Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Оптимізація алгоритмів пошуку інформації на веб-сайті з використанням методів машинного навчання

Виконав: студент VI курсу, групи СНнм-61
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

_____ Богуцький М.І.
(підпис) (прізвище та ініціали)

Керівник _____ Готович В.А.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ _____
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ _____
(підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«____» _____ 202_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Богуцькому Михайлу Івановичу
(прізвище, ім'я, по батькові)

1. Тема роботи Оптимізація алгоритмів пошуку інформації на веб-сайті з використанням методів машинного навчання

Керівник роботи Готович Володимир Анатолійович, к.т.н., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 29 » листопада 2024 року № 4/7-1139

2. Термін подання студентом завершеної роботи 26 травня 2025р.

3. Вихідні дані до роботи Науково-технічні публікації (друковані та розміщені в Інтернеті)

4. Зміст роботи (перелік питань, які потрібно розробити)

Вступ. 1 Аналіз предметної області. 1.1 Основні характеристики задачі та особливості алгоритмів пошуку інформації на веб-сайті, 1.2 Аналіз сучасних технологій для побудови пошукових систем, 1.3 Математичні методи, що використовують для аналізу пошукових алгоритмів 1.5 Аналіз моделей машинного навчання, 1.6 Галузі застосування, 1.7 Основні вимоги до пошукової системи 2 Аналіз методів, систем та проєктування пошукових систем. 2.1 Огляд архітектурних підходів до побудови сайтів з інтегрованим пошуком, 2.2 Архітектура та структура сайту з пошуковою системою, 2.3 Вибір оптимального стеку технологій. 2.4 Проектування алгоритмів попередньої обробки запитів користувачів релевантної видачі результатів. 2.5 2.5 Аналіз та проектування методів машинного навчання. 2.6 Моделювання стурктури даних для пошукової системи. 3. Реалізація веб- сайту з оптимізованою пошуковою системою, 3.1 Загальна архітектурна система, 3.2 Серверна логіка для реалізації алгоритмів, 3.3 Реалізація алгоритмів рекомендацій, 3.4 Система підказок та історії пошуку, 3.5 Оптимізація алгоритмів Пошуку на веб-сайті за допомогогою машинного навчання 3.6 Реалізація головної сторінки сайту, 3.6 Загальний опис архітектури веб- додаттку, 3.7 Реалізація особистого кабінету користувача, 3.8 Адміністративна панель, 3.9 Висновки до третього розділу.
4 Охорона праці та безпека в надзвичайних ситуаціях. Висновки. Додатки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Сенчишин В.С., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., ст. викладач		

7. Дата видачі завдання 29 листопада 2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	29.11.2024	
2.	Підбір та опрацювання наукових публікацій, збір	02.12.2024-10.01.2025	
	даних по темі роботи		
3.	Виконання дослідження згідно теми кваліфікаційної	13.01.2025-14.03.2025	
	роботи		
4.	Оформлення розділу «Аналіз основних підходів	17.03.2025-28.03.2025	
	до оптимізації алгоритмів пошуку на веб-сайті»		
5.	Оформлення розділу «Аналіз методів та підходів до	31.03.2025-11.04.2025	
	проєктування систем пошуку інформації на		
	вебсайтах»		
6.	Оформлення розділу «Реалізація веб-сайту з	14.04.2025-25.04.2025	
	оптимізованою пошуковою системою»		
7.	Виконання завдання до підрозділу «Охорона праці»	28.04.2025-02.05.2025	
8.	Виконання завдання до підрозділу «Безпека в	28.04.2025-02.05.2025	
	надзвичайних ситуаціях»		
9.	Оформлення кваліфікаційної роботи	05.05.2025-09.05.2025	
10.	Нормоконтроль	12.05.2025-15.05.2025	
11.	Перевірка на плагіат	16.05.2024	
12.	Попередній захист кваліфікаційної роботи	19.05.2025	
13.	Захист кваліфікаційної роботи	26.05.2025	

Студент _____ Богуцький М.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Готович В.А.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Оптимізація алгоритмів пошуку інформації на веб-сайті за допомогою машинного навчання // Кваліфікаційна робота освітнього рівня «Магістр» // Богуцький Михайло Іванович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНнм-61 // Тернопіль, 2025 // С. 82, рис. – 16, табл. – 1, додат. – 3, бібліогр. – 58.

Ключові слова: вебдодаток, система пошуку інформації, алгоритм пошуку інформації на веб-сайті, машинне навчання, оптимізація алгоритму пошуку інформації.

Кваліфікаційна робота присвячена розв'язанню задачі оптимізації системи пошуку інформації на вебсайті.

В першому розділі описані теоретичні основи пошукових технологій, висвітлено поняття, принципи та класифікації методів пошуку, розглянуто існуючі підходи до реалізації пошукових систем. Також Проаналізовано сучасні тренди та технології в галузі.

В другому розділі наведено вимоги та проєктну структуру системи пошуку інформації на вебсайті, досліджено функціональні потреби користувача та адміністрування, подано схеми бази даних, інтерфейсів та логіки взаємодії.

В третьому розділі описано реалізацію основних модулів веб-додатку, проаналізовано архітектурні рішення та їх ефективність. Наведено результати реалізації інтерфейсу користувача вебсайту, функції пошуку інформації, видачі рекомендацій користувачеві та кабінету користувача.

Об'єкт дослідження: алгоритми пошуку інформації на вебсайті на прикладі вебмагазину.

Предмет дослідження: оптимізація відомих алгоритмів пошуку інформації на вебсайті шляхом застосування методів машинного навчання.

ANNOTATION

Optimization of Information Retrieval Algorithms on a Website Using Machine Learning // Qualification Thesis for the Master's Degree // Mykhailo Bohutskyi // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, Group SNnm-61 // Ternopil, 2025 // 82 pages, figures – 16, tables – 1, appendices – 3, references – 58.

Keywords: web application, information retrieval system, website information retrieval algorithm, machine learning, information retrieval algorithm optimization.

The qualification work is devoted to solving the problem of optimizing the information search system on the website.

The first section describes the theoretical foundations of search technologies, highlights the concepts, principles and classifications of search methods, considers existing approaches to the implementation of search systems. Also, modern trends and technologies in the industry are analyzed.

The second section presents the requirements and design structure of the information search system on the website, examines the functional needs of the user and administration, presents database schemes, interfaces and interaction logic.

The third section describes the implementation of the main modules of the web application, analyzes architectural solutions and their effectiveness. The results of the implementation of the website user interface, information search functions, issuing recommendations to the user and the user's account are presented.

The object of research: algorithms for searching information on the website using the example of a web store.

The subject of research: optimization of known algorithms for searching information on the website by applying machine learning methods.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект.

AJAX (англ. Asynchronous JavaScript and XML) – асинхронна технологія обміну даними між клієнтом і сервером без перезавантаження сторінки.

API (англ. Application Programming Interface) – програмний інтерфейс для взаємодії між різними програмними компонентами.

CMS (англ. Content Management System) – система керування вмістом сайту.

CRUD (англ. Create, Read, Update, Delete) – базові операції над даними в інформаційних системах.

CSV (англ. Comma-Separated Values) – формат зберігання табличних даних у вигляді тексту.

CSS (англ. Cascading Style Sheets) – каскадні таблиці стилів для оформлення веб-сторінок.

DBMS (англ. Database Management System) – система керування базами даних.

DOM (англ. Document Object Model) – об’єктна модель документа в веб-програмуванні.

FAQ (англ. Frequently Asked Questions) – розділ поширених запитань.

FNIR (англ. False Negative Identification Rate) – коефіцієнт помилково-негативної ідентифікації.

FPIR (англ. False Positive Identification Rate) – коефіцієнт помилково-позитивної ідентифікації.

HTML (англ. HyperText Markup Language) – мова розмітки гіпертексту.

HTTP (англ. HyperText Transfer Protocol) – протокол передачі гіпертексту.

HTTPS (англ. HyperText Transfer Protocol Secure) – безпечний протокол передачі гіпертексту.

ID (англ. Identifier) – унікальний ідентифікатор елемента або запису.

JSON (англ. JavaScript Object Notation) – формат обміну структурованими даними.

JS (англ. JavaScript) – мова програмування для веб-розробки.

MVC (англ. Model-View-Controller) – шаблон архітектури програмного забезпечення.

MySQL – система керування реляційними базами даних з відкритим кодом.

NIST (англ. National Institute of Standards and Technology) – Національний інститут стандартів і технологій (США).

PHP (англ. PHP: Hypertext Preprocessor) – мова серверного програмування для веб-розробки.

RAM (англ. Random Access Memory) – оперативна пам’ять.

REST (англ. Representational State Transfer) – архітектурний стиль для побудови API.

SEO (англ. Search Engine Optimization) – пошукова оптимізація веб-ресурсів.

SQL (англ. Structured Query Language) – мова структурованих запитів до бази даних.

UI (англ. User Interface) – інтерфейс користувача.

URL (англ. Uniform Resource Locator) – уніфікований локатор ресурсу в інтернеті.

UX (англ. User Experience) – досвід взаємодії користувача з системою.

XML (англ. eXtensible Markup Language) – розширювана мова розмітки.

ЗМІСТ

ВСТУП

Актуальність теми. З розвитком вебтехнологій та зростанням кількості інформації, що зберігається на онлайн-ресурсах, пошук релевантних даних стає критично важливим для ефективної взаємодії користувача з вебдодатками. Сучасні користувачі очікують швидкого та точного пошуку, персоналізованих рекомендацій і зручної навігації. У цьому контексті особливої ваги набуває впровадження інтелектуальних систем пошуку, що використовують методи машинного навчання для аналізу поведінки користувачів, запитів і контенту. Водночас більшість комерційних пошукових систем є закритими та недоступними для детального аналізу їхніх алгоритмів, що обмежує можливості адаптації та розвитку відкритих або спеціалізованих рішень. Тому розробка ефективної інформаційної системи пошуку з елементами рекомендацій та персоналізації на основі сучасних підходів є актуальним напрямом науково-практичної діяльності.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Магістр» є розробка та впровадження ефективної системи пошуку інформації у вебдодатку з використанням інструментів машинного навчання для підвищення релевантності результатів пошуку та покращення користувацького досвіду. Для досягнення поставленої мети необхідно виконати такі завдання:

– проаналізувати сучасний стан розвитку систем та алгоритмів пошуку інформації на вебсайтах та вебдодатках;

– виконати порівняння ефективності класичних і сучасних інтелектуальних підходів до розв’язання задач пошуку інформації на вебсайті та видачі персоналізованих рекомендацій користувачам на основі результатів пошуку;

– проаналізувати методи збору та обробки пошукових запитів користувачів вебсайту;

– розробити функціональний вебдодаток із вбудованою пошуковою системою та елементами рекомендаційної системи.

Об’єкт дослідження: алгоритми пошуку інформації на вебсайті на прикладі вебмагазину.

Предмет дослідження: оптимізація відомих алгоритмів пошуку інформації на вебсайті шляхом застосування методів машинного навчання.

Наукова новизна одержаних в кваліфікаційній роботі **результатів** полягає у вдосконаленні підходів до реалізації пошукових систем у вебдодатках шляхом інтеграції елементів машинного навчання та рекомендаційних алгоритмів.

Практичне значення одержаних результатів. Виконано макетування та прототипування вебдодатку у вигляді вебмагазину з інтегрованою пошуковою системою та рекомендаційними модулями.

Апробація результатів магістерської роботи. Основні результати проведенихдосл на XII науково-технічній та XIII міжнародній студентській науково-практичній науковій конференції молодих учених та студентів Тернопільського національного технічного університету імені Івана Пулюя (м. Тернопіль, 2024 р.).

Публікації. Основні результати кваліфікаційної роботи опубліковано у двох працях конференції (додатки А та Б).

Структура й обсяг кваліфікаційної роботи. Кваліфікаційна робота складається зі вступу, чотирьох розділів, висновків, списку літератури з 58 найменувань та 3 додатків. Загальний обсяг кваліфікаційної роботи складає 82 сторінки, з них 77 сторінок основного тексту, який містить 16 рисунків та 1 таблицю.

1 АНАЛІЗ ОСНОВНИХ ПІДХОДІВ ДО ОПТИМІЗАЦІЇ АЛГОРИТМІВ ПОШУКУ ІНФОРМАЦІЇ НА ВЕБСАЙТАХ

1.1 Основні характеристики задачі та особливості алгоритмів пошуку інформації на вебсайтах

Вебсередовище стало основною платформою для взаємодії між користувачем та інформаційними ресурсами. Зокрема, динамічний розвиток електронної комерції, онлайн-сервісів та мультимедійних платформ формує нові вимоги до якості, швидкості та точності пошуку інформації. Користувачі очікують миттєвої та персоналізованої відповіді на свої запити, що зумовлює потребу у вдосконаленні традиційних підходів до побудови пошукових систем.

Базові алгоритми пошуку, які орієнтовані лише на ключові слова або просту індексацію, дедалі більше втрачають ефективність через збільшення обсягів даних, складність запитів та високі очікування щодо релевантності результатів. У відповідь на це, зростає інтерес до впровадження інтелектуальних методів, зокрема машинного навчання, що дозволяють враховувати контекст, історію поведінки користувача, синоніміку та семантичні зв’язки між даними.

Пошукова система вебсайту, на відміну від глобальних пошукових систем, таких як Google чи Bing, працює в межах обмеженої, однак часто глибоко структурованої бази даних. Це дозволяє використовувати додаткові метадані, ієрархії та внутрішні зв’язки між об’єктами, що створює передумови для розширеного функціоналу – фільтрації, сортування, автозаповнення запитів, рекомендацій тощо. Водночас така локальна специфіка вимагає ретельного проєктування алгоритмів пошуку, оскільки навіть невеликі помилки в індексації або обробці запитів можуть призвести до нерелевантних результатів або взагалі їх відсутності.

Сучасні користувачі очікують, що система пошуку буде не лише точною, але й інтуїтивно зрозумілою, швидкою, контекстно чутливою та здатною до персоналізації. Ці очікування формують нові виклики перед розробниками, які

повинні забезпечити як технічну ефективність пошукової логіки, так і відповідність поведінковим та лінгвістичним моделям користувачів. Наприклад, система повинна розпізнавати синоніми, допускати помилки у введенні, реагувати на запити у вигляді питань або фраз, а також враховувати історію попередніх дій на сайті.

Важливо також, що пошук у вебсередовищі не є лише функцією відображення результатів за ключовими словами. Це складний інформаційний процес, який поєднує в собі елементи інформаційного пошуку, мовної обробки, аналізу поведінки користувача, а також етапи збору, фільтрації, ранжування та подання даних. З урахуванням динамічного характеру контенту на багатьох сучасних вебресурсах, пошукова система повинна бути адаптивною, масштабованою та здатною до самонавчання.

1.1.1 Вплив якості результатів пошуку інформації на конверсію та досвід користувача

Якість пошуку в інтернет-магазині безпосередньо впливає на те, наскільки швидко та легко користувачі знаходять потрібні товари. Якщо система пошуку працює ефективно, користувач може швидко отримати релевантні результати, що підвищує ймовірність здійснення покупки. Навпаки, якщо пошук неточний, повільний або видає нерелевантні товари, це може призвести до втрати зацікавленості, збільшення часу на пошук або навіть відмови від покупки.

Зручна пошукова система на рисунку 1.1 знижує рівень фрустрації, підвищує лояльність до бренду та сприяє переходу користувача до наступних етапів воронки продажів від перегляду товару до здійснення покупки [5].

Зображення є інфографікою, що демонструє взаємозв’язок між ключовими складовими, які впливають на оптимізацію конверсії користувачів на вебсайті. Графічно зображено, що користувацький досвід (UX), аналітика, копірайтинг і тестування – це не ізольовані елементи, а компоненти єдиного процесу, який формує ефективну взаємодію користувача з сайтом. У центрі уваги - досягнення

мети у вигляді покращення конверсії, тобто перетворення відвідувачів сайту на активних користувачів чи покупців. Візуальне поєднання цих елементів підкреслює, що якість пошуку є лише частиною більшої системи, яка включає весь ланцюг взаємодії з користувачем.

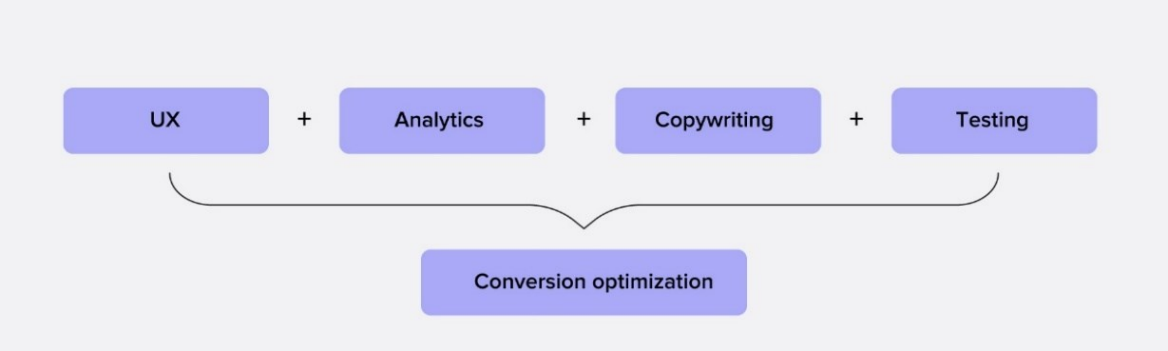


Рисунок 1.1 – Оптимізація конверсії сайті

Досвід користувача тісно пов’язаний з інтуїтивністю та швидкістю роботи пошукового механізму. Якщо пошукова система розпізнає неточності у запитах, виправляє помилки та пропонує релевантні альтернативи, користувачі рідше стикаються з ситуаціями, коли вони не можуть знайти необхідний товар, та покидають веб-сайт в пошуку кращих варіантів. Також важливим є врахування персональних вподобань та історії пошуків, що дозволяє покращити точність видачі та дозволить затримати користувача на сайті довше чим зазвичай, порівняння метрик до і після оптимізації пошуку візуалізовано в таблиці 1.1

Таблиця 1.1 – Порівняння значень метрик до та після оптимізації

Метрика	**До оптимізації**	**Після оптимізації**
Конверсія (%)	2.1	3.8
Середній час на сайті	1 хв 45 сек	3 хв 20 сек
Відсоток виходу з пошуку	60%	35%

У цьому випадку конверсія означає, що користувач, який скористався пошуком на вебсайті, виконав цільову дію, яку вважають бажаним результатом для власника сайту.

Ефективний пошук також зменшує навантаження на інші елементи навігації сайту. Якщо користувачі можуть швидко знайти товари без необхідності довго переглядати категорії та підкатегорії, це покращує загальний досвід використання магазину. Крім того, правильне ранжування результатів допомагає виділити найбільш популярні та актуальні товари, що також позитивно впливає на процес вибору.

Якщо пошук не відповідає очікуванням користувачів, це може призвести до зростання показника відмов, зниження середнього часу перебування на сайті та втрати потенційних покупців. Водночас, добре налаштований пошуковий механізм сприяє збільшенню лояльності клієнтів і підвищенню ймовірності повторних візитів, оскільки користувачі знають, що можуть швидко знайти потрібний товар без зайвих зусиль.

1.1.2 Основні проблеми та виклики при розробці пошукових систем для вебсайтів

Розробка ефективної пошукової системи для інтернет-магазину стикається з низкою проблем і викликів, пов’язаних із точністю, швидкістю та релевантністю результатів. Однією з головних труднощів є розуміння намірів користувача. Люди можуть вводити запити по-різному: хтось шукає товар за назвою бренду, інші використовують загальні описи або навіть припускаються граматичних помилок. Пошукова система повинна адаптуватися до таких варіацій, щоб забезпечити коректну видачу.

Ще одним викликом є правильне ранжування товарів. Недостатньо просто знайти всі релевантні результати - важливо відобразити їх у такому порядку, щоб користувач одразу побачив найбільш підходящі варіанти. Неефективне ранжування може призвести до того, що популярні або високооцінені товари

будуть знаходитись занадто далеко у списку результатів, а менш актуальні позиції займатимуть перші місця.

Проблемою також є велика кількість товарів у каталозі, що ускладнює швидкість пошуку. Чим більше даних необхідно обробити, тим складніше забезпечити миттєву відповідь на запит. Оптимізація індексації та використання ефективних алгоритмів стає критично важливим завданням для збереження швидкодії пошукової системи.

Окрему складність створює персоналізація пошуку. Для точного підбору товарів відповідно до вподобань користувача необхідно аналізувати його попередні пошуки, перегляди та покупки. Це вимагає застосування алгоритмів машинного навчання та збереження великих обсягів даних, що може викликати питання продуктивності та безпеки.

Нарешті, одним із ключових викликів є багатомовність та локалізація пошуку. Якщо магазин працює у різних країнах, потрібно враховувати відмінності у мовних моделях, варіантах написання назв та культурних особливостях запитів. Це ускладнює пошуковий механізм, оскільки він має коректно обробляти запити незалежно від регіону та специфіки мови користувача.

1.2 Аналіз сучасних технологій для побудови пошукових систем

Побудова ефективної пошукової системи у сучасному вебсередовищі вимагає використання комплексу технологій, які охоплюють як серверну (бекенд), так і клієнтську (фронтенд) частину застосунку, а також інструменти обробки запитів, управління даними, асинхронної взаємодії та візуалізації результатів. Вибір відповідних технологічних рішень визначається кількома критеріями: масштабованість, продуктивність, зручність розробки, гнучкість адаптації під специфіку домену та можливість інтеграції з алгоритмами машинного навчання.

Одним із популярних фреймворків для розробки серверної частини вебдодатків є Laravel - PHP-фреймворк, який підтримує структуру MVC (Model-View-Controller), має потужну систему маршрутизації, ORM (Eloquent), інструменти для кешування, роботи з чергами, а також гнучку підтримку RESTful API. Laravel надає ефективні засоби для реалізації логіки обробки пошукових запитів, їхньої валідації, попередньої обробки та взаємодії з базами даних. Завдяки інтеграції з пакетами на кшталт Laravel Scout або TNTSearch можна реалізувати повнотекстовий пошук або підключити Elasticsearch [6].

