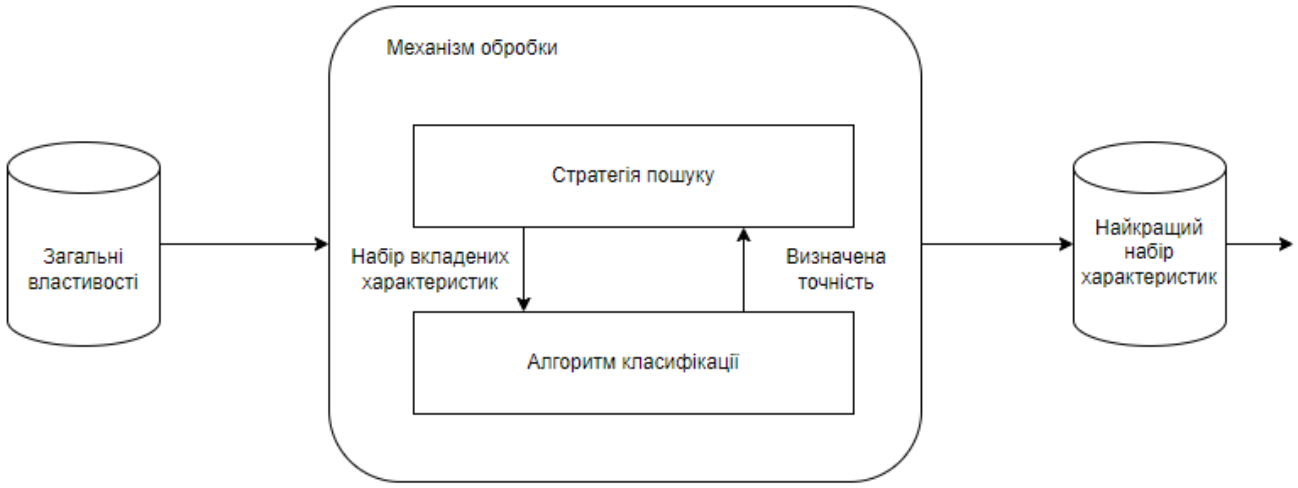

Рис. 4.3. Процес відбору даних

Останнім часом кілька дослідників вивчали вибір ознак та кластеризацію разом із єдиним чи уніфікованим критерієм. Для вибору ознак у неконтрольованому навчанні алгоритми навчання призначені для пошуку природного угруповання прикладів у просторі ознак. Таким чином, вибір ознак у неконтрольованому навчанні спрямований на пошук хорошого підмножини ознак, яке формує кластери високої якості для заданої кількості кластерів. Проте традиційні підходи до вибору ознак із використанням одного критерію оцінки показали обмежені можливості з погляду

виявлення знань та підтримки прийняття рішень. Це пов'язано з тим, що особи, які приймають рішення, повинні одночасно враховувати кілька цілей, що суперечать одна одній. Зокрема, жоден критерій неконтрольованого вибору функцій не є найкращим для кожної програми, і тільки особа, яка приймає рішення, може визначити відносні ваги критеріїв для своєї програми.

4.3. Формування аналітичної інформації

Інформаційна аналітика - це термін, який використовується для опису збору та аналізу даних. Використання потужних комп'ютерів для виявлення закономірностей та тенденцій у наборі даних не нове, але за останні кілька років воно значно розширилося. В результаті якість вихідних даних досить висока, щоб їх можна було використовувати як інструмент у процесі прийняття рішень. Найчастіше інформаційна аналітика використовується для вивчення бізнес-даних за допомогою комбінації методів статистичного аналізу та інтелектуального аналізу даних. Метою цього аналізу є виявлення тенденцій, які можуть призвести до дії. Статистики часто виявляють цікаві тенденції чи закономірності у поведінці. Однак у бізнес-середовищі організація повинна знати про шаблони, які можна використовувати для отримання доходів або зменшення втрат.

