МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет інформаційних технологій
Кафедра інтелектуальних технологій

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

НА ТЕМУ

Програмний модуль підбору відео ігор

Галузь знань **12 «Інформаційні технології»**

Спеціальність **122 «Комп'ютерні науки»**

Освітня програма «**Комп'ютерні науки»**

Освітній рівень: бакалавр

Виконав: студент 4 курсу, групи КН- 42

_Пхайко С. Д_____ 
(прізвище та ініціали)

Керівник Мінаєва Ю. І. _____
(прізвище та ініціали)
_____ доц., к.т.н _____
(науковий ступінь, звання)

Випускна кваліфікаційна робота бакалавра допущена до захисту
рішенням кафедри *інтелектуальних технологій*
Протокол № 13 від 05.06.2023 р.
зав. кафедри_____ доц. Іларіонов О.Є.

Київ – 2023

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інтелектуальних технологій

Спеціальність 122 «Комп'ютерні науки»

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних технологій

Іларіонов О.Є.

_____

"___"_____2023 р.

ЗАВДАННЯ

НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

_____Пхайко Софія Дмитрівна_____

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи)

_____Програмний модуль підбору відео ігор_____

затверджена протоколом засідання кафедри від « 11 » листопада 2022 р. № 4

2. Термін здачі студентом закінченого проекту (роботи) 29 травня 2023 року

3. Вихідні дані до проекту (роботи)

_____Програмний модуль ,що надає користувачеві можливість оцінити та отримати рекомендовані відео ігри_____

4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)

_1.Аналітичний огляд області застосування рекомендаційних систем _____

_2. Розробка архітектури програмного модулю _____

_3.Тестування програмного модулю_____

5. Перелік презентаційного матеріалу (з точним зазначенням обов'язкових презентацій)

_Актуальність теми(1 слайд), аналіз предметної області(2 слайди), рекомендаційні системи та існуючі рішення (2-4 слайди) , вимоги (1 слайд), опис методу та метрик (3-4 слайди), опис бази даних та інтерфейс модулю (3 слайди) , висновки (1 слайд)_____

6. Консультанти з випускної кваліфікаційної роботи із зазначенням розділів випускної кваліфікаційної роботи, що їх стосуються

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 15 лютого 2023 року

Керівник _____ /доц., к.т.н. Мінаєва Ю.І. /
(підпис) (ПІБ)

Завдання прийняв до виконання ___ / Пхайко С.Д. /
(підпис) (ПІБ)

КАЛЕНДАРНИЙ ПЛАН

ор. №	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів випускної кваліфікаційної роботи	Примітка
1	Аналіз постановки задачі, формалізація задачі, аналіз літературних джерел та написання 1 розділу роботи	16.02.2023 – 01.03.2023	
2	Проектно-технологічна реалізація програмного модулю	01.03.2023 – 7.04.2023	
3	Розробка та тестування модулю для підбору відео ігор	10.04.2023 – 7.05.2023	
4	Оформлення пояснювальної записки та презентації	10.07.2023 – 19.05.2023	

Студент-дипломник ____ / Пхайко С. Д /
(підпис) (ПІБ)

Керівник випускної кваліфікаційної роботи _____ /доц., к.т.н. Мінаєва Ю.І. /
(підпис) (ПІБ)

Анотація

Пхайко Софія Дмитрівна виконала випускну кваліфікаційну роботу на тему «Програмний модуль підбору відео ігор» за спеціальністю 122 – «Комп'ютерні науки».

У випускній кваліфікаційній роботі проведено аналіз існуючих рішень та методів для реалізації рекомендаційних систем, розглянуто бізнес процеси та сформовано функціональні та нефункціональні вимоги до програмного модулю, розроблено програмне забезпечення, що може надавати користувачеві рекомендації щодо підбору відео ігор.

Ключові слова: рекомендації, система, користувач.

Summary

Sofiya Dmytrivna Phaiko completed the graduation qualification work on the topic "Software module for selecting video games" in the specialty 122 - "Computer Science".

In the final qualification work, an analysis of existing solutions and methods for the implementation of recommendation systems was carried out, business processes were considered and functional and non-functional requirements for the software module were formed, software was developed that can provide the user with recommendations on the selection of video games.

Keywords: recommendations, system, user

ЗМІСТ

Перелік умовних позначень і скорочень

РС – рекомендаційна система

KNN - K Nearest Neighbors

ВСТУП

Рекомендаційні системи є спеціальними програмами, які ставлять за мету надавати користувачам рекомендації щодо різних продуктів або сервісів, враховуючи їхні переваги та інтереси.

З появою Інтернету обсяг доступної інформації, з якою ми зіштовхуємося щодня, значно зріс. Це означає, що ми мусимо здійснювати вибір серед величезної кількості альтернатив, коли шукаємо щось конкретне. З іншого боку, власники інтернет-магазинів та сервісів зацікавлені в персоналізованій рекламі та рекомендаціях для кожного користувача, оскільки цей підхід може значно збільшити їхні прибутки. Це призвело до зростання інтересу до розробки та вдосконалення рекомендаційних систем.

Сьогодні існує безліч веб-сайтів, що надають різні типи контенту, такі як новини, блоги, музика та кіно. Кожен з них містить величезний обсяг інформації, але не вся вона може бути цікавою для конкретного відвідувача. Рекомендаційні системи використовуються для підбору контенту, який буде корисним саме для цього користувача. На відміну від пошукових систем, рекомендаційна система не потребує чіткого запиту. Користувачеві пропонується оцінити деякі об'єкти зі зібраної колекції, а на основі цих оцінок система будує припущення і повертає результати, які максимально відповідають його потребам.

Рекомендаційні системи є необхідними в наш час, оскільки вони значно зменшують час пошуку корисної та цікавої інформації.

Метою є розробка програмного модулю для підбору відео ігор, які максимально відповідають смаку користувача або групи користувачів, за рахунок того, що в системі буде наявна достатня кількість застосунків, які користувач зможе обрати.

Предметом дослідження даної дипломної роботи являються методи формування рекомендацій.

Об’єктом дослідження є засоби надання рекомендацій.

РОЗДІЛ 1 АНАЛІТИЧНИЙ ОГЛЯД ОБЛАСТІ ЗАСТОСУВАННЯ РЕКОМЕНДАЦЙНИХ СИСТЕМ

Рекомендаційні системи знаходять широке застосування у багатьох галузях, включаючи торгівлю, музику, фільми, відеоігри, книги, новини, подорожі та багато іншого. Основна мета рекомендаційних систем - надання користувачам інформації, яка відповідає їхнім потребам та інтересам.

Одним з найбільш популярних використань рекомендаційних систем - в електронній комерції. Багато інтернет-магазинів використовують рекомендаційні системи, щоб пропонувати споживачам товари, які вони можуть їх зацікавити. Наприклад, Amazon використовує свою рекомендаційну систему для підбору товарів, які можуть сподобатися користувачеві на основі його попередніх покупок та переглядів.

Іншим прикладом успішного використання рекомендаційних систем є музичні сервіси, такі як Spotify та Pandora. Ці сервіси аналізують музичні вподобання користувачів, щоб рекомендувати пісні та артистів, які можуть сподобатися. Це допомагає користувачам відкривати нову музику та зберігати їх улюблені композиції в одному місці.

Також рекомендаційні системи широко використовуються в інтернет-фільмах та відеоіграх. Наприклад, Netflix використовує рекомендаційну систему, щоб допомогти своїм користувачам знайти фільми та телешоу, які їм можуть сподобатися, на основі їхніх попередніх переглядів та вподобань.

У відеоіграх, таких як Fortnite, рекомендаційні системи можуть рекомендувати гравцям певні режими гри, які відповідають вподобанням та відповідному рівню навичок, або рекомендувати зброю і обладнання, що можуть бути корисними в конкретній ситуації у грі.

1.1 Аналіз предметної області

Рекомендаційні системи для відеоігор є дуже популярними серед гравців і виробників. Це обумовлено тим, що ці системи можуть значно полегшити процес

пошуку нових ігор та допомогти знайти ігри, які відповідають індивідуальним інтересам користувача.

Одним з найпопулярніших прикладів рекомендаційних систем для відеоігор є система Steam Recommendations, яка використовується на платформі Steam. Ця система аналізує ігрову активність користувачів, їхній інвентар та інші параметри, щоб надати індивідуальні рекомендації щодо нових ігор або розпродажів. За допомогою цієї системи Steam отримує значний прибуток, продавши більше ігор та підвищивши лояльність користувачів.

Іншим прикладом є рекомендаційна система Twitch, яка використовується на платформі для трансляцій відеоігор. Ця система аналізує перегляди трансляцій користувачів та рекомендує інші трансляції, які вони можуть бути зацікавлені в перегляді. Крім того, система Twitch також пропонує ігри, які можуть зацікавити користувачів, відповідно до їхніх інтересів та активності.

Загалом, рекомендаційні системи для відеоігор є дуже важливим елементом для виробників та гравців, оскільки вони можуть полегшити процес вибору ігор та збільшити продажі. Кількість користувачів таких систем становить мільйони, а прибуток компаній, які використовують рекомендаційні системи, може бути значним.[11]

Отже у сфері відео ігор PC мають певні переваги, а саме:

1. Покращення геймплею: можуть допомогти гравцеві знайти ігри, які найбільше підходять їхнім смакам та інтересам, що поліпшує їхній досвід гри. Крім того, системи рекомендацій можуть рекомендувати ігри, які мають схожі геймплеї, що може допомогти гравцеві розширити свої інтереси та відкрити нові види ігор.[11]
2. Збільшення продажів: можуть рекомендувати гравцям ігри, які їм сподобаються, що збільшує ймовірність їх покупки. Це дозволяє видавництвам та розробникам отримувати більше прибутку від продажу ігор.[11]

3. Підвищення залучення користувачів: можуть допомогти залучити нових користувачів до відеоігрової індустрії. Це можливо завдяки тому, що системи рекомендацій можуть виявити ігри, які можуть бути цікавими для новачків у світі відеоігор.[11]
4. Покращення реклами: можуть допомогти виробникам ігор показувати рекламу ігор конкретним групам аудиторії, що може збільшити ефективність рекламної кампанії.[11]
5. Замість того, щоб переглядати велику кількість ігор та фільтрувати їх вручну, користувач може отримати рекомендації на основі свого профілю та історії гри. Це зменшує час і зусилля, необхідні для пошуку ігор, і забезпечує більш точні рекомендації.[11]
6. Рекомендаційні системи можуть знизити витрати на маркетинг. Замість того, щоб витрачати гроші на рекламу ігор, компанії можуть використовувати рекомендаційні системи для того, щоб залучити користувачів до ігор через власні ресурси.[11]

1.2 Аналіз основних підходів для реалізації програмного модулю

Рекомендаційні системи - це програмні комплекси, які можуть бути поділені на чотири типи в залежності від підходів, які вони використовують. Нижче наведено опис підходів:

Коллаборативна фільтрація (Collaborative Filtering) - рекомендації засновані на історії оцінок як самого користувача, так і інших користувачів. Цей підхід має теоретично високу точність, але при цьому має одну важливу проблему - високий поріг входу.[8]

Засновані на контенті (content-based) - рекомендації засновані на даних, зібраних про кожен конкретний товар. Користувачеві рекомендуються об'єкти, схожі на ті, якими він раніше цікавився або вже купував.[8]

Оцінювання подібності на основі контенту - це метод рекомендацій, який використовує оцінки та вподобання користувачів для знаходження схожих

об'єктів та рекомендацій користувачеві. Цей метод дозволяє залучити нових користувачів з перших кроків, не чекаючи на отримання додаткової інформації про їх вподобання. За допомогою цього можна навіть рекомендувати об'єкти, які ще не мають оцінок від інших користувачів. Однак, основними недоліками методу є сильна залежність від тематики та зниження точності рекомендацій при збільшенні обсягу даних. Також, цей метод не є універсальним і може бути обмежений застосуванням у деяких сферах.[16]

Засновані на знаннях (knowledge-based) - рекомендації засновані на знаннях про предметну область (а не про кожен товар). Такий тип рекомендацій має високу точність, пропонуючи користувачеві те, що йому потрібно. Крім цього, система вивчає і аналізує взаємозв'язки між об'єктами, враховує ряд додаткових опцій, що відносяться до індивідуальних властивостей конкретного користувача.[8]

Додаткові переваги РС на основі знань включають можливість пропонувати нові або менш популярні об'єкти, що покращує диверсифікацію рекомендацій та розширює можливості вибору користувача. Також, цей тип систем не залежить від історії попередніх взаємодій користувача з ресурсом, що робить їх ефективними для нових користувачів.

Однак, системи на основі знань мають обмежену корисність у випадках, коли знання про предметну область обмежені або не повністю охоплюють всі можливі варіації вибору користувача. Крім того, розробка та підтримка таких систем можуть бути дорогими, оскільки потребують експертних знань та трудомістких процесів моделювання знань.

Гібридні (hybrid) - рекомендації засновані на комбінуванні коллаборативного і тематичних підходів, що дозволяє уникнути більшості недоліків, властивих кожній системі.[8]

Перевагою гібридних РС є їх велика швидкодія та здатність до досягнення кращих результатів порівняно з іншими методами. Однак, недоліками є дуже високі витрати на розробку такої системи, оскільки її реалізація є дуже складною,

а також складність її підтримки, оскільки навіть незначні зміни в роботі можуть призвести до змін у роботі алгоритму.

Існують два методи збору інформації про користувачів: явний та неявний збір даних. Явний збір передбачає, що користувачі самостійно надають інформацію про свої уподобання та інтереси, наприклад, відповідаючи на запитання. Неявний збір, у свою чергу, передбачає збір інформації без прямого запиту до користувача, використовуючи різні методи, такі як фіксування покупок, переглядів, оцінок та коментарів на сайтах. Проте, такі методи можуть створювати етичні проблеми, оскільки необхідно захищати персональні дані користувачів. [8]

1.3Аналіз бізнес-процесів предметної області

Аналіз бізнес-процесів є невід'ємною частиною будь-якого проекту створення чи розвитку корпоративної інформаційної системи. Використання функціональної моделі "ЯК Є" дозволяє чітко зафіксувати, які ділові процеси здійснюються на підприємстві, які інформаційні об'єкти використовуються при виконанні ділових процесів і окремих операцій. Функціональна модель "ЯК Є" є відправною точкою для аналізу потреб підприємства, виявлення проблем і "вузьких" місць, а також розробки проекту вдосконалення ділових процесів.

Основною метою бізнес-процесу є задоволення вимог клієнтів. Всіх клієнтів можна розділити на кілька типів:

- первинні клієнти, які одержують первинний вихід,
- вторинні клієнти, які знаходяться поза процесом і одержують вторинні виходи,
- непрямі клієнти, які не одержують первинного виходу, але є наступними в ланцюжку, тому пізніший за часом вихід відображається на них, та
- зовнішні непрямі клієнти-споживачі.

Для даного процесу була побудована бізнес-архітектура, яка зображена на рис.1.1, за допомогою фреймворку TOGAF відповідно до поточного стану ("AS IS").