З погляду фронтенду, для створення адаптивного і зручного інтерфейсу використовуються такі бібліотеки та фреймворки, як Tailwind CSS та Bootstrap 5. Tailwind дозволяє реалізовувати компонентно-орієнтований підхід до стилізації, надаючи розробнику гнучкість і контроль над візуальним представленням результатів пошуку. Bootstrap, у свою чергу, забезпечує швидке макетування інтерфейсу і підтримку кросбраузерності, що важливо для зручного користувацького досвіду [6].

Для візуалізації та взаємодії з пошуковими результатами, зокрема при реалізації каруселей товарів, галерей або прокручування списків, ефективним є використання бібліотеки SwiperJS. Вона дозволяє створювати високопродуктивні слайдери, що добре інтегруються з Tailwind та адаптуються до мобільних пристроїв.

Крім того, для забезпечення динамічної взаємодії без перезавантаження сторінки застосовується Ajax (Asynchronous JavaScript and XML), що дозволяє реалізувати «живий пошук» (live search), автозаповнення полів (autocomplete) та миттєве оновлення результатів без втрати продуктивності або зручності користування. З технічного боку це забезпечується через запити XMLHttpRequest або з використанням бібліотек типу Axios чи jQuery Ajax.

З-поміж альтернативних технологій, які також активно використовуються у сфері побудови пошукових систем, слід відзначити Elasticsearch як високопродуктивний пошуковий рушій на основі Apache Lucene, що забезпечує масштабований повнотекстовий пошук з підтримкою морфології, вагових

моделей і агрегування результатів. Також зростає роль GraphQL як альтернативи REST для отримання гнучких і оптимізованих запитів до пошукових сервісів.

Загалом, технологічна екосистема сучасного пошуку формується на перетині традиційної веброзробки та інтелектуальної обробки даних. Комплексне використання таких інструментів, як Laravel, Tailwind CSS, Ajax і SwiperJS, дозволяє створювати швидкі, масштабовані й зручні для користувача системи, які забезпечують високу релевантність результатів і позитивний досвід взаємодії з платформою.

1.3 Математичні методи, що використовуються для аналізу пошукових алгоритмів

Оптимізація пошукових алгоритмів на веб-сайтах, зокрема в інтернет-магазинах, потребує використання ґрунтовних математичних підходів для аналізу даних, виявлення закономірностей та покращення релевантності результатів. Математичне моделювання дозволяє здійснювати формалізацію пошукового процесу, що забезпечує точнішу обробку запитів та структурування інформації. У цьому контексті застосовуються статистичні методи, векторні простори, алгоритми кластеризації та математичні моделі ранжування.

Одним із базових підходів є статистичний аналіз запитів, що дозволяє виявити частотні закономірності в поведінці користувачів на рис. 1.2.

Цей метод ґрунтується на підрахунку частоти використання певних термінів у запитах, аналізі кількості кліків по результатах, а також оцінці конверсії для різних типів запитів. Завдяки статистичному аналізу можна ідентифікувати найпопулярніші категорії товарів, виявити сезонні тенденції та сформувати прогноз щодо майбутнього попиту. Крім того, статистика дає змогу ефективно виявляти неефективні або неоднозначні запити, які потребують уточнення чи додаткової обробки.

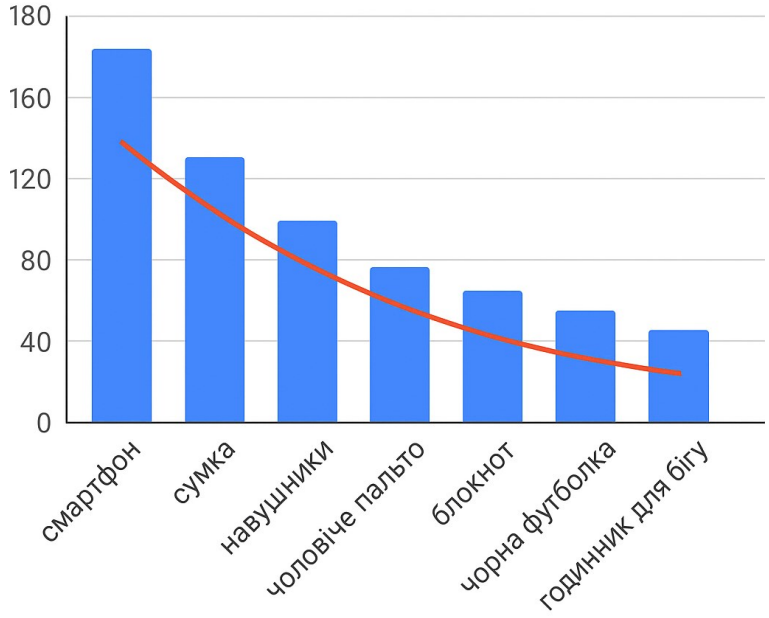


Рисунок 1.2 – Статистичний аналіз запитів

Іншим важливим підходом є модель векторного простору, яка дозволяє представити тексти (запити користувачів і описи товарів) у вигляді векторів у багатовимірному просторі. У цій моделі кожен документ або запит розглядається як вектор, компоненти якого відповідають окремим словам або ознакам. Найчастіше застосовується TF-IDF (Term Frequency–Inverse Document Frequency) як вагова схема, що дозволяє підкреслити значущість окремих термінів у межах усього корпусу даних. Порівняння подібності між запитом і товаром відбувається шляхом обчислення косинусної відстані між відповідними векторами. Це забезпечує об'єктивну метрику релевантності.

Ще одним напрямом є використання кластеризації, яка дає змогу групувати схожі запити або товари на рис.1.3.

З математичної точки зору, кластеризація передбачає поділ множини об’єктів на неперетинні підмножини (кластери) на основі заданої метрики подібності. Часто застосовуються такі алгоритми, як k-середніх (k-means), DBSCAN або ієрархічна кластеризація. У сфері пошуку це дозволяє: по-перше, зменшити обсяг пошукової вибірки за рахунок попередньої сегментації; по-друге, формувати персоналізовані підказки або результати, орієнтуючись на групу запитів, до якої належить користувач.

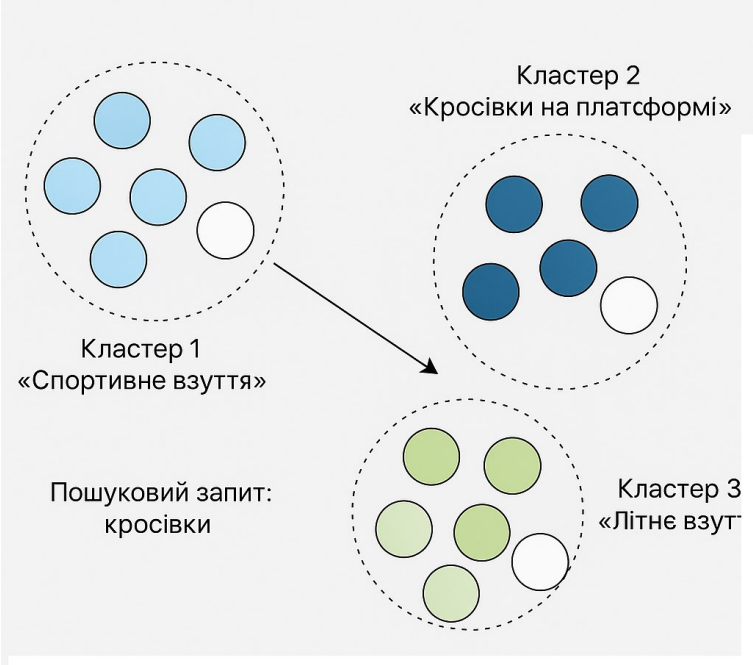


Рисунок 1.3 – Використання кластеризації для покращення алгоритмів пошуку

Не менш значущими є алгоритми ранжування, які використовуються для визначення порядку виведення результатів пошуку на рис. 1.4 [7].

Рисунок 1.4 – Алгоритми ранжування

Базова математична модель ранжування – це побудова функції оцінки релевантності, яка надає кожному результату числовий рейтинг. Ця функція може базуватись на лінійних моделях, таких як регресійний аналіз, або на складніших нелінійних функціях, що включають вагові коефіцієнти різних

параметрів (відповідність ключовим словам, популярність, рейтинг, кількість переглядів тощо). Більш сучасні підходи, включно з використанням методів машинного навчання (наприклад, Learning to Rank), будуються на великому наборі ознак і використовують математичну оптимізацію для навчання моделі ранжування на основі історичних даних.

Особливої уваги заслуговує обробка текстових даних, що включає в себе попередню нормалізацію запитів (стемінг, лематизація), видалення стоп-слів, побудову словників синонімів та багатомовну обробку. Усі ці етапи базуються на лінгвістичному та статистичному аналізі, що дозволяє зменшити кількість нерелевантних результатів та підвищити точність пошуку.

1.4 Методи покращення взаємодії користувача з пошуковою системою

Взаємодія користувача з пошуковою системою є одним із ключових чинників успішності інтернет-магазину, оскільки від ефективності пошуку залежить швидкість знаходження товару та загальне враження від роботи з платформою. Тому оптимізація взаємодії спрямована на зменшення когнітивного навантаження користувача, підвищення релевантності результатів та забезпечення інтуїтивно зрозумілого процесу пошуку.

Одним із базових методів покращення взаємодії є впровадження функції автодоповнення запиту. Автодоповнення дозволяє користувачу бачити підказки ще до завершення введення запиту, що сприяє прискоренню процесу пошуку та зменшенню кількості орфографічних помилок. Ефективне автодоповнення базується на аналізі популярних запитів, історії пошуку, а також передбачає обробку синонімів і варіацій написання.

Ще одним важливим аспектом є надання фільтрів та механізмів сортування результатів пошуку. Можливість обмежити результати за різними критеріями (ціною, брендом, рейтингом, категорією тощо) дає користувачу змогу швидко звузити вибір до найбільш релевантних товарів. Інтуїтивно зрозумілий інтерфейс для застосування фільтрів суттєво підвищує зручність використання пошуку.

Впровадження механізмів обробки помилок у запитах також є важливою складовою покращення взаємодії. Пошукова система має автоматично розпізнавати друкарські помилки, надавати пропозиції щодо виправлення запиту або автоматично адаптувати пошук до найближчих за значенням результатів. Це дозволяє уникнути ситуацій, коли користувач залишається без результатів через незначну неточність у формулюванні запиту.

Особливу увагу слід приділяти візуалізації результатів пошуку. Важливими елементами є чітка структура карток товарів, наявність ключової інформації (ціна, назва, зображення), можливість швидкого перегляду деталей без переходу на окрему сторінку. Використання сучасних UI-компонентів, таких як каруселі товарів або випадаючі підказки, сприяє створенню більш динамічного та привабливого інтерфейсу.

Додатковим засобом оптимізації є впровадження персоналізації пошуку. Системи, що враховують попередні дії користувача (перегляди товарів, здійснені покупки, збережені фільтри), можуть пропонувати більш релевантні результати та адаптувати процес пошуку до індивідуальних уподобань кожного користувача.

1.5 Аналіз моделей машинного навчання

Моделі машинного навчання (ML) відіграють ключову роль в удосконаленні систем пошуку в онлайн-магазинах, забезпечуючи більш релевантні результати, персоналізацію та ефективну обробку запитів користувачів. Застосування таких моделей дозволяє подолати обмеження традиційного пошуку на основі ключових слів і забезпечити кращу взаємодію з користувачем [10].

Одним із підходів є використання векторного подання тексту за допомогою моделей обробки природної мови (NLP) [9], зокрема таких як Word2Vec, FastText, або сучасні трансформери (наприклад, BERT або його оптимізовані версії типу MiniLM). Ці моделі дозволяють аналізувати семантику

запиту, зіставляючи його зі змістом товарних описів навіть при неточному введенні або вживанні синонімів.

Для аналізу поведінки користувачів ефективними є алгоритми кластеризації (наприклад, K-Means, DBSCAN) та моделі класифікації (наприклад, логістична регресія, дерева рішень, XGBoost), які дозволяють виявляти шаблони пошукової активності, розпізнавати наміри та адаптувати результати під конкретні групи користувачів.

Рекомендаційні системи на базі моделей колаборативної фільтрації (наприклад, ALS – alternating least squares), а також гібридні підходи з використанням нейронних мереж дозволяють не лише покращити пошук, а й підвищити конверсію завдяки персоналізованим підказкам, що враховують історію переглядів, покупки та схожість між товарами.

Інтеграція моделей ранжування [8] (наприклад, Learning to Rank, LambdaMART) забезпечує динамічне упорядкування результатів видачі відповідно до ймовірної корисності для конкретного користувача, що покращує релевантність та швидкість пошуку.

1.6 Галузі застосування вебсайтів з функцією пошуку інформації

Вебсайти широко застосовуються в інформаційних, комерційних, освітніх, адміністративних та розважальних сферах, виконуючи функції комунікації, надання послуг і поширення даних. З розвитком цифрових технологій зростає обсяг доступної інформації, що висуває підвищені вимоги до ефективності пошукових алгоритмів, які забезпечують швидкий і релевантний доступ до потрібного контенту в межах окремого ресурсу або системи вебсторінок, зокрема веб-ресурси комерційного спрямування, набули широкого розповсюдження в усіх сферах економічної та соціальної діяльності. Онлайн-магазини, як один із ключових інструментів електронної комерції, є актуальними не лише у традиційному роздрібному продажі, але й у низці суміжних галузей,

де цифровізація процесів сприяє підвищенню ефективності обслуговування споживачів і зменшенню операційних витрат.

Передусім, електронна комерція є основою сучасної торгівлі товарами широкого вжитку - одягом, електронікою, побутовою технікою, косметикою, книгами тощо. Завдяки гнучкості веб-інтерфейсів, можливості інтеграції платіжних систем і засобів логістики, онлайн-магазини забезпечують повний цикл обслуговування замовлень - від вибору товару до його доставки кінцевому споживачу.

У промисловості веб-ресурси комерційного типу використовуються для B2B-продажів (business-to-business), що включає оптову торгівлю обладнанням, сировиною, комплектуючими та інструментами. Веб-платформи дозволяють реалізовувати автоматизоване оформлення замовлень, інтеграцію з ERP-системами підприємств, а також здійснювати попередню технічну консультацію через чат-ботів і системи підтримки.

У сфері послуг застосування онлайн-магазинів є не менш актуальним. Зокрема, йдеться про продаж нематеріальних продуктів, таких як програмне забезпечення, електронні книги, музика, квитки на заходи, а також послуг - наприклад, консультацій, онлайн-курсів, вебінарів, підписок. Для цього використовуються спеціалізовані платформи з інтегрованою авторизацією, доступом до цифрових товарів та аналітичними модулями.

Окремої уваги заслуговує медична та фармацевтична галузь, де онлайн-магазини дозволяють реалізовувати медикаменти, медичні вироби та засоби гігієни з можливістю консультацій і верифікації рецептів. У таких випадках особливе значення має забезпечення відповідності вимогам законодавства, зокрема у сфері захисту персональних даних та контролю за обігом лікарських засобів.

У сфері культури та мистецтва онлайн-комерція використовується для реалізації предметів ручної роботи, творів мистецтва, декоративних виробів. Такі веб-платформи часто мають індивідуальний дизайн і підвищену увагу до

візуального компонента, що формує відповідний емоційний зв’язок із користувачем.

1.7 Основні вимоги до системи пошуку інформації на вебсайті

Пошукова система повинна забезпечувати високу точність і релевантність видачі, що досягається за рахунок впровадження складних алгоритмів морфологічного аналізу та лемматизації тексту, а також глибокого контекстного аналізу запитів користувачів. Важливим аспектом є використання методів обробки синонімів, сприйняття варіативності мовних конструкцій і фразеологічних одиниць, що дозволяє максимально наблизити результати до намірів користувача. Для підтримки високої продуктивності пошуку необхідно впроваджувати ефективні методи індексації великих обсягів даних, оптимізовані запити до бази даних і механізми кешування результатів, що суттєво знижують час відповіді системи навіть при значних навантаженнях [11].

Пошукові алгоритми повинні передбачати підтримку нечіткого пошуку, що забезпечує коректну обробку запитів з орфографічними помилками, опечатками або неточностями, а також можливість часткового співпадіння за ключовими словами. Важливою функцією є гнучкість у формуванні пошукових запитів, яка реалізується через підтримку фільтрації, сортування, а також пошуку по різних полях бази даних, що дозволяє користувачам швидко знаходити потрібну інформацію за різними критеріями. Інтерфейс пошукової системи повинен бути інтегрований у вебзастосунок із застосуванням технологій AJAX, що дозволяють динамічно оновлювати результати пошуку в режимі реального часу без необхідності перезавантаження сторінок, підвищуючи таким чином зручність і швидкість взаємодії користувача з системою.

Для забезпечення стабільної роботи в умовах зростаючого обсягу даних і одночасної роботи великої кількості користувачів, система повинна підтримувати масштабованість як на рівні програмного забезпечення, так і на рівні інфраструктури, включно з можливістю горизонтального масштабування

та розподіленої обробки запитів. Одним із ключових аспектів є також забезпечення безпеки, що включає захист від SQL-ін’єкцій, обмеження частоти запитів, валідацію вхідних даних і загальний захист від типових вразливостей вебзастосунків. Реалізація цих вимог гарантує надійність роботи пошукової системи, швидкість обробки запитів і високу якість результатів у складних і динамічних умовах сучасних вебресурсів.

1.8 Висновок до першого розділу

В першому розділі кваліфікаційної роботи описано результат проведеного аналітичного дослідження було встановлено, що сучасні пошукові системи в веб-середовищі потребують комплексного підходу до обробки інформації, що базується на поєднанні різноманітних методів аналізу даних.

Виявлено, що ефективність пошукових алгоритмів залежить не лише від швидкості індексації великих обсягів інформації, а й від глибини розуміння семантики та контексту користувацьких запитів. Особливу увагу приділено механізмам морфологічного аналізу, нормалізації запитів та фільтрації шуму, що дозволяють підвищити точність видачі релевантних результатів.

Аналіз галузей застосування показав, що пошукові системи повинні адаптуватися до специфіки різних типів вебресурсів, забезпечуючи гнучкість і масштабованість, що критично важливо для підтримки різнорідних типів контенту - від інформаційних порталів до комерційних платформ.

Визначені основні вимоги до пошукових систем, зокрема швидкодія, точність, безпека, стійкість до навантажень і зручність інтеграції, формують чіткі орієнтири для розробників. Врахування цих факторів у процесі оптимізації пошукових алгоритмів забезпечує не лише підвищення ефективності їх роботи, а й покращення користувацького досвіду завдяки більш релевантній та швидкій видачі інформації.

2 АНАЛІЗ МЕТОДІВ ТА ПІДХОДІВ ДО ПРОЄКТУВАННЯ СИСТЕМ ПОШУКУ ІНФОРМАЦІЇ НА ВЕБСАЙТАХ

2.1 Огляд архітектурних підходів до побудови сайтів з інтегрованою системою пошуку інформації

Вибір архітектури вебсайту з інтегрованою пошуковою системою суттєво впливає на продуктивність, масштабованість і зручність подальшого розвитку проєкту. Залежно від складності функціоналу, очікуваного навантаження та специфіки користувацьких сценаріїв, застосовуються різні архітектурні підходи, кожен з яких по-своєму визначає реалізацію пошуку.

Найпростішим і традиційним варіантом є монолітна архітектура, де вся логіка сайту, включно з пошуковою системою, об’єднана в єдине програмне середовище. Цей підхід зручний для реалізації невеликих проєктів із базовим функціоналом: він дозволяє швидко розробити і розгорнути систему, забезпечує цілісність коду та спрощує його супровід. Проте зі збільшенням обсягів даних і кількості запитів навантаження на пошук зростає, а можливості його оптимізації залишаються обмеженими через спільність ресурсів з усією системою.

Щоб уникнути цих обмежень, дедалі частіше використовується мікросервісна архітектура, яка дозволяє винести пошукову систему в окремий сервіс. У такому підході кожен модуль виконує чітко визначену роль, а пошуковий компонент може масштабуватися незалежно від інших частин сайту. Це забезпечує гнучкість, кращу розподілену обробку запитів та можливість інтеграції зі спеціалізованими пошуковими платформами, як-от Elasticsearch чи Algolia, що значно підвищує якість і швидкодію пошуку.

У мікросервісному підході все частіше реалізовуються сучасні інтерфейсні архітектури, зокрема SPA (Single Page Application). У таких додатках пошуковий інтерфейс функціонує без оновлення сторінки, що забезпечує високу швидкість і зручність користування. Запити до пошукового API надсилаються асинхронно, що дозволяє оперативно відображати результати без затримок. Проте це

потребує ретельного проектування API та захисту від надмірного навантаження на бекенд.

Зі SPA природно пов’язаний API-first підхід, у якому вся взаємодія клієнта з пошуковою системою (і не лише) відбувається через чітко визначені API-ендпоінти. Це дозволяє централізовано управляти логікою пошуку, забезпечити повторне використання функціоналу в різних середовищах (веб, мобільні додатки, сторонні сервіси) та масштабувати систему в майбутньому без зміни клієнтської частини. Такий підхід передбачає високий рівень стандартизації та документованості інтерфейсів, що особливо важливо при роботі з великими обсягами пошукових запитів.

Також важливим фактором у виборі архітектури є підхід до рендерингу сторінок. У SSR (Server-Side Rendering) контент генерується на сервері до надсилання клієнту, що підвищує швидкість першого завантаження та полегшує індексацію пошуковими системами. Це критично для сайтів, де швидкість і SEO-оптимізація відіграють ключову роль. Натомість CSR (Client-Side Rendering), який характерний для SPA, забезпечує кращу динамічність і взаємодію користувача з інтерфейсом після початкового завантаження, однак потребує додаткових механізмів SEO-оптимізації та обробки даних.

2.2 Архітектура та структура вебсайту з пошуковою системою

Ефективна архітектура є фундаментом для стабільної, масштабованої та зручної в підтримці пошукової системи. Загалом, архітектуру такого вебзастосунку можна представити як багаторівневу систему. Рівень представлення відповідає за відображення інформації користувачеві та обробку його взаємодії з системою, включаючи вебсторінки на HTML-шаблонах з адаптивним дизайном завдяки Tailwind CSS та Bootstrap 5, клієнтську логіку на JavaScript з використанням SwiperJS для каруселей та Ajax для асинхронних запитів, а також контролери у фреймворці Laravel для обробки HTTP-запитів та керування відображенням.

