Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Комп'ютерного моделювання та інформаціних технологій
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти перший (магістерський)

Дослідження підходів до побудови рекомендаційної системи маркетплейсу з використанням машинного навчання та поведінкових і контекстних даних користувачів
(тема)

Виконав:

здобувач 2 курсу, групи *СПРм-24-1*
Лавріненко В.В.
(прізвище, ініціали)

Спеціальність *122 – Комп'ютерні науки*
(код і назва спеціальності)

Тип програми *освітньо-наукова*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Системне проектування*

Керівник проф. каф. СТ Міщеряков Ю.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ *проф. Гребеннік І.В.*
(підпис) (прізвище, ініціали)

2026 р.

Я, як студент ХНУРЕ розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

19.05.2026 Лавріненко В.В,

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведено

Керівник кваліфікаційної роботи Міщеряков Ю.В.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Системотехніки

Рівень вищої освіти *перший (бакалаврський)*

Спеціальність *122 – Комп'ютерні науки*

(код і повна назва)

Освітня програма *Комп'ютерні науки та технології*

(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«_____»_____2026р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Лавріненку Владиславу Валерійовичу

(прізвище, ім'я, по батькові)

1. Тема роботи: Дослідження підходів до побудови рекомендаційної системи маркетплейсу з використанням машинного навчання та поведінкових і контекстних даних користувачів

затверджена наказом по університету від

2. Термін здачі студентом роботи до екзаменаційної комісії 10 червня 2025 р.

3. Вихідні дані до проекту _____

4. Перелік питань, що потрібно опрацювати в роботі: аналіз предметної області, характеристика предметної області маркетплейсів та особливості взаємодії користувачів з товарами, роль рекомендаційних систем у маркетплейсах та їх вплив на персоналізацію сервісу, поведінкові та контекстні дані користувачів як основа для формування рекомендацій, аналіз підходів до побудови рекомендаційних систем, огляд існуючих реалізованих рекомендаційних систем у сучасних маркетплейсах, порівняльний аналіз підходів та обґрунтування вибору трьох підходів для реалізації, постановка задачі кваліфікаційної роботи, проєктування інформаційної системи, формування функціональних вимог до системи маркетплейсу, визначення критеріїв та метрик оцінювання якості рекомендацій, визначення архітектурних особливостей реалізації системи на платформі .net, проєктування структури бази даних маркетплейсу, підготовка набору даних для тестування рекомендаційних алгоритмів, розробка рекомендаційної системи, реалізація базового функціоналу маркетплейсу на платформі .net, наповнення бази даних тестовими даними, реалізація першого підходу до формування рекомендацій, реалізація другого підходу до формування рекомендацій, реалізація третього підходу до формування рекомендацій, інтеграція рекомендованих товарів у функціонал маркетплейсу, експериментальне дослідження та порівняльний аналіз реалізованих підходів, організація експериментального дослідження, проведення експериментів для кожного з реалізованих підходів, аналіз отриманих результатів, порівняння ефективності підходів за обраними метриками, визначення переваг та недоліків кожного

підходу.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів).

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни	Примітка
1	*Аналіз предметної області маркетплейсів*	*26.01.2026 – 01.02.2026*	*Виконано*
2	*Поведінкові та контексті дані користувачів я основа для формування рекомендацій*	*01.02.2026 – 07.02.2026*	*Виконано*
3	*Аналіз підходів до побудови рекомендацій*	*07.02.2026 – 16.02.2026*	*Виконано*
4	*Огляд реалізованих систем*	*16.02.2026 – 21.02.2026*	*Виконано*
5	*Постановка задачі кваліфікаційної роботи*	*21.02.2026 – 23.02.2026*	*Виконано*
6	*Формування функціональних вимог*	*23.02.2026 – 12.03.2026*	*Виконано*
7	*Визначення критеріїв оцінювання якості рек.*	*12.03.2026 – 15.03.2026*	*Виконано*
8	*Проектування структури БД*	*15.03.2026 – 18.03.2026*	*Виконано*
9	*Підготовка набору даних для тестування*	*18.03.2026 – 20.03.2026*	*Виконано*
10	*Реалізація базового функціоналу маркетплейсів*	*20.03.2026 – 16.04.2026*	*Виконано*
11	*Реалізація трьох підходів до формуванн рекомендацій*	*16.04.2026 – 18.04.2026*	*Виконано*
12	*Організація експериментального дослідження*	*18.04.2026 – 27.04.2026*	*Виконано*
13	*Проведення експериментів*	*27.04.2026 – 29.04.2026*	*Виконано*
14	*Аналіз результатів*	*29.04.2026 – 10.05.2026*	*Виконано*
15	*Порівняння ефективності підходів*	*10.05.2026 – 12.05.2026*	*Виконано*
16	*Визначення переваг та недоліків кожного підходу*	*12.05.2026 – 13.05.2026*	*Виконано*

Дата видачі завдання 26 січня 2026 р.

Здобувач _____ Лавріненко В.В.

Керівник роботи _____ Міщеряков Ю.В.

РЕФЕРАТ

Звіт кваліфікаційної роботи: 126 с., 3 табл., 11 рис, 26 дж.

АЛГОРИТМ РЕКОМЕНДАЦІЙ, БАЗА ДАНИХ, ГІБРИДНИЙ ПІДХІД, ЕЛЕКТРОННА КОМЕРЦІЯ, КОЛАБОРАТИВНА ФІЛЬТРАЦІЯ, КОНТЕНТНО-ОРІЄНТОВАНІ РЕКОМЕНДАЦІЇ, МАРКЕТПЛЕЙС, ПЕРСОНАЛІЗАЦІЯ, РЕКОМЕНДАЦІЙНА СИСТЕМА, CONTEXT-AWARE RECOMMENDATION, HYBRID RECOMMENDATION SYSTEM, MARKETPLACE, RECOMMENDER SYSTEM.

Об’єкт дослідження – процес формування персоналізованих рекомендацій у системах електронної комерції.

Предмет дослідження – методи, алгоритми та програмні засоби реалізації рекомендаційних систем у маркетплейсах.

Мета роботи – дослідження та реалізація підходів до формування рекомендацій у системі маркетплейсу, а також проведення порівняльного аналізу ефективності реалізованих алгоритмів за допомогою метрик оцінювання якості рекомендацій.

Результатом роботи є реалізована серверна частина маркетплейсу з інтегрованою рекомендаційною підсистемою, яка підтримує декілька підходів до формування рекомендацій, а також механізм автоматизованого експериментального оцінювання ефективності рекомендаційних алгоритмів.

ABSTRACT

Content: 126 p., 3 tab., 11 fig., 26 sourses.

COLLABORATIVE FILTERING, CONTENT-BASED RECOMMENDATIONS, CONTEXT-AWARE RECOMMENDATION, DATABASE, ELECTRONIC COMMERCE, HYBRID APPROACH, HYBRID RECOMMENDATION SYSTEM, MARKETPLACE, PERSONALIZATION, RECOMMENDER SYSTEM, RECOMMENDATION ALGORITHM.

The object of research is the process of generating personalized recommendations in e-commerce systems.

The subject of research is methods, algorithms, and software tools for implementing recommender systems in marketplaces.

The purpose of the work is to study and implement approaches to recommendation generation in a marketplace system, as well as to conduct a comparative analysis of the effectiveness of the implemented algorithms using recommendation quality evaluation metrics.

The result of the work is an implemented server-side marketplace system with an integrated recommender subsystem that supports multiple recommendation generation approaches, as well as a mechanism for automated experimental evaluation of recommender algorithm effectiveness.

ЗМІСТ

ВСТУП

Сучасний розвиток електронної комерції супроводжується стрімким збільшенням кількості товарів, користувачів та обсягів інформації, що обробляються цифровими платформами. Маркетплейси стали одними з найбільш поширених типів інформаційних систем, які забезпечують взаємодію між продавцями та покупцями, надаючи можливість пошуку, перегляду та придбання великої кількості товарів у межах єдиного середовища.

Зі збільшенням кількості товарів у системах електронної комерції виникає проблема ефективного пошуку продукції, яка найбільше відповідає інтересам та потребам користувачів. Великий обсяг інформації ускладнює процес вибору товарів, збільшує час пошуку необхідної продукції та може негативно впливати на зручність використання маркетплейсу.

Одним із найбільш ефективних способів підвищення рівня персоналізації сучасних інформаційних систем є використання рекомендаційних механізмів. Рекомендаційні системи дозволяють автоматично аналізувати інформацію про товари, поведінку користувачів та інші дані з метою формування персоналізованих рекомендацій. Використання таких механізмів сприяє покращенню взаємодії користувачів із системою та підвищенню ефективності електронної комерції.

Рекомендаційні системи активно використовуються у сучасних маркетплейсах, соціальних мережах, стримінгових сервісах та інших цифрових платформах. Якість роботи рекомендаційних алгоритмів безпосередньо впливає на рівень персоналізації системи, релевантність запропонованого контенту та загальну ефективність роботи інформаційної системи.

Важливим напрямом розвитку рекомендаційних систем є дослідження підходів до формування рекомендацій та оцінювання ефективності їх роботи. Різні методи побудови рекомендацій мають власні особливості, переваги та

обмеження, що потребує проведення порівняльного аналізу та визначення найбільш ефективних підходів для використання у системах електронної комерції.

У межах роботи виконується дослідження особливостей побудови рекомендаційних систем для маркетплейсів, проєктування серверної частини інформаційної системи та реалізація механізмів формування рекомендацій. Також у роботі проводиться експериментальне оцінювання ефективності реалізованих рішень за допомогою метрик оцінювання якості рекомендацій.

Під час виконання роботи здійснюється проєктування функціональної структури рекомендаційної підсистеми, побудова архітектури серверної частини маркетплейсу, створення структури бази даних та інтеграція рекомендаційного функціоналу у загальну систему електронної комерції.

Практичним результатом роботи є реалізована серверна частина маркетплейсу з інтегрованим механізмом формування рекомендацій та можливістю проведення автоматизованого експериментального оцінювання ефективності рекомендаційних алгоритмів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика предметної області маркетплейсів та особливості взаємодії користувачів з товарами

Маркетплейс є різновидом електронної торговельної платформи, яка забезпечує взаємодію між продавцями та покупцями в межах єдиного інформаційного середовища. Основною особливістю такого типу систем є те, що платформа виступає посередником між сторонами угоди, надаючи інструменти для розміщення товарів, їх пошуку, порівняння та придбання. На відміну від традиційних інтернет-магазинів, де всі товари належать одному продавцю, маркетплейси дозволяють розміщувати продукцію великій кількості незалежних продавців, що значно розширює асортимент і підвищує конкурентність пропозицій[1].

Сучасні маркетплейси виконують роль комплексних цифрових екосистем, які об’єднують різноманітні функціональні модулі для організації електронної комерції. До основних можливостей таких платформ належать управління каталогом товарів, пошук і фільтрація продукції, формування замовлень, обробка платежів, система рейтингів та відгуків, а також аналітичні інструменти для продавців. Завдяки централізованій інфраструктурі користувачі отримують можливість взаємодіяти з великою кількістю товарів і продавців у межах одного інтерфейсу, що значно спрощує процес придбання товарів.

Ключовою характеристикою маркетплейсів є наявність великого та постійно зростаючого каталогу товарів, який може включати тисячі або навіть мільйони позицій. Така кількість інформації створює певні труднощі для користувачів під час пошуку необхідної продукції, оскільки перегляд усіх доступних варіантів стає практично неможливим. У зв’язку з цим особливо

важливим елементом функціонування маркетплейсів стають інструменти навігації та персоналізації, які допомагають користувачам швидко знаходити релевантні товари серед великої кількості пропозицій.

Взаємодія користувачів із товарами на маркетплейсах відбувається через різні типи дій, що формують поведінкові дані. До таких дій належать перегляд сторінок товарів, використання пошукових запитів, додавання товарів до кошика або списку бажань, а також здійснення покупок. Кожна з цих взаємодій містить інформацію про інтереси користувача та може використовуватися для аналізу його вподобань. Зібрані дані дозволяють системі формувати уявлення про потреби користувачів і покращувати процес підбору релевантних товарів.

Ще однією особливістю маркетплейсів є динамічний характер інформації про товари. Асортимент продукції постійно змінюється внаслідок появи нових пропозицій, зміни цін, оновлення характеристик товарів або появи нових продавців. У таких умовах ефективне управління інформацією та підтримка актуальності даних стають важливими завданнями інформаційної системи. Це також ускладнює процес пошуку товарів для користувачів, оскільки велика кількість альтернативних пропозицій може створювати інформаційне перевантаження[2].

З огляду на зазначені особливості, маркетплейси потребують використання спеціалізованих механізмів аналізу даних, які дозволяють покращити навігацію користувачів у великому каталозі товарів. Одним із найбільш ефективних інструментів для вирішення цієї задачі є рекомендаційні системи, що дозволяють автоматично формувати персоналізовані пропозиції на основі характеристик товарів та історії взаємодії користувачів із платформою. Використання таких систем сприяє підвищенню зручності користування маркетплейсом та покращує ефективність процесу пошуку товарів.

Рекомендаційна система є спеціалізованим програмним механізмом, призначеним для автоматичного формування персоналізованих пропозицій

для користувачів інформаційної системи. Основним завданням таких систем є аналіз даних про об’єкти та поведінку користувачів з метою визначення найбільш релевантних елементів, які можуть викликати інтерес у конкретного користувача. У контексті електронної комерції рекомендаційні системи застосовуються для підбору товарів, які відповідають індивідуальним уподобанням користувачів, що дозволяє значно спростити процес пошуку необхідної продукції.

Необхідність використання рекомендаційних систем зумовлена стрімким зростанням обсягів інформації в сучасних цифрових платформах. У великих маркетплейсах кількість доступних товарів може досягати десятків або сотень тисяч позицій, що значно ускладнює навігацію для користувачів. У таких умовах виникає проблема інформаційного перевантаження, коли користувачеві складно самостійно знайти найбільш релевантні товари серед великої кількості альтернатив. Рекомендаційні системи дозволяють автоматизувати процес відбору інформації та надавати користувачеві персоналізований перелік пропозицій.

Функціонування рекомендаційної системи базується на аналізі різних типів даних, які можуть характеризувати як самі об’єкти, так і поведінку користувачів. До таких даних належать характеристики товарів, історія переглядів, інформація про здійснені покупки, оцінки користувачів, а також інші дії, що відображають їхні інтереси. Обробка цих даних дозволяє системі виявляти закономірності та формувати припущення щодо того, які товари можуть бути найбільш цікавими для конкретного користувача.

Важливою особливістю рекомендаційних систем є їх здатність до персоналізації інформації. На відміну від стандартних методів пошуку або фільтрації, які надають однакові результати для всіх користувачів, рекомендаційні системи формують індивідуальні пропозиції для кожного користувача окремо. Це дозволяє значно підвищити ефективність взаємодії користувача з інформаційною системою, оскільки пропоновані елементи більшою мірою відповідають його інтересам та потребам[3].

У сучасних інформаційних системах рекомендаційні механізми застосовуються в різних сферах, зокрема в електронній комерції, потокових сервісах, соціальних мережах та інформаційних порталах. У маркетплейсах рекомендаційні системи можуть використовуватися для пропонування схожих товарів, персоналізованих добірок продукції, а також для відображення популярних або релевантних товарів на основі поведінки користувачів. Використання таких механізмів дозволяє покращити користувацький досвід та сприяє підвищенню ефективності роботи платформи.

Взаємодія користувачів із товарами у маркетплейсах має ряд специфічних особливостей, які відрізняють такі системи від інших типів електронної комерції. Однією з ключових характеристик є значна різноманітність дій користувача під час роботи з платформою. Користувачі можуть не лише здійснювати покупку товару, а й виконувати низку інших дій, зокрема переглядати сторінки товарів, здійснювати пошук за ключовими словами, застосовувати фільтри, порівнювати характеристики різних товарів, додавати товари до кошика або списку бажань. Кожна з таких дій формує окремий тип поведінкових даних, які можуть використовуватися для аналізу інтересів користувачів.

Іншою важливою особливістю є велика кількість альтернативних пропозицій одного й того самого товару або товарів зі схожими характеристиками. На маркетплейсах один і той самий товар може пропонуватися різними продавцями з відмінностями у вартості, умовах доставки або рейтингу продавця. Це створює для користувача широкий вибір, але одночасно ускладнює процес прийняття рішення. У таких умовах користувач змушений аналізувати значний обсяг інформації, що формує специфічні патерни поведінки, наприклад часті переходи між сторінками схожих товарів.

Ще однією характерною рисою є наявність непрямих взаємодій з товарами, які не завжди завершуються покупкою. У багатьох випадках користувач може переглядати товар лише з інформаційною метою,

порівнювати його з іншими варіантами або зберігати для майбутньої покупки. Такі дії також мають важливе значення для аналізу поведінки користувачів, оскільки вони дозволяють виявляти інтерес до певних категорій товарів навіть за відсутності фактичної покупки.

Важливою особливістю маркетплейсів є також велика кількість нових товарів, які постійно додаються до каталогу. Через це користувачі регулярно взаємодіють із раніше невідомими їм товарами. На відміну від деяких інших комерційних систем, де асортимент може залишатися відносно стабільним, маркетплейси характеризуються динамічним оновленням каталогу. Це створює додаткові складності для аналізу поведінки користувачів, оскільки система повинна враховувати взаємодії з новими об’єктами.

Крім того, поведінка користувачів у маркетплейсах часто має дослідницький характер. Перед здійсненням покупки користувач може переглядати велику кількість товарів, читати відгуки, оцінювати рейтинги продавців та аналізувати характеристики продукції. Такий тип поведінки відрізняється від багатьох інших комерційних систем, де процес покупки є більш прямолінійним і включає меншу кількість етапів взаємодії з товаром.

Таким чином, взаємодія користувачів із товарами у маркетплейсах характеризується великою кількістю поведінкових дій, високою варіативністю пропозицій та постійним оновленням каталогу продукції. Ці особливості формують значні обсяги даних про поведінку користувачів, що створює передумови для використання методів аналізу даних і рекомендаційних систем для підвищення ефективності навігації та персоналізації контенту.

Значна кількість доступних товарів і велика кількість альтернативних пропозицій створюють складність у процесі пошуку релевантної продукції. У таких умовах виникає необхідність використання механізмів автоматизованого аналізу даних, які дозволяють підвищити ефективність навігації користувачів у системі та забезпечити персоналізований підбір товарів. Рекомендаційні системи виступають одним із ключових інструментів вирішення цієї задачі, оскільки дозволяють формувати індивідуальні

пропозиції на основі характеристик товарів і поведінкових даних користувачів.

1.2 Роль рекомендаційних систем у маркетплейсах та їх вплив на персоналізацію сервісу

Стрімке зростання обсягів інформації в електронній комерції призвело до появи великої кількості товарів, доступних для користувачів у межах однієї платформи. У сучасних маркетплейсах асортимент може налічувати сотні тисяч або навіть мільйони позицій, що значно ускладнює процес пошуку необхідного товару. У таких умовах традиційні інструменти навігації, зокрема пошук і фільтрація, не завжди дозволяють швидко знайти найбільш релевантні пропозиції. Рекомендаційні системи застосовуються для автоматизації цього процесу та забезпечення більш ефективної взаємодії користувачів із великими каталогами товарів.

Основною функцією рекомендаційних систем є аналіз даних про користувачів та об’єкти системи з метою формування персоналізованих пропозицій. Такі системи дозволяють визначати товари, які з найбільшою ймовірністю можуть зацікавити конкретного користувача. Формування рекомендацій базується на аналізі історії взаємодії користувача з платформою, характеристик товарів, а також поведінки інших користувачів. Використання цих даних дозволяє системі виявляти закономірності та прогнозувати потенційні інтереси користувачів.

Важливою роллю рекомендаційних систем у маркетплейсах є підвищення рівня персоналізації сервісу. На відміну від стандартного відображення каталогу товарів, який є однаковим для всіх користувачів, рекомендаційні механізми дозволяють формувати індивідуальні добірки продукції. Це означає, що кожен користувач може отримувати різний набір

товарів, сформований на основі його попередньої поведінки, інтересів та вподобань. Персоналізація сприяє підвищенню зручності користування системою та скорочує час, необхідний для пошуку потрібного товару.

Застосування рекомендаційних систем також позитивно впливає на ефективність функціонування маркетплейсів. Персоналізовані рекомендації дозволяють підвищити ймовірність того, що користувач знайде товар, який відповідає його потребам. Це може призводити до збільшення кількості переглядів товарів, підвищення рівня взаємодії користувачів із платформою та зростання кількості здійснених покупок. Таким чином рекомендаційні системи виконують не лише інформаційну, але й економічну функцію, сприяючи підвищенню ефективності роботи електронних торговельних платформ.

Крім того, рекомендаційні системи допомагають користувачам відкривати нові товари, які вони могли б не знайти під час звичайного пошуку. У великих каталогах багато товарів можуть залишатися непоміченими через їхнє розташування в результатах пошуку або недостатню популярність. Рекомендаційні алгоритми дозволяють привертати увагу користувачів до таких товарів, якщо вони відповідають їхнім інтересам. Це сприяє більш рівномірному розподілу уваги користувачів між різними товарами та продавцями на платформі.

Персоналізація сервісу є одним із ключових напрямів розвитку сучасних цифрових платформ, зокрема маркетплейсів. У системах електронної комерції персоналізація передбачає адаптацію контенту, інтерфейсу та пропозицій відповідно до індивідуальних характеристик і поведінки конкретного користувача. Рекомендаційні системи відіграють у цьому процесі важливу роль, оскільки дозволяють формувати індивідуалізовані добірки товарів на основі аналізу даних про взаємодію користувачів із платформою.

Застосування рекомендаційних алгоритмів дозволяє відмовитися від однакового представлення товарів для всіх користувачів. Замість цього система може формувати унікальний набір пропозицій для кожного

користувача залежно від його попередніх дій у системі. До таких дій можуть належати перегляди товарів, здійснені покупки, використані пошукові запити, а також взаємодія з певними категоріями або брендами. Аналіз цих даних дозволяє визначати інтереси користувача та пропонувати товари, які найбільшою мірою відповідають його вподобанням.

Важливою особливістю персоналізації є можливість адаптації різних елементів інтерфейсу маркетплейсу. Рекомендаційні системи можуть використовуватися для формування персоналізованих блоків на головній сторінці, відображення рекомендованих товарів на сторінках каталогу, а також для пропонування схожих або пов’язаних товарів під час перегляду конкретної позиції. Такий підхід дозволяє користувачеві швидше знаходити релевантні пропозиції та зменшує кількість дій, необхідних для пошуку потрібного товару.

Персоналізація також сприяє підвищенню рівня залученості користувачів у роботу платформи. Коли користувач отримує пропозиції, що відповідають його інтересам, зростає ймовірність подальшої взаємодії з системою, зокрема перегляду товарів або здійснення покупок. У результаті користувач отримує більш комфортний досвід роботи з платформою, а маркетплейс підвищує ефективність використання власного каталогу товарів.

Крім того, використання рекомендаційних систем дозволяє враховувати не лише довгострокові інтереси користувача, але й поточний контекст його взаємодії з системою. Наприклад, рекомендації можуть формуватися на основі останніх переглянутих товарів, поточних пошукових запитів або дій, виконаних у межах однієї сесії. Такий підхід дозволяє забезпечити більш гнучку персоналізацію сервісу та підвищити релевантність запропонованих товарів.

Рекомендаційні системи є важливим компонентом сучасних маркетплейсів, оскільки дозволяють ефективно працювати з великими каталогами товарів і значними обсягами даних про взаємодію користувачів із платформою. Використання таких систем забезпечує автоматичний підбір

релевантних товарів на основі характеристик продукції та поведінкових даних користувачів.

Застосування рекомендаційних алгоритмів сприяє підвищенню рівня персоналізації сервісу маркетплейсу. Формування індивідуальних добірок товарів дозволяє адаптувати контент платформи до інтересів конкретного користувача, що покращує зручність навігації та скорочує час пошуку необхідної продукції. Персоналізовані рекомендації також підвищують рівень взаємодії користувачів із системою та сприяють більш ефективному використанню каталогу товарів.

1.3 Поведінкові та контекстні дані користувачів як основа для формування рекомендацій

Поведінкові дані користувача являють собою інформацію про дії, які користувач здійснює під час взаємодії з інформаційною системою. Такі дані відображають спосіб використання платформи та дозволяють аналізувати інтереси, уподобання і наміри користувачів. У системах електронної комерції поведінкові дані формуються в процесі роботи користувача з каталогом товарів, пошуковими інструментами, сторінками товарів та іншими функціональними елементами платформи.

До поведінкових даних належать різні типи взаємодій користувача з товарами та інтерфейсом системи. Зокрема, це можуть бути перегляди сторінок товарів, пошукові запити, використання фільтрів, додавання товарів до кошика або списку бажань, а також фактичні покупки. Кожна з таких дій містить інформацію про інтерес користувача до певних товарів або категорій продукції. Накопичення подібних даних дозволяє формувати історію взаємодії користувача з платформою.

Особливістю поведінкових даних є те, що вони формуються

автоматично під час використання системи та не потребують прямого введення інформації користувачем. На відміну від анкетних або демографічних даних, які користувач може надавати під час реєстрації, поведінкові дані відображають реальні дії та рішення користувача. Завдяки цьому вони можуть більш точно характеризувати інтереси користувача та його реальні наміри щодо вибору товарів.

Аналіз поведінкових даних дозволяє виявляти закономірності у взаємодії користувачів із товарами. Наприклад, система може визначати товари, які користувач переглядає найчастіше, категорії продукції, що викликають найбільший інтерес, або послідовність дій, які передують здійсненню покупки. Отримана інформація може використовуватися для прогнозування майбутніх інтересів користувача та формування персоналізованих рекомендацій.

Використання поведінкових даних є одним із ключових елементів функціонування рекомендаційних систем у маркетплейсах. Аналіз історії взаємодії користувача з товарами дозволяє системі визначати схожість між товарами, виявляти подібність поведінки різних користувачів та формувати рекомендації, які з найбільшою ймовірністю відповідатимуть інтересам конкретного користувача[4].

Контекстні дані користувача являють собою інформацію про умови та обставини, у яких відбувається взаємодія користувача з інформаційною системою. На відміну від поведінкових даних, що відображають конкретні дії користувача, контекстні дані описують зовнішні фактори, які можуть впливати на його інтереси, потреби та рішення під час використання платформи. У системах електронної комерції такі дані можуть використовуватися для більш точного формування рекомендацій та адаптації контенту відповідно до поточної ситуації користувача.

До контекстних даних можуть належати різні характеристики середовища взаємодії користувача із системою. Зокрема, це можуть бути час і дата взаємодії, місцезнаходження користувача, тип пристрою, що

використовується для доступу до платформи, операційна система або браузер. Такі параметри дозволяють враховувати особливості використання системи в різних умовах і формувати рекомендації, які відповідають поточному контексту взаємодії.

Важливу роль відіграє також контекст поточної сесії користувача. До нього можуть належати останні переглянуті товари, активні пошукові запити, вибрані фільтри або категорії каталогу. Подібна інформація дозволяє системі визначати поточний інтерес користувача та пропонувати товари, які відповідають його поточним намірам. Використання таких даних дозволяє формувати рекомендації, що є актуальними саме в момент взаємодії користувача з платформою.

Контекстні дані також можуть відображати більш загальні умови використання системи, наприклад сезонність або популярність певних категорій товарів у конкретний період часу. Наприклад, у певні періоди року користувачі можуть проявляти більший інтерес до певних типів товарів, що може враховуватися під час формування рекомендацій. Урахування таких факторів дозволяє системі адаптувати пропозиції відповідно до змін у поведінці користувачів.

Використання контекстних даних у рекомендаційних системах дозволяє підвищити релевантність сформованих рекомендацій. Поєднання інформації про поведінку користувача з даними про умови його взаємодії з платформою дає змогу отримати більш повне уявлення про його потреби та інтереси. Це сприяє формуванню більш точних персоналізованих рекомендацій та покращує загальний досвід користування маркетплейсом[5].

Поведінкові та контекстні дані користувачів є ключовими джерелами інформації для побудови рекомендаційних систем у маркетплейсах. Саме ці типи даних дозволяють системі отримати уявлення про інтереси користувача, його наміри та умови, у яких відбувається взаємодія з платформою. Використання таких даних створює основу для формування персоналізованих рекомендацій, які враховують як історію дій користувача, так і поточний

контекст використання системи.

Поведінкові дані відображають реальні дії користувача в системі, що робить їх одним із найбільш інформативних джерел для аналізу його вподобань. Перегляди товарів, пошукові запити, додавання товарів до кошика або здійснення покупок дозволяють визначити, які категорії товарів викликають найбільший інтерес. Аналіз таких дій дає змогу виявляти закономірності у поведінці користувачів та прогнозувати їхні подальші дії, що є важливим для формування релевантних рекомендацій.

Контекстні дані доповнюють поведінкову інформацію та дозволяють враховувати умови, у яких користувач взаємодіє з системою. Інтереси користувача можуть змінюватися залежно від часу, місця або поточної мети використання платформи. Наприклад, один і той самий користувач може шукати різні типи товарів у різні моменти часу або в різних ситуаціях. Урахування контекстних факторів дозволяє адаптувати рекомендації до поточних потреб користувача.

Поєднання поведінкових і контекстних даних дозволяє отримати більш повне уявлення про взаємодію користувача з маркетплейсом. Поведінкові дані характеризують довгострокові інтереси користувача, тоді як контекстні дані дозволяють враховувати поточні умови використання системи. Використання обох типів інформації підвищує точність прогнозування інтересів користувача та сприяє формуванню більш релевантних рекомендацій[6].