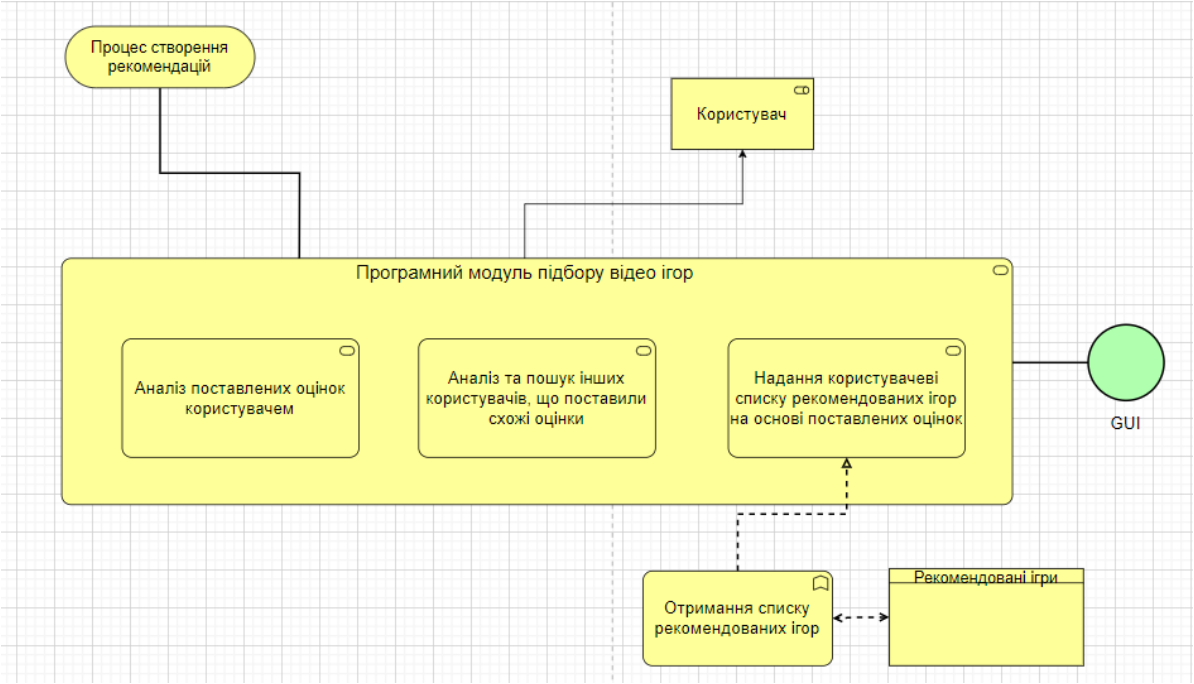

Рисунок 1.1 – Бізнес модель ЯК Є

1.4 Огляд існуючих рішень

Розглянемо нижче дві платформи, що користуються популярністю серед користувачів:

Першою із таких платформ є Steam. Ця платформа виступає засобом техзахисту авторських прав, платформою для багатокористувацьких ігор і потокового мовлення, а також соцмережею для геймерів. До того ж, клієнт Steam забезпечує установлення і регулярне оновлення ігор, їх зберігання на хмарному сервері, текстовий і голосовий зв'язок між гравцями.

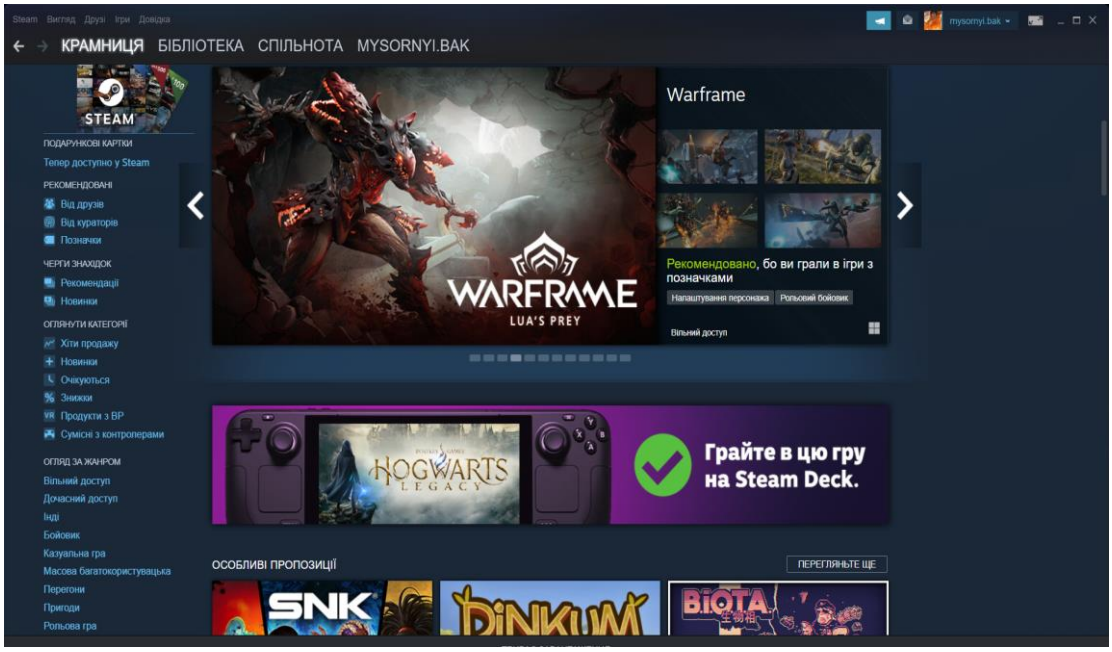

Рисунок 1.2 – Головне меню платформи Steam

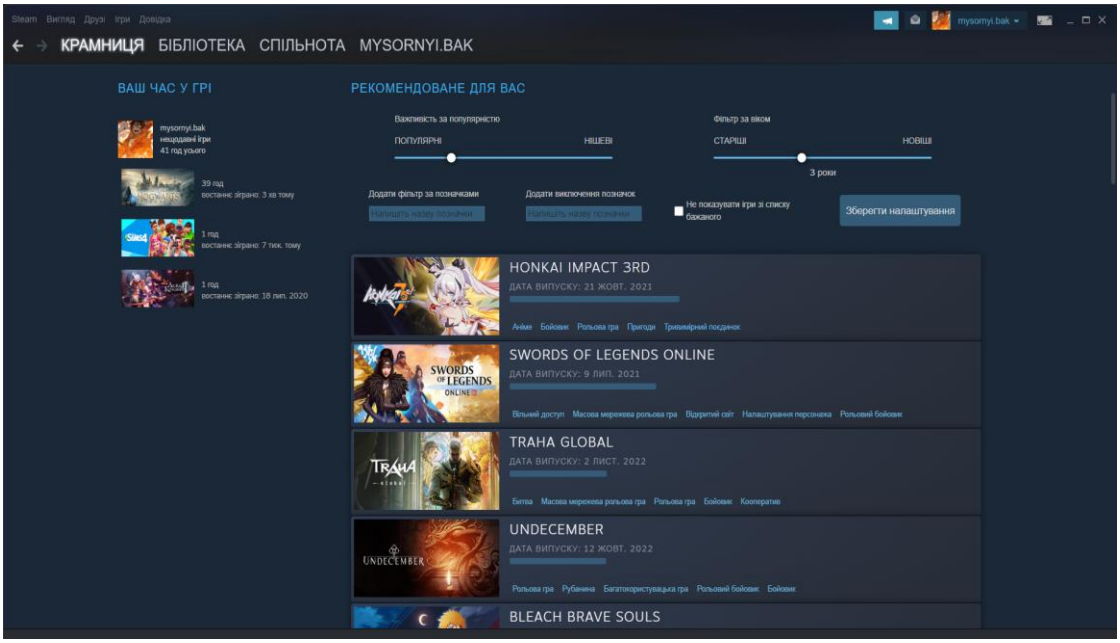

Рисунок 1.3 – Особисті рекомендації

Другою такою ж популярною платформою є Epic Games Store.

Epic Games Store - це магазин цифрових ігор, що запущений компанією Epic Games, що займається комп'ютерними іграми, найбільш відома своєю розробкою як Unreal Engine і Fortnite.

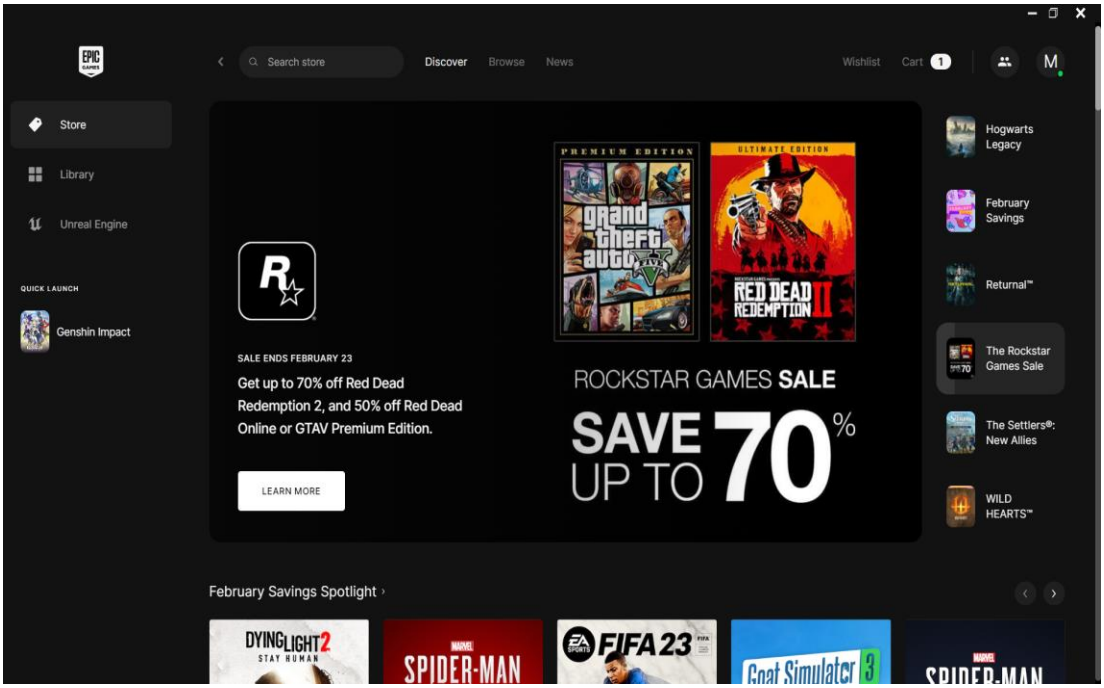

Рисунок 1.4 – Головна сторінка

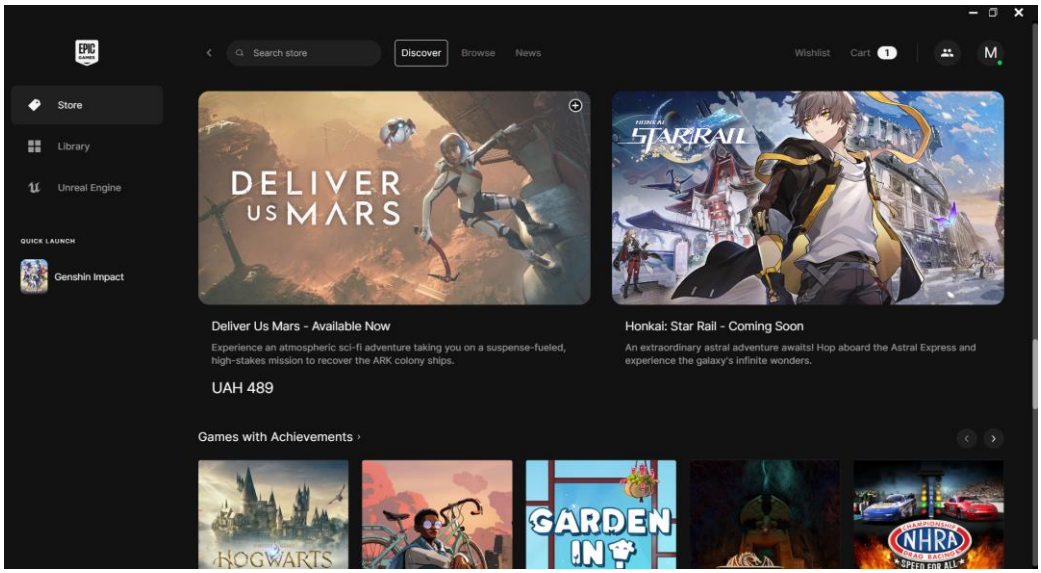

Рисунок 1.5 – Підбірка рекомендацій

Порівняння рекомендаційних систем Steam та Epic Games Store можна провести за наступними критеріями:

Таблиця 1. Порівняльна таблиця

Критерій	Steam	Epic Games Store(EGS)
Бібліотека сягає понад 5 тисяч ігор	+	-

Точність рекомендацій підбору ігор за вподобаннями користувача	+	-
Повідомлення про рекомендації	-	-
Наявність відгуків користувачів	+	-
Наявність оцінок	+	+
Підтримка різних платформ	+	-
Зручний інтерфейс	+	+

- Бібліотека сягає понад 5 тисяч ігор: Steam має значно більший обсяг бібліотеки ігор порівняно з EGS, тому рекомендаційна система Steam може пропонувати більше варіантів для користувачів.
- Точність рекомендацій: Чим більше точність, тим більш ефективно рекомендаційна система працює. Steam використовує різноманітні алгоритми рекомендацій та надає користувачам більше інформації про те, як вони були створені. EGS також має рекомендаційну систему, проте вона може бути менш точною, оскільки її алгоритми рекомендацій можуть бути менш розвиненими.
- Повідомлення про рекомендації: Steam надсилає сповіщення про рекомендації користувачеві, що може допомогти зберегти його інтерес в платформі. EGS не надає таких повідомлень.
- Наявність відгуків користувачів: Steam дозволяє користувачам залишати відгуки про ігри, що може допомогти користувачам прийняти рішення щодо придбання. EGS не має такої можливості.
- Підтримка різних платформ: Steam підтримує ігри на різних платформах, таких як Windows, MacOS та Linux, що робить його

платформою з великою аудиторією. EGS зосереджується в основному на іграх для Windows.

- Інтерфейс: Steam має дуже зручний та легкий у використанні інтерфейс, який дозволяє користувачам швидко знайти та купити ігри, переглядати свій профіль та історію покупок, а також взаємодіяти з іншими користувачами у спільноті. Epic Games Store має сучасний та естетичний дизайн інтерфейсу, але деякі користувачі можуть вважати його менш зручним для використання. Крім того, Epic Games Store пропонує кращу інтеграцію з бібліотекою Unreal Engine та іншими сервісами Epic Games.

Узагальнюючи, можна сказати, що обидві платформи мають переваги і недоліки.

1.5 Постановка задачі

На сьогодні багато веб-сайтів використовують РС для своїх користувачів. Вони пропонують різні варіанти, такі як пов’язані продукти або направляють людей зі схожими інтересами, які зареєструвалися на цьому сайті. Ці системи оброблюють багато інформації, щоб показати переваги потенційних користувачів.

Такі механізми покращують взаємодію між користувачами та веб-сайтом, тому що замість того, щоб надавати статичну інформацію, вона змінюється динамічно: інструкції створюються окремо для кожного користувача на основі його попередньої активності на веб-сайті, а також можна брати інформацію від інших відвідувачів. У більшості випадків розробка системи наведення полягає в модернізації алгоритму.

Основна мета розвитку рекомендаційних сервісів полягає у забезпеченні відвідувачів веб-сайтів найточнішими рекомендаціями, які відповідають їхнім запитам у конкретний момент часу. Цього можна досягти лише за умови

постійного вдосконалення математичних моделей та алгоритмів, що підтримують функціонування рекомендаційних сервісів.

Методологія IDEF дозволяє зображувати та аналізувати складні системи через різноманітні моделі діяльності. Одна з них - IDEF0, що дозволяє побудувати контекстну діаграму процесу діяльності програмного застосунку з різних точок зору. Це дає можливість краще зрозуміти роботу системи та проаналізувати її потенційні недоліки.

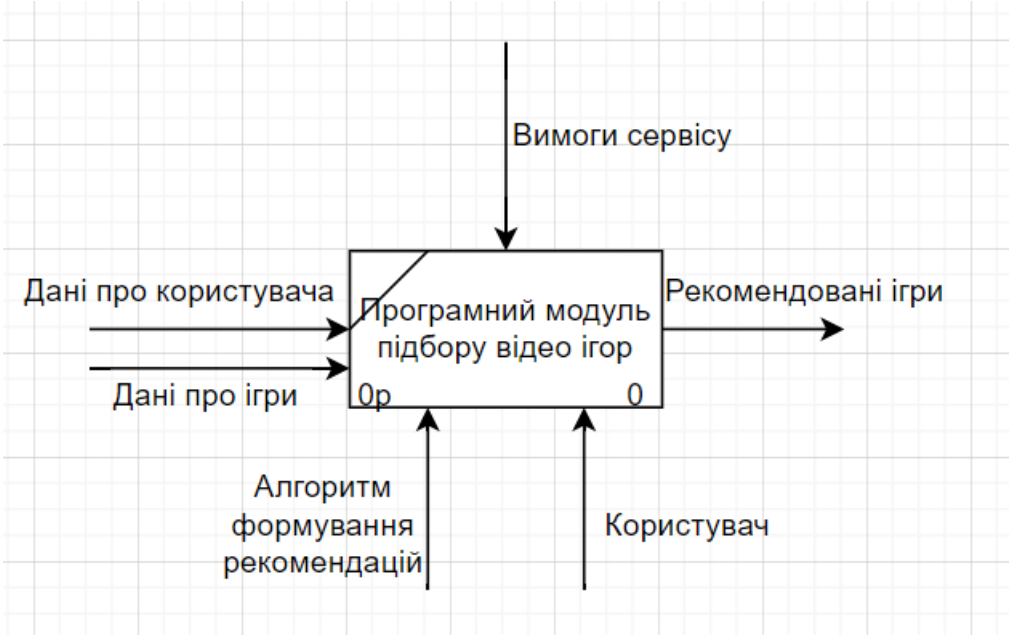

Рисунок 1.7 – Контекстна діаграма процесу діяльності

На контекстній діаграмі процесу діяльності визначаються головні актори, які взаємодіють з системою, та показуються основні потоки даних між ними. Вона допомагає ідентифікувати зовнішні системи, людей або інші процеси, які взаємодіють з розглянутим процесом діяльності.