Рівень даних забезпечує взаємодію з джерелами даних, де зберігається проіндексована інформація. Основними компонентами є пошуковий індекс, часто реалізований за допомогою технологій на кшталт Elasticsearch або Solr, джерела даних, що містять оригінальну інформацію, та модуль індексації, що відповідає за збір, обробку та додавання даних до пошукового індексу.

У контексті використання фреймворку Laravel, структура файлів та компонентів організована наступним чином: контролери знаходяться в app/Http/Controllers/, маршрути визначені у routes/web.php, Blade-шаблони розміщені в resources/views/, JavaScript-файли, включаючи код для Ajax та SwiperJS, зберігаються у public/js/. Сервісні класи реалізують бізнес-логіку і можуть розміщуватися в app/ або окремих піддиректоріях, а конфігураційні файли для підключення до баз даних та пошукового індексу знаходяться у config/ [12].

Застосування такої архітектури з чітким розділенням відповідальності та використанням сучасних веб-технологій, таких як PHP Laravel, Tailwind CSS, Bootstrap 5, SwiperJS та Ajax, дозволяє створити ефективний, масштабований та зручний у використанні вебзастосунок з потужною пошуковою функціональністю на рис. 2.1.

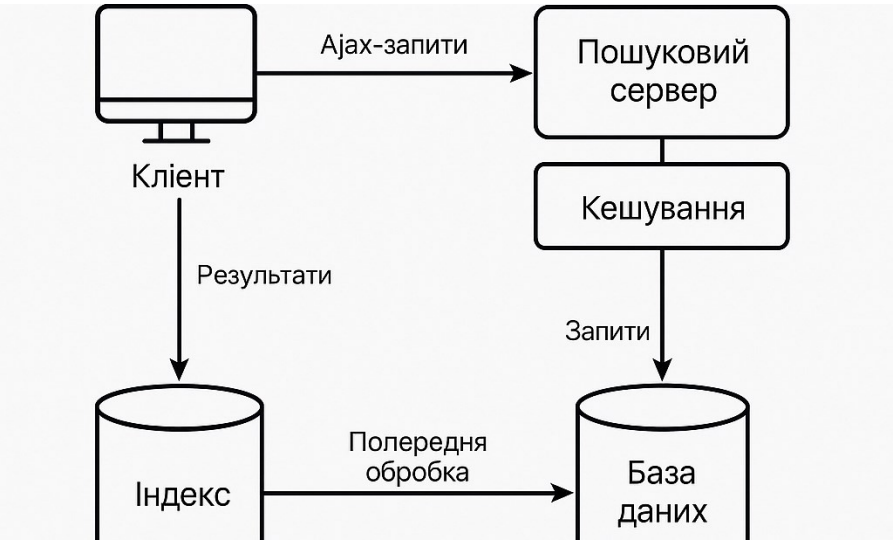


Рисунок 2.1 – Структура вебзастосунку з пошуковою системою

Рівень бізнес-логіки містить основні алгоритми та правила роботи пошукової системи. Ключові компоненти включають обробник пошукових запитів для аналізу вхідних запитів, пошуковий рушій для взаємодії з індексом, модуль ранжування для визначення порядку результатів, модуль обробки результатів для їх форматування та API, що у Laravel реалізується через маршрути та контролери, для взаємодії з рівнем представлення [13].

2.3 Вибір оптимального стеку технологій для реалізації сайту з системою пошуку інформації

Враховуючи специфіку задач, пов’язаних з ефективною обробкою пошукових запитів, швидким відгуком інтерфейсу та масштабованістю, доцільно проаналізувати потенціал кожної із обраних технологій: PHP (Laravel), Tailwind CSS, MySQL, Ajax, SwiperJS.

Laravel – один із найпопулярніших PHP-фреймворків - забезпечує гнучку структуру проєкту, розвинуту підтримку шаблонів (Blade), потужні можливості роботи з базами даних (Eloquent ORM) та зручні інструменти для створення RESTful API. Це дозволяє ефективно реалізувати пошукову логіку як окремий маршрут або контролер, а також інтегрувати складні алгоритми ранжування, фільтрації та кешування. Laravel також має вбудовану підтримку черг, що дає змогу обробляти ресурсоємні запити (наприклад, індексацію товарів чи ведення пошукових підказок) у фоновому режимі.

Однією з ключових переваг використання фреймворку Laravel у контексті реалізації пошукової системи є його архітектура, що базується на шаблоні Model-View-Controller (MVC). Цей підхід забезпечує чітке розділення логіки даних, обробки запитів та представлення інтерфейсу, що особливо важливо при організації пошукового функціоналу. Зокрема, пошуковий запит обробляється в контролері, який взаємодіє з моделлю для отримання відповідних даних з бази, після чого результати передаються у вигляд (view) для відображення користувачу. Такий структурований підхід не лише спрощує підтримку та

масштабування проєкту, але й забезпечує більш чисту та безпечну обробку запитів, включаючи фільтрацію введення, валідацію параметрів пошуку та оптимізований доступ до бази даних на рис. 2.2 [14].

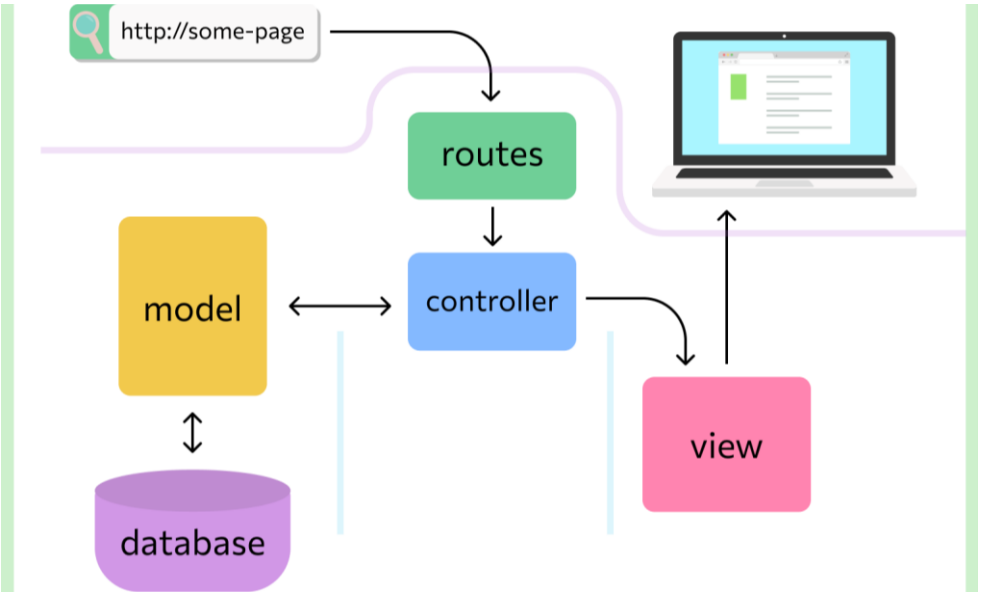


Рисунок 2.2 – Структура MVC в Laravel

Для ефективної реалізації пошукового функціоналу необхідна надійна та продуктивна система управління даними, яка б забезпечувала швидкий доступ до великої кількості записів. У цьому контексті MySQL виступає в ролі основної системи управління базами даних, що дозволяє реалізувати як прості текстові запити, так і більш складні фільтраційні або агреговані вибірки. Завдяки індексації, повнотекстовому пошуку (FULLTEXT), можливостям оптимізації запитів і складній фільтрації, MySQL може забезпечити базову ефективність пошуку без залучення зовнішніх пошукових платформ. Крім того, взаємодія з Eloquent ORM у Laravel дозволяє зручно моделювати запити й застосовувати умовну логіку без значних витрат на розробку. Завдяки чіткому визначенню таких зв’язків у моделях Laravel, побудова запитів для пошуку стає очевидною та гнучкою. Наприклад, для пошуку всіх продуктів певного бренду з урахуванням лише активних варіантів достатньо виконати простий скрипт який подано в лістингу 2.1

Лістинг 2.1 – Скрипт побудови запиту

```
$products = Product::whereHas('brand', fn($q) =>
    $q->where('name', 'LIKE', "%{$searchBrand}%")
)->where('status', 'active')
  ->with(['variants' => fn($q) => $q->where('status', 'active')])
  ->paginate(20);
```

Побудова таких відносно складних SQL-запитів перетворюється на лаконічні виклики методів у моделях, що значно підвищує ефективність розробки. Завдяки архітектурному паттерну MVC та чітко визначеним зв'язкам у Eloquent-моделях, структура даних і логіка їх взаємодії залишаються прозорими та легкими для розуміння. При цьому розробник отримує можливість оптимізувати пошукові запити за допомогою таких механізмів, як агреговані фільтрації, жорстке завантаження пов'язаних моделей і використання індексів у базі даних. Будь-які зміни у структурі даних, як-от додавання нових полів або зв'язків, одразу відображаються у відповідних моделях, що дозволяє уникнути зайвого переписування SQL-коду та сприяє гнучкості й масштабованості проєкту на рис. 2.3.

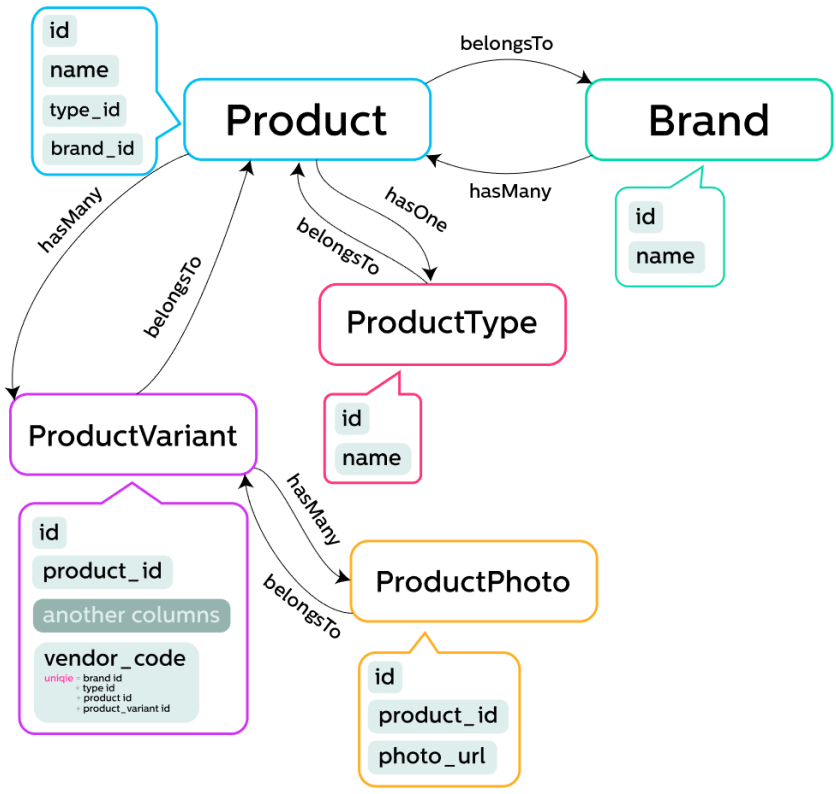


Рисунок 2.3 – Модель даних Eloquent ORM

На діаграмі показано базову модель даних, створену з використанням Eloquent ORM у Laravel. Вона описує основні сутності та їх зв’язки. Товар (Product) пов’язаний із категорією (ProductType) і брендом (Brand), а також має багато варіантів (ProductVariant) і фотографій (ProductPhoto). Варианти і фото, у свою чергу, пов’язані з товаром і між собою. Brand і ProductType – прості сутності, які пов’язані з товарами через відповідні зв’язки. Усі відносини реалізовані за допомогою стандартних Eloquent методів.

Для покращення користувацького досвіду в контексті пошуку важливо забезпечити оперативну взаємодію інтерфейсу з сервером. У цьому аспекті ключову роль відіграє Ajax – технологія, яка дозволяє відправляти запити до сервера без перезавантаження сторінки. Завдяки Ajax пошук може працювати в режимі реального часу, відображаючи результати миттєво після введення символів. Це значно підвищує зручність і швидкість взаємодії користувача з сайтом.

Для створення адаптивного, сучасного й швидкодіючого інтерфейсу застосовується Tailwind CSS [15]. Його утилітарна модель дозволяє гнучко стилізувати компоненти пошуку, форми, списки результатів та інші елементи без зайвого перевантаження DOM. Tailwind також сприяє створенню легкого фронтенду, що прискорює рендеринг сторінки та покращує загальну продуктивність пошукового інтерфейсу.

SwiperJS, хоч і не є безпосередньо дотичним до пошукової логіки, використовується для реалізації динамічних компонентів інтерфейсу, таких як слайдери результатів або рекомендовані товари на основі пошуку. Його сумісність із Tailwind та можливість керування подіями через JavaScript дозволяє інтегрувати його в систему пошуку для візуального покращення презентації результатів [16].

Узагальнюючи, обраний стек технологій забезпечує повноцінну підтримку розробки як функціональної, так і продуктивної пошукової системи. Laravel і MySQL формують надійну серверну основу, Ajax забезпечує інтерактивність, Tailwind CSS відповідає за ефективне стилізування інтерфейсу, а SwiperJS

підвищує якість візуального сприйняття результатів пошуку. У комплексі ці інструменти дозволяють реалізувати масштабований та адаптивний сайт з сучасною пошуковою системою, що відповідає вимогам як функціональності, так і зручності користування.

2.4 Проєктування алгоритмів попередньої обробки запитів користувача та формування релевантної видачі результатів

Ефективність пошукової системи значною мірою залежить від того, наскільки точно і швидко вона може інтерпретувати запити користувачів. Для цього важливо забезпечити якісну попередню обробку тексту запиту перед безпосереднім пошуком по базі даних. Така обробка дозволяє зменшити вплив варіативності природної мови на результати пошуку та підвищити релевантність відповідей. У межах даного проєкту реалізується послідовність етапів попередньої обробки, які враховують специфіку української та англійської мов.

Першим етапом є токенізація – процес розбиття запиту на окремі смислові одиниці, або токени. Наприклад, запит «Смартфон Apple з великим екраном» буде розділено на набір окремих слів. Цей крок дозволяє працювати з кожною одиницею запиту окремо, спрощуючи подальшу обробку.

Наступним етапом є приведення всіх токенів до нижнього регістру. Такий підхід усуває чутливість до регістру, що є критично важливим при порівнянні тексту в умовах пошуку. Наприклад, слова «Apple» і «apple» будуть трактуватися як однакові, що дозволяє уникнути втрати релевантних результатів.

Після нормалізації регістру виконується фільтрація запиту шляхом видалення так званих стоп-слів - слів, які не несуть суттєвого смислового навантаження у контексті пошуку. До таких слів належать прийменники, сполучники, займенники, частки тощо. Видалення цих слів дозволяє зосередитися на ключових термінах запиту.

Заключним етапом є зведення кожного токена до його базової форми за допомогою стемінгу або лематизації. Стемінг передбачає обрізання слова до

його кореня (наприклад, «покупка», «купити», «покупець» можуть зводитися до «куп»), тоді як лематизація передбачає приведення слова до його словникової форми. У контексті даної системи можливе використання лематизаторів або словникових баз для найбільш уживаних мов, що підвищує точність пошуку при збереженні продуктивності системи.

Після обробки запиту користувача система переходить до етапу пошуку відповідних результатів у базі даних. На цьому етапі особливо важливо забезпечити релевантність - тобто відповідність знайдених результатів інформаційному запиту користувача. Релевантність досягається шляхом зіставлення ключових слів із текстовим вмістом полів бази даних, врахування ваги окремих полів, рейтингових характеристик товарів та динамічного сортування.

В основі реалізації лежить концепція часткового збігу, коли результати підбираються не лише за точним співпадінням, а й за частковим входженням ключових слів у відповідні поля: назву товару, опис, категорію або бренд. Для цього використовуються SQL-запити з умовами LIKE на рис. 2.4.

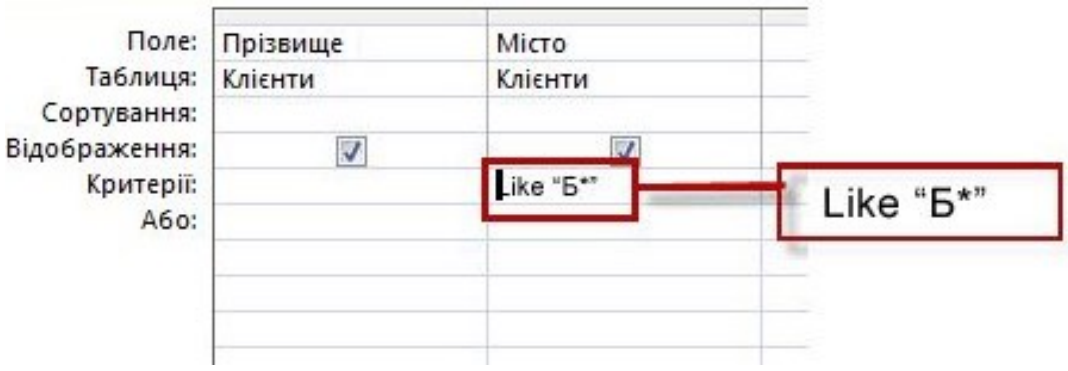


Рисунок 2.4 – Використання умови LIKE для пошуку даних

Іноді з підтримкою повнотекстового пошуку через MATCH AGAINST, якщо структура бази та обсяг даних дозволяють. Це дозволяє ефективно віднаходити результати навіть у випадках, коли запит не є точним або містить помилки.

Для підвищення якості видачі застосовується ранжування результатів відповідно до визначених критеріїв. Найвищу релевантність отримують товари, в яких збіг виявлено в назві, а вже потім - у додаткових полях, таких як опис чи категорія. Така стратегія дає змогу підняти найбільш релевантні товари вище у списку результатів, навіть якщо збіги за іншими параметрами незначні.

Окрім текстової відповідності, у систему пошуку можуть бути інтегровані додаткові критерії сортування: популярність товару, кількість продажів, рейтинг користувачів, наявність у магазині, дата додавання та інші. Це дає змогу формувати гнучку видачу, яка відповідає не лише змісту запиту, а й очікуванням користувача щодо якості та актуальності товару.

Щоб забезпечити відповідність видачі фільтрам користувача, результати пошуку також динамічно фільтруються за категоріями, брендами, ціновим діапазоном та іншими параметрами. Усі ці обмеження накладаються одночасно із текстовим пошуком, що дозволяє формувати точні вибірки товарів, релевантні як за змістом, так і за параметрами.

2.5 Аналіз та проєктування методів машинного навчання

Інтеграція машинного навчання у вебсайт з пошуковою системою, розроблений на основі Laravel та MySQL, потребує адаптації класичних алгоритмів до обмежень серверного оточення, орієнтованого на PHP. У межах цього проєкту доречним є застосування алгоритмів, які не вимагають високих обчислювальних ресурсів і можуть бути реалізовані без зовнішніх бібліотек - шляхом ручного програмування логіки у PHP та SQL-запитах.

Одним із найбільш придатних методів є TF-IDF (Term Frequency–Inverse Document Frequency) [17], який дозволяє оцінити релевантність кожного товару або сторінки до пошукового запиту. У межах MySQL реалізація TF-IDF можлива через попередній підрахунок частот термінів у текстових полях (наприклад, назвах, описах, тегах), збереження цих коефіцієнтів у окремих таблицях, і подальше ранжування результатів за зваженою оцінкою. Серверна частина на

PHP забезпечує трансформацію запиту користувача у набір термінів та їх зіставлення з базою.

Другий підхід – наївний байєсівський класифікатор, який може бути використаний для визначення ймовірного наміру користувача на основі попередніх сесій. Наприклад, на основі клікстріму, категорій відвіданих товарів та виконаних запитів можливо побудувати спрощену модель, яка через умовні ймовірності (задані у SQL або PHP) визначає найбільш релевантні категорії для наступного пошуку на рисунку 2.5. Такі моделі зручно реалізовуються в Laravel через механізми кешування, а також завдяки фоновим завданням, які аналізують накопичену історію.

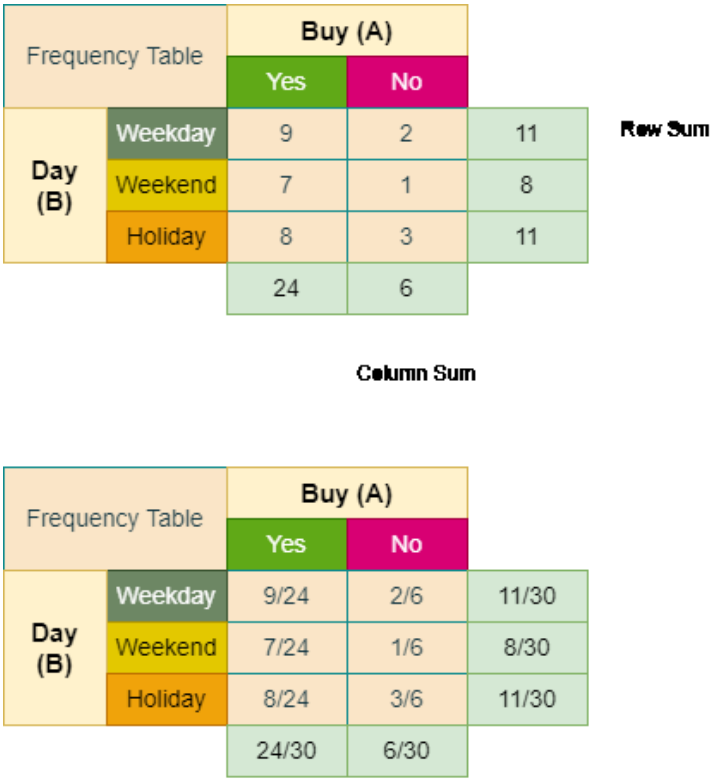


Рисунок 2.5 – Використання кластеризації для покращення алгоритмів пошуку

На прикладі таблиці, ілюструється обчислення апостеріорної ймовірності покупки товару в залежності від типу дня. У верхній таблиці подано частотний розподіл випадків, де ознакою виступає день (Weekday, Weekend, Holiday), а цільовим класом - результат (Buy = Yes / No). Наприклад, у будній день товар купували 9 разів і не купували - 2.

Ці розрахунки можна реалізувати у Laravel за допомогою стандартних SQL-запитів, які агрегують частоти по ознаках. На основі накопиченої статистики у базі даних, PHP-код виконує обчислення апостеріорних ймовірностей для кожного класу і приймає рішення про найбільш вірогідний результат.