Застосування поведінкових і контекстних даних також дозволяє ефективніше працювати з великими каталогами товарів, характерними для маркетплейсів. Аналіз цих даних дає змогу системі визначати найбільш релевантні товари серед великої кількості доступних варіантів і пропонувати їх користувачеві. Це підвищує ефективність навігації у системі та сприяє покращенню користувацького досвіду під час використання маркетплейсу.

1.4 Аналіз підходів до побудови рекомендаційних систем

Існує кілька підходів до побудови рекомендаційних систем, які відрізняються принципами аналізу даних та методами формування рекомендацій. Кожен із підходів використовує різні джерела інформації та алгоритмічні методи для визначення найбільш релевантних об’єктів для користувача.

Контентно-орієнтований підхід (Content-Based Filtering). Контентно-орієнтовані рекомендаційні системи формують рекомендації на основі характеристик об’єктів та попередніх дій користувача. У таких системах аналізуються властивості товарів або іншого контенту, з яким взаємодіяв користувач, після чого підбираються інші об’єкти зі схожими характеристиками. Наприклад, якщо користувач переглядав або купував товари певної категорії, система може рекомендувати інші товари з подібними параметрами. Основною перевагою цього підходу є можливість формування рекомендацій навіть за відсутності інформації про інших користувачів[7].

Колаборативна фільтрація (Collaborative Filtering). Колаборативна фільтрація базується на аналізі взаємодій великої кількості користувачів із товарами або іншими об’єктами системи. Основна ідея цього підходу полягає в тому, що користувачі з подібними інтересами можуть мати схожі вподобання щодо товарів. Якщо певні користувачі демонструють подібну поведінку, товари, які сподобалися одному з них, можуть бути рекомендовані іншому. Такий підхід дозволяє враховувати колективний досвід користувачів та часто використовується у великих інформаційних системах[7].

Гібридний підхід (Hybrid Recommendation Systems). Гібридні рекомендаційні системи поєднують кілька різних методів формування рекомендацій, зокрема контентно-орієнтований підхід і колаборативну фільтрацію. Метою такого поєднання є зменшення недоліків окремих підходів та підвищення якості рекомендацій. Наприклад, система може одночасно

враховувати характеристики товарів і поведінку інших користувачів для формування більш точних рекомендацій. Гібридні системи широко застосовуються в сучасних цифрових платформах завдяки своїй гнучкості та високій ефективності[7].

Підхід на основі знань (Knowledge-Based Recommendation). Рекомендаційні системи на основі знань використовують попередньо визначені правила або моделі, що описують взаємозв'язки між характеристиками товарів та потребами користувачів. У таких системах рекомендації формуються на основі логічних правил або експертних знань, що дозволяють визначити, які товари можуть бути найбільш відповідними для конкретного користувача. Подібні підходи часто застосовуються у системах, де покупки здійснюються рідко або потребують детального аналізу характеристик товарів[7].

Демографічний підхід (Demographic-Based Recommendation). Демографічні рекомендаційні системи формують рекомендації на основі загальних характеристик користувачів, таких як вік, стать, місцезнаходження або інші соціально-демографічні параметри. Користувачі, які мають схожі характеристики, можуть отримувати подібні рекомендації. Такий підхід не потребує великої кількості даних про поведінку користувача, однак точність рекомендацій може бути нижчою порівняно з іншими методами[7].

Контекстно-орієнтований підхід (Context-Aware Recommendation). Контекстно-орієнтовані рекомендаційні системи враховують додаткову інформацію про умови взаємодії користувача із системою. До таких факторів можуть належати час, місцезнаходження користувача, тип пристрою або поточні дії в межах сесії. Урахування контексту дозволяє формувати рекомендації, які відповідають поточній ситуації користувача та його актуальним потребам. Такий підхід дозволяє підвищити релевантність рекомендацій у динамічних середовищах[7].

Контентно-орієнтований підхід до побудови рекомендаційних систем базується на аналізі характеристик об'єктів, з якими користувач взаємодіяв

раніше, та формуванні рекомендацій на основі їхньої подібності до інших об’єктів у системі. Основна ідея цього підходу полягає у припущенні, що користувач у майбутньому буде зацікавлений у тих об’єктах, які мають схожі характеристики з об’єктами, що вже викликали його інтерес. У контексті маркетплейсів це означає, що система аналізує властивості товарів, переглянутих або придбаних користувачем, після чого пропонує інші товари з подібними параметрами.

Функціонування такого підходу передбачає представлення товарів у вигляді набору ознак або параметрів. До таких ознак можуть належати категорія товару, бренд, технічні характеристики, ключові слова опису, ціновий діапазон або інші атрибути, що характеризують товар. Для кожного користувача формується певний профіль інтересів, який відображає характеристики товарів, з якими він взаємодіяв. Після цього система порівнює характеристики інших товарів із профілем користувача та визначає рівень їхньої схожості. Товари з найвищим рівнем подібності можуть бути запропоновані як рекомендації.

Контентно-орієнтовані системи часто використовують методи обчислення схожості між об’єктами, зокрема косинусну міру подібності або інші статистичні методи. Для цього товари можуть бути представлені у вигляді векторів ознак, де кожна характеристика відповідає певному параметру. Порівняння таких векторів дозволяє визначити, наскільки два товари є подібними між собою. Чим більший рівень схожості між характеристиками товару та інтересами користувача, тим більшою є ймовірність того, що цей товар буде рекомендований.

Серед переваг контентно-орієнтованого підходу можна виділити можливість формування рекомендацій без необхідності аналізу поведінки інших користувачів. Це дозволяє системі працювати навіть у випадках, коли кількість користувачів або взаємодій із системою є відносно невеликою. Крім того, такий підхід дозволяє ефективно працювати з новими товарами, оскільки рекомендації можуть формуватися на основі їхніх характеристик навіть за

відсутності історії взаємодій.

До недоліків контентно-орієнтованого підходу належить обмеженість рекомендацій у межах характеристик, які вже присутні у профілі користувача. Система зазвичай пропонує товари, схожі на ті, з якими користувач взаємодіяв раніше, що може зменшувати різноманітність рекомендацій. Крім того, ефективність такого підходу значною мірою залежить від якості та повноти опису товарів, оскільки недостатня кількість характеристик може ускладнювати визначення подібності між об’єктами.

Колаборативна фільтрація є одним із найбільш поширених підходів до побудови рекомендаційних систем, який базується на аналізі колективної поведінки користувачів. Основна ідея цього підходу полягає в тому, що користувачі з подібними інтересами або схожою історією взаємодії з товарами можуть демонструвати схожі вподобання. У такому випадку товари, які зацікавили одного користувача, можуть бути рекомендовані іншому користувачу зі схожою поведінкою. На відміну від контентно-орієнтованого підходу, колаборативна фільтрація не потребує детального опису характеристик товарів, оскільки рекомендації формуються на основі даних про взаємодії користувачів із системою.

Функціонування колаборативної фільтрації зазвичай базується на побудові матриці взаємодій користувачів і товарів. У такій матриці кожен рядок відповідає певному користувачеві, а кожен стовпець – окремому товару. Значення в матриці можуть відображати різні типи взаємодій, наприклад оцінку товару, факт перегляду, додавання до кошика або здійснення покупки. На основі цієї матриці система може визначати ступінь подібності між користувачами або між товарами та використовувати ці дані для формування рекомендацій.

Існує два основних варіанти реалізації колаборативної фільтрації: user-based та item-based підходи. У user-based підході система визначає користувачів зі схожими вподобаннями та рекомендує товари, які були цікавими для подібних користувачів. У item-based підході аналізується

схожість між товарами на основі взаємодій користувачів із ними. Якщо два товари часто переглядаються або купуються разом, система може рекомендувати один із них користувачеві, який уже взаємодіяв з іншим.

Серед переваг колаборативної фільтрації можна виділити здатність виявляти приховані закономірності у поведінці користувачів та формувати рекомендації, які не обмежуються лише характеристиками товарів. Завдяки аналізу колективного досвіду користувачів система може пропонувати товари, які користувач міг би не знайти під час звичайного пошуку. Такий підхід часто забезпечує високу якість рекомендацій у системах із великою кількістю користувачів і взаємодій.

До недоліків колаборативної фільтрації належить проблема так званого «холодного старту». У випадках, коли в системі з’являється новий користувач або новий товар, відсутність історії взаємодій ускладнює формування рекомендацій. Крім того, ефективність такого підходу значною мірою залежить від обсягу доступних даних, оскільки недостатня кількість взаємодій між користувачами і товарами може призводити до зниження точності рекомендацій.

Гібридний підхід до побудови рекомендаційних систем передбачає поєднання кількох різних методів формування рекомендацій з метою підвищення їхньої точності та ефективності. Найчастіше в таких системах комбінуються контентно-орієнтовані методи та колаборативна фільтрація, однак можуть використовуватися й інші підходи, зокрема контекстно-орієнтовані або знаннєві моделі. Основна ідея гібридних систем полягає у використанні сильних сторін різних алгоритмів і компенсації їхніх недоліків шляхом інтеграції результатів кількох методів.

Функціонування гібридної рекомендаційної системи може реалізовуватися різними способами. Одним із поширених варіантів є зважене поєднання результатів кількох алгоритмів, коли кожен метод формує власний список рекомендацій або оцінку релевантності товарів, після чого результати об’єднуються у спільний рейтинг. Іншим підходом є послідовне використання

методів, коли один алгоритм формує попередній список кандидатів, а інший уточнює результати. Також можливе використання комбінованих моделей, які одночасно враховують характеристики товарів і поведінку користувачів.

Гібридні системи дозволяють враховувати різні джерела інформації про користувачів і товари. Наприклад, система може аналізувати характеристики товарів, історію взаємодій користувачів із системою, а також контекстні дані, що описують умови використання платформи. Поєднання таких даних дозволяє формувати більш повну модель інтересів користувача та підвищувати релевантність рекомендацій. У маркетплейсах гібридні системи часто використовуються для формування персоналізованих списків товарів, які враховують як індивідуальні інтереси користувача, так і популярність товарів серед інших користувачів.

Серед основних переваг гібридного підходу можна виділити підвищення якості рекомендацій та зменшення впливу недоліків окремих алгоритмів. Поєднання кількох методів дозволяє частково вирішувати проблему холодного старту, покращувати різноманітність рекомендацій та підвищувати стабільність роботи системи. Крім того, гібридні системи є більш гнучкими, оскільки можуть адаптуватися до різних типів даних та умов використання платформи.

До недоліків гібридного підходу належить підвищена складність реалізації та налаштування системи. Інтеграція кількох алгоритмів потребує додаткових обчислювальних ресурсів, а також ретельного підбору параметрів для їхньої ефективної взаємодії. Крім того, складність архітектури може ускладнювати процес підтримки та масштабування системи, особливо у великих інформаційних платформах.

Демографічний підхід до побудови рекомендаційних систем базується на використанні соціально-демографічних характеристик користувачів для формування рекомендацій. У межах цього підходу передбачається, що користувачі з подібними демографічними параметрами можуть мати схожі інтереси та вподобання щодо товарів або послуг. До таких характеристик

можуть належати вік, стать, місцезнаходження, рівень доходу, професія або інші параметри, які дозволяють віднести користувача до певної групи. На основі належності до певної демографічної категорії система може пропонувати товари, що є популярними або найбільш релевантними для відповідної групи користувачів.

Функціонування демографічних рекомендаційних систем передбачає попередню сегментацію користувачів за певними параметрами. Після формування таких сегментів система аналізує популярність або рівень взаємодії з товарами всередині кожної групи. На основі отриманих результатів можуть формуватися рекомендації для користувачів, які належать до відповідної демографічної категорії. Наприклад, користувачам певного вікового діапазону можуть рекомендуватися товари, які користуються попитом серед інших користувачів із подібними характеристиками.

Однією з особливостей демографічного підходу є те, що він не потребує великої історії взаємодій користувача із системою. Рекомендації можуть формуватися навіть для нових користувачів на основі їхніх базових характеристик, отриманих під час реєстрації або визначених іншими способами. Завдяки цьому демографічні методи можуть застосовуватися на початкових етапах використання платформи, коли поведінкові дані ще відсутні або їх недостатньо для побудови більш складних моделей рекомендацій.

До основних переваг демографічного підходу належить відносна простота реалізації та можливість формування рекомендацій навіть за обмеженої кількості поведінкових даних. Такий підхід дозволяє швидко сегментувати користувачів і пропонувати їм релевантні товари на основі загальних закономірностей споживчої поведінки в межах певних груп. Крім того, він може бути корисним для вирішення проблеми холодного старту для нових користувачів.

До недоліків демографічного підходу належить обмежена точність рекомендацій, оскільки користувачі з однаковими демографічними

характеристиками можуть мати різні інтереси та вподобання. Використання лише демографічних параметрів не дозволяє враховувати індивідуальну поведінку користувача та його реальні дії в системі. У результаті сформовані рекомендації можуть бути менш персоналізованими порівняно з підходами, що базуються на аналізі поведінкових або контекстних даних.

Контекстно-орієнтований підхід до побудови рекомендаційних систем передбачає формування рекомендацій з урахуванням умов, у яких користувач взаємодіє з інформаційною системою. На відміну від традиційних рекомендаційних алгоритмів, що враховують лише характеристики товарів або історію взаємодії користувача з платформою, контекстно-орієнтовані системи аналізують додаткові фактори, які можуть впливати на інтереси користувача в конкретний момент часу. До таких факторів можуть належати час взаємодії, місцезнаходження користувача, тип пристрою, поточні пошукові запити, попередні дії в межах однієї сесії або інші умови використання системи.

Основною особливістю цього підходу є можливість врахування змін інтересів користувача залежно від ситуації. Інтереси користувачів можуть змінюватися в різні моменти часу або в різних умовах використання системи. Наприклад, користувач може шукати різні товари залежно від часу доби, поточного завдання або контексту використання платформи. Урахування таких факторів дозволяє формувати рекомендації, які відповідають актуальним потребам користувача, а не лише його загальним довгостроковим інтересам.

Контекстно-орієнтований підхід відрізняється від контентно-орієнтованого тим, що основна увага приділяється не характеристикам самих товарів, а умовам взаємодії користувача із системою. У контентно-орієнтованих системах рекомендації формуються на основі схожості характеристик товарів, з якими користувач взаємодіяв раніше. У контекстно-орієнтованих системах додатково враховується інформація про поточну ситуацію використання системи. Таким чином, контекстно-орієнтований

підхід дозволяє формувати рекомендації, що враховують не лише інтереси користувача, але й обставини їхнього прояву.

До основних переваг контекстно-орієнтованого підходу належить можливість підвищення релевантності рекомендацій шляхом урахування додаткових факторів взаємодії користувача із системою. Використання контекстних даних дозволяє формувати рекомендації, які краще відповідають поточним потребам користувача. Такий підхід також дозволяє враховувати короткострокові інтереси користувача, що особливо важливо у динамічних інформаційних системах, таких як маркетплейси.

До недоліків контекстно-орієнтованого підходу належить складність збору та обробки контекстних даних. Не всі типи контекстної інформації можуть бути доступними або точними, що може ускладнювати використання цього підходу на практиці. Крім того, використання великої кількості контекстних параметрів може збільшувати складність моделей рекомендаційної системи та потребувати додаткових обчислювальних ресурсів для обробки даних.

Таким чином, контентно-орієнтовані системи базуються на аналізі характеристик товарів і попередніх взаємодій користувача, тоді як колаборативна фільтрація використовує інформацію про поведінку інших користувачів. Гібридні підходи поєднують різні методи для підвищення точності рекомендацій. Демографічні системи формують рекомендації на основі характеристик користувачів, а контекстно-орієнтовані підходи враховують умови взаємодії користувача із системою.

Різноманітність підходів до побудови рекомендаційних систем дозволяє обирати найбільш доцільні методи залежно від типу інформаційної системи, обсягу доступних даних та особливостей взаємодії користувачів із платформою. Поєднання різних джерел даних та методів аналізу дозволяє підвищити ефективність рекомендацій і забезпечити більш точний підбір релевантних товарів для користувачів маркетплейсу.

1.5 Огляд існуючих реалізованих рекомендаційних систем у сучасних маркетплейсах

Для аналізу практичного застосування рекомендаційних систем у маркетплейсах доцільно розглянути кілька відомих платформ електронної комерції, що активно використовують алгоритми персоналізації. Як приклади можна виділити український маркетплейс Rozetka, американський маркетплейс Amazon та європейський маркетплейс Allegro. Ці платформи мають великий каталог товарів і використовують різні механізми рекомендацій для покращення користувацького досвіду.

Rozetka є одним із найбільших українських маркетплейсів електронної комерції, який пропонує широкий асортимент товарів різних категорій. На платформі використовуються рекомендаційні механізми для підбору товарів, що можуть бути цікавими для користувача. Зокрема, на сторінках товарів відображаються блоки зі схожими товарами, популярними товарами або товарами, які часто переглядають разом. Такі рекомендації формуються на основі характеристик товарів, їхньої популярності та поведінки користувачів у системі. Це дозволяє користувачам швидше знаходити альтернативні варіанти товарів і спрощує процес вибору[8].

Amazon є одним із найвідоміших прикладів використання рекомендаційних систем у сфері електронної комерції. Платформа використовує складні алгоритми персоналізації, що аналізують історію переглядів, покупки користувачів, оцінки товарів та інші поведінкові дані. Одним із характерних елементів системи рекомендацій Amazon є блоки «Customers who bought this item also bought» та «Recommended for you», які пропонують користувачам товари на основі поведінки інших покупців. Завдяки використанню колаборативної фільтрації та гібридних алгоритмів Amazon може формувати персоналізовані рекомендації для мільйонів користувачів[9].

Allegro є одним із найбільших маркетплейсів у Європі, що активно використовується у Польщі та інших країнах регіону. Платформа застосовує рекомендаційні механізми для персоналізації пропозицій і покращення навігації у великому каталозі товарів. Користувачам можуть відображатися рекомендації товарів на основі їхніх попередніх переглядів, пошукових запитів або покупок. Також система може пропонувати схожі товари або популярні пропозиції в межах певної категорії. Використання рекомендаційних алгоритмів дозволяє платформі підвищувати релевантність пропозицій і покращувати досвід користування сервісом[10].

Серед розглянутих маркетплейсів основним об’єктом подальшого аналізу доцільно обрати маркетплейс Rozetka. Такий вибір зумовлений тим, що ця платформа є одним із найбільших та найбільш популярних маркетплейсів електронної комерції в Україні. Rozetka має широкий асортимент товарів різних категорій, значну кількість користувачів і продавців, а також активно використовує механізми персоналізації та рекомендацій для покращення користувацького досвіду.

Ще одним важливим фактором вибору є подібність функціональних можливостей платформи до тих, що планується реалізувати в межах розроблюваної системи. Маркетплейс Rozetka надає користувачам можливість переглядати каталог товарів, виконувати пошук і фільтрацію, порівнювати характеристики товарів, додавати товари до кошика та здійснювати покупки. У процесі взаємодії користувачів із платформою формуються значні обсяги поведінкових даних, які можуть використовуватися для формування рекомендацій.

На платформі також реалізовано різні механізми рекомендацій товарів, які відображаються в інтерфейсі користувача. Наприклад, під час перегляду сторінки товару користувач може бачити блоки зі схожими товарами, альтернативними пропозиціями або товарами, які часто переглядають разом. Подібні рекомендації допомагають користувачам швидше знаходити релевантні товари та спрощують процес прийняття рішення про покупку.

Використання Rozetka як прикладу для аналізу дозволяє розглянути практичну реалізацію рекомендаційних механізмів у сучасному маркетплейсі. Аналіз функціонування таких механізмів може бути корисним для визначення вимог до розроблюваної системи рекомендацій, а також для формування підходів до реалізації алгоритмів підбору товарів у межах дослідження.

Одним із прикладів реалізації рекомендаційних механізмів на маркетплейсі Rozetka є блок рекомендацій «Рекомендації на основі ваших переглядів», який відображається у вигляді горизонтального слайдера з товарами. Цей елемент інтерфейсу формує персоналізовану добірку товарів на основі історії переглядів користувача, тобто тих товарів, сторінки яких він відкривав раніше під час роботи з платформою. Скріншот такого блоку наведено на рисунку 1.1.

Рисунок 1.1 – Блок рекомендацій на основі переглядів користувача

У межах цього блоку система аналізує товари, які користувач переглядав у поточній або попередніх сесіях, після чого підбирає інші товари зі схожими характеристиками або з тієї ж категорії. Рекомендовані товари відображаються у вигляді карток, що містять основну інформацію про товар, зокрема зображення, назву, рейтинг, кількість відгуків, актуальну ціну, можливі знижки та умови доставки. Такий формат подання дозволяє користувачеві

швидко ознайомитися з альтернативними пропозиціями без необхідності виконувати додатковий пошук.

Слайдер має горизонтальну структуру та дозволяє переглядати більшу кількість рекомендованих товарів шляхом прокручування. Кожна картка товару також містить додаткові елементи взаємодії, зокрема кнопку додавання товару до кошика та можливість додати товар до списку обраного. Це спрощує процес переходу від перегляду рекомендацій до здійснення покупки.

Використання такого механізму рекомендацій дозволяє маркетплейсу враховувати поточні інтереси користувача та пропонувати товари, які можуть бути для нього найбільш релевантними. Завдяки цьому користувач отримує можливість швидше знаходити альтернативні або схожі товари, що покращує навігацію у великому каталозі продукції та підвищує зручність користування платформою.

Під час переходу на сторінку конкретного товару маркетплейс Rozetka оновлює блок «Рекомендації на основі ваших переглядів», додаючи до нього щойно переглянутий товар. На представленому прикладі після відкриття сторінки принтера Epson EcoTank L1250 цей товар з’являється у списку рекомендованих елементів. Це свідчить про те, що система рекомендацій відстежує історію переглядів користувача та використовує її для формування персоналізованого набору товарів. Зміни блоку наведено на рисунку 1.2.

Рисунок 1.2 – Зміни в блоці рекомендацій на основі переглядів

Механізм роботи такого блоку можна описати як формування рекомендацій на основі історії переглянутих товарів. Кожен новий товар, сторінку якого відкриває користувач, додається до внутрішнього списку взаємодій користувача із системою. На основі цього списку система може формувати рекомендації, що включають як самі переглянуті товари, так і інші товари, пов’язані з ними за певними характеристиками.

У результаті у блоці рекомендацій одночасно можуть відображатися товари з різних категорій, якщо користувач переглядав їх під час попередніх взаємодій із платформою. Наприклад, у представленому випадку поряд із принтером відображаються портативні зарядні станції, блок живлення та SSD-накопичувач, які користувач переглядав раніше. Це демонструє, що система формує список рекомендацій на основі послідовності переглядів користувача, а не лише на основі категорії поточного товару.

Такий механізм дозволяє користувачеві швидко повернутися до товарів, які він переглядав раніше, або продовжити їх порівняння. Крім того, подібний блок може використовуватися як джерело даних для більш складних алгоритмів рекомендацій, оскільки історія переглядів є важливим поведінковим сигналом для визначення інтересів користувача.

Під час подальшого використання платформи можна помітити, що товари додаються до блоку «Рекомендації на основі ваших переглядів» лише у випадку, коли користувач переходить безпосередньо на сторінку відповідного товару. Звичайний перегляд товарів у каталозі, у списках пошуку або в інших рекомендаційних блоках не призводить до появи товару в цьому списку. Таким чином, формування цього блоку залежить саме від факту відкриття сторінки товару.

У результаті такий механізм більше нагадує відображення історії переглянутих товарів, ніж повноцінну систему рекомендацій у класичному розумінні. Список товарів фактично відображає послідовність об’єктів, з якими користувач взаємодіяв під час перегляду їхніх сторінок. Нові товари

додаються до списку після їх перегляду, що дозволяє користувачеві швидко повернутися до раніше переглянутих позицій.

Подібний підхід може виконувати допоміжну функцію у навігації по сайту, оскільки користувачі часто переглядають декілька товарів перед прийняттям рішення про покупку. Наявність такого блоку дозволяє легко знайти раніше переглянуті товари без необхідності повторного пошуку в каталозі. Це спрощує процес порівняння різних варіантів товарів.

Водночас така реалізація демонструє, що цей конкретний блок не формує рекомендації на основі аналізу схожості товарів або поведінки інших користувачів. Його функціонування базується на відображенні історії переглядів користувача, що є одним із типів поведінкових даних. Подібні дані можуть використовуватися як один із компонентів більш складних рекомендаційних алгоритмів у межах маркетплейсу.

Окрім блоку, що фактично відображає історію переглянутих товарів, маркетплейс Rozetka також використовує інший механізм рекомендацій, представлений у вигляді секції «Найкращі пропозиції для вас». На відміну від попереднього блоку, цей елемент інтерфейсу вже демонструє більш складну логіку формування рекомендацій і надає користувачеві товари, які мають схожість із раніше переглянутими. Такий блок наведений на рисунку 1.3.

У межах цієї секції система аналізує товари, сторінки яких були відвідані користувачем, після чого формує набір рекомендацій, що містить схожі або тематично пов’язані товари. Наприклад, якщо користувач переглядав сторінки принтерів, система може запропонувати інші моделі принтерів, сумісні пристрої або супутні товари з тієї ж категорії. Такий підхід дозволяє користувачеві швидше знаходити альтернативні варіанти товарів або інші продукти, що можуть відповідати його інтересам.

Особливістю роботи цього механізму є те, що система не просто відображає раніше переглянуті товари, а формує нові пропозиції на основі їхніх характеристик. Кожен переглянутий товар може впливати на формування рекомендацій, додаючи до списку один або кілька схожих товарів.

У результаті в блоці рекомендацій можуть з’являтися 1–2 пропозиції для кожного товару, сторінку якого відвідував користувач.

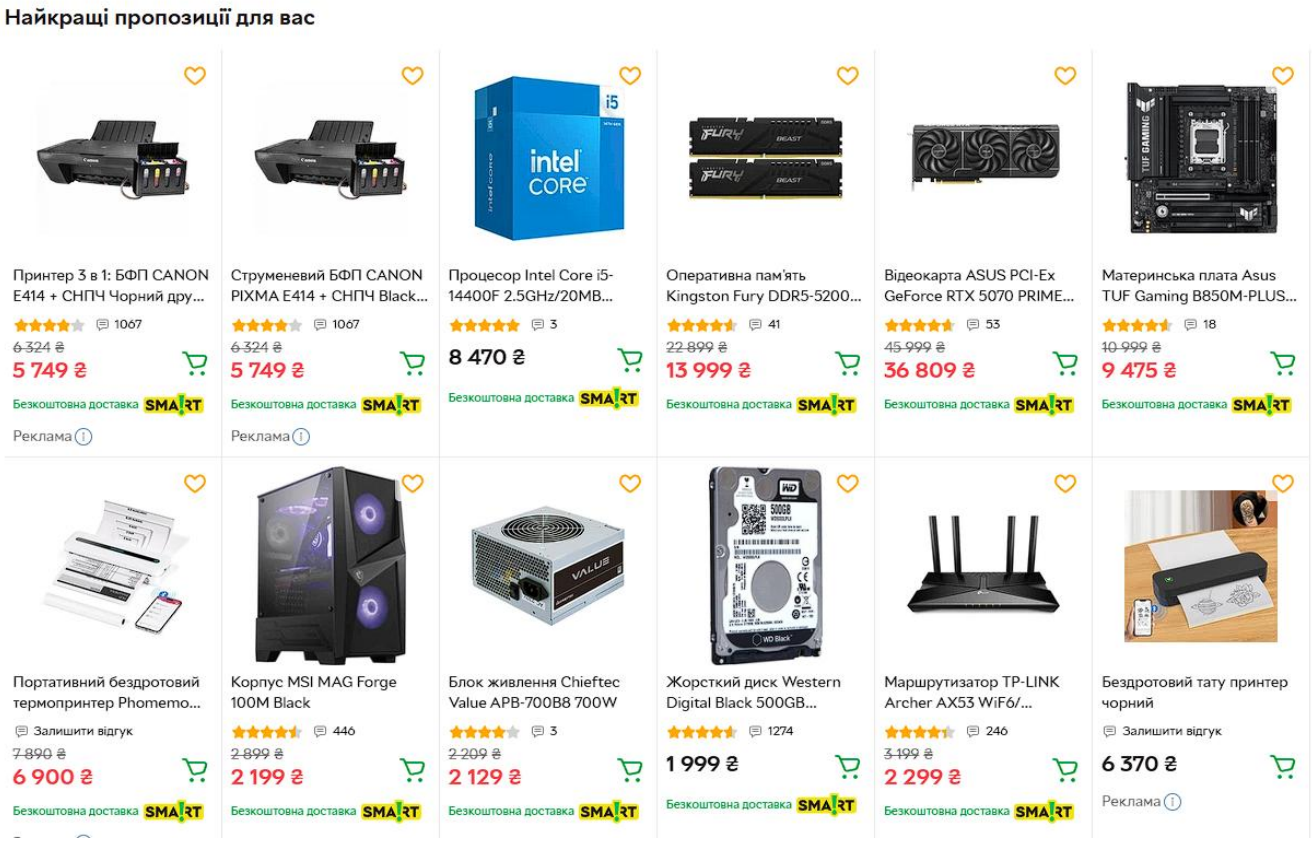

Рисунок 1.3 – Найкращі пропозиції платформи

Такий механізм є ближчим до класичної реалізації рекомендаційної системи, оскільки він використовує аналіз характеристик товарів або інших параметрів для підбору релевантних пропозицій. У порівнянні з блоком історії переглядів цей підхід забезпечує більш корисні рекомендації для користувача, оскільки пропонує не лише вже переглянуті товари, але й нові варіанти, які можуть відповідати його інтересам.

