МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Розроблення програмних засобів для аналізу

та дослідження активності користувачів на

WEB-ресурсі

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки

Освітня програма Комп'ютерні науки

Виконав: студент групи МгІТ-1-22

Курочка Владислав Олександрович

Науковий керівник к.т.н.,доц. .

Резанова Вікторія Георгіївна

Рецензент Щербань В.Ю.

Київ 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

Факультет мехатроніки та комп'ютерних технологій

Кафедра комп'ютерних наук

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри КН

Володимир ЩЕРБАНЬ.

"12" 09 2023 року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Курочці Владислав Олександровичу

1. Тема кваліфікаційної роботи Розроблення програмних засобів для аналізу та дослідження активності користувачів на WEB-ресурсі

науковий керівник роботи Резанова Вікторія Георгіївна, доц.,к.т.н.

затверджені наказом КНУТД від "_12_"вересня_2023 року №_210-уч_

2. Вихідні дані до роботи: Розробки кафедри комп'ютерних наук; рекомендована література, додатки.

3. Зміст дипломної роботи: Вступ; Розділ 1 ТЕОРЕТИЧНИЙ; РОЗДІЛ 2 АРХІТЕКТУРНІ РІШЕННЯ РОЗРОБКИ ДОДАТКІВ ТА АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ; РОЗДІЛ 3 РОЗРОБКА ВЕБ-ДОДАТКУ; Висновки; Список літератури; ДОДАТОК А програмний код; ДОДАТОК Б Стаття: ДОДАТОК В Презентація..

4. Дата видачі завдання 1 вересня 2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапу кваліфікаційної роботи	Орієнтовний термін виконання	Примітка про виконання
1	Вступ	15.09.2023	
2	Розділ 1. ТЕОРЕТИЧНИЙ; РОЗДІЛ	20.09.2023	
3	Розділ 2. АРХІТЕКТУРНІ РІШЕННЯ РОЗРОБКИ ДОДАТКІВ ТА АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ	30.09.2023	
4	Розділ 3. РОЗРОБКА ВЕБ-ДОДАТКУ	10.10.2023	
5	Висновки	25.10.2023	
6	Оформлення (чистовий варіант)	1.11.2023	
7	Подача кваліфікаційної роботи (проєкту) науковому керівнику для відгуку (за 14 днів дозахисту)	4.11.2023	
8	Подача кваліфікаційної роботи (проєкту) для рецензування (за 12 днів дозахисту)	6.11.2023	
9	Перевірка кваліфікаційної роботи (проєкту) на наявність ознак плагіату (за 10 днів до захисту)	8.11.2023	1% - 10%
10	Подання кваліфікаційної роботи (проєкту) на завідувачу кафедри (за 7днів до захисту)	10.11.2023	

З завданням ознайомлений:

Студент(-ка) _____ (підпис) Курочка Владислав

Науковий керівник _____ (підпис) Резанова Вікторія

АНОТАЦІЯ

Курочка В.О. Розроблення програмних засобів для аналізу та дослідження активності користувачів на WEB-ресурсі

Кваліфікаційна робота за спеціальністю 122 - «Комп'ютерні науки» – Київський національний університет технологій та дизайну, Київ, 2023 рік.

Кваліфікаційна робота присвячена розробленню веб-додатку рекомендацій на основі поведінкових даних користувачів з використанням основних методів фільтрації рекомендаційних систем.

В роботі були проаналізовані системи рекомендаційні системи, їх види, основні методи фільтрації. Також був проведений аналіз відмінностей рекомендаційних систем, їх переваги та недоліки. Далі приведена структура додатку, вибір технологій та обґрунтування їх вибору. Після, описується реалізація цієї системи.

Ключові слова: рекомендаційні системи, веб-додаток, JavaScript, Node JS, методи фільтрації, розробка, методи, Python, Express JS.

ANNOTATION

Development of a web-based recommendation service based on the analysis of user behavioral data.

Qualification work in specialty 122 - "Computer Science" - Kyiv National University of Technology and Design, Kyiv, 2023.

The qualification work is devoted to the development of a web-based recommendation application based on user behavioral data using basic filtering methods of recommender systems.

The work analyzes recommender systems, their types, and basic filtering methods. We also analyzed the differences between recommender systems, their advantages and disadvantages. Next, the structure of the application, the choice of technologies and the rationale for their choice are presented. Finally, the implementation of the system is described.

Keywords: recommender systems, web application, JavaScript, Node JS, filtering methods, development, methods, Python, Express JS.

Зміст

ВСТУП

Актуальність теми. В сучасному світі, де технології та Інтернет постійно розвиваються, веб-додатки займають важливе місце у повсякденному житті, особливо в аспектах, як особистісного зростання, так і зручності у щоденних справах. Онлайн магазини, як приклад, економлять час, дозволяючи робити покупки не виходячи з дому. Розвиток веб-технологій надає компаніям великі переваги, дозволяючи їм займати провідні позиції на ринку завдяки створенню якісних веб-ресурсів.

Швидке завантаження веб-сторінок є критично важливим для успіху сайту, адже користувачі звикли до миттєвого доступу до інформації. Тому оптимізація веб-сторінок стає пріоритетом. Сучасні веб-розробки та стратегії конкуренції, включаючи алгоритми рекомендацій товарів, є важливими для забезпечення унікального користувацького досвіду. Ці системи аналізують взаємодію користувачів з платформою, пропонуючи товари, які відповідають їхнім індивідуальним інтересам.

Професійно створені веб-сайти здатні миттєво збирати та обробляти дані, дозволяючи онлайн-платформам швидко адаптуватися до змін у поведінці користувачів. Вони впроваджують технології штучного інтелекту та машинного навчання для оптимізації процесів рекомендацій, що підвищує точність та результативність рекомендацій.

Цей підхід не тільки стимулює ріст продажів через персоналізовані пропозиції, але й зміцнює відданість клієнтів, спонукаючи їх до повторних відвідин платформи. Така стратегія підвищує конкурентоспроможність, пропонуючи унікальний індивідуальний досвід кожному користувачу.

Мета роботи. У рамках кваліфікаційної роботи, фокус ставиться на створенні сучасного веб-додатку з використанням передових фронтенд-технологій та методика проектування, а також використання основних методів фільтрації для рекомендації товарів, які відповідають перевагам користувача.

Завдання роботи. Проект спрямований на створення удосконаленої версії алгоритму рекомендації, заснованого на сингулярному розкладі матриць, та конструювання передового веб-додатка із застосуванням Angular фреймворк. Це має підсилити комерційний потенціал магазину.

Об’єкт та предмет дослідження. Об'єктом дослідження є веб-додаток електронної комерції. Предметом є розробка алгоритму рекомендацій, спрямованого на підвищення ефективності та задоволеності користувачів.

Методи та засоби дослідження. Аналіз актуальних літературних джерел та дослідженні передових практик у галузі веб-розробки. Основна увага приділяється вивченню архітектури сучасних веб-додатків, з особливим акцентом на розробку клієнтської частини та ефективність моделей даних. Це включає детальне дослідження логіки створення інтерфейсів користувача, а також аналіз способів збору, оброблення та використання даних для оптимізації функціональності веб-додатків.

Наукова новизна та практичне значення отриманих результатів.

Наукова новизна виявляється у розробці вдосконаленого алгоритму рекомендацій, що враховує сучасні тенденції веб-додатків та користувацьких даних. Практичне значення полягає у підвищенні ефективності онлайн-магазинів та забезпеченні більш персоналізованого досвіду для користувачів, що може бути використане в різних бізнес-сферах.

Результати магістерської роботи були апробовані в наукової статті: Розробка веб-сервісу рекомендацій на основі аналізу поведінкових даних користувачів / Резанова, В. Г., Курочка В.О /

https://er.knutd.edu.ua/handle/123456789/24826

РОЗДІЛ 1 ТЕОРЕТИЧНИЙ

Розвиток рекомендаційних систем має довгу історію, яка почалася ще до появи Інтернету, але значно активізувалася з розвитком цифрових технологій. Давайте розглянемо ключові моменти у розвитку цих систем.

Спочатку рекомендаційні системи зустрічалися в бібліотеках, де бібліотекарі пропонували книги читачам на основі їхніх інтересів. Ці ранні системи використовували базові алгоритми, що орієнтувалися на категорії та жанри книг.

У 1990-х, з розширенням Інтернету, почався аналіз контенту для рекомендацій. Системи почали аналізувати характеристики об'єктів, як-от ключові слова та авторів, для виявлення та порівняння схожих об'єктів.

У кінці 1990-х з'явилися алгоритми колаборативної фільтрації. Вони аналізували взаємодію користувачів з об'єктами, такими як оцінки чи покупки, для виявлення подібностей між користувачами або об'єктами і рекомендували нові об'єкти, які сподобалися схожим користувачам.

З розвитком Інтернету та соціальних мереж, рекомендаційні системи стали більш індивідуалізованими. Вони використовують дані про інтереси, поведінку та демографічні особливості користувачів для підвищення точності рекомендацій.

Сучасні системи часто використовують гібридні підходи, комбінуючи колаборативну фільтрацію, контент-фільтрацію та інші методи, для забезпечення більш точних та різноманітних рекомендацій.

1.1 ВИДИ СИСТЕМ РЕКОМЕНДАЦІЙ

У сфері електронної комерції існують різні методи рекомендацій товарів, кожен з яких спрямований на створення індивідуалізованого досвіду для користувачів. Давайте детально розглянемо основні типи цих алгоритмів:

1.1.1 Колаборативна фільтрація:

Цей алгоритм працює шляхом збору та аналізу інформації про переваги користувачів. Він виявляє схожості між користувачами або між продуктами на основі їхніх покупок або оцінок. На підставі цього, система може рекомендувати продукти, які користувалися популярністю серед схожих користувачів або які є схожими на ті, які користувач вже оцінив позитивно. Принцип цієї фільтрації можна побачити на рисунку 1.1

Рис. 1.1 Принцип Колаборативної фільтрації

Ці алгоритми ґрунтуються на принципі "подібність приваблює" [4]. Вони дозволяють виявити схожі переваги між користувачами або між продуктами для створення рекомендацій. Важливо враховувати потенційні проблеми, такі як "холодний старт" (коли система має обмежену інформацію про нових користувачів або продукти) і масштабованість при великій кількості даних.

Серед алгоритмів колаборативної фільтрації можна виділити:

1) На основі користувача колаборативна фільтрація:

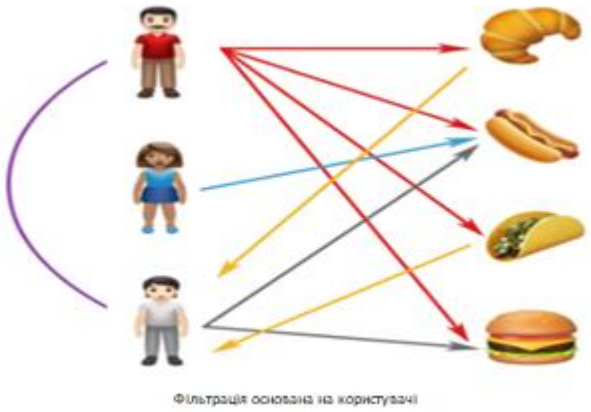


Рисунок 1.2 Колаборативна фільтрація на основі користувача

Цей підхід використовує аналіз схожості між користувачами для створення рекомендацій. Система аналізує взаємодію користувачів із продуктами (наприклад, оцінки) і шукає користувачів зі схожими смаками або перевагами. Потім вона ідентифікує продукти, які сподобалися цим схожим користувачам, але ще не були оцінені цільовим користувачем, і рекомендує їх. Принцип цієї методики можна побачити на рисунку 1.2

2) На основі товарів колаборативна фільтрація

Цей підхід зосереджений на аналізі схожості між самими об'єктами (товарами). Система досліджує, як користувачі взаємодіють з продуктами, і на основі цього визначає схожість між товарами, засновану на перевагах користувачів.

Головна відмінність цього методу від на основі користувачів колаборативної фільтрації полягає в акценті на даних про товари, а не користувачів. Система використовує ці дані для рекомендації інших продуктів, які схожі на ті, які вже сподобалися користувачу.

Цей метод дозволяє системі рекомендацій запропонувати продукти, які можуть бути цікаві користувачу, засновуючись на його попередніх виборах та вподобаннях, що допомагає підвищити якість та точність рекомендацій. Візуальне представлення цього принципу можна побачити на рисунку 1.3.

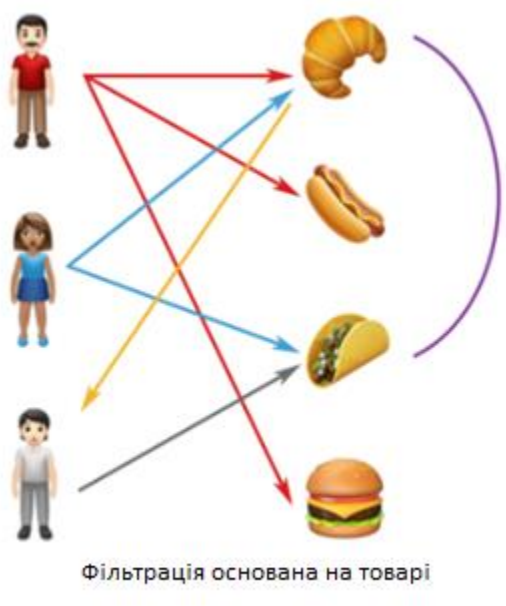


Рисунок 1.3 Колаборативна фільтрація на основі товарів

3) Гібридна колаборативна фільтрація

Гібридний метод колаборативної фільтрації інтегрує в собі кілька підходів до аналізу даних, що стосуються переваг користувачів і характеристик предметів. Він гармонійно об'єднує елементи користувацько-орієнтованої та об'єктно-орієнтованої фільтрації, доповнюючи їх іншими техніками підбору рекомендацій. Така багатогранна система, представлена на рис. 1.4, використовує вагові коефіцієнти для калібрування рекомендацій, спрямованих на забезпечення високої точності та індивідуалізації пропозицій користувачам.