Сучасному бізнесу потрібні всі можливості та переваги, які вони можуть отримати. Завдяки таким перешкодам, як ринки, що швидко змінюються, економічна невизначеність, зміна політичного ландшафту, вибагливі споживчі настрої та навіть глобальні пандемії, сьогодні компанії працюють із меншою можливістю помилок. Компанії, які хочуть не тільки залишатися в бізнесі, але й процвітати, можуть покращити свої шанси на успіх, зробивши розумний вибір і відповідаючи на запитання: «Що таке аналіз даних?» І як особа чи організація робить цей вибір? Вони роблять це, збираючи якомога більше корисної, дієвої інформації, а потім використовуючи її для прийняття більш обґрунтованих рішень! Ця стратегія є здоровим глуздом, і вона стосується як особистого життя, так і бізнесу. Ніхто не

приймає важливих рішень, попередньо не з’ясувавши, що поставлено на карту, плюси і мінуси та можливі результати. Так само жодна компанія, яка хоче досягти успіху, не повинна приймати рішення на основі поганих даних. Організаціям потрібна інформація; їм потрібні дані. Ось тут аналіз даних вступає в картину. Тепер, перш ніж детально розбиратися в методах аналізу даних, давайте спочатку зрозуміємо, що таке аналіз даних.

Кожна компанія, зацікавлена в інформаційній аналітиці, має інвестувати як у персонал, так і у технології. Інформаційна аналітика зазвичай організується як частина бізнес-аналітики у відділі інформаційних технологій. Співробітники, які працюють з аналітикою, зазвичай мають вищу освіту в галузі статистики, вищої математики, інформаційних технологій чи програмування. Нерідко всі співробітники мають ступінь магістра та доктора наук у будь-якій із цих областей. Рівень навичок та знань, необхідних для ефективного використання цих інструментів, розробки запитів та аналізу результатів досить високий. З технологічного погляду виконання необхідного складного аналізу потрібен інструмент бізнес-аналітики чи управління статистичними даними. На додаток до спеціалізованого програмного забезпечення, необхідні значні інвестиції в обладнання або засоби зв'язку. В ідеальному сценарії бізнес-дані із системи планування ресурсів підприємства (ERP) доступні за допомогою інструмента аналізу. Звичайний необов'язковий метод полягає в дублюванні відповідних даних в окреме сховище даних, яке використовується виключно для звітів, інтелектуального аналізу або аналізу даних. Незважаючи на те, що багато груп, організацій і експертів мають різні способи підходу до аналізу даних, більшість із них можна сформулювати як універсальне визначення. Аналіз даних - це процес очищення, зміни й обробки необроблених даних, а також вилучення корисної й релевантної інформації, яка допомагає компаніям приймати зважені рішення. Процедура допомагає зменшити ризики, пов’язані з прийняттям рішень, надаючи корисну інформацію та статистичні дані, часто представлені у вигляді діаграм, зображень, таблиць і графіків. Простий приклад аналізу даних можна побачити щоразу, коли ми приймаємо рішення в нашому повсякденному житті, оцінюючи те, що сталося в минулому або що станеться, якщо ми приймемо таке рішення.