Аналіз реалізації рекомендаційних механізмів на маркетплейсі Rozetka дозволяє виділити низку переваг та недоліків такої системи.

До переваг рекомендаційної системи Rozetka можна віднести використання поведінкових даних користувачів, зокрема історії переглядів товарів. Це дозволяє системі враховувати інтереси користувача під час

формування персоналізованих пропозицій. Також позитивною особливістю є наявність декількох блоків рекомендацій, які виконують різні функції: один із них відображає переглянуті товари, що спрощує повернення до раніше переглянутих позицій, тоді як інший пропонує схожі товари на основі попередніх взаємодій користувача. Такий підхід дозволяє покращити навігацію у великому каталозі товарів та спрощує процес пошуку альтернативних варіантів продукції.

Ще однією перевагою є інтеграція рекомендацій безпосередньо в інтерфейс сторінок маркетплейсу. Рекомендовані товари відображаються у вигляді окремих блоків на сторінках товарів або в інших частинах сайту, що робить їх помітними для користувача та підвищує ймовірність взаємодії з ними. Крім того, використання рекомендацій на основі схожості товарів дозволяє пропонувати користувачеві нові товари, які можуть відповідати його інтересам.

Водночас система рекомендацій Rozetka має і певні недоліки. Один із основних недоліків полягає в тому, що деякі рекомендаційні блоки фактично виконують функцію історії переглядів, а не повноцінної системи рекомендацій. Такі блоки відображають лише товари, сторінки яких були відкриті користувачем, і не пропонують нові варіанти продукції, що знижує рівень персоналізації.

Іншим недоліком є обмежена різноманітність рекомендацій. У багатьох випадках система пропонує лише кілька схожих товарів для кожного переглянутого об’єкта, що може обмежувати можливість користувача знайти більш широкий спектр альтернативних пропозицій. Крім того, рекомендації можуть формуватися переважно на основі характеристик товарів або історії переглядів, що не завжди дозволяє враховувати складніші закономірності поведінки користувачів.

Таким чином, система рекомендацій маркетплейсу Rozetka демонструє базову реалізацію механізмів персоналізації, що використовує поведінкові дані користувачів і аналіз схожості товарів. Водночас існують можливості для

подальшого вдосконалення таких механізмів шляхом використання більш складних алгоритмів рекомендацій, здатних враховувати ширший спектр даних про взаємодію користувачів із платформою.

Аналіз існуючих рекомендаційних механізмів у сучасних маркетплейсах показує, що такі системи активно використовуються для персоналізації сервісу та покращення навігації у великих каталогах товарів. Приклади маркетплейсів Rozetka, Amazon та Allegro демонструють використання різних типів рекомендацій, що базуються на поведінкових даних користувачів, популярності товарів або схожості продукції.

Проведений аналіз також показує, що рекомендаційні системи можуть реалізовуватися з різним рівнем складності – від простого відображення історії переглядів до формування персоналізованих пропозицій на основі характеристик товарів та поведінки користувачів. Використання таких механізмів дозволяє покращити користувацький досвід та спростити процес пошуку релевантних товарів у маркетплейсі.

1.6 Порівняльний аналіз підходів та обґрунтування вибору трьох підходів для реалізації

У сучасних рекомендаційних системах використовується кілька основних підходів до формування рекомендацій, кожен із яких має власні принципи роботи, переваги та обмеження. Серед найбільш поширених підходів можна виділити контентно-орієнтовані системи, колаборативну фільтрацію, гібридні рекомендаційні системи, демографічний підхід та контекстно-орієнтовані системи. Усі ці методи використовують різні типи даних про користувачів і об’єкти системи для визначення релевантних рекомендацій[11].

Контентно-орієнтований підхід базується на аналізі характеристик

товарів та історії взаємодії користувача з платформою. Система формує рекомендації, пропонуючи товари, що мають подібні характеристики до тих, з якими користувач уже взаємодіяв. Колаборативна фільтрація використовує інформацію про поведінку інших користувачів і дозволяє визначати товари, які можуть бути цікавими для користувача на основі схожості їхніх уподобань. Гібридні системи поєднують кілька методів формування рекомендацій, що дозволяє підвищити точність результатів і компенсувати недоліки окремих підходів.

Демографічний підхід передбачає формування рекомендацій на основі соціально-демографічних характеристик користувачів. Користувачі з подібними параметрами можуть отримувати схожі рекомендації, що дозволяє формувати пропозиції навіть за відсутності великої кількості поведінкових даних. Контекстно-орієнтовані системи, у свою чергу, враховують додаткові фактори взаємодії користувача із системою, зокрема час, місцезнаходження, тип пристрою або поточні дії користувача в межах сесії.

Кожен із зазначених підходів має певні переваги та обмеження. Контентно-орієнтовані системи добре працюють із характеристиками товарів, але можуть формувати менш різноманітні рекомендації. Колаборативна фільтрація дозволяє використовувати колективний досвід користувачів, однак потребує значної кількості даних про взаємодії. Демографічні та контекстно-орієнтовані підходи можуть покращувати точність рекомендацій у певних умовах, але зазвичай використовуються як допоміжні джерела інформації. Гібридні системи дозволяють поєднувати різні методи, однак їх реалізація є більш складною[12].

З урахуванням особливостей маркетплейсів та доступних типів даних доцільним є використання трьох основних підходів для подальшої реалізації та експериментального дослідження: контентно-орієнтованого підходу, колаборативної фільтрації та гібридного підходу. Таке поєднання дозволяє дослідити різні принципи формування рекомендацій: використання характеристик товарів, аналіз колективної поведінки користувачів та

комбінування кількох алгоритмічних методів. Порівняння результатів роботи цих підходів дозволить оцінити їх ефективність у контексті маркетплейсу та визначити найбільш доцільний варіант застосування рекомендаційної системи.

Порівняння всіх наведених підходів було наведено в таблиці 1.1.

У таблиці 1.1 наведено порівняльну характеристику основних підходів до побудови рекомендаційних систем, що застосовуються у сучасних інформаційних платформах. Кожен із представлених підходів має власний принцип формування рекомендацій, а також використовує різні типи даних про користувачів і об’єкти системи.

Контентно-орієнтований підхід базується на аналізі характеристик товарів і попередніх взаємодій користувача із системою. Рекомендації формуються на основі подібності між товарами, що дозволяє пропонувати користувачеві об’єкти зі схожими властивостями. Такий підхід не потребує аналізу поведінки інших користувачів, однак може обмежувати різноманітність рекомендацій.

Колаборативна фільтрація використовує інформацію про взаємодії великої кількості користувачів із товарами. Завдяки аналізу колективної поведінки система може визначати подібність між користувачами або товарами та формувати рекомендації на основі цих закономірностей. Такий підхід здатний забезпечувати високу точність рекомендацій, однак його ефективність залежить від обсягу доступних даних.

Гібридні рекомендаційні системи поєднують кілька методів формування рекомендацій, що дозволяє використовувати переваги різних підходів. Комбінування контентно-орієнтованих і колаборативних методів дозволяє підвищити точність рекомендацій і частково усунути недоліки окремих алгоритмів. Водночас реалізація таких систем є більш складною з технічної точки зору.

Таблиця 1.1 – Порівняння підходів до побудови рекомендаційних систем

Підхід	Основний принцип роботи	Джерела даних	Переваги	Недоліки
Контентно-орієнтований	Формування рекомендацій на основі схожості характеристик товарів, з якими користувач взаємодіяв раніше	Характеристики товарів, історія переглядів і покупок	Може працювати без даних інших користувачів; добре працює з новими товарами	Обмежена різноманітність рекомендацій; залежність від якості опису товарів
Колаборативна фільтрація	Аналіз поведінки великої кількості користувачів для визначення подібних інтересів	Дані взаємодії користувачів із товарами (перегляди, покупки, оцінки)	Виявляє приховані закономірності інтересів користувачів; висока точність при великому обсязі даних	Проблема холодного старту; потребує великої кількості взаємодій
Гібридний підхід	Комбінування кількох методів рекомендацій для підвищення точності	Характеристики товарів, поведінкові дані користувачів, інші джерела	Підвищена точність рекомендацій; зменшення недоліків окремих методів	Складність реалізації та налаштування

Продовження таблиці 1.1

Підхід	Основний принцип роботи	Джерела даних	Переваги	Недоліки
Демографічний підхід	Формування рекомендацій на основі соціально-демографічних характеристик користувачів	Вік, стать, місцезнаходження та інші демографічні параметри	Простота реалізації; може використовуватися для нових користувачів	Низький рівень персоналізації; не враховує реальну поведінку користувача
Контекстно-орієнтований	Урахування умов взаємодії користувача із системою під час формування рекомендацій	Час, місцезнаходження, тип пристрою, дії в межах сесії	Врахування поточних потреб користувача; підвищення релевантності рекомендацій	Складність збору та обробки контекстних даних

Демографічний та контекстно-орієнтований підходи використовують додаткову інформацію про користувачів або умови їхньої взаємодії із системою. Демографічні системи формують рекомендації на основі характеристик користувачів, тоді як контекстно-орієнтовані системи враховують ситуаційні фактори, такі як час або місцезнаходження. Хоча ці підходи можуть покращувати точність рекомендацій у певних умовах, вони зазвичай використовуються як допоміжні механізми у складі більш складних рекомендаційних систем.

Для подальшого дослідження та практичної реалізації в межах роботи доцільно обрати три основні підходи до побудови рекомендаційних систем: контентно-орієнтований підхід, колаборативну фільтрацію та гібридний підхід. Вибір саме цих методів зумовлений їх широким використанням у сучасних інформаційних системах, а також можливістю застосування у маркетплейсах, де доступні як характеристики товарів, так і дані про взаємодію користувачів із платформою.

Контентно-орієнтований підхід було обрано через можливість формування рекомендацій на основі характеристик товарів та індивідуальної історії взаємодії користувача із системою. Такий підхід добре підходить для маркетплейсів, оскільки дозволяє використовувати структуровану інформацію про товари, зокрема їхні категорії, атрибути та інші параметри. Крім того, контентно-орієнтовані системи можуть ефективно працювати навіть за відносно невеликої кількості користувачів, що робить цей підхід зручним для реалізації та тестування.

Колаборативна фільтрація була обрана як один із найбільш поширених і ефективних методів побудови рекомендаційних систем. Її використання дозволяє враховувати закономірності у поведінці користувачів та формувати рекомендації на основі колективного досвіду взаємодії з товарами. Для маркетплейсів цей підхід є особливо актуальним, оскільки велика кількість користувачів і взаємодій із товарами створює значні обсяги поведінкових даних, які можуть бути використані для аналізу та формування рекомендацій.

Гібридний підхід було обрано з метою дослідження можливості поєднання різних методів формування рекомендацій. Комбінування контентно-орієнтованих алгоритмів і колаборативної фільтрації дозволяє використовувати переваги обох підходів та частково компенсувати їхні недоліки. Такий підхід дозволяє враховувати як характеристики товарів, так і поведінку користувачів, що може сприяти підвищенню точності та різноманітності рекомендацій.

Обрані підходи відрізняються принципами формування рекомендацій та використовують різні типи даних, що створює можливість для їх порівняльного дослідження. Реалізація цих методів у межах маркетплейсу дозволить оцінити їхню ефективність у реальних умовах використання системи, а також визначити, який із підходів забезпечує найбільш релевантні рекомендації для користувачів платформи.

Різні підходи до побудови рекомендаційних систем використовують різні типи даних і принципи формування рекомендацій. Контентно-орієнтовані методи базуються на характеристиках товарів та індивідуальній історії взаємодії користувача із системою. Колаборативна фільтрація використовує дані про поведінку інших користувачів, що дозволяє виявляти закономірності у виборі товарів. Гібридні системи поєднують кілька методів формування рекомендацій, що дозволяє підвищити їхню точність і ефективність.

З урахуванням особливостей маркетплейсів та доступних типів даних для подальшої реалізації та дослідження було обрано контентно-орієнтований підхід, колаборативну фільтрацію та гібридний підхід. Використання цих методів дозволяє дослідити різні принципи формування рекомендацій та провести їх порівняльний аналіз у межах розроблюваної інформаційної системи маркетплейсу.

1.7 Постановка задачі кваліфікаційної роботи

Метою кваліфікаційної роботи є дослідження підходів до побудови рекомендаційних систем для маркетплейсів та визначення ефективності різних алгоритмів формування рекомендацій на основі поведінкових і контекстних даних користувачів. У рамках роботи необхідно дослідити особливості використання рекомендаційних механізмів у сучасних платформах електронної комерції, а також визначити основні принципи їх побудови.

Для досягнення поставленої мети необхідно розробити інформаційну систему маркетплейсу, що дозволяє користувачам переглядати каталог товарів, виконувати пошук продукції та взаємодіяти з товарами. У процесі роботи системи повинні збиратися дані про взаємодії користувачів із товарами, зокрема перегляди сторінок товарів та інші дії, які можуть використовуватися як джерело поведінкової інформації.

У межах розроблюваної системи необхідно реалізувати кілька різних підходів до формування рекомендацій товарів. Зокрема передбачається реалізація контентно-орієнтованого підходу, колаборативної фільтрації та гібридного підходу, що поєднує елементи двох попередніх методів. Кожен із цих підходів повинен бути реалізований у вигляді окремого алгоритму рекомендацій, здатного формувати персоналізовані пропозиції для користувачів системи.

Для забезпечення роботи рекомендаційних алгоритмів необхідно спроєктувати структуру бази даних маркетплейсу, яка дозволить зберігати інформацію про товари, користувачів та їхні взаємодії з платформою. На основі цих даних повинна формуватися вибірка для тестування реалізованих алгоритмів рекомендацій[13].

У процесі виконання роботи необхідно провести експериментальне дослідження ефективності реалізованих підходів до формування

рекомендацій. Для цього передбачається використання відповідних метрик оцінювання якості рекомендацій, що дозволять порівняти результати роботи різних алгоритмів.

Результатом виконання кваліфікаційної роботи має стати реалізована інформаційна система маркетплейсу з інтегрованим модулем рекомендацій товарів, а також проведений порівняльний аналіз ефективності різних підходів до побудови рекомендаційних систем. Отримані результати повинні дозволити визначити найбільш доцільний підхід до формування рекомендацій у системах електронної комерції.

2 ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Формування функціональних вимог до системи формування рекомендацій маркетплейсу

Для формалізації функціональних вимог до інформаційної системи доцільно використовувати методології структурного моделювання, що дозволяють наочно представити логіку роботи системи та взаємозв’язки між її складовими. Одним із поширених інструментів такого моделювання є методологія IDEF0, яка застосовується для опису функціональної структури систем, процесів та потоків інформації між ними[14].

Методологія IDEF0 призначена для побудови функціональних моделей системи у вигляді ієрархії процесів, де кожен процес відображає певну функцію або набір дій, що виконуються в межах системи. Основною особливістю цієї методології є використання стандартизованого способу подання процесів у вигляді блоків та стрілок, що відображають інформаційні потоки. Такий підхід дозволяє формалізувати складні процеси функціонування інформаційної системи та представити їх у вигляді зрозумілої структурної моделі[15].

У рамках IDEF0 кожна функція системи зображається у вигляді прямокутного блоку, до якого підводяться чотири типи стрілок. Вхідні дані відображають інформацію або ресурси, які використовуються для виконання процесу. Вихідні дані представляють результати роботи функції. Керуючі впливи визначають правила, обмеження або умови, відповідно до яких виконується процес. Механізми відображають ресурси або інструменти, за допомогою яких реалізується відповідна функція системи.

Однією з ключових переваг IDEF0 є можливість поетапної деталізації процесів за допомогою механізму декомпозиції. На верхньому рівні моделі відображається загальний процес функціонування системи, після чого він

послідовно деталізується на підпроцеси, які описують окремі аспекти роботи системи. Такий підхід дозволяє послідовно переходити від загального опису функціонування системи до більш детального аналізу окремих компонентів[16].

Використання діаграми IDEF0 у межах розробки системи рекомендацій маркетплейсу дозволяє формалізувати процес формування рекомендацій, визначити основні джерела даних, що використовуються алгоритмами рекомендацій, а також відобразити взаємодію між компонентами системи. Зокрема, за допомогою цієї діаграми можна представити процес функціонування рекомендаційної системи, а також окремі підходи до формування рекомендацій, що використовуються у межах дослідження.

Побудова функціональної моделі системи на основі IDEF0 дозволяє чітко визначити структуру системи рекомендацій маркетплейсу, встановити зв’язки між її складовими та сформувати основу для подальшого проєктування програмної реалізації системи. Така модель також полегшує аналіз функціональних можливостей системи та забезпечує наочне представлення її логіки роботи.

Основним процесом функціонування системи рекомендацій у межах розроблюваного маркетплейсу було обрано процес формування рекомендацій для користувачів на основі доступних даних про товари та взаємодії користувачів із платформою. Саме цей процес визначає логіку роботи рекомендаційної системи та відображає взаємодію між даними, алгоритмами та компонентами програмної системи. Загальна функціональна модель такого процесу представлена на рисунку 2.1.

На діаграмі зображено контекстну модель процесу «Процес функціонування системи рекомендацій маркетплейсу», яка описує основні інформаційні потоки, що використовуються під час формування рекомендацій. Центральний блок моделі відображає основну функцію системи, яка полягає у формуванні персоналізованих рекомендацій товарів для користувачів маркетплейсу. До цього процесу надходять різні типи вхідних

даних, що використовуються алгоритмами рекомендацій для аналізу поведінки користувачів та характеристик товарів.

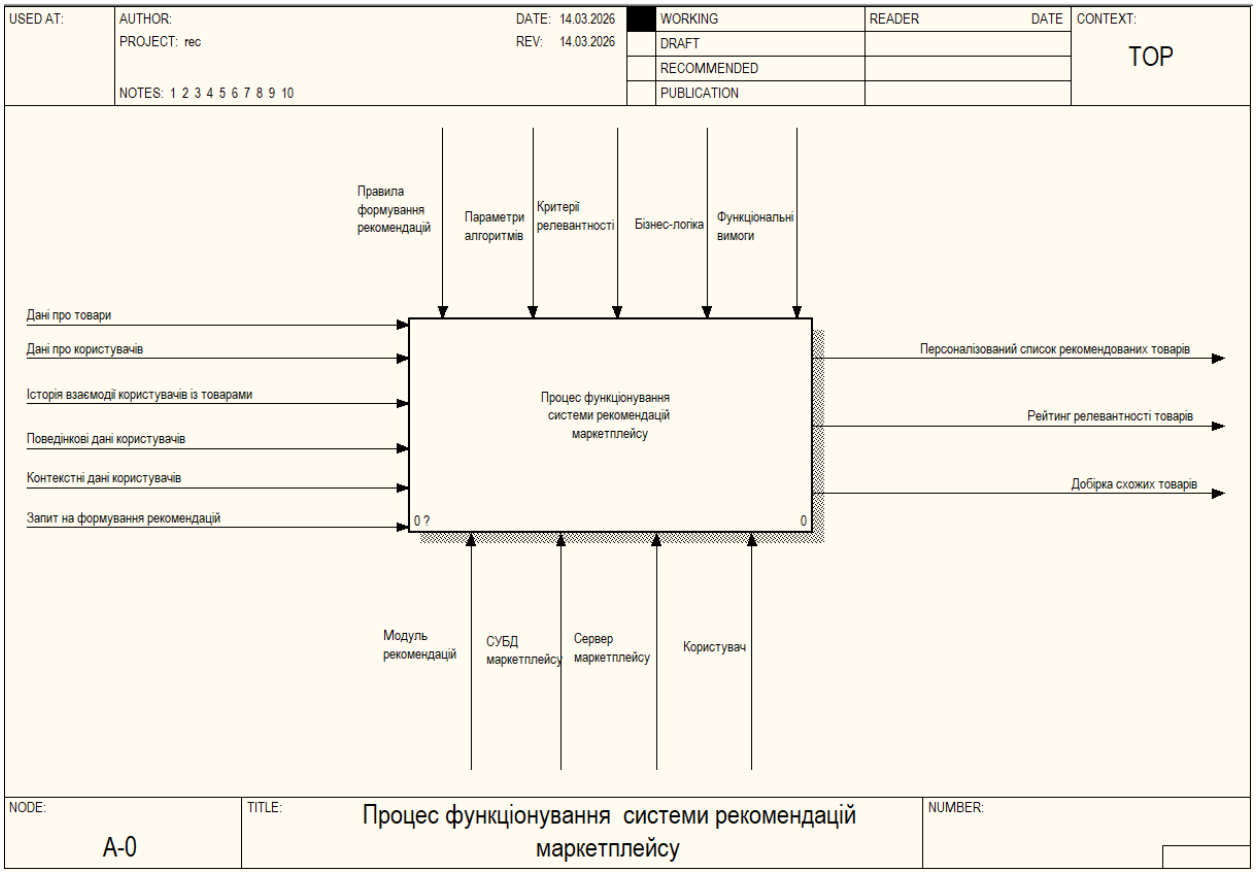

Рисунок 2.1 – Діаграма A-0 основного процесу системи

До вхідних даних процесу належать дані про товари, дані про користувачів, історія взаємодії користувачів із товарами, поведінкові дані користувачів та контекстні дані користувачів. Дані про товари містять інформацію про характеристики продукції, що представлена на маркетплейсі, зокрема категорії, параметри, описи та інші атрибути. Дані про користувачів містять інформацію про профілі користувачів платформи. Історія взаємодії відображає дії користувачів у системі, такі як перегляди товарів, додавання товарів до кошика або здійснення покупок. Поведінкові та контекстні дані дозволяють додатково враховувати умови, за яких користувач взаємодіє з системою.

Також до процесу надходить запит на формування рекомендацій, який

ініціює роботу рекомендаційної системи. Такий запит може формуватися під час перегляду користувачем сторінок маркетплейсу або при відкритті сторінки конкретного товару. Отримання цього запиту запускає процес обробки даних та формування персоналізованих рекомендацій.

Керування процесом здійснюється за допомогою набору правил та параметрів, що визначають принципи роботи рекомендаційної системи. До таких керуючих впливів належать правила формування рекомендацій, параметри алгоритмів, критерії релевантності, бізнес-логіка маркетплейсу та функціональні вимоги до системи. Ці елементи визначають, яким чином алгоритми рекомендацій повинні обробляти дані та за якими принципами формувати результати.

Виконання процесу забезпечується за допомогою відповідних механізмів, до яких належать модуль рекомендацій, система керування базою даних маркетплейсу, серверна частина маркетплейсу та користувач платформи. Модуль рекомендацій реалізує алгоритми формування рекомендацій, база даних забезпечує зберігання необхідної інформації, а сервер маркетплейсу виконує обробку запитів і взаємодію між компонентами системи.

Результатом виконання процесу є формування персоналізованого списку рекомендованих товарів, рейтингу релевантності товарів та добірки схожих товарів. Ці результати використовуються для відображення рекомендацій у різних частинах інтерфейсу маркетплейсу та сприяють покращенню навігації користувача в каталозі товарів.

Декомпозиція основного процесу функціонування системи рекомендацій представлена на рисунку 2.2. На цьому рівні деталізації загальний процес формування рекомендацій розділено на три основні підпроцеси, кожен з яких відповідає окремому підходу до побудови рекомендаційної системи. Такий підхід дозволяє відобразити різні алгоритмічні механізми формування рекомендацій та показати їх взаємозв’язок у межах єдиної інформаційної системи маркетплейсу.

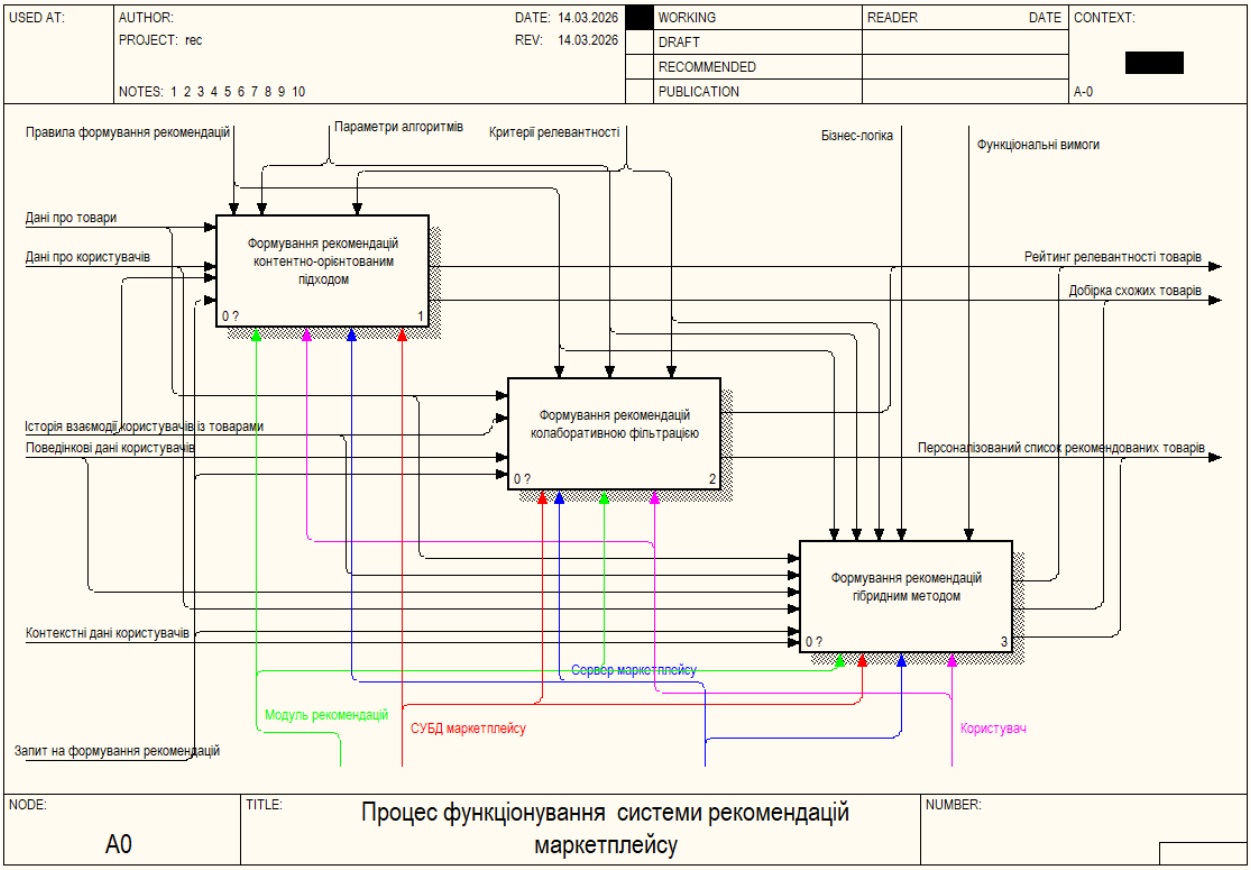

Рисунок 2.2 – Декомпозиція основного процесу системи

Першим підпроцесом є формування рекомендацій контентно-орієнтованим підходом. У межах цього процесу рекомендації формуються на основі характеристик товарів та історії взаємодії користувача з ними. Для роботи алгоритму використовуються дані про товари та дані про користувачів, що дозволяє сформувати профіль інтересів користувача. Отримані результати можуть використовуватися для визначення подібності між товарами та формування добірки схожих товарів, а також для розрахунку рейтингу релевантності товарів.

Другим підпроцесом є формування рекомендацій методом колаборативної фільтрації. У цьому випадку основою для формування рекомендацій виступають поведінкові дані користувачів та історія їхніх взаємодій із товарами. Алгоритм аналізує взаємозв’язки між користувачами або товарами, визначає закономірності у поведінці користувачів та на основі цього формує перелік товарів, які можуть бути цікавими для конкретного

користувача. Результатом такого аналізу є прогнозована оцінка релевантності товарів для користувача.

Третім підпроцесом є формування рекомендацій гібридним методом. Цей підхід поєднує результати контентно-орієнтованого методу та колаборативної фільтрації, що дозволяє врахувати як характеристики товарів, так і поведінкові особливості користувачів. У межах цього процесу здійснюється об’єднання отриманих рекомендацій та їх подальше ранжування відповідно до визначених критеріїв релевантності. Використання гібридного підходу дозволяє підвищити точність рекомендацій та зменшити обмеження, характерні для окремих методів.

До підпроцесів надходять ті самі вхідні дані, що використовуються у загальному процесі функціонування системи рекомендацій. До них належать дані про товари, дані про користувачів, історія взаємодії користувачів із товарами, поведінкові дані користувачів, контекстні дані користувачів та запит на формування рекомендацій. Керування процесами здійснюється за допомогою правил формування рекомендацій, параметрів алгоритмів, критеріїв релевантності, бізнес-логіки маркетплейсу та функціональних вимог до системи.

Реалізація зазначених процесів здійснюється за допомогою механізмів системи, до яких належать модуль рекомендацій, система керування базою даних маркетплейсу, серверна частина маркетплейсу та користувач платформи. Результатом роботи підпроцесів є формування персоналізованого списку рекомендованих товарів, рейтингу релевантності товарів та добірки схожих товарів, які передаються до інтерфейсу маркетплейсу для подальшого відображення користувачеві.

Декомпозиція процесу формування рекомендацій контентно-орієнтованим підходом представлена на рисунку 2.3. Ця діаграма деталізує один із підпроцесів системи рекомендацій та відображає послідовність етапів, необхідних для формування рекомендацій на основі характеристик товарів та історії взаємодії користувача з ними.