1.6 Визначення функціональних та нефункціональних вимог

Кожен користувач має різні вподобання, й постійно зайнятий пошуком чогось нового у напрямку, що їх цікавить. Для роботи, розвитку чи інших потреб користувачеві потрібні різні групи веб-сервісів, додатків тощо. Інколи дуже важко знайти деякі застосунки у певній категорії , бо система рекомендує дещо

інше чи те ,що не цікавить . Іноді навіть через незручний інтерфейс користувач має проблеми із пошуком потрібних додатків .

Відповідно до вищевказаних причин, вимоги до системи будуть полягати у тому, щоб створити програмний модуль , що буде зручним та зрозумілим у використанні.

Для коректного функціонування веб-застосунку достатньо щоб версія браузеру відповідала, або була вищою однієї з нижче наведених:

- Edge 38.0+
- Chrome 53.0+
- Safari 8.0+
- Firefox 42.0+

Для будь-якого браузеру не має бути вимкнена підтримка JavaScript (вмикається/ вимикається в налаштуваннях браузеру). Необхідне з’єднання із мережею Інтернет.

Вимоги до веб-серверу:

Для того щоб система працювала коректно для всіх користувачів, обсяг оперативної пам’яті має бути щонайменше 4 Гб, і не менше 20 Гб вільних на жорсткому диску.

На основі проведеного аналізу, програмний застосунок також має задовольняти такі вимоги:

Функціональні вимоги:

1. Відображення списку відео ігор у зручному форматі.
2. Персоналізований підбір відеоігор: Система повинна аналізувати вподобання користувача, щоб зробити персоналізований підбір відеоігор.
3. Показ рейтингу відеоігор: Система повинна відображати середню оцінку відеоігор серед усіх користувачів, щоб користувач міг зробити свій вибір на підставі цього.
4. Зручний та зрозумілий інтерфейс.

Нефункціональні вимоги:

1. Швидкодія. Система повинна бути досить швидкою, щоб користувачі не чекали занадто довго на відповідь від системи.
2. Надійність. Система повинна бути стабільною та надійною, щоб користувачі не мали проблем з використанням.
3. Система повинна забезпечувати безпеку персональних даних користувачів.
4. Система повинна бути легкою в використанні, щоб користувачі могли швидко зрозуміти, як користуватися системою.
5. Система повинна бути легкою в масштабуванні, щоб можна було збільшувати обсяги обробки даних та кількість користувачів без зниження продуктивності.
6. Система повинна бути сумісною з різними платформами та браузерами, щоб користувачі могли використовувати її на будь-якому пристрої.
7. Система повинна бути ефективною в роботі, щоб користувачі могли швидко знайти потрібну інформацію та відеоігри, які їм підходять.

ВИСНОВКИ ДО РОЗДІЛУ 1

Були проаналізовані основні підходи, що застосовуються для реалізації програмного забезпечення в даній області.

Проведений аналіз бізнес-процесів предметної області дозволив отримати уявлення про ключові процеси, проблеми та потреби, які вимагають уваги при розробці програмного забезпечення.

Був проведений огляд існуючих рішень, які вже використовуються в даній області. Це дозволило оцінити наявні можливості, переваги та недоліки існуючих систем і використати ці знання при розробці нового програмного забезпечення.

Поставлено задачу розробки нового програмного забезпечення, яка відповідає виявленим потребам та вимогам предметної області. Це створює основу для подальшої розробки та імплементації програмного продукту.

Визначені основні вимоги до програмного забезпечення, які повинні бути враховані під час розробки. Ці вимоги включають функціональність, надійність, ефективність та інші аспекти, необхідні для досягнення успішного впровадження програмного забезпечення.

РОЗДІЛ 2 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМНОГО МОДУЛЮ ПІДБОРУ ВІДЕО ІГОР

2.1 Колаборативна фільтрація

Колаборативна фільтрація є ефективним способом для створення рекомендаційних систем, оскільки вона використовує дані про поведінку користувачів, які є ключовими для розуміння їх вподобань. За допомогою цього методу можна швидко та ефективно знайти спільні ознаки серед користувачів та рекомендувати їм нові продукти, що може призвести до збільшення продажів та задоволеності користувачів.

Однак, колаборативна фільтрація має деякі обмеження. Вона не може враховувати вподобання користувачів, які не мають схожої поведінки з іншими користувачами, що може призвести до втрати можливих продажів. Крім того, коли користувач має мало історії покупок або рейтингів, система може бути менш точною при рекомендації продуктів.

Існує два підходи до колаборативної фільтрації: колаборативна фільтрація на основі користувачів (user-based) та колаборативна фільтрація на основі предметів (item-based). В колаборативній фільтрації на основі користувачів враховується інформація про інтереси та поведінку користувачів, а в колаборативній фільтрації на основі предметів враховується схожість між предметами.[18]

Колаборативна фільтрація, що базується на користувачах(user-based): Розраховується подібність людей у матриці користувача та елемента. Наприклад, ми вважаємо, що є три користувачі та чотири гри. Припустимо, що перший користувач оцінів усі запропоновані ігри , другий оцінів лише «гру2» ,а третій – «гру2» та «гру3». Отже, для третього користувача будуть рекомендовані такі ігри: «гра1» та «гра4», адже перший та третій користувач схожі за типом. Цей приклад зображено на рис.2.1

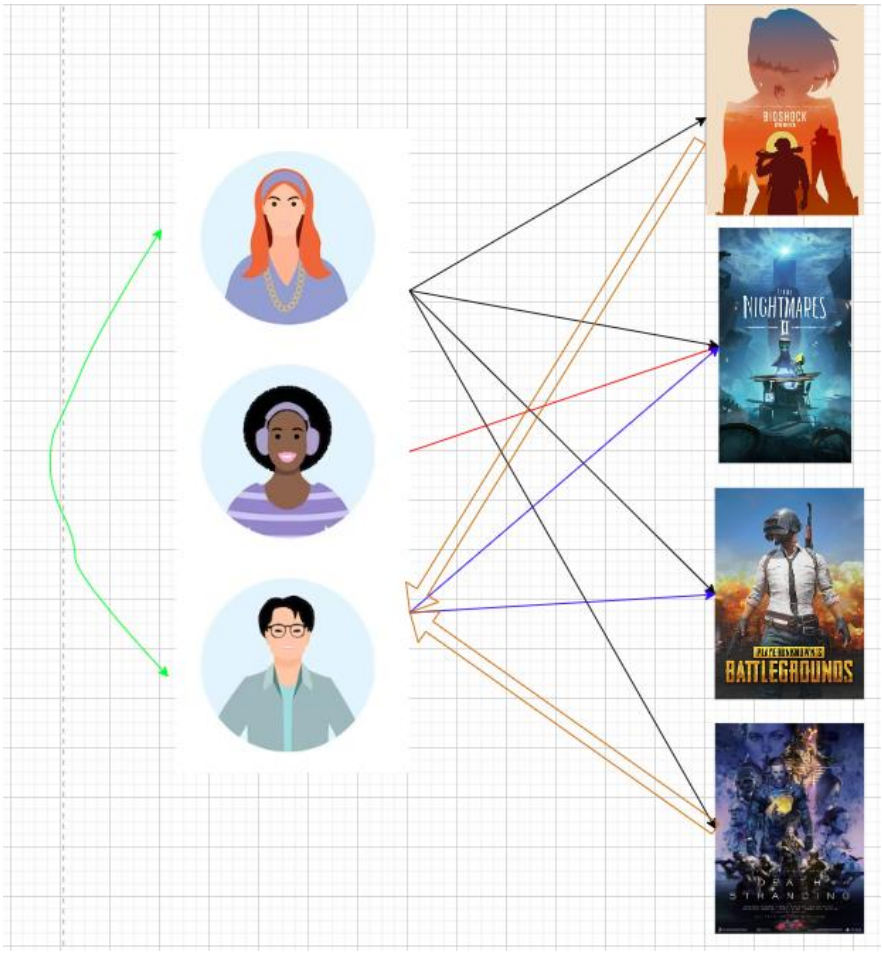

Рисунок 2.1 – User-based колаборативна фільтрація

Колаборативна фільтрація на основі користувачів має деякі проблеми. У цій системі кожен рядок матриці є користувачем. Тому порівняння та пошук подібності між ними є обчислювально важким і витрачає занадто багато обчислювальних ресурсів. Крім того, звички людей можна змінити.

Щоб вирішити ці проблеми, проаналізуємо іншу систему рекомендацій, якою є колаборативна фільтрація на основі елементів:

Розраховується подібність елементів у матриці користувача та елемента. Наприклад, припустимо ,що в нас є три користувача та ігри(«гра1», «гра2», «гра3», «гра4»). Тут єдина відмінність з колаборативною фільтрацією на основі користувачів полягає в тому, що ми бачимо схожі ігри, а не схожих користувачів. Тобто, як видно з рис.2.2 усі користувачі оцінили «гра3», лише перший користувач оцінив «гра4» і разом з другим користувачем оцінив «гра1». Отже, «гра2» була не оцінена користувачами, тому система не може рекомендувати її

третьому користувачеві. На рис2.2 позначено схожі ігри стрілкою, а саме «гра1» та «гра4»,а це означає, що третій користувач отримає «гра1» як рекомендацію.

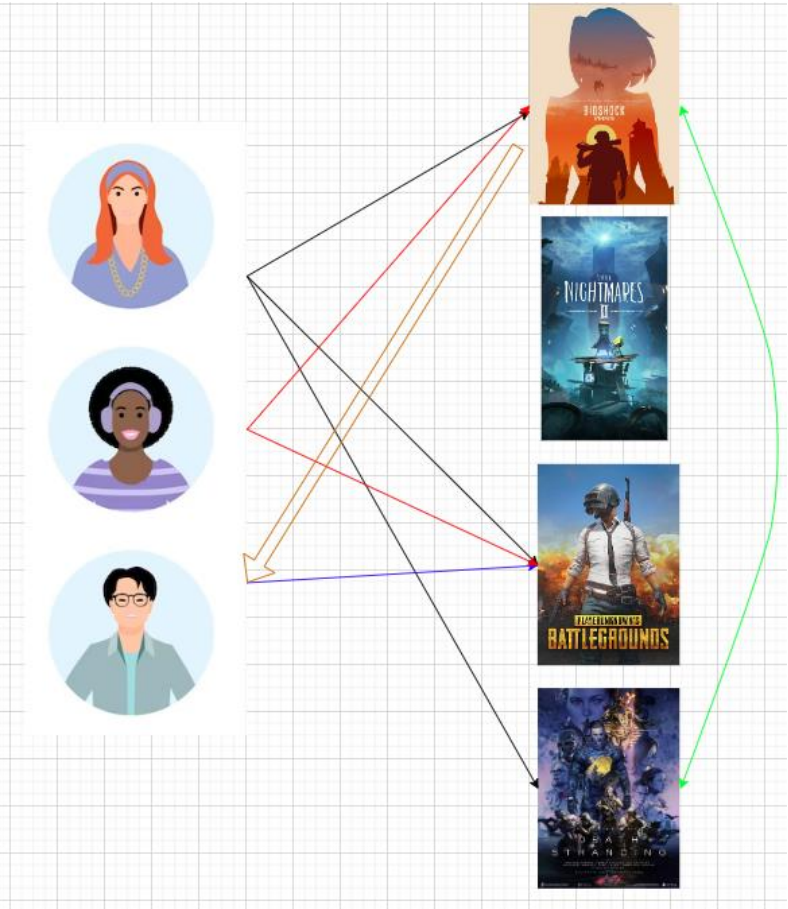

Рисунок 2.2 – Item-based колаборативна фільтрація

У загальних системах рекомендацій використовується колаборативна фільтрація на основі елементів. Колаборативна фільтрація на основі елементів покращена, щоб вирішити проблему колаборативної фільтрації на основі вподобань користувача. Оскільки розум і звички людей можуть змінюватися, а предмети не змінюються.

2.1.1 Переваги та проблеми

Переваги колаборативної фільтрації User-based:

- Враховує індивідуальні вподобання та поведінку користувачів, що дає можливість зробити більш персоналізовані рекомендації.
- Має можливість працювати з новими об’єктами, що не були розглянуті системою раніше.

- Не потребує додаткової інформації про об’єкти.

Переваги колаборативної фільтрації Item-based:

- Має можливість працювати з великими обсягами даних та великою кількістю об’єктів.
- Більш простий алгоритм, що дає можливість швидкої обробки даних та надання рекомендацій.
- Не потребує врахування поведінки користувачів, що робить його менш залежним від індивідуальних вподобань та поведінки.

Для реалізації програмного модуля буде реалізовано саме колаборативну фільтрацію на основі елементів (User -based).

Незважаючи на те, що РС надають багато переваг для користувачів, РС також можуть стикатися з рядом проблем. Ось декілька з них:

1. Проблема холодного старту. Коли рекомендаційна система починає роботу з нуля, у неї немає достатньої кількості даних про користувачів та їхній профіль, щоб надавати точні рекомендації.
2. Якщо рекомендаційна система отримує велику кількість даних, можуть виникнути проблеми з фільтруванням цих даних та наданням коректних рекомендацій.
3. Іноді рекомендаційна система може надавати рекомендації тільки найпопулярніших продуктів або послуг, ігноруючи менш популярні, але можливо більш підходящі для конкретного користувача.
4. Рекомендаційні системи ґрунтуються на даних про користувачів і товари. Якщо ці дані мають помилки, пропуски або неповноту, це може призвести до неточних або неправильних рекомендацій.
5. Рекомендаційні системи можуть використовувати особисту інформацію користувачів для створення рекомендацій, що може порушувати їх приватність та створювати проблеми з етикою. Наприклад, системи можуть використовувати дані про політичні

переконання або релігійні переконання користувачів, що може бути неприйнятним для багатьох користувачів.

6. Рекомендаційні системи можуть мати обмежену здатність до персоналізації, що означає, що вони можуть надавати рекомендації, які не враховують унікальних інтересів або потреб користувачів.

Проаналізувавши усі проблеми РС можна сказати, що одним з головних недоліків колаборативної фільтрації є проблема "холодного старту".[9] Це означає, що система не може зробити рекомендації для нових користувачів або для продуктів, з якими не було достатньо взаємодій. Це обмеження стає особливо проблематичним в тому випадку, якщо в базі даних мало даних про взаємодії користувачів з продуктами. У такому випадку може бути важко створити рекомендації, які будуть корисними для користувачів.

Для уникнення цього недоліку рекомендації будуть надаватись лише після того ,як користувач оцінить декілька ігор, наприклад він оцінив 5 ігор і отримав 3 гри рекомендовані системою.

У процесі роботи програмного модулю користувачеві спочатку надається набір рекомендацій. Після цього, розробники аналізують зворотний зв'язок користувача з метою оцінки відповідності рекомендацій його вподобанням та оцінок інших користувачів. Після цього знову пропонуються рекомендації користувачеві.