Рисунок 1.4 – Принцип роботи гібридной фільтрації

1.1.2 Заснований на контенті (Content-based)

Підхід, що базується на аналізі вмісту товарів, використовує детальний розгляд атрибутів продуктів для створення рекомендацій. Через ретельний аналіз описів продуктів, ключових слів та інших важливих характеристик, алгоритм виявляє подібності між різними предметами. Заснований на виявленій схожості, алгоритм надає користувачам пропозиції товарів, що відповідають раніше виявленим уподобанням. Ця методологія фокусується на атрибутах об'єктів продажу, таких як ключові слова, описи, жанри, та інші метадані, щоб запропонувати рекомендації. Цей механізм створює індивідуальний профіль для кожного користувача, використовуючи дані про їх попередні вподобання та взаємодії з продуктами, і на цій основі пропонує подібні товари. Графічне представлення цієї концепції можна побачити на рис 1.5.

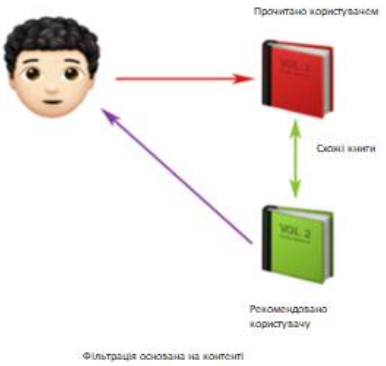


Рисунок 1.5 – Принцип роботи заснований на контенті

1.1.3 Гібридні системи

Гібридизація систем рекомендацій означає інтеграцію різноманітних методологій та алгоритмів для створення більш досконалих та індивідуалізованих пропозицій для користувачів. Використовуючи силу комбінованих стратегій, такі системи здобувають широке визнання у сфері цифрової комерції та інтернет-сервісів.

Сутність гібридних систем полягає в тому, що вони аплікують мікс різних моделей рекомендацій — від колаборативної фільтрації до контент-базованих алгоритмів, а також розглядають популярність, демографічні дані та інші критерії. Це дозволяє системі адаптуватися до різноманітних характеристик та уподобань користувачів.

Гібридні системи рекомендацій здійснюють інтеграцію різноманітних джерел та технік для надання точних індивідуалізованих пропозицій. Вони використовують синергію між спільними та унікальними властивостями різних алгоритмів, наприклад, агрегуючи інсайти з колаборативної фільтрації та контентно-орієнтованих рекомендацій.

Інноваційний підхід гібридних систем полягає у поєднанні прогнозів з множини алгоритмів, утворюючи комплексні рекомендації. Моделі машинного навчання можуть вносити свій вклад у прогнозування вподобань, які потім ансамблюються з традиційними методами.

Контекстуальна інформація, така як місцезнаходження користувача, час доби, а також поведінкові дані, інтегруються для тонкої настройки рекомендацій, забезпечуючи релевантність у конкретних ситуаціях. Використання цих даних дозволяє системі динамічно адаптуватися до змін у потребах та перевагах користувачів.

Ключовою перевагою гібридних систем є їх універсальність і здатність до надання високоякісних рекомендацій, які враховують як історію взаємодій, так і актуальні контекстуальні сигнали. Однак, вони вимагають ретельного аналізу та налаштування, а також мають високі вимоги до ресурсів для розробки та обслуговування.

Загалом, гібридні системи рекомендацій пропонують глибоку адаптацію до індивідуальних потреб користувачів, ефективно використовуючи дані для підвищення задоволеності клієнтів та комерційної ефективності. Розробка таких

систем є складним процесом, але вони мають потенціал забезпечити значну конкурентну перевагу на ринку.

1.2 Відмінності рекомендаційних систем

Порівняючи різні системи рекомендацій, можна виділити унікальні особливості, переваги та недоліки кожної з них.

1) Колаборативна фільтрація

Переваги:

Відкритість рекомендацій: Ця система здатна рекомендувати товари, які не мають детального опису чи контенту, виходячи з схожості між користувачами або товарами, базуючись на їхній історії взаємодії.

Виявлення складних зв'язків: Здатна ідентифікувати складні патерни взаємодій між користувачами та товарами, що можуть залишитися непоміченими для систем, заснованих на контенті.

Використання існуючих даних: Колаборативна фільтрація оперує наявними даними про взаємодії користувачів з товарами, тому не потребує додаткової інформації про характеристики або контент об'єктів.

Недоліки колаборативної фільтрації:

Проблема "холодного старту": Може виникнути проблема при наданні рекомендацій новим користувачам або для нових товарів через відсутність історії взаємодії.

Проблема розрідженості матриці: У великих системах матриця взаємодій може бути неповною, що ускладнює аналіз та знижує точність рекомендацій.

Ризик обмеження рекомендацій: Система може схилятися до рекомендації вузького спектра товарів, базуючись лише на попередніх взаємодіях користувачів, не враховуючи нові можливості чи інтереси.

2) Плюси та мінуси алгоритмів на основі контенту

Плюси алгоритмів, заснованих на контенті:

- Урахування особистих переваг: Ці алгоритми здатні адаптуватися до унікальних вподобань користувачів, аналізуючи атрибути та характеристики об'єктів.
- Мінімізація проблеми холодного старту: Здатні надавати рекомендації новим користувачам або для нових об'єктів, оскільки вони фокусуються на характеристиках самих об'єктів.
- Ефективність у системах з обмеженими даними: Особливо корисні в системах, де кількість даних обмежена.

Мінуси алгоритмів, заснованих на контенті:

- Обмеження виявлення складних зв'язків: Можуть не виявити складніші взаємозв'язки між користувачами та об'єктами, оскільки вони зосереджені лише на характеристиках об'єктів.
- Проблема врахування нових тенденцій: Можуть бути менш адаптивними до змін в уподобаннях користувачів або нових тенденцій, оскільки вони орієнтовані на статичні характеристики об'єктів.

4) Плюси та мінуси гібридних систем

Переваги гібридних систем рекомендацій товарів:

- Краща точність рекомендацій: Гібридні системи, поєднуючи різні методи, забезпечують більш точні рекомендації, використовуючи дані з колаборативної фільтрації, контенту, демографії тощо.
- Широкий спектр рекомендацій: Можуть пропонувати рекомендації, засновані не лише на схожості користувачів або об'єктів, а й на

інших факторах, розширюючи можливості для врахування різних аспектів та потреб користувачів.

Недоліки гібридних систем рекомендацій товарів:

- Складність інтеграції: Комбінування різних підходів може бути складним та вимагати значних ресурсів для налаштування та підтримки.
- Вибір та налаштування алгоритмів: Потребує витонченого підбору та налаштування алгоритмів для оптимальної ефективності, що може бути складним завданням.
- Обмеження в широті та глибині використання даних: Хоча гібридні системи поєднують різні методи, вони все одно можуть мати обмеження у використанні великої кількості даних та їх різноманітності.
- Проблема холодного старту у гібридних системах рекомендацій: Гібридні системи можуть зіткнутися з проблемою недостатності даних про нових користувачів або продукти для забезпечення точних рекомендацій. Для вирішення цієї проблеми можна використовувати альтернативні стратегії, такі як рекомендації на основі популярності або контексту.
- Вимоги до обчислювальних ресурсів: Гібридні системи потребують значних обчислювальних ресурсів через використання комбінованих алгоритмів та методів. Обчислювальна складність може стати викликом у випадку обробки великих обсягів даних або в умовах обмеженого доступу до ресурсів.

Загальний аналіз гібридних систем: Незважаючи на зазначені вище недоліки, гібридні системи рекомендацій мають значні переваги, які часто переважають їх обмеження. Вони забезпечують точніші та більш персоналізовані рекомендації,

а також ефективно враховують різноманітні аспекти поведінки та потреб користувачів. Хоча розробка та підтримка таких систем може вимагати значних зусиль та ресурсів, вони здатні принести значну цінність, покращуючи загальний досвід користувачів.

Висновки до розділу

Колаборативна фільтрація відрізняється здатністю до рекомендацій, що не вимагають детальної інформації про об'єкти, але стикається з проблемами холодного старту та обмеженістю в охопленні різноманітних об'єктів. Це робить її ефективною для систем із великою кількістю історичних даних, але менш відповідною для нових або менш популярних об'єктів.

Алгоритми, засновані на контенті, надають перевагу індивідуальним вподобанням користувачів, але можуть бути обмежені у виявленні складних взаємозв'язків між об'єктами. Вони ефективні у ситуаціях, де доступні детальні описи об'єктів, але можуть бути недостатні для врахування змін у перевагах користувачів.

Гібридні системи об'єднують кращі аспекти різних підходів, пропонуючи точні та персоналізовані рекомендації, але водночас потребують складнішої розробки та підтримки. Вони здатні адаптуватися до різних контекстів та враховувати різноманітні потреби користувачів.

У цілому, кожен із розглянутих підходів має свої переваги та обмеження, що робить їх відповідними для різних сценаріїв використання. Вибір конкретного підходу чи комбінації підходів залежить від специфіки задачі, доступних даних та обчислювальних ресурсів.

РОЗДІЛ 2 АРХІТЕКТУРНІ РІШЕННЯ РОЗРОБКИ ДОДАТКІВ ТА АЛГОРИТМІЧНЕ ЗАБЕЗПЕЧЕННЯ

Цей дипломний проект має за мету створити веб-застосунок, який надає персоналізовані пропозиції засновані на індивідуальних смаках покупців. Застосунок буде побудований з використанням передового фреймворку Angular, що інтегрує найкращі практики розробки інтерфейсу. Цей підхід надасть власнику платформи засоби для економії коштів при подальшому розширенні та оновленні системи, а також сприятиме зростанню доходів через запровадження продуманої системи рекомендацій товарів, орієнтованої на конкретні запити користувачів.

2.1 Фронтенд частина

Зі структурой фронтенд частини ви можете ознайомитись на рис 2.1

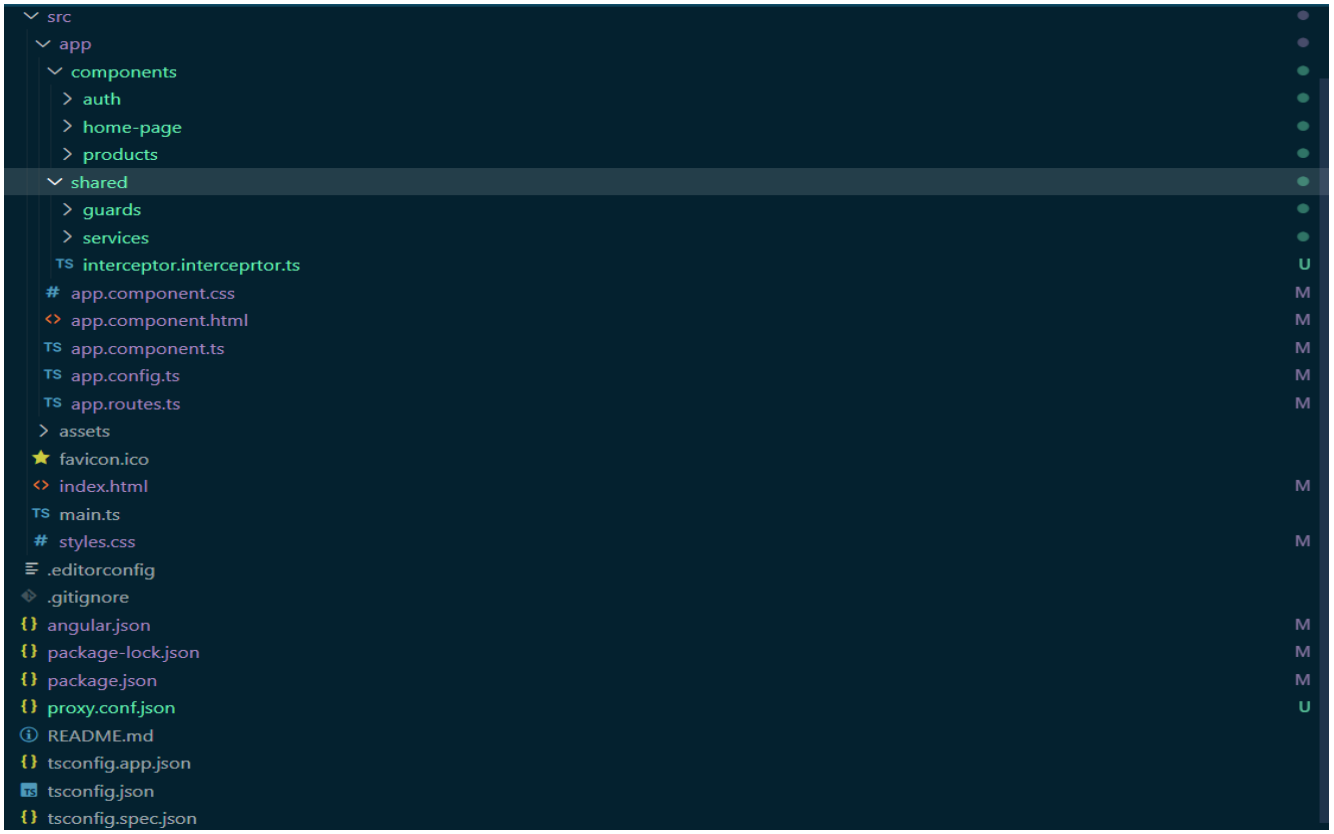


Рис. 2.1 Структура Фротенд частини

Структура фронтенд-частини системи, заснована на фреймворку Angular, розділена на модулі, компоненти та служби, що дозволяє ефективно управляти

кодом і ресурсами. Кожен модуль має визначені завдання і залежності, які спрощують масштабування та підтримку додатку.

У кореневій папці src розташовано основні файли та каталоги проекту:

- app - основний модуль додатку, який включає:
- components - підкаталоги для кожного компонента (наприклад, auth для аутентифікації, home-page для головної сторінки, products для представлення товарів), що містять HTML, CSS та TypeScript файли.
- shared - включає служби (services), охоронців (guards), та інші елементи, що використовуються між різними компонентами додатку.
- assets - медіа-файли, такі як зображення та стилі, які використовуються у додатку.
- styles.css - глобальні стилі для всього додатку.

Angular використовує компонентно-орієнтований підхід, що дозволяє розділяти UI і логіку на незалежні блоки для легкого повторного використання та тестування.

Для стилізації використовуються Bootstrap та Angular Material, що надають змогу створювати адаптивний та доступний інтерфейс. Bootstrap вносить у проект готові стилі та компоненти, такі як кнопки, форми та навігаційні елементи. Angular Material пропонує комплексний набір готових до використання компонентів згідно з Material Design від Google, включаючи інструменти для навігації, роботи з формами, списками, вікнами та іншими UI елементами.