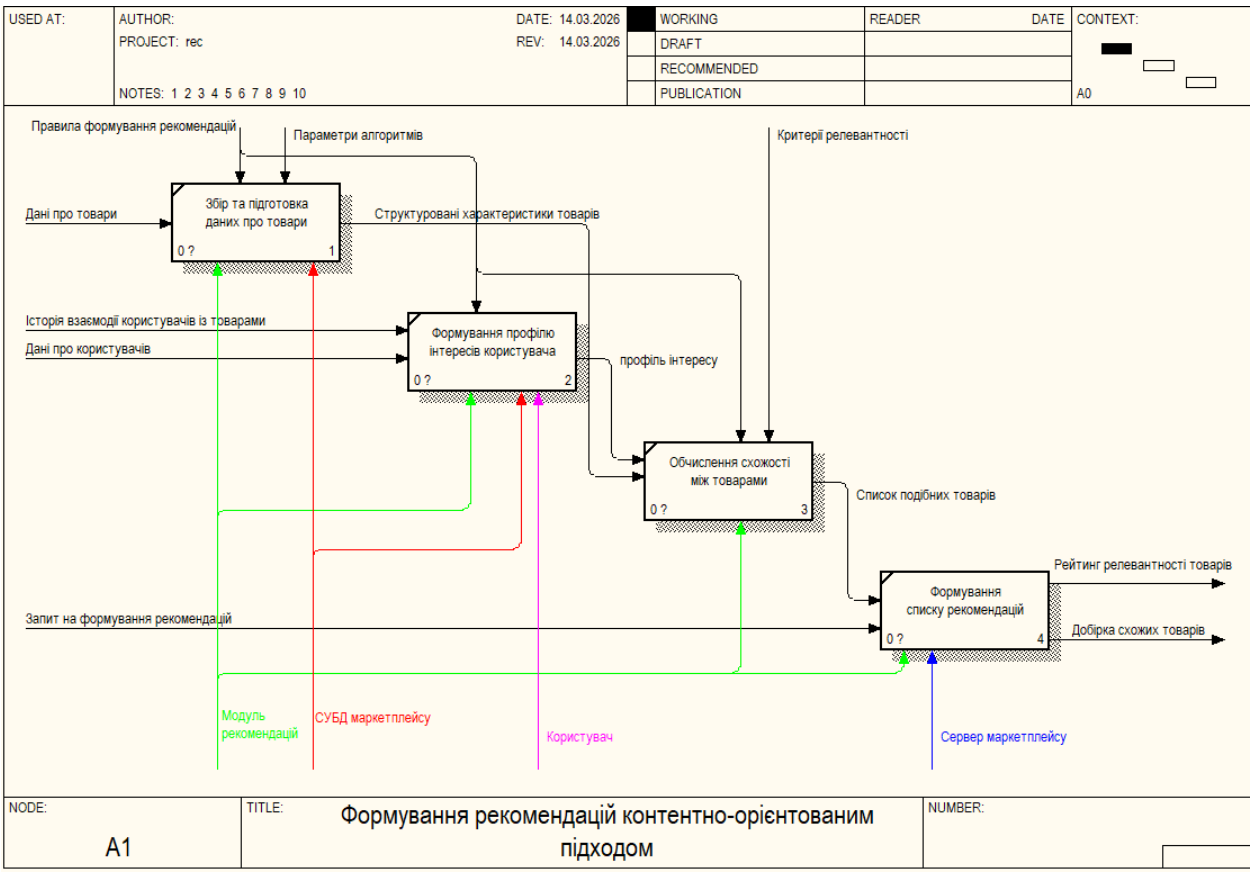

Рисунок 2.3 – Діаграма декомпозиції A1

Першим етапом процесу є збір та підготовка даних про товари. На цьому етапі система отримує інформацію про товари, що зберігається у базі даних маркетплейсу. До таких даних належать різноманітні характеристики товарів, зокрема категорії, технічні параметри, опис продукції та інші атрибути. Після отримання даних виконується їх підготовка та структуризація, що дозволяє сформувати набір характеристик товарів, придатний для подальшого аналізу алгоритмами рекомендацій.

Другим етапом є формування профілю інтересів користувача. Для цього використовуються дані про користувачів та історія їхніх взаємодій із товарами. Аналізуються товари, сторінки яких переглядав користувач, а також інші дії, що виконувалися в системі. На основі цієї інформації формується профіль інтересів користувача, який відображає характеристики товарів, що можуть бути для нього найбільш релевантними.

Наступним етапом є обчислення схожості між товарами. На цьому етапі

система використовує структуровані характеристики товарів разом із профілем інтересів користувача для визначення ступеня подібності між товарами. У результаті виконання цього процесу формується список товарів, які мають найбільшу схожість із товарами, що раніше зацікавили користувача.

Завершальним етапом процесу є формування списку рекомендацій. На основі списку подібних товарів виконується ранжування результатів відповідно до визначених критеріїв релевантності. Після цього формується підсумковий перелік рекомендованих товарів, який передається до інтерфейсу маркетплейсу для подальшого відображення користувачеві. Результатом роботи процесу є рейтинг релевантності товарів та добірка схожих товарів.

Функціонування процесу здійснюється за допомогою модуля рекомендацій, системи керування базою даних маркетплейсу, серверної частини системи та користувача платформи. Керування виконанням окремих етапів процесу забезпечується правилами формування рекомендацій, параметрами алгоритмів та критеріями релевантності, що визначають принципи роботи рекомендаційної системи.

Декомпозиція процесу формування рекомендацій методом колаборативної фільтрації представлена на рисунку 2.4. Ця діаграма деталізує логіку роботи рекомендаційного алгоритму, який формує рекомендації на основі аналізу взаємодій користувачів із товарами. На відміну від контентно-орієнтованого підходу, у цьому випадку основну роль відіграють поведінкові дані користувачів та закономірності їхньої взаємодії з товарами маркетплейсу.

Першим етапом процесу є збір та підготовка даних про взаємодії користувачів із товарами. На цьому етапі система отримує інформацію про історію взаємодії користувачів із товарами, зокрема перегляди сторінок товарів, додавання товарів до кошика або інші дії користувачів у системі. Також враховуються поведінкові дані користувачів, що відображають їхню активність на платформі. Отримані дані обробляються та структуруються для подальшого використання в алгоритмах рекомендацій.

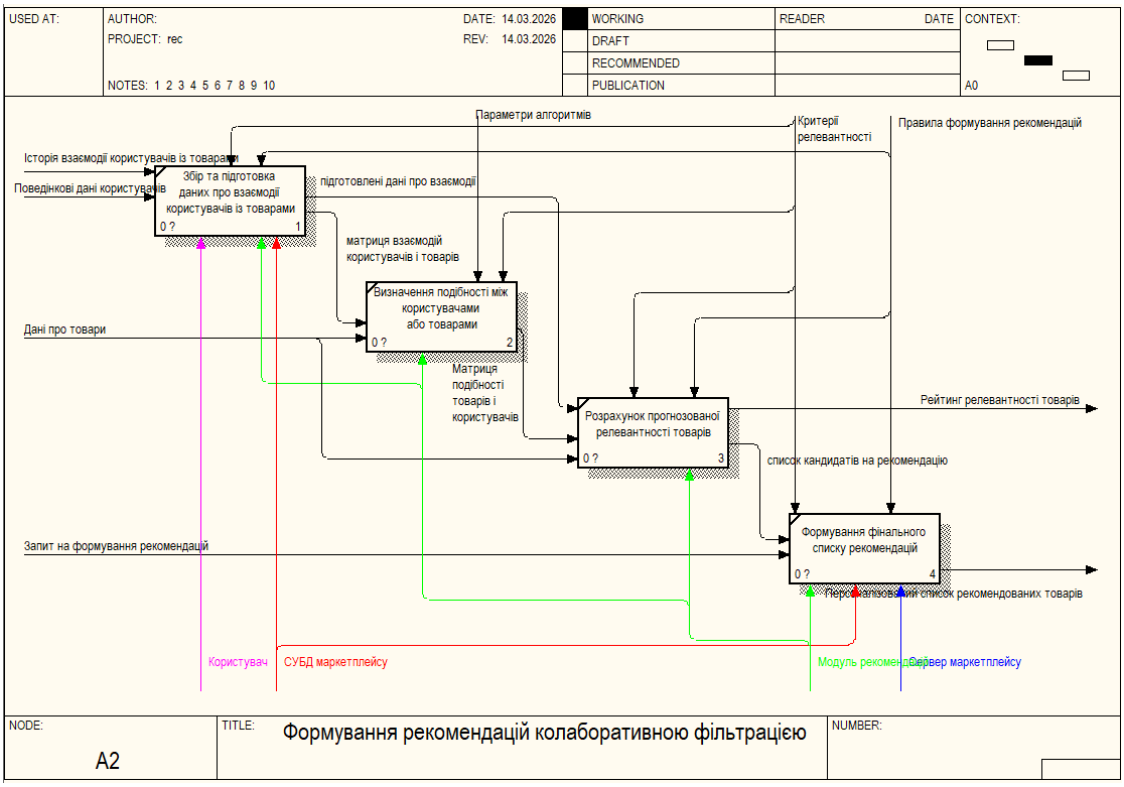

Рисунок 2.4 – Діаграма декомпозиції A2

Другим етапом є визначення подібності між користувачами або товарами. На основі підготовлених даних формується матриця взаємодій користувачів і товарів, яка відображає зв’язки між користувачами та об’єктами маркетплейсу. За допомогою цієї матриці система визначає ступінь подібності між користувачами або між товарами, що дозволяє виявити схожі патерни поведінки. Результатом цього етапу є формування матриці подібності, яка використовується для подальшого прогнозування рекомендацій.

Наступним етапом є розрахунок прогнозованої релевантності товарів. На основі отриманої матриці подібності система обчислює ймовірну релевантність товарів для конкретного користувача. Для цього аналізуються взаємодії інших користувачів із подібними інтересами або взаємозв’язки між товарами, що часто переглядаються або купуються разом. У результаті формується список кандидатів на рекомендацію разом із відповідними оцінками релевантності.

Завершальним етапом є формування фінального списку рекомендацій. На цьому етапі система відбирає найбільш релевантні товари з отриманого

списку кандидатів та виконує їх ранжування відповідно до визначених критеріїв релевантності. У результаті формується персоналізований список рекомендованих товарів, який передається до інтерфейсу маркетплейсу для подальшого відображення користувачеві. Додатково формується рейтинг релевантності товарів, який використовується під час сортування результатів.

Реалізація зазначених етапів здійснюється за допомогою модуля рекомендацій, системи керування базою даних маркетплейсу, серверної частини платформи та користувача системи. Керування процесом виконується на основі правил формування рекомендацій, параметрів алгоритмів та критеріїв релевантності, що визначають принципи роботи рекомендаційної системи маркетплейсу.

Декомпозиція процесу формування рекомендацій гібридним методом представлена на рисунку 2.5. Ця діаграма деталізує роботу алгоритму рекомендацій, який поєднує кілька підходів до формування рекомендацій для підвищення точності та релевантності результатів. Гібридний підхід дозволяє одночасно використовувати характеристики товарів, поведінкові дані користувачів та результати роботи інших алгоритмів рекомендацій.

Першим етапом процесу є отримання даних для формування рекомендацій. На цьому етапі система отримує необхідну інформацію з різних джерел, зокрема дані про товари, дані про користувачів, історію взаємодії користувачів із товарами, поведінкові та контекстні дані користувачів. Отримані дані обробляються та підготовлюються для подальшого використання алгоритмами рекомендацій.

Другим етапом є формування рекомендацій контентно-орієнтованим методом. На цьому етапі система аналізує характеристики товарів та профіль інтересів користувача для визначення товарів, які мають подібні властивості до тих, що раніше переглядав або використовував користувач. У результаті формується список рекомендацій, отриманий за допомогою контентно-орієнтованого підходу.

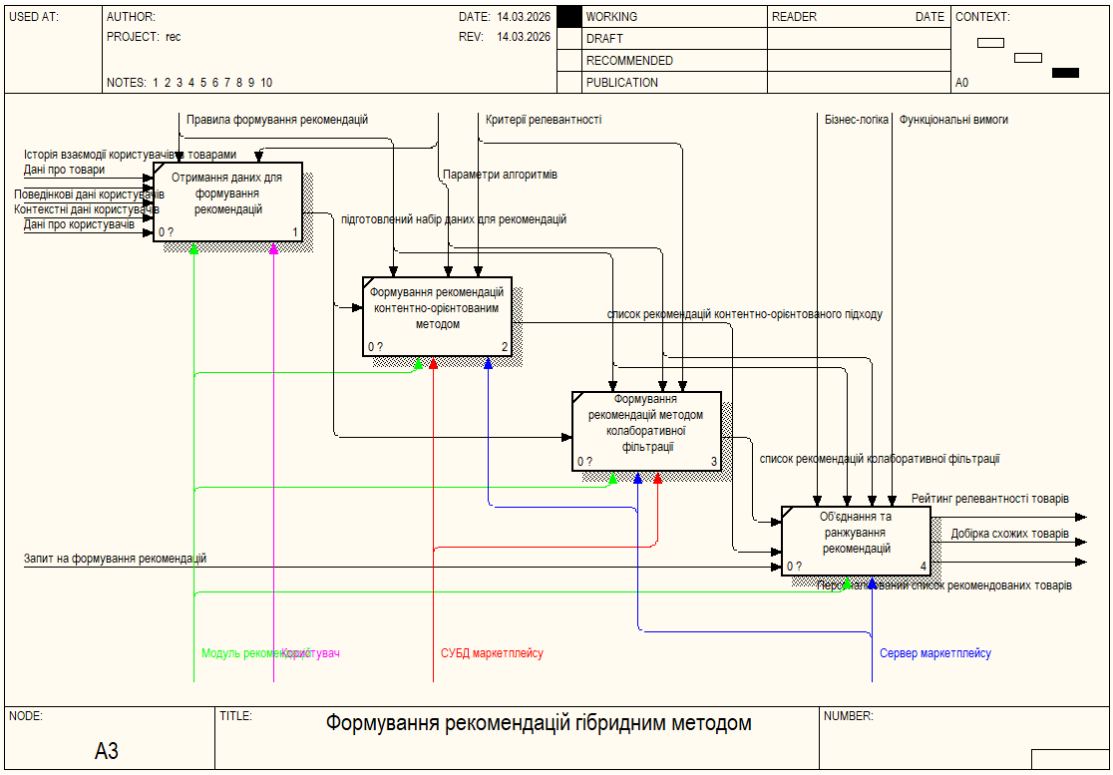

Рисунок 2.5 – Діаграма декомпозиції A3

Третім етапом є формування рекомендацій методом колаборативної фільтрації. У межах цього процесу система використовує поведінкові дані користувачів та історію їхніх взаємодій із товарами для виявлення закономірностей у поведінці користувачів. На основі таких закономірностей формується окремий список рекомендацій, який відображає товари, що можуть бути цікавими для користувача з урахуванням поведінки інших користувачів.

Завершальним етапом є об’єднання та ранжування рекомендацій. На цьому етапі система поєднує результати, отримані контентно-орієнтованим методом та методом колаборативної фільтрації. Отримані рекомендації аналізуються та ранжуються відповідно до визначених критеріїв релевантності. У результаті формується фінальний персоналізований список рекомендованих товарів, а також рейтинг релевантності товарів та добірка схожих товарів, які використовуються для відображення рекомендацій у системі маркетплейсу.

Функціонування процесу забезпечується модулем рекомендацій,

системою керування базою даних маркетплейсу, серверною частиною платформи та користувачем системи. Керування процесом здійснюється на основі правил формування рекомендацій, параметрів алгоритмів, критеріїв релевантності, бізнес-логіки маркетплейсу та функціональних вимог до системи, що визначають принципи роботи рекомендаційної системи.

Побудова функціональної моделі системи рекомендацій маркетплейсу за допомогою методології IDEF0 дозволила формалізувати основні процеси формування рекомендацій та визначити взаємозв’язки між компонентами системи. У межах побудованої моделі було визначено основний процес функціонування рекомендаційної системи, а також виконано його декомпозицію на окремі підпроцеси, що відповідають різним підходам до формування рекомендацій.

У результаті моделювання було виділено три основні підходи до формування рекомендацій: контентно-орієнтований підхід, метод колаборативної фільтрації та гібридний підхід. Для кожного з них було побудовано окрему функціональну декомпозицію, яка відображає послідовність обробки даних та формування рекомендацій.

Побудована функціональна модель дозволяє наочно представити структуру системи рекомендацій маркетплейсу, визначити необхідні джерела даних, а також встановити взаємодію між алгоритмами рекомендацій та компонентами програмної системи. Отримані результати можуть використовуватися як основа для подальшого проєктування структури системи та реалізації алгоритмів рекомендацій.

2.2 Визначення критеріїв та метрик оцінювання якості рекомендацій

Під час дослідження ефективності рекомендаційних систем виникає необхідність об’єктивного оцінювання результатів роботи алгоритмів.

Оскільки рекомендації формуються на основі складних моделей аналізу даних, оцінити їхню якість лише за допомогою суб’єктивних спостережень або інтуїтивних припущень неможливо. Саме тому у дослідженнях рекомендаційних систем використовуються спеціальні метрики оцінювання.

Метрики оцінювання – це формалізовані математичні показники, які дозволяють кількісно визначити якість роботи алгоритму. Вони дають можливість оцінити, наскільки сформовані рекомендації відповідають очікуваним результатам, а також дозволяють порівнювати ефективність різних алгоритмів між собою. Використання таких метрик є стандартною практикою у задачах машинного навчання та систем рекомендацій.

Основною перевагою використання метрик є можливість усунути суб’єктивність під час оцінювання результатів. Без застосування формальних показників оцінювання дослідник змушений покладатися на власні спостереження або окремі приклади рекомендацій, що може призводити до неточних або упереджених висновків. Метрики дозволяють перетворити оцінювання якості рекомендацій на формалізований процес, який базується на числових показниках.

Крім того, використання метрик дозволяє виконувати порівняльний аналіз різних алгоритмів рекомендацій. У межах одного дослідження можуть реалізовуватися кілька підходів до формування рекомендацій, наприклад контентно-орієнтований підхід, колаборативна фільтрація та гібридні алгоритми. Використання єдиних метрик оцінювання дозволяє визначити, який із цих алгоритмів демонструє кращі результати на однаковому наборі даних.

Також метрики дозволяють оцінити різні аспекти якості рекомендацій, зокрема точність рекомендацій, повноту, якість ранжування або різноманітність запропонованих товарів. Це дає можливість більш детально проаналізувати поведінку алгоритмів та визначити їх сильні та слабкі сторони.

У системах рекомендацій використовується значна кількість метрик, які дозволяють оцінювати різні аспекти якості роботи алгоритмів. Частина з них

оцінює точність рекомендацій, інші характеризують якість ранжування елементів у списку, а також здатність алгоритму знаходити різноманітні або нові рекомендації. Вибір конкретних метрик залежить від цілей дослідження та особливостей реалізованої рекомендаційної системи.

Однією з найбільш поширених метрик є Precision (точність). Ця метрика визначає частку релевантних товарів серед усіх рекомендованих. Іншими словами, Precision показує, яка частина запропонованих системою товарів дійсно відповідає інтересам користувача. Чим вище значення цієї метрики, тим більш точними є рекомендації.

Ще однією важливою метрикою є Recall (повнота). Вона характеризує здатність алгоритму знаходити всі релевантні товари серед доступних у системі. Якщо Precision оцінює точність рекомендацій, то Recall показує, наскільки повно система знаходить потенційно цікаві для користувача об’єкти.

Для поєднання показників точності та повноти часто використовується метрика F1-score. Вона представляє собою гармонійне середнє значень Precision та Recall і дозволяє отримати узагальнену оцінку якості рекомендацій. Ця метрика особливо корисна у випадках, коли необхідно одночасно враховувати як точність рекомендацій, так і їх повноту.

Для оцінювання якості ранжування рекомендацій використовується метрика NDCG (Normalized Discounted Cumulative Gain). Вона враховує не лише наявність релевантних товарів у списку рекомендацій, але й їх позицію у цьому списку. Чим ближче релевантні товари знаходяться до початку списку рекомендацій, тим вище значення цієї метрики.

Ще однією метрикою, яка використовується для оцінювання позицій релевантних елементів у списку рекомендацій, є MAP (Mean Average Precision). Вона обчислює середню точність рекомендацій з урахуванням позиції кожного релевантного елемента. Така метрика дозволяє оцінити, наскільки ефективно система ранжує рекомендовані об’єкти.

У рекомендаційних системах також використовується метрика Hit Rate.

Вона показує, чи присутній хоча б один релевантний товар у сформованому списку рекомендацій. Ця метрика часто застосовується для оцінювання рекомендаційних систем, де користувач взаємодіє лише з невеликою кількістю рекомендованих товарів.

Ще однією важливою характеристикою рекомендацій є Coverage (покриття). Ця метрика показує, яку частину всіх товарів системи рекомендаційний алгоритм здатний пропонувати користувачам. Високе значення покриття означає, що система використовує широкий спектр товарів у рекомендаціях, а не обмежується лише невеликою кількістю популярних позицій.

Також у рекомендаційних системах використовується метрика Diversity (різноманітність), яка оцінює ступінь відмінності між рекомендованими товарами. Висока різноманітність рекомендацій дозволяє пропонувати користувачам більш широкий вибір товарів і зменшує ризик того, що всі рекомендації будуть надто схожими між собою.

Для проведення експериментального дослідження ефективності рекомендаційних алгоритмів необхідно обрати набір метрик, які дозволять оцінити різні аспекти якості сформованих рекомендацій. Оскільки рекомендаційна система повинна не лише пропонувати релевантні товари, але й правильно ранжувати їх у списку та забезпечувати різноманітність рекомендацій, доцільно використовувати кілька взаємодоповнюючих метрик.

Для подальшого дослідження було обрано п’ять основних метрик оцінювання: Precision, Recall, F1-score, NDCG та Diversity. Використання саме цього набору метрик дозволяє оцінити як точність рекомендацій, так і якість їх ранжування та різноманітність запропонованих товарів.

2.3 Визначення архітектурних особливостей реалізації системи на платформі .NET

Реалізація інформаційної системи рекомендацій маркетплейсу потребує

вибору відповідної архітектурної моделі, яка дозволить забезпечити масштабованість, підтримуваність та розширюваність програмного забезпечення. У межах розроблюваної системи для організації взаємодії між компонентами серверної частини було обрано архітектурний підхід, що базується на використанні платформи .NET та сучасних принципів побудови веб-сервісів.

Платформа .NET надає широкий набір інструментів для розробки серверних застосунків, зокрема механізми побудови REST API, систему керування залежностями, а також підтримку сучасних архітектурних підходів. Використання цієї платформи дозволяє створювати масштабовані веб-сервіси, які можуть обробляти значну кількість запитів та ефективно взаємодіяти з базами даних і зовнішніми компонентами системи.

Для організації внутрішньої структури серверної частини системи було обрано архітектурний патерн Mediator. Цей патерн призначений для зменшення рівня зв’язності між компонентами системи шляхом централізації взаємодії між ними. Замість прямої взаємодії між різними частинами програми всі запити передаються через спеціальний посередницький механізм, який відповідає за їх обробку та маршрутизацію.

Застосування патерна Mediator дозволяє розділити логіку обробки запитів від логіки роботи контролерів. Контролери в такій архітектурі виконують роль точки входу до системи та відповідають лише за приймання HTTP-запитів і передачу їх до відповідних обробників. Безпосередня бізнес-логіка при цьому реалізується у спеціалізованих компонентах, що виконують обробку запитів.

У межах розроблюваної системи кожна операція API реалізується у вигляді окремого набору класів. Такий підхід дозволяє чітко розділити структуру даних запиту, структуру відповіді та логіку виконання операції. Для цього кожна операція складається з трьох основних елементів: request, response та operation.

Клас request використовується для опису структури даних, які

передаються до API під час виконання певного запиту. У цьому класі визначаються параметри, необхідні для виконання відповідної операції. Такий підхід дозволяє формалізувати структуру вхідних даних та забезпечити їхню валідацію.

Клас response використовується для опису структури даних, які повертає система у відповідь на виконання запиту. У ньому визначається набір полів, що містять результати виконання операції. Це дозволяє забезпечити стандартизований формат відповідей API та спростити обробку результатів на стороні клієнтського застосунку.

Клас operation містить безпосередню логіку виконання запиту. У цьому класі реалізується обробка отриманих даних, виконуються необхідні операції з базою даних та формуються результати, що передаються у відповідь клієнту. Такий підхід дозволяє ізолювати бізнес-логіку від інших компонентів системи та підвищити модульність програмного забезпечення.

Кожна операція системи реєструється у механізмі Dependency Injection, який є вбудованим компонентом платформи .NET. Використання цього механізму дозволяє автоматично керувати створенням та життєвим циклом об’єктів у системі. Це значно спрощує взаємодію між різними компонентами програми та забезпечує більш гнучке управління залежностями.

У контролерах системи доступ до операцій реалізується за допомогою механізму інжекції залежностей. Для цього відповідні операції передаються до контролера як залежності. Такий підхід дозволяє уникнути прямого створення об’єктів у коді контролера та забезпечує більш чисту архітектуру застосунку.

Отримання операцій у контролерах виконується за допомогою механізму [FromServices], який дозволяє отримувати необхідні сервіси безпосередньо з контейнера залежностей. У цьому випадку контролер отримує готовий екземпляр операції, який уже налаштований у системі керування залежностями.

Такий підхід дозволяє значно зменшити зв’язність між компонентами системи, оскільки контролери більше не залежать від конкретних реалізацій

бізнес-логіки. Замість цього вони взаємодіють із абстрактними операціями, що виконують необхідні дії.

Застосування архітектурного патерна Mediator у поєднанні з механізмами Dependency Injection платформи .NET дозволяє створити гнучку та модульну архітектуру серверної частини системи. Така структура полегшує підтримку коду, спрощує додавання нових функцій та забезпечує зручну організацію логіки роботи рекомендаційної системи маркетплейсу.

2.4 Проєктування структури бази даних маркетплейсу

Проєктування структури бази даних є важливим етапом розробки інформаційної системи маркетплейсу, оскільки саме база даних забезпечує зберігання та обробку інформації про товари, користувачів, їх взаємодії та інші дані, необхідні для функціонування системи рекомендацій. Правильно спроєктована структура бази даних дозволяє забезпечити ефективне зберігання даних, швидкий доступ до них та можливість подальшого розширення системи.

Для реалізації бази даних у межах розроблюваної системи використовується система керування базами даних Microsoft SQL Server. Ця СУБД є однією з найбільш поширених платформ для створення корпоративних інформаційних систем і широко використовується у застосунках, що розробляються на платформі .NET. Використання Microsoft SQL Server забезпечує високу продуктивність роботи з даними, надійність зберігання інформації та підтримку складних запитів до бази даних.

Створення та налаштування бази даних здійснюється за допомогою інструмента Microsoft SQL Server Management Studio (SSMS). Це спеціалізоване середовище для керування сервером баз даних, яке дозволяє створювати бази даних, проєктувати таблиці, визначати зв’язки між ними, а

також виконувати адміністрування та моніторинг роботи системи.

У межах реалізації інформаційної системи база даних маркетплейсу створюється безпосередньо у середовищі Microsoft SQL Server Management Studio. Під час розробки у цьому середовищі створюється нова база даних, у якій визначаються необхідні таблиці, атрибути та зв’язки між ними. Такий підхід дозволяє одразу розгорнути базу даних на сервері та використовувати її для подальшої розробки серверної частини застосунку.

Після створення бази даних у SSMS виконується проєктування її структури, яке включає визначення основних сутностей предметної області маркетплейсу, таких як користувачі, товари, категорії товарів, взаємодії користувачів із товарами та інші елементи системи. Для кожної сутності створюється відповідна таблиця, у якій визначаються атрибути, типи даних та ключові обмеження.

Також у процесі проєктування визначаються зв’язки між таблицями, що дозволяє відобразити логічну структуру предметної області маркетплейсу. Використання зовнішніх ключів забезпечує цілісність даних та дозволяє коректно відображати взаємозв’язки між різними об’єктами системи.

Створена база даних надалі використовується серверною частиною системи, реалізованою на платформі .NET. Серверний застосунок виконує операції читання та запису даних у базу даних, а також використовує ці дані для формування рекомендацій користувачам маркетплейсу. Таким чином, база даних виступає центральним компонентом системи, що забезпечує зберігання інформації та підтримує роботу рекомендаційних алгоритмів.

Для забезпечення коректної роботи рекомендаційної системи база даних маркетплейсу повинна містити не лише стандартні сутності електронної комерції, але й структури, що дозволяють зберігати інформацію про взаємодію користувачів із товарами та їхні характеристики. Наявність таких даних є критично важливою для реалізації різних підходів до формування рекомендацій, зокрема контентно-орієнтованого підходу, колаборативної фільтрації та гібридного методу.

Однією з ключових особливостей бази даних є наявність детального опису характеристик товарів. Для контентно-орієнтованого підходу система повинна мати доступ до атрибутів товарів, які можуть використовуватися для визначення схожості між ними. До таких характеристик можуть належати категорія товару, бренд, технічні параметри, опис, ціновий діапазон та інші властивості. Саме ці дані дозволяють алгоритму визначати подібні товари та формувати рекомендації на основі їхніх характеристик.

Другою важливою особливістю є зберігання інформації про користувачів системи. База даних повинна містити таблиці, що описують профілі користувачів, їх ідентифікатори та основні атрибути. Наявність таких даних дозволяє пов’язувати конкретні дії з відповідними користувачами та формувати персоналізовані рекомендації.