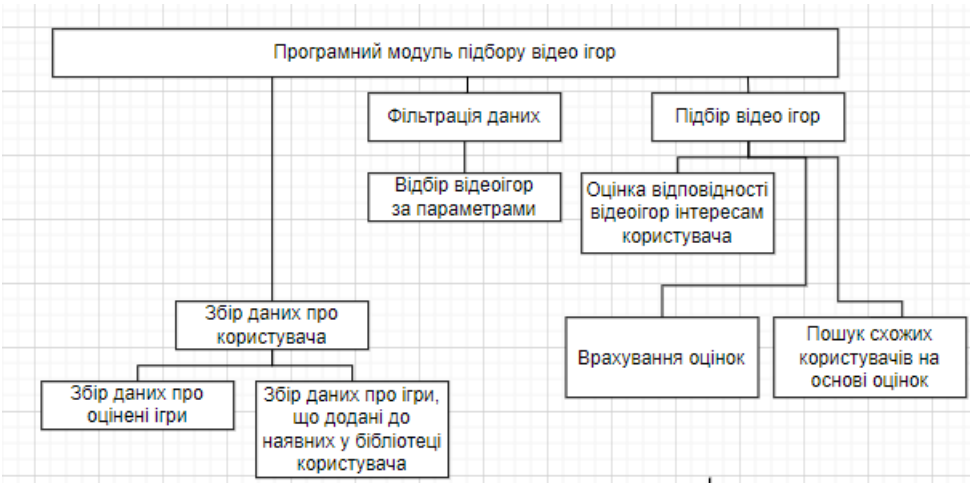

Рисунок 2.3 – Дерево функцій

На основі побудованого дерева функцій та контекстної діаграми проведена декомпозиція контекстної діаграми діяльності застосунку. Цей процес включає розбиття системи на більш малий рівень абстракції, розподіл функцій та діяльностей між різними компонентами та модулями системи.

Декомпозиція контекстної діаграми діяльності програмного модулю дозволяє розширити розуміння системи та показати більш детальний опис діяльностей та їх взаємозв'язків. Кожен компонент системи може бути розглянутий як окремий модуль, який виконує певну функцію або набір дій.

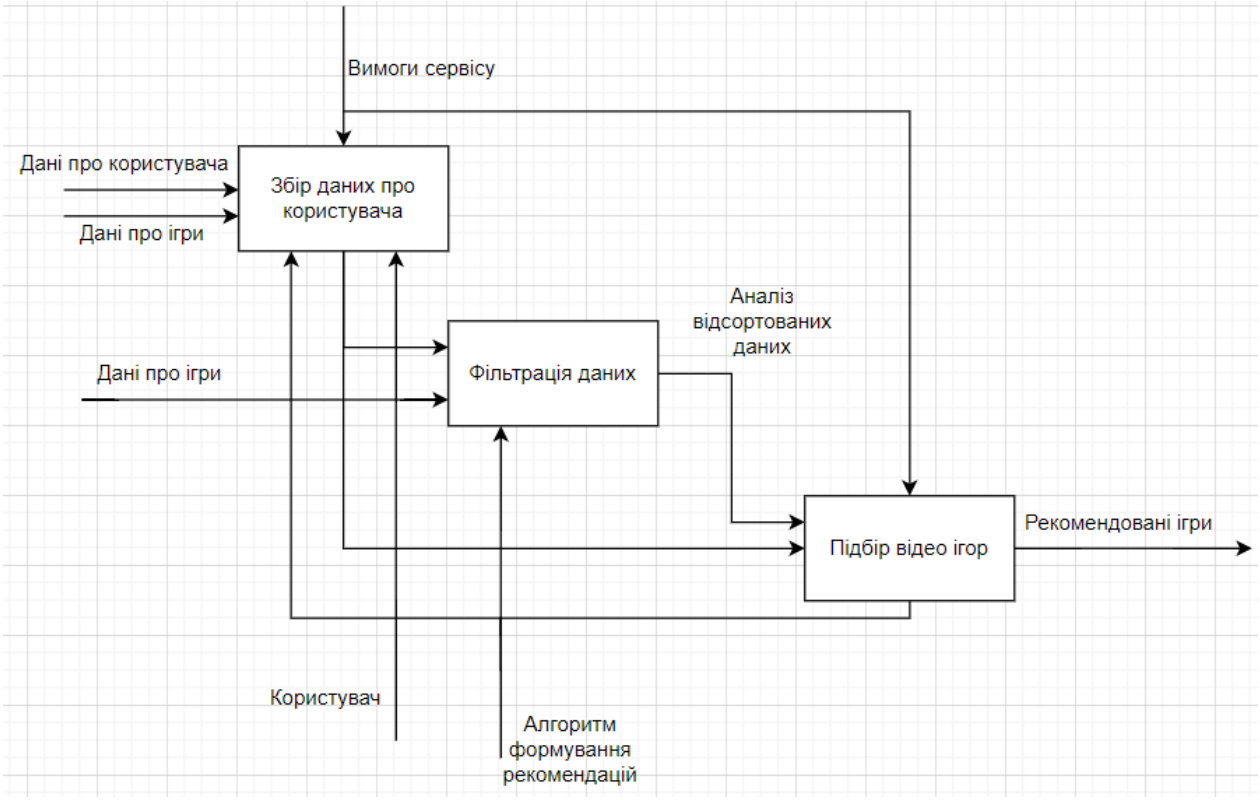

Рисунок 2.3 – Декомпозиція контекстної діаграми діяльності

Ця система призначена для забезпечення користувачів найбільш точними рекомендаціями, що відповідають їхнім потребам. Головна мета полягає у максимальному задоволенні потреб користувачів шляхом рекомендацій, що враховують вподобання користувача та оцінки інших користувачів.

2.2 Опис методів для реалізації

Алгоритм колаборативної фільтрації на основі користувача (User-Based Collaborative Filtering) використовує інформацію про зацікавленості користувачів у певних предметах для рекомендації інших предметів, які можуть зацікавити цих користувачів зі схожими інтересами. Основна ідея алгоритму полягає у наступних кроках:

1. Побудова матриці користувач-предмет: У цьому кроці створюється матриця, де рядки відповідають користувачам, а стовпці - предметам. Значення в матриці вказують на зацікавленість користувача у певному предметі. Наприклад, якщо користувач А купив яблука, банани та груші, а користувач Б купив банани та кавуни, відповідні елементи матриці будуть мати значення 1.

2. Обчислення схожості між користувачами: В цьому кроці обчислюється схожість між користувачами за допомогою метрики подібності, наприклад, косинусної подібності. Це дозволяє знайти набір схожих користувачів для кожного цільового користувача.

3. Прогнозування зацікавленості: За допомогою методу K Nearest Neighbors (KNN) або аналогічного алгоритму обираються K найбільш схожих користувачів для цільового користувача. Зацікавленість цільового користувача в невідомих предметах прогнозується шляхом врахування зацікавленості цих схожих користувачів у цих предметах.

4. Генерація рекомендацій: На основі прогнозованих значень зацікавленості генеруються рекомендації для цільового користувача. Це може бути список найкращих рекомендованих предметів або рейтинги рекомендованих предметів.

Це опис базового алгоритму колаборативної фільтрації на основі користувачів. У реальних системах можуть бути використані різні покращення і оптимізації, такі як врахування ваги зацікавленості користувача у відповідні предмети або використання інших метрик.

Першим кроком у цьому процесі є створення моделі, яка базується на визначенні подібності між усіма парами товарів. Для цього можна

використовувати різні методи визначення подібності. Один з найбільш поширених методів - це косинусна подібність.

Формула косинусної подібності:

$$Similarity(\overrightarrow{A}, \overrightarrow{B}) = \frac{\vec{A} * \vec{B}}{\|\vec{A}\| * \|\vec{B}\|} \quad (2.1)$$

Другий етап передбачає виконання системи рекомендацій. Він використовує елементи (вже оцінені користувачем), які найбільше схожі на відсутній елемент для створення рейтингу. Тому прогнози будуть створюватись на основі рейтингів подібних продуктів. Обчислюється це за допомогою формули, яка обчислює рейтинг для певного продукту за допомогою зваженої суми рейтингів інших подібних продуктів.

2.2.1 K - найближчих сусідів

Завдання класифікації вирішується за допомогою аналітичних моделей, які називаються класифікаторами. Класифікація об'єкта означає подання набору його ознак у вигляді вектора на вхід класифікатору, що повинен призначити йому мітку або номер класу. На сьогоднішній день існує велика кількість різних видів класифікаторів, які будуються з використанням як статистичних методів (логістична регресія, дискримінантний аналіз), так і методів машинного навчання (нейронні мережі, дерева рішень та інші).

Необхідність використання різноманітних методів класифікації в аналізі даних обумовлена тим, що різні завдання можуть мати свої особливості, пов'язані зі способом подання даних, їх кількістю та якістю, що вимагає вибору відповідного класифікатора. Тому вибір класифікатора, який відповідає особливостям аналізованого завдання, є важливим фактором для отримання правильного рішення.

Класифікація є частиною інтелектуальних технологій аналізу даних, оскільки в повсякденному житті людина, зіставляючи нові об'єкти навколишнього світу з уже відомими, оцінює ступінь їх подібності. На основі цієї оцінки об'єкт асоціюється з певною групою або класом. Таким чином,

класифікація є найбільш "природним" способом отримання знань про процеси та явища, що відбуваються у навколишньому світі.

З урахуванням сказаного можна припустити, що методи класифікації будуть використовувати формалізоване поняття "подібності", міру якої можна оцінити за допомогою певної функції. У статистичних методах аналізу, мірою подібності є можливість належати об'єкта до певного класу, що оцінюється для кожного класу. Потім вибирається той клас, для якого ця можливість найбільша.

У метричних методах мірою подібності є відстань (наприклад, евклідова) у векторному просторі, де кожен об'єкт представлений своїм вектором ознак. Логіка проста: новий об'єкт ймовірніше належить до того ж класу, що й більшість його найближчих сусідів. Метричні методи часто використовуються для побудови класифікаторів на основі машинного навчання.

Статистичні методи мають перевагу у хорошій математичній обґрунтованості, але недоліком є низька пояснювальна здатність. Використання ймовірнісних оцінок дозволяє точно передбачити клас об'єкта, але не пояснити, чому саме такий результат. Тому результати статистичних методів класифікації можуть бути складними для розуміння та інтерпретації.

Метричні методи мають характер евристичних рішень і можуть призводити до неточних та неоднозначних результатів, але це вважається прийнятним у більшості практичних випадків. Однак вони мають високу пояснювальну здатність, і їх результати легко інтерпретувати. В простих випадках можна застосовувати правило: об'єкт належить до того ж класу, що й більшість його найближчих сусідів.

Один з типових методів класифікації, що використовує цю логіку, - це метод k-найближчих сусідів (KNN). Цей метод належить до класу непараметричних, тобто не вимагає припущень про статистичний розподіл навчальної вибірки. Тому класифікаційні моделі, побудовані за допомогою KNN, також є непараметричними. Це означає, що структура моделі визначається даними, а не жорстко наперед. [11]

Оскільки ознаки, за якими проводиться класифікація, можуть мати різну природу і діапазони значень, корисно нормалізувати навчальні дані для поліпшення результатів класифікації.

Алгорит KNN:

Нехай є набір даних, що складається з n спостережень X_i(i=1,...,n), кожному з яких заданий клас C_i (j=1,...,m). Тоді на його основі може бути сформована навчальна множина, всі приклади якої являють собою пари X_i, C_i.

Алгоритм KNN можна розділити на дві прості фази: навчання та класифікації. Під час навчання алгоритм просто запам'ятовує вектори ознак спостережень та його мітки класів (тобто. приклади). Також задається параметр алгоритму k, який задає кількість "сусідів", які будуть використовуватися при класифікації.

На фазі класифікації пред'являється новий об'єкт, для якого мітка класу не задана. Він визначають k найближчих (у сенсі деякої метрики) попередньо класифікованих спостережень. Потім вибирається клас, якому належить більшість з k найближчих прикладів-сусідів, і до цього класу відноситься об'єкт, що класифікується.

Пояснимо роботу алгоритму за допомогою рисунка 2.3.

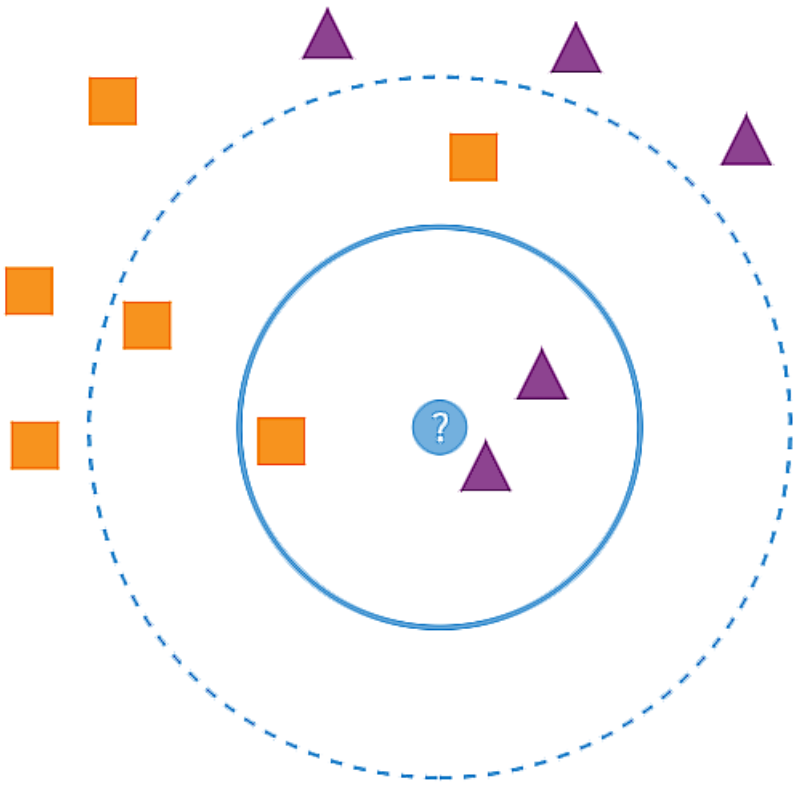

Рисунок 2.4 – Принцип роботи алгоритму KNN

Колом представлено об'єкт, який потрібно класифікувати, відносячи до одного з двох класів "трикутники" та "квадрати". Якщо вибрати k=3, то з трьох найближчих об'єктів два виявляться трикутниками і один квадратом. Відтак новому об'єкту буде надано клас «трикутник». Якщо задати k=5, то з п'яти «сусідів» два будуть «трикутники» і три «квадрати», в результаті об'єкт, що класифікується, буде розпізнаний як «квадрат».

Вибір параметра k є важливим для коректних результатів класифікації. Якщо значення параметра мало, виникає ефект перенавчання, коли рішення щодо класифікації приймається з урахуванням малого числа прикладів і має низьку значимість. Це схоже на перенавчання у деревах рішень, коли в них багато правил, що належать до невеликої кількості прикладів. Якщо встановити k=1, то алгоритм просто присвоювати будь-якому новому спостереженню мітку класу найближчого об'єкта.

Крім цього, слід враховувати, що використання невеликих значень k збільшує вплив шумів на результати класифікації, коли невеликі зміни даних

приводять до великих змін у результатах класифікації. Але при цьому межі класів виявляються більш вираженими (клас при голосуванні перемагає з великим рахунком).

Навпаки, якщо значення параметра занадто велике, то процесі класифікації бере участь багато об'єктів, які стосуються різних класів. Така класифікація виявляється дуже грубою і погано відбиває локальні особливості набору даних. Таким чином, вибір параметра k є компромісом між точністю та узагальнюючою здатністю моделі.

При великих значеннях параметра k зменшується шумування результатів класифікації, але знижується вираженість меж класів.

У завданнях бінарної класифікації буває доцільно вибрати k як непарне число, оскільки це дозволяє уникнути рівності «голосів» щодо класу нового спостереження.

У найпростішому випадку клас нового об'єкта може бути визначений простим вибором класу, що найчастіше зустрічається серед k прикладів. Однак на практиці це не завжди вдале рішення, наприклад, якщо частота появи для двох або більше класів виявляється однаковою. Крім цього розумно припустити, що не всі учні мають однакову значущість для визначення класу. У цьому випадку використовують деяку функцію, за допомогою якої визначається клас, що називається функцією поєднання (combination function).