Кодова база організована так, що сприяє чистоті коду та легкості внесення змін, що є критично важливим для довгострокової підтримки та розвитку проекту. Завдяки модульності та компонентності Angular, система легко адаптується під змінні вимоги бізнесу та може швидко реагувати на нові тренди ринку.

2.2 Бекенд частина

Зі структурой бекенд частини можно ознайомитись на рис. 2.2

Рис. 2.2 Структура Backend частини

- controllers: Папка містить модулі JavaScript, які використовуються для обробки запитів. Вони включають:
- authController.js: Забезпечує функціонал авторизації користувачів.
- collaborativeFilter.js: відповідає за рекомендаційну систему на основі співпраці між користувачами.
- productsController.js: Управління товарами або продуктами.
- ratingController.js: Управління рейтингами користувачів або товарів.
- recommendationController.js: відповідає за видачу рекомендацій.
- trackController.js: слідкування за певними діями користувачів або процесами.
- helpers: Папка, яка містить утиліти або вспомогательні функції.
- helper.js:містити загальні функції, що використовуються різними контролерами.

- models: Папка з моделями для представлення даних, які використовуються в базі даних.
- Item.js: Модель товарів.
- Rating.js: Модель для рейтингів.
- User.js: Модель для користувачів.
- node_modules: Папка з бібліотеками та залежностями проекту Node.js.
- python: Папка, що містить Python скрипти, для реалізації алгоритмів машинного навчання та обробки даних.
- collaborative_filtering.py: Скрипт для колаборативного фільтрування.
- content_based.py: Скрипт для контент-орієнтованого фільтрування.
- routes: Папка з модулями, що визначають маршрути для HTTP запитів.
- authRoute.js: Маршрути для авторизації.
- productsRoute.js: Маршрути для продуктів.
- ratingRoute.js: Маршрути для рейтингу.
- recommendRoute.js: Маршрути для рекомендацій.
- .env: Файл змінних середовища, що зазвичай містить конфіденційну інформацію.
- index.js: Головний файл сервера, який запускає додаток Node.js.
- package-lock.json і package.json: Файли, які містять інформацію про проект і список залежностей.

2.3 Методики та алгоритми для реалізації застосунку

Для реалізації системи рекомендації на основі різних видів фільтрації, були застосовані різні методики та алгоритми, які ми розглянемо нижче.

2.3.1 Тематична модель (Topic Modeling)

Тематична модель (Topic Modeling) є методом машинного навчання, який застосовується для аналізу великих обсягів текстових даних, щоб виявити приховані тематичні структури у документах. Основна ідея полягає в тому, що

документи складаються зі суміші різних тем, і ці теми можуть бути виявлені шляхом аналізу частоти слів у тексті.

Основні аспекти тематичної моделі:

1. Автоматизований Аналіз Тексту: Тематичне моделювання дозволяє автоматизувати процес аналізу тексту, замість того, щоб вручну категоризувати або тегувати контент.
2. Виявлення Тем: Воно виявляє теми, які найчастіше зустрічаються в наборі документів, дозволяючи зрозуміти основні ідеї або концепції, які обговорюються в тексті.
3. Методи Тематичного Моделювання: Найпоширенішими методами є Latent Dirichlet Allocation (LDA), Latent Semantic Analysis (LSA), і Non-negative Matrix Factorization (NMF). Кожен з цих методів має свої особливості в тому, як вони ідентифікують теми та розподіляють їх у документах.
4. Застосування: Тематичне моделювання широко застосовується в різних областях, таких як обробка природної мови, аналіз соціальних медіа, інформаційний пошук, рекомендаційні системи тощо.
5. Переваги: Головними перевагами є можливість ефективно обробляти великі обсяги даних, а також надання інтуїтивного розуміння прихованих патернів у текстах.
6. Виклики: Серед викликів – визначення кількості тем, інтерпретація виявлених тем, а також обмеження, пов'язані з неоднозначністю мови та контекстуальними варіаціями.

Тематична модель є потужним інструментом в арсеналі обробки природної мови, який допомагає зрозуміти складність і різноманітність текстових даних, забезпечуючи глибший інсайт в контент. Цей метод стає незамінним в сучасній епосі Big Data, де потрібно швидко та ефективно аналізувати великі обсяги невпорядкованої текстової інформації.

2.3.2 Векторні представлення (Vector Representations)

Векторні представлення у контексті обробки природної мови (NLP) та машинного навчання є фундаментальним поняттям, яке дозволяє комп'ютерним системам розуміти та обробляти текстові дані. Цей підхід перетворює слова, фрази або цілі документи у числові вектори, що значно спрощує їх аналіз та обробку алгоритмами машинного навчання.

Основні аспекти векторних представлень:

1. **Концепція:** Векторне представлення перетворює слова або текстові елементи у високовимірні числові вектори. Кожне слово або фраза представляється як точка у високовимірному просторі, де схожі слова мають схожі векторні представлення.
2. **Застосування:** Векторні представлення використовуються у багатьох задачах NLP, таких як машинний переклад, автоматичне резюмування тексту, аналіз емоцій, класифікація текстів, створення рекомендаційних систем тощо.
3. **Переваги:** Ці методи дозволяють комп'ютерам більш точно обробляти та розуміти мовні структури, а також виявляти семантичні відносини між словами і фразами.
4. **Виклики:** Одним із головних викликів є обробка полісемії (коли одне слово має кілька значень) та контекстуальної змінності мови. Крім того, створення ефективних векторних представлень може вимагати великих обсягів даних та обчислювальних ресурсів.
5. **Тенденції розвитку:** Сучасні дослідження у цій області спрямовані на створення ще більш точних та контекстуально залежних моделей векторних представлень, які могли б ще краще враховувати нюанси природної мови.

Векторні представлення відіграють ключову роль у розвитку обробки природної мови та штучного інтелекту загалом, дозволяючи машинам

ефективно "розуміти" та обробляти мову на рівні, близькому до людського сприйняття. Це відкриває широкі можливості для різних застосувань, від автоматизації обслуговування клієнтів до розширеного аналізу великих даних.

2.3.3 Розклад матриць

Matrix Factorization (MF), або розклад матриці, є фундаментальною технікою в області машинного навчання та обробки даних, яка знаходить широке застосування, особливо у системах рекомендацій. Ця техніка полягає у розкладі великої матриці на дві або більше менших матриць таким чином, що їхній добуток наближено дорівнює початковій матриці.

Основні аспекти Matrix Factorization:

1. **Призначення:** MF використовується для виявлення прихованих особливостей або факторів у даних, які можуть пояснити спостережувані взаємодії між змінними. Наприклад, у контексті рекомендаційних систем, це може означати виявлення прихованих факторів, що впливають на вподобання користувачів.
2. **Алгоритми:** Найвідомішими алгоритмами MF є Singular Value Decomposition (SVD) та Alternating Least Squares (ALS). SVD використовується у задачах, де матриці є щільними, тоді як ALS більше підходить для розріджених матриць.
3. **Застосування в Рекомендаційних Системах:** У рекомендаційних системах MF використовується для прогнозування невідомих рейтингів або вподобань користувачів, шляхом аналізу взаємодій між користувачами та продуктами.
4. **Переваги:** MF допомагає виявити складні взаємозв'язки та закономірності в даних, які не можуть бути легко виявлені іншими методами. Вона ефективна у виявленні прихованих особливостей, зменшенні розмірності та підвищенні якості прогнозування.

5. **Виклики:** Однією з головних проблем при реалізації MF є вирішення проблеми "холодного старту", коли нові користувачі або продукти мають обмежену історію взаємодій. Крім того, точність моделі MF може бути обмежена шумом та викидами в даних.
6. **Оптимізація та Покращення:** Існують різні методи для оптимізації MF, включаючи регуляризацію для запобігання перенавчанню та різні стратегії оцінювання для покращення точності прогнозування.

Matrix Factorization є потужним інструментом у руках дослідників та розробників, що дозволяє з глибшим розумінням аналізувати великі набори даних та створювати більш точні та персоналізовані рекомендації. Вона є ключовою складовою сучасних рекомендаційних систем, які є невід'ємною частиною багатьох онлайн-сервісів та електронної комерції.

2.3.4 Сингулярне Розкладання Значень

Сингулярне Розкладання Значень (Singular Value Decomposition, SVD) є однією з найбільш могутніх технік матричного аналізу в лінійній алгебрі, широко застосовуваною в областях обробки даних, машинного навчання, сигнальної обробки, та інших. Цей метод дозволяє розкласти будь-яку прямокутну матрицю на три інші матриці з певними властивостями.

Основні характеристики Сингулярного Розкладання Значень:

1. **Математичний Принцип:** Для даної матриці A розміру $m \times n$, SVD розкладає її на три матриці - U, Σ і $*V*$, де U і V є ортонормованими матрицями, а Σ є діагональною матрицею.
2. **Компоненти Розкладу:**
    - $\boldsymbol{U}$**:** Матриця розміру $m \times m$, що містить ліві сингулярні вектори.
    - $\boldsymbol{\Sigma}$**:** Діагональна матриця розміру $m \times n$ з сингулярними значеннями на діагоналі.
    - $\boldsymbol{V*}$**:** Спряжена транспонована матриця розміру $n \times n$, що містить праві сингулярні вектори.

3. **Застосування:** SVD використовується в багатьох областях, включаючи розпізнавання образів, статистичний аналіз, обробку сигналів, компресію даних, та особливо в системах рекомендацій, де воно допомагає у виявленні прихованих факторів в інтеракціях між користувачами та предметами.
4. **Переваги:** Основною перевагою SVD є його здатність виявляти внутрішню структуру даних, зменшуючи розмірність, при цьому зберігаючи найбільш значущу інформацію.
5. **Виклики:** Одним з основних викликів є обчислювальна складність, особливо для великих матриць, а також необхідність у великих обсягах даних для ефективного застосування SVD.
6. **Оптимізація та Покращення:** Існують різні варіації SVD, такі як Truncated SVD, які використовуються для підвищення ефективності та зменшення обчислювального навантаження.

SVD є ключовим інструментом в аналізі даних та машинному навчанні, що дозволяє виявляти та використовувати глибокі внутрішні структури даних, тим самим підвищуючи якість моделювання та прогнозування.

2.3.5 Косинусна схожість між користувачами в колаборативній фільтрації:

Косинусна схожість між користувачами в колаборативній фільтрації є ключовим методом для виявлення схожості між різними користувачами на основі їхніх вподобань чи поведінки. Цей метод допомагає у створенні персоналізованих рекомендацій, використовуючи інформацію про те, що подобається одним користувачам, для прогнозування потенційних інтересів інших.

Косинусна схожість між користувачами в колаборативній фільтрації - це метод вимірювання схожості між двома користувацькими профілями на основі їх взаємодій з різними продуктами або послугами. Цей метод використовується

для визначення, наскільки схожі переваги чи інтереси двох користувачів, і відіграє ключову роль у системах рекомендацій.

Формула Косинусної Схожості:

Косинусна схожість між двома векторами A та B обчислюється за формулою: cosine similarity(A,B) = A·B / ‖A‖‖B‖ де:

A·B - це скалярний добуток векторів A та B,

‖A‖ та ‖B‖ - норми (довжини) векторів A та B відповідно.

Принцип Косинусної Схожості у Колаборативній Фільтрації:

1. Векторизація Користувацьких Профілів: Кожен користувач представляється вектором, який містить його взаємодію з різними товари або послугами, наприклад, рейтинги, які він залишив.
2. Обчислення Косинусної Схожості: Косинусна схожість між двома користувацькими векторами обчислюється як косинус кута між цими векторами. Це дає показник схожості між користувацькими профілями, який варіюється від -1 (повна відмінність) до 1 (ідентичність).
3. Використання у Рекомендаційних Системах: Цей метод дозволяє ідентифікувати користувачів із схожими інтересами або вподобаннями. Якщо два користувачі мають високу косинусну схожість, то вподобання одного з них можуть бути використані для рекомендацій іншому.
4. Переваги: Косинусна схожість ефективна для роботи з розрідженими даними, які часто зустрічаються в користувацьких рейтингах, і вона відносно невразлива до різних масштабів рейтингів.
5. Обмеження: Один з основних недоліків полягає в тому, що метод може не враховувати важливість окремих елементів вектора, особливо в ситуаціях, коли користувачі мають відмінні вподобання або взаємодіяли з різними наборами товарів.
6. Застосування: Косинусна схожість широко використовується в системах колаборативної фільтрації, особливо у ситуаціях, де потрібно знайти

"сусідів" для конкретного користувача з метою рекомендації товарів або послуг.

Використання косинусної схожості між користувачами дозволяє системам рекомендацій виявляти схожі шаблони поведінки та вподобань серед великих груп користувачів, підвищуючи якість та релевантність рекомендацій.

2.3.5 Косинусна схожість між користувачем і товаром в контент-орієнтованій фільтрації

Косинусна схожість між користувачем і товаром в контент-орієнтованій фільтрації - це метод визначення ступеня відповідності між інтересами користувача та атрибутами товару. Цей підхід дозволяє рекомендаційним системам пропонувати товари, які найкраще відповідають унікальним перевагам індивідуального користувача.

Принцип Косинусної Схожості у Контент-орієнтованій Фільтрації:

1. Векторизація Характеристик: Як користувачів, так і товарів представляють у вигляді векторів. Вектор користувача може включати інформацію про його переваги або попередню взаємодію з продуктами, тоді як вектор товару відображає його характеристики, такі як категорії, описи, теги тощо.
2. Застосування у Рекомендаційних Системах: Косинусна схожість використовується для оцінювання релевантності товарів для конкретного користувача. Товари, які мають високу схожість з вектором інтересів користувача, вибираються для рекомендацій.
3. Переваги: Косинусна схожість є ефективною для порівняння векторів незалежно від їх довжини, що робить її ідеальною для порівняння різноманітних типів даних.
4. Обмеження: Одним з недоліків є те, що цей метод може не враховувати важливість окремих атрибутів і не завжди ефективний у ситуаціях, де атрибути мають різний вплив на рішення користувачів.

Косинусна схожість у контент-орієнтованій фільтрації є важливим інструментом, що дозволяє системам рекомендацій точніше зрозуміти та враховувати індивідуальні переваги користувачів, підвищуючи якість і точність рекомендацій.