Ще одним важливим елементом структури бази даних є зберігання історії взаємодій користувачів із товарами. До таких взаємодій можуть належати перегляди сторінок товарів, додавання товарів до кошика, додавання до списку обраного або здійснення покупок. Саме ці дані використовуються алгоритмами колаборативної фільтрації для визначення закономірностей у поведінці користувачів та побудови матриці взаємодій між користувачами і товарами.

Також база даних повинна підтримувати зберігання поведінкових даних користувачів. До таких даних можуть належати частота взаємодії з товарами, тривалість перегляду сторінок, послідовність переглянутих товарів або інші показники активності користувача. Наявність таких даних дозволяє більш точно визначати інтереси користувачів і підвищувати якість рекомендацій.

Крім поведінкових даних, у базі даних можуть зберігатися контекстні дані користувачів, які враховують умови взаємодії користувача із системою. До таких даних можуть належати час виконання дії, тип пристрою користувача, місцезнаходження або інші фактори, що можуть впливати на релевантність рекомендацій.

Ще однією важливою особливістю структури бази даних є можливість

ефективного виконання аналітичних запитів. Оскільки рекомендаційні алгоритми працюють із великими обсягами даних, структура таблиць повинна забезпечувати швидкий доступ до інформації та можливість виконання складних запитів для аналізу взаємодій користувачів і товарів.

На рисунку 2.6 представлено діаграму структури бази даних маркетплейсу, яка відображає основні сутності системи та зв'язки між ними. Дана модель бази даних забезпечує зберігання інформації, необхідної як для функціонування самого маркетплейсу, так і для роботи системи формування рекомендацій.

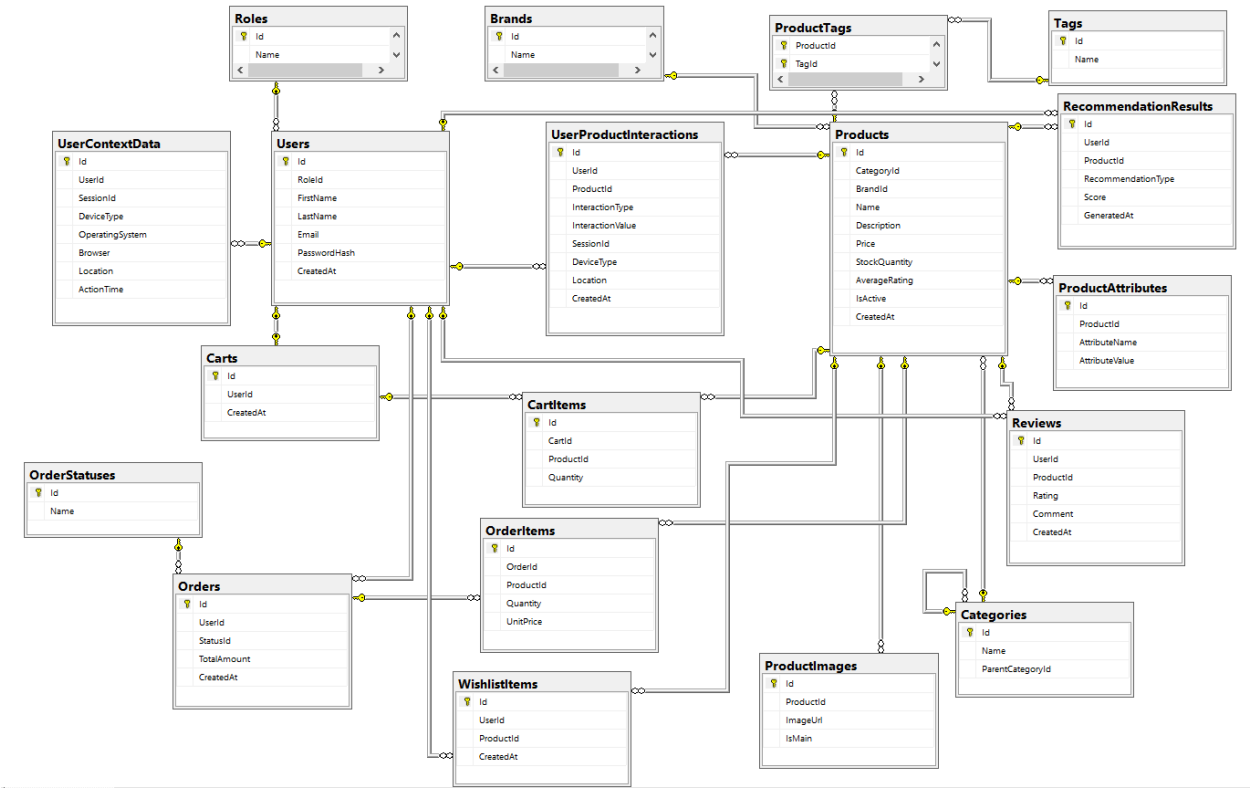

Рисунок 2.6 – Діаграма бази даних системи маркетплейсу

Центральним елементом структури є таблиця Products, у якій зберігається основна інформація про товари. Вона містить такі атрибути, як назва товару, опис, ціна, рейтинг, кількість доступних одиниць на складі та ознака активності товару. Таблиця пов'язана з таблицями Categories та Brands, що дозволяє класифікувати товари за категоріями та виробниками. Ієрархічна структура категорій реалізована за допомогою поля ParentCategoryId, що дає

можливість формувати вкладені категорії товарів.

Для розширеного опису товарів у системі використовується таблиця ProductAttributes, яка дозволяє зберігати набір характеристик товарів у вигляді пар «назва – значення». Крім того, таблиця ProductImages призначена для зберігання посилань на зображення товарів. Для семантичної класифікації товарів використовується механізм тегів, реалізований через таблиці Tags та ProductTags, що дозволяє одному товару відповідати декільком тегам.

Інформація про користувачів системи зберігається у таблиці Users, яка містить персональні дані користувача та посилання на таблицю Roles, що визначає роль користувача в системі. Такий підхід дозволяє реалізувати механізм керування правами доступу до функціоналу маркетплейсу.

Важливу роль у системі рекомендацій відіграє таблиця UserProductInteractions, яка зберігає інформацію про взаємодії користувачів із товарами. У цій таблиці фіксуються різні типи взаємодій, зокрема перегляд товару, додавання до кошика або інші дії користувача. Наявність таких даних дозволяє аналізувати поведінку користувачів та використовувати її при формуванні рекомендацій, особливо у межах алгоритмів колаборативної фільтрації.

Для збереження контекстної інформації про дії користувачів використовується таблиця UserContextData, у якій фіксуються дані про сесію користувача, тип пристрою, браузер, операційну систему, місцезнаходження та час виконання дії. Ця інформація може використовуватися для більш точного аналізу поведінкових характеристик користувачів.

Операції, пов’язані з покупками товарів, реалізуються через таблиці Orders та OrderItems, що дозволяють зберігати інформацію про оформлені замовлення та товари, які входять до складу кожного замовлення. Статуси замовлень зберігаються у таблиці OrderStatuses. Окрім цього, система підтримує роботу кошика користувача через таблиці Carts та CartItems, а також список бажаних товарів, який реалізовано через таблицю WishlistItems.

Оцінювання товарів користувачами здійснюється за допомогою таблиці

Reviews, де зберігаються рейтинги товарів та текстові відгуки. Ці дані можуть використовуватися для додаткового аналізу популярності товарів.

Окрему роль у системі відіграє таблиця RecommendationResults, яка використовується для збереження результатів роботи алгоритмів рекомендацій. У цій таблиці фіксується ідентифікатор користувача, рекомендований товар, тип використаного алгоритму рекомендацій та обчислене значення релевантності.

Представлена структура бази даних забезпечує зберігання як характеристик товарів, так і історії взаємодії користувачів із системою, що створює необхідну основу для реалізації контентно-орієнтованих, колаборативних та гібридних алгоритмів формування рекомендацій у маркетплейсі.

2.5 Підготовка набору даних для тестування рекомендаційних алгоритмів

У межах проведення експериментального дослідження ефективності алгоритмів рекомендацій виникає необхідність підготовки набору тестових даних, який дозволить змоделювати роботу маркетплейсу та сформувати достатній обсяг інформації для аналізу поведінки користувачів. Такий набір даних має містити різноманітні записи про користувачів, товари, їх характеристики, а також інформацію про взаємодії користувачів із товарами.

Для формування тестового набору даних було використано можливості генеративного штучного інтелекту, зокрема мовної моделі GPT. Використання штучного інтелекту дозволило автоматизувати процес створення великої кількості синтетичних записів, що імітують реальні дані маркетплейсу. Згенерований набір даних включає інформацію про користувачів, категорії товарів, бренди, характеристики товарів, а також різні типи взаємодій

користувачів із товарами, такі як перегляди сторінок товарів, додавання товарів до кошика, додавання до списку бажань та здійснення покупок.

Під час формування набору даних було забезпечено різноманітність користувачів та товарів, що дозволяє створити реалістичну модель функціонування маркетплейсу. Дані про користувачів включають основну інформацію про облікові записи, тоді як дані про товари містять відомості про категорії, бренди, характеристики та інші атрибути, необхідні для роботи контентно-орієнтованих алгоритмів рекомендацій. Окрему увагу було приділено генерації історії взаємодій користувачів із товарами, оскільки саме ці дані є основою для роботи алгоритмів колаборативної фільтрації.

Згенерований набір даних було збережено у форматі CSV, що забезпечує зручність подальшої обробки та імпорту до системи керування базами даних. Використання такого формату дозволяє легко інтегрувати підготовлені дані до створеної структури бази даних маркетплейсу, зокрема шляхом використання стандартних засобів імпорту даних у Microsoft SQL Server Management Studio.

На рисунку 2.7 представлено приклад фрагмента підготовленого набору даних, який використовується для заповнення таблиці Products у базі даних маркетплейсу. Дані представлені у табличному вигляді та були сформовані у форматі CSV, що дозволяє зручно імпортувати їх до системи керування базами даних.

Кожен запис у файлі відповідає окремому товару, який може бути представлений у каталозі маркетплейсу. Для кожного товару зберігається набір атрибутів, необхідних для коректної роботи системи рекомендацій. Зокрема, поле Id містить унікальний ідентифікатор товару, який використовується для встановлення зв’язків між таблицями бази даних. Поля CategoryId та BrandId визначають категорію товару та виробника відповідно, що дозволяє реалізувати механізми класифікації товарів.

Поле Name містить назву товару, тоді як Description зберігає короткий текстовий опис. Ці дані можуть використовуватися у контентно-орієнтованих алгоритмах рекомендацій для аналізу характеристик товарів та визначення

їхньої схожості.

Id	Category Id	Brand Id	Name	Description	Price	Stock Quantity	Average Rating	Is Active	Created At
1	101	201	Wireless Mouse	Ergonomic wirel	19.99	120	4.5	1	2026-03-10
2	101	202	Gaming Mouse	High precision g	39.99	80	4.7	1	2026-03-10
3	102	203	Mechanical Keyb	RGB mechanical	89.99	60	4.6	1	2026-03-10
4	102	203	Office Keyboard	Compact office l	29.99	150	4.3	1	2026-03-10
5	103	204	27 inch Monitor	Full HD IPS mon	189.99	40	4.4	1	2026-03-10
6	103	205	24 inch Monitor	LED office monit	139.99	55	4.2	1	2026-03-10
7	104	206	Laptop Stand	Adjustable alum	34.99	100	4.5	1	2026-03-10
8	104	207	Cooling Pad	Laptop cooling p	27.99	75	4.1	1	2026-03-10
9	105	208	USB-C Hub	Multiport USB-C	49.99	90	4.4	1	2026-03-10
10	105	208	USB Hub	4-port USB hub	15.99	200	4	1	2026-03-10
11	106	209	External SSD 1TB	Portable high sp	129.99	65	4.8	1	2026-03-10
12	106	209	External HDD 2T	Portable storage	89.99	70	4.3	1	2026-03-10
13	107	210	Gaming Headset	Surround sound	59.99	85	4.6	1	2026-03-10
14	107	211	Bluetooth Headp	Wireless noise c	119.99	50	4.7	1	2026-03-10
15	108	212	Webcam HD	1080p streaming	44.99	95	4.2	1	2026-03-10
16	108	212	Webcam 4K	Ultra HD webcar	79.99	45	4.6	1	2026-03-10
17	109	213	Gaming Chair	Ergonomic gami	199.99	30	4.5	1	2026-03-10
18	109	213	Office Chair	Comfortable offi	149.99	35	4.3	1	2026-03-10
19	110	214	Desk Lamp	LED desk lamp w	24.99	110	4.4	1	2026-03-10
20	110	214	Smart Lamp	WiFi smart desk	39.99	90	4.5	1	2026-03-10
21	111	215	Portable Speake	Bluetooth portal	49.99	100	4.4	1	2026-03-10
22	111	215	Smart Speaker	Voice assistant s	79.99	70	4.6	1	2026-03-10
23	112	216	Power Bank 100	Compact mobile	29.99	120	4.3	1	2026-03-10
24	112	216	Power Bank 200	High capacity po	39.99	95	4.5	1	2026-03-10
25	113	217	Smart Watch	Fitness tracking	149.99	60	4.4	1	2026-03-10
26	113	217	Fitness Band	Lightweight fitne	59.99	130	4.2	1	2026-03-10
27	114	218	Tablet 10 inch	Android tablet d	229.99	40	4.3	1	2026-03-10
28	114	218	Tablet 8 inch	Compact tablet	179.99	55	4.1	1	2026-03-10
29	115	219	Printer Inkjet	Home inkjet prir	119.99	35	4.2	1	2026-03-10

Рисунок 2.7 – Приклад згенерованого файлу тестових даних

Фінансові та кількісні характеристики товарів представлені у полях Price та StockQuantity, які зберігають інформацію про вартість товару та кількість одиниць, доступних на складі. Поле AverageRating містить середню оцінку товару, сформовану на основі відгуків користувачів, що дозволяє враховувати популярність товару при формуванні рекомендацій.

Поле IsActive використовується для визначення статусу товару у каталозі маркетплейсу, що дозволяє відображати лише доступні товари. Поле CreatedAt зберігає дату додавання товару до системи.

Наведений фрагмент набору даних демонструє приклад синтетично згенерованої інформації про товари, яка використовується для наповнення бази даних та подальшого тестування алгоритмів формування рекомендацій.

Використання такого тестового набору дозволяє змоделювати роботу реального маркетплейсу та забезпечити достатній обсяг даних для проведення експериментального дослідження ефективності різних підходів до формування рекомендацій.

Таким чином, сформований набір синтетичних даних створює основу для проведення тестування алгоритмів рекомендацій. Наявність різноманітних записів про користувачів, товари та їх взаємодії дозволяє оцінити ефективність різних підходів формування рекомендацій та провести подальший аналіз їхньої якості за допомогою обраних метрик оцінювання.

3 РОЗРОБКА РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

3.1 Реалізація базового функціоналу маркетплейсу на платформі .NET

У межах реалізації програмної частини системи передбачено створення базового функціоналу серверної частини маркетплейсу на платформі .NET. Серверна частина системи відповідає за обробку запитів користувачів, взаємодію з базою даних, отримання та збереження інформації про товари, користувачів, замовлення, взаємодії з товарами та результати роботи рекомендаційних алгоритмів. Основний акцент реалізації робиться саме на backend-частині, оскільки вона забезпечує логіку роботи системи та формує дані, необхідні для подальшого відображення на клієнтській стороні.

Базовий функціонал маркетплейсу включає реалізацію операцій для роботи з основними сутностями системи. До таких сутностей належать товари, категорії, бренди, користувачі, кошик, список бажань, замовлення, відгуки та взаємодії користувачів із товарами. Наявність такого функціоналу є необхідною умовою для подальшої роботи рекомендаційної підсистеми, оскільки алгоритми рекомендацій використовують інформацію про товари, їх характеристики та історію дій користувачів у системі.

Для організації серверної логіки використовується архітектурний підхід, побудований на основі патерна Mediator. Його основна ідея полягає у зменшенні прямої залежності між компонентами системи за рахунок винесення логіки виконання окремих запитів у спеціальні обробники. У межах розроблюваної системи контролери не містять бізнес-логіки, а виконують роль точки входу для HTTP-запитів. Після отримання запиту контролер передає його до відповідної операції, яка реалізує необхідну логіку обробки.

Використання такого підходу є доцільним для даного проєкту, оскільки система містить значну кількість різних операцій, пов’язаних із роботою маркетплейсу та рекомендаційної підсистеми. Якщо розміщувати всю логіку

безпосередньо у контролерах, код швидко стає складним для підтримки, розширення та тестування. Винесення кожної дії в окрему операцію дозволяє зробити структуру системи більш зрозумілою та модульною.

Кожна реалізація запиту до API оформлюється у вигляді окремого набору файлів, що містять структуру запиту, структуру відповіді та операцію обробки. Файл request описує вхідні параметри, які передаються до API. Файл response визначає структуру даних, які повертаються клієнту після виконання запиту. Файл operation містить основну логіку обробки запиту, включаючи звернення до бази даних, фільтрацію, сортування, формування результату та повернення відповіді.

Операції підключаються до контролерів за допомогою механізму Dependency Injection, який є вбудованою частиною платформи .NET. Це дозволяє не створювати екземпляри класів вручну, а отримувати їх із контейнера залежностей. У методах контролерів такі операції можуть передаватися через атрибут [FromServices], що дозволяє явно вказати, що відповідний об’єкт має бути отриманий із DI-контейнера.

Перевагою такого підходу є чітке розділення відповідальності між компонентами системи. Контролер відповідає лише за приймання HTTP-запиту та повернення результату, тоді як уся бізнес-логіка зосереджується в окремих операціях. Це спрощує підтримку програмного коду, дозволяє легше додавати нові функції та робить реалізацію більш придатною для модульного тестування.

Приклад реалізації такої операції наведено на лістингу 3.1. У ньому показано операцію отримання списку товарів маркетплейсу. Клас GetProductsOperation отримує залежність MarketplaceDbContext через конструктор, що дозволяє виконувати запити до бази даних за допомогою Entity Framework Core. У методі ExecuteAsync виконується обробка параметрів сторінки та кількості елементів на сторінці, після чого формується запит до таблиці товарів.

Лістинг 3.1 – Приклад реалізації операцій системи

```
using MarketplaceRecommendation.Api.Data;
using Microsoft.EntityFrameworkCore;
namespace MarketplaceRecommendation.Api.Operations.Products.GetProducts;
public class GetProductsOperation
{
    private readonly MarketplaceDbContext _db;
    public GetProductsOperation(MarketplaceDbContext db) => _db = db;
     public  async  Task<GetProductsResponse>  ExecuteAsync(GetProductsRequest
request)
    {
        var page = request.Page < 1 ? 1 : request.Page;
       var pageSize = request.PageSize is < 1 or > 100 ? 20 : request.PageSize;
        var query = _db.Products
            .Include(p => p.Category)
            .Include(p => p.Brand)
            .Include(p => p.Images);
        var total = await query.CountAsync();
        var items = await query
            .OrderByDescending(p => p.CreatedAt)
            .Skip((page - 1) * pageSize)
            .Take(pageSize)
            .Select(p => new ProductListItem
            {
                Id           = p.Id,
                Name         = p.Name,
                CategoryName = p.Category.Name,
                BrandName    = p.Brand.Name,
                Price        = p.Price,
                AverageRating= p.AverageRating,
                StockQuantity= p.StockQuantity,
                IsActive     = p.IsActive,
                   MainImageUrl  =  p.Images.Where(i  =>  i.IsMain).Select(i  =>
i.ImageUrl).FirstOrDefault()
                                                          ??  p.Images.Select(i  =>
i.ImageUrl).FirstOrDefault()
            })
            .ToListAsync();
        return new GetProductsResponse
        {
            Items      = items,
            TotalCount = total,
            Page       = page,
            PageSize   = pageSize
        };
    }
}
```

У процесі виконання операції застосовується завантаження пов’язаних даних про категорію, бренд та зображення товару. Це дозволяє сформувати повноцінний список товарів, який містить не лише базову інформацію про товар, але й додаткові дані, необхідні для відображення у каталозі маркетплейсу. Також у запиті реалізовано сортування товарів за датою створення, пагінацію та проєкцію результатів у DTO-модель ProductListItem.

Результатом виконання операції є об’єкт GetProductsResponse, який містить список товарів, загальну кількість записів, номер поточної сторінки та розмір сторінки. Такий формат відповіді є зручним для подальшого використання на клієнтській стороні, оскільки дозволяє реалізувати посторінкове відображення каталогу товарів.

На рисунку 3.1 представлено структуру реалізованих операцій серверної частини маркетплейсу. У межах системи було реалізовано базовий набір функцій, необхідних для забезпечення повноцінної роботи маркетплейсу та подальшого функціонування рекомендаційної підсистеми.

Структура реалізації побудована за принципом винесення кожної окремої операції у власний модуль. Для кожної функції створюється окрема директорія, яка містить файли request, response та operation. Такий підхід дозволяє логічно ізолювати різні типи операцій та забезпечити зрозумілу структуру серверної частини системи.

У системі реалізовано модуль авторизації користувачів, який містить операції реєстрації нового користувача та входу до системи. Ці операції забезпечують базову автентифікацію користувачів маркетплейсу та створюють основу для подальшої персоналізації рекомендацій.

Для роботи з брендами товарів реалізовано набір CRUD-операцій, який включає створення бренду, отримання списку брендів, оновлення інформації про бренд та його видалення. Аналогічний набір функцій реалізовано і для категорій товарів. Це дозволяє адмініструвати структуру каталогу маркетплейсу та підтримувати актуальність товарної класифікації.

Функціонал роботи з товарами реалізований через окремий набір операцій, що включає отримання списку товарів, отримання товару за ідентифікатором, створення нового товару, оновлення інформації про товар, видалення товару, пошук товарів та отримання товарів певної категорії. Саме цей функціонал формує основу каталогу маркетплейсу та забезпечує доступ до інформації про товари.

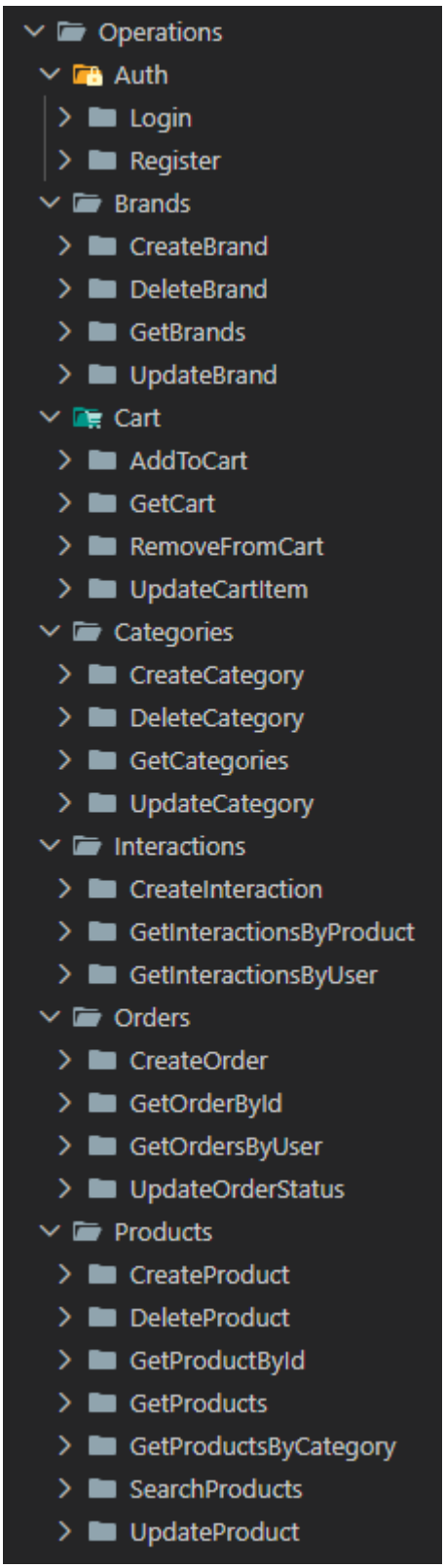

Рисунок 3.1 – Приклад реалізованих стандартних функцій в системі

У системі також реалізовано функціонал роботи з кошиком користувача. До нього належать операції додавання товару до кошика, отримання вмісту кошика, оновлення кількості товару та видалення товару з кошика. Реалізація

такого функціоналу є важливою складовою електронного маркетплейсу, оскільки дозволяє користувачам формувати замовлення.

Для роботи із замовленнями реалізовано операції створення нового замовлення, отримання інформації про конкретне замовлення, отримання списку замовлень користувача та зміни статусу замовлення. Дані про покупки користувачів у подальшому можуть використовуватись рекомендаційною системою для аналізу поведінкових характеристик користувачів.

Окрему роль у системі відіграє модуль взаємодій користувачів із товарами. У ньому реалізовано операції створення взаємодії, отримання взаємодій користувача та отримання взаємодій для конкретного товару. Саме ці дані використовуються алгоритмами рекомендацій для побудови профілів користувачів та визначення схожості між товарами або користувачами.

Таким чином, у межах серверної частини маркетплейсу було реалізовано повний базовий функціонал, необхідний для роботи електронного торговельного майданчика. Реалізована структура операцій створює основу для подальшого впровадження та тестування алгоритмів рекомендаційної системи.

3.2 Наповнення бази даних тестовими даними

Після формування структури бази даних та підготовки набору тестових даних наступним етапом реалізації системи стало наповнення бази даних інформацією про користувачів, товари, категорії, бренди та взаємодії користувачів із товарами. Оскільки тестовий набір даних був попередньо сформований у форматі CSV, виникла необхідність автоматизованого перенесення цих даних до бази даних маркетплейсу.

Для реалізації такого механізму у серверній частині системи було створено спеціальний модуль заповнення бази даних – **DbSeeder**. Основним

призначенням цього компонента є автоматичне створення тестових записів у таблицях бази даних під час запуску застосунку. Використання такого підходу дозволяє швидко отримати готове тестове середовище без необхідності ручного додавання великої кількості записів.

Механізм Seed використовується для початкового заповнення таблиць бази даних інформацією, необхідною для функціонування маркетплейсу та тестування рекомендаційних алгоритмів. У процесі виконання модуля до бази даних додаються категорії товарів, бренди, товари, користувачі, взаємодії користувачів із товарами, замовлення, відгуки та інші сутності системи.

Виклик механізму заповнення бази даних виконується під час запуску серверної частини застосунку. Для цього створюється окрема область видимості сервісів через CreateScope(), після чого із контейнера залежностей отримується екземпляр MarketplaceDbContext. Далі виконується перевірка існування бази даних за допомогою методу EnsureCreatedAsync(), а після цього запускається процедура автоматичного заповнення даними через виклик методу DbSeeder.SeedAsync(db).

Приклад виклику механізму заповнення бази даних наведено на лістингу 3.2. Представлений код демонструє процес автоматичного створення структури бази даних та її подальшого наповнення тестовими даними під час запуску застосунку.

Лістинг 3.2 – запуск сіду бази даних

```
using (var scope = app.Services.CreateScope())
{
                                        var                 db                  =
scope.ServiceProvider.GetRequiredService<MarketplaceDbContext>();
    await db.Database.EnsureCreatedAsync();
    await DbSeeder.SeedAsync(db);
}
```

Такий підхід значно спрощує процес тестування системи, оскільки дозволяє швидко відновлювати тестове середовище та повторно виконувати експерименти з різними алгоритмами рекомендацій. Крім того, автоматизоване наповнення бази даних забезпечує однаковий початковий

набір даних для всіх експериментів, що є важливим для коректного порівняння ефективності різних підходів формування рекомендацій.

3.3 Реалізація контентно-орієнтованого підходу до формування рекомендацій

Першим реалізованим підходом формування рекомендацій у системі став контентно-орієнтований метод. Основна ідея цього підходу полягає у формуванні рекомендацій на основі характеристик товарів, з якими користувач уже взаємодіяв раніше. У процесі роботи алгоритм аналізує категорії товарів, бренди, теги, атрибути, ціновий діапазон та інші характеристики, після чого визначає товари, які є найбільш схожими на ті, що вже викликали інтерес користувача.

Реалізація алгоритму складається з декількох основних етапів. На першому етапі система отримує всі активні товари разом із пов’язаними даними, необхідними для подальшого аналізу. Далі виконується отримання історії взаємодій конкретного користувача з товарами. На основі цих взаємодій формується профіль інтересів користувача, який містить інформацію про найбільш релевантні категорії, бренди, теги та характеристики товарів.

Після формування профілю користувача система переходить до оцінювання потенційних товарів-кандидатів для рекомендації. Для кожного товару обчислюється числовий показник релевантності, який враховує ступінь схожості характеристик товару з профілем користувача. Після завершення оцінювання товари сортуються за отриманим значенням релевантності, а система відбирає найбільш релевантні результати.

Отримані рекомендації зберігаються у таблиці RecommendationResults, що дозволяє використовувати результати рекомендацій у подальших етапах

роботи системи. На завершальному етапі формується відповідь API, яка містить інформацію про рекомендовані товари, їх назви, категорії, бренди, рейтинги, ціну та обчислений показник релевантності.

Першим етапом роботи алгоритму є отримання списку всіх активних товарів маркетплейсу разом із пов’язаними даними, необхідними для аналізу. Приклад реалізації цього етапу наведено на лістингу 3.3.