У звичайному випадку використовують так зване просте невиважене голосування (simple unweighted voting). У цьому передбачається, що це k прикладів мають однакове право «голосу» незалежно від відстань до класифікованого об'єкта.

Однак, логічно припустити, що чим далі приклад розташований від об'єкта, що класифікується, в просторі ознак, тим нижче його значимість для визначення класу. Тому для покращення результатів класифікації вводять зважування прикладів залежно від їхньої віддаленості. І тут використовують зважене голосування (weighted voting).

В основі ідеї зваженого голосування лежить введення «штрафу» для класу, залежно від того, наскільки приклади, що відносяться до нього, віддалені від класифікованого об'єкта. Такий «штраф» представляється як сума величин, обернених квадрату відстаней від прикладу j-го класу до об'єкта, що класифікується (часто дане значення називають показником близькості):

$$Q_j = \sum_{j=1}^{n_j} \frac{1}{D^2(x, a_{ij})}, \qquad (2.2)$$

де D - оператор обчислення відстані, x - вектор ознак об'єкта, що класифікується, a_{ij} - і-й приклад j-го класу. Таким чином, "перемагає" той клас, для якого величина Q_j виявиться найбільшою. При цьому також знижується ймовірність того, що класи отримають однакову кількість голосів.

Якщо значення ознак безперервні, то як відстань між об'єктами зазвичай використовується відстань Евкліда, а якщо категоріальні, то може використовуватися відстань Хеммінга.

Алгоритм KNN є чутливим до дисбалансу класів у навчальних даних: алгоритм «схильний» до зміщення рішення у бік домінуючого класу, оскільки об'єкти, що належать до нього, просто частіше потрапляють до числа найближчих сусідів. Одним із способів вирішення цієї проблеми є застосування різних способів зважування під час «голосування».

Слід зазначити, що ставлення сусідства перестав бути коммутативним, тобто. якщо для запису B найближчим сусідом є запис A, це означає, що B– найближчий сусід A. Цю ситуацію представлено на рисунку 2.4.

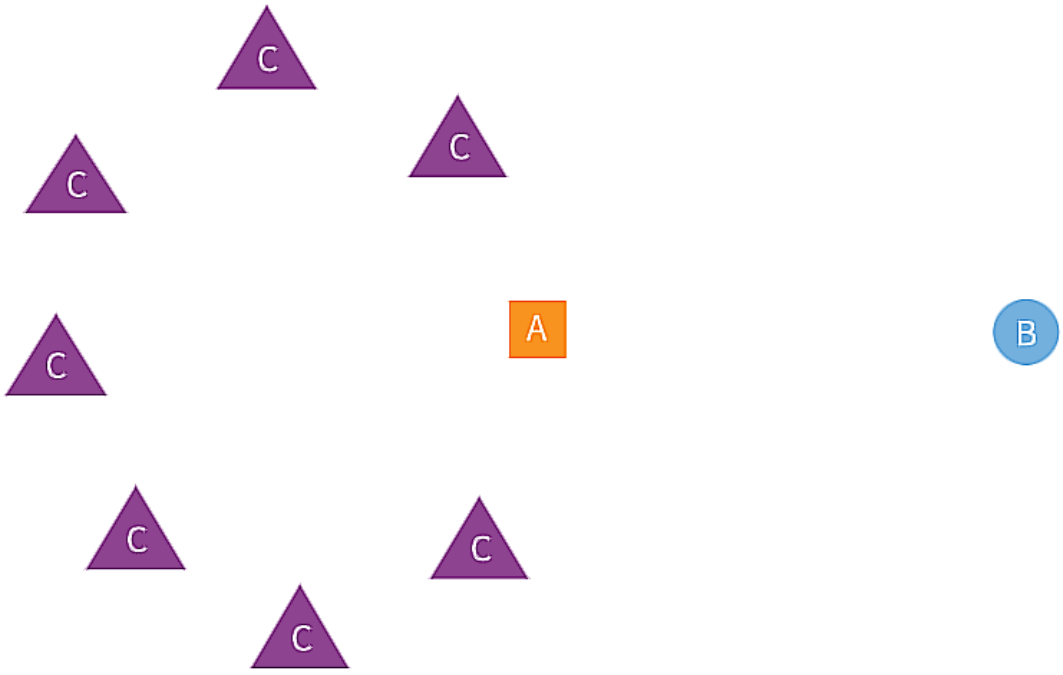

Рисунок 2.5 - Зворотне сусідство

При k=1 найближчої точки B буде точка A, а A — точка C. Навіть зі збільшенням коефіцієнта до k=7, точка B як і не входитиме до сусідів A.

Ще однією проблемою алгоритму KNN, характерною, втім, і більшості методів класифікації, є різна значимість ознак з погляду визначення класу об'єктів. Врахування фактора значущості ознак в алгоритмі може дозволити підвищити точність класифікації.

Для цього аналітик або експерт на основі суб'єктивної або деякої формальної оцінки може задати рівень значущості ознаки, висловивши його за допомогою числового коефіцієнта (позначимо його s від англ significance — значимість), який враховується при обчисленні відстані між прикладами та об'єктом, що класифікується:

$$D_E = \sqrt{s_1(x_1 - a_1)^2 + s_2(x_2 - a_2)^2 + \cdots + s_p(x_p - a_p)^2} \quad (2.3)$$

де s_i (i = 1..p) - Коефіцієнт значимості для i-го ознаки, p - кількість ознак вихідного набору даних. Такий прийом називається розтягуванням осей і він дозволяє збільшити або зменшити внесок ознаки у обчислення відстані від

прикладу до об'єкта, що класифікується. Якщо s_i >1, то завдяки відповідному ознакою відстань між прикладом об'єктом, що класифікується, зростає і внесок у визначення класу падає, а якщо s_i <1, то навпаки.

2.3 Оцінка якості рекомендаційних систем

Для оцінки рекомендаційних систем важливо правильно обрати метрики оцінки якості, оскільки неправильна система оцінювання якості може призвести до суттєвої переоцінки або недооцінки справжньої точності алгоритму чи моделі. Можна застосовувати різні набори метрик:

- метрики точності(MAE, MSE, RMSE);
- метрики прийняття рішень (ROC, Precision/recall);
- метрики оцінки ранжування результатів;
- бізнес-метрики (збільшення продаж, конверсії).

Дослідження користувачів — це сеанс тестування, під час якого користувачів просять виконати певні завдання в системі та надати відгук до та після взаємодії з нею.

Головна мета рекомендаційних систем полягає в досягненні покриття, точності, різноманітності, неочікуваності, новизни, надійності та масштабованості. Щоб визначити, наскільки успішно вдається досягти цих метрик, можна використовувати метрики оцінки.

2.3.1 Метрики точності прогнозування

Показники точності прогнозування є критичними для оцінки ефективності рекомендаційних систем. Ці метрики відображають рівень точності, з якою система може передбачити, які предмети сподобаються користувачам. Якщо прогнози не точні, то рекомендації можуть бути некорисними для користувачів і не мати бажаного впливу на бізнес. [11]

Середня абсолютна похибка (MAE) - це показник, що вимірює середню абсолютну різницю між прогнозованими та фактичними значеннями. В

контексті рекомендаційних систем, MAE обчислюється як середня абсолютна різниця між прогнозованим рейтингом користувача та його фактичним рейтингом. Чим нижче значення MAE, тим краще точність прогнозування системи. [11]

$$Mean\ Absolute\ Error(MAE) = \frac{Sum\ of\ Absolute\ Errors}{Nimber\ of\ Predictions} = \frac{\sum_{i=1}^{n}|y_i - \widehat{y_1}|}{n} \qquad (2.4)$$

Окрім MAE, існують інші показники точності прогнозування, такі як середньоквадратична похибка (MSE) та коренева середньоквадратична похибка (RMSE). Ці показники також дозволяють виміряти точність прогнозів, але зазвичай використовуються в різних контекстах. [11]

Наприклад, MSE та RMSE можуть бути корисними, коли важлива точність прогнозів на певних значеннях рейтингу (наприклад, точність прогнозів для дуже популярних або дуже не популярних предметів). Однак, MAE є більш зручним показником, якщо розміри помилок є важливими для розуміння користувачів та менеджерів проекту. [11]

$$Mean\ Squared\ Error(MSE) = \frac{Sum\ of\ Squared\ Errors}{Nimber\ of\ Predictions} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2 \qquad (2.5)$$

$$RootMeanSquaredError(RMSE) = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2} \qquad (2.6)$$

2.3.2 Метрики прийняття рішень

Точність підтримки прийняття рішень використовується для оцінки ефективності рекомендаційних систем з бінарними значеннями, такими як перегляд товару, додавання в список бажань або лайки. У таких системах продукти, які відповідають вподобанням користувача, відносяться до позитивного класу, тоді як ті, що не відповідають, відносяться до негативного класу. [11]

Кожен прогноз може бути помилковим або істинним, тому є чотири можливих випадки: істинно позитивний (коли правильно визначено, що продукт подобається користувачеві), істинно негативний (коли правильно визначено, що

продукт не подобається користувачеві), хибно позитивний (коли помилково визначено, що продукт сподобається користувачеві) та хибно негативний (коли помилково визначено, що продукт не сподобається користувачеві).

Помилковий негативний клас відповідає помилці першого роду, тоді як помилковий позитивний клас - помилці другого роду. В залежності від кількості рекомендованих товарів, кількість помилково визначених товарів у класах false positive та false negative може змінюватись.

Таблиця 2.1 – Матриця помилок

	Прогнозований клас		
Дійсний клас		+	-
	+	TP (True positive)	FN (False negative)
	-	FP (False positive)	TN (True negative)

Ще один спосіб виразити збалансованість цієї взаємодії - це крива precision/recall. Зазвичай, чим більше істинних позитивів ви визначаєте, тим більше помилкових результатів супроводжує це. У контексті машинного навчання, це графічний інструмент для знаходження найкращої балансу між точністю та повнотою моделі. Криву компромісу можна визначити за допомогою метрик precision/recall, які відображають, як класифікатор визначає позитивні та негативні екземпляри. [11]

Метрика precision вимірює, яка частка прикладів, визначених класифікатором як позитивні, дійсно належить до позитивного класу. Це особливо важливо в задачах, де нам важливо, щоб модель правильно класифікувала позитивні приклади, тоді як recall (повнота) відображає частку дійсних позитивних відповідей, які були виявлені алгоритмом з усіх наявних позитивних відповідей. Тобто, це показник того, наскільки добре алгоритм здатен знайти всі позитивні відповіді. [10]

Другим методом для побудови кривої компромісу є ROC-крива, яка відображає залежність між показниками True Positive Rate (TPR) та False Positive Rate (FPR). TPR вимірює відношення кількості правильно визначених позитивних прикладів до загальної кількості позитивних прикладів(2.7), тоді як FPR вимірює відношення кількості помилково визначених позитивних прикладів до загальної кількості негативних прикладів(2.8). Чим ближче ROC-крива до верхнього лівого кута, тим більш висока ефективність моделі. [15]

$$Precision = \frac{TP}{TP + FP} \quad (2.7)$$

$$Recall = \frac{TP}{TP + FN} \quad (2.8)$$

2.4 Опис бази даних

Розробка бази даних починається з концептуального проектування, яке є першим етапом у створенні бази даних. На цьому етапі визначаються основні сутності та зв'язки, які існують в системі, що буде використовуватися для організації та збереження даних.

Сутності представляють реальні або абстрактні об'єкти, які важливі для системи, інформація про які повинна бути збережена. У програмному модулі, що розроблюється, сутностями можуть бути "Користувач", "Гра", "Наявність гри". Кожна сутність має свої атрибути, які визначають характеристики цих сутностей. Наприклад, для сутності "Користувач" атрибутами будуть: "Ім’я", "Кімната(пароль)" і "IP-адреса".

Зв'язки визначають зв'язки між різними сутностями. Вони показують, як інформація взаємодіє та пов'язана між сутностями. Наприклад, у системі є зв'язок між сутностями "Користувач" і "Наявність гри", що показує, що декілька ігор можуть належати одному користувачеві.

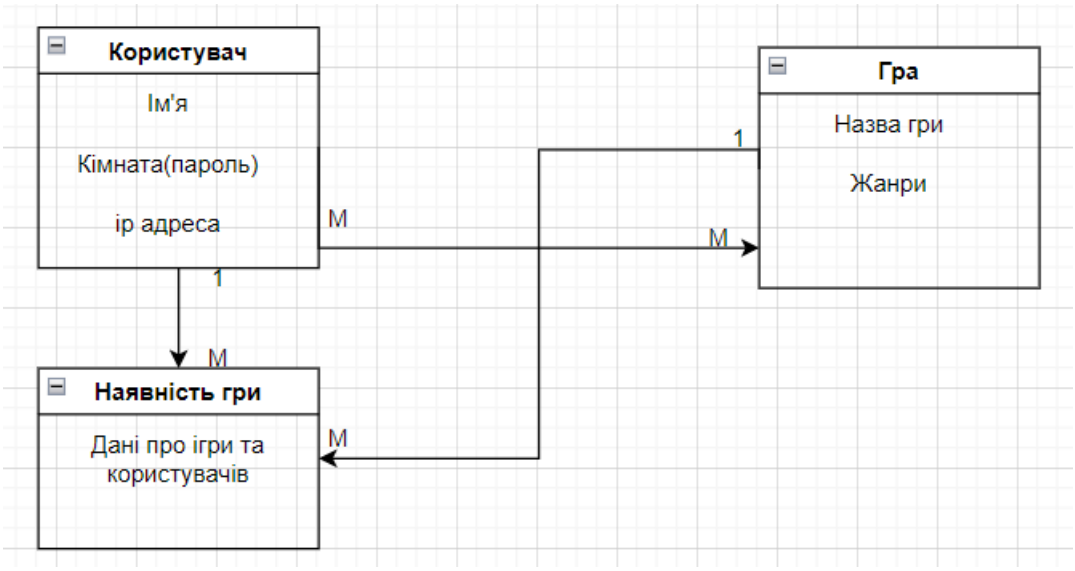

Рисунок 2.6 – Концептуальна модель бази даних

Під час концептуального проектування також визначаються правила і обмеження, що стосуються даних. Наприклад, можуть бути встановлені правила щодо унікальності значень певних атрибутів або обмеження їх значень до певного діапазону.

Концептуальне проектування бази даних вимагає вивчення вимог до системи, взаємодії з користувачами та аналізу потреб і процесів, які будуть використовуватися в системі. В результаті цього процесу створюється модель, яка відображає загальну структуру бази даних і формує основу для подальшої розробки та реалізації фізичної моделі бази даних.

Після побудови концептуальної моделі баз даних, наступним кроком є побудова логічної моделі баз даних. Логічна модель визначає структуру бази даних без прив'язки до конкретної системи управління базами даних (СУБД) і використовує поняття зв'язків, сутностей, атрибутів та правил відносин для представлення даних.

На рисунку 2.7 зображена логічна модель бази даних. Логічна модель надає абстрактний опис структури бази даних, де визначаються сутності, їх атрибути та відношення між ними. Логічна модель бази даних може бути використана для подальшої розробки фізичної моделі.