В рамках реалізації автозаповнення пошукового рядка ефективно працює метод k-найближчих сусідів (k-NN) у спрощеній формі, який полягає в пошуку найбільш схожих запитів або товарів на основі схожості строк (наприклад, через відстань Левенштейна або косинусну подібність). Обчислення таких метрик може виконуватись на рівні PHP, тоді як частина підготовки даних і вибірки реалізується у MySQL через LIKE-запити або порівняння попередньо збережених токенізованих форм запитів [18].

Окрему роль відіграє кластеризація, яку було описано раніше — зокрема, прості варіанти алгоритму k-середніх. У випадку вебсайту це може бути використано для формування тематичних блоків рекомендованих товарів або для персоналізованого відображення результатів пошуку. Кластеризація за основними характеристиками товарів (ціна, бренд, популярність, категорія) може бути виконана періодично у фонових процесах, а результати зберігаються у вигляді груп у базі даних для швидкого доступу під час запитів.

У контексті подальшого вдосконалення системи підказок та автоматичного доповнення запитів важливим напрямом є інтеграція методів обробки природної мови (Natural Language Processing, NLP). Застосування NLP дає змогу реалізувати інтелектуальний аналіз текстових запитів користувача, що включає виявлення помилок, розпізнавання синонімів, морфологічний аналіз та обробку запитів у вільній формі.

Попри те, що більшість сучасних NLP-бібліотек розробляються переважно для Python (наприклад, spaCy або Hugging Face Transformers), у середовищі PHP також можливе використання таких технологій через зовнішні сервіси або API. Зокрема, у проектах на базі Laravel доцільно впроваджувати NLP-функціонал.

Використання зовнішніх NLP API, таких як TextRazor, MeaningCloud, Aylien або Google Cloud Natural Language API. Ці сервіси надають REST-інтерфейси, що легко інтегруються у Laravel через HTTP-клієнт Http::get() або Http::post() з пакету Illuminate\Support\Facades\Http. Вони забезпечують розпізнавання сутностей, ключових слів, синонімів і граматичних зв’язків.

2.6 Моделювання структури даних для системи пошуку інформації

Ефективність роботи пошукової системи значною мірою залежить від раціонально спроектованої структури бази даних. В умовах використання серверного фреймворку Laravel [21] та СУБД MySQL [20] ключовими аспектами при побудові структури даних є нормалізація, індексація, формат збереження текстових та числових значень, а також забезпечення швидкого доступу до релевантної інформації.

У рамках моделювання бази даних застосовано принципи третьої нормальної форми (3NF), що дозволяє уникнути надлишкового дублювання даних, зменшити ризик аномалій при оновленні та забезпечити логічну цілісність таблиць. Наприклад, інформація про товари, категорії, теги, бренди та інші сутності зберігається у відповідних таблицях із встановленими зовнішніми ключами, що гарантує коректність зв’язків при виконанні запитів.

Особливу увагу приділено індексації полів, які найчастіше залучаються до пошукових запитів. Зокрема, для оптимізації повнотекстового пошуку використано механізми індексації типу FULLTEXT, що дозволяє швидко здійснювати пошук за назвою, описом або ключовими словами. Також використано комбіновані індекси для фільтраційних полів, таких як категорія, ціна, бренд, що значно пришвидшує виконання складних запитів з кількома умовами.

Формат збереження текстової інформації реалізовано через типи VARCHAR або TEXT, залежно від обсягу передбачуваних даних. Для числових та булевих значень застосовуються відповідні типи INT, DECIMAL, BOOLEAN

для забезпечення точності та мінімального використання пам’яті. Особливу увагу також приділено підтримці мультимовності: текстові поля зберігаються у кодуванні utf8mb4, що гарантує підтримку повного набору символів, включно з емодзі та символами розширених мов.

У межах цього розділу доцільно розглянути зображення які продемонстровані в Додатку В. Іллюстрації концептуальної схеми бази даних, що відображає ключові сутності, зв’язки між таблицями та типи асоціацій.

На поданих діаграмах відтворено структурно функціональну модель реляційної бази даних інтернет-магазину, у якій ядрена частина логіки клієнт-орієнтованих операцій поєднана з підсистемами адміністрування та аутентифікації. На першій схемі зображено механізм реєстрації покупців і обробки їхніх замовлень із одночасним забезпеченням системи повідомлень і лояльності через бонусні нарахування. Структури таблиць покупців, замовлень, повідомлень і бонусів прокладені взаємозв’язками «один-до-багатьох» та «багато-до-багатьох», що гарантує фіксацію всіх етапів взаємодії користувача з сервісом та врахування накопичених за дії винагород.

Друга схема розкриває окремий простір адміністративного інтерфейсу, де роль системного адміністратора реалізована через окремі сутності, пов’язані із керуванням каталогом товарів, налаштуваннями платформи та обробкою замовлень із власними бонусними механізмами. Зв’язки між таблицями адмін-користувачів, адмін-замовлень і адмін-бонусів побудовані за схожими принципами, що дозволяє розмежувати права та відповідальність при виконанні адміністративних операцій.

На третій схемі відображено класичний підхід до аутентифікації та профілювання кінцевого користувача: окремі таблиці реєстрації й входу синхронізовані з основним реєстром користувачів, який у свою чергу розширюється механізмами формування списку бажань і відгуків. Ця підсистема забезпечує збереження персональних даних, історії авторизації та суб’єктивних оцінок товарів у єдиній цілісній моделі.

Використання ORM Eloquent у Laravel дає змогу автоматизувати частину роботи з формуванням складних SQL-запитів, зберігаючи гнучкість і масштабованість системи. Визначення реляцій типу one-to-many, many-to-many та polymorphic дозволяє легко формувати зв’язки між сутностями та використовувати їх у фільтраційних запитах без потреби ручного написання об’ємного SQL-коду [19].

2.7 Забезпечення продуктивності та масштабованості системи пошуку інформації на вебсайті

Із зростанням кількості товарів у базі даних та зростаючим числом одночасних користувацьких запитів постає необхідність забезпечення стабільної роботи пошукової системи без втрати швидкодії. Продуктивність пошуку є критично важливою складовою загального користувацького досвіду, особливо в контексті електронної комерції, де затримка в кілька секунд може вплинути на прийняття рішення користувача щодо покупки.

Одним із ключових аспектів підвищення ефективності є оптимізація SQL-запитів. Для цього застосовується попередній аналіз планів виконання запитів, виключення надлишкових об'єднань таблиць (JOIN), а також фільтрація даних ще на рівні бази, щоб мінімізувати обсяги переданої інформації. Зокрема, для реалізації пошуку за назвою товару важливо уникати повнотекстового сканування полів без індексації, оскільки це суттєво уповільнює обробку запитів при великому обсязі даних.

У межах архітектури системи використання індексів є базовим підходом до підвищення продуктивності. Правильно створені індекси на полях, які найчастіше використовуються у пошукових фільтрах (наприклад, name, category_id, price), дозволяють базі даних швидко знаходити відповідні рядки без повного сканування таблиці.

Додатковим засобом прискорення обробки однотипних запитів є кешування. Результати пошукових запитів, що часто повторюються,

зберігаються у тимчасовому сховищі (наприклад, Redis або Memcached). Це дозволяє уникнути надмірного звертання до бази даних у випадках, коли результат для однакового запиту вже обчислено раніше.

З боку фронтенду також використовуються методи оптимізації - зокрема, lazy loading, який дозволяє завантажувати лише ту частину даних, що потрібна на даний момент користувачу. Це скорочує обсяг переданої інформації та прискорює час відгуку інтерфейсу, особливо в мобільному середовищі.

Що стосується масштабованості, система може розвиватися у двох напрямах: вертикальному (шляхом додавання ресурсів на один сервер) та горизонтальному (через розподіл навантаження на декілька серверів). У випадку горизонтального масштабування можлива реалізація кластеризованого підходу до баз даних або окреме виділення серверів для обробки запитів пошуку, кешування і збереження логіки відображення товарів.

2.8 Висновки до другого розділу

У процесі аналізу методів, систем і підходів до проєктування архітектури сайту з пошуковою системою було здійснено комплексне вивчення як загальних принципів побудови вебархітектури, так і специфіки реалізації ефективного пошуку. Розглянуті архітектурні стилі, зокрема монолітна, мікросервісна, SPA та SSR/CSR-моделі, дозволили визначити їх вплив на масштабованість, швидкодію та підтримку системи, що в контексті пошуку має вирішальне значення. Окрему увагу було приділено API-орієнтованому підходу, який забезпечує гнучке інтегрування функціональностей пошуку між клієнтською та серверною частинами.

Аналіз обраного технологічного стеку – PHP (Laravel), Tailwind CSS, MySQL, Ajax, SwiperJS - виявив його придатність для розробки продуктивного сайту з розширеними можливостями пошуку. Laravel надає гнучкі засоби для реалізації логіки запитів, роботи з базою даних та формування RESTful API, тоді як MySQL забезпечує високошвидкісну обробку структурованих даних і гнучке

індексування. Tailwind CSS і SwiperJS дозволяють створити адаптивний, швидкий інтерфейс, що важливо для UX під час роботи з динамічними результатами пошуку.

Узагальнення особливостей методів машинного навчання, потенційно застосовних у рамках проєкту, дозволило виокремити релевантні до обраного стеку технологій підходи – зокрема, обробку запитів через словникову нормалізацію, використання статистичних моделей для побудови фільтрів, а також класифікацію результатів на основі історичних даних користувача.

Проведений аналіз створює концептуальну основу для формування ефективної, масштабованої та технічно узгодженої пошукової системи в межах сучасного вебзастосунку, що відповідає функціональним та продуктивним вимогам.

3 РЕАЛІЗАЦІЯ ВЕБСАЙТУ З ОПТИМІЗОВАНОЮ ПОШУКОВОЮ СИСТЕМОЮ

3.1 Загальна архітектура системи

У контексті розробленого веб-сервісу інтернет-магазину техніки загальна структура системи передбачає поділ на декілька логічно взаємопов’язаних компонентів, кожен з яких виконує чітко визначені функції в межах повного циклу взаємодії користувача з платформою. Архітектура системи реалізована за принципами MVC (Model–View–Controller), що сприяє чіткому розділенню відповідальностей та масштабованості коду. Система включає окремі компоненти для управління товарами, пошуку, відображення детальної інформації, кошика, обробки замовлень, особистого кабінету користувача, а також адміністративної панелі. Ключовим модулем є пошукова система, яка реалізує логіку фільтрації та сортування товарів, враховуючи ключові слова, історію запитів та параметри продукції [22].

3.2 Серверна логіка для реалізації алгоритмів пошуку інформації

У рамках реалізації функціоналу пошуку товарів у вебсистемі інтернет-магазину центральну роль відіграє серверна логіка, побудована на основі фреймворку Laravel. Саме на серверному рівні формується механізм, який забезпечує динамічну обробку користувацьких запитів, ефективну взаємодію з базою даних і видачу релевантних результатів. Запити, що надходять із клієнтської частини, асинхронно передаються через AJAX до серверного маршруту, де ініціюється логіка пошуку [23].

Уся реалізація побудована з урахуванням швидкодії та масштабованості. У відповідному контролері здійснюється пошук на основі часткового збігу введеного тексту з назвами товарів у базі даних. Застосовується оператор LIKE з динамічним формуванням шаблону запиту, що дає змогу обробляти запити в

реальному часі при кожному введеному символі. Цей підхід дозволяє забезпечити функцію автозаповнення, що підвищує зручність взаємодії користувача з інтерфейсом.

Особливу увагу в архітектурі серверної логіки приділено відстеженню дій користувача. У контексті пошуку реалізовано механізм, який дозволяє зберігати історію нещодавніх переглядів. Кожного разу, коли користувач відкриває сторінку конкретного товару, його ідентифікатор реєструється або у локальному сховищі, або передається на сервер для збереження в базі. Завдяки цьому формується індивідуальний перелік останніх переглянутих позицій, який відображається в інтерфейсі як персоналізований блок на рис. 3.1.

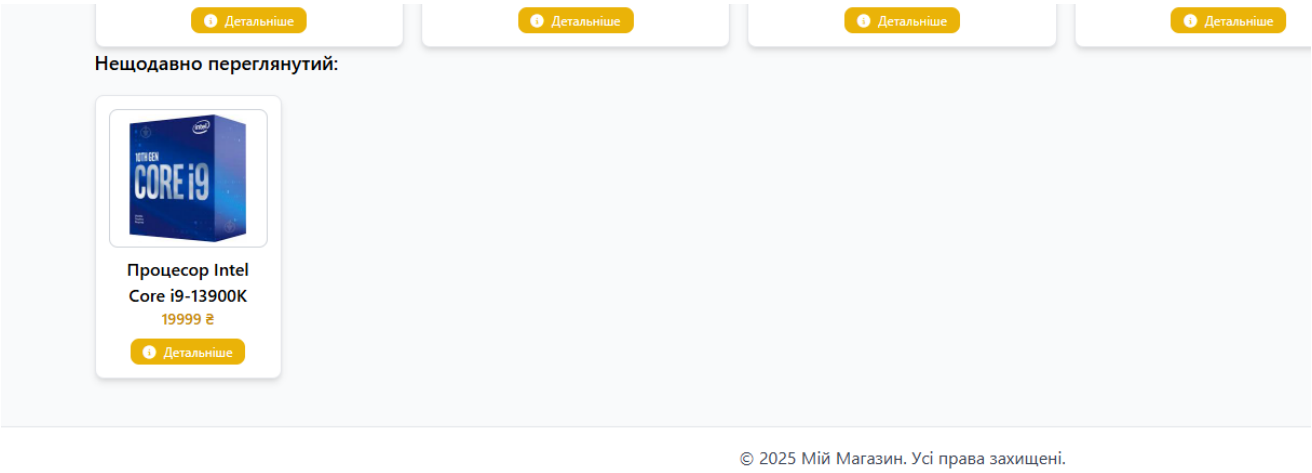


Рисунок 3.1 – Блок останніх переглянутих позицій

Такий підхід дозволяє не тільки покращити користувацький досвід, а й підвищити ймовірність повторної взаємодії з товаром, який вже привернув увагу покупця.

Серверна частина також враховує логіку побудови рейтингових або популярних запитів, що можуть формуватись як агреговані статистичні моделі на основі частоти повторення введених фраз. У перспективі така інформація може бути використана для автоматичного доповнення запитів, підказок або пріоритезації товарів у результатах видачі на рис. 3.2.

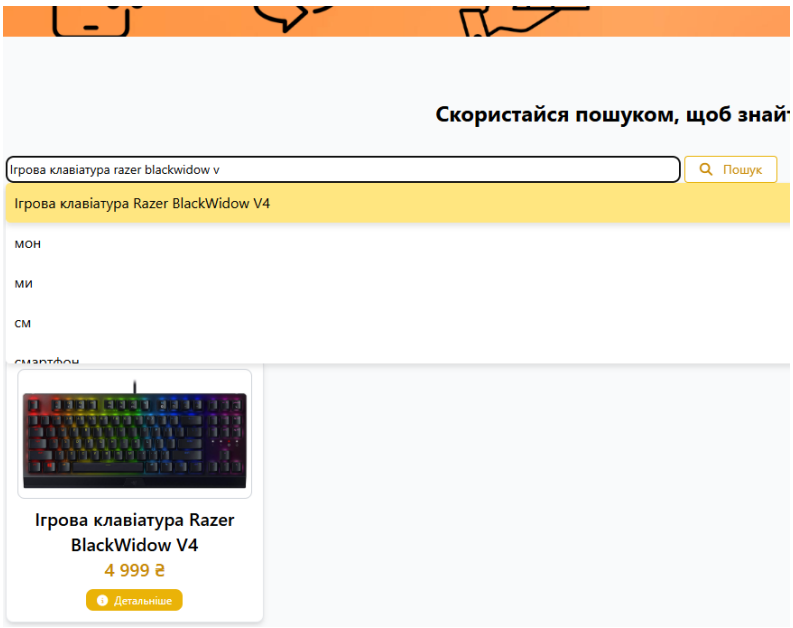


Рисунок 3.2 – Автоматичне доповнення запиту

Автоматичне доповнення запиту - це функція пошуку, яка в режимі реального часу підставляє підказки з назвами товарів, що частково збігаються з уже введеними символами, надсилаючи асинхронний AJAX-запит до бази даних (або кешованого списку) й миттєво відображаючи до п’яти найбільш релевантних варіантів під полем введення.

3.3 Реалізація алгоритмів

3.3.1 Алгоритм пошуку товарів

Пошуковий механізм у даній вебсистемі базується на комунікації між клієнтською частиною, яка ініціює запит, та серверною логікою, яка обробляє запит і формує відповідь. Центральною точкою є маршрут у Laravel, який приймає текст пошукового запиту. Цей маршрут передає дані до контролера, де реалізовано логіку взаємодії з базою даних.

У контролері застосовується метод, що отримує рядок запиту з HTTP-запиту, фільтрує товари з урахуванням часткового збігу та повертає список знайдених результатів у форматі JSON. З боку клієнта JavaScript, з

використанням AJAX-запитів, динамічно передає текст при кожному введенні символу, що дає змогу формувати підказки в режимі реального часу без перезавантаження сторінки.

Крім цього, реалізовано модуль збереження ідентифікаторів останніх переглянутих товарів у локальному сховищі браузера. Коли користувач відкриває сторінку товару, його ID записується у масив, який при завантаженні головної сторінки читається, і відповідні товари завантажуються через додатковий AJAX-запит на сервер. У відповідь сервер повертає повну інформацію про кожен товар, що дозволяє сформувати блок нещодавніх переглядів.

Суттєвим елементом реалізації є запити до бази даних із фільтрацією по полях назви товарів. У SQL-частині використовується оператор пошуку з умовою часткового збігу (через LIKE), а також обмеження кількості результатів, що повертаються, для зменшення навантаження на сервер і пришвидшення відповіді.

Загальна структура пошукового алгоритму орієнтована на швидкість реакції та масштабованість. Завдяки застосуванню Laravel як серверного фреймворку, вдається досягти чіткої маршрутизації, захисту від некоректних запитів, а також гнучкого контролю над структурою відповідей, що значно полегшує інтеграцію з фронтендом. Текст алгоритму подано в лістингу 3.1

Лістинг 3.1 – Search Products Algorithm

```
function renderSuggestions(query) {
searchSuggestionsList.innerHTML = "";
if (query.length === 0) { searchSuggestionsList.style.display =
"none";
return;
}
let matches = products.filter(p =>
p.name.toLowerCase().includes(query.toLowerCase())).slice(0, 5);
matches.forEach(p => {
let li = document.createElement("li");
li.textContent = p.name;
li.addEventListener("click", () => {
searchInput.value = p.name;
searchSuggestionsList.style.display = "none";
```

```
searchBtn.click(); // автоматично натискаємо кнопку після вибору
});
searchSuggestionsList.appendChild(li);
});
searchSuggestionsList.style.display = matches.length ? "block" :
"none";
}
searchInput.addEventListener("focus", function() {
if (history.length > 0) {
searchHistoryList.style.display = "block";
renderHistory();
}
});
searchInput.addEventListener("input", function() {
renderSuggestions(searchInput.value);
});
searchInput.addEventListener("change", function() {
let value = searchInput.value.trim();
if (value && !history.includes(value)) {
history.unshift(value);
history = history.slice(0, 5);
localStorage.setItem("searchHistory", JSON.stringify(history));
}
renderHistory();
});
```

Пошукова система даного вебресурсу побудована на взаємодії між фронтендом, що відповідає за ініціацію пошуку, та серверною частиною, яка реалізована у Laravel [22]. Центральним елементом пошукової логіки на сервері є метод search() у відповідному контролері, наприклад, ProductController. Ця функція обробляє запити, отримані через маршрут, заданий у web.php, і виконує вибірку з бази даних за частковим збігом введеної користувачем назви товару.

Метод search() використовує конструкцію Product::where('name', 'LIKE', '%' . $request->input('query') . '%')->limit(10)->get();, яка дозволяє швидко знаходити товари, назва яких містить введене слово або його частину. Результати повертаються у форматі JSON, що забезпечує простоту інтеграції з клієнтським JavaScript-кодом на рис. 3.3.

З боку фронтенду виконується AJAX-запит за допомогою функції, яка реагує на подію keyup у полі пошуку [23]. В цьому запиті текст передається до серверного маршруту /search, де Laravel обробляє його і повертає список відповідних товарів. У JavaScript-коді результат запиту обробляється й

відображається як інтерактивний список підказок у реальному часі, без перезавантаження сторінки. Основна логіка реалізована у функції fetchSearchResults(query) із викликом fetch('/search?query=' + query) та асинхронним оновленням DOM.

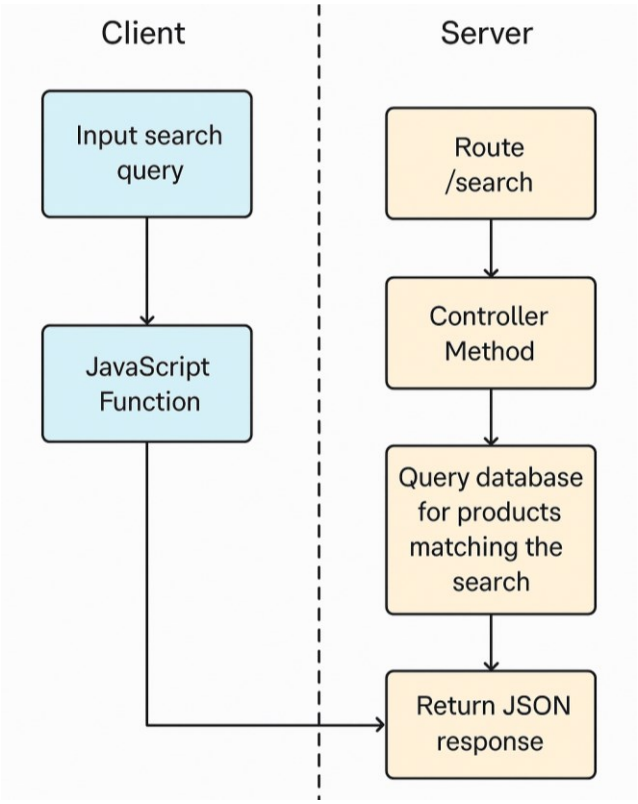


Рисунок 3.3 – Принцип взаємодії клієнта з сервером при пошуку

Щодо збереження останніх переглянутих товарів, ключовою є функція на стороні клієнта saveRecentlyViewed(productId), яка записує ідентифікатори товарів у localStorage. При завантаженні головної сторінки, через виклик функції loadRecentlyViewed(), зчитується масив ID, які передаються до бекенду через маршрут /recent-products. Там контролер ProductController через метод getByIds(Request $request) реалізує вибірку Product::whereIn('id', $request->input('ids'))->get();, формуючи відповідь із детальною інформацією про кожен товар.