Лістинг 3.3 – Отримання списку активних товарів

```
var allProducts = await _db.Products
    .Include(p => p.Category)
    .Include(p => p.Brand)
    .Include(p => p.Images)
    .Include(p => p.Attributes)
    .Include(p => p.ProductTags)
    .Where(p => p.IsActive)
    .ToListAsync();
```

У представленому коді виконується звернення до таблиці Products за допомогою Entity Framework Core. Для кожного товару додатково завантажуються пов’язані сутності, зокрема категорія товару, бренд, зображення, характеристики та теги. Для цього використовуються методи Include(), які дозволяють виконати eager loading пов’язаних даних.

Також у запиті використовується фільтрація за полем IsActive, що дозволяє отримати лише активні товари, доступні для рекомендації користувачам. Після цього результати запиту перетворюються у список за допомогою методу ToListAsync().

Отриманий набір даних використовується як основа для подальшого аналізу товарів та обчислення їхньої схожості з інтересами користувача. Саме ці дані надалі беруть участь у побудові профілю користувача та розрахунку релевантності рекомендацій.

Наступним етапом роботи алгоритму є отримання інформації про взаємодії користувача з товарами. Приклад реалізації цього етапу наведено на лістингу 3.4.

Лістинг 3.4 – Отримання інформації про взаємодію користувача з товарами

```
var interactions = await _db.UserProductInteractions
    .Where(i => i.UserId == request.UserId)
    .ToListAsync();

if (!interactions.Any())
    return new List<RecommendationItemResponse>();

var interactedProductIds = interactions.Select(i => i.ProductId).ToHashSet();
var interactedProducts = allProducts.Where(p => interactedProductIds.Contains(p.Id)).ToList();
```

У представленому коді виконується звернення до таблиці UserProductInteractions, яка містить історію дій користувачів у системі. За допомогою фільтрації по полю UserId система отримує лише ті взаємодії, які належать конкретному користувачу, для якого формуються рекомендації.

Після отримання даних виконується перевірка на наявність взаємодій. Якщо користувач ще не виконував жодних дій із товарами, система повертає порожній список рекомендацій. Такий підхід дозволяє уникнути виконання подальших обчислень у випадках, коли відсутні дані для побудови профілю інтересів користувача.

Далі з отриманих взаємодій формується набір ідентифікаторів товарів, з якими користувач уже взаємодіяв. Для цього використовується структура HashSet, яка забезпечує швидкий пошук елементів. Після цього зі списку всіх товарів відбираються лише ті товари, які присутні в історії взаємодій користувача.

Наступним етапом роботи алгоритму є формування профілю інтересів користувача на основі товарів, з якими він взаємодіяв раніше. Приклад реалізації цього етапу наведено на лістингу 3.5.

Лістинг 3.5 – Формування профілю інтересів

```
var categoryWeights  = new Dictionary<int, double>();
var brandWeights     = new Dictionary<int, double>();
var tagWeights       = new Dictionary<int, int>();
```

```
            var attrValues       = new HashSet<string>();
            var prices           = new List<decimal>();

            foreach (var interaction in interactions)
            {
                var product = allProducts.FirstOrDefault(p => p.Id == interaction.ProductId);
                if (product is null) continue;
                double weight = RecommendationMathHelper.GetInteractionWeight(interaction.InteractionType);

                categoryWeights.TryGetValue(product.CategoryId, out var cw);
                categoryWeights[product.CategoryId] = cw + weight;

                brandWeights.TryGetValue(product.BrandId, out var bw);
                brandWeights[product.BrandId] = bw + weight;

                foreach (var pt in product.ProductTags)
                {
                    tagWeights.TryGetValue(pt.TagId, out var tw);
                    tagWeights[pt.TagId] = tw + 1;
                }

                foreach (var attr in product.Attributes)
                    attrValues.Add($»{attr.AttributeName}:{attr.AttributeValue}»);

                prices.Add(product.Price);
            }

            double avgUserPrice    = prices.Count > 0 ? (double)prices.Average() : 0;
              var preferredCats        = categoryWeights.OrderByDescending(kv => kv.Value).Take(3).Select(kv => kv.Key).ToHashSet();
               var preferredBrands      = brandWeights.OrderByDescending(kv => kv.Value).Take(3).Select(kv => kv.Key).ToHashSet();
                var preferredTags         = tagWeights.OrderByDescending(kv => kv.Value).Take(5).Select(kv => kv.Key).ToHashSet();
```

У представленому коді для кожної взаємодії користувача виконується пошук відповідного товару та визначення ваги взаємодії залежно від її типу. Більш важливі дії, наприклад покупка товару або додавання до кошика, отримують більшу вагу, ніж звичайний перегляд сторінки товару.

У процесі обробки взаємодій система накопичує інформацію про категорії товарів, бренди, теги та характеристики, які найчастіше зустрічаються серед товарів, що зацікавили користувача. Крім того, виконується збір інформації про ціновий діапазон товарів, із якими взаємодіяв користувач.

Після завершення аналізу формується набір найбільш релевантних категорій, брендів та тегів користувача. Отриманий профіль інтересів

використовується на наступному етапі для оцінювання схожості інших товарів та формування персоналізованих рекомендацій.

Після формування профілю інтересів користувача система переходить до етапу оцінювання товарів-кандидатів для рекомендації. Приклад реалізації цього етапу наведено на лістингу 3.6.

Лістинг 3.6 – Оцінювання товарів-кандидатів

```
                            var     candidates     =     allProducts.Where(p     =>
!interactedProductIds.Contains(p.Id)).ToList();
            var scores     = new Dictionary<int, double>();

            foreach (var product in candidates)
            {
                double categoryScore = preferredCats.Contains(product.CategoryId) ?
1.0 : 0.0;
                 double brandScore    = preferredBrands.Contains(product.BrandId) ?
1.0 : 0.0;

                       var  productTagIds      =  product.ProductTags.Select(pt  =>
pt.TagId).ToHashSet();
                double tagScore      = preferredTags.Count > 0
                        ? (double)preferredTags.Intersect(productTagIds).Count()  /
Math.Max(preferredTags.Union(productTagIds).Count(), 1)
                    : 0.0;

                        var  productAttrs        =  product.Attributes.Select(a  =>
$»{a.AttributeName}:{a.AttributeValue}»).ToHashSet();
                double attrScore     = attrValues.Count > 0
                           ?  (double)attrValues.Intersect(productAttrs).Count()  /
Math.Max(attrValues.Union(productAttrs).Count(), 1)
                    : 0.0;

                double priceRange    = avgUserPrice * 0.3;
                double priceSimilarity = priceRange > 0
                  ? Math.Max(0, 1.0 - Math.Abs((double)product.Price - avgUserPrice)
/ (avgUserPrice + 1))
                    : 1.0;

                double ratingScore   = product.AverageRating / 5.0;

                scores[product.Id] = categoryScore  * 0.30
                                   + brandScore     * 0.20
                                   + tagScore       * 0.20
                                   + attrScore      * 0.15
                                   + priceSimilarity * 0.10
                                   + ratingScore    * 0.05;
            }

            var topIds = scores.OrderByDescending(kv => kv.Value)
                               .Take(request.Count)
                               .Select(kv => kv.Key)
                               .ToList();
```

У представленому коді зі списку всіх товарів відбираються лише ті товари, з якими користувач ще не взаємодіяв. Для кожного такого товару виконується розрахунок числового показника релевантності на основі декількох характеристик.

Під час обчислення оцінки враховується належність товару до пріоритетних категорій користувача, збіг бренду, схожість тегів та характеристик товару, близькість ціни до середнього цінового діапазону користувача, а також середній рейтинг товару. Для кожного з цих параметрів обчислюється окремий показник схожості.

Після цього формується фінальний показник релевантності товару шляхом зваженого додавання всіх отриманих оцінок. Найбільшу вагу у формулі мають категорія товару та бренд, оскільки саме ці характеристики найбільш точно відображають інтереси користувача у межах маркетплейсу.

Після завершення обчислень товари сортуються за значенням релевантності, а система відбирає найбільш релевантні товари для подальшого формування списку рекомендацій.

Після завершення розрахунку релевантності товарів система переходить до етапу збереження сформованих рекомендацій у базі даних. Приклад реалізації цього етапу наведено на лістингу 3.7.

Лістинг 3.7 – Збереження сформованих рекомендацій

```
            var existing = await _db.RecommendationResults
                 .Where(r => r.UserId == request.UserId && r.RecommendationType == 
RecommendationType.ContentBased)
                .ToListAsync();
            _db.RecommendationResults.RemoveRange(existing);

            var results = topIds.Select(id => new RecommendationResult
            {
                UserId             = request.UserId,
                ProductId          = id,
                RecommendationType = RecommendationType.ContentBased,
                Score              = Math.Round(scores[id], 4),
                GeneratedAt        = DateTime.UtcNow
            }).ToList();

            _db.RecommendationResults.AddRange(results);
            await _db.SaveChangesAsync();
```

У представленому коді спочатку виконується пошук уже існуючих рекомендацій для поточного користувача, сформованих контентно-орієнтованим алгоритмом. Після цього попередні результати видаляються з таблиці RecommendationResults, що дозволяє уникнути дублювання застарілих рекомендацій.

Далі для кожного відібраного товару створюється новий об’єкт RecommendationResult, який містить ідентифікатор користувача, ідентифікатор товару, тип алгоритму рекомендацій, значення релевантності та дату формування рекомендації.

Після формування списку рекомендацій результати додаються до бази даних за допомогою методу AddRange(), а потім зберігаються через SaveChangesAsync(). Це дозволяє використовувати сформовані рекомендації у подальших етапах роботи системи та повторно отримувати їх без необхідності повторного виконання всіх обчислень.

Завершальним етапом роботи алгоритму є формування відповіді API для повернення списку рекомендацій клієнтській частині системи. Приклад реалізації цього етапу наведено на лістингу 3.8.

Лістинг 3.8 – Формування відповіді

```
return topIds.Select(id =>
{
    var product = candidates.First(p => p.Id == id);
    return new RecommendationItemResponse
    {
        ProductId          = product.Id,
        ProductName        = product.Name,
        CategoryName       = product.Category?.Name ?? string.Empty,
        BrandName          = product.Brand?.Name ?? string.Empty,
        Price              = product.Price,
        AverageRating      = product.AverageRating,
        Score              = Math.Round(scores[id], 4),
        RecommendationType = nameof(RecommendationType.ContentBased),
            ImageUrl              =  product.Images.FirstOrDefault(i =>
i.IsMain)?.ImageUrl
    };
}).ToList();
```

У представленому коді для кожного рекомендованого товару формується об’єкт RecommendationItemResponse, який містить основну

інформацію про товар. До відповіді включаються ідентифікатор товару, назва, категорія, бренд, ціна, середній рейтинг, обчислений показник релевантності та тип рекомендаційного алгоритму.

Також до відповіді додається посилання на головне зображення товару, що дозволяє використовувати отримані дані для безпосереднього відображення рекомендацій у клієнтському інтерфейсі маркетплейсу.

Після формування об’єктів RecommendationItemResponse система повертає готовий список рекомендацій користувачу. Отриманий результат є фінальним етапом роботи контентно-орієнтованого алгоритму рекомендацій.

У результаті реалізації першого підходу до формування рекомендацій було створено контентно-орієнтований алгоритм рекомендацій, який формує персоналізовані рекомендації на основі характеристик товарів та історії взаємодій користувача із системою. Реалізований алгоритм використовує дані про категорії товарів, бренди, теги, характеристики товарів, ціновий діапазон та рейтинг товарів для визначення рівня їхньої схожості з інтересами користувача.

У процесі роботи алгоритму виконується отримання даних про товари та взаємодії користувача, формування профілю інтересів користувача, обчислення релевантності товарів-кандидатів, збереження результатів рекомендацій у базі даних та формування відповіді API для клієнтської частини системи.

Реалізований підхід дозволяє формувати персоналізовані рекомендації навіть у випадках, коли відсутня інформація про поведінку інших користувачів. Основною перевагою такого підходу є можливість аналізу характеристик товарів та побудови рекомендацій виключно на основі інтересів конкретного користувача.

Результатом реалізації став повноцінний серверний модуль контентно-орієнтованих рекомендацій, інтегрований у структуру маркетплейсу та готовий до подальшого тестування й порівняльного аналізу з іншими алгоритмами формування рекомендацій.

3.4 Реалізація підходу колаборативної фільтрації до формування рекомендацій

Другим реалізованим підходом до формування рекомендацій у системі став метод колаборативної фільтрації. Основна ідея цього підходу полягає в аналізі поведінки користувачів та пошуку схожих шаблонів взаємодії з товарами. На відміну від контентно-орієнтованого підходу, у цьому випадку рекомендації формуються не на основі характеристик товарів, а на основі дій інших користувачів із подібними інтересами.

У процесі роботи алгоритм аналізує історію взаємодій користувачів із товарами, формує векторні представлення поведінки користувачів, визначає ступінь схожості між користувачами та на основі цього прогнозує товари, які можуть бути цікавими поточному користувачеві. Для оцінювання схожості використовується метод cosine similarity, який дозволяє визначити рівень подібності між поведінковими векторами користувачів.

Після визначення схожих користувачів система обчислює оцінки релевантності товарів, виконує нормалізацію отриманих результатів, формує фінальний список рекомендацій та зберігає результати у базі даних. На завершальному етапі формується відповідь API для клієнтської частини системи.

Першим етапом роботи алгоритму є отримання історії взаємодій усіх користувачів із товарами. Приклад реалізації цього етапу наведено на лістингу 3.9.

Лістинг 3.9 – Отримання історії взаємодій усіх користувачів із товарами

```
var allInteractions = await _db.UserProductInteractions.ToListAsync();
if (!allInteractions.Any()) return new List<RecommendationItemResponse>();
```

У представленому коді виконується отримання всіх записів із таблиці UserProductInteractions за допомогою Entity Framework Core. Отримані дані

містять інформацію про взаємодії користувачів із товарами, зокрема перегляди, додавання товарів до кошика, покупки та інші дії.

Після отримання даних виконується перевірка на наявність взаємодій у системі. Якщо таблиця не містить жодного запису, алгоритм повертає порожній список рекомендацій, оскільки відсутні дані для аналізу поведінки користувачів та формування рекомендацій.

Наступним етапом роботи алгоритму є формування векторного представлення взаємодій користувачів із товарами. Приклад реалізації цього етапу наведено на лістингу 3.10.

Лістинг 3.10 – Формування векторного представлення взаємодій користувачів із товарами

```
var userVectors = new Dictionary<int, Dictionary<int, double>>();
        foreach (var interaction in allInteractions)
        {
            if (!userVectors.ContainsKey(interaction.UserId))
               userVectors[interaction.UserId] = new Dictionary<int, double>();

                                                    double       weight       =
RecommendationMathHelper.GetInteractionWeight(interaction.InteractionType);
             userVectors[interaction.UserId].TryGetValue(interaction.ProductId,
out var existing);
            if (weight > existing)
               userVectors[interaction.UserId][interaction.ProductId] = weight;
        }

        if (!userVectors.ContainsKey(request.UserId))
            return new List<RecommendationItemResponse>();

        var currentUserVector  = userVectors[request.UserId];
        var interactedProductIds = currentUserVector.Keys.ToHashSet();
```

У представленому коді для кожного користувача створюється окремий вектор взаємодій, який містить інформацію про товари, з якими користувач взаємодіяв, та вагу відповідної взаємодії. Вага визначається залежно від типу дії користувача, наприклад перегляд товару має меншу вагу, ніж покупка або додавання товару до кошика.

Під час обробки взаємодій система формує словник userVectors, у якому для кожного користувача зберігається набір товарів та відповідні числові

значення взаємодії. Якщо користувач виконував декілька дій із одним товаром, система залишає найбільшу вагу взаємодії.

Після завершення формування векторів виконується перевірка на наявність даних для поточного користувача. Далі формується набір товарів, із якими вже взаємодіяв користувач. Отримані дані використовуються на наступному етапі для пошуку схожих користувачів та формування рекомендацій.

Наступним етапом роботи алгоритму є пошук схожих користувачів та розрахунок оцінок релевантності товарів-кандидатів. Приклад реалізації цього етапу наведено на лістингу 3.11.

Лістинг 3.11 – Пошук схожих користувачів та розрахунок оцінок релевантності товарів-кандидатів

```
            var candidateScores = new Dictionary<int, double>();

            foreach (var (otherUserId, otherVector) in userVectors)
            {
                if (otherUserId == request.UserId) continue;
                                                    double       similarity       =
RecommendationMathHelper.CosineSimilarity(currentUserVector, otherVector);
                if (similarity <= 0) continue;

                foreach (var (productId, weight) in otherVector)
                {
                    if (interactedProductIds.Contains(productId)) continue;
                    candidateScores.TryGetValue(productId, out var score);
                    candidateScores[productId] = score + similarity * weight;
                }
            }

                                if      (!candidateScores.Any())      return      new
List<RecommendationItemResponse>();
```

У представленому коді система послідовно порівнює вектор поточного користувача з векторами інших користувачів. Для визначення ступеня схожості використовується метод cosine similarity, який дозволяє оцінити подібність поведінки користувачів на основі їх взаємодій із товарами.

Якщо отримане значення схожості є додатним, система аналізує товари, з якими взаємодіяв інший користувач. При цьому товари, з якими вже

взаємодіяв поточний користувач, виключаються з подальшого аналізу.

Для кожного товару-кандидата обчислюється оцінка релевантності, яка формується на основі добутку рівня схожості користувачів та ваги взаємодії з відповідним товаром. Таким чином, товари, популярні серед найбільш схожих користувачів, отримують вищі оцінки.

Після завершення обчислень система перевіряє наявність сформованих кандидатів на рекомендацію. Якщо релевантні товари не були знайдені, алгоритм повертає порожній список рекомендацій.

Після завершення розрахунку оцінок релевантності система переходить до етапу нормалізації та відбору найбільш релевантних товарів. Приклад реалізації цього етапу наведено на лістингу 3.12.

Лістинг 3.12 – Етап нормалізації та відбору найбільш релевантних товарів

```
                                        var            normalized           =
RecommendationMathHelper.NormalizeScores(candidateScores);
            var topIds     = normalized.OrderByDescending(kv => kv.Value)
                                       .Take(request.Count)
                                       .Select(kv => kv.Key)
                                       .ToList();
```

У представленому коді виконується нормалізація отриманих оцінок рекомендацій за допомогою методу NormalizeScores. Це дозволяє привести значення релевантності до єдиного діапазону та забезпечити коректне порівняння результатів між різними товарами.

Після нормалізації система сортує товари за спаданням значення релевантності та відбирає необхідну кількість рекомендацій відповідно до параметра request.Count. Далі формується список ідентифікаторів товарів, які мають найвищі показники релевантності та будуть використані у наступних етапах формування рекомендацій.

Після визначення найбільш релевантних товарів система переходить до отримання повної інформації про рекомендовані товари. Приклад реалізації

цього етапу наведено на лістингу 3.13.

Лістинг 3.13 – Отримання повної інформації про рекомендовані товари

```
var products = await _db.Products
      .Include(p => p.Category)
      .Include(p => p.Brand)
      .Include(p => p.Images)
      .Where(p => topIds.Contains(p.Id) && p.IsActive)
      .ToListAsync();
```

У представленому коді виконується звернення до таблиці Products для отримання товарів, ідентифікатори яких були відібрані на попередньому етапі. Разом із товарами додатково завантажуються пов’язані дані про категорії, бренди та зображення товарів за допомогою методів Include().

Також у запиті використовується перевірка поля IsActive, що дозволяє включати до списку рекомендацій лише активні товари, доступні для користувачів маркетплейсу.

Отримані дані використовуються на наступних етапах для збереження результатів рекомендацій та формування фінальної відповіді API.

Наступним етапом роботи алгоритму є збереження сформованих рекомендацій у базі даних. Приклад реалізації цього етапу наведено на лістингу 3.14.

Лістинг 3.14 – Збереження сформованих рекомендацій у базі даних

```
        var existingResults = await _db.RecommendationResults
             .Where(r => r.UserId == request.UserId && r.RecommendationType ==
RecommendationType.Collaborative)
            .ToListAsync();
        _db.RecommendationResults.RemoveRange(existingResults);

        var results = topIds
            .Where(id => products.Any(p => p.Id == id))
            .Select(id => new RecommendationResult
            {
                UserId             = request.UserId,
                ProductId          = id,
                RecommendationType = RecommendationType.Collaborative,
                Score              = Math.Round(normalized[id], 4),
                GeneratedAt        = DateTime.UtcNow
            }).ToList();
```

```
_db.RecommendationResults.AddRange(results);
await _db.SaveChangesAsync();
```

У представленому коді спочатку виконується пошук уже існуючих рекомендацій для поточного користувача, сформованих методом колаборативної фільтрації. Після цього попередні результати видаляються з таблиці RecommendationResults, що дозволяє уникнути дублювання або використання застарілих рекомендацій.

Далі для кожного рекомендованого товару створюється новий об’єкт RecommendationResult, який містить ідентифікатор користувача, ідентифікатор товару, тип рекомендаційного алгоритму, значення релевантності та дату генерації рекомендації.

Після формування списку результатів рекомендації додаються до бази даних через метод AddRange(), а потім зберігаються за допомогою SaveChangesAsync(). Це дозволяє використовувати сформовані рекомендації у подальшій роботі системи та забезпечує можливість їх повторного отримання без необхідності повторного виконання всіх обчислень.

Завершальним етапом роботи алгоритму є формування відповіді API для повернення списку рекомендацій клієнтській частині системи. Приклад реалізації цього етапу наведено на лістингу 3.15.

Лістинг 3.15 – Формування відповіді API для повернення списку рекомендацій

```
return topIds
    .Where(id => products.Any(p => p.Id == id))
    .Select(id =>
    {
        var product = products.First(p => p.Id == id);
        return new RecommendationItemResponse
        {
            ProductId          = product.Id,
            ProductName        = product.Name,
           CategoryName       = product.Category?.Name ?? string.Empty,
            BrandName          = product.Brand?.Name ?? string.Empty,
            Price              = product.Price,
            AverageRating      = product.AverageRating,
            Score              = Math.Round(normalized[id], 4),
                                                RecommendationType    =
```

```
nameof(RecommendationType.Collaborative),
                            ImageUrl            = product.Images.FirstOrDefault(i =>
i.IsMain)?.ImageUrl
                        };
                    }).ToList();
```

У представленому коді для кожного рекомендованого товару формується об’єкт RecommendationItemResponse, який містить основну інформацію про товар. До відповіді включаються ідентифікатор товару, назва, категорія, бренд, ціна, середній рейтинг, обчислене значення релевантності та тип використаного алгоритму рекомендацій.

Також до відповіді додається посилання на головне зображення товару, що дозволяє використовувати сформовані рекомендації безпосередньо у клієнтському інтерфейсі маркетплейсу.

Після формування об’єктів RecommendationItemResponse система повертає готовий список рекомендацій користувачу. Отриманий результат є фінальним етапом роботи алгоритму колаборативної фільтрації.

У результаті реалізації другого підходу до формування рекомендацій було створено алгоритм колаборативної фільтрації, який формує персоналізовані рекомендації на основі аналізу поведінки користувачів та пошуку схожих шаблонів взаємодії з товарами. Реалізований алгоритм використовує історію взаємодій користувачів із товарами для побудови векторних моделей поведінки та визначення ступеня схожості між користувачами.

У процесі роботи алгоритму виконується отримання даних про взаємодії користувачів, побудова поведінкових векторів, обчислення схожості між користувачами за допомогою cosine similarity, формування оцінок релевантності товарів, нормалізація результатів та побудова фінального списку рекомендацій.

Основною перевагою реалізованого підходу є можливість формування рекомендацій на основі поведінки інших користувачів, навіть якщо товари мають різні характеристики. Це дозволяє виявляти приховані закономірності у взаємодіях користувачів із товарами та рекомендувати товари, які можуть

бути цікавими користувачеві на основі колективного досвіду інших користувачів системи.

3.5 Реалізація гібридного підходу до формування рекомендацій

Третім реалізованим підходом до формування рекомендацій у системі став гібридний метод. Основна ідея цього підходу полягає у поєднанні результатів контентно-орієнтованого алгоритму та методу колаборативної фільтрації для формування більш точних і стабільних рекомендацій. Додатково у процесі формування рекомендацій враховуються контекстні дані користувача, що дозволяє підвищити релевантність сформованих результатів.

У процесі роботи алгоритм отримує результати двох попередньо реалізованих рекомендаційних моделей, виконує нормалізацію їхніх оцінок, аналізує контекстні характеристики поведінки користувача та формує єдиний інтегрований показник релевантності для кожного товару. Після цього система виконує ранжування товарів, зберігає результати рекомендацій у базі даних та формує фінальну відповідь API.

Першим етапом роботи алгоритму є отримання результатів роботи контентно-орієнтованого алгоритму та методу колаборативної фільтрації. Приклад реалізації цього етапу наведено на лістингу 3.16.

Лістинг 3.16 – Отримання результатів роботи контентно-орієнтованого алгоритму та методу колаборативної фільтрації

```
                // 1. Get CB and collaborative recommendations with larger count for
merging
                int innerCount = Math.Max(request.Count * 3, 50);

                var cbResults     = await _contentBased.ExecuteAsync(
                   new GetContentBasedRecommendationsRequest { UserId = request.UserId,
Count = innerCount });
                var collabResults = await _collaborative.ExecuteAsync(
                   new GetCollaborativeRecommendationsRequest { UserId = request.UserId,
Count = innerCount });
```

У представленому коді виконується виклик двох окремих сервісів рекомендацій – контентно-орієнтованого та колаборативного. Для кожного алгоритму формується запит із передачею ідентифікатора користувача та кількості необхідних рекомендацій.

Змінна innerCount використовується для збільшення кількості внутрішніх кандидатів на рекомендацію. Це дозволяє отримати ширший набір товарів від кожного алгоритму та підвищити якість подальшого об’єднання результатів у межах гібридного підходу.

Отримані результати двох алгоритмів надалі використовуються для формування єдиної системи оцінювання та побудови фінального списку рекомендацій.

Наступним етапом роботи алгоритму є підготовка результатів двох рекомендаційних моделей до подальшої обробки. Приклад реалізації цього етапу наведено на лістингу 3.17.

Лістинг 3.17 – Підготовка результатів двох рекомендаційних моделей до подальшої обробки

```
            var cbScores     = cbResults.ToDictionary(r => r.ProductId, r => r.Score);
            var collabScores = collabResults.ToDictionary(r => r.ProductId, r => r.Score);
```

У представленому коді результати контентно-орієнтованого алгоритму та алгоритму колаборативної фільтрації перетворюються у словники, де ключем виступає ідентифікатор товару, а значенням – обчислений показник релевантності.

Такий підхід дозволяє забезпечити швидкий доступ до оцінок рекомендацій для кожного товару та спрощує подальше об’єднання результатів двох алгоритмів. Отримані словники використовуються на наступних етапах для нормалізації оцінок та розрахунку фінального показника релевантності у межах гібридного методу рекомендацій.

Після формування словників оцінок система переходить до етапу нормалізації результатів двох алгоритмів рекомендацій. Приклад реалізації цього етапу наведено на лістингу 3.18.

Лістинг 3.18 – Етап нормалізації результатів двох алгоритмів рекомендацій

```
var cbNorm     = RecommendationMathHelper.NormalizeScores(cbScores);
var collabNorm = RecommendationMathHelper.NormalizeScores(collabScores);
```

У представленому коді виконується нормалізація оцінок контентно-орієнтованого алгоритму та алгоритму колаборативної фільтрації за допомогою методу NormalizeScores. Це дозволяє привести значення релевантності до єдиного діапазону та забезпечити коректне поєднання результатів двох різних підходів.

Нормалізація є важливим етапом гібридного алгоритму, оскільки різні рекомендаційні моделі можуть формувати оцінки у різних масштабах. Приведення оцінок до спільного діапазону дозволяє уникнути домінування одного з алгоритмів під час розрахунку фінального показника релевантності.

Наступним етапом роботи алгоритму є отримання контекстної інформації про інтереси користувача. Приклад реалізації цього етапу наведено на лістингу 3.19.

Лістинг 3.19 – Отримання контекстної інформації про інтереси користувача

```
var contextCategoryIds = await _db.UserProductInteractions
        .Where(i => i.UserId == request.UserId)
        .GroupBy(i => _db.Products.Where(p => p.Id == i.ProductId).Select(p
=> p.CategoryId).FirstOrDefault())
        .OrderByDescending(g => g.Count())
        .Take(2)
        .Select(g => g.Key)
        .ToListAsync();
```

У представленому коді виконується аналіз історії взаємодій користувача

з товарами для визначення найбільш популярних категорій товарів. Для цього взаємодії групуються за категоріями товарів, після чого виконується підрахунок кількості взаємодій у кожній категорії.

Далі категорії сортуються за кількістю взаємодій, а система відбирає декілька найбільш популярних категорій користувача. Отримані дані використовуються як контекстна інформація під час формування фінального показника релевантності у межах гібридного алгоритму рекомендацій.

Після отримання контекстної інформації система переходить до формування загального набору товарів-кандидатів для гібридного алгоритму. Приклад реалізації цього етапу наведено на лістингу 3.20.

Лістинг 3.20 – Формування загального набору товарів-кандидатів для гібридного алгоритму

```
var allProductIds = cbScores.Keys.Union(collabScores.Keys).ToHashSet();

// Load products for context scoring
var products = await _db.Products
    .Include(p => p.Category)
    .Include(p => p.Brand)
    .Include(p => p.Images)
    .Where(p => allProductIds.Contains(p.Id) && p.IsActive)
    .ToListAsync();

var productMap = products.ToDictionary(p => p.Id);
```

У представленому коді виконується об’єднання ідентифікаторів товарів, отриманих від контентно-орієнтованого алгоритму та алгоритму колаборативної фільтрації. Це дозволяє сформувати єдиний список товарів, які братимуть участь у подальшому розрахунку фінального показника релевантності.