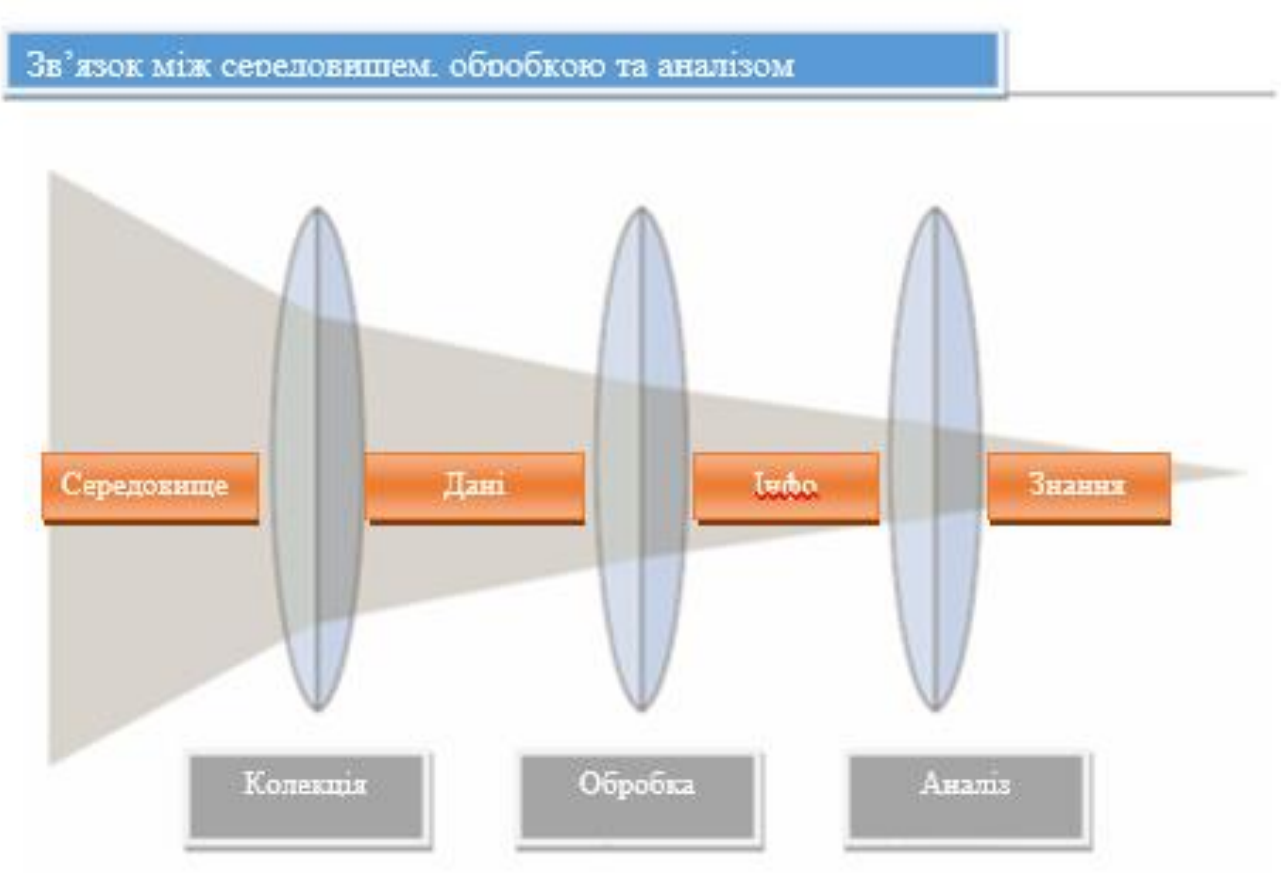

Рис. 4.4. Процес формування аналізу

По суті, це процес аналізу минулого чи майбутнього та прийняття рішення на основі цього аналізу. Ось список причин, чому сьогодні аналіз даних є такою важливою частиною ведення бізнесу:

- краще націлювання на клієнтів: Ви не хочете витрачати дорогоцінний час, ресурси та гроші свого бізнесу на створення рекламних кампаній, націлених на демографічні групи, які практично не зацікавлені в товарах і послугах, які ви пропонуєте. Аналіз даних допомагає зрозуміти, на чому слід зосередити свої рекламні зусилля;
- краще розуміння своїх цільових клієнтів: аналіз даних відстежує ефективність ваших продуктів і кампаній у вашій цільовій демографічній групі. Завдяки аналізу даних ваш бізнес може отримати краще уявлення про споживацькі звички вашої цільової аудиторії, наявний дохід і найімовірніші сфери інтересів. Ці дані допомагають підприємствам встановлювати ціни, визначати тривалість рекламних кампаній і навіть допомагати прогнозувати необхідну

кількість товарів;