3.3.2 Алгоритм збереження історії пошуку

Підсистема збереження історії пошуку та переглядів користувача відіграє ключову роль у підвищенні персоналізації та зручності взаємодії з вебресурсом. У сучасних вебдодатках, таких як інтернет-магазини, інтеграція таких механізмів дозволяє адаптувати інтерфейс до індивідуальних уподобань користувача та підвищити ймовірність повторної взаємодії з платформою. У реалізації даного проєкту використовуються як клієнтські, так і серверні компоненти, що працюють зі сховищами даних локального характеру [24].

Одним із основних сценаріїв, що реалізовано, є збереження останніх п’яти пошукових запитів користувача. Для цього застосовується функція renderHistory(), яка оперує з масивом history, ітеративно формуючи елементи списку <li> та прив’язуючи до кожного обробник події click, що повторно виконує пошук за вибраним терміном [25]. Таким чином реалізовано швидке повторення попередніх запитів без необхідності повторного введення. Текст даного алгоритму можна переглянути в лістингу 3.2.

Лістинг 3.2 – function renderHistory

```
function renderHistory() {
searchHistoryList.innerHTML = "";
history.slice(0, 5).forEach(term => {
let li = document.createElement("li");
li.textContent = term;
li.addEventListener("click", () => {
searchInput.value = term;
searchHistoryList.style.display = "none";
searchBtn.click(); // автоматично натискаємо кнопку після вибору
});
searchHistoryList.appendChild(li);
});
}
```

Крім того, реалізовано збереження нещодавно переглянутих товарів [26]. При кожному переході на сторінку товару, у cookie записується інформація про відповідний об'єкт (його id, name, price, image). Далі, на головній сторінці сайту, здійснюється декодування цих даних з cookie та генерація сітки товарів за

допомогою мови PHP. Перевірка здійснюється на основі масиву $recently_viewed, що динамічно формується з JSON-представлення у cookie. Алгоритм можна переглянути в тексті лістингу 3.3.

Лістинг 3.3 – function renderHistory

```
<!-- Нещодавно переглянуті -->
<!-- Перевірка нещодавно переглянутих товарів -->
<?php
$recently_viewed = isset($_COOKIE['recently_viewed']) ? 
json_decode($_COOKIE['recently_viewed'], true) : [];
?>
<?php if (!empty($recently_viewed)): ?>
<h2 class="text-lg font-semibold mb-4">Нещодавно переглянутий:</h2>
<div class="grid grid-cols-2 sm:grid-cols-3 md:grid-cols-4 lg:grid-
cols-6 xl:grid-cols-8 gap-4">
<?php foreach ($recently_viewed as $viewed_product): ?>
<div class="bg-white p-3 rounded-lg shadow-md border border-gray-
200 text-center">
<div class="w-full h-32 mb-2 border border-gray-300 p-1 rounded-lg 
flex items-center justify-center">
<img src="<?= htmlspecialchars($viewed_product['image']) ?>" 
alt="<?= htmlspecialchars($viewed_product['name']) ?>" 
class="w-full h-full object-contain">
</div>
<h2 class="text-md font-semibold"><?= 
htmlspecialchars($viewed_product['name']) ?></h2>
<p class="text-sm font-semibold text-yellow-600"><?= 
htmlspecialchars($viewed_product['price']) ?> ₴</p>
<a href="product.php?id=<?= htmlspecialchars($viewed_product['id']) 
?>" 
class="inline-block bg-yellow-500 text-white text-xs px-3 py-1 mt-2 
rounded-lg hover:bg-yellow-600 transition">
<i class="fas fa-info-circle mr-1"></i> Детальніше
</a>
</div>
<?php endforeach; ?>
</div>
<?php else: ?>
<p>Немає нещодавно переглянутих товарів.</p>
<?php endif; ?>
</div>
```

Такий підхід дозволяє зберігати дані локально без потреби у серверних запитах до бази даних, що зменшує навантаження та пришвидшує доступ до персоналізованих елементів. Водночас, збереження даних у cookie має обмеження за обсягом і тривалістю життя, тому за потреби масштабування ці

механізми можуть бути перенесені на серверну сторону із використанням баз даних та сесій користувачів.

3.4 Система підказок та історії пошуку

Для підвищення зручності взаємодії з формою пошуку реалізована система підказок, яка працює у тісному зв’язку з автоматичним доповненням запиту. Окрім динамічної видачі релевантних товарів, вона також враховує популярні запити, нещодавно переглянуті позиції або часто шукані категорії, що дозволяє користувачеві швидше орієнтуватися серед можливих варіантів [27].

Історія пошуку ведеться на боці клієнта та зберігається у локальному сховищі браузера. Це забезпечує збереження останніх введених запитів на рисунку 3.4 навіть після оновлення сторінки або повторного входу до сайту.

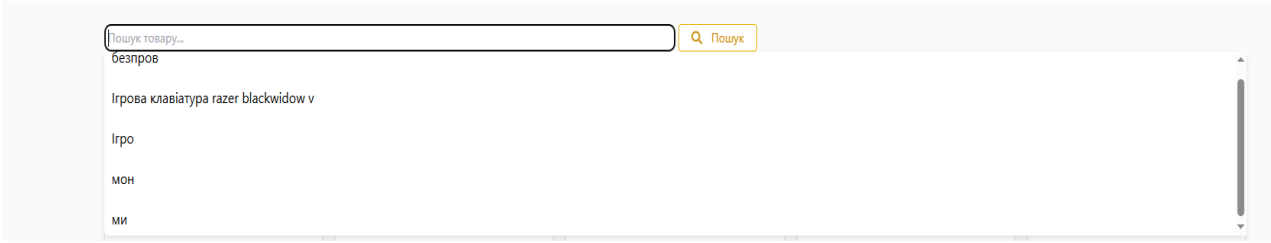


Рисунок 3.4 – Збережені остані введені записи

У разі повторного фокусування на полі пошуку користувачу відображається список останніх запитів, які він здійснював, що скорочує час на повторне введення та сприяє підвищенню залученості.

Крім того, у перспективі можливе розширення функціоналу шляхом інтеграції механізмів персоналізації – наприклад, ранжування підказок з урахуванням поведінки конкретного користувача або сезонності запитів.

3.5 Оптимізація алгоритмів пошуку інформації на вебсайті за допомогою машинного навчання

Зі зростанням обсягу даних, що обробляються в межах сучасних e-commerce систем, зокрема платформ з динамічним асортиментом і великою кількістю однотипних запитів, традиційні алгоритми пошуку на основі часткового текстового збігу почали демонструвати обмежену ефективність.

Такий підхід не дозволяє повною мірою враховувати семантичну близькість запитів, індивідуальні вподобання користувачів або контекст їхньої попередньої активності. У зв'язку з цим у проєкті було реалізовано інтеграцію алгоритмів машинного навчання (Machine Learning, ML), які доповнюють класичну логіку пошуку, рекомендацій і сортування товарів персоналізованими механізмами.

У межах реалізації використано Python-модулі з підтримкою бібліотек scikit-learn, pandas, LightFM та sentence-transformers. Серверна частина ML-системи розгорнута у вигляді окремого Flask-сервісу, який обробляє запити з Laravel-додатку через REST API на рис.3.5.

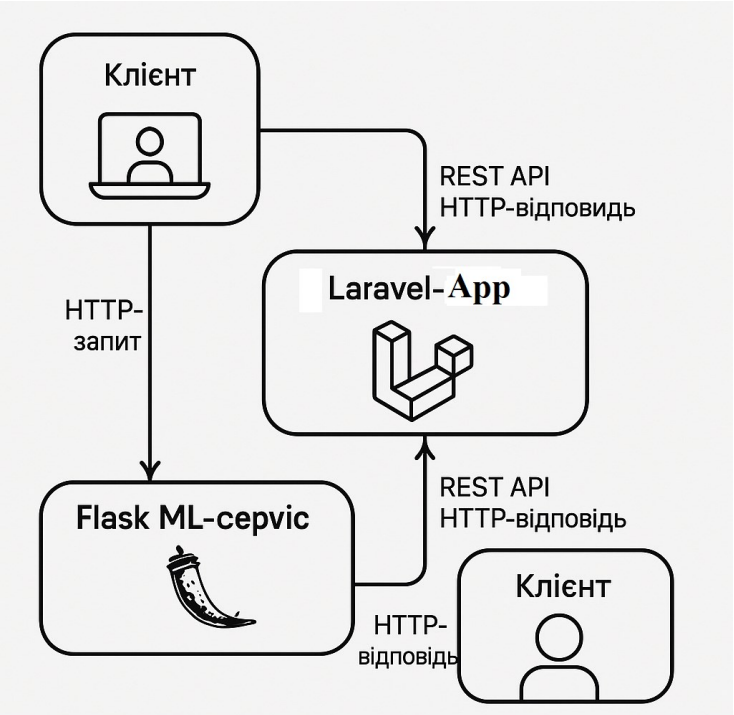


Рисунок 3.5 – Схема інтеграції Laravel з ML-сервісом

Рекомендаційний модуль працює в такий спосіб: після авторизації користувача його ідентифікатор передається на Python-сервіс через запит, сформований у Laravel-контролері. У відповіді система повертає перелік ID товарів, релевантних до поведінкової моделі даного користувача. Отримані дані далі використовуються для побудови блоку "Рекомендоване для вас" на головній сторінці.

Для реалізації кластеризації товарів за подібністю застосовано алгоритм K-Means, який автоматично формує групи на основі таких параметрів, як категорія, діапазон цін, популярність і бренд. Це дозволяє не лише покращити навігацію, а й формувати блоки «Схожі товари» у картці продукту. Модель оновлюється щоденно, після автоматичного витягу нових записів з бази Laravel у форматі CSV, який потім обробляється в Python-скрипті.

Також упроваджено механізм передбачення інтересу користувача до певного товару з використанням логістичної регресії. Для цього враховується комбінація факторів: частота переглядів, попередні взаємодії, категорія товару, середній рейтинг, а також географічне розташування (за IP-адресою). На основі побудованої моделі система повертає ймовірнісну оцінку, яка використовується для динамічного ранжування товарів у результатах пошуку.

На стороні Laravel реалізовано повноцінну інтеграцію з машинним навчальним сервісом за допомогою HTTP-клієнта, який використовується для комунікації із зовнішнім Python-ендпоінтом. У відповідному контролері Laravel сформовано метод, що отримує ідентифікатор авторизованого користувача та надсилає POST-запит на локальний ML-сервер, розгорнутий на Flask. У тілі запиту передається user_id, на основі якого Python-модель формує перелік рекомендованих товарів. У відповідь сервіс повертає масив ідентифікаторів (product_ids), які потім використовуються для отримання відповідних товарів із бази даних через метод whereIn. Таким чином забезпечується безперервна інтеграція між PHP-додатком та машинним навчанням без втрати продуктивності або необхідності дублювання логіки на стороні Laravel, лістинг

коду на стороні Laravel, який здійснює інтеграцію з ML-сервісом описано в лістингу 3.4.

Лістинг 3.4 – Інтеграція Laravel з ML-сервісом

```
use Illuminate\Support\Facades\Http;

public function getRecommendations(Request $request)
{
    $userId = auth()->id();

    $response = Http::post('http://127.0.0.1:5000/api/recommend', [
        'user_id' => $userId,
    ]);

    $recommendedIds = $response->json()['product_ids'];

    return Product::whereIn('id', $recommendedIds)->get();
}
```

Для обробки запитів користувачів, з метою покращення якості пошуку, в проєкті реалізовано модуль, заснований на технологіях обробки природної мови (Natural Language Processing, NLP). Основна його задача полягає у попередньому аналізі вхідного пошукового запиту, що надходить із форми на стороні клієнта, із подальшою семантичною обробкою у спеціалізованому Python-сервісі. У цьому модулі використовується комбінація класичних і сучасних підходів: модель TF-IDF виконує початкову вагову оцінку ключових термінів, а трансформерна архітектура на базі BERT (у реалізації через sentence-transformers) дозволяє зіставити значення запиту із семантичними векторними представленнями товарів у базі.

Для забезпечення інтеграції NLP-моделі з основною системою пошуку, було реалізовано спеціалізований метод у контролері Laravel, який відповідає за попередню обробку користувацького запиту. Цей метод здійснює виклик до зовнішнього Python-сервісу, що працює як мікросервіс із підтримкою трансформерних моделей обробки тексту та описано в лістингу 3.5.

Лістинг 3.5 – Метод що здійснює обробку пошукового запиту

```
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Http;
use App\Models\Product;

public function semanticSearch(Request $request)
{
    $query = $request->input('query');

    $response = Http::post('http://127.0.0.1:5000/api/nlp-process',
[
        'text' => $query,
    ]);

    $terms = $response->json()['terms'];


    $products = Product::where(function ($q) use ($terms) {
        foreach ($terms as $term) {
            $q->orWhere('name', 'LIKE', '%' . $term . '%');
        }
    })->limit(10)->get();

    return response()->json($products);
}
```

На практиці інтеграція реалізована через REST API: під час кожного нового запиту користувача Laravel-контролер передає текст запиту до Python-сервісу, використовуючи метод POST. У відповідь сервіс повертає масив термінів і синонімічних конструкцій, які модуль NLP вважає релевантними до початкової фрази. Далі Laravel динамічно формує SQL-запит із розширеним фільтром, який дозволяє охопити більше релевантних варіантів навіть у разі граматичних помилок або некоректно сформульованих запитів. У поданому фрагменті коду реалізовано метод semanticSearch, який здійснює обробку пошукового запиту шляхом звернення до зовнішнього NLP-сервісу. Текст, введений користувачем, надсилається у вигляді POST-запиту до Python-ендпоінту, що відповідає за семантичний аналіз і формування розширеного списку релевантних термінів. Після отримання відповіді, Laravel-контролер використовує ці терміни для побудови динамічного SQL-запиту до таблиці товарів, реалізуючи пошук з урахуванням синонімів. Отримані результати

повертаються у форматі JSON для подальшого використання на клієнтській стороні.

3.6 Реалізація головної сторінки сайту

Головна сторінка виступає ключовим елементом першого враження про сайт, тому її структура та логіка побудови орієнтовані на максимально швидке залучення користувача до взаємодії з платформою [28].

У верхній частині сторінки розміщено пошуковий рядок з підтримкою автоматичного доповнення запиту та системою підказок. Цей елемент є основним способом взаємодії з каталогом товарів і виконує роль навігаційного ядра.

Одразу під полем пошуку розташовані блоки з персоналізованими або загальними рекомендаціями товарів на рисунку 3.6. Вони формуються залежно від активності користувача (якщо така присутня), або базуються на популярних позиціях, сезонних акціях чи новинках. Таке розташування рекомендацій дозволяє швидко запропонувати користувачеві щось релевантне навіть без потреби введення запиту, що знижує поріг входу у взаємодію з платформою [29].

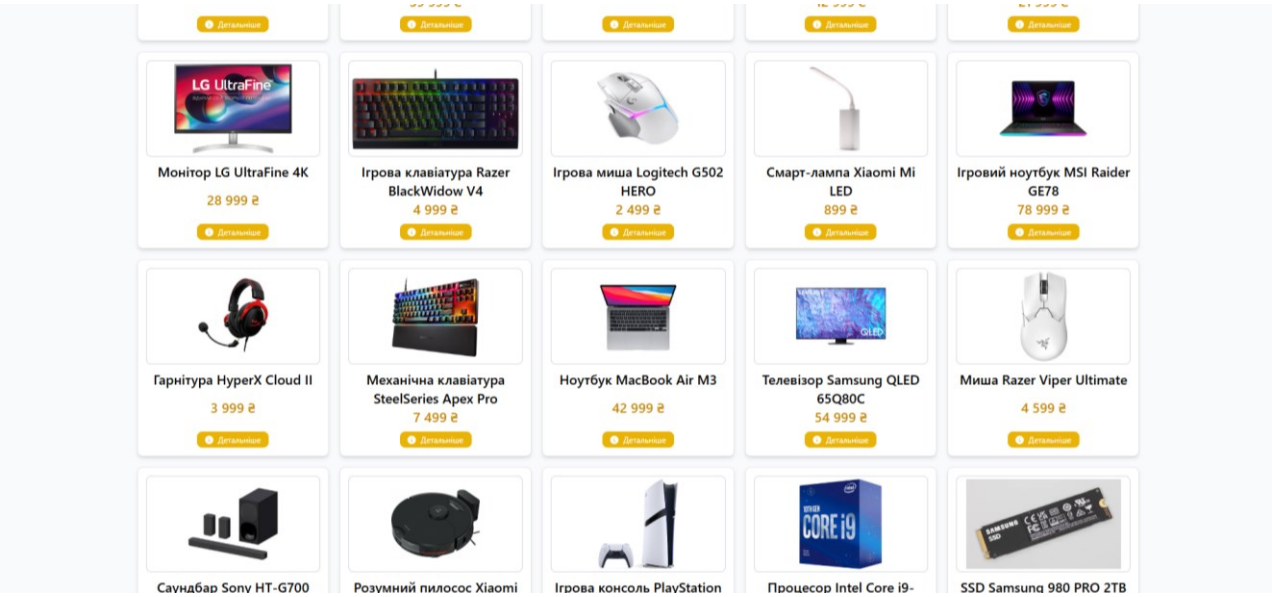


Рисунок 3.6 – Рекомендації для користувачів

Для нових користувачів передбачено коротке вітальне повідомлення з поясненням основного функціоналу сайту та закликом до авторизації або реєстрації. Цей елемент адаптується залежно від статусу сесії та виводиться лише за потреби.

Візуальне оформлення головної сторінки виконано з акцентом на контент, з використанням адаптивної сітки та технологій Tailwind CSS / Bootstrap 5, що забезпечує швидке завантаження та сумісність з більшістю сучасних пристроїв [29].

3.7 Реалізація особистого кабінету користувача

Особистий кабінет користувача є важливим елементом функціональності веб-додатку, що забезпечує персоналізований доступ до облікових даних, історії взаємодій з платформою та додаткових сервісів. Реалізація цього компоненту передбачала створення окремого інтерфейсу, доступного лише після авторизації, з чітким поділом на змістові підрозділи, які дозволяють зручно керувати власною інформацією [30].

Структура особистого кабінету охоплює декілька функціональних розділів. У розділі «Мій профіль» реалізовано відображення основних персональних даних користувача, таких як ім’я, електронна пошта, номер телефону та адреса, а також можливість їх оновлення через відповідні форми із захистом від CSRF-атак. Важливу роль у цьому розділі відіграє забезпечення валідації введених даних як на стороні клієнта, так і на сервері.

Інший розділ – «Історія бонусів» – дає змогу користувачеві переглядати динаміку накопичення та витрати бонусних балів, якщо така система запроваджена. Для цього передбачено автоматичне витягування інформації з відповідної таблиці в базі даних, що містить хронологічний список транзакцій з бонусним рахунком.

У межах розділу «Мої замовлення» користувач має доступ до історії зроблених покупок із зазначенням дати, статусу виконання, вартості та складу

кожного замовлення. Це забезпечує прозорість процесу обслуговування та надає користувачу можливість швидко орієнтуватися у власній активності на сайті. За необхідності реалізовано також механізм повторного оформлення і завантаження супровідних документів (наприклад, електронних чеків або інвойсів).

Останній ключовий розділ - «Список бажань» - дозволяє зберігати обрані товари для подальшого перегляду чи придбання. Така функціональність побудована на асинхронній взаємодії з сервером, що дає змогу додавати або видаляти позиції без перезавантаження сторінки, використовуючи Ajax-запити та динамічне оновлення DOM-елементів.

Усі розділи об’єднані єдиною навігаційною структурою та мають узгоджене стилістичне оформлення, що відповідає загальному дизайну сайту на рис. 3.7.

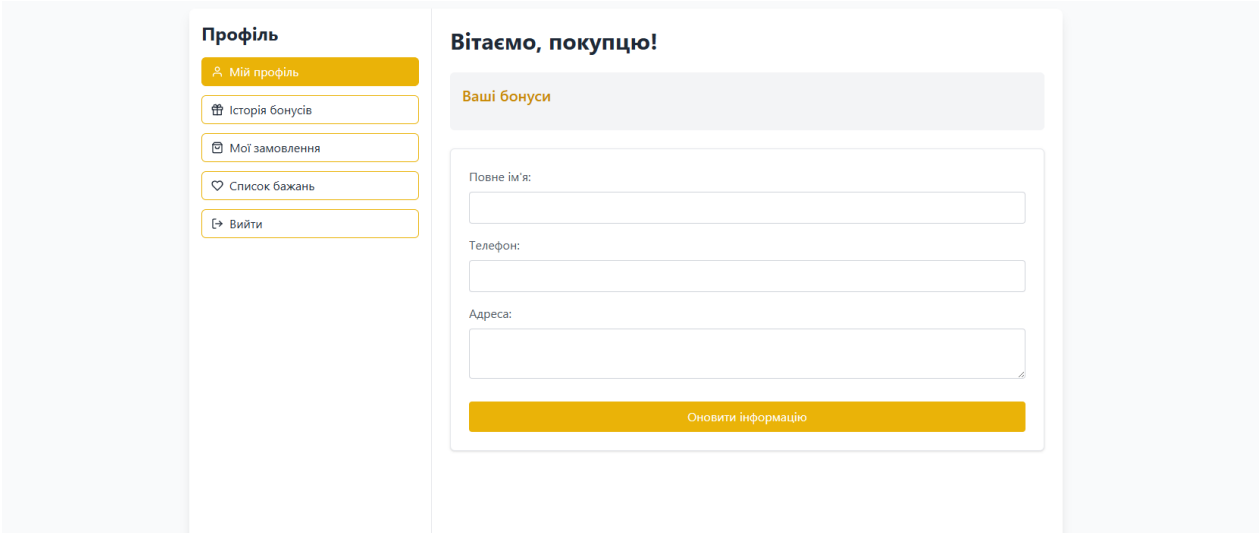


Рисунок 3.7– Особистий кабінет користувача

Особливу увагу при реалізації особистого кабінету було приділено безпеці доступу, захисту даних користувачів, а також оптимізації інтерфейсу для мобільних пристроїв, що забезпечує адаптивність і зручність використання на різних платформах.

