

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

До захисту допущено:

Завідувач кафедри

_____ Олександр РОЛІК

«__» _____ 2024 р.

Дипломний проєкт
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інтегровані інформаційні системи»
спеціальності 126 «Інформаційні системи та технології»
на тему: «Система рекомендації фільмів»

Виконав:

Студент IV курсу, групи ІА-01

Антонюк Степан Петрович _____

Керівник:

Старший викладач кафедри ІСТ

Тимофєєва Юлія Сергіївна _____

Рецензент:

Асистент кафедри ОТ

Каплунов Артем Володимирович _____

Засвідчую, що у цьому дипломному
проєкті немає запозичень з праць інших
авторів без відповідних посилань.

Студент _____

Київ – 2024 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інтегровані інформаційні системи»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Олександр РОЛІК

«__» _____ 2024 р.

ЗАВДАННЯ
на дипломний проєкт студенту
Антонюку Степану Петровичу

1. Тема проєкту «Система рекомендації фільмів», керівник проєкту Тимофєєва Юлія Сергіївна, старший викладач кафедри ІСТ, затверджені наказом по університету від «27» травня 2024 р. № 2112-с.
2. Термін подання студентом проєкту: 10 червня 2024 року.
3. Вихідні дані до проєкту: веб інтерфейс має бути зрозумілим і простим у використанні користувачів; рекомендації повинні надаватися достатньо швидко (в межах двох секунд).
4. Зміст пояснювальної записки: вступ, опис предметної області, огляд існуючих рішень, проєктування схем, вибір інструментів, робота з даними для рекомендаційної системи, реалізація рекомендаційної системи, вебзастосунок, висновки.
5. Перелік графічного матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо): діаграма компонентів, діаграма прецедентів, діаграма послідовності, діаграма діяльності.
7. Дата видачі завдання: 8 березня 2024 року.

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Аналіз предметної області	19.04.2024	
2	Робота над backend	26.04.2024	
3	Робота над frontend	03.05.2024	
4	Робота над рекомендаційною системою	10.05.2024	
5	Оформлення перших розділів ПЗ	17.05.2024	
6	Доопрацювання застосунку	29.05.2024	
7	Оформлення ПЗ	07.06.2024	
8	Подання проєкту	10.06.2024	

Студент

Степан АНТОНЮК

Керівник

Юлія ТИМОФЄЄВА

АНОТАЦІЯ

Система рекомендації фільмів.

Проект містить 60 с. тексту, 26 рисунків, 10 таблиці, посилання на 28 літературні джерела, додатки та 4 конструкторських документів.

РЕКОМЕНДАЦІЙНА СИСТЕМА, КОНТЕНТНО-ОРІЄНТОВАНА
ФІЛЬТРАЦІЯ, ФІЛЬМ, PYTHON, PANDAS, ОБРОБКА ДАНИХ,
ВЕБЗАСТОСУНОК, REACT.

Об'єктом розроблення є система рекомендації фільмів.

Мета розробки — підвищення точності та швидкості надання рекомендацій фільмів.

У дипломному проекті було розроблено систему рекомендації фільмів, яка базується на контентно-орієнтованій фільтрації. Дані для рекомендаційної системи належать до набору «MovieLens». З використанням бібліотеки Pandas, ці дані були очищені і перетворені у потрібну форму. Взаємодія з рекомендаційною системою відбувається через вебзастосунок. Backend вебзастосунку реалізований за допомогою вебфреймворку FastAPI, тоді як frontend через бібліотеку React. Результати тестування показали високу точність рекомендацій та стабільність роботи вебзастосунку. В результаті було отримано працюючу систему, здатну надавати рекомендації фільмів швидко і з задовільною точністю.

Отримані результати можуть бути корисними при створенні аналогічних об'єктів.

SUMMARY

Movie recommendation system.

The project contains 60 pages of text, 26 figures, 10 tables, links to 28 literary sources, annexes and 4 design documents.

Keywords: recommendation system, content-based filtering, movie, python, pandas, data processing, web application, react.

The object of development is a movie recommendation system. The purpose of the development is to increase the precision and speed of movie recommendations.

The graduation project developed a movie recommendation system using content-based filtering. The data for the recommendation system belongs to the MovieLens dataset. Using the Pandas library, the data was cleaned and transformed into a suitable shape. Interactions with the recommendation system occur through a web application. The backend of this web application was developed using the FastAPI web framework, while the frontend was based on the React library. Testing results showed high accuracy of recommendations and stability of the web application. As a result, a working system capable of providing movie recommendations quickly and with satisfying precision was created.

The results obtained can be useful in creation of similar objects.