- економія на операційні витрати. Аналіз даних показує, які сфери вашого бізнесу потребують більше ресурсів і грошей, а які не приносять продуктивності, тому їх слід скоротити або повністю ліквідувати;
- кращі методи вирішення проблем: обґрунтовані рішення, швидше за все, будуть успішними. Дані надають підприємствам інформацію. Ви можете побачити, куди веде цей прогрес. Аналіз даних допомагає компаніям зробити правильний вибір і уникнути дорогих пасток;
- отримання більш точних даних: якщо ви хочете приймати обґрунтовані рішення, вам потрібні дані, але це ще щось. Дані, про які йдеться, мають бути точними. Аналіз даних допомагає компаніям отримувати релевантну точну інформацію, придатну для розробки майбутніх маркетингових стратегій, бізнес-планів і перебудови бачення чи місії компанії.

Відповідь на запитання «що таке аналіз даних» — це лише перший крок. Зараз ми розглянемо, як це виконується. Процес аналізу даних або етапи аналізу даних включають збір усієї інформації, її обробку, дослідження даних і використання для пошуку закономірностей та інших відомостей. Процес складається з:

- збір вимог до даних: запитайте себе, чому ви проводите цей аналіз, який тип аналізу даних ви хочете використовувати та які дані ви плануєте аналізувати;
- збір даних: керуючись вимогами, які ви визначили, настав час зібрати дані з ваших джерел. Джерела включають тематичні дослідження, опитування, інтерв'ю, анкети, пряме спостереження та фокус-групи. Обов'язково впорядкуйте зібрані дані для аналізу;
- очищення даних: не всі дані, які ви збираєте, будуть корисними, тому настав час їх очистити. У цьому процесі ви видаляєте пробіли, дублікати записів і основні помилки. Очищення даних є обов'язковим перед відправкою інформації на аналіз;
- аналіз даних: тут ви використовуєте програмне забезпечення для аналізу даних та інші інструменти, які допоможуть вам інтерпретувати та зрозуміти

дані та робити висновки. До інструментів аналізу даних належать Excel, Python, R, Looker, Rapid Miner, Chartio, Metabase, Redash і Microsoft Power BI;

- інтерпретація даних: тепер, коли у вас є результати, вам потрібно їх інтерпретувати та придумати найкращі варіанти дій на основі ваших висновків;
- візуалізація даних. Візуалізація даних — це вигадливий спосіб сказати: «графічно покажіть вашу інформацію так, щоб люди могли її прочитати та зрозуміти». Ви можете використовувати діаграми, графіки, карти, маркери або безліч інших методів. Візуалізація допомагає отримати цінну інформацію, допомагаючи вам порівнювати набори даних і спостерігати за зв'язками.

Величезна частина роботи дослідника полягає в аналізі даних. Це буквально визначення «дослідження». Однак сьогоднішня інформаційна ера регулярно створює приливну хвилю даних, достатню для того, щоб переповнити навіть найвідданішого дослідника. Таким чином, аналіз даних відіграє ключову роль у перетворенні цієї інформації в більш точну та актуальну форму, що полегшує дослідникам виконання їхньої роботи. Аналіз даних також надає дослідникам широкий вибір різних інструментів, таких як описова статистика, інференційний аналіз і кількісний аналіз. Отже, підводячи підсумок, аналіз даних пропонує дослідникам кращі дані та кращі способи їх аналізу та вивчення.

Сьогодні існує півдюжини популярних типів аналізу даних, які зазвичай використовуються у світі технологій і бізнесу. Вони є:

- діагностичний аналіз. Діагностичний аналіз дає відповідь на запитання «Чому це сталося?» Використовуючи дані, отримані в результаті статистичного аналізу (докладніше про це пізніше!), аналітики використовують діагностичний аналіз для виявлення закономірностей у даних. В ідеалі аналітики знаходять схожі закономірності, які існували в минулому, і, отже, використовують ці рішення, щоб, сподіваємося, вирішити поточні проблеми;
- прогнозний аналіз: Прогнозний аналіз відповідає на запитання: «Що найімовірніше станеться?» Використовуючи шаблони, знайдені в старих

даних, а також поточні події, аналітики прогнозують майбутні події. Хоча не існує такого поняття, як 100-відсоткове точне прогнозування, шанси покращуються, якщо аналітики мають багато детальної інформації та дисципліну, щоб її ретельно дослідити;

- приписний аналіз: змішайте всі відомості, отримані з інших типів аналізу даних, і ви отримаєте приписний аналіз. Іноді проблему неможливо вирішити лише за допомогою одного типу аналізу, а натомість вимагає кількох аналізів.