3.8 Адміністративна панель вебсайту

Адміністративна панель є ключовим інструментом управління вмістом, користувацькою активністю та параметрами функціонування веб-додатку. Її реалізація забезпечує зручний інтерфейс для адміністраторів, що дозволяє ефективно контролювати роботу системи, здійснювати моніторинг процесів, а також оперативно вносити необхідні зміни до бази даних без потреби прямого втручання в код.

Основною особливістю розробленої адміністративної частини є інтеграція функціональності, яка відповідає ключовим напрямкам керування проєктом. В адміністративному інтерфейсі реалізовано засоби для управління товарами: додавання, редагування, категоризація, встановлення цін, завантаження зображень, зміна статусу наявності тощо. Усі ці дії супроводжуються валідацією даних і передбачають підтримку мультимовності при потребі [31].

Окрему увагу приділено обробці замовлень, де адміністраторам надається доступ до повної інформації про оформлені покупки, включаючи деталі клієнтів, поточний статус замовлення, спосіб оплати та доставки. Передбачено можливість змінювати статуси виконання, надсилати електронні повідомлення користувачам та генерувати супровідну документацію.

Крім того, адміністративна панель містить модулі для перегляду та керування списком зареєстрованих користувачів, їх ролями, активністю та зверненнями. З метою збереження цілісності даних доступ до цих функцій обмежується через рольову систему авторизації, що забезпечує поділ прав між адміністраторами, модераторами та іншими потенційними типами облікових записів.

Також реалізовано аналітичний модуль, який збирає статистику взаємодії користувачів з сайтом: популярність товарів, частоту пошукових запитів, кількість реєстрацій, кількість переглядів і замовлень. Ці дані можуть бути використані для подальшого вдосконалення бізнес-логіки веб-додатку та прийняття стратегічних рішень.

З технічної точки зору, адміністративна панель функціонує як окремий маршрут із захищеним доступом, побудована з використанням сучасних фреймворків для бек- та фронтенду, що забезпечує адаптивність, зручність навігації та високу швидкодію інтерфейсу на рис. 3.8 [32].

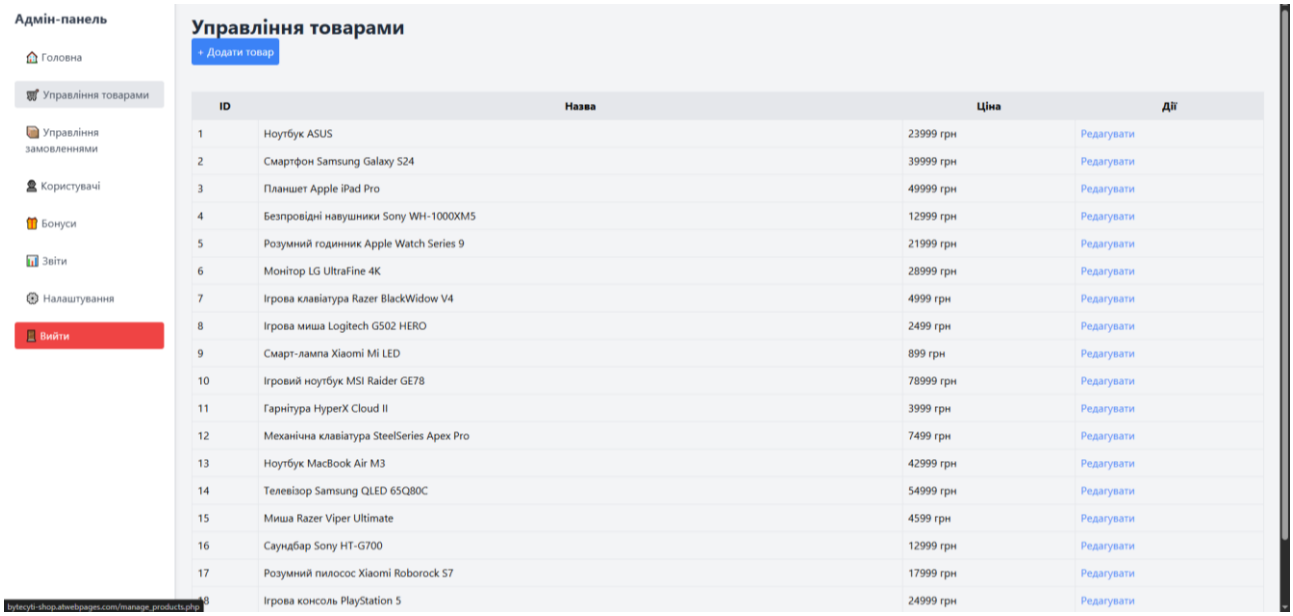


Рисунок 3.8 – Адміністративна панель

Дизайн адміністративного інтерфейсу виконаний у спрощеному, але інформативному стилі, що відповідає принципам usability та дозволяє мінімізувати час навчання персоналу.

3.9 Висновки до третього розділу

У межах даного розділу було здійснено практичну реалізацію основних функціональних компонентів веб-сайту з інтегрованою пошуковою системою. Розробка охопила всі ключові елементи сучасного інтерфейсу взаємодії користувача з торговельною платформою: від стартової сторінки з рекомендаційними блоками до особистого кабінету, адміністративної панелі та системи динамічних підказок.

Особливу увагу приділено архітектурному підходу до побудови веб-додатку, що базується на розподіленій логіці клієнт-серверної взаємодії, використанні асинхронних запитів та гнучкому компонентному поділі. Це дозволило забезпечити масштабованість, високу швидкодію й адаптивність ресурсу для різних категорій користувачів.

Розроблена пошукова система з автоматичним доповненням запиту та історією пошуку створює умови для швидкого доступу до релевантних товарів, що значно покращує користувацький досвід. Впроваджені рекомендаційні блоки та персоналізовані елементи інтерфейсу сприяють зростанню залученості відвідувачів і підвищенню конверсії.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Обов’язкові медичні огляди працівників певних категорій

Порядок проведення медичних оглядів працівників певних категорій, розроблений відповідно до статті 17 Закону України «Про охорону праці», регламентує процедуру проходження попередніх (під час прийняття на роботу) та періодичних (у процесі трудової діяльності) медичних оглядів. У цьому Порядку зокрема передбачено обов’язковість таких оглядів для працівників, які виконують роботу з використанням комп’ютерної техніки, а також для осіб віком до 21 року, для яких встановлено щорічне проходження медичного огляду[33].

Дія зазначеного Порядку поширюється на всі підприємства, установи та організації незалежно від форми власності й виду економічної діяльності, включаючи їхні філії та інші відокремлені підрозділи, а також на фізичних осіб – суб’єктів підприємницької діяльності, які використовують найману працю відповідно до чинного законодавства. Крім того, Порядок стосується осіб, які самостійно забезпечують себе роботою, а також установ і органів, що здійснюють медичні огляди працівників та мають повноваження щодо встановлення діагнозів професійних захворювань[33].

Обов’язкове проходження як попереднього, так і періодичних медичних оглядів передбачено з метою виявлення можливих протипоказань до роботи з комп’ютером, оцінки впливу умов праці на стан здоров’я працівників та забезпечення належного рівня охорони праці проводиться під час прийняття на роботу з метою:

– визначення стану здоров'я працівника і реєстрації вихідних об'єктивних показників здоров'я та можливості виконання без погіршення стану здоров'я професійних обов'язків в умовах дії конкретних шкідливих та небезпечних факторів виробничого середовища і трудового процесу;

– виявлення професійних захворювань (отруєнь), що виникли раніше при роботі на попередніх виробництвах, та попередження виробничо зумовлених і професійних захворювань (отруєнь);

Періодичні медичні огляди проводяться з метою:

– своєчасного виявлення ранніх ознак гострих і хронічних професійних захворювань (отруєнь), загальних та виробничо зумовлених захворювань у працівників;

– забезпечення динамічного спостереження за станом здоров'я працівників в умовах дії шкідливих та небезпечних виробничих факторів і трудового процесу;

– вирішення питання щодо можливості працівника продовжувати роботу в умовах дії конкретних шкідливих та небезпечних виробничих факторів і трудового процесу;

– розробки індивідуальних та групових закладів охорони здоров'я та реабілітаційних заходів працівникам, що віднесені за результатами медичного огляду до групи ризику;

– проведення відповідних оздоровчих заходів.

Роботодавець зобов’язаний забезпечувати безпечні та нешкідливі умови праці відповідно до вимог законодавства України, зокрема Закону України «Про охорону праці», Кодексу законів про працю України, а також державних санітарних норм і правил, встановлених відповідно до ДСТУ [34]. Виконання цих вимог передбачає обов’язкове проходження медичних оглядів, профілактику професійних захворювань та контроль за умовами праці працівників.

Проходження медичного огляду є обов’язковою вимогою трудового законодавства, що прямо передбачено КЗпП:

– роботодавець зобов’язаний за свої кошти організувати проведення попереднього (при прийнятті на роботу) і періодичних (протягом трудової діяльності) медоглядів певних категорій працівників (див. вище) (ст. 169 КЗпП);

– працівник зобов’язаний проходити в установленому порядку попередні та періодичні медогляди (ст. 159 КЗпП).

Відповідно до Кодексу законів про працю України, як роботодавець, так і працівник зобов’язані дотримуватись установлених вимог щодо охорони праці, зокрема проходження медичних оглядів. Згідно зі статтею 21 Закону України № 1645 «Про захист населення від інфекційних хвороб», профілактичні медичні огляди є обов’язковими для працівників окремих професій, виробництв та організацій, дотримання якої надає право:

– роботодавцеві - допустити працівників до виконання їх трудових обов’язків;

– працівникам - виконувати свою роботу.

Враховуючи високий рівень психофізіологічного навантаження, пов’язаного з роботою оператора персонального комп’ютера, необхідність регулярного медичного огляду є обґрунтованою та обов’язковою. Тривале перебування за монітором, статичне положення тіла, інтенсивне зорове та розумове навантаження можуть призводити до професійних захворювань, зокрема порушень зору, опорно-рухового апарату та нервової системи. З метою своєчасного виявлення негативних змін у стані здоров’я, роботодавець повинен забезпечити проходження працівниками відповідних медичних оглядів згідно з вимогами чинного законодавства. У випадку якщо працівник відмовляється проходити медогляд, у роботодавця з’являються підстави для відсторонення його від роботи без збереження зарплати і притягнення до дисциплінарної відповідальності.

4.2 Способи і засоби пожежогасіння

В умовах надзвичайного стану, спричиненого збройною агресією проти України, питання забезпечення пожежної безпеки набуває особливої важливості. Постійні обстріли цивільних об’єктів та об’єктів промислової інфраструктури суттєво підвищують ризик виникнення пожеж, що вимагає належного рівня готовності до їх локалізації та ліквідації. У цьому контексті організація ефективної системи пожежогасіння, а також вибір і застосування відповідних

способів і засобів є критично важливими для мінімізації можливих наслідків та забезпечення безпеки населення й виробничих процесів. Належний рівень організації пожежної безпеки - один із ключових елементів промислової безпеки підприємства. Щоб його досягнути, необхідно вжити низку заходів, зокрема забезпечити території підприємств, будинків, споруд, приміщень, технологічних установок первинними засобами пожежогасіння, які використовують на початку боротьби з пожежами, для їхньої локалізації та ліквідації [35].

Вимога щодо забезпечення первинними засобами пожежогасіння поширюється також на будівлі, споруди та приміщення, обладнані будь-якими типами систем пожежогасіння, пожежної сигналізації або внутрішніми пожежними кран-комплектами.

Засоби пожежогасіння переважно це речовини або їх комплекс, придатні для використання людиною для локалізації і (або) ліквідування пожежі на її початковій стадії (ДСТУ 2272:2006 [55] «Пожежна безпека. Терміни та визначення основних понять»). До первинних засобів пожежогасіння належать:

– вогнегасники;
– ящики з піском;
– бочки з водою;
– покривала з негорючого теплоізоляційного матеріалу;
– пожежні відра, совкові лопати, пожежний інструмент - кирки, сокири, багри, ломи тощо.

Ці засоби використовують на початку боротьби з пожежами, щоб їх локалізувати та ліквідувати. Найефективнішим первинним засобом пожежогасіння є вогнегасник. Порядок розміщенням вогнегасників на об’єкті врегульований Правилами з експлуатації та типовими нормами належності вогнегасників, затвердженими наказом Міністерства внутрішніх справ (далі - МВС) України від 15.01.2018 № 25. Документ поширюється на будинки і приміщення різного призначення, що експлуатуються, підприємства, установи та організації (незалежно від виду їх діяльності і форм власності) та механічні транспортні засоби [37].

Під час вибору засобів пожежогасіння потрібно врахувати фізико-хімічні та пожежонебезпечні властивості горючих речовин і матеріалів, їх взаємодію з вогнегасними речовинами, а також площу виробничих приміщень, відкритих майданчиків та установок.

Відповідальними особами за своєчасне й повне оснащення об'єктів засобами пожежогасіння, їх технічне обслуговування, навчання працівників правилам користування вогнегасниками є власники цих об'єктів або орендарі згідно з договором оренди. До початку експлуатації об'єкти - будинки, споруди, приміщення, технологічні установки - забезпечити первинними засобами пожежогасіння згідно з наказом МВС України від 15.01.2018 № 25. Необхідну кількість таких засобів окремо для кожного поверху та приміщення визначає відповідальний за пожежну безпеку на об'єкті. Переносні вогнегасники:

– навісити на вертикальні конструкції на висоті не більше ніж 1,5 м від рівня підлоги до нижнього торця вогнегасника та на відстані від дверей, достатній, щоб їх повністю відчинити;

– установити у шафи пожежних кран-комплектів, спеціальні тумби, на підставки, що надійно закріплені, на підлозі (якщо дозволяє конструкційне виконання), у пожежні щити (стенди).

Якщо в одному приміщенні розміщені декілька різних за пожежною небезпекою виробництв, не відділених одне від одного протипожежними стінами, то всі ці приміщення потрібно забезпечити вогнегасниками, пожежним інвентарем та іншими видами засобів пожежогасіння за нормами найбільш небезпечного виробництва. У будинках адміністративного та побутового призначення і громадських будинках на кожному поверсі повинні знаходитись не менше двох переносних (порошкових, водопінних або водяних) вогнегасників з масою заряду вогнегасної речовини 5 кг і більше. Окрім того, потрібно передбачити по одному газовому вогнегаснику з величиною заряду вогнегасної речовини 3 кг і більше:

– на 20 м² площі підлоги в офісних приміщеннях з оргтехнікою, коморах, електрощитових, вентиляційних камерах та інших технічних приміщеннях;

– 50 м² площі підлоги в приміщеннях архівів, машинних залів, бібліотек, музеїв.

У приміщеннях, де розміщено комп'ютерну техніку (зокрема в офісних приміщеннях), необхідно передбачити облаштування систем пожежної безпеки відповідно до вимог чинного законодавства України. Згідно наказом МВС України № 1417 від 30.12.2014 року, в таких приміщеннях обов'язково мають бути наявні первинні засоби пожежогасіння, системи автоматичного виявлення та оповіщення про пожежу, а в разі потреби - установки автоматичного пожежогасіння.

Для офісів з комп'ютерним обладнанням рекомендовано використовувати вогнегасники, призначені для гасіння пожеж класу Е, тобто таких, що пов'язані з електрообладнанням під напругою. Вибір засобів пожежогасіння здійснюється відповідно до вимог ДСТУ 4297:2004 [56] «Пожежна безпека. Техніка пожежогасіння. Вогнегасники. Загальні технічні вимоги».

Крім того, при облаштуванні офісних приміщень із комп'ютерами необхідно дотримуватися вимог ДБН В.2.5-56:2014 щодо електробезпеки, зокрема уникати перевантаження електромереж, забезпечити належну вентиляцію для запобігання перегріву обладнання, використовувати негорючі або важкогорючі оздоблювальні матеріали, а також забезпечити вільний доступ до евакуаційних шляхів відповідно до положень ДБН В.1.1-7:2016.

Відповідно до вимог стандарту ДСТУ 3972:2000 [57] «Пожежна безпека», первинні засоби пожежогасіння розміщують на пожежних щитах (стендах) червоного кольору, які встановлюються у виробничих, складських, допоміжних приміщеннях, будівлях, спорудах, а також на території підприємств. Пожежні щити повинні бути встановлені на об'єктах із загальною площею понад 200 м² із нормативним розрахунком – один щит на кожні 5000 м² захищуваної площі. На

таких щитах розміщують ті первинні засоби пожежогасіння, які можуть бути ефективно застосовані у конкретному приміщенні, споруді або установці.[37].

На пожежному щиті вказується порядковий номер після літерного індексу «ПЩ» та номер телефону для виклику пожежно-рятувальних підрозділів. Пожежні щити мають забезпечувати:

– захист вогнегасників від потрапляння прямих сонячних променів;

– захист знімних комплектувальних виробів від використання не за призначенням;

– зручність та оперативність зняття (витягання) закріплених комплектувальних виробів.

Пожежні щити встановлюються так, щоб вони не створювали перешкоди під час евакуації людей у разі пожежі.

Пожежний ручний інструмент на пожежному щиті має періодично обслуговуватись:

– очищати від пилу, бруду та слідів корозії;

– відновлювати фарбування відповідно до стандартів ;

– випрямляти ломи та суцільнометалеві гаки після використання;

– відновлювати потрібні кути загострювання інструмента.

Пожежні покривала використовуються для гасіння невеликих осередків пожеж у разі займання речовин, горіння яких не може відбуватися без доступу повітря.

Відповідно норм ДСТУ 3972-2000 [57] «Пожежна безпека» та ДСТУ 5091-2009 [58] «Покривала пожежні. Загальні технічні умови», встановлено такі нормативи щодо габаритів, конструкції та умов експлуатації первинних засобів пожежогасіння:

– Розмір пожежного покривала має бути не менше ніж 1×1 м, у місцях застосування та зберігання легкозаймистих речовин — 2×1,5 м, горючих речовин - 2×2 м;

– Ящики для піску повинні мати місткість 0,5, 1,0 або 3,0 м³ і бути укомплектованими совковою лопатою. Місткість ящиків для піску, які є

елементом конструкції пожежного стенда, має бути не менше ніж 0,1 м³. Конструкція ящика має забезпечувати зручність діставання піску та унеможливлювати потрапляння сміття й атмосферних опадів;

Бочки з водою встановлюють у виробничих, складських та інших приміщеннях, спорудах у разі відсутності внутрішнього протипожежного водогону та за наявності горючих матеріалів. Їх кількість визначається з розрахунку одна бочка на 250-300 м² захищуваної площі. Місткість бочки для води має бути не менше ніж 0,2 м³. Укомплектовується її пожежним відром місткістю не менше ніж 0,008 м³.

Базуючись на нормах ДСТУ 2272:2006, ДСТУ 4297:2004, ДСТУ 3972:2000 та ДСТУ 5091:2009, забезпечення офісів з комп'ютерною технікою первинними засобами пожежогасіння - обов'язкова вимога. Використання відповідних вогнегасників, покривал, ящиків з піском та дотримання правил їх розміщення дає змогу швидко реагувати на займання й мінімізувати збитки. Це особливо важливо в умовах воєнного стану, коли зростає ризик пожеж через обстріли та перебої в електропостачанні.

4.3 Підвищення стійкості роботи комп'ютеризованих систем в умовах дії електромагнітного імпульсу ядерного вибуху

На даному етапу розвитку технологій комп'ютеризовані системи відіграють ключову роль у різноманітних сферах діяльності – від цивільного управління і промисловості до військової техніки. Забезпечення їх безперебійної та надійної роботи є однією з пріоритетних задач інженерії та інформаційної безпеки. Однак комп'ютерні системи вразливі до зовнішніх електромагнітних впливів, особливо до електромагнітного імпульсу (EMI), що виникає внаслідок ядерного вибуху. Вплив такого імпульсу здатен викликати серйозні пошкодження електронних компонентів, що може призвести до збоїв у роботі обладнання та втрати інформації. У зв'язку з цим дослідження методів підвищення стійкості комп'ютеризованих систем до дії EMI є надзвичайно

актуальним завданням, що має важливе значення для захисту критично важливих інфраструктур [38].

Електромагнітний імпульс – це короткочасний, але інтенсивний сплеск електромагнітного поля, який виникає внаслідок швидких енергетичних процесів, зокрема ядерного вибуху. Такий імпульс характеризується різким зростанням напруженості електричного і магнітного полів у дуже короткий проміжок часу, що створює значні електромагнітні завади для електронних систем.

Ядерний вибух створює електромагнітний імпульс завдяки взаємодії гамма-променів з атмосферою, що призводить до утворення високошвидкісних електронів, які, у свою чергу, породжують потужне електромагнітне поле. Цей імпульс має складну структуру і зазвичай розділяється на кілька фаз - E1, E2 і E3. Фаза E1 характеризується дуже швидким підйомом електричного поля з тривалістю кілька наносекунд, що здатне вивести з ладу напівпровідникові пристрої. Фаза E2 триває довше і схожа на грозові розряди за характеристиками, а фаза E3 є повільною зміною магнітного поля, яка може викликати проблеми в електромережах та інфраструктурі живлення.

Ключовими параметрами електромагнітного імпульсу є амплітуда поля, тривалість імпульсу, спектр частот і географічне поширення дії. Розуміння цих характеристик є необхідною передумовою для розробки ефективних методів захисту комп’ютеризованих систем від шкідливого впливу EMI.

Електромагнітний імпульс (EMI), що виникає під час ядерного вибуху, чинить потужний деструктивний вплив на комп’ютеризовані системи. Його енергія може проникати в електроніку через різні шляхи, спричиняючи тимчасові або постійні відмови обладнання, збої в обробці інформації та втрати даних. Вплив EMI на електронні компоненти має як фізичний, так і функціональний характер, що створює суттєві загрози надійності та безпеці систем.

Основними механізмами впливу EMI на комп’ютеризовані системи є індукція високовольтних імпульсів у провідниках, електростатичний розряд,

нагрівання компонентів і зміщення логічних рівнів у цифрових схемах. Електромагнітний імпульс генерує потужні електричні поля, які проникають у внутрішні ланцюги пристроїв через живлення, сигнальні дроти, антени або безпосередньо через корпус. Це викликає появу імпульсних струмів і напруг, які можуть перевищувати максимально допустимі значення для електронних елементів, що призводить до їх пошкодження або неправильного функціонування.