Після цього система виконує завантаження інформації про відповідні товари з бази даних, включаючи категорії, бренди та зображення товарів. Додатково застосовується перевірка активності товарів, що дозволяє виключити недоступні позиції зі списку рекомендацій.

На завершальному етапі формується словник productMap, який

забезпечує швидкий доступ до інформації про товари під час подальшого розрахунку гібридних оцінок рекомендацій.

Наступним етапом роботи алгоритму є розрахунок фінального показника релевантності для кожного товару в межах гібридного підходу. Приклад реалізації цього етапу наведено на лістингу 3.21.

Лістинг 3.21 – Розрахунок фінального показника релевантності для кожного товару в межах гібридного підходу

```
            var hybridScores = new Dictionary<int, double>();
            foreach (var productId in allProductIds)
            {
                cbNorm.TryGetValue(productId, out var cbScore);
                collabNorm.TryGetValue(productId, out var collabScore);

                double contextScore = 0;
                      if (productMap.TryGetValue(productId, out var prod) &&
contextCategoryIds.Contains(prod.CategoryId))
                    contextScore = 1.0;

                 hybridScores[productId] = cbScore * 0.45 + collabScore * 0.45 +
contextScore * 0.10;
            }

            var topIds = hybridScores.OrderByDescending(kv => kv.Value)
                                     .Take(request.Count)
                                     .Select(kv => kv.Key)
                                     .ToList();
```

У представленому коді для кожного товару отримуються нормалізовані оцінки контентно-орієнтованого алгоритму та алгоритму колаборативної фільтрації. Додатково виконується перевірка належності товару до найбільш популярних категорій користувача, визначених на попередньому етапі. Якщо товар належить до однієї з таких категорій, йому нараховується додатковий контекстний бал.

Фінальний показник релевантності формується шляхом зваженого об’єднання результатів двох алгоритмів та контекстної оцінки. Основна вага розподіляється між контентно-орієнтованим і колаборативним підходами, тоді як контекстний компонент використовується як додатковий фактор уточнення рекомендацій.

Після завершення розрахунків товари сортуються за значенням фінального показника релевантності, а система відбирає необхідну кількість найбільш релевантних рекомендацій для користувача.

Після завершення розрахунку фінальних оцінок система переходить до етапу збереження результатів гібридних рекомендацій у базі даних. Приклад реалізації цього етапу наведено на лістингу 3.22.

Лістинг 3.22 – Етап збереження результатів гібридних рекомендацій у базі даних

```
            var existing = await _db.RecommendationResults
                 .Where(r => r.UserId == request.UserId && r.RecommendationType ==
RecommendationType.Hybrid)
                .ToListAsync();
            _db.RecommendationResults.RemoveRange(existing);

            var results = topIds
                .Where(id => productMap.ContainsKey(id))
                .Select(id => new RecommendationResult
                {
                    UserId             = request.UserId,
                    ProductId          = id,
                    RecommendationType = RecommendationType.Hybrid,
                    Score              = Math.Round(hybridScores[id], 4),
                    GeneratedAt        = DateTime.UtcNow
                }).ToList();

            _db.RecommendationResults.AddRange(results);
            await _db.SaveChangesAsync();
```

У представленому коді спочатку виконується пошук уже існуючих рекомендацій для поточного користувача, сформованих гібридним алгоритмом. Після цього попередні результати видаляються з таблиці RecommendationResults, що дозволяє уникнути дублювання або використання застарілих рекомендацій.

Далі для кожного рекомендованого товару створюється новий об’єкт RecommendationResult, який містить ідентифікатор користувача, ідентифікатор товару, тип алгоритму рекомендацій, значення фінальної релевантності та дату генерації рекомендації.

Після формування списку результатів рекомендації додаються до бази

даних через метод AddRange(), а потім зберігаються за допомогою SaveChangesAsync(). Це дозволяє використовувати результати гібридного алгоритму у подальшій роботі системи та забезпечує можливість їх повторного отримання без необхідності повторного виконання всіх обчислень.

Завершальним етапом роботи гібридного алгоритму є формування відповіді API для повернення сформованих рекомендацій клієнтській частині системи. Приклад реалізації цього етапу наведено на лістингу 3.23.

Лістинг 3.23 – Формування відповіді API для повернення сформованих рекомендацій

```
            return topIds
                .Where(id => productMap.ContainsKey(id))
                .Select(id =>
                {
                    var product = productMap[id];
                    return new RecommendationItemResponse
                    {
                        ProductId          = product.Id,
                        ProductName        = product.Name,
                       CategoryName       = product.Category?.Name ?? string.Empty,
                        BrandName          = product.Brand?.Name ?? string.Empty,
                        Price              = product.Price,
                        AverageRating      = product.AverageRating,
                        Score              = Math.Round(hybridScores[id], 4),
                        RecommendationType = nameof(RecommendationType.Hybrid),
                          ImageUrl            = product.Images.FirstOrDefault(i =>
i.IsMain)?.ImageUrl
                    };
                }).ToList();
```

У представленому коді для кожного рекомендованого товару формується об’єкт RecommendationItemResponse, який містить основну інформацію про товар. До відповіді включаються ідентифікатор товару, назва, категорія, бренд, ціна, середній рейтинг, фінальний показник релевантності та тип використаного алгоритму рекомендацій.

Також до відповіді додається посилання на головне зображення товару, що дозволяє використовувати сформовані рекомендації безпосередньо у клієнтському інтерфейсі маркетплейсу.

Після формування об’єктів RecommendationItemResponse система повертає готовий список рекомендацій користувачу. Отриманий результат є

фінальним етапом роботи гібридного алгоритму рекомендацій.

У результаті реалізації третього підходу до формування рекомендацій було створено гібридний алгоритм рекомендацій, який поєднує переваги контентно-орієнтованого підходу та методу колаборативної фільтрації. Реалізований алгоритм використовує результати двох попередніх моделей рекомендацій та додатково враховує контекстні характеристики поведінки користувача для формування фінального показника релевантності товарів.

У процесі роботи алгоритму виконується отримання результатів контентно-орієнтованих і колаборативних рекомендацій, нормалізація оцінок, аналіз найбільш популярних категорій користувача, об’єднання результатів двох моделей та розрахунок фінального гібридного показника релевантності.

Основною перевагою реалізованого підходу є можливість компенсувати недоліки окремих алгоритмів рекомендацій. Контентно-орієнтований підхід забезпечує аналіз характеристик товарів, тоді як колаборативна фільтрація дозволяє враховувати поведінку інших користувачів. Додаткове використання контекстних даних дозволяє підвищити актуальність сформованих рекомендацій відповідно до поточних інтересів користувача.

3.6 Інтеграція рекомендованих товарів у функціонал маркетплейсу

Після реалізації трьох основних підходів до формування рекомендацій наступним етапом стала інтеграція рекомендаційної підсистеми у загальний функціонал маркетплейсу. Основною метою цього етапу є забезпечення можливості отримання персоналізованих рекомендацій через API маркетплейсу та подальше використання сформованих результатів у клієнтській частині системи.

Для інтеграції рекомендацій у серверну частину маркетплейсу було реалізовано окремий модуль Recommendations, який містить набір операцій

для роботи з усіма трьома рекомендаційними алгоритмами. До складу модуля входять операції формування контентно-орієнтованих рекомендацій, рекомендацій методом колаборативної фільтрації та гібридного підходу.

Кожен із реалізованих алгоритмів представлений окремою операцією, що дозволяє незалежно викликати необхідний метод рекомендацій через API системи. Такий підхід забезпечує можливість окремого тестування кожного алгоритму, проведення порівняльного аналізу результатів та оцінювання ефективності різних підходів до формування рекомендацій.

Крім трьох основних алгоритмів, у системі також реалізовано дві допоміжні операції. Перша операція відповідає за одночасний запуск усіх рекомендаційних алгоритмів та автоматичне формування повного набору рекомендацій для користувача. Використання такого механізму значно спрощує проведення експериментального дослідження, оскільки дозволяє одним запитом отримати результати роботи всіх реалізованих підходів.

Друга допоміжна операція призначена для отримання вже сформованих рекомендацій із бази даних. Вона використовується для подальшого відображення результатів рекомендацій у системі та забезпечує можливість повторного отримання рекомендацій без необхідності повторного виконання алгоритмів.

Список реалізованих операцій модуля рекомендацій наведено на рисунку 3.2. На рисунку представлено структуру директорій та операцій, реалізованих у серверній частині маркетплейсу для підтримки рекомендаційної підсистеми.

На лістингу 3.24 наведено реалізацію контролера рекомендаційної підсистеми, який забезпечує доступ до всіх реалізованих алгоритмів формування рекомендацій через HTTP API. Контролер виступає точкою взаємодії між клієнтською частиною маркетплейсу та серверними операціями формування рекомендацій.

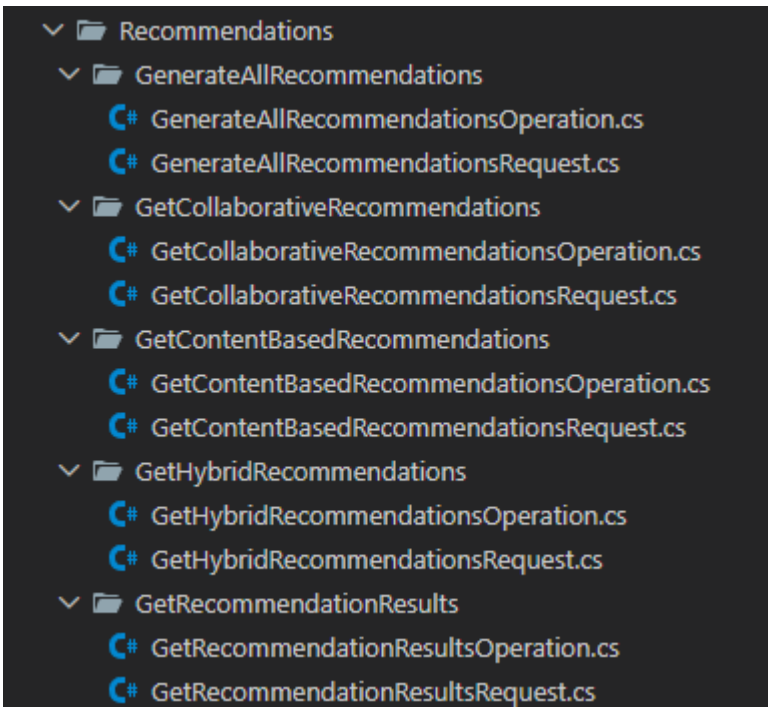

Рисунок 3.2 – Структура директорій та операцій, що була реалізована для інтеграції рекомендацій в систему

Лістинг 3.24 – Реалізація контролера рекомендаційної підсистеми

```
        [HttpGet(«content-based/{userId:int}»)]
          public async Task<IActionResult> GetContentBased(
              int userId,
              [FromQuery] int count = 10,
                 [FromServices] GetContentBasedRecommendationsOperation operation =
null!)
          {
                          var    result    =    await    operation.ExecuteAsync(new
GetContentBasedRecommendationsRequest { UserId = userId, Count = count });
              return Ok(result);
          }

          [HttpGet(«collaborative/{userId:int}»)]
          public async Task<IActionResult> GetCollaborative(
              int userId,
              [FromQuery] int count = 10,
                [FromServices] GetCollaborativeRecommendationsOperation operation =
null!)
          {
                          var    result    =    await    operation.ExecuteAsync(new
GetCollaborativeRecommendationsRequest { UserId = userId, Count = count });
              return Ok(result);
          }

          [HttpGet(«hybrid/{userId:int}»)]
          public async Task<IActionResult> GetHybrid(
              int userId,
              [FromQuery] int count = 10,
              [FromServices] GetHybridRecommendationsOperation operation = null!)
          {
```

```
                        var result = await operation.ExecuteAsync(new
GetHybridRecommendationsRequest { UserId = userId, Count = count });
            return Ok(result);
        }

        [HttpGet(«results/{userId:int}»)]
        public async Task<IActionResult> GetResults(
            int userId,
            [FromQuery] RecommendationType? type = null,
            [FromServices] GetRecommendationResultsOperation operation = null!)
        {
                        var result = await operation.ExecuteAsync(new
GetRecommendationResultsRequest { UserId = userId, Type = type });
            return Ok(result);
        }

        [HttpPost(«generate-all/{userId:int}»)]
        public async Task<IActionResult> GenerateAll(
            int userId,
            [FromServices] GenerateAllRecommendationsOperation operation)
        {
                        var result = await operation.ExecuteAsync(new
GenerateAllRecommendationsRequest { UserId = userId });
            return Ok(result);
        }
```

У представленому коді реалізовано окремі endpoint-и для кожного з трьох алгоритмів рекомендацій. Метод GetContentBased() відповідає за виклик контентно-орієнтованого алгоритму, метод GetCollaborative() – за формування рекомендацій методом колаборативної фільтрації, а метод GetHybrid() – за роботу гібридного підходу.

Кожен метод приймає ідентифікатор користувача та кількість рекомендацій, які необхідно сформувати. Далі виконується створення відповідного request-об’єкта та передача його до operation-класу через механізм Dependency Injection із використанням атрибута [FromServices].

Також у контролері реалізовано допоміжний endpoint GetResults(), який дозволяє отримувати вже сформовані результати рекомендацій із бази даних. Додатково цей метод підтримує фільтрацію за типом рекомендаційного алгоритму, що дозволяє окремо отримувати результати контентно-орієнтованого, колаборативного або гібридного підходу.

Метод GenerateAll() використовується для одночасного запуску всіх реалізованих алгоритмів рекомендацій. У процесі його виконання система формує рекомендації для користувача всіма трьома підходами та зберігає

результати у базі даних. Реалізація такого endpoint-а значно спрощує проведення експериментів та подальше порівняння ефективності різних рекомендаційних моделей.

Усі методи контролера реалізовані з використанням асинхронного підходу async/await, що дозволяє оптимізувати роботу серверної частини та зменшити блокування потоків під час виконання запитів до бази даних.

У результаті інтеграції рекомендаційної підсистеми в структуру маркетплейсу було реалізовано централізований механізм доступу до всіх алгоритмів формування рекомендацій через REST API. У систему були інтегровані контентно-орієнтований, колаборативний та гібридний підходи, а також додаткові операції для автоматичного запуску всіх алгоритмів і отримання збережених результатів рекомендацій.

Реалізований механізм забезпечує можливість окремого тестування кожного алгоритму, проведення порівняльного аналізу їхньої ефективності та подальшого використання сформованих рекомендацій у функціоналі маркетплейсу. Інтеграція рекомендаційної підсистеми у серверну архітектуру .NET забезпечила модульність, розширюваність та зручність подальшого розвитку системи.

4 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ РЕАЛІЗОВАНИХ ПІДХОДІВ

4.1 Організація експериментального дослідження

Організація експериментального дослідження передбачає перевірку ефективності трьох реалізованих підходів до формування рекомендацій у межах розробленої серверної частини маркетплейсу. Для дослідження використовуються контентно-орієнтований алгоритм, алгоритм колаборативної фільтрації та гібридний метод, що поєднує результати двох попередніх підходів із додатковим урахуванням контекстної інформації.

Експеримент проводиться на основі підготовленого тестового набору даних, яким була наповнена база даних маркетплейсу. Цей набір містить інформацію про користувачів, товари, категорії, бренди, характеристики товарів, а також історію взаємодій користувачів із товарами. Саме ці взаємодії використовуються як основа для визначення інтересів користувачів та подальшого оцінювання якості сформованих рекомендацій.

Для проведення експерименту обирається група користувачів, для яких послідовно запускаються всі три алгоритми рекомендацій. Кожен алгоритм формує список рекомендованих товарів для одного й того самого користувача. Це дозволяє порівнювати результати в однакових умовах, оскільки всі підходи працюють з однаковою базою даних та однаковою історією взаємодій користувачів.

Після формування рекомендацій результати роботи кожного алгоритму зберігаються у таблиці RecommendationResults. Для кожного рекомендованого товару фіксується ідентифікатор користувача, ідентифікатор товару, тип алгоритму рекомендацій, обчислений показник релевантності та дата генерації результату. Це дозволяє повторно отримувати результати рекомендацій та використовувати їх для подальшого аналізу.

У процесі експерименту визначається, наскільки добре кожен алгоритм здатний формувати релевантні рекомендації для користувачів. Для цього результати рекомендацій порівнюються з фактичною історією взаємодій користувачів із товарами. Релевантними вважаються ті товари, які відповідають інтересам користувача за категорією, брендом, тегами, характеристиками або поведінковими патернами взаємодій.

Оцінювання якості рекомендацій виконується за допомогою п’яти метрик: Precision, Recall, F1-score, NDCG та Diversity. Метрика Precision дозволяє визначити частку релевантних товарів серед усіх рекомендованих. Recall показує, яку частину потенційно релевантних товарів зміг знайти алгоритм. F1-score використовується для отримання узагальненої оцінки, що враховує баланс між точністю та повнотою рекомендацій.

Для оцінювання якості ранжування використовується метрика NDCG, яка враховує позицію релевантних товарів у списку рекомендацій. Це важливо, оскільки у реальних системах користувач найчастіше взаємодіє саме з першими елементами списку. Метрика Diversity використовується для оцінювання різноманітності рекомендацій та дозволяє визначити, чи не пропонує алгоритм надто однотипні товари.

Результатом експериментального дослідження має стати порівняльна оцінка ефективності трьох реалізованих підходів до формування рекомендацій. На основі отриманих значень метрик буде визначено, який алгоритм забезпечує найкращу точність, повноту, якість ранжування та різноманітність рекомендацій у межах розробленої системи маркетплейсу.

На лістингу 4.1 наведено основний фрагмент коду запуску експериментального дослідження, у межах якого виконується послідовне тестування трьох реалізованих підходів до формування рекомендацій. Даний код відповідає за вибір користувачів, підготовку навчальної та тестової частин даних, запуск рекомендаційних алгоритмів та формування результатів для подальшого збереження у CSV-файл.

Лістинг 4.1 – Код запуску експерименту

```
        var userIds = await _db.UserProductInteractions
                    .Select(i => i.UserId)
                    .Distinct()
                    .ToListAsync();

                foreach (int userId in userIds)
                {
                    // 2. Унікальні товари впорядковані за першою взаємодією
                    var uniqueProducts = await _db.UserProductInteractions
                        .Where(i => i.UserId == userId)
                        .GroupBy(i => i.ProductId)
                         .Select(g => new { ProductId = g.Key, FirstAt = g.Min(i =>
i.CreatedAt) })
                        .OrderBy(x => x.FirstAt)
                        .Select(x => x.ProductId)
                        .ToListAsync();

                      // 3. Пропускаємо юзерів з недостатньою кількістю унікальних
товарів
                    if (uniqueProducts.Count < request.MinInteractions)
                        continue;

                    // 4. Train/Test split 80/20
                     int trainSize        = (int)Math.Ceiling(uniqueProducts.Count *
0.8);
                    var trainProductIds = uniqueProducts.Take(trainSize).ToHashSet();
                    var testProductIds  = uniqueProducts.Skip(trainSize).ToHashSet();

                    if (testProductIds.Count == 0)
                        continue;

                   // 5. Отримуємо рекомендації (будуються ТІЛЬКИ на train-взаємодіях)
                     // Беремо більший пул (K*4) щоб тест-товар мав більше шансів
потрапити до топ-K
                    int recommendCount = Math.Max(request.K * 4, 40);

                       var cbIds      = await _algorithms.ContentBasedAsync(userId,
trainProductIds, recommendCount);
                       var collabIds = await _algorithms.CollaborativeAsync(userId,
trainProductIds, recommendCount);
                            var hybridIds = await _algorithms.HybridAsync(userId,
trainProductIds, recommendCount);

                    // 6. Категорії рекомендованих товарів для метрики Diversity
                    var allIds = cbIds.Union(collabIds).Union(hybridIds).ToList();
                    var productCategories = await _db.Products
                        .Where(p => allIds.Contains(p.Id))
                        .Select(p => new { p.Id, p.CategoryId })
                        .ToDictionaryAsync(p => p.Id, p => p.CategoryId);

                    // 7. Обчислюємо метрики
                                                    rows.Add(ComputeRow(userId,
«ContentBased»,    cbIds,          testProductIds,   productCategories,   request.K,
startedAt));
                          rows.Add(ComputeRow(userId,  «Collaborative»,  collabIds,
testProductIds, productCategories, request.K, startedAt));
                        rows.Add(ComputeRow(userId,  «Hybrid»,           hybridIds,
testProductIds, productCategories, request.K, startedAt));
                }
```

Першим етапом виконується отримання списку користувачів, для яких у системі наявна історія взаємодій із товарами. Для цього з таблиці UserProductInteractions вибираються унікальні ідентифікатори користувачів. Такий підхід дозволяє проводити експеримент лише для тих користувачів, які вже мають певну активність у маркетплейсі, оскільки саме історія взаємодій є основою для формування та перевірки рекомендацій.

Далі для кожного користувача окремо формується список унікальних товарів, з якими він взаємодіяв. У коді взаємодії групуються за ProductId, після чого для кожного товару визначається дата першої взаємодії. Потім товари впорядковуються за цією датою. Такий підхід дозволяє відтворити послідовність інтересів користувача в часі та уникнути дублювання одного й того самого товару в експериментальній вибірці.

Наступним кроком виконується перевірка кількості унікальних товарів користувача. Якщо користувач взаємодіяв із меншою кількістю товарів, ніж визначено параметром MinInteractions, він пропускається. Це необхідно для того, щоб експеримент проводився лише для користувачів, які мають достатню кількість даних для поділу на навчальну та тестову частини. За недостатньої кількості взаємодій оцінювання якості рекомендацій може бути некоректним.

Після цього виконується поділ даних користувача на навчальну та тестову частини у співвідношенні 80/20. Перші 80% товарів, упорядкованих за часом першої взаємодії, використовуються як навчальна частина, а решта 20% – як тестова. Навчальна частина використовується для формування рекомендацій, а тестова – для перевірки того, чи зміг алгоритм передбачити товари, які користувач обрав пізніше.

У коді також передбачена перевірка на наявність тестових товарів. Якщо після поділу тестова множина є порожньою, користувач пропускається. Це дозволяє уникнути ситуації, коли неможливо обчислити метрики якості рекомендацій через відсутність еталонних товарів для порівняння.

Далі визначається кількість рекомендацій, які будуть сформовані

кожним алгоритмом. Для цього використовується змінна recommendCount, що обчислюється як максимальне значення між K*4 та 40. Такий підхід дозволяє отримати ширший пул рекомендованих товарів, з якого надалі оцінюються перші K позицій. Це підвищує ймовірність потрапляння релевантного товару до сформованого списку рекомендацій.

Після підготовки даних для користувача виконується запуск трьох рекомендаційних алгоритмів: контентно-орієнтованого, колаборативного та гібридного. Кожен алгоритм отримує ідентифікатор користувача, навчальну множину товарів та кількість рекомендацій, які необхідно сформувати. Важливо, що рекомендації будуються лише на основі train-даних, тобто тестові товари не використовуються під час формування рекомендацій. Це забезпечує коректність експерименту.

На наступному етапі формується спільний набір ідентифікаторів товарів, рекомендованих усіма трьома алгоритмами. Для цих товарів із бази даних отримуються категорії, які надалі використовуються для обчислення метрики Diversity. Ця метрика дозволяє оцінити різноманітність рекомендацій, тобто визначити, наскільки товари у списку рекомендацій відрізняються між собою за категоріями.

Завершальним етапом є обчислення метрик якості рекомендацій для кожного алгоритму. Для цього викликається метод ComputeRow, який отримує ідентифікатор користувача, тип алгоритму, список рекомендованих товарів, тестову множину товарів, категорії товарів, значення K та час запуску експерименту. У результаті для кожного користувача формується три рядки результатів: окремо для контентно-орієнтованого підходу, колаборативної фільтрації та гібридного методу.

4.2 Проведення експериментів для кожного з реалізованих підходів

Після реалізації механізму запуску експериментального дослідження наступним етапом стало безпосереднє проведення експериментів для кожного

з реалізованих підходів формування рекомендацій. Експериментальне дослідження проводиться окремо для контентно-орієнтованого алгоритму, алгоритму колаборативної фільтрації та гібридного методу рекомендацій.

У процесі проведення експерименту система автоматично виконує формування рекомендацій для користувачів, обчислює значення метрик якості рекомендацій та формує структурований набір результатів. Для кожного алгоритму окремо визначаються значення Precision, Recall, F1-score, NDCG та Diversity, що дозволяє оцінити точність, повноту, якість ранжування та різноманітність сформованих рекомендацій.

Результати експериментального дослідження автоматично зберігаються у форматі CSV. Використання такого формату дозволяє зручно експортувати результати, відкривати їх у табличних редакторах та виконувати подальший аналіз отриманих даних. Кожен рядок CSV-файлу містить результати оцінювання окремого алгоритму для конкретного користувача.

У сформованому файлі зберігається інформація про ідентифікатор користувача, тип рекомендаційного алгоритму, значення всіх метрик оцінювання, кількість рекомендованих товарів, кількість релевантних товарів та дату проведення експерименту. Такий підхід дозволяє проводити подальше порівняння ефективності реалізованих підходів формування рекомендацій.

Приклад сформованого CSV-файлу з результатами проведення експериментального дослідження наведено на лістингу 4.2.