Статистичний аналіз: Статистичний аналіз відповідає на запитання «Що сталося?» Цей аналіз охоплює збір даних, аналіз, моделювання, інтерпретацію та представлення за допомогою інформаційних панелей. Статистичний аналіз розбивається на дві підкатегорії:

- описовий: Описовий аналіз працює з повними чи вибірковими підсумковими числовими даними. Він ілюструє середні значення та відхилення в безперервних даних, а також відсотки та частоти в категорійних даних;
- інференційний: Інференційний аналіз працює із зразками, отриманими з повних даних. Аналітик може прийти до різних висновків на основі того самого комплексного набору даних, просто вибравши різні вибірки.

Аналіз тексту: аналіз тексту, який також називають «інтелектуальним аналізом даних», використовує бази даних і інструменти інтелектуального аналізу даних, щоб виявити шаблони, що містяться у великих наборах даних. Він перетворює необроблені дані на корисну бізнес-інформацію. Аналіз тексту, мабуть, є найпростішим і найпрямішим методом аналізу даних. Деякі фахівці використовують терміни «методи аналізу даних» і «техніки аналізу даних» як синоніми. Щоб ще більше ускладнити ситуацію, іноді люди також додають у бійку обговорені раніше «типи аналізу даних»! Ми сподіваємося тут встановити різницю між тим, які типи аналізу даних існують, і різними способами його використання. Хоча існує багато доступних методів аналізу даних, усі вони поділяються на один із двох основних типів: якісний аналіз і кількісний аналіз.

Аналіз якісних даних: метод якісного аналізу даних отримує дані за допомогою слів, символів, зображень і спостережень. Цей метод не використовує статистику.

Найпоширеніші якісні методи включають:

- content Analysis для аналізу поведінкових і словесних даних;
- наративний аналіз для роботи з даними, зібраними з інтерв'ю, щоденників, опитувань.
- обґрунтована теорія для розробки причинно-наслідкових пояснень даної події шляхом вивчення та екстраполяції одного чи кількох минулих випадків.

Кількісний аналіз даних: методи аналізу статистичних даних збирають необроблені дані та обробляють їх у числові дані. Кількісні методи аналізу включають:

- перевірка гіпотез для оцінки істинності певної гіпотези чи теорії для набору даних або демографічних показників;
- середнє або середнє значення визначає загальну тенденцію суб'єкта шляхом ділення суми списку чисел на кількість елементів у списку.

Визначення розміру вибірки використовує невелику вибірку, взяту з більшої групи людей і проаналізовану. Отримані результати вважаються репрезентативними для всього організму. Ми можемо далі розширити наше обговорення аналізу даних, показавши різні методи, розбиті на різні концепції та інструменти

Висновки до розділу 4

З метою покращення роботи сервісів платформи де головною задачею є керування оголошеннями розшуку між користувачами було реалізовано аналітичний метод на основі підходу *Сьоренсена-Дайса*.

Обраний підхід у якості основного надає можливість раціонально використовувати кількість розрахункових ітерацій та запобігає виконанню зайвих розрахунків. Основні задачі розробленого методу полягають в допомозі пошуку оголошень за визначеними критеріями відповідно до описаних категорій моделей даних. Навіть з урахуванням можливих помилок, наявних у початковому наборі даних, метод, хоч і з певною похибкою, проте надасть результати, що з великою ймовірністю будуть відповідати потребам користувача.

Метод покращено додатковими механізмами перевірки наборів даних, для уникнення пустих запитів до системи або запитів, які не відповідають мінімально необхідним критеріям для нормального функціонування аналітичного методу.