Крім того, імпульсні завади можуть змінювати логічні стани мікропроцесорів і пам’яті, викликаючи програмні збої або помилкові обробки даних. В окремих випадках це призводить до втрати цілісності інформації або повної зупинки роботи системи.

До найбільш вразливих компонентів належать:

– Мікросхеми та інтегральні схеми – особливо напівпровідникові пристрої з низькою напругою живлення та високою щільністю елементів, які легко виходять з ладу під впливом імпульсних перенапруг.

– Джерела живлення – можуть бути пошкоджені внаслідок раптових високовольтних імпульсів, що призводить до збоїв у подачі живлення та виходу обладнання з ладу.

– Комунікаційні лінії (кабелі, антени) – слугують каналами проникнення імпульсних завад у внутрішні системи, підсилюючи їхній вплив.

– Приклади наслідків впливу EMI на роботу систем

– Наслідки впливу електромагнітного імпульсу на комп’ютеризовані системи можуть бути різноманітними:

– Тимчасові збої у роботі пристроїв, що проявляються у випадкових перезавантаженнях, помилках у програмному забезпеченні або збоях в обробці даних.

– Постійні пошкодження електронних компонентів, що призводять до необхідності їх заміни або ремонту.

– Втрата критично важливої інформації через порушення роботи пам’яті або каналів передачі даних.

– Повна непрацездатність систем управління або контролю, що може мати катастрофічні наслідки в залежності від сфери застосування (наприклад, в авіації, енергетиці або військовій техніці).

Методи підвищення стійкості комп'ютеризованих систем до електромагнітного імпульсу (EMI) базуються на комплексному підході, який включає як пасивні, так і активні засоби захисту, спрямовані на мінімізацію впливу імпульсних завад на електронні компоненти та забезпечення безперебійної роботи систем. Одним із фундаментальних методів є екранізація та захист корпусів, що передбачає створення провідних або феромагнітних оболонок, які екранують внутрішні елементи від зовнішніх електромагнітних полів за допомогою явища екранування (екрануюча здатність оцінюється коефіцієнтом ослаблення електромагнітного поля). Додатково застосовують фільтри та подавлювачі імпульсних завад, які монтуються на входах живлення та сигнальних лініях для згладжування високочастотних імпульсів, зменшуючи їх амплітуду до безпечних рівнів [38].

Проектування захищених схем включає використання спеціальних архітектур із ізоляцією критичних елементів, застосування компонентів із підвищеною стійкістю до перенапруг, наприклад, з технологією Silicon on Insulator (SOI) або із захистом від статичної електрики (ESD), а також оптимізацію топології друкованих плат з урахуванням мінімізації індуктивності та ємності, що знижує сприйнятливість до імпульсів. Резервування і дублювання систем полягає у створенні резервних каналів обробки інформації або дублюванні критичних модулів, що дозволяє забезпечити безперервність функціонування навіть у випадку часткових відмов, викликаних EMI. Особливу увагу приділяють захисту джерел живлення та комунікаційних ліній шляхом встановлення спеціалізованих захисних пристроїв (наприклад, варисторів, газових розрядників, фільтрів LC), що перешкоджають проходженню імпульсних перенапруг. Для стандартизації вимог до стійкості застосовуються нормативні документи, серед яких MIL-STD-461 і MIL-STD-464 – це військові стандарти, які визначають методи випробувань та критерії допустимого рівня

електромагнітних завад, що мають дотримуватися при проектуванні електронних систем із підвищеною стійкістю до EMI. Використання цих методів у комплексі забезпечує значне підвищення надійності комп'ютеризованих систем у складних електромагнітних умовах [39].

Існуючі технології та рішення захисту від електромагнітного імпульсу (EMI) включають широкий спектр апаратних і програмних засобів, спрямованих на запобігання або мінімізацію негативного впливу імпульсних завад на комп'ютеризовані системи. До основних технологій належать багатошарова екранізація корпусів із застосуванням металевих або феромагнітних матеріалів, інтеграція активних фільтрів і подавлювачів імпульсів у лініях живлення та зв'язку, а також розробка мікросхем із підвищеним рівнем стійкості до перенапруг, зокрема на основі кремнієвих пластин із ізоляцією Silicon on Insulator (SOI). Відомі системи з підвищеною стійкістю до EMI включають військову авіоніку, системи управління ракетними комплексами, а також критичні енергетичні та телекомунікаційні об'єкти, у яких застосовано комплексний захист із резервуванням каналів передачі та живлення, мультиступеневою екранізацією і посиленими фільтрами.

Практичні рекомендації для розробників комп'ютеризованих систем зосереджені на початковому етапі проектування, де необхідно враховувати можливі електромагнітні загрози, впроваджувати багаторівневі системи захисту, ретельно підбирати компоненти з підтвердженою стійкістю до EMI, а також проводити випробування відповідно до стандартів MIL-STD-461 та MIL-STD-464. Крім того, важливо впроваджувати процедури регулярного моніторингу та технічного обслуговування систем захисту, що дозволяє підтримувати їх ефективність у довгостроковій перспективі. Такий комплексний підхід забезпечує високу надійність і безпеку комп'ютеризованих систем у складних електромагнітних умовах.

4.4 Висновки до четвертого розділу

В розділі розглянуто ключові питання, що стосуються захисту працівників та безпечного функціонування підприємства в різних умовах.

Описано порядок проведення обов’язкових медичних оглядів для працівників окремих категорій, зокрема тих, хто працює в умовах підвищеного ризику або виконує завдання, пов’язані з високим навантаженням на зір, нервову систему та опорно-руховий апарат.

Розглянуто основні способи і засоби пожежогасіння, які повинні бути наявні на підприємстві, зокрема вогнегасники, пожежні щити, сигналізація та системи автоматичного пожежогасіння, а також правила їх використання.

Також, описано заходи, які підвищують стійкість комп’ютеризованих систем у разі впливу електромагнітного імпульсу, що виникає під час ядерного вибуху – включаючи екранування обладнання, заземлення, використання фільтрів та дублювання критичних елементів систем.

ВИСНОВКИ

У кваліфікаційній роботі здійснено комплексне дослідження та практичну реалізацію вебсайту з системою пошуку інормації, що базується на сучасних інформаційних технологіях і принципах машинного навчання. Робота складається з трьох розділів, у яких послідовно розкрито теоретичні, проектні та прикладні аспекти теми.

В першому розділі кваліфікаційної роботи:

– Наведено огляд сучасного стану пошукових систем у вебсередовищі

– Розглянуто принципи функціонування інформаційного пошуку на вебсайті

– Висвітлено основні класифікації методів фільтрації та рекомендацій

– Проаналізовано технології машинного навчання в контексті пошуку інформації на вебсайті

– Досліджено вплив архітектури системи на якість результатів пошуку

– Обґрунтовано вибір інструментів реалізації вебдодатку з системою пошуку інформації

– Сформовано вимоги до системи з урахуванням користувацьких потреб.

В другому розділі кваліфікаційної роботи:

– Описано функціональну структуру та архітектуру розробленої системи

– Досліджено механізми зберігання, індексації та обробки запитів користувачів вебсайту

– Наведено порівняльний опис підходів до реалізації пошуку й рекомендацій

В третьому розділі кваліфікаційної роботи:

– Розроблено основні компоненти веб-додатку, включно з пошуковою формою, системою підказок, рекомендаціями, особистим кабінетом і адміністративною панеллю

– Запропоновано інтерактивну структуру інтерфейсу з адаптивною логікою

– Спроєктовано моделі бази даних і логіку обробки асинхронних запитів

– Протестовано працездатність системи в умовах реального використання

У розділі «Охорона праці та безпека в надзвичайних ситуаціях» проаналізовано нормативно-правову базу щодо обов’язкових медичних оглядів працівників окремих категорій, зокрема осіб, які працюють із комп’ютерною технікою. Описано особливості організації заходів безпеки у разі виникнення надзвичайних ситуацій техногенного та природного характеру. Розглянуто підходи до забезпечення стійкої роботи комп’ютеризованих систем в умовах дії електромагнітного імпульсу ядерного вибуху та запропоновано відповідні технічні й організаційні рішення.

ПЕРЕЛІК ДЖЕРЕЛ

1. Любомир МАТІЙЧУК, Володимир ГОТОВИЧ, Віталій БОНАР (Статті. ТНТУ ім.І.Пулюя), Порівняння ефективності методів некерованого машинного навчання для виявлення аномалій в obd2 даних.

2. ЛВ Волинець, НА Гарматюк, ВА Готович (Статті. ТНТУ ім.І.Пулюя), Великі за обсягом набори біомедичних даних та машинне навчання.

3. ВА Готович, АВ Мачужак (Матеріали XI Міжнародної науково-практичної конференції молодих учених та студентів „ Актуальні задачі сучасних технологій “), Застосування методології CI/CD для автоматизації процесів тестування та розгортання програмного забезпечення.

4. Grigorii Shymchuk, Iaroslav Lytvynenko, Roman Hromyak, Sergii Lytvynenko, Volodymyr Hotovych, Gas Consumption Forecasting Using Machine Learning Methods and Taking Into Account Climatic Indicators.

5. Що таке CRO, і оптимізація конверсії, URL: https://netpeak.net/uk/blog/shcho-take-cro-i-yak-optimizatsiya-konversii-opomagae-zbil-shiti-pributok/ (дата звернення: 20.05.2025)

6. Створення сайтів, URL: https://surl.li/nbooti (дата звернення: 20.05.2025)

7. Алгоритми ранжування, URL: https://rubika.agency/ua/glossary/algoritm-ranzhiruvannja (дата звернення: 20.05.2025)

8. Ранжування сайту, URL: https://wezom.com.ua/ua/blog/ranzhirovanie (дата звернення: 20.05.2025)

9. Обробка природної мови (NLP), URL: https://metinvest.digital/ua/page/1052 (дата звернення: 20.05.2025)

10. Machine Learning, ML, URL: https://www.it.ua/knowledge-base/technology-innovation/machine-learning (дата звернення: 20.05.2025)

11. Пошукові Системи: історія, розвиток та сучасні тенденції, URL: https://webmate.ua/poshukovi-sistemi-istoriya-rozvitok-ta-suchasni-tendenciyi (дата звернення: 20.05.2025).

12. Структура файлів подань у Laravel, URL: https://code.mu/ru/php/framework/laravel/book/prime/views/files-structure/ (дата звернення: 20.05.2025)..

13. Eloquent: API Resources, URL: https://laravel.com/docs/12.x/eloquent-resources (дата звернення: 20.05.2025).

14. Understanding the MVC Архітектура в Laravel: A Comprehensive Guide, URL: https://fkrihnif.medium.com/understanding-the-mvc-architecture-in-laravel-a-comprehensive-guide-8f620cc139b6 (дата звернення: 20.05.2025)

15. tailwindcss URL: https://tailwindcss.com/ (дата звернення: 20.05.2025)

16. Swiper.js URL: https://refine.dev/blog/swiper-js/ (дата звернення: 20.05.2025)

17. TF-IDF (term frequency-inverse document frequency), URL: https://surl.li/yomtsu. (дата звернення: 20.05.2025)

18. Оптимізація SQL, URL: https://dou.ua/forums/topic/47863/ // Дата доступу до ресурсу: 20.05.20225р.

19. Eloquent ORM від Laravel, URL: https://surl.li/xtlhiz (дата звернення: 20.05.2025)

20. СУБД MySQL URL: http://www.znannya.org/?view=PHP_SUBD_mysql (дата звернення: 20.05.2025)

21. Переваги фрейморку Laravel для веб-розробки, URL: https://wezom.com.ua/ua/blog/17-preimuschestv-ispolzovanija-laravel-v-it-industrii (дата звернення: 20.05.2025)

22. Архітектура сайту URL: режим доступу до ресурсу: https://coi.ua/blog/Cbc/Website-Structures-How-to-Choose-the-Best-Option-for-Your-Web-Project/ (дата звернення: 20.05.2025).

23. AJAX для новачків URL: https://habr.com/ru/articles/14246/ (дата звернення: 20.05.2025)

24. Сухий О. Л., Міленін В. М., Тарадайнік В. М. Алгоритми пошуку в інформаційних системах.

25. Пошукова оптимізація, URL: https://apollon.guru/seo/trendy-seo-prosuvannya/ (дата звернення: 20.05.2025).

26. Рекомендації на сайті як збільшення конверсії, URL: https://esputnik.com/uk/blog/tovarni-rekomendaciyi-na-sajti-instrument-zbilshennya-konversiyi (дата звернення: 20.05.2025).

27. Просування сайту пошуковими підказками, URL: https://seok.ua/uk/prodvizhenie-sajta-poiskovymi-podskazkami.html (дата звернення: 20.05.2025).

28. Головна сторінка: її роль для сайту та правильне оформлення, URL: https://surl.lu/nnckhl. (дата звернення: 20.05.2025)

29. Як правильно оформити головну сторінку сайту, URL: https://hostiq.ua/blog/ukr/main-page/ (дата звернення: 20.05.2025)

30. Розробка особистого кабінету, URL:https://voll.com.ua/uk/dorabotka-sajta/razrabotka-lichnogo-kabineta (дата звернення: 20.05.2025).

31. Що таке адмінка сайту, URL: https://ukrline.com.ua/ua/blog/108666-4.php (дата звернення: 20.05.2025).

32. Як організовано розробку адмінки, URL: https://surl.li/leajyy. (дата звернення: 20.05.2025).

33. Обов'язкові медичні огляди працівників певних категорій, URL: https://protocol.ua/ua/pro_ohoronu_pratsi_stattya_17/ (дата звернення: 20.05.2025)

34. Порядок проведення медичних оглядів працівників певних категорій, URL: https://surl.li/kjrjii. (дата звернення: 20.05.2025).

35. Способи припинення процесу горіння, URL: https://pidru4niki.com/1460091739057/bzhd/sposobi_zasobi_pozhezhogasinnya (дата звернення: 20.05.2025).

36. Способи та засоби пожежогасіння, URL: https://buklib.net/books/32189/ (дата звернення: 20.05.2025).

37. Комплекс заходів, спрямованих на ліквідацію пожежі, URL: https://surl.lu/xqrhii (дата звернення: 20.05.2025)

38. Електромагнітний імпульс ядерного вибуху, URL: https://surl.li/fvhqcv (дата звернення: 20.05.2025)

39. Вимоги безпеки під час роботи на ПК, URL: Режим доступу до ресурсу: https://sites.google.com/site/ohoronapraci44/33-vimogi- bezpeki-pid-cas-roboti-na-pk. (дата звернення: 20.05.2025).

40. Т.А. Григорова. Дослідження методів машинного навчаня для пошуку інформації / Т.А. Григорова, В.П. Ляшенко, О.О. Москаленко (Статті. Кременчуцький національний університет ім. М. Остроградського)

41. Ден Гасфілд. Integer Linear Programming in Computational and Systems Biology: An Entry-Level Text and Course / Ден Гасфілд, 2019 – с. 428 (Лінійне програмування).

42. Структури даних та алгоритми, URL: https://studfile.net/preview/10025806/page:21/. (дата звернення: 20.05.2025).

43. К.М. Онищенко. Аналіз методів обробки природної мови / К.М. Онищенко, Я.І. Данієль, Р.О. Каманєв.

44. Charu C. Aggarwal. Recommender Systems: The Textbook / Charu C. Aggarwal, 2016 – с. 518 (Recommender Systems).

45. Antonio Mele. Django 5 By Example - Fifth Edition: Build powerful and reliable Python web applications from scratch 5th ed. Edition / Antonio Mele, 2024 - с.820.

46. Структури даних та алгоритми, URL: https://studfile.net/preview/10025806/page:21/. (дата звернення: 20.05.2025)

47. К.М. Онищенко. Аналіз методів обробки природної мови / К.М. Онищенко, Я.І. Данієль, Р.О. Каманєв.

48. Charu C. Aggarwal. Recommender Systems: The Textbook / Charu C. Aggarwal, 2016 – с. 518 (Recommender Systems).

49. Cathy Tanimura. SQL for Data Analysis. Advanced Techniques for Transforming Data into Insights. 1st Ed. / Cathy Tanimura, 2021 – с. 350.

50. MVC Architectural Pattern - Web Platform, URL: https://doka.guide/tools/architecture-mvc/ (дата звернення: 20.05.2025).

51. Тернопільський національний технічний університет імені Івана Пулюя, URL:https://tntu.edu.ua/?p=uk/main (дата звернення: 20.05.2025)

52. PHP docs, URL: https://www.php.net/docs.php URL: https://tntu.edu.ua/?p=uk/main (дата звернення: 20.05.2025).

53. Ajax - MDN Web Docs Glossary: Definitions of Web-related, URL: https://developer.mozilla.org/en-US/docs/Glossary/AJAX

54. laravel docs, URL: https://laravel.com/docs/11.x (дата звернення: 20.05.2025).

55. ДСТУ 2272:2006. Охорона праці. Терміни та визначення основних понять. — [На заміну ДСТУ 2272-93 ; чинний від 2007-01-01]. — Вид. офіц. — Київ : Держспоживстандарт України, 2006. — 16 с.

56. ДСТУ 4297:2004. Пожежна безпека. Вогнегасники. Загальні технічні вимоги, методи випробувань. — [Чинний від 2005-07-01]. — Вид. офіц. — Київ : Держспоживстандарт України, 2005. — 14 с.

57. ДСТУ 3972:2000. Безпека дорожнього руху. Технічні засоби регулювання дорожнього руху. Загальні технічні умови. — [Чинний від 2001-01-01]. — Вид. офіц. — Київ : Держстандарт України, 2001. — 18 с.

58. ДСТУ 5091-2009. Пластмаси. Методи випробування на твердість. — [Чинний від 2010-01-01]. — Вид. офіц. — Київ : ДП «УкрНДНЦ», 2010. — 12 с.

ДОДАТКИ

Тези першої конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний технічний університет імені Івана Пулюя (Україна)
Університет імені П'єра і Марії Кюрі (Франція)
Маріборський університет (Словенія)
Технічний університет у Кошице (Словаччина)
Вільнюський технічний університет ім. Гедимінаса (Литва)
Наукове товариство ім. Т.Шевченка

АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ

Збірник
тез доповідей

XIII Міжнародної науково-практичної конференції молодих учених та студентів
11-12 грудня 2024 року

УКРАЇНА
ТЕРНОПІЛЬ – 2024

38. **М.І. Богуцький**
ОПТИМІЗАЦІЯ АЛГОРИТМІВ ПОШУКУ ІНФОРМАЦІЇ НА ВЕБ-САЙТІ З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ
39. **М.О. Скоробогата, Л.П. Дмитроца**
ВПЛИВ ГЕНЕРАТИВНОГО ШІ НА МАРКЕТИНГОВІ КОМУНІКАЦІЇ КОМПАНІЇ "TECHNOVAAPP"
40. **Микитишин А.Г., Чуревич Б.В., Дильовий О.О.**
ПРОМИСЛОВІ СИСТЕМИ ЕНЕРГОМОНІТОРИНГУ НА ОСНОВІ ІНТЕРНЕТУ РЕЧЕЙ
41. **Н. М. Головецький, І.О. Баран**
ОГЛЯД ІСНУЮЧИХ ПРОЕКТІВ LPWAN
42. **Н. Сороківська, В.Яцишин**
КОМП'ЮТЕРИЗОВАНІ СИСТЕМИ МОНІТОРИНГУ БІОРІЗНОМАНІТТЯ З ВБУДОВАНИМИ ІНТЕЛЕКТУАЛЬНИМИ СЕРВІСАМИ
43. **Н.А. Шевченко, Г.В. Шимчук, У.А. Гарматюк**
ОПТИМІЗАЦІЯ МЕТРИК МАРШРУТИЗАЦІЇ ДЛЯ ЗАБЕЗПЕЧЕННЯ СТІЙКОСТІ ТА НАДІЙНОСТІ IP-МЕРЕЖ
44. **Н.А. Шевченко, Г.В. Шимчук, У.А. Гарматюк**
ІНТЕГРАЦІЯ ТЕХНОЛОГІЇ МЕРЕЖЕВОЇ ВІРТУАЛІЗАЦІЇ VRF У БАГАТОКОЛІЙНУ МАРШРУТИЗАЦІЮ
45. **Н.М. Топольніцький, Р.М. Небесний**
ЗАСТОСУВАННЯ GEOMETRY NODES В ПРОГРАМНОМУ ПАКЕТІ BLENDER ДЛЯ ЗАДАЧ ПРОЦЕДУРНОГО МОДЕЛЮВАННЯ
46. **Н.М. Топольніцький, Р.М. Небесний**
ЗАСТОСУВАННЯ GEOMETRY NODES В ПРОГРАМНОМУ ПАКЕТІ BLENDER ДЛЯ ЗАДАЧ ПРОЦЕДУРНОГО МОДЕЛЮВАННЯ
47. **Назар Осідак, Олександр Цвігун**
ЗАСТОСУВАННЯ ШІ ДЛЯ ЗАДАЧ АВТОМАТИЗОВАНОГО ТЕСТУВАННЯ
48. **О.А. Заяць**
МЕТОДИ ТА ІНСТРУМЕНТИ АНАЛІЗУ І КЛАСИФІКАЦІЇ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ
49. **О.А. Саган, К.Б. Швирло**
АНАЛІЗ SQL-ІН'ЄКЦІЙ:ТИПИ АТАК ТА МЕТОДИ ЗАХИСТУ
50. **О.В. Палка**
ІНФОРМАЦІЙНІ ПАНЕЛІ ЯК ІНФОРМАЦІЙНО-ТЕХНОЛОГІЧНИЙ ІНСТРУМЕНТ ДЛЯ ВІДСТЕЖЕННЯ KPI У РОЗУМНОМУ МІСТІ
51. **О.В. Смолій; А.Г. Микитишин; І.В. Булич**
ОПТИМІЗАЦІЯ ВКЛЮЧЕННЯ НАВАНТАЖЕННЯ ПРИ РЕЗЕРВНОМУ ЖИВЛЕННІ НА МАЛОМУ ВИРОБНИЧОМУ ПІДПРИЄМСТВІ ТА ЙОГО ІНТЕГРАЦІЯ З ТЕХНОЛОГІЧНИМ ПРОЦЕСОМ
52. **О.В. Шкварок; А.О. Курило; І.Р. Козбур; Ю.Б. Капаціла**
РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ РОЗУМНОГО БУДИНКУ НА БАЗІ ІНТЕРНЕТУ РЕЧЕЙ ТА БЕЗДРОТОВИХ МЕРЕЖ
53. **О.Д. Пакон, Р.М. Небесний**
ОСНОВНІ ВИКЛИКИ ДЛЯ СИСТЕМ КЕРУВАННЯ МОБІЛЬНИМИ ПРИСТРОЯМИ
54. **О.Д. Пакон, Р.М. Небесний**
АНАЛІЗ ЕФЕКТИВНОСТІ ВПРОВАДЖЕННЯ MDM СИСТЕМ ДЛЯ КОРПОРАТИВНОГО СЕРЕДОВИЩА

УДК 004.4

М.І. Богуцький
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ОПТИМІЗАЦІЯ АЛГОРИТМІВ ПОШУКУ ІНФОРМАЦІЇ НА ВЕБ-САЙТІ З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ

M.I. Boguzkiy

OPTIMIZATION OF INFORMATION SEARCH ALGORITHMS ON A WEBSITE USING MACHINE LEARNING METHODS

Для вирішення задачі оптимізації алгоритмів пошуку інформації застосовуються різні підходи та стратегії, які залежать від специфіки веб-ресурсу та вимог користувачів. Одним із можливих шляхів є застосування методів машинного навчання, таких як глибинне навчання та обробка природної мови (NLP) (рис. 1), що допомагають краще розуміти значення запитів, враховувати контекст, аналізувати наміри користувачів і формувати більш релевантні результати [1].