Лістинг 4.2 – Результати експерименту

```
UserId,RecommendationType,Precision,Recall,F1Score,Ndcg,Diversity,RecommendationCount
,RelevantCount,GeneratedAt
2,ContentBased,0.1000,0.3333,0.1538,0.1815,0.4000,40,3,2026-05-11 18:12:04
2,Collaborative,0.0000,0.0000,0.0000,0.0000,0.7000,40,3,2026-05-11 18:12:04
2,Hybrid,0.1000,0.3333,0.1538,0.1672,0.5000,40,3,2026-05-11 18:12:04
3,ContentBased,0.0000,0.0000,0.0000,0.0000,0.4000,40,3,2026-05-11 18:12:04
3,Collaborative,0.1000,0.3333,0.1538,0.1564,0.6000,40,3,2026-05-11 18:12:04
3,Hybrid,0.1000,0.3333,0.1538,0.1357,0.4000,40,3,2026-05-11 18:12:04
4,ContentBased,0.2000,0.6667,0.3077,0.6257,0.3000,40,3,2026-05-11 18:12:04
4,Collaborative,0.2000,0.6667,0.3077,0.6364,0.6000,40,3,2026-05-11 18:12:04
4,Hybrid,0.2000,0.6667,0.3077,0.4776,0.5000,40,3,2026-05-11 18:12:04
5,ContentBased,0.2000,0.6667,0.3077,0.3836,0.3000,40,3,2026-05-11 18:12:04
5,Collaborative,0.2000,0.6667,0.3077,0.4632,0.5000,40,3,2026-05-11 18:12:04
5,Hybrid,0.2000,0.6667,0.3077,0.3236,0.4000,40,3,2026-05-11 18:12:04
```

```
6,ContentBased,0.1000,0.3333,0.1538,0.2021,0.3000,40,3,2026-05-11 18:12:04
6,Collaborative,0.1000,0.3333,0.1538,0.2346,0.6000,40,3,2026-05-11 18:12:04
6,Hybrid,0.1000,0.3333,0.1538,0.2021,0.4000,40,3,2026-05-11 18:12:04
7,ContentBased,0.1000,1.0000,0.1818,0.6309,0.4000,40,1,2026-05-11 18:12:04
7,Collaborative,0.0000,0.0000,0.0000,0.0000,0.6000,31,1,2026-05-11 18:12:04
7,Hybrid,0.0000,0.0000,0.0000,0.0000,0.6000,40,1,2026-05-11 18:12:04
8,ContentBased,0.1000,1.0000,0.1818,0.6309,0.3000,40,1,2026-05-11 18:12:04
8,Collaborative,0.0000,0.0000,0.0000,0.0000,0.7000,32,1,2026-05-11 18:12:04
8,Hybrid,0.1000,1.0000,0.1818,0.3155,0.5000,40,1,2026-05-11 18:12:04
9,ContentBased,0.1000,1.0000,0.1818,0.3010,0.3000,40,1,2026-05-11 18:12:04
9,Collaborative,0.1000,1.0000,0.1818,0.6309,0.5000,26,1,2026-05-11 18:12:04
9,Hybrid,0.1000,1.0000,0.1818,0.6309,0.4000,40,1,2026-05-11 18:12:04
10,ContentBased,0.1000,1.0000,0.1818,0.3869,0.4000,40,1,2026-05-11 18:12:04
10,Collaborative,0.1000,1.0000,0.1818,0.3155,0.5000,33,1,2026-05-11 18:12:04
10,Hybrid,0.1000,1.0000,0.1818,0.3869,0.4000,40,1,2026-05-11 18:12:04
11,ContentBased,0.1000,1.0000,0.1818,0.3869,0.4000,40,1,2026-05-11 18:12:04
11,Collaborative,0.1000,1.0000,0.1818,1.0000,0.6000,33,1,2026-05-11 18:12:04
11,Hybrid,0.1000,1.0000,0.1818,1.0000,0.4000,40,1,2026-05-11 18:12:04
12,ContentBased,0.1000,1.0000,0.1818,0.6309,0.4000,40,1,2026-05-11 18:12:04
12,Collaborative,0.1000,1.0000,0.1818,0.4307,0.6000,32,1,2026-05-11 18:12:04
12,Hybrid,0.1000,1.0000,0.1818,0.6309,0.5000,40,1,2026-05-11 18:12:04
13,ContentBased,0.1000,1.0000,0.1818,1.0000,0.4000,40,1,2026-05-11 18:12:04
13,Collaborative,0.1000,1.0000,0.1818,1.0000,0.6000,32,1,2026-05-11 18:12:04
13,Hybrid,0.1000,1.0000,0.1818,1.0000,0.5000,40,1,2026-05-11 18:12:04
14,ContentBased,0.1000,1.0000,0.1818,1.0000,0.5000,40,1,2026-05-11 18:12:04
14,Collaborative,0.1000,1.0000,0.1818,0.5000,0.6000,27,1,2026-05-11 18:12:04
14,Hybrid,0.1000,1.0000,0.1818,0.6309,0.5000,40,1,2026-05-11 18:12:04
15,ContentBased,0.1000,1.0000,0.1818,1.0000,0.4000,40,1,2026-05-11 18:12:04
15,Collaborative,0.1000,1.0000,0.1818,0.4307,0.6000,33,1,2026-05-11 18:12:04
15,Hybrid,0.1000,1.0000,0.1818,0.5000,0.5000,40,1,2026-05-11 18:12:04
16,ContentBased,0.1000,1.0000,0.1818,0.6309,0.4000,40,1,2026-05-11 18:12:04
16,Collaborative,0.1000,1.0000,0.1818,0.6309,0.5000,29,1,2026-05-11 18:12:04
16,Hybrid,0.1000,1.0000,0.1818,0.6309,0.5000,40,1,2026-05-11 18:12:04
17,ContentBased,0.2000,1.0000,0.3333,0.9197,0.4000,40,2,2026-05-11 18:12:04
17,Collaborative,0.2000,1.0000,0.3333,0.9197,0.6000,32,2,2026-05-11 18:12:04
17,Hybrid,0.2000,1.0000,0.3333,1.0000,0.5000,40,2,2026-05-11 18:12:04
18,ContentBased,0.1000,1.0000,0.1818,0.6309,0.5000,40,1,2026-05-11 18:12:04
18,Collaborative,0.1000,1.0000,0.1818,0.5000,0.7000,36,1,2026-05-11 18:12:04
18,Hybrid,0.1000,1.0000,0.1818,0.6309,0.7000,40,1,2026-05-11 18:12:04
19,ContentBased,0.1000,1.0000,0.1818,0.5000,0.6000,40,1,2026-05-11 18:12:04
19,Collaborative,0.1000,1.0000,0.1818,0.5000,0.6000,36,1,2026-05-11 18:12:04
19,Hybrid,0.1000,1.0000,0.1818,0.6309,0.6000,40,1,2026-05-11 18:12:04
20,ContentBased,0.1000,1.0000,0.1818,1.0000,0.5000,40,1,2026-05-11 18:12:04
20,Collaborative,0.1000,1.0000,0.1818,0.5000,0.7000,36,1,2026-05-11 18:12:04
20,Hybrid,0.1000,1.0000,0.1818,0.6309,0.6000,40,1,2026-05-11 18:12:04
22,ContentBased,0.0000,0.0000,0.0000,0.0000,0.3000,40,2,2026-05-11 18:12:04
22,Collaborative,0.1000,0.5000,0.1667,0.2372,0.6000,40,2,2026-05-11 18:12:04
22,Hybrid,0.1000,0.5000,0.1667,0.1772,0.5000,40,2,2026-05-11 18:12:04
23,ContentBased,0.0000,0.0000,0.0000,0.0000,0.4000,40,2,2026-05-11 18:12:04
23,Collaborative,0.1000,0.5000,0.1667,0.3869,0.6000,40,2,2026-05-11 18:12:04
23,Hybrid,0.1000,0.5000,0.1667,0.1846,0.4000,40,2,2026-05-11 18:12:04
24,ContentBased,0.1000,0.5000,0.1667,0.2372,0.3000,40,2,2026-05-11 18:12:04
24,Collaborative,0.0000,0.0000,0.0000,0.0000,0.4000,40,2,2026-05-11 18:12:04
24,Hybrid,0.0000,0.0000,0.0000,0.0000,0.4000,40,2,2026-05-11 18:12:04
25,ContentBased,0.0000,0.0000,0.0000,0.0000,0.4000,40,3,2026-05-11 18:12:04
25,Collaborative,0.1000,0.3333,0.1538,0.1672,0.6000,40,3,2026-05-11 18:12:04
25,Hybrid,0.1000,0.3333,0.1538,0.1815,0.5000,40,3,2026-05-11 18:12:04
```

Результати проведення експериментального дослідження підтвердили

коректність роботи реалізованого механізму оцінювання рекомендаційних алгоритмів. У процесі виконання експерименту система успішно виконала формування рекомендацій для всіх трьох реалізованих підходів, обчислення метрик якості рекомендацій та автоматичне збереження отриманих результатів у форматі CSV.

4.3 Аналіз отриманих результатів

Таблиця 4.1 демонструє результати експериментального дослідження ефективності трьох реалізованих підходів до формування рекомендацій у межах розробленої системи маркетплейсу. Для кожного користувача окремо були отримані значення метрик Precision, Recall, F1-score, NDCG та Diversity для контентно-орієнтованого алгоритму, колаборативної фільтрації та гібридного підходу.

Таблиця 4.1 – Результати експерименту

UserId	RecommendationType	Precision	Recall	F1Score	Ndcg	Diversity	RecommendationCount	RelevantCount
2	ContentBased	0.1	0.33	0.15	0.18	0.4	40	3
2	Collaborative	0	0	0	0	0.7	40	3
2	Hybrid	0.1	0.33	0.15	0.17	0.5	40	3
3	ContentBased	0	0	0	0	0.4	40	3
3	Collaborative	0.1	0.33	0.15	0.16	0.6	40	3
3	Hybrid	0.1	0.33	0.15	0.14	0.4	40	3
4	ContentBased	0.2	0.67	0.31	0.63	0.3	40	3
4	Collaborative	0.2	0.67	0.31	0.64	0.6	40	3
4	Hybrid	0.2	0.67	0.31	0.48	0.5	40	3
5	ContentBased	0.2	0.67	0.31	0.38	0.3	40	3
5	Collaborative	0.2	0.67	0.31	0.46	0.5	40	3
5	Hybrid	0.2	0.67	0.31	0.32	0.4	40	3
6	ContentBased	0.1	0.33	0.15	0.2	0.3	40	3
6	Collaborative	0.1	0.33	0.15	0.23	0.6	40	3
6	Hybrid	0.1	0.33	0.15	0.2	0.4	40	3

Продовження таблиці 4.1

UserId	RecommendationType	Precision	Recall	F1Score	Ndcg	Diversity	RecommendationCount	RelevantCount
7	ContentBased	0.1	1	0.18	0.63	0.4	40	1
7	Collaborative	0	0	0	0	0.6	31	1
7	Hybrid	0	0	0	0	0.6	40	1
8	ContentBased	0.1	1	0.18	0.63	0.3	40	1
8	Collaborative	0	0	0	0	0.7	32	1
8	Hybrid	0.1	1	0.18	0.32	0.5	40	1
9	ContentBased	0.1	1	0.18	0.3	0.3	40	1
9	Collaborative	0.1	1	0.18	0.63	0.5	26	1
9	Hybrid	0.1	1	0.18	0.63	0.4	40	1
10	ContentBased	0.1	1	0.18	0.39	0.4	40	1
10	Collaborative	0.1	1	0.18	0.32	0.5	33	1
10	Hybrid	0.1	1	0.18	0.39	0.4	40	1
11	ContentBased	0.1	1	0.18	0.39	0.4	40	1
11	Collaborative	0.1	1	0.18	1	0.6	33	1
11	Hybrid	0.1	1	0.18	1	0.4	40	1
12	ContentBased	0.1	1	0.18	0.63	0.4	40	1
12	Collaborative	0.1	1	0.18	0.43	0.6	32	1
12	Hybrid	0.1	1	0.18	0.63	0.5	40	1
13	ContentBased	0.1	1	0.18	1	0.4	40	1
13	Collaborative	0.1	1	0.18	1	0.6	32	1
13	Hybrid	0.1	1	0.18	1	0.5	40	1
14	ContentBased	0.1	1	0.18	1	0.5	40	1
14	Collaborative	0.1	1	0.18	0.5	0.6	27	1
14	Hybrid	0.1	1	0.18	0.63	0.5	40	1
15	ContentBased	0.1	1	0.18	1	0.4	40	1
15	Collaborative	0.1	1	0.18	0.43	0.6	33	1
15	Hybrid	0.1	1	0.18	0.5	0.5	40	1
16	ContentBased	0.1	1	0.18	0.63	0.4	40	1
16	Collaborative	0.1	1	0.18	0.63	0.5	29	1
16	Hybrid	0.1	1	0.18	0.63	0.5	40	1
17	ContentBased	0.2	1	0.33	0.92	0.4	40	2
17	Collaborative	0.2	1	0.33	0.92	0.6	32	2
17	Hybrid	0.2	1	0.33	1	0.5	40	2
18	ContentBased	0.1	1	0.18	0.63	0.5	40	1
18	Collaborative	0.1	1	0.18	0.5	0.7	36	1
18	Hybrid	0.1	1	0.18	0.63	0.7	40	1
19	ContentBased	0.1	1	0.18	0.5	0.6	40	1

Продовження таблиці 4.1

UserId	RecommendationType	Precision	Recall	F1Score	Ndcg	Diversity	RecommendationCount	RelevantCount
19	Collaborative	0.1	1	0.18	0.5	0.6	36	1
19	Hybrid	0.1	1	0.18	0.63	0.6	40	1
20	ContentBased	0.1	1	0.18	1	0.5	40	1
20	Collaborative	0.1	1	0.18	0.5	0.7	36	1
20	Hybrid	0.1	1	0.18	0.63	0.6	40	1
22	ContentBased	0	0	0	0	0.3	40	2
22	Collaborative	0.1	0.5	0.17	0.24	0.6	40	2
22	Hybrid	0.1	0.5	0.17	0.18	0.5	40	2
23	ContentBased	0	0	0	0	0.4	40	2
23	Collaborative	0.1	0.5	0.17	0.39	0.6	40	2
23	Hybrid	0.1	0.5	0.17	0.18	0.4	40	2
24	ContentBased	0.1	0.5	0.17	0.24	0.3	40	2
24	Collaborative	0	0	0	0	0.4	40	2
24	Hybrid	0	0	0	0	0.4	40	2
25	ContentBased	0	0	0	0	0.4	40	3
25	Collaborative	0.1	0.33	0.15	0.17	0.6	40	3
25	Hybrid	0.1	0.33	0.15	0.18	0.5	40	3
2	ContentBased	0.1	0.33	0.15	0.18	0.4	40	3
2	Collaborative	0	0	0	0	0.7	40	3
2	Hybrid	0.1	0.33	0.15	0.17	0.5	40	3

Отримані результати показують, що всі реалізовані алгоритми здатні формувати рекомендації, які частково або повністю відповідають інтересам користувачів. Значення метрик Precision та Recall свідчать про наявність релевантних товарів серед сформованих рекомендацій. При цьому для окремих користувачів спостерігаються як високі, так і нульові значення окремих метрик, що є характерним для рекомендаційних систем, побудованих на основі поведінкових даних.

Контентно-орієнтований підхід продемонстрував стабільну роботу у випадках, коли користувач мав виражену історію взаємодій із товарами певних категорій або брендів. Алгоритм успішно формував рекомендації на основі характеристик товарів, з якими користувач взаємодіяв раніше. При цьому в окремих випадках спостерігалося зниження Precision через рекомендацію

товарів зі схожими характеристиками, які не входили до тестового набору користувача.

Алгоритм колаборативної фільтрації показав більш варіативні результати. Для частини користувачів цей підхід забезпечив вищі значення NDCG та Diversity, що свідчить про кращу якість ранжування та більшу різноманітність рекомендацій. Водночас у деяких випадках алгоритм не зміг сформувати релевантні рекомендації через недостатню схожість між користувачами або обмежену кількість взаємодій у тестовому наборі.

Гібридний підхід у більшості випадків продемонстрував найбільш збалансовані результати. Поєднання контентно-орієнтованого алгоритму та колаборативної фільтрації дозволило компенсувати окремі недоліки кожного з підходів. Додаткове використання контекстної інформації також позитивно вплинуло на релевантність рекомендацій та якість їх ранжування.

Значення метрики Recall у багатьох випадках наближалося до одиниці. Це свідчить про те, що алгоритми успішно знаходили більшість релевантних товарів із тестового набору користувачів. Водночас значення Precision залишалися порівняно невисокими, що пояснюється значною кількістю рекомендованих товарів у сформованих списках рекомендацій.

Метрика NDCG показала, що в багатьох випадках релевантні товари розташовувалися на верхніх позиціях списку рекомендацій. Це є важливим показником ефективності рекомендаційної системи, оскільки користувачі найчастіше взаємодіють саме з першими елементами списку рекомендованих товарів.

Значення Diversity продемонстрували різний рівень різноманітності рекомендацій для різних алгоритмів. Найвищі показники Diversity переважно спостерігалися у алгоритму колаборативної фільтрації, що пояснюється використанням поведінкових шаблонів інших користувачів. Контентно-орієнтований підхід, навпаки, частіше формував рекомендації схожих товарів, через що різноманітність рекомендацій була нижчою.

Отримані результати підтверджують працездатність реалізованих

алгоритмів рекомендацій та демонструють можливість їх використання у межах інформаційної системи маркетплейсу. Проведене експериментальне дослідження дозволило порівняти ефективність різних підходів та визначити особливості їх поведінки в умовах роботи з поведінковими та контекстними даними користувачів.

4.4 Порівняння ефективності підходів за обраними метриками

Порівняння ефективності реалізованих підходів виконувалося на основі значень метрик Precision, Recall, F1-score, NDCG та Diversity, отриманих у процесі експериментального дослідження. Використання декількох метрик дозволило комплексно оцінити якість роботи рекомендаційних алгоритмів з точки зору точності, повноти, якості ранжування та різноманітності рекомендацій.

Графічне представлення середніх значень метрик для всіх реалізованих алгоритмів наведено на рисунку 4.1. На графіку відображено усереднені результати контентно-орієнтованого підходу, алгоритму колаборативної фільтрації та гібридного методу за метриками Precision, Recall, F1, NDCG та Diversity.

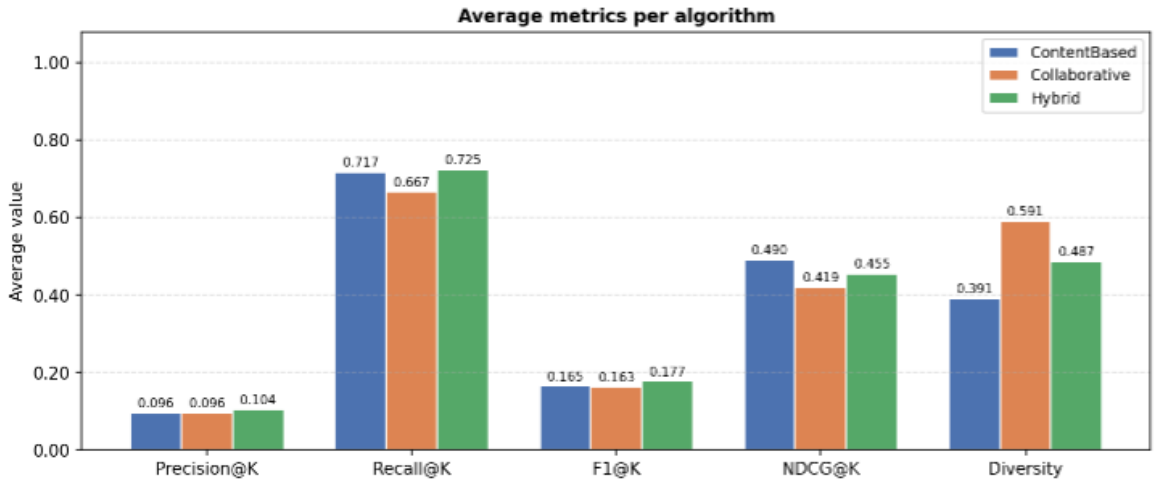

Рисунок 4.1 – Середні значення метрик алгоритмів

Аналіз графіка показує, що гібридний підхід продемонстрував найкращі результати за більшістю метрик. Зокрема, саме гібридний алгоритм отримав найвищі значення Recall та F1, що свідчить про здатність алгоритму знаходити найбільшу кількість релевантних товарів та забезпечувати найбільш збалансоване співвідношення між точністю та повнотою рекомендацій.

Контентно-орієнтований підхід показав найкраще значення NDCG серед усіх реалізованих алгоритмів. Це означає, що релевантні товари у цьому підході частіше розташовувалися на верхніх позиціях списку рекомендацій. Такий результат пояснюється використанням характеристик товарів та прямим аналізом інтересів користувача.

Алгоритм колаборативної фільтрації продемонстрував найвищий показник Diversity. Це свідчить про те, що даний підхід формував найбільш різноманітні рекомендації серед усіх реалізованих алгоритмів. Використання поведінкових шаблонів інших користувачів дозволило системі рекомендувати ширший спектр товарів різних категорій.

Значення Precision для всіх трьох алгоритмів залишилися близькими між собою. Це свідчить про те, що всі реалізовані підходи демонструють подібний рівень точності при формуванні рекомендацій, хоча кожен алгоритм досягає цього результату різними способами.

Графічне представлення кількості випадків, у яких алгоритми не змогли сформувати релевантні рекомендації, наведено на рисунку 4.2. На діаграмі відображено частку рядків експерименту, для яких одночасно значення Precision та Recall дорівнювали нулю.

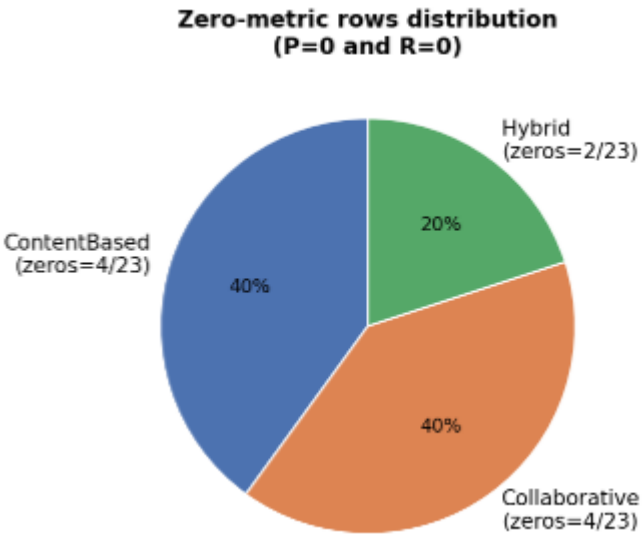

Рисунок 4.2 – Кількість нульових рядків алгоритмів

Аналіз результатів показує, що контентно-орієнтований підхід та алгоритм колаборативної фільтрації мали однакову кількість нульових результатів – по 4 випадки із 23 проведених експериментів. Це свідчить про те, що в окремих ситуаціях алгоритми не змогли сформувати рекомендації, які б містили релевантні товари з тестового набору користувача.

Гібридний підхід продемонстрував кращу стабільність роботи та мав лише 2 випадки повної відсутності релевантних рекомендацій. Такий результат пояснюється поєднанням двох різних підходів формування рекомендацій, що дозволяє компенсувати недоліки окремих алгоритмів та зменшити ймовірність отримання порожніх або нерелевантних результатів.

Результати експериментального дослідження підтвердили працездатність усіх трьох реалізованих підходів до формування рекомендацій у межах розробленої системи маркетплейсу. Кожен із алгоритмів продемонстрував здатність формувати релевантні рекомендації на основі поведінкових, контентних та контекстних даних користувачів.

Контентно-орієнтований підхід показав стабільність при роботі з характеристиками товарів та забезпечив хорошу якість ранжування рекомендацій. Алгоритм колаборативної фільтрації продемонстрував найвищу різноманітність рекомендацій, проте в окремих випадках залежав від

кількості доступних поведінкових даних користувачів.

Найбільш збалансовані результати продемонстрував гібридний підхід, який поєднав переваги двох базових алгоритмів та забезпечив найкращі загальні показники за сукупністю обраних метрик. Додатково гібридний метод показав найменшу кількість випадків повної відсутності релевантних рекомендацій, що підтверджує його вищу стабільність у порівнянні з окремими підходами.

4.5 Визначення переваг та недоліків кожного підходу

В результаті дослідження було визначено, що контентно-орієнтований підхід має перевагу у вигляді стабільної роботи навіть за відсутності великої кількості користувачів у системі. Алгоритм формує рекомендації на основі характеристик товарів та особистої історії взаємодій користувача, що дозволяє забезпечувати персоналізацію навіть для нових або малоактивних користувачів.

Алгоритм колаборативної фільтрації має перевагу у вигляді здатності виявляти приховані поведінкові закономірності між користувачами. Це дозволяє рекомендувати товари, які користувач раніше не переглядав, але які можуть бути для нього цікавими на основі поведінки схожих користувачів.

Гібридний підхід має основну перевагу у вигляді поєднання сильних сторін контентно-орієнтованого алгоритму та колаборативної фільтрації. Це дозволяє підвищити стабільність рекомендацій, зменшити кількість нерелевантних результатів та забезпечити більш збалансовані значення метрик оцінювання.

У таблиці 4.2 наведено узагальнене порівняння переваг та недоліків трьох реалізованих підходів до формування рекомендацій. Таблиця дозволяє наочно оцінити сильні та слабкі сторони контентно-орієнтованого алгоритму, колаборативної фільтрації та гібридного методу.

Таблиця 4.2 – Узагальнена таблиця результатів дослідження

Підхід	Переваги	Недоліки
Контентно орієнтований	Стабільна робота при малій кількості користувачів; персоналізація на основі характеристик товарів; можливість пояснення причин рекомендацій	Низька різноманітність рекомендацій; схильність до рекомендації схожих товарів; залежність від якості опису товарів
Колаборативна фільтрація	Виявлення поведінкових закономірностей між користувачами; висока різноманітність рекомендацій; можливість рекомендації нових товарів	Проблема «холодного старту»; залежність від великої кількості взаємодій; нестабільність при недостатній кількості даних
Гібридний підхід	Поєднання переваг двох алгоритмів; більш збалансовані результати; підвищення стабільності рекомендацій; зменшення кількості нерелевантних результатів	Вища складність реалізації; збільшення обчислювальних витрат; необхідність підтримки декількох моделей рекомендацій

Представлене порівняння демонструє, що кожен із підходів має власні особливості. При цьому гібридний підхід дозволяє найбільш ефективно поєднати переваги двох базових алгоритмів та мінімізувати їх основні недоліки.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи проведено дослідження підходів до побудови рекомендаційних систем для маркетплейсів, що дало змогу визначити особливості їх використання у сучасних інформаційних системах електронної комерції.

Виконано аналіз основних типів рекомендаційних алгоритмів, серед яких контентно-орієнтований підхід, колаборативна фільтрація, гібридний метод, демографічний та контекстно-орієнтований підходи, що дало змогу визначити їх переваги, недоліки та особливості практичного використання.

Проведено порівняльний аналіз існуючих підходів до формування рекомендацій, що дало змогу обґрунтувати вибір контентно-орієнтованого алгоритму, алгоритму колаборативної фільтрації та гібридного підходу для подальшої реалізації та дослідження.

Виконано аналіз рекомендаційних систем сучасних маркетплейсів, зокрема платформ Rozetka, Amazon та Allegro, що дало змогу дослідити особливості реалізації рекомендаційних механізмів у реальних системах електронної комерції.

Сформовано постановку задачі кваліфікаційної роботи, що дало змогу визначити основні етапи створення серверної частини маркетплейсу з інтегрованою рекомендаційною підсистемою та механізмом експериментального оцінювання алгоритмів рекомендацій.

Побудовано функціональну модель рекомендаційної підсистеми з використанням методології IDEF0, що дало змогу формалізувати основні процеси формування рекомендацій та виконати їх структурну декомпозицію.

Визначено основні метрики оцінювання якості рекомендаційних алгоритмів, серед яких Precision, Recall, F1-score, NDCG та Diversity, що дало змогу забезпечити комплексне оцінювання ефективності реалізованих алгоритмів рекомендацій.

Визначено архітектурні особливості реалізації серверної частини

маркетплейсу на платформі .NET, що дало змогу побудувати масштабовану структуру серверного застосунку з використанням патерну Mediator та механізму Dependency Injection.

Сформовано тестовий набір даних із інформацією про користувачів, товари та історію взаємодій, що дало змогу забезпечити проведення експериментального дослідження реалізованих рекомендаційних алгоритмів.

Реалізовано базовий функціонал серверної частини маркетплейсу, включаючи роботу з товарами, категоріями, брендами та взаємодіями користувачів, що дало змогу створити основу для інтеграції рекомендаційної підсистеми.

Реалізовано контентно-орієнтований алгоритм, алгоритм колаборативної фільтрації та гібридний метод рекомендацій, що дало змогу дослідити особливості роботи різних підходів до формування персоналізованих рекомендацій.

Реалізовано повний цикл роботи рекомендаційних алгоритмів, включаючи отримання даних із бази, аналіз взаємодій користувачів, формування рекомендацій, збереження результатів та повернення відповідей через API, що дало змогу інтегрувати рекомендаційний функціонал у серверну частину маркетплейсу.

Реалізовано механізм автоматизованого експериментального дослідження, що дало змогу виконувати автоматичний запуск усіх реалізованих алгоритмів рекомендацій та проводити їх порівняльне оцінювання.

Проведено аналіз отриманих результатів експериментального дослідження та виконано порівняння ефективності реалізованих підходів за обраними метриками, що дало змогу визначити особливості та ефективність кожного алгоритму рекомендацій.

Результатом виконання кваліфікаційної роботи стала реалізована серверна частина маркетплейсу з інтегрованою рекомендаційною підсистемою.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Laudon K. C., Traver C. G. E-commerce 2023: Business, Technology, Society. 18th ed. Pearson, 2023. 912 p.

2. Chopra S. Supply Chain Management: Strategy, Planning, and Operation. 8th ed. Pearson, 2022. 720 p.

3. Ricci F., Rokach L., Shapira B. Recommender Systems Handbook. 3rd ed. Springer, 2022. 1548 p.

4. Aggarwal C. C. Recommender Systems: The Textbook. Springer, 2016. 498 p.

5. Adomavicius G., Tuzhilin A. Context-Aware Recommender Systems // Recommender Systems Handbook. Springer, 2015. P. 191–226.

6. Jannach D., Zanker M., Felfernig A., Friedrich G. Recommender Systems: An Introduction. Cambridge University Press, 2010. 336 p.

7. Burke R. Hybrid Web Recommender Systems // The Adaptive Web. Springer, 2007. P. 377–408.

8. Rozetka. Інтернет-магазин та маркетплейс товарів. URL: https://rozetka.com.ua/ua/ (дата звернення: 12.03.2026).

9. Amazon. Amazon.com Official Website. URL: https://www.amazon.com/ (дата звернення: 12.03.2026).

10. Allegro. Allegro Official Marketplace Platform. URL: https://allegro.pl/ (дата звернення: 12.03.2026).

11. Ricci F., Rokach L., Shapira B., Kantor P. B. Recommender Systems Handbook. Springer, 2011. 842 p.

12. Bobadilla J., Ortega F., Hernando A., Gutiérrez A. Recommender Systems Survey // Knowledge-Based Systems. 2013. Vol. 46. P. 109–132.

13. Elmasri R., Navathe S. B. Fundamentals of Database Systems. 7th ed. Pearson, 2015. 1280 p.

14. Marca D. A., McGowan C. L. SADT: Structured Analysis and Design Technique. McGraw-Hill, 1988. 392 p.

15. FIPS PUB 183. Integration Definition for Function Modeling (IDEF0). National Institute of Standards and Technology, 1993. 128 p.

16. Colquhoun G. J., Baines R. W., Crossley R. E. A State of the Art Review of IDEF0 // International Journal of Computer Integrated Manufacturing. 1993. Vol. 6, No. 4. P. 252–264.

17. Koren Y., Bell R., Volinsky C. Matrix Factorization Techniques for Recommender Systems // Computer. 2009. Vol. 42, No. 8. P. 30–37.

18. Resnick P., Varian H. R. Recommender Systems // Communications of the ACM. 1997. Vol. 40, No. 3. P. 56–58.

19. Schafer J. B., Konstan J., Riedl J. Recommender Systems in E-Commerce // Proceedings of the 1st ACM Conference on Electronic Commerce. 1999. P. 158–166.

20. Sarwar B., Karypis G., Konstan J., Riedl J. Item-Based Collaborative Filtering Recommendation Algorithms // Proceedings of the 10th International Conference on World Wide Web. 2001. P. 285–295.

21. Lops P., Gemmis M. de, Semeraro G. Content-based Recommender Systems: State of the Art and Trends // Recommender Systems Handbook. Springer, 2011. P. 73–105.

22. Cover T. M., Thomas J. A. Elements of Information Theory. 2nd ed. Wiley-Interscience, 2006. 776 p.

23. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. 9th ed. McGraw-Hill, 2019. 992 p.

24. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002. 533 p.

25. Microsoft. ASP.NET Core Documentation. URL: https://learn.microsoft.com/aspnet/core/ (дата звернення: 12.03.2026).

26. Microsoft. Entity Framework Core Documentation. URL: https://learn.microsoft.com/ef/core/ (дата звернення: 12.03.2026).