Під час реалізації основної функції використовувалися останні методи мови програмування для організації та фільтрації наборів даних, що надає максимальну швидкодію роботи системи та зменшує час взаємодії компонентів системи з іншими інтерфейсами.

Реалізовано метод виводу результатів аналізу даних для покращення безпосередньої перевірки якості роботи методу та формування зручної структури з якою є можливість ефективно взаємодіяти.

ВИСНОВКИ

Семантичний підхід, здається, пропонує інтелектуальний вимір подібності. Цей вимір дуже підходить для пошуку тексту або документів, які дійсно схожі та відповідають змісту контексту. Однак семантична схожість зазвичай залежить від мови та предметної області, тому вона застосовується не до всіх мов. Інакше кажучи, якщо онтологія мови ще не доступна, її потрібно спочатку побудувати. Посилаючись на підходи до подібності тексту, видно, що семантична подібність дуже раціональна для пошуку подібності документів. Також результати дослідження показали, що для мети порівняння тексту, який підкреслює лексичну схожість, ігноруючи субстанцію значення, підходить підхід лексичної подібності. Ці вимірювання можна використовувати для виявлення дублювання або плагіату, не переймаючись контекстом документа. Підходи до подібності рядків принципово не залежать від мови, тому добре працюють для мов різних країн.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ

1. J. Wang, G. Li, and J. Fe, “Fast-join: An efficient method for fuzzy token matching based string similarity join” 2011. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1109/ICDE.2011.5767865
2. A. Kulkarni, C. More, M. Kulkarni, and V. Bhandekar, “Text Analytic Tools for Semantic Similarity” 2016. [Електронний ресурс]. – Режим доступу: http://imperialjournals.com/index.php/IJIR/article/view/688
3. R. Mihalcea, C. Corley, C. Strapparava, “Corpus-based and knowledge-based measures of text semantic similarity” 2006. [Електронний ресурс]. – Режим доступу: http://www.aaai.org/Papers/AAAI/2006/AAAI06-123.pdf
4. A. Budanitsky , G. Hirst, “Evaluating WordNet-based Measures of Lexical Semantic Relatedness” Comput. Linguist 2006. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1162/coli.2006.32.1.13
5. T. Slimani, “Description and Evaluation of Semantic Similarity Measures Approaches” 2013. [Електронний ресурс]. – Режим доступу: https://doi.org/10.5120/13897-1851
6. J. J. Lastra-Díaz, A. García-Serrano, M. Batet, M. Fernández, and F. Chirigati, “HESML: A scalable ontology-based semantic similarity measures library with a set of reproducible experiments and a replication dataset,” 2017. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1016/j.is.2017.02.002
7. L. Meng, R. Huang, J. Gu, “A review of semantic similarity measures in wordnet,” 2013. [Електронний ресурс]. – Режим доступу: https://pdfs.semanticscholar.org/da95/ceaf335971205f83c8d55f2292463fada4ef.pdf
8. G. Varelas, E. Voutsakis, P. Raftopoulou, E. G. M. Petrakis,E. E. Milios, “Semantic similarity methods in wordNet and their application to information retrieval on the web” 2005. [Електронний ресурс]. – Режим доступу: https://doi.org/10.1145/1097047.1097051

9. Документація серверного середовища. [Електронний ресурс]. – Режим доступу: https://nodejs.org/uk/about/
10. Документація фреймворку серверного середовища. [Електронний ресурс]. – Режим доступу: https://developer.mozilla.org/ru/docs/Learn/Serverside/Express_Nodejs/
11. Документація БД. [Електронний ресурс]. – Режим доступу: https://www.guru99.com/what-is-mongodb.html
12. Документація СУБД. [Електронний ресурс]. – Режим доступу: https://mongoosejs.com/
13. Документація фреймоврку клієнта. [Електронний ресурс]. – Режим доступу: https://nextjs.org/learn/foundations/about-nextjs/what-is-nextjs