Висновки до розділу

У цьому було розглянуто ключові архітектурні рішення та алгоритмічне забезпечення, необхідне для розробки веб-застосунку, який здатен надавати персоналізовані пропозиції на основі індивідуальних уподобань користувачів. Використання передового фреймворку Angular для фронтенд-частини та ефективних рішень для бекенду забезпечує гнучку та оптимізовану структуру системи, що сприяє легкості управління кодом, масштабуванням та оновленням додатку.

Розгляд різних методик та алгоритмів, таких як тематичне моделювання, векторні представлення, розклад матриць, сингулярне розкладання значень та косинусна схожість, підкреслює глибокий аналітичний підхід проекту до розробки рекомендаційної системи. Ці техніки дозволяють не тільки ефективно обробляти великі обсяги даних, але й забезпечують точне та інтуїтивно зрозуміле виявлення уподобань користувачів, а також генерування відповідних рекомендацій товарів.

Загалом, архітектурне та алгоритмічне забезпечення, описане в цьому розділі, становить фундамент для створення високоефективного, гнучкого та адаптивного веб-застосунку, здатного задовольнити потреби сучасних користувачів та динамічно реагувати на зміни ринку. Використання сучасних технологій та інноваційних підходів до обробки даних є ключем до успіху проекту, спрямованого на забезпечення високої якості обслуговування користувачів і збільшення доходів бізнесу.

РОЗДІЛ 3 РОЗРОБКА ВЕБ-ДОДАТКУ

3.1 Мови програмування та технології для реалізації

3.1.1 Javascript

Зважаючи на універсальність JavaScript, варто відзначити, що ця мова програмування є стандартом для розробки веб-додатків. Вона підтримується всіма сучасними браузерами, що робить її ідеальним вибором для створення програм, які працюють на будь-якому пристрої з доступом до мережі Інтернет.

JavaScript використовується як на клієнтській, так і на серверній стороні розробки. На клієнтській стороні вона дозволяє створювати інтерактивні елементи веб-сторінок та забезпечує можливість взаємодії з користувачем на браузері. На серверній стороні, завдяки платформі, такій як Node.js, JavaScript використовується для створення потужних та масштабованих серверних додатків.

Крім того, JavaScript має багату екосистему з численними бібліотеками та фреймворками. Ці інструменти допомагають веб-розробникам прискорити розробку та покращити функціональність веб-застосунків. Наприклад, фреймворки, такі як React і Vue.js, роблять можливим створення інтерактивних веб-додатків, а Angular надає інструменти для розробки складних веб-проектів.

Загалом, ці характеристики роблять JavaScript популярним вибором для веб-розробки і надають йому широкий спектр можливостей.

3.1.2 TypeScript

TypeScript: Статично типізована мова програмування

TypeScript - це мова програмування, яка базується на JavaScript і надає можливість визначати типи даних для змінних, параметрів функцій і об'єктів. Це допомагає зробити ваш код більш надійним та легше підтримуваним.

Основні переваги TypeScript:

Статична типізація: TypeScript дозволяє визначити типи для змінних, що допомагає виявляти помилки на етапі розробки і полегшує роботу з великими проектами.

Орієнтована на об'єкти розробка: TypeScript підтримує об'єктно-орієнтований підхід до програмування, включаючи підтримку класів і інтерфейсів.

Розширена інфраструктура: Є багато інструментів і бібліотек, які спрощують розробку на TypeScript, включаючи популярні фреймворки, такі як Angular і React.

Використання TypeScript: TypeScript можна використовувати для розробки веб-додатків, мобільних додатків, бекенд-систем і багатьох інших сфер програмування.

З TypeScript ваш код стає більш структурованим і менше схильним до помилок, що робить його потужним інструментом для розробки програмного забезпечення.

3.1.3 Node.js

Node.js - це середовище виконання JavaScript, яке дозволяє виконувати JavaScript на серверній стороні. Основні характеристики та переваги Node.js включають:

Серверні застосунки: Node.js дозволяє створювати серверні застосунки з використанням JavaScript. Це дозволяє розробникам створювати високоефективні та масштабовані серверні додатки, які можуть обробляти багато запитів одночасно.

Асинхронність: Однією з ключових особливостей Node.js є асинхронність. Він використовує неблокуючий ввод/вивід, що дозволяє обробляти багато запитів без блокування виконання інших операцій. Це особливо корисно для обробки багатьох одночасних запитів.

Велика екосистема: Node.js має велику та активну екосистему бібліотек і модулів, які дозволяють розробникам легко розширювати функціональність своїх додатків. Це включає в себе бібліотеки для роботи з мережами, файловою системою, базами даних і багато інших.

Спільнота та підтримка: Node.js має активну спільноту розробників, яка надає підтримку та розвиває цю платформу. Є багато документації, учбових матеріалів та сторонніх розширень, які полегшують роботу з Node.js.

Швидкість виконання: Node.js базується на движку V8 від Google, який є вельми швидким і виконує JavaScript з високою швидкістю. Це робить Node.js відмінним вибором для обробки запитів у реальному часі та великих обчислювальних завдань.

Node.js використовується для створення різних типів застосунків, включаючи веб-сервери, API, чат-сервери, додатки для роботи з базами даних і багато інших. Він є популярним інструментом у веб-розробці та має широкий спектр застосувань.

3.1.4 Express Js

Express.js, або просто Express, є веб-фреймворком для Node.js, який спрощує створення веб-додатків та веб-серверів. Він надає ряд корисних функцій і інструментів для розробки веб-додатків, що допомагає розробникам швидко створювати потужні та ефективні застосунки. Основні характеристики і переваги Express.js включають:

Маршрутизація: Express дозволяє визначити маршрути для обробки HTTP-запитів. Ви можете визначити, які дії повинні бути виконані при отриманні конкретного запиту (GET, POST, PUT, DELETE і т. д.) і на який URL-шлях вони повинні відповідати.

Мідлвари: Express використовує концепцію мідлварів (middleware), що дозволяє обробляти запити перед тим, як вони досягнуть маршруту. Це дозволяє

виконувати автентифікацію, авторизацію, перевірку валідності даних та інші операції перед обробкою запиту.

Шаблонізація: Express дозволяє використовувати шаблонізацію для створення динамічних веб-сторінок. Ви можете використовувати популярні двіжки шаблонів, такі як EJS або Handlebars, для відображення даних на сторінці.

Робота з HTTP-запитами та відповідями: Express надає зручний інтерфейс для роботи з HTTP-запитами та відповідями. Ви можете легко отримувати дані з запитів, надсилати HTTP-відповіді, встановлювати заголовки і багато іншого.

Розширюваність: Express є дуже розширюваним фреймворком. Ви можете додавати власні мідлвари, розширювати функціональність за допомогою сторонніх модулів і створювати власні розширення для веб-додатків.

Активна спільнота: Express має велику та активну спільноту розробників, що робить його добре підтримуваним і надійним фреймворком для веб-розробки.

Express.js є дуже популярним вибором для створення веб-серверів і веб-додатків на платформі Node.js. Він допомагає розробникам ефективно створювати API, веб-сайти, веб-додатки і багато іншого.

3.1.5 Python

Python - це високорівнева мова програмування, яка відома своєю простотою та читабельністю коду. Вона має широке застосування у різних галузях програмування і включає в себе наступні основні характеристики та переваги:

1. Читабельний синтаксис: Python відомий своїм читабельним і лаконічним синтаксисом, що робить його ідеальним для початківців та досвідчених розробників. Код на Python легко читається і розуміється людьми.

2. Крос-платформеність: Python є крос-платформеним, що означає, що ви можете виконувати код Python на різних операційних системах, включаючи Windows, macOS та Linux.

3. Багата стандартна бібліотека: Python має велику стандартну бібліотеку, яка включає в себе різноманітні модулі і функції для розв'язання різних завдань. Це дозволяє вам швидко створювати програми без необхідності писати код з нуля.

4. Застосування у різних галузях: Python використовується у багатьох галузях, включаючи веб-розробку, наукові дослідження, штучний інтелект, обробку даних, аналітику, робототехніку та інше. Він має багато бібліотек і фреймворків, що полегшують роботу в цих областях.

5. Активна спільнота: Python має велику та активну спільноту розробників, що надає підтримку, створює сторонні бібліотеки та фреймворки, і розвиває мову. Це допомагає вирішувати проблеми та підтримувати Python у актуальному стані.

6. Розширюваність: Python дозволяє вам використовувати модулі, написані на C/C++, що робить його розширюваним та дозволяє взаємодіяти з іншими мовами програмування.

Python є популярним вибором для різних завдань програмування, від розробки веб-додатків до наукових досліджень та штучного інтелекту. Його простота та потужність роблять його однією з найпопулярніших мов програмування у світі.

3.2 Реалізація та тестування веб-додатку

З реалізацію коду методів фільтрування можна ознайомитись в Додатку А.

Основні кроки для тестування веб-додатку

Крок 1: Встановлення середовища NodeJS

Для початку роботи з програмою, переконайтеся, що у вас встановлене середовище NodeJS.

Крок 2: Відкриття проєкту у середовищі розробки

Відкрийте проєкт у вашому обраному середовищі розробки, наприклад, Visual Studio або будь-якому іншому, яке вам зручно використовувати.

Крок 3: Запуск веб-застосунку

Після відкриття проекту виконайте наступні дії:

1. В командному рядку (CLI) введіть команду **ng serve**.
2. Після виконання команди, веб-застосунок буде доступний за посиланням **http://localhost:4200** у вашому браузері.

Крок 4: Аутентикація

По замовчуванню, якщо ви не авторизовані, ви будете перенаправлені на аутентикаційну сторінку (home page). На цій сторінці вам буде запропоновано два варіанти:

- Увійти до сайту, якщо у вас вже є обліковий запис.
- Зареєструватись, якщо у вас немає облікового запису.

рис 3.1 містить зображення автентифікаційної сторінки.

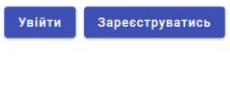


Рис.3.1 сторінка для входу або реєстрації

Це всі необхідні кроки для початку роботи з програмою. Дотримуйтеся інструкцій та насолоджуйтеся використанням веб-застосунку.

Якщо користувач натисне кнопку "Увійти" (Sign In), то з'явиться модальне вікно для введення особистих даних. Це вікно має наступний вміст:

1. Форма для заповнення особистих даних користувача: В цій формі користувач повинен ввести свою електронну пошту та пароль.
2. Кнопка "Закрити": Користувач може закрити модальне вікно, якщо він не бажає авторизуватися на даний момент.

3. Кнопка "Увійти": Кнопка для відправки запиту на сервер для автентифікації. Проте, ця кнопка буде заблокована до того моменту, поки користувач не введе усі обов'язкові поля для заповнення, тобто електронну пошту та пароль.

Після успішного запиту на сервер та автентифікації, користувач буде автоматично переправлений на головну сторінку веб-застосунку.

Для кращого розуміння, ось зображення модального вікна для автентифіка́ції:

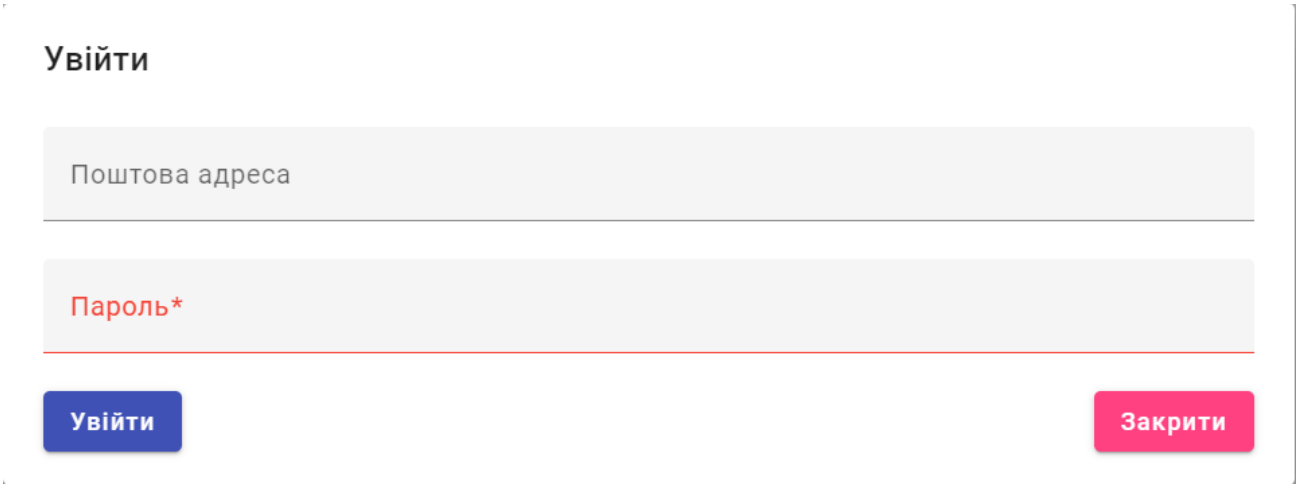


Рис.3.2 модальне вікно для входу

Це модальне вікно дозволяє користувачеві зручно та безпечно ввести свої облікові дані для входу до системи.

Якщо користувач натисне кнопку "Зареєструватися", то з'явиться модальне вікно для введення особистих даних для реєстрації. Це вікно має наступний вміст:

1. Форма для заповнення особистих даних користувача: У цій формі користувач повинен ввести свою електронну пошту, ім'я, новий пароль та повторне введення паролю для підтвердження.
2. Кнопка "Закрити": Користувач може закрити модальне вікно, якщо він не бажає реєструватися на даний момент.

3. Кнопка "Зареєструватися": Кнопка для відправки запиту на сервер для реєстрації. Проте, ця кнопка буде заблокована до того моменту, поки користувач не введе усі обов'язкові поля для заповнення, тобто електронну пошту, ім'я та паролі.

Після успішного запиту на сервер та реєстрації, користувач буде автоматично перенаправлений на головну сторінку веб-застосунку.