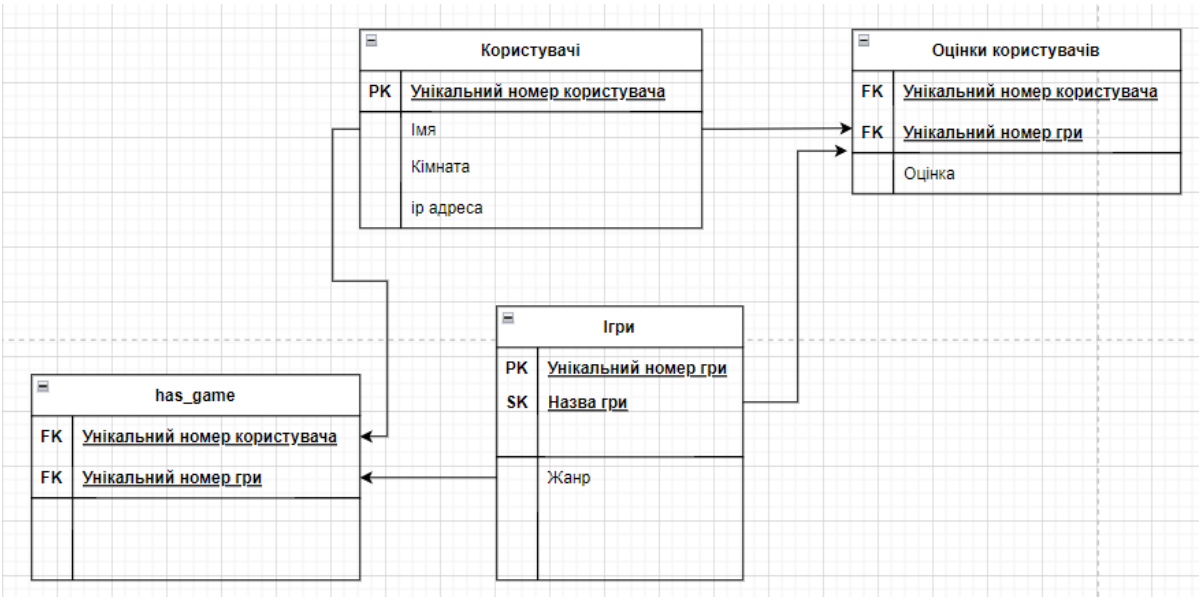

Рисунок 2.7 – Логічна модель бази даних

Таким чином, побудова логічної моделі бази даних є важливим кроком у процесі розробки бази даних, що дозволяє визначити структуру та відношення між даними, незалежно від конкретної СУБД.

На рисунку 2.8 зображено фізичну модель бази даних, яка визначає всі моделі реляційних даних та об'єкти бази даних. Фізична модель даних створюється з використанням рідної мови бази даних системи управління базами даних (СУБД). Це означає, що вона відображає структуру та організацію даних в самому СУБД, використовуючи його специфічні концепції та властивості.

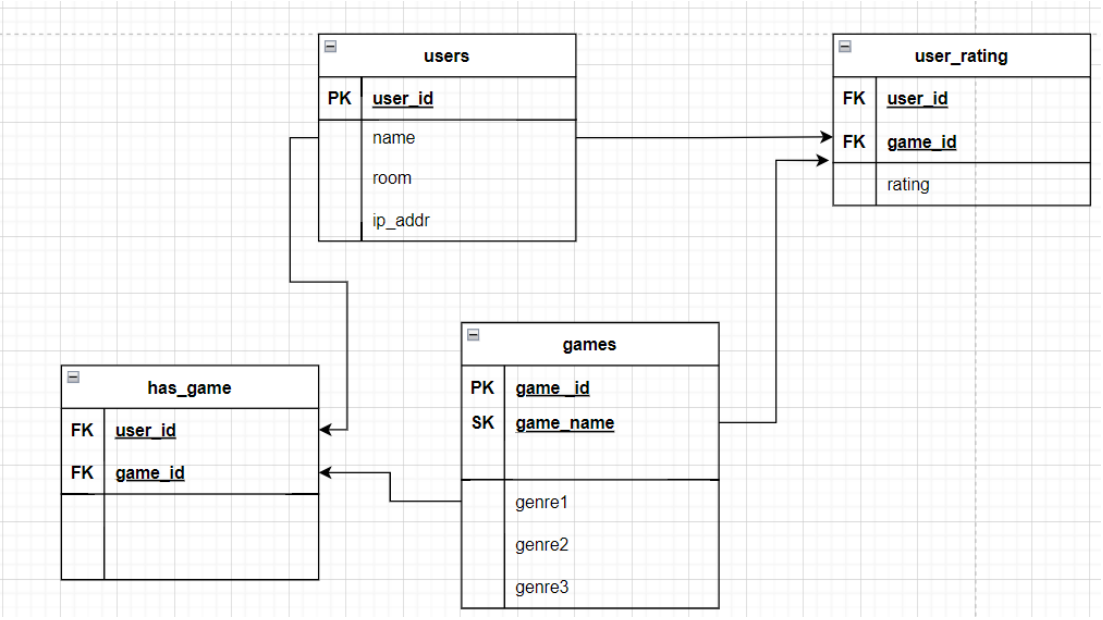

Рисунок 2.8 – Фізична модель бази даних

Таблиці з описом полів бази даних надають детальний огляд структури бази даних і допомагають розуміти призначення та властивості кожного поля, що використовується для збереження даних.

Опис призначення та властивостей полів бази даних наведено у таблицях нижче.

Таблиця 2.2 – «users» - користувачі

	Тип	Розмір	Первинний ключ	Зовнішній ключ	Обмеження
user_id	int	11	+	-	Унікальний
name	varchar	32	-	-	-
room	varchar	5	-	-	-
ip_add	varchar	32	-	-	-

Таблиця містить наступні поля: "id користувача", "ім'я", "кімната (тобто пароль)" та "ip-адреса".

Поле "id користувача" є унікальним для кожного користувача і служить ідентифікатором, який відрізняє одного користувача від іншого. Це означає, що в таблиці не може бути двох користувачів з однаковим id.

З метою підвищення безпеки було введено поле "ip-адреса". Ip-адреса вказує на унікальний ідентифікатор комп'ютера або пристрою в мережі Інтернет. Використання ip-адреси дозволяє запобігти несанкціонованому доступу до користувача навіть у випадку, якщо хтось знає пароль користувача.

Наприклад, якщо інший користувач дізнається пароль до аккаунту, але намагається зайти з іншої ip-адреси, система створить новий з іншою із адресою. Це означає, що навіть якщо інший користувач має правильний логін та пароль, він не зможе отримати доступ до чужого облікового запису, якщо його ip-адреса не співпадає з зареєстрованою в системі.

Такий підхід до безпеки зменшує ризик несанкціонованого доступу до користувачів, оскільки вимагає від зловмисників не лише знання паролю, але й використання правильної ip-адреси, щоб успішно авторизуватися в системі.

Таблиця 2.3 – «games» - список ігор

	Тип	Розмір	Первинний ключ	Зовнішній ключ	Обмеження
game_id	int	11	+	-	Унікальний
game_name	varchar	128	-	+	Унікальний
genre1	varchar	24	-	-	-
genre2	varchar	24	-	-	-
genre3	varchar	24	-	-	-

Поля "game_id" та "game_name" є унікальними для кожної ,адже вони служать ідентифікаторами, які відрізняють ігри. Це означає, що не може бути двох однакових ігор, тобто навіть якщо спробувати додати ігри з різними id , але з однаковою назвою, то можна побачити повідомлення, що така гра вже існує.

Також в ігор є декілька полів з жанрами, їх всього три, щоб була можливість додати декілька , наприклад, є рольова гра у жанрі жахів, отже можемо записати для такої гри два жанри «Рольова» та «Жахи».

Таблиця 2.4 – «has_game» - користувачі, які мають певну гру

	Тип	Розмір	Первинний ключ	Зовнішній ключ	Обмеження
user_id	int	11	-	+	Унікальний
game_id	int	11	-	+	Унікальний

Таблиця бази даних "has_game" містить інформацію про користувачів, які мають певну гру. Вона містить наступні поля:

1. user_id: Це поле є зовнішнім ключем, яке посилається на поле "user_id" з іншої таблиці, що містить інформацію про користувачів. Це поле вказує на ідентифікатор користувача, який має певну гру. Зв'язок з таблицею користувачів дозволяє встановити зв'язок між користувачем і грою.
2. game_id: Це поле також є зовнішнім ключем, яке посилається на поле "game_id" з таблиці "game". Воно вказує на ідентифікатор гри, яку має користувач. Зв'язок з таблицею ігор дозволяє встановити зв'язок між грою і користувачем.

Обидва поля, user_id і game_id, використовуються як зовнішні ключі, що посилаються на відповідні первинні ключі в інших таблицях. Це дозволяє створити зв'язок між таблицею "has_game" і таблицями користувачів та ігор.

Крім того, таблиця "has_game" може мати обмеження на унікальність комбінації user_id і game_id, що означає, що один користувач не може мати повторні записи для однієї гри.

Ця таблиця дозволяє зберігати інформацію про зв'язок між користувачами і їх іграми, вказуючи, які користувачі мають доступ до певних ігор.

Таблиця 2.4 – «users_rating» - оцінки користувачів

	Тип	Розмір	Первинний ключ	Зовнішній ключ	Обмеження
user_id	int	11	-	+	Унікальний
game_id	int	11	-	+	Унікальний
rating	float		-	-	-

Таблиця бази даних "users_rating" представляє інформацію про оцінки користувачів для певних ігор. Вона містить наступні поля:

1. user_id: Це поле є зовнішнім ключем, яке посилається на поле "user_id" з іншої таблиці, що містить інформацію про користувачів. Воно ідентифікує користувача, який надав оцінку. Зв'язок з таблицею користувачів дозволяє встановити зв'язок між оцінкою і конкретним користувачем.
2. game_id: Це поле також є зовнішнім ключем, яке посилається на поле "game_id" з таблиці "game". Воно ідентифікує гру, для якої надана оцінка. Зв'язок з таблицею ігор дозволяє встановити зв'язок між оцінкою і конкретною грою.
3. rating: Це поле містить оцінку, яку користувач надав певній грі. Оцінка може бути числовим значенням типу float, що дозволяє використовувати десяткові значення для більш точного вираження рейтингу.

Обидва поля, user_id і game_id, використовуються як зовнішні ключі, що посилаються на відповідні первинні ключі в інших таблицях. Це дозволяє створити зв'язок між таблицею "users_rating" і таблицями користувачів та ігор.

Таблиця "users_rating" може мати обмеження на унікальність комбінації user_id і game_id, що означає, що один користувач може надати лише одну оцінку для однієї гри.

Ця таблиця дозволяє зберігати оцінки користувачів для різних ігор, зв'язуючи оцінку з конкретним користувачем і грою.

ВИСНОВКИ ДО РОЗДІЛУ 2

Колаборативна фільтрація є потужним підходом до побудови рекомендаційних систем, який базується на спільній інформації про поведінку користувачів. Вона дозволяє рекомендувати об'єкти користувачам на основі їхньої схожості з іншими користувачами. Переваги колаборативної фільтрації включають високу персоналізацію рекомендацій, здатність враховувати змінні вподобання користувачів та здатність працювати з новими об'єктами. Проте, є й деякі проблеми, зокрема, розрідженість даних, холодний старт для нових користувачів та об'єктів.

Метод k-найближчих сусідів (KNN) є одним з методів реалізації колаборативної фільтрації. Він використовує схожість між користувачами або об'єктами, щоб здійснювати рекомендації. KNN визначає k найближчих сусідів до даного користувача чи об'єкта і базується на їхніх вподобаннях для формування рекомендацій.

Були розроблені проектні рішення для реалізації модулю підбору відео ігор, проведено структурно-функціональний аналіз, розроблена концептуальна модель предметної області та фізична модель БД.

Для оцінки якості рекомендаційних систем використовуються метрики точності прогнозування. Ці метрики вимірюють наскільки точні прогнози системи в порівнянні з реальними вподобаннями користувачів.

РОЗДІЛ 3 ФУНКЦІОНАЛЬНИЙ АНАЛІЗ ТА ТЕСТУВАННЯ

3.1 Вибір програмного забезпечення

Для розробки Web-додатку використовують редактори коду, такі як Sublime Text, Atom, або інтегроване середовище розробки (IDE), такі як Eclipse або IntelliJ IDEA. Вибір конкретного інструменту залежить від особистих вподобань та потреб користувача.

Найпростішим інструментом для цієї задачі можна назвати Visual Studio Code від компанії Microsoft. Це програмне забезпечення є безкоштовним і його може завантажити будь який користувач, незалежно від платформи на якій здійснюється розробка продукту. Це може бути як операційна система Windows від Microsoft, так і будь яка Unix система.

За допомогою цієї програми можна легко редагувати код, або ж створювати документи HTML, CSS чи JavaScript. Основна версія програми є безкоштовною, та при бажанні можна придбати версію із розширеним функціоналом для створення комерційних продуктів чи використання віддаленого сховища.

Серед плюсів програмного забезпечення Visual Studio Code можна відзначити той факт, що воно легко встановлюється і не потребує значних ресурсів технічного засобу на якому буде використовуватись. Є версія не тільки для персональних комп’ютерів, але і для мобільних телефонів чи планшетів. Це дозволяє отримати можливість продовжувати розробку, незалежно від пристрою, часу, та місцезнаходження користувача.

Для даного проекту вистачить безкоштовної версії програмного забезпечення, тому що розробка буде виконуватись локально і розширений функціонал попросту не знадобиться.

Мовою програмування було обрано PHP, адже вона є широко використовуваною мовою програмування, особливо серед веб-розробників. Її походження пов'язане зі створенням Расмусом Лердорфом, який почав розробку PHP як набору інструментів для полегшення створення динамічних веб-сторінок.

Однак, з плином часу PHP перетворилася на загально призначену мову програмування.

Основне використання PHP полягає у розробці серверних додатків, зокрема генерації HTML-коду, який потім відображається веб-браузерами. PHP працює на більшості веб-серверів і здатна взаємодіяти з різноманітними базами даних. Вона має велику кількість вбудованих функцій і розширень, які полегшують роботу зі змінними, рядками, масивами, роботою з файлами та іншими завданнями, що виникають під час розробки веб-додатків.[20]

PHP також підтримує об'єктно-орієнтоване програмування, що дозволяє створювати класи, об'єкти та використовувати спадкування, поліморфізм та інші концепції ООП. Вона також підтримує розробку веб-сервісів, роботу з HTTP-запитами та відповідями, обробку форм, аутентифікацію користувачів та багато іншого. [19]

PHP має велику активну спільноту розробників, що забезпечує наявність багатьох корисних бібліотек, фреймворків та інструментів для прискорення процесу розробки. Вона підтримується на багатьох платформах і має хорошу сумісність з різними операційними системами та веб-серверами.

Загалом, PHP є потужним інструментом для розробки веб-додатків, який надає розробникам багато можливостей для створення функціональних та ефективних веб-рішень.[19]

У якості фреймворку для створення стилю веб-додатку було використано Bootstrap. Він є одним з найпопулярніших фреймворків для розробки веб-інтерфейсів, надає набір готових компонентів, стилів і JavaScript-плагінів, які допомагають швидко та легко створювати сучасні, адаптивні та привабливі веб-сайти. [22]

Основні переваги використання Bootstrap:

1. Адаптивний дизайн: Bootstrap надає готові класи і компоненти, що дозволяють створювати веб-сторінки, які добре виглядають і

працюють на різних пристроях і екранах, включаючи мобільні пристрої. [22]

2. Готові компоненти: Bootstrap містить багато готових компонентів, таких як кнопки, форми, навігація, каруселі, модальні вікна та інші. Це дозволяє ефективно будувати веб-інтерфейси без необхідності писати весь код з нуля. [22]
3. Стилізація: Bootstrap надає заздалегідь визначені стилі, які можна застосовувати до різних елементів і компонентів. Це спрощує процес стилізації веб-сайту та забезпечує єдність вигляду. [22]
4. Підтримка кросс-браузерності: Bootstrap підтримує всі сучасні браузери і забезпечує однаковий вигляд і функціональність на різних платформах. [22]
5. Легкість використання: Bootstrap має зрозумілу документацію і простий синтаксис класів, що дозволяє швидко навчитися і почати працювати з фреймворком. [22]

Bootstrap також надає можливості розширення та налаштування, що дозволяє використовувати лише необхідні компоненти і стилі для проекту. Він підтримується активною спільнотою розробників. [22]

Для баз даних було обрано : phpMyAdmin - це безкоштовний інструмент управління базами даних MySQL через веб-інтерфейс. Він надає зручну графічну оболонку для керування базами даних, таблицями, запитами, користувачами та іншими аспектами MySQL. [20]

phpMyAdmin дозволяє виконувати різноманітні завдання пов'язані з базами даних, такі як:

- Створення, видалення та редагування баз даних і їх таблиць.
- Виконання SQL-запитів та перегляд результатів.
- Управління користувачами та їх привілеями.
- Імпорт та експорт даних у різних форматах, таких як SQL, CSV та інші.

- Оптимізація та настроювання баз даних.