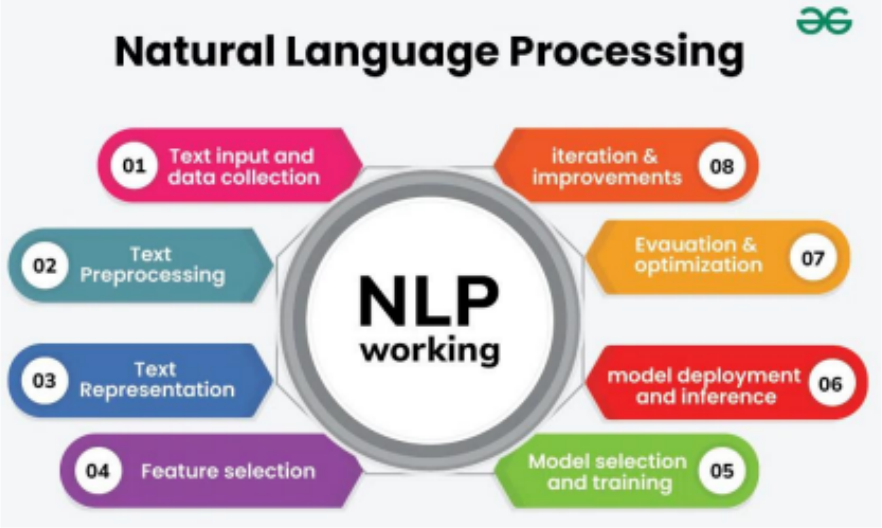


Рисунок 1. Обробка природньої мови

Можна також інтегрувати алгоритми кластеризації для групування схожих запитів або застосовувати моделі ранжування на основі даних користувацької поведінки, щоб забезпечити персоналізацію пошуку.

Сучасні алгоритми пошуку інформації на веб-сайтах є основою для забезпечення швидкого доступу до даних і формування позитивного користувацького досвіду. Одним із найпростіших методів є прямий пошук, який підходить для невеликих наборів даних, але має низьку ефективність для складних запитів і великих обсягів інформації. Бінарний пошук значно швидший, проте його використання обмежується впорядкованими даними, що вимагає попереднього сортування [2].

Популярним підходом є використання інвертованих індексів, які забезпечують швидкий пошук у текстових базах, але створення та зберігання таких індексів потребує

значних ресурсів [3]. Повнотекстовий пошук дозволяє реалізовувати складні запити, проте ефективність цього методу залежить від правильного ранжування результатів. Алгоритми ранжування, такі як PageRank, аналізують метадані й зв'язки між елементами, забезпечуючи релевантність, але вимагають постійного вдосконалення для актуалізації результатів.

Методи обробки природної мови значно покращують пошук, дозволяючи враховувати контекст і синтаксис запиту, але висока складність і ресурсоємність є їх недоліками. Загалом, хоча відомі алгоритми мають свої переваги, вони стикаються з низкою обмежень, що відкриває простір для їх подальшої оптимізації [4].

Інструменти для реалізації таких підходів включають бібліотеки машинного навчання, такі як TensorFlow або PyTorch, які дозволяють створювати та тренувати моделі для пошуку та аналізу даних. Також можна використовувати спеціалізовані пошукові платформи з підтримкою AI, наприклад Elasticsearch, що забезпечують високу швидкість обробки запитів і налаштування алгоритмів ранжування. Іншим перспективним методом є застосування recommendation systems, які базуються на колаборативній фільтрації та допомагають прогнозувати інтереси користувачів, а також семантичний пошук, що дозволяє знаходити інформацію за змістом, а не лише за ключовими словами [5]. Рекомендаційні системи – це технологія, що надає користувачам персоналізовані рекомендації на основі аналізу даних. Їх основна мета — покращити взаємодію користувача із сервісом, пропонуючи релевантний контент, зокрема товари, фільми, книги, музику чи послуги. Для досягнення цієї мети використовуються різні підходи, серед яких колаборативна фільтрація, що базується на поведінці схожих користувачів, та контентна фільтрація, яка аналізує характеристики об'єктів, що вже зацікавили користувача [5].

Сучасні системи часто поєднують ці методи, створюючи гібридні рішення, які є більш гнучкими і точними. Додатково враховуються зовнішні фактори, такі як місце розташування, час доби чи особливості пристрою. Для глибшого аналізу даних дедалі частіше використовують методи глибокого навчання, що дають змогу знаходити приховані закономірності.

Рекомендаційні системи широко застосовуються в електронній комерції, стрімінгових сервісах, соціальних мережах, освіті, туризмі та інших галузях. Вони допомагають користувачам швидше знаходити потрібне, водночас підвищуючи їхній інтерес до платформи. Проте розробка таких систем стикається з викликами, як-от необхідність персоналізації, обробка великих обсягів даних та проблема холодного старту, коли відсутня інформація про нових користувачів чи продукти. Розробка такої системи потребує комплексного підходу, що поєднує машинне навчання, інженерію даних та ефективне використання інструментів для обробки великих обсягів даних.

Література

1. Т.А. Григорова. Дослідження методів машинного навчаня для пошуку інформації / Т.А. Григорова, В.П. Ляшенко, О.О. Москаленко (Статті. Кременчуцький національний університет ім. М. Остроградського)
2. Ден Гасфілд. Integer Linear Programming in Computational and Systems Biology: An Entry-Level Text and Course / Ден Гасфілд, 2019 – с. 428 (Лінійне програмування).
3. Структури даних та алгоритми [Електронний ресурс] : https://studfile.net/preview/10025806/page:21/. Доступ до ресурсу: 24.11.2024
4. К.М. Онищенко. Аналіз методів обробки природної мови / К.М. Онищенко, Я.І. Даніель, Р.О. Каманєв.
5. Charu C. Aggarwal. Recommender Systems: The Textbook / Charu C. Aggarwal, 2016 – с. 518 (Recommender Systems).

Тези другої конференції

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18–19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 004.4
М. І. Богуцький
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ОПТИМІЗАЦІЯ АЛГОРИТМІВ ПОШУКУ ІНФОРМАЦІЇ НА ВЕБ-САЙТІ З ВИКОРИСТАННЯМ МЕТОДІВ МАШИННОГО НАВЧАННЯ

UDC 004.4
M. I. Boguzkiy

OPTIMIZATION OF INFORMATION SEARCH ALGORITHMS ON A WEBSITE USING MACHINE LEARNING METHODS

Обсяг інформації доступної в мережі Інтернет, зростає з неймовірною швидкістю, що створює нові виклики для ефективного пошуку даних та їх персоналізації. Розробка веб-сайту, який слугуватиме платформою для впровадження і тестування таких технологій, є важливим кроком у вивченні їх можливостей, особливостей та недоліків. Існує багато рішень для розробки такого проекту, одним з популярних та доволі оптимальних є Python та його фреймворки. Фреймворки Django та Flask дозволяють створити потужний та стабільний бекенд, здатний обробляти запити користувачів, виконувати пошукові операції та генерувати рекомендації в режимі реального часу [1].

Інтеграція пошукових алгоритмів на сайті забезпечується використанням індексованих баз даних, що значно скорочує час виконання запитів.[4] Наприклад, використання бібліотек як Whoosh або Elasticsearch дозволяє додатково оптимізувати процеси пошуку завдяки потужним інструментам для аналізу тексту, ранжування та врахування контексту запиту. Для підвищення якості пошуку впроваджуються алгоритми обробки природної мови (NLP), які допомагають розпізнавати значення запитів, аналізувати їх синтаксис та враховувати індивідуальні особливості користувачів [3].

Для зберігання даних про користувачів, товари, історію пошуку та взаємодії використовується база даних. Дані можуть бути попередньо структуровані та індексовані для забезпечення максимальної продуктивності пошукових алгоритмів [5]. Для інтеграції моделей машинного навчання та обробки даних використовуються бібліотеки, такі як TensorFlow або PyTorch, які дозволяють створювати гнучкі рішення для персоналізації пошуку та рекомендацій. Використовуючи технології обробки природної мови, запит обробляється для врахування контексту, після чого формується запит до бази даних [2].

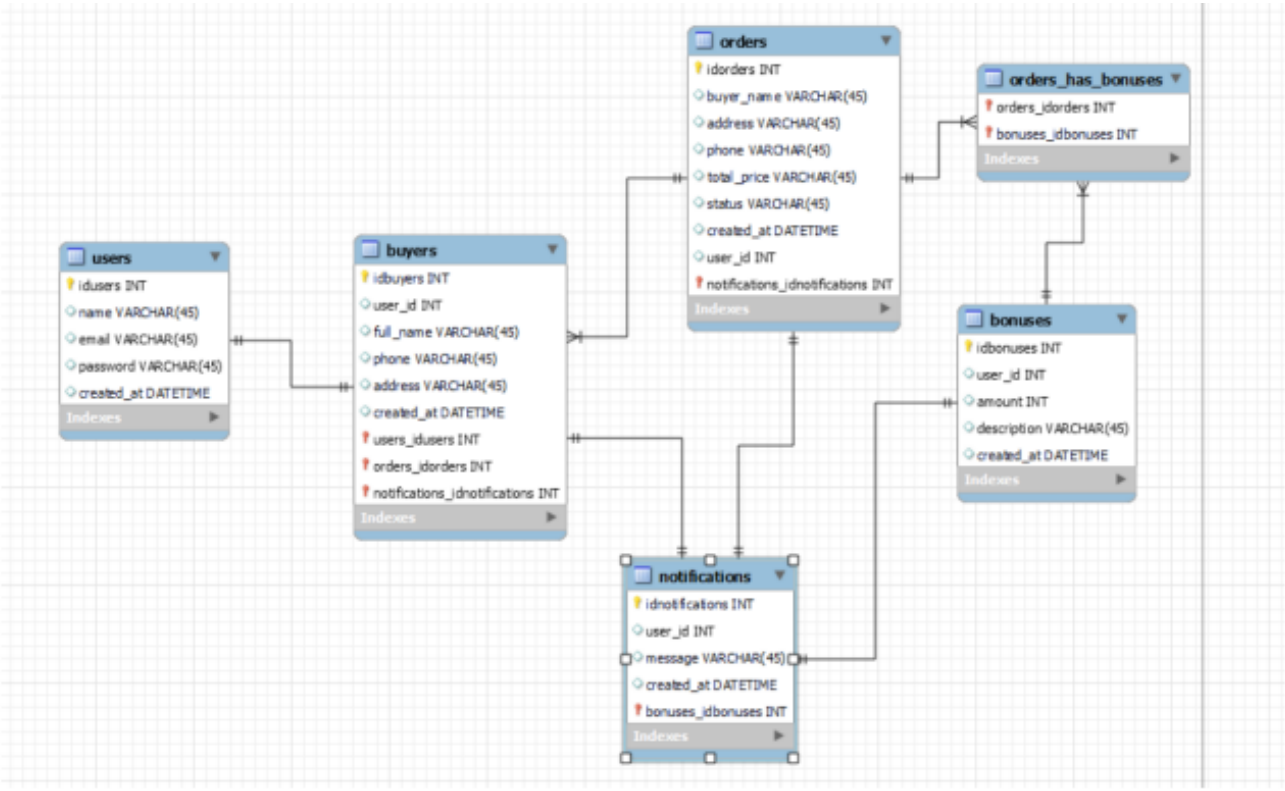
Перевагою такого підходу є його гнучкість і масштабованість. Веб-сайт може адаптуватися до зростаючих вимог користувачів, підтримувати інтеграцію нових алгоритмів і обробляти великі обсяги даних. Інтеграція рекомендаційних систем дозволяє пропонувати користувачам товари чи контент, які максимально відповідають їхнім інтересам, підвищуючи задоволеність та залученість. Водночас існують певні недоліки, які потрібно враховувати під час розробки. Складність інтеграції алгоритмів, вимагає значних обчислювальних ресурсів, що може збільшити витрати на підтримку системи. Проблема конфіденційності даних, адже персоналізовані рекомендації потребують збору й обробки даних про користувачів. Крім того, система може зіткнутися з проблемою холодного старту, коли бракує даних для нових користувачів чи об'єктів, що впливає на якість пошуку та рекомендацій.

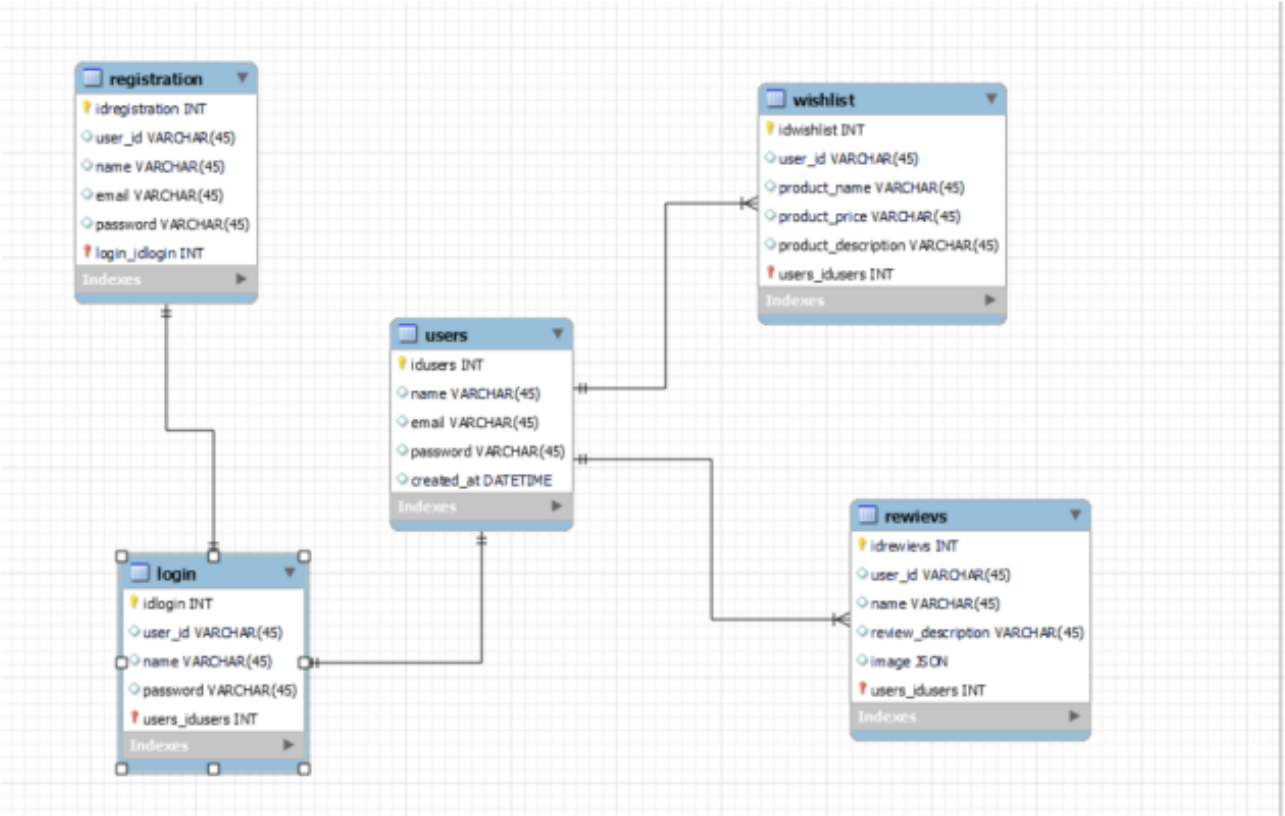
Література

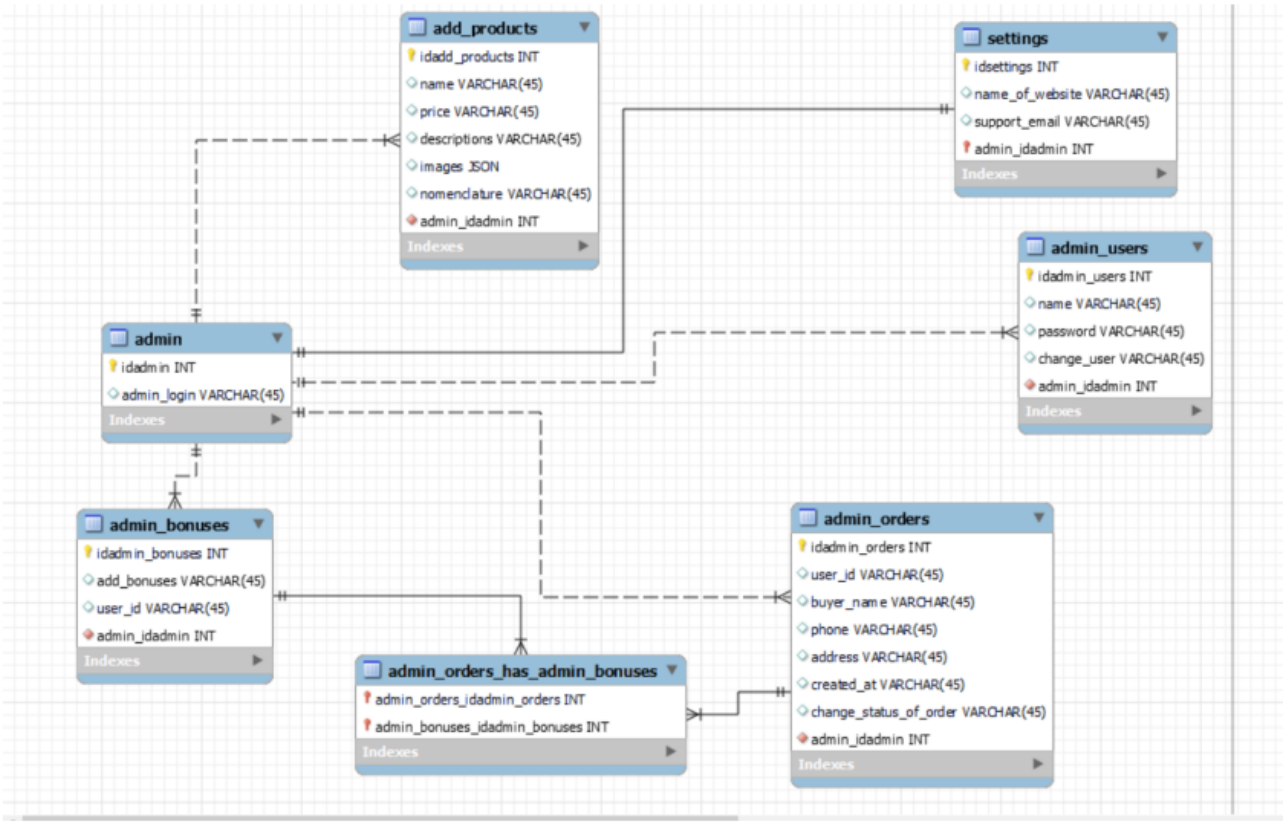
1. Antonio Mele. Django 5 By Example - Fifth Edition: Build powerful and reliable Python web applications from scratch 5th ed. Edition / Antonio Mele, 2024 - с.820.
2. Структури даних та алгоритми [Електронний ресурс] : https://studfile.net/preview/10025806/page:21/. Доступ до ресурсу: 04.12.2024
3. К.М. Онищенко. Аналіз методів обробки природної мови / К.М. Онищенко, Я.І. Даніель, Р.О. Каманєв.
4. Charu C. Aggarwal. Recommender Systems: The Textbook / Charu C. Aggarwal, 2016 – с. 518 (Recommender Systems).
5. Cathy Tanimura. SQL for Data Analysis. Advanced Techniques for Transforming Data into Insights. 1st Ed. / Cathy Tanimura, 2021 – с. 350.

Додаток В

Концептуальна схема БД






add_products
idadd_products INT
name VARCHAR(45)
price VARCHAR(45)
descriptions VARCHAR(45)
images JSON
nomenclature VARCHAR(45)
admin_idadmin INT
Indexes
settings
idsettings INT
name_of_website VARCHAR(45)
support_email VARCHAR(45)
admin_idadmin INT
Indexes
admin_users
idadmin_users INT
name VARCHAR(45)
password VARCHAR(45)
change_user VARCHAR(45)
admin_idadmin INT
Indexes
admin
idadmin INT
admin_login VARCHAR(45)
Indexes
admin_bonuses
idadmin_bonuses INT
add_bonuses VARCHAR(45)
user_id VARCHAR(45)
admin_idadmin INT
Indexes
admin_orders
idadmin_orders INT
user_id VARCHAR(45)
buyer_name VARCHAR(45)
phone VARCHAR(45)
address VARCHAR(45)
created_at VARCHAR(45)
change_status_of_order VARCHAR(45)
admin_idadmin INT
Indexes
admin_orders_has_admin_bonuses
admin_orders_idadmin_orders INT
admin_bonuses_idadmin_bonuses INT
Indexes