Для кращого розуміння, ось зображення модального вікна для реєстрації:

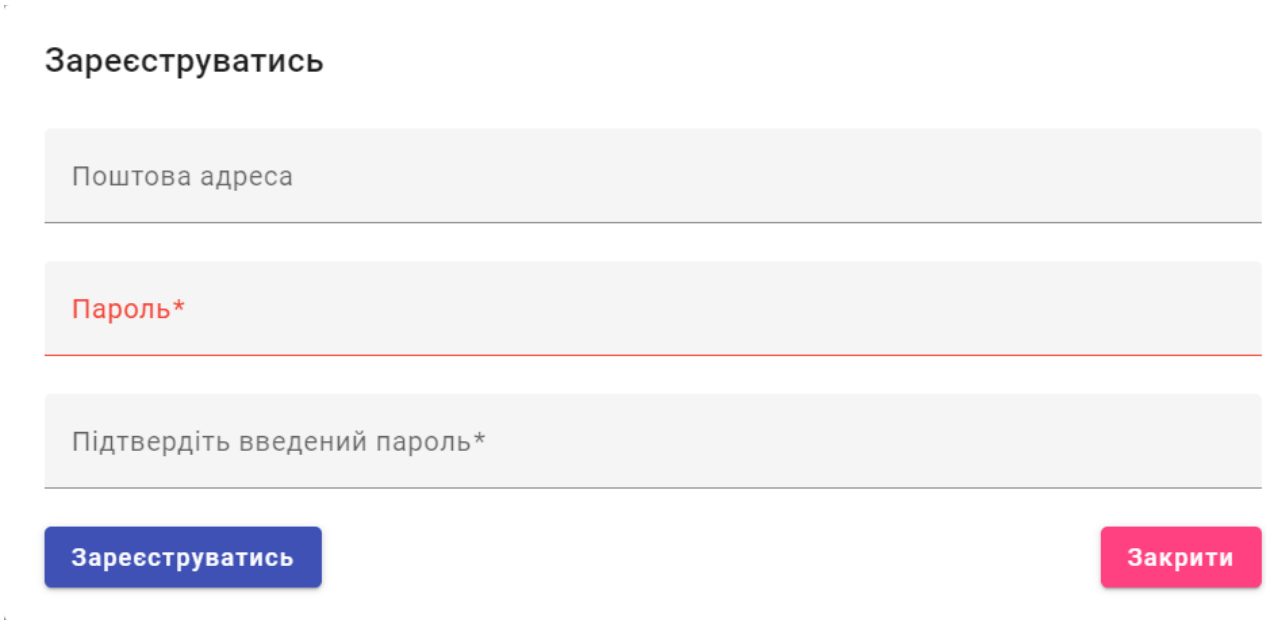


Рис.3.3 модальне вікно для реєстрації

Після того, як користувач натисне на відповідну кнопку (наприклад, "Увійти" або "Зареєструватися") і успішно виконає дію, його буде перенаправлено на головну сторінку веб-застосунку.

Головна сторінка містить наступний функціонал:

1. Список доступних товарів: На головній сторінці зображено перелік усіх доступних товарів. Кожен товар включає зображення товару, ціну та назву.
2. Кнопка для придбання: Кожен товар має кнопку, за допомогою якої користувач може придбати цей товар. При натисканні на цю кнопку і при умові успішної покупки, користувачу буде показане модальне вікно із пропозицією залишити особистий відгук про придбаний товар.

3. Модальне вікно для відгуків: Модальне вікно для відгуків з'являється після покупки товару і надає користувачу можливість залишити свій відгук про придбаний товар. Відгук може включати текстовий коментар та оцінку товару.

4. Кнопки для отримання рекомендацій з різних систем

Головна сторінка має наступний вигляд рис. 3.4

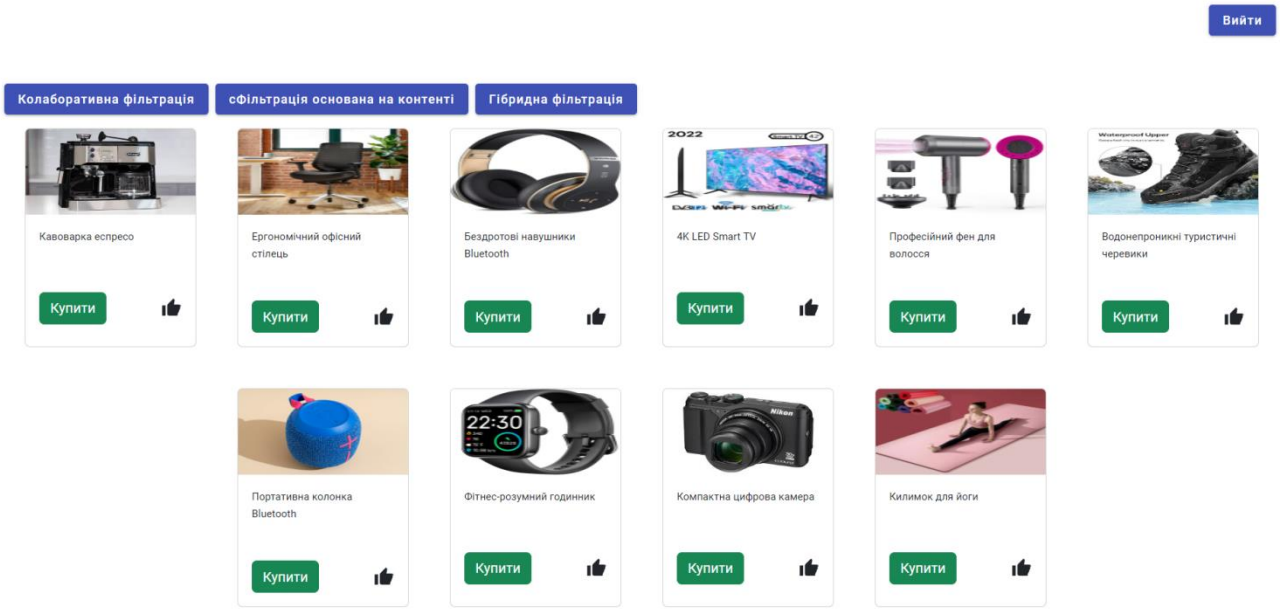


Рис.3.4 головна сторінка

При натисканні на кнопку "Придбати" після успішної покупки товару, з'являється модальне вікно для оцінки придбаного товару. Це вікно призначене для збирання даних, які можуть бути використані для подальшого аналізу та рекомендаційних систем.

Модальне вікно містить наступний функціонал:

1. Форма оцінки товару: В модальному вікні користувач може залишити оцінку придбаного товару. Це може бути числова оцінка або зіркова рейтингова система.

2. Кнопка "Зберегти": Після заповнення форми оцінки та коментаря користувач натискає кнопку "Зберегти", щоб зберегти свій відгук.

3. Кнопка "Скасувати": Кнопка "Скасувати" дозволяє користувачеві закрити модальне вікно без збереження відгуку.

Модальне вікно для оцінки придбаного товару має наступний вигляд рис.3.6 (це приклад):

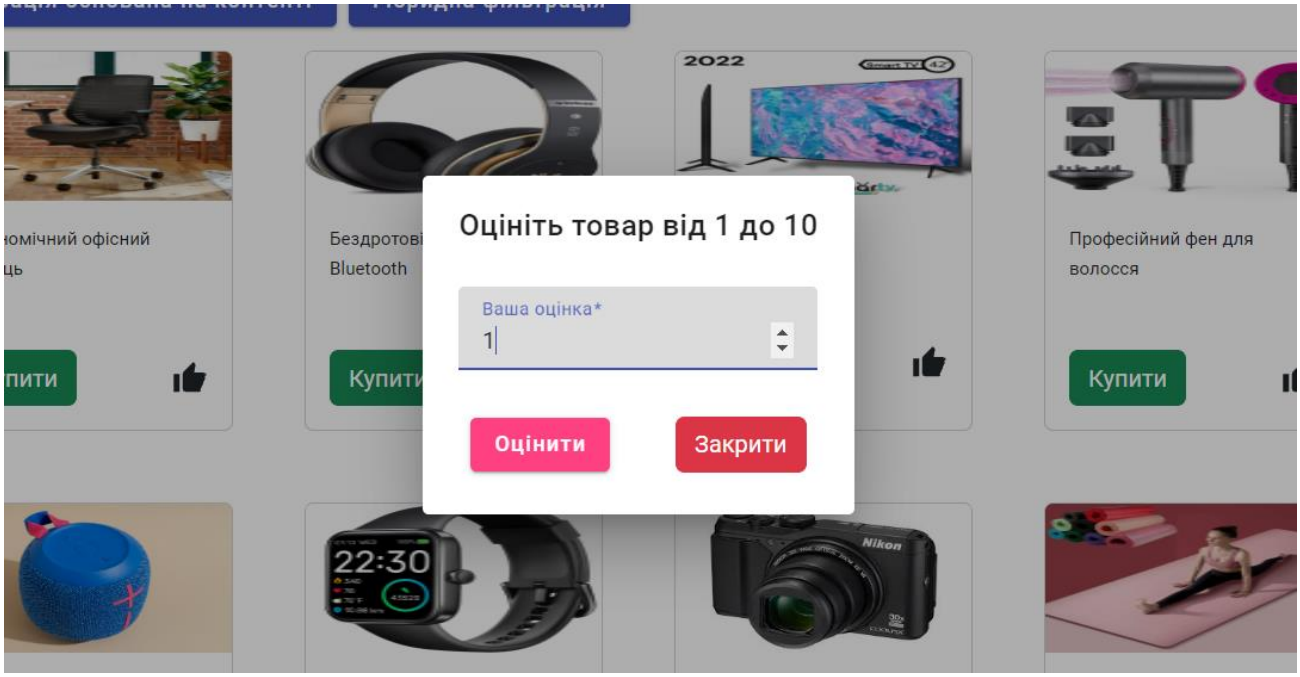


Рис.3.5 модальне вікно оцінки товару

Після успішного підтвердження відгук про товар – модальне вікно зачиняється.

3.2.1 Перевірка колаборативної фільтрації

Попередні дії ми виконали для користувача test7@gmail.com

Тепер ми можемо перевірити наскільки діє наш алгоритм для колаборативної фільтрації

Натискаємо кнопку колаборатива фільтрація, яка повинно викликати api для запуску алгорітма.

На кінці я подаю векторне представлення матриці , для перевірки точності данних рис.3.7

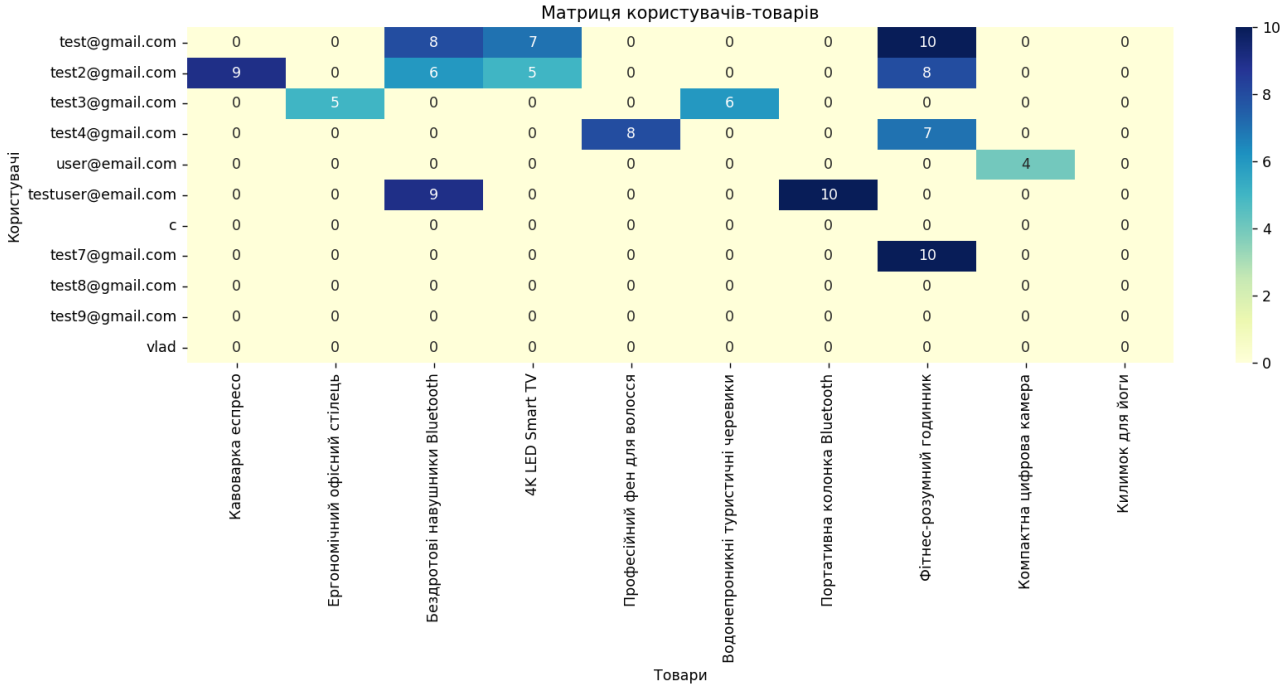


Рис.3.6 матриця даних колаборативного фільтрування

Процес колаборативної фільтрації:

1. **Збір оцінок**: Система збирає оцінки від різних користувачів для різних предметів (як вказано в матриці).
2. **Виявлення подібних користувачів**: Для кожного користувача система шукає інших користувачів зі схожими оцінками. Наприклад, якщо два користувачі високо оцінили одні й ті ж предмети, вони вважаються схожими.
3. **Визначення невідомих оцінок**: Використовуючи оцінки подібних користувачів, система передбачає, як високо користувач може оцінити предмет, для якого оцінка ще не була надана (поля з нулями в матриці).
4. **Рекомендація предметів**: На основі передбачених оцінок, система вибирає предмети з найвищими прогнозованими оцінками, щоб рекомендувати їх користувачеві.

Отже зважаючи на кроки та проаналізувавши рис. 3.7. можна прорахувати, щокористувач test@gmail.com та test7@gmail.com(на якому ми тестуємо) схожі,

отже товари користувача test@gmail.com мають бути зарекомендовані Wireless Bluetooth Headphones та 4K LED Smart TV. Після завершення виконання апі, ми відкриваємо модальне вікно, з рекомендуючими товарами рис.3.8

Рис.3.7 модальне вікно з рекомендаціями для колаборативної фільтрації

Можемо зробити висновок, що алгоритм виконується вірно.