PhpMyAdmin має простий інтерфейс, який дозволяє навігувати по базам даних та виконувати різні операції без необхідності вручну вводити SQL-код. Він також має вбудовані інструменти для валідації та форматування SQL-запитів, що полегшує роботу з ними. [20]

PhpMyAdmin є популярним інструментом серед веб-розробників і дозволяє зручно та ефективно управляти базами даних MySQL без необхідності встановлювати додаткові програми чи використовувати командний рядок. [20]

3.2 Опис інтерфейсу

Запустивши додаток, користувач спочатку бачить сторінку авторизації, де йому потрібно ввести своє ім'я та "кімнату" або пароль. Цей етап є необхідним для ідентифікації користувача та надання йому доступу до основних функцій та персоналізованого досвіду в додатку.

У поле для імені користувач вводить своє особисте ім'я або псевдонім, яке буде використовуватись для його ідентифікації в системі. Ім'я може бути унікальним для кожного користувача, щоб уникнути конфліктів та забезпечити відповідну ідентифікацію в межах додатку.

"Кімната" або пароль є додатковим елементом для входу в додаток. Це може бути певний номер який дозволяє обмежити доступ до додатку лише авторизованим користувачам.

Введення імені та "кімнати" або пароля є необхідним етапом перед входом в основний інтерфейс додатку. Після успішної авторизації користувач отримує можливість взаємодіяти з функціями, відповідно до свого облікового запису, та отримує персоналізований досвід використання додатку.

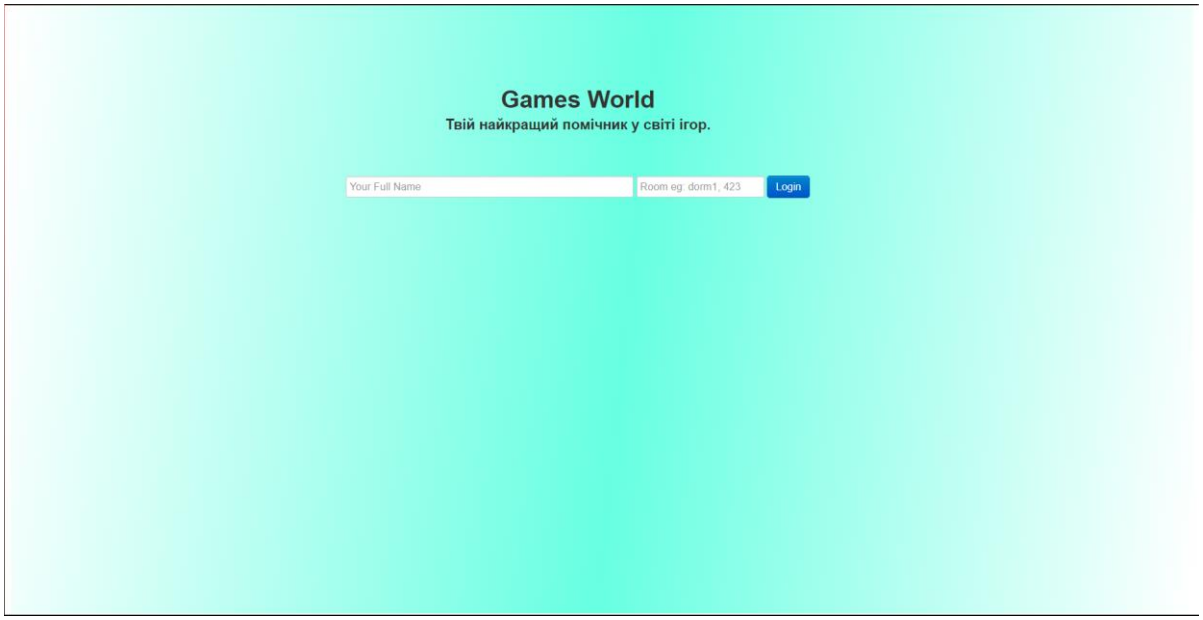

Рисунок 3.1 – Сторінка авторизації користувача

Авторизувавшись, користувач потрапляє на головну сторінку , що зображено на рисунку 3.2.

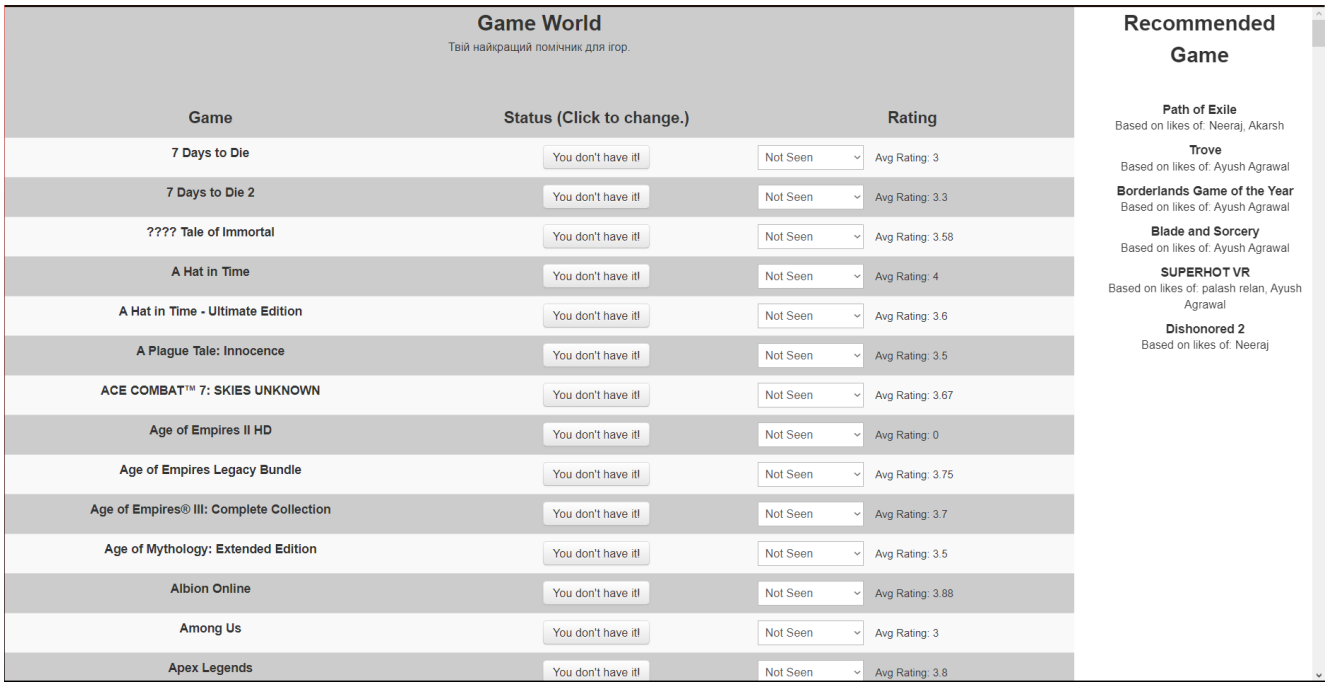

Рисунок 3.2 – Головна сторінка

З рисунку 3.2 можна зрозуміти, що на головній сторінці сторінці присутній список всіх ігор, доступних для користувача, а також невеликий блок з рекомендованими іграми. В цьому списку кожна гра має своє відповідне поле, де

користувач може висловити свою оцінку гри. Поряд з полем для оцінки можна помітити середній рейтинг гри серед інших користувачів.

Для кращого розуміння переходу між сторінками, нижче представлена схема переходів між ними. Схема показує послідовність сторінок і зв'язки між ними, що дозволяє користувачеві легше навігувати та розуміти взаємодію між різними частинами додатку чи веб-сайту.

.Крім того, у списку ігор користувач має можливість позначити кнопкою, що він володіє певною грою. Це може бути корисно для відстеження та організації власної колекції ігор.

Рисунок 3.3 – Список користувачів, що мають певну гру

Коли користувач позначає гру як свою, він може відкрити відповідний блок, де буде вказано, які інші користувачі також володіють цією грою. Це створює можливість для відстежування наявності бажаних ігор ,наприклад, у друзів, якщо вони також користуються даною платформою.

3.3 Тестування програмного модулю

Тестування програмного модулю є процесом перевірки його функціональності, надійності та відповідності вимогам. Під час тестування програмного модулю проводяться різні види тестів для виявлення помилок, дефектів та недоліків у роботі.

Для перевірки працездатності рекомендаційної системи потрібно виконати кілька кроків. Першим кроком є перевірка працездатності системи авторизації користувача. Це необхідно, оскільки перед доступом до рекомендаційних

функцій системи, користувач повинен пройти авторизацію, підтвердивши свою ідентичність.

Отже, дя тестування будуть виконані наступні кроки:

1. Запустити рекомендаційну систему та відрикити сторінку авторизації користувача.

2. В поле введення імені користувача введемо значення "sophi".

3. У поле кімнати введемо значення "423". Це значення ми використовуємо в ролі паролю для перевірки правильності обробки системою введених даних.

4. Натиснемо кнопку "Login" або аналогічний елемент, щоб надіслати введені дані на перевірку.

5. Система авторизації повинна обробити надіслані дані та записати їх. Після цього система має надати доступ до функцій рекомендаційної системи.

В результаті цих дій ми зможемо оцінити працездатність системи авторизації та продовжити тестування рекомендаційної системи в цілому.

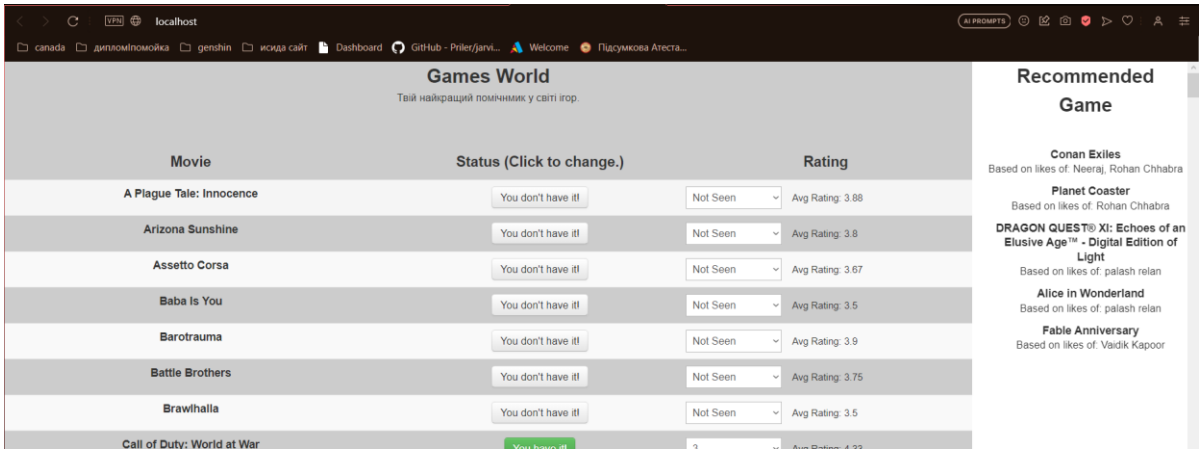

Рисунок 3.4 – Результат тестування авторизації користувача

Також можна побачити , що у таблиці з даними користувачів з’явився новий користувач з тими даними, що були введені під час авторизації.

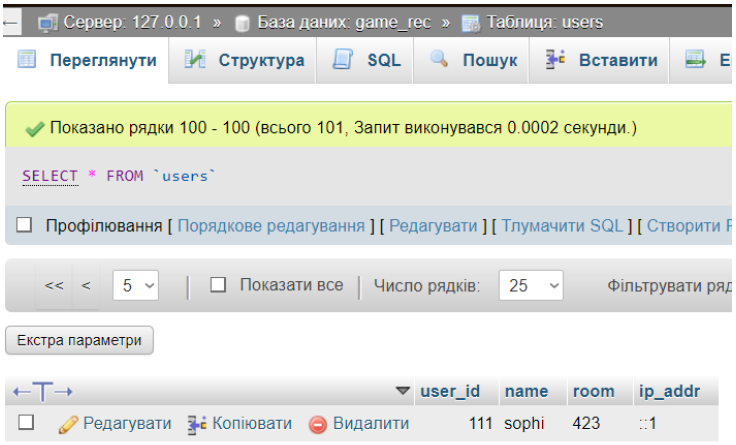

Рисунок 3.5 – Таблиця з даними користувачів

Отже, після успішної перевірки працездатності системи авторизації, ми готові перейти до тестування самої рекомендаційної системи. Для цього ми вирішили оцінити декілька ігор з усього списку доступних. Виконавши цей крок, ми зможемо отримати результати роботи системи та переконатися в її ефективності.

Нижче наведена таблиця з даними ігор, які ми плануємо оцінити:

Таблиця 3.1 – Оцінені ігри для тестування

Назва гри	Оцінка
Call of Duty: World at War	3
Car Mechanic Simulator 2018	3.5
Reventure	5
Wallpaper Engine	2

Для кожної гри ми будемо встановлювати оцінку, що дозволить системі збирати дані та аналізувати вподобання користувачів. Оцінка може бути представлена у вигляді числа, тобто від 1 до 5.

Для отримання рекомендованих ігор може вистачити навіть двох ігор, у деяких випадках, а саме тоді , коли у системі є подібні користувачі, адже система надає рекомендації на основі оцінок інших користувачів, що схожі до користувача «sophi».

Отже, після оцінки 4 ігор, що представлені в таблиці 3.5, користувач може побачити у бічному блоці список з декількох ігор та користувачів на основі, яких було надано рекомендації.

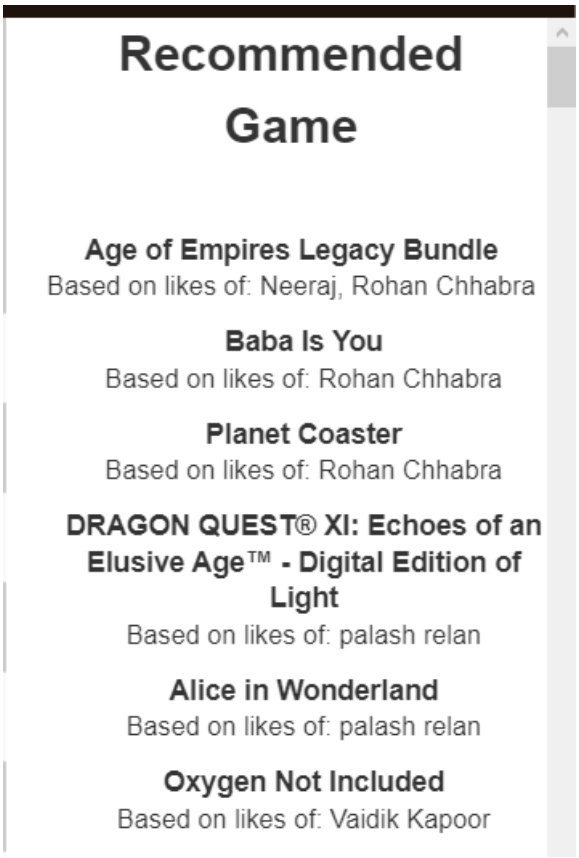

Рисунок 3.6 – Рекомендовані ігри

На рисунку 3.6 можна побачити ,що користувач отримав декілька рекомендованих ігор. Також можна побачити поле з іменами користувачів, оцінки яких схожі до вподобань користувача з іменем «sophi».

Даний список рекомендацій може бути змінений, в залежності від того ,які ще ігри користувач оцінив. Тому оцінимо ще декілька ігор.

Таблиця 3.2 – Оцінені ігри

Назва гри	Оцінка
Call of Duty: World at War	3
Car Mechanic Simulator 2018	3.5
Reventure	5
Wallpaper Engine	2
DOOM	1
DRAGON QUEST® XI: Echoes of an Elusive Age™ - Digital Edition of Light	3.5

Far Cry® 5	5
Forts	4
GOD WARS The Complete Legend	2.5
MapleStory 2	2.5

В результаті користувач бачить вже інший список ігор.