3.2.2 Перевірка фільтрації на основі контенту

Модель товарів

Id	ідентифікатор товару
Name	назва
Category	категорії до яких входить товар
features	функціональність.
Image	посилання на картинку

Модель користуача

Id	ідентифікатор користувача
username	Ім’я користувача
password	Захешований пароль
preferences	Вподобання користувача
purchasedItems	Куплені товари
viewedItems	Товари які користувач дивився
ratingsGiven	Масив ідентифікаторів товарів, для яких користувач відгукнувся

З алгоритмом фільтрації на основі контенту можно ознайомитись у додадтку А.

Попередньо я вже обновив модель для testuser@email.com,

Додавши інформацію про його куплені, оцінені та переглянуті товари.

Виходячи з цього, в нього сформувались такі вподобання :

1)Електроніка

2) Фітнес

3) Кемпінг

Після заповнення даних, можемо починати тестування.

1) Натискаємо кнопку contentBasedFiltering.
2) Робимо запит на апі та очікуємо виконання.
3) Кроки виконання алгоритму:
    a. Аналіз характеристик предметів: Спочатку визначаються атрибути або характеристики - категорія, технічні специфікації тощо.
    b. Створення профілю користувача: Для кожного користувача на основі його попередніх оцінок створюється профіль, який показує його переваги щодо цих атрибутів. Наприклад, якщо користувач високо оцінив " Wireless Bluetooth Headphones " і "4K LED Smart TV", то профіль може підкреслити високу цінність для користувача технологічної складності та якості продукту.

c. Порівняння предметів з профілем: Коли користувач шукає новий предмет, система порівнює його атрибути з профілем користувача. Якщо характеристики предмета відповідають профілю, цей предмет може бути рекомендований.

d. Рекомендація предметів: На основі цього порівняння система вибирає предмети, які найкраще відповідають індивідуальним перевагам користувача, та рекомендує їх

4) Після виконання отримуємо рекомендації а відкриваємо модальне вікно з ними рис.3.9

Товар	Категорія
Bluetooth Portable Speaker	Електроніка
Yoga Mat	Фітнес, Кемпінг
Fitness Smartwatch	Фітнес, Електроніка
Waterproof Hiking Boots	Кемпінг
Compact Digital Camera	Електроніка

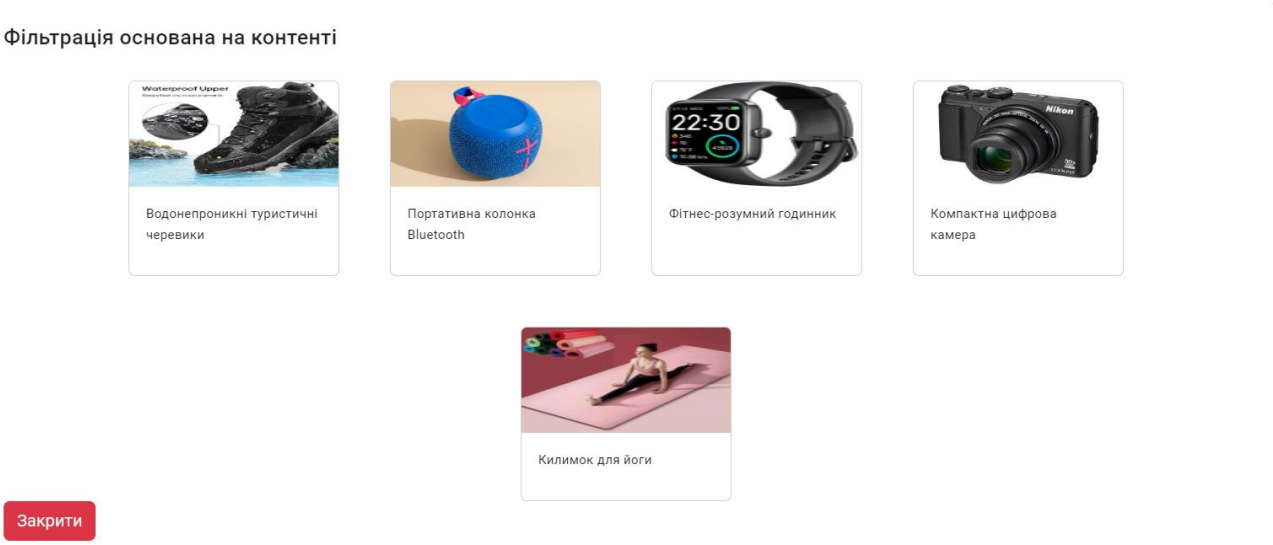


Рис.3.8 модальне вікно з рекомендаціями для фільтрації на основі контенту

Опираючись на побажання користувача та категоріями товарів які були рекомендовані, можемо зробити висновок, що алгоритм працює.

3.2.3 Перевірка фільтрації гібридної фільтрації

Гібридна фільтрація комбінує елементи колаборативної фільтрації та фільтрації на основі контенту для покращення рекомендаційних систем. Вона прагне уникнути обмежень, які мають ці два методи окремо, таких як проблема холодного старту в колаборативній фільтрації або обмежене різноманіття рекомендацій у фільтрації на основі контенту.

Кроки тестування:

1) Натискаємо кнопку hybridFiltering, та робимо запит на api.
2) Кроки виконання алгорітму:
    - Об'єднання інформації: Система використовує оцінки користувачів, щоб знайти подібності між ними (колаборативна фільтрація), а також аналізує характеристики предметів, які вони оцінили (фільтрації на основі контенту).
    - Створення профілю користувача: Профіль користувача формується на основі його власних оцінок та характеристик оцінених предметів, а також оцінок та вподобань подібних користувачів.
    - Поєднання рекомендацій: Система враховує як схожість з іншими користувачами, так і переваги користувача до певних характеристик предметів для формування рекомендацій.
    - Адаптація та оптимізація: Гібридна система може адаптуватися до змін у поведінці користувачів, динамічно оновлюючи рекомендації на основі нових даних.
    - Видача рекомендацій: Користувачу надаються рекомендації, які базуються на комплексному аналізі його минулих оцінок та вподобань, а також на вподобаннях користувачів з схожими інтересами

3) Після виконання api, отримуємо рекомендації та відкриваємо модальне вікно для відображення їх(див. рис.3.9). Дивлячись на результати колаборативна фільтрації та фільтрації на основі контенту , можемо зрозуміти, що ми отримали уніфікований результат рекомендацій виконання двох попередніх алгоритмів.

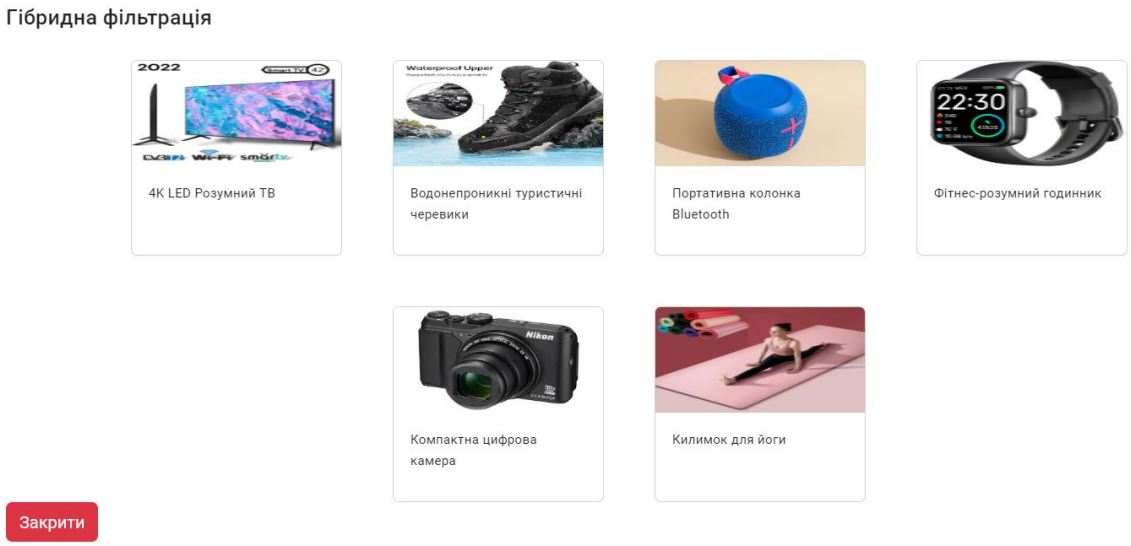


Рис. 3.9 модальне вікно з рекомендаціями для гібридної фільтрації

ВИСНОВКИ ДО РОЗДІЛУ

У цьому розділі висвітлюється процес реалізації та тестування веб-додатку з використанням трьох різних методів фільтрації для рекомендаційних систем: колаборативної фільтрації, фільтрації на основі контенту та гібридної фільтрації. Кожен метод пройшов кроки налаштування середовища, запуску веб-додатку, аутентикації користувача та роботи з модальними вікнами для входу, реєстрації та рекомендацій.

Колаборативна фільтрація використовувала оцінки користувачів для виявлення подібності між ними та рекомендувала продукти на основі цих подібностей. Фільтрація на основі контенту аналізувала характеристики товарів, які користувачі вже оцінили, та рекомендувала подібні товари, засновані на цих

характеристиках. Гібридна фільтрація поєднувала обидва ці підходи для створення більш точних та персоналізованих рекомендацій.

Тестування показало, що кожен із методів ефективно рекомендував товари, які відповідали інтересам та вподобанням користувачів, що підтверджується відображенням рекомендованих товарів у відповідних модальних вікнах. Отже, можна зробити висновок, що реалізовані алгоритми фільтрації ефективні та можуть бути використані для покращення досвіду користувачів у веб-додатку.

ВИСНОВКИ

У даній роботі було проведено дослідження та розроблено систему рекомендацій товарів, використовуючи різні алгоритми, зокрема колаборативну фільтрацію та алгоритм заснований на контенті. Розглянуті алгоритми були порівняні за їхньою ефективністю та перевагами.

У першому розділі проведено аналіз різних видів систем рекомендацій, включаючи колаборативну фільтрацію, алгоритми засновані на контенті та гібрідній фільтрації.В результаті порівняльного аналізу були виділені переваги та недоліки кожного з цих підходів.

У другому розділі була представлена структура розробленої системи, включаючи вибір мови програмування, середовища розробки та використання сторонніх бібліотек. Описано архітектуру фронтед та бекенд застосунку та використані сучасні технології розробки.

У третьому розділі було надано математичну постановку задач із детальним поясненням математичного підґрунтя стосовно реалізації алгоритму.

У четвертому розділі проведено експериментальне дослідження та перевірку правильності роботи алгоритмів та застосунку. Демонструється коректна робота логіки програми.

Результатом роботи є працездатна система рекомендацій товарів, яка використовує алгоритми машинного навчання та може бути використана для побудови веб-застосунку. Система дозволяє користувачам отримувати персоналізовані рекомендації щодо товарів у магазині. Розроблене рішення є актуальним та має потенціал для подальшого розвитку та масштабування.

Таке поєднання сучасних методів веб-розробки з математичним підґрунтям дозволяє створити високоефективну систему рекомендацій для електронного магазину та забезпечити покращення користувацького досвіду та збільшення обсягів продажів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1) Документація Angular [Електронний ресурс] - Режим доступу https://angular.dev/
2) Документація [Електронний ресурс] - Режим доступу Express JS
3) Річчі, Ф., Рокач, Л., і Шапіра, Б. Посібник з систем рекомендацій. Спрінгер., 2015 р., 1003 с.
4) Белл, Р. М., Корен, Й., та Волінський, К. Моделювання взаємозв'язків на різних рівнях для підвищення точності великих систем рекомендацій. KDD '07, 2007, с.104
5) Лопс, П., де Гемміс, М., та Семераро, Г. Системи рекомендацій на основі контенту: стан мистецтва та тенденції. Посібник з систем рекомендацій, Спрінгер., 2011, 1003.с
6) Аггарвал, К. К.. Системи рекомендацій: Підручник. Спрінгер., 2016, 498 с.
7) Корен, Й., Белл, Р., та Волінський, К. , Техніки матричної факторизації для систем рекомендацій. Комп'ютер, 2007 47с..
8) Лінден, Г., Сміт, Б., та Йорк, Дж. (2003). Рекомендації Amazon.com: Колаборативне фільтрування від предмету до предмету. IEEE Інтернет-обчислення, 2003 р., 80 с..
9) Сарвар, Б., Каріпіс, Г., Констан, Дж., та Рідл, Дж. Алгоритми рекомендаційних систем на основі колаборативного фільтрування предметів. WWW '01., 2001 р., 295 с.
10) Бобаділла, Дж., Ортега, Ф., Ернандо, А., та Гутьєррес, А. Огляд систем рекомендацій. Системи на основі знань, 2013 р., 46 с.
11) Шафер, Дж. Б., Констан, Дж. А., та Рідл, Дж. (2001). Рекомендаційні застосування в електронній комерції. Майнінг та відкриття знань, 2001 р., 5 с.
12) Бурк, Р Гібридні рекомендаційні системи: Огляд та експерименти. Моделювання користувачів та адаптована взаємодія, 2002 р. 12с.

13) Адомавічус, Г., та Тузілін, А. До наступного покоління рекомендаційних систем: огляд сучасного стану мистецтва та можливі розширення. IEEE Транзакції знань та інженерії даних, 2005, 17с.

14) Рекомендаційна система [Електронний ресурс] - Режим доступу Wikipedia

15) Surprise - бібліотека для побудови та аналізу систем рекомендацій, що працюють з явними даними рейтингів. [Електронний ресурс] Режим доступу https://github.com/NicolasHug/Surprise

16) Scikit-learn - машина для навчання з великим набором алгоритмів, включаючи рекомендаційні системи. [Електронний ресурс] Режим доступу https://github.com/scikit-learn/scikit-learn

17) Pandas - бібліотека для аналізу даних, яка пропонує зручні структури даних та операції для маніпулювання числовими таблицями та часовими рядами. [Електронний ресурс] Режим доступу https://github.com/pandas-dev/pandas

18) Numpy - основна бібліотека для наукових обчислень у Python, що підтримує великі багатовимірні масиви та матриці. [Електронний ресурс] Режим доступу https://github.com/numpy/numpy

Додаток А

Програмний код

1) Колаборативна фільтрація:

```
import json
import sys
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics.pairwise import cosine_similarity
from scipy import sparse
import numpy as np


def collaborative_filtering(user_id, users, items, ratings, N=1):
    item_ids = [item['_id'] for item in items]
    # Витягуємо ID користувачів зі списку користувачів
    user_ids = [user['_id'] for user in users]
    item_names = [item['name'] for item in items]
    # Витягуємо імена користувачів зі списку користувачів
    user_names = [user['username'] for user in users]

    try:
        # Створення матриці користувач-продукт
        matrix = []
        for user in users:
            if '_id' not in user:
                raise ValueError(f"Відсутній '_id' у користувача: {user}")
            row = []
            for item in items:
                if '_id' not in item:
                    raise ValueError(f"Відсутній '_id' у товара: {item}")
                # Шукаємо рейтинг, який відповідає даному користувачу та продукту
                rating = next((r['rating'] for r in ratings if str(r['userId']) == str(user['_id']) and str(r['itemId']) == str(item['_id'])), 0)
                row.append(rating)
            matrix.append(row)

        # Перетворення матриці на розріджену для підвищення ефективності
        matrix_sparse = sparse.csr_matrix(matrix)

        # Розрахунок косинусної подібності між користувачами
        similarities = cosine_similarity(matrix_sparse)

        # Визначення індексу даного користувача
        user_index = next(i for i, user in enumerate(users) if str(user['_id']) == user_id)

        # Визначення індексів N найбільш схожих користувачів
        most_similar_user_indices = similarities[user_index].argsort()[-(N+1):-1]

        # Визначення товарів, які оцінили ці користувачі, але не оцінив поточний користувач
        unrated_items = [(i, item['_id']) for i, item in enumerate(items) if matrix[user_index][i] == 0 and any(matrix[other_user_index][i] > 0 for other_user_index in most_similar_user_indices)]
        # Отримання ID п'яти найкращих товарів
        recommendations = [str(item_id) for _, item_id in unrated_items[:5]]
```

```
50)
51)         # Виведення рекомендацій
52)         print(json.dumps(recommendations))
53)
54)         # Після отримання рекомендацій конвертуємо матрицю для візуалізації
55)         ratings_matrix = np.array(matrix)
```

2) Фільтрація на основі контенту

```
3)  import sys
4)  import json
5)  from sklearn.feature_extraction.text import TfidfVectorizer
6)  from sklearn.metrics.pairwise import cosine_similarity

7)  def generate_user_profile(user, items, ratings):
8)  # Об'єднання вподобань користувача, переглянутих, куплених та вподобаних товарів
9)  interests = user.get('preferences', [])
10) interests += [item['name'] for item in items if item['_id'] in user.get('viewedItems', []) +
    user.get('purchasedItems', []) + user.get('likedItems', [])]
11) return ' '.join(interests)

12) def content_based_filtering(user_id, users, items, ratings):
13) user = next((u for u in users if u['_id'] == user_id), None)
14) if not user:
15) raise ValueError(f"Користувача з ID {user_id} не знайдено")

16) user_profile = generate_user_profile(user, items, ratings)

17) # Векторизація
18) tfidf_vectorizer = TfidfVectorizer(stop_words='english')
19) item_tfidf_matrix = tfidf_vectorizer.fit_transform([item['description'] for item in items])
20) user_tfidf = tfidf_vectorizer.transform([user_profile])

21) # Розрахунок схожості
22) cosine_similarities = cosine_similarity(user_tfidf, item_tfidf_matrix).flatten()
23) recommended_item_indices = cosine_similarities.argsort()[-5:][::-1]
24) recommended_items = [items[index]['_id'] for index in recommended_item_indices]

25) return recommended_items

26) if __name__ == '__main__':
27) user_id = sys.argv[1]
28) users = json.loads(sys.argv[2])
29) items = json.loads(sys.argv[3])
30) ratings = json.loads(sys.argv[4])

31) try:
32) recommendations = content_based_filtering(user_id, users, items, ratings)
33) print(json.dumps(recommendations))
34) except Exception as e:
35) print(f"Помилка: {str(e)}", file=sys.stderr)
36) sys.exit(1)
37)
```

3) Гібридна фільтарція

```
4)  export const hybridFiltering = async (req, res) => {
5)    try {
6)      const userId = req.user._id;
7)      const users = await User.find({});
8)      const items = await Item.find({});
9)      const ratings = await Rating.find({});
10)
11)     // Запуск скрипта для колаборативної фільтрації
12)     const collaborativePython = spawn('python', ['-u', './python/collaborative_filtering.py', userId,
    JSON.stringify(users), JSON.stringify(items), JSON.stringify(ratings)]);
13)     let collaborativeData = '';
14)
15)     // Запуск скрипта для контент-орієнтованої фільтрації
16)     const contentBasedPython = spawn('python', ['-u', './python/content_based.py', userId, JSON.stringify(users),
    JSON.stringify(items), JSON.stringify(ratings)]);
17)     let contentBasedData = '';
18)
19)     collaborativePython.stdout.on('data', (data) => {
20)       collaborativeData += data.toString();
21)     });
22)
23)     contentBasedPython.stdout.on('data', (data) => {
24)       contentBasedData += data.toString();
25)     });
26)
27)     // Обробка завершення обох скриптів
28)     Promise.all([
29)       new Promise((resolve, reject) => {
30)         collaborativePython.on('close', (code) => {
31)           if (code !== 0) reject('Помилка у скрипті колаборативної фільтрації');
32)           else resolve();
33)         });
34)       }),
35)       new Promise((resolve, reject) => {
36)         contentBasedPython.on('close', (code) => {
37)           if (code !== 0) reject('Помилка у скрипті контент-орієнтованої фільтрації');
38)           else resolve();
39)         });
40)       })
41)     ]).then(() => {
42)       // Комбінування результатів обох методів
43)       const collaborativeRecommendations = JSON.parse(collaborativeData);
44)       const contentBasedRecommendations = JSON.parse(contentBasedData);
45)
46)       // Приклад комбінації: об'єднання та видалення дублікатів
47)       const combinedRecommendations = [...new Set([...collaborativeRecommendations,
    ...contentBasedRecommendations])];
48)
49)       res.status(201).json(combinedRecommendations);
50)     }).catch((error) => {
51)       console.error(error);
52)       res.status(500).send('Помилка при обробці гібридних рекомендацій.');
53)     });
54)   } catch (error) {
55)     console.error('Помилка сервера:', error);
56)     res.status(500).send('Внутрішня помилка сервера.');
57)   }
58) };
```

Додаток Б

Стаття

величини міжфазного натягу, об'ємної концентрації полімеру дисперсної фази, та в'язкостей вхідних компонентів. Крім того, створене програмне забезпечення має зручний інтерфейс і дає можливість зручно візуалізувати результати проведених розрахунків.

Висновки

Виконана робота підтверджує можливість використання підходів класичної механіки для опису реологічної поведінки розплавів сумішей полімерів. Розроблено програмне забезпечення, яке здійснює візуалізацію процесу утворення мікрофібрилярних структур, що може бути використано для легкого і економічного вивчення досліджуваного процесу, а також зручного і наочного його прогнозування.

Ключові слова: програмне забезпечення, мікрофібрилярні структури, математична модель, графічний інтерфейс.

Література

1. Резанова В.Г., Резанова Н.М. Програмне забезпечення для дослідження полімерних систем. Монографія. – К.: АртЕк, 2020. – 358 с.
2. Резанова В.Г., Резанова Н.М. Програмне забезпечення для оптимізації складу багатокомпонентних сумішей. Монографія.- К.:АртЕк. - 2022. 315с.
3. Stroustrup B. Programming: Principles and Practice Using C++ (2nd Edition). Addison-Wesley Professional, 2014. – 1312 p.
4. Stroustrup B. The C++ Programming Language Fourth Edition. Addison-Wesley, 2013. – 1366 p.

КУРОЧКА В.О., РЕЗАНОВА В.Г.

РОЗРОБКА ВЕБ-СЕРВІСУ РЕКОМЕНДАЦІЙ НА ОСНОВІ АНАЛІЗУ ПОВЕДІНКОВИХ ДАНИХ КОРИСТУВАЧІВ

KUROCHKA V.O., REZANOVA V.G.

DEVELOPMENT OF A WEB-BASED RECOMMENDATION SERVICE BASED ON THE ANALYSIS OF USER BEHAVIORAL DATA

In today's information society, web services that provide personalized information and recommendations have become essential and popular. With the advancement of data analysis and machine learning technologies, developing web recommendation services based on user behavior analysis is a relevant task in the field of information technology. This article discusses the development of a recommendation web service that analyzes user behavior data to provide personalized recommendations, aiming to enhance user experience and meet their individual needs.

The problem statement emphasizes the need to develop a web service for recommendations based on user behavior analysis. The tasks involved in achieving this goal include data collection and processing, data analysis, development of a recommendation algorithm, and the creation of a user-friendly web interface.

The main task of this project is to develop a web service for recommendations based on user behavior analysis. The service aims to improve the user experience and satisfy individual

needs by providing personalized recommendations that consider user preferences, activities, and context.

In conclusion, the development of a web service for recommendations based on user behavior analysis is an important and timely task in today's information society. Leveraging user behavior data and machine learning algorithms enables the improvement of user experience and the provision of personalized recommendations. Developing such a service involves data collection and processing, algorithm development, and the implementation of a user-friendly web interface. Further advancements in these services can enhance personalization and effectiveness in information retrieval for users.

Вступ

У сучасному інформаційному суспільстві веб-сервіси, що надають користувачам персоналізовану інформацію та рекомендації, стали необхідними та популярними. Завдяки розвитку технологій аналізу даних та машинного навчання, розробка веб-сервісів рекомендацій на основі аналізу поведінкових даних користувачів є актуальною задачею в галузі інформаційних технологій. В даній статті розглянуто проект розробки веб-сервісу рекомендацій, що здатний аналізувати поведінкові дані користувачів та надавати персоналізовані рекомендації з метою поліпшення користувацького досвіду та задоволення їхніх індивідуальних потреб.

Постановка завдання

Основною метою даного проекту є розробка веб-сервісу рекомендацій на основі аналізу поведінкових даних користувачів. Задачі, що повинні бути виконані для досягнення цієї мети, включають:

1. Збір та обробка даних: Збір та збереження поведінкових даних користувачів, таких як перегляди сторінок, взаємодія з контентом, історія покупок тощо. Дані повинні бути збережені в структурованому форматі та готові до подальшого аналізу.
2. Аналіз даних: Використання методів аналізу даних для виявлення патернів та залежностей в поведінці користувачів. Цей етап включає в себе застосування алгоритмів машинного навчання, статистичних методів та інших інструментів для виявлення корисної інформації з великого обсягу даних.
3. Розробка рекомендаційного алгоритму: Розробка алгоритму, який здатний генерувати персоналізовані рекомендації на основі аналізу поведінкових даних. Цей алгоритм повинен враховувати індивідуальні вподобання користувачів, їхній контекст та змінювати рекомендації в залежності від зміни поведінки користувача.

4. Розробка веб-інтерфейсу: Створення інтуїтивно зрозумілого веб-інтерфейсу, який дозволить користувачам легко використовувати сервіс та отримувати персоналізовані рекомендації.

Основна частина

Основною задачею проекту є розробка веб-сервісу рекомендацій на основі аналізу поведінкових даних користувачів. Для досягнення цієї мети можна використовувати такі технології та кроки дій, з використанням Angular:

1. Збір та обробка даних: Для збору поведінкових даних користувачів можна використовувати різноманітні інструменти, такі як аналітика веб-сторінок, сеанси взаємодії з користувачами, історії покупок тощо. Дані можуть бути збережені у базі даних для подальшої обробки. Для обробки та аналізу даних можна використовувати мови програмування, такі як Python, та бібліотеки для аналізу даних, наприклад, Pandas та NumPy.

2. Аналіз поведінкових даних: Після збору даних можна використовувати методи аналізу даних для виявлення патернів та залежностей в поведінці користувачів. Це може включати використання статистичних методів, алгоритмів машинного навчання, наприклад, класифікації, кластеризації або рекомендаційних алгоритмів.

3. Розробка рекомендаційного алгоритму: На основі аналізу поведінкових даних можна розробити рекомендаційний алгоритм, який здатний генерувати персоналізовані рекомендації для кожного користувача. Цей алгоритм може використовувати методи колаборативного фільтрування, контентного аналізу або гібридні підходи. Для реалізації алгоритму можна використовувати мову програмування Python та спеціалізовані бібліотеки, наприклад, scikit-learn або TensorFlow.

4. Розробка веб-інтерфейсу з використанням Angular: Для надання користувачам зручного доступу до сервісу та отримання персоналізованих рекомендацій можна використовувати фреймворк Angular. Angular дозволяє розробляти потужні односторінкові додатки з динамічною взаємодією з користувачем. Він може бути використаний для створення фронтенду веб-інтерфейсу вашого веб-сервісу рекомендацій.

Таким чином, для розробки веб-сервісу рекомендацій з використанням Angular можна використовувати технології, такі як Python, Pandas, NumPy, scikit-learn, TensorFlow та Angular. Кожен крок дій

передбачає використання відповідних інструментів та технологій для досягнення поставленої мети проекту.

Висновки

Розробка веб-сервісу рекомендацій на основі аналізу поведінкових даних користувачів є актуальною та важливою задачею в сучасному інформаційному суспільстві. Використання поведінкових даних та алгоритмів машинного навчання дозволяє покращити користувацький досвід та забезпечити персоналізовані рекомендації для кожного користувача. Розробка веб-сервісу рекомендацій вимагає збору та обробки даних, розробки алгоритмів та реалізації зручного веб-інтерфейсу. Подальше вдосконалення таких сервісів може сприяти покращенню персоналізації та ефективності інформаційного отримання користувачами.