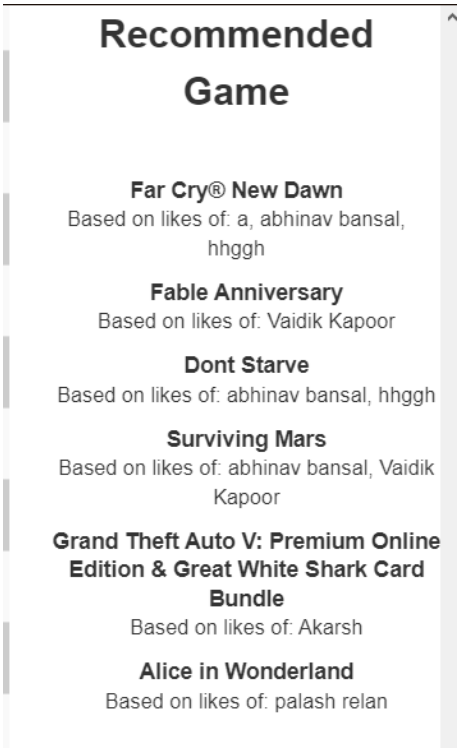

Рисунок 3.7 – Рекомендовані ігри

Також рекомендації ще залежать від оцінок ,тому візьмемо список ігор з таблиці 3.2 та змінимо деякі оцінки на більш позитивні та негативні.

Зміна оцінок допоможе вплинути на алгоритм рекомендацій. Наприклад, якщо користувач отримав рекомендації на основі попередніх оцінок, то зміна оцінки на позитивну або негативну спрямовуватиме програмний модуль до рекомендування інших ігор, які відповідають новим оцінкам.

Цей підхід дозволяє персоналізувати пропозиції для користувачів, враховуючи їхні вподобання та змінені оцінки. В результаті, користувачі отримають рекомендовані ігри, які більше відповідають їхнім смакам та оновленим вподобанням.

Таблиця 3.2 – Оцінені ігри

Назва гри	Оцінка
Call of Duty: World at War	5
Car Mechanic Simulator 2018	1
Reventure	1.5
Wallpaper Engine	5
DOOM	3.5
DRAGON QUEST® XI: Echoes of an Elusive Age™ - Digital Edition of Light	5
Far Cry® 5	2
Forts	4.5
GOD WARS The Complete Legend	1
MapleStory 2	2.5

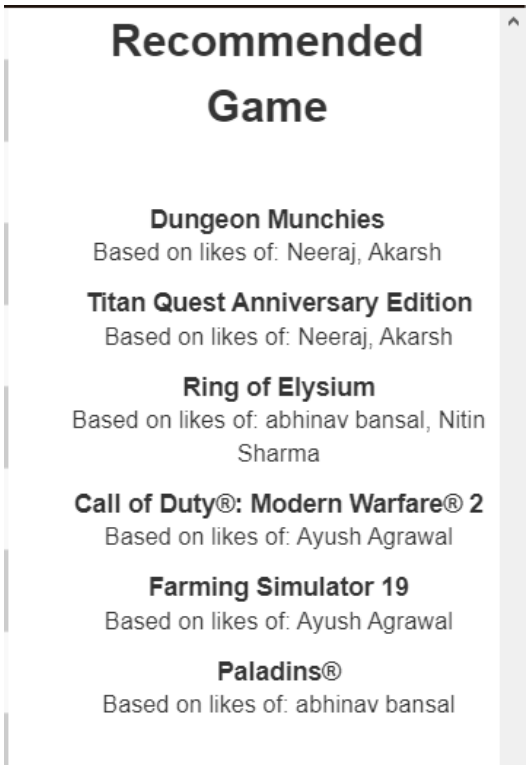

Рисунок 3.8 – Рекомендовані ігри після зміни оцінок

З цього рисунку видно на скільки сильно програмний модуль залежить від оцінок користувачів та їх вподобань, адже за допомогою їх знаходяться інші користувачі, що поставили схожі оцінки для ігор з таблиць 3.2 та 3.3. І тільки тоді користувач може отримати свої рекомендації.

Програмний модуль також має працювати з великими об'ємами даних. Щоб продемонструвати цю можливість, можемо встановити оцінки для близько 100 ігор.

У цьому випадку, для кожної з ігор можна присвоїти довільні оцінки в межах діапазону від 1 до 5. Наприклад, деякі ігри можуть мати оцінку 1, що вказує на низький рівень задоволення користувача, тоді як інші ігри можуть мати оцінку 5, що вказує на високий рівень задоволення користувача.

Цей процес дозволяє згенерувати різноманітні оцінки для великої кількості ігор, що дозволяє виконувати тестування та перевірку працездатності програмного модуля при роботі з великим об'ємом даних. Таким чином, можна впевнитись, що програмний модуль може ефективно працювати з великою кількістю даних та забезпечувати швидкі та точні рекомендації незалежно від обсягу вхідних даних.

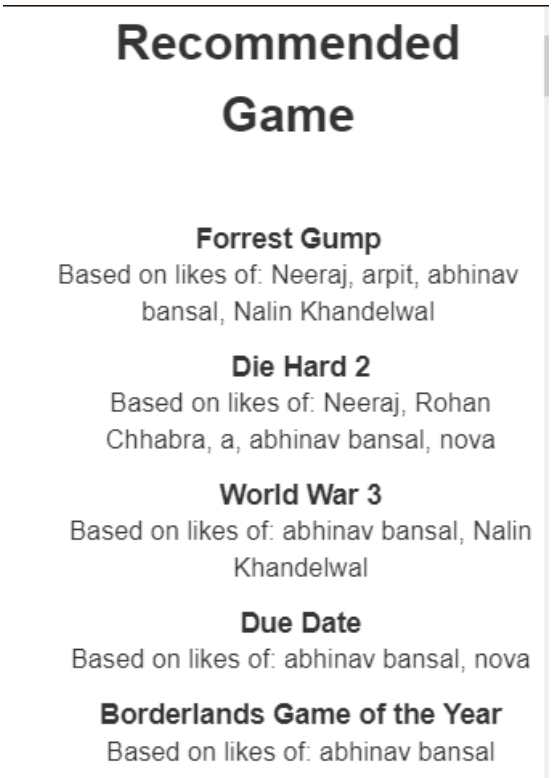

Рисунок 3.9 – Надані рекомендації при оцінці близько 100 ігор

З рисунку 3.9 можна побачити, що програмний модуль працює коректно та користувач може побачити декілька рекомендованих ігор.

На основі наданої інформації з рис. 3.9 можна припустити, що більшість рекомендацій на екрані зроблені на основі оцінок користувача з іменем "abhinav bansal". Це може вказувати на те, що цей конкретний користувач оцінив значну

кількість ігор так само як і користувач з іменем «sophi» та оцінив ще багато інших ігор.

Для проведення цього тестування було додано кількох користувачів до таблиці з даними про оцінки. Це зроблено з метою перевірки працездатності та ефективності модуля при обробці великих обсягів даних.

Процес додавання багатьох користувачів, які оцінили усі ігри, допомагає впевнитись, що програмний модуль здатний обробляти та аналізувати великі обсяги даних. Це також дозволяє виявити можливі проблеми або помилки у модулі, які можуть виникнути при роботі зі значною кількістю користувачів та їхніми оцінками.

Такий підхід до тестування допомагає забезпечити, що програмний модуль працює належним чином та забезпечує точні та релевантні рекомендації для кожного користувача, незалежно від обсягу вхідних даних та кількості користувачів.

ВИСНОВКИ ДО РОЗДІЛУ 3

Для зручного користування програмним забезпеченням було описано роботу інтерфейсу. Це включає в себе структуру, розміщення елементів інтерфейсу, функціональність кожного елементу, взаємодію з користувачем та інші аспекти, що стосуються взаємодії з програмою.

Була розроблена програмна реалізація модулю підбору відео ігор ,також проведено тестування програмного продукту для виявлення помилок у роботі програми та тестування роботи рекомендаційної системи з різною кількістю вхідних даних.

Висновки

Після проведення аналізу основних підходів, бізнес-процесів та існуючих рішень у області рекомендаційних систем з підбору відео ігор, було визначено ключові потреби та вимоги, які варто враховувати під час розробки нового програмного модулю. На основі цих вимог була сформульована задача розробки продукту.

Одним з ефективних підходів до побудови рекомендаційних систем є колаборативна фільтрація, яка дозволяє рекомендувати об'єкти на основі спільної інформації про поведінку користувачів. Метод k-найближчих сусідів (KNN) є одним з методів реалізації колаборативної фільтрації.

Оцінка якості рекомендаційних систем вимагає використання метрик точності прогнозування, які порівнюють прогнози системи з реальними вподобаннями користувачів. Це дозволяє оцінити ефективність та точність рекомендацій.

Були розроблені проектні рішення для реалізації модулю підбору відео ігор, проведено структурно-функціональний аналіз, розроблена концептуальна модель предметної області та фізична модель БД.

Для забезпечення зручного користування програмним забезпеченням був описаний інтерфейс, який включає структуру, розміщення елементів, функціональність та взаємодію з користувачем.

Була розроблена програмна реалізація модулю підбору відео ігор ,також проведено тестування програмного продукту для виявлення помилок у роботі програми та тестування роботи рекомендаційної системи з різною кількістю вхідних даних.

Література

1. Lemire D., Maclachlan A. Slope One Predictors for Online Rating-Based Collaborative Filtering. SDM, 2005, pp. 471-475.
2. Melville P., Sindhwani V. Recommender Systems, Encyclopedia of Machine Learning, Claude Sammut and Geoffrey Webb (Eds), Springer, 2010.
3. Desfray P., Raymond G. Introduction to TOGAF Models. Modeling Enterprise Architecture with TOGAF. 2014. P. 93–102.
4. Friberg P. TOGAF und ArchiMate. Softwarearchitektur pragmatisch. München, 2022. P. 117–171..
5. Recommendation system. International journal of artificial intelligence and machine learning. 2021. Vol. 11, no. 2. P. 0.
6. Recommendation system using autoencoders / D. Ferreira et al. Applied sciences. 2020. Vol. 10, no. 16. P. 5510.
7. TOGAF Version 9 TOGAF. Van Haren Publishing, 2009.
8. Hybrid Recommender Systems: Survey and Experiments. [Електронний ресурс] - Режим доступу до ресурсу: https://pdfs.semanticscholar.org/5880/b9bc3f75f4649b8ec819c3f983a14fca9927.pdf
9. Charu C. Aggarwal. Recommender Systems: The Textbook, 2016.
10. S. H. S.g Chee, J. H., and K. Wang. Rectree: An efficient collaborative filtering method. Lecture Notes in Computer Science, 2114, 2017.
11. Рогинський О., Бабкова Н., Гулієва Д. РЕКОМЕНДАЦІЙНІ СИСТЕМИ ЯК ІНСТРУМЕНТ ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ВЕБ-СЕРВІСІВ.
12. Kim Folk. Practical Recommender Systems, 2019.
13. Deepak K. Agarwal and Bee-Chung Chen. Statistical Methods for Recommender Systems. 2016.
14. Hagino T. HTML5. The Journal of The Institute of Image Information and Television Engineers. 2011. Vol. 65, no. 4. P. 467–470.

15. Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning. — Springer, 2001.
16. Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). Recommender Systems: An Introduction. Cambridge University Press
17. Adomavicius, G., & Tuzhilin, A. (2005). Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6), 734-749.
18. Melville, P., & Sindhwani, V. (2010). Recommender Systems. Encyclopedia of Machine Learning
19. Butler T. PHP and MySQL: Novice to Ninja. SitePoint Pty, Limited, 2021. 680 p.
20. Shiflett C., Baker M., Reiersol D. PHP in Action: Objects, Design, Agility. Manning Publications, 2007. 525 p.
21. Thomson L., Welling L. PHP and MySQL Web Development. Pearson Education, Limited, 2004.
22. Cochran D., Whitley I., Jobsen B. Bootstrap 4 Site Blueprints. Packt Publishing - ebooks Account, 2016. 404 p.

Додаток А

```
<?php

/*function exceptions_error_handler($severity, $message, $filename, $lineno) {
    if (error_reporting() == 0) {
        return;
    }
    if (error_reporting() & $severity) {
        throw new ErrorException($message, 0, $severity, $filename, $lineno);
    }
}
set_error_handler('exceptions_error_handler');*/

function Recommendation($user_id, $dbh){
    $query = 'SELECT * FROM `user_rating` ORDER BY `user_id`';
    $stmt = $dbh->prepare($query);
    $return = $stmt->execute();

    $ratings = array();

    while($row = $stmt->fetch()){
        $ratings[$row['user_id']][$row['game_id']] = $row['rating'];
    }
    $recommendations = getRecommendation($ratings, $user_id);
    $html = '';
    if(!empty($recommendations)){
        $games = getgames($recommendations[0], $dbh, $recommendations[1]);

        foreach($games[0] as $game){
            $html .= '
                <div class="span12" style="margin-top: 10px;">
                      <h4 style="cursor:pointer;" title="Click to see who has this
game." class="has-game" data-id="'.$game[1].'">'.$game[0].'</h4><span>Based on likes
of: ';

            foreach($games[1][$game[0]] as $user){
                $html .= $user.', ';
            }

            $html = rtrim($html, " ,");
            $html .= '</span></div>';
        }
    }
    echo $html;
}
```

```
function similarity($prefs, $p1, $p2){
    $si = array();
    if(!array_key_exists($p1, $prefs)) return 0;
    foreach($prefs[$p1] as $item => $value){
            if(array_key_exists($item, $prefs[$p2])){
                    $si[$item] = 1;
            }
    }

    $n = sizeof($si);
    if($n == 0)
        return 0;
        #For numerator sigma(x*y) - (sigma(x)* sigma(y)/n)
    #              sigma(x*y) - ( sum1 * sum2 / n)
    $sum1 = 0;
    $sum2 = 0;
    $pSum = 0;
    $sum1sq = 0;
    $sum2sq = 0;
    foreach($si as $it => $rate){
        $sum1 += $prefs[$p1][$it];
        $sum2 += $prefs[$p2][$it];
        $pSum += $prefs[$p1][$it] * $prefs[$p2][$it];
        $sum1sq += $prefs[$p1][$it] * $prefs[$p1][$it];
        $sum2sq += $prefs[$p2][$it] * $prefs[$p2][$it];
    }
         #For denominator underoot( ( sigma(x^2)-((sigma(x))^2/n) ) * ( sigma(y^2)-
((sigma(y)^2)/n) ) )
    #                underoot( ( sum1sq-(sum^2/n) ) * ( sum2sq-((sum2^2)/n) ) )
    $num = $pSum - ($sum1 * $sum2 / $n);
    $den = sqrt(($sum1sq - (($sum1 * $sum1) / $n)) * ($sum2sq - (($sum2 * $sum2) /
$n)));
    if($den == 0)
        return 0;
        $sim_corr = $num/$den;
    return $sim_corr;
}

function getRecommendation($prefs, $person, $n = 5){
    $totals = array();
    $simSums = array();
    $based_on_users = array();
    foreach($prefs as $other=>$val){
            if($other == $person) continue;
            $sim = similarity($prefs, $person, $other);
            if($sim <= 0) continue;
            foreach($prefs[$other] as $item => $rating){
                    if(!in_array($item, array_keys($prefs[$person]))){
                            $totals[$item] = 0;
```

```
                            $totals[$item] += $prefs[$other][$item]*$sim;

                            $simSums[$item] = 0;
                            $simSums[$item] += $sim;
                            $based_on_users[$item][] = $other;
                    }
            }
    }
    $rankings = array();
    foreach($totals as $item => $total){
        $rankings[$total] = $item;
    }
    krsort($rankings);
    $list = array();
    foreach($rankings as $total => $item){
        $list[] = $item;
    }
    return array($list, $based_on_users);
}

function getUser($dbh){
    $query = 'SELECT `user_id`, `name` FROM `users`';
    $stmt = $dbh->prepare($query);
    $return = $stmt->execute();
    $user = array();
    while($row = $stmt->fetch()){
        $user[$row['user_id']] = $row['name'];
    }
    return $user;
}

function getgames($r, $dbh, $u){
    $users = getUser($dbh);
    $related = array();
    $query = 'SELECT `game_id`, `game_name` FROM `games` WHERE `game_id` = ?';
    $stmt = $dbh->prepare($query);
    $games = array();
    foreach($r as $i){
        $stmt->bindParam(1, $i, _int);
        $return = $stmt->execute();

        while($row = $stmt->fetch()){
            $games[] = array($row['game_name'], $row['game_id']);
            foreach($u[$i] as $id){
                $related[$row['game_name']][] = $users[$id];
            }
        }

    }
```

```
    return array($games,$related);
}
?>
```