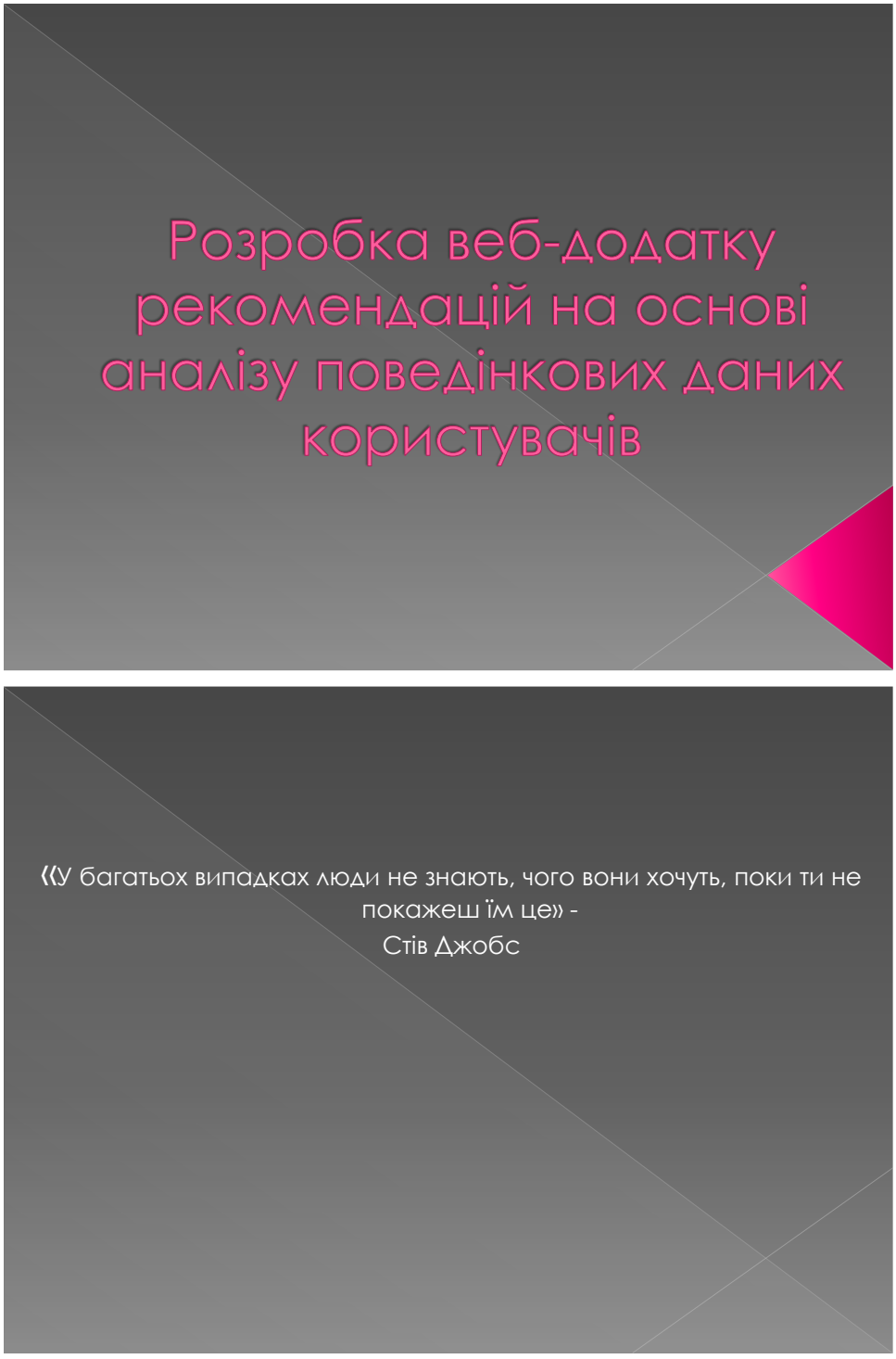
Ключові слова: програмне забезпечення, веб-сервіси рекомендацій, поведінкові дані користувачів, персоналізовані рекомендації, збір та обробка даних, аналіз даних, рекомендаційний алгоритм, веб-інтерфейс, користувацький досвід, індивідуальні потреби, поведінкові патерни, машинне навчання, статистичні методи, контекст, покращення персоналізації, ефективність.

Література

1. Resnick, P., & Varian, H. R. (1997). Рекомендаційні системи. Комунікації ACM, 40(3), 56-58.
2. Adomavicius, G., & Tuzhilin, A. (2005). До наступного покоління рекомендаційних систем: огляд сучасного стану та можливих розширень. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734-749.
3. Ricci, F., Rokach, L., & Shapira, B. (2015). Посібник з рекомендаційних систем. Springer.
4. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Оцінка рекомендаційних систем з колаборативним фільтруванням. ACM Transactions on Information Systems, 22(1), 5-53.
5. Burke, R. (2002). Гібридні рекомендаційні системи: огляд та експерименти. User Modeling and User-Adapted Interaction, 12(4), 331-370.
6. Koren, Y., Bell, R., & Volinsky, C. (2009). Техніки матричного розкладання для рекомендаційних систем. Computer, 42(8), 30-37.
7. Angular Отримано з https://angular.io/
8. van Rossum, G., & Drake, F. L. (2009). Посібник з Python 3. CreateSpace.

Додаток В

Презентація



Актуальність теми

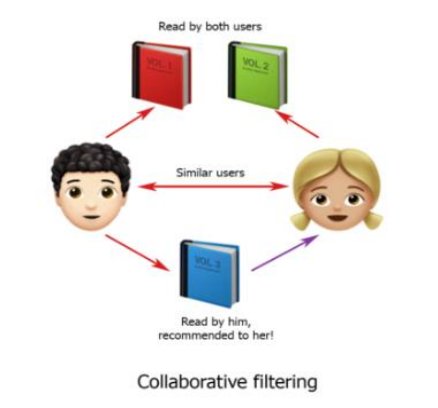
- Сучасний світ неможливо уявити без веб-додатків, які відіграють важливу роль в нашому повсякденному житті та бізнесі. Рекомендаційні системи є ключовою складовою веб-додатків, оскільки вони забезпечують персоналізований досвід користувачів, підвищуючи задоволеність та лояльність клієнтів, а також сприяють зростанню продажів.

Основні види систем рекомендацій

- **Колаборативна фільтрація**
- **Засновані на контенті**
- **Гібридні системи**

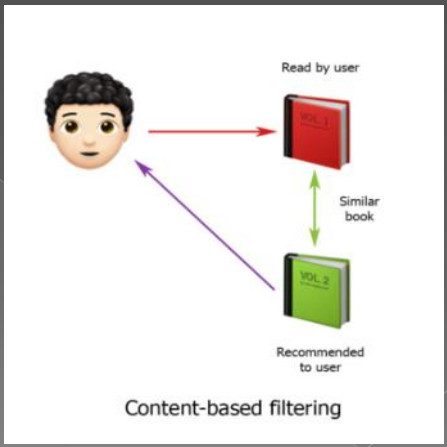
Колаборативна фільтрація

Цей алгоритм працює шляхом збору та аналізу інформації про переваги користувачів. Він виявляє схожості між користувачами або між продуктами на основі їхніх покупок або оцінок. На підставі цього, система може рекомендувати продукти, які користувалися популярністю серед схожих користувачів або які є схожими на ті, які користувач вже оцінив позитивно.



Засновані на контенті

- Підхід, що базується на аналізі вмісту товарів, використовує детальний розгляд атрибутів продуктів для створення рекомендацій. Через ретельний аналіз описів продуктів, ключових слів та інших важливих характеристик, алгоритм виявляє подібності між різними предметами. Заснований на виявленій схожості, алгоритм надає користувачам пропозиції товарів, що відповідають раніше виявленим уподобанням.



Гібридизація систем рекомендацій означає інтеграцію різноманітних методологій та алгоритмів для створення бі

Гібридні системи

- Гібридизація систем рекомендацій означає інтеграцію різноманітних методологій та алгоритмів для створення більш досконалих та індивідуалізованих пропозицій для користувачів. Використовуючи силу комбінованих стратегій, такі системи здобувають широке визнання у сфері цифрової комерції та інтернет-сервісів.
- Сутність гібридних систем полягає в тому, що вони аплікують мікс різних моделей рекомендацій — від колаборативної фільтрації до контент-базованих алгоритмів.

Технології для розробки

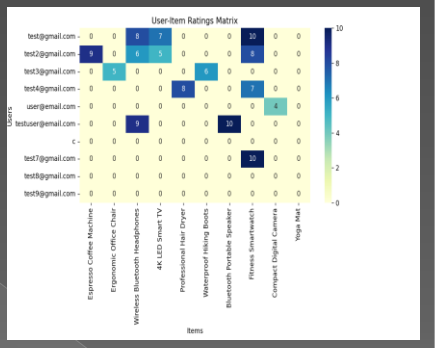
- Javascript
- TypeScript
- Node.js
- Express Js
- Python

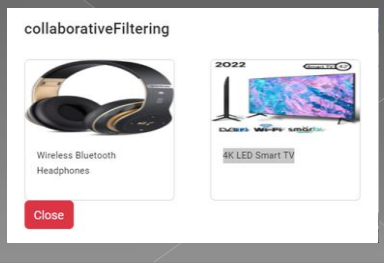
Методики та Алгоритми для реалізації застосунку

- **Тематична модель (Topic Modeling):**
  - Автоматичний аналіз тексту для виявлення основних тем.
  - Застосування: обробка природної мови, аналіз соціальних медіа.
- **Векторні представлення (Vector Representations):**
  - Перетворення слов і фраз у числові вектори для аналізу тексту.
  - Застосування: машинний переклад, класифікація текстів.
- **Розклад матриць (Matrix Factorization, MF):**
  - Розкладання великих матриць на менші для виявлення прихованих особливостей.
  - Застосування: прогнозування вподобань у рекомендаційних системах.
- **Сингулярне Розкладання Значень (SVD):**
  - Математичний аналіз для виявлення внутрішньої структури даних.
  - Застосування: розпізнавання образів, статистичний аналіз.
- **Косинусна Схожість:**
  - Вимірювання схожості між користувачами чи між користувачем та товаром.
  - Застосування: персоналізація пропозицій у рекомендаційних системах.

Перевірка колаборативної фільтрації

- 1) Авентифікація (Користувач: test7@gmail.com
- 2)Дія: Натискання кнопки collaborativeFiltering для активації алгоритму.
- **3)Процес Колаборативної Фільтрації:**
  - **Збір Оцінок:** Аналіз оцінок користувачів по різним товарам.
  - **Виявлення Подібних Користувачів:** Пошук користувачів зі схожими вподобаннями.
  - **Визначення Невідомих Оцінок:** Прогнозування оцінок для невідомих товарів.
  - **Рекомендація Предметів:** Вибір товарів з найвищими прогнозованими оцінками.
- **Результати:**
  - Рекомендовані товари: Wireless Bluetooth Headphones, 4K LED Smart TV.
  - Візуалізація рекомендацій у модальному вікні





Перевірка фільтрації на основі контенту

Модель товарів

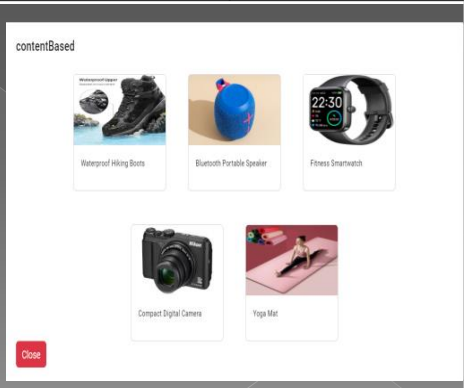
Id	ідентифікатор товару
Name	назва
Category	категорії до яких входить товар
features	функціональність.
Image	посилання на картинку

Модель користуача

Id	ідентифікатор користувача
username	Ім'я користувача
password	Захешований пароль
preferences	Вподобання користувача
purchasedItems	Куплені товари
viewedItems	Товари які користувач дивився
ratingsGiven	Масив ідентифікаторів товарів, для яких користувач відгукнувся

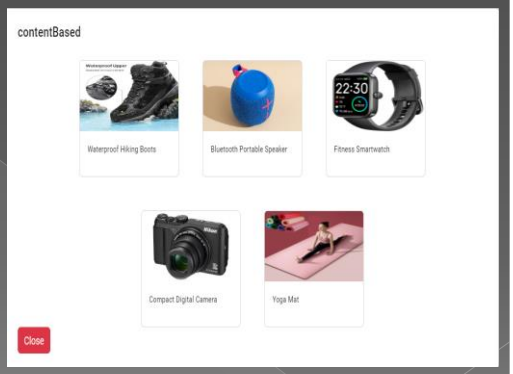
- **Тестування з Користувачем:** testuser@email.com
- **Підготовка:** Оновлення моделі користувача з даними про куплені, оцінені, та переглянуті товари.
- **Вподобання Користувача:**
  - Електроніка
  - Фітнес
  - Кемпінг
- **Процедура Тестування:**
  - Активація: Натиснення кнопки contentBasedFiltering.
  - API запит: Очікування виконання алгоритму.
  - Кроки Алгоритму: a. Аналіз характеристик товарів. b. Створення профілю користувача. c. Порівняння товарів з профілем користувача. d. Рекомендація товарів.
  - Результати: Відображення рекомендованих товарів у модальному вікні (Рис.3.9).
- **Рекомендовані Товари:**
  - Bluetooth Portable Speaker (Електроніка)
  - Yoga Mat (Фітнес, Кемпінг)
  - Fitness Smartwatch (Фітнес, Електроніка)
  - Waterproof Hiking Boots (Кемпінг)
  - Compact Digital Camera (Електроніка)

Товар	Категорія
Bluetooth Portable Speaker	Електроніка
Yoga Mat	Фітнес, Кемпінг
Fitness Smartwatch	Фітнес, Електроніка
Waterproof Hiking Boots	Кемпінг
Compact Digital Camera	Електроніка



Перевірка фільтрації гібридної фільтрації

- **Активація Алгоритму:** Натискання кнопки hybridFiltering та виконання API запиту.
- **Процес Гібридної Фільтрації:**
  - Об'єднання інформації з колаборативної та контентної фільтрації.
  - Створення комплексного профілю користувача.
  - Поєднання рекомендацій, враховуючи схожості з іншими користувачами та індивідуальні переваги.
  - Адаптація та оптимізація рекомендацій на основі нових даних.
  - Видача комплексних рекомендацій.



Висновок

У цій роботі було розроблено систему рекомендацій товарів, використовуючи різноманітні алгоритми, включно з колаборативною фільтрацією та контент-базованим підходом. Було проведено порівняльний аналіз цих алгоритмів, визначено їх ефективність та переваги. Описано структуру системи, включаючи вибір технологій та архітектуру, а також детально розглянуто математичну базу алгоритмів. Експериментальне дослідження підтвердило правильність роботи алгоритмів. Результатом стала функціональна система рекомендацій, готова до використання у веб-застосунках, яка забезпечує персоналізовані рекомендації і має потенціал для подальшого розвитку. Ця система поєднує сучасні методи веб-розробки з математичним підґрунтям для покращення користувацького досвіду та збільшення продажів.