Номер рядка	Формат	Позначення	Найменування	Кільк. аркушів	Номер екзем.	Примітка		
1			Документація загальна					
2								
3			Знову розроблена					
4								
5	A4	IA01.020БАК.006 ПЗ	Пояснювальна записка	60				
6	A3	IA01.020БАК.006 Д1	Система рекомендації фільмів. Діаграма компонентів	1				
7								
8	A3	IA01.020БАК.006 Д2	Система рекомендації фільмів. Діаграма прецедентів.	1				
9								
10	A3	IA01.020БАК.006 Д3	Система рекомендації фільмів. Діаграма послідовності.	1				
11								
12	A3	IA01.020БАК.006 Д4	Система рекомендації фільмів. Діаграма діяльності.	1				
13								
14								
15								
16								
17								
18								
19								
20								
21								
22								
23								
24								
25								
26								
27								
28								
					IA01.020БАК.006 ТП			
Зм.	Аркуш	№ докум.	Підпис	Дата				
Розроб.		Антонюк С.П.						
Керівн.		Тимофєєва Ю.С.						
					Система рекомендації фільмів. Відомість проекту	Літ.	Аркуш	Аркушів
							1	1
						КПІ ім. Ігоря Сікорського Група ІА-01		
Затв.								

**Пояснювальна записка
до дипломного проєкту
на тему: «Система рекомендації фільмів»**

Київ – 2024 року

ЗМІСТ

ВСТУП	4
1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ	6
1.1 Загальна характеристика рекомендаційних систем	6
1.2 Характеристика колаборативної фільтрації.....	6
1.3 Характеристика контентно-орієнтованої фільтрації	8
1.4 Вибір типу рекомендаційної системи.....	9
Висновки до розділу 1	10
2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ.....	11
2.1 PickAMovieForMe.....	11
2.2 Netflix	12
Висновки до розділу 2.....	13
3 ПОСТАНОВКА ВИМОГ	14
3.1 Функціональні вимоги	14
3.2 Нефункціональні вимоги	15
Висновки до розділу 3.....	15
4 ПРОЄКТУВАННЯ СХЕМ.....	16
4.1 Архітектура клієнт-сервер	16
4.2 Розроблення діаграми прецедентів.....	17
4.3 Розроблення діаграми послідовності.....	18
4.4 Розроблення діаграми діяльності.....	19
4.5 Сценарії використання системи.....	20
Висновки до розділу 4.....	27
5 ВИБІР ІНСТРУМЕНТІВ.....	29
5.1 Вибір мов програмування.....	29
5.2 Перелік головних вебфреймворків та бібліотек проєкту	30
5.3 Вибір РСУБД	31
5.4 Вибір середовища розробки	32

					ІА01.020БАК.006 ПЗ			
Зм.	Лист	№ докум.	Підпис		Система рекомендації фільмів. Пояснювальна записка			
Розробив	Антонюк С.П.							
Перевірив	Тимофєєва Ю.С.							
Затв.								
						Літ.	Арк.	Аркушів
						Т	2	60
						КПІ ім. Ігоря Сікорського Група ІА-01		

5.5 Додаткові інструменти.....	32
Висновки до розділу 5.....	32
6 РОБОТА З ДАНИМИ ДЛЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ	33
6.1 Характеристика обраного набору даних.....	33
6.2 Обробка даних	33
Висновки до розділу 6.....	42
7 РЕАЛІЗАЦІЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ.....	43
7.1 Робота над рекомендаційною системою	43
Висновки до розділу 7.....	47
8 ВЕБЗАСТОСУНОК	48
8.1 Backend	48
8.2 Frontend.....	50
8.3 Тестування вебзастосунку і рекомендаційної системи	51
Висновки до розділу 8.....	56
ВИСНОВКИ.....	57
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	58
ДОДАТОК А.....	61

ВСТУП

З моменту зародження в кінці 19 століття і до теперішніх часів було створено десятки (якщо не сотні) тисяч фільмів. Серед цієї неймовірної кількості розважального контенту, стає дедалі важче знайти щось дійсно варте уваги. Вибір фільму для перегляду починає займати багато часу, а результат не завжди виправдовує очікування. Ця проблема стала особливо актуальною в епоху стрімінгових сервісів, де доступ до тисяч фільмів та серіалів лежить всього лиш за декількома рухами пальця.

Саме тут на допомогу приходять найрізноманітніші рекомендаційні системи, створені, щоб допомагаючи користувачам знайти фільми, які їм сподобаються. В залежності від реалізації, ці системи можуть рекомендувати схожі фільми до переглянутих, абсолютно протилежні до звичних користувачу фільмів, що належать до рідкісних жанрів, але які теж можуть сподобатися користувачеві тощо. Проте, попри весь прогрес за останні роки, ці системи досі часто лишають бажати кращого через недостатнє розуміння кінцевого користувача, швидкість, та інші проблеми.

На основі вищесказаного, об'єктом дослідження даного дипломного проєкту стане процес надавання рекомендацій фільмів.

Предмет дослідження — реалізація системи рекомендації фільмів зі зручним та зрозумілим користувацьким вебінтерфейсом, здатну швидко надавати рекомендації фільмів задовільної якості.

Мета дипломного проєкту — покращення швидкості і точності надання рекомендацій.

Для задоволення вищепоставленої мети, необхідно буде виконати наступні задачі:

- проаналізувати предметну область;
- розглянути існуючі рішення, визначаючи їхні переваги та недоліки;
- сформулювати вимоги до системи, яку буде розроблено;
- визначити який тип рекомендаційної системи буде реалізовано;

					ІА01.020БАК.006 ПЗ	Арк.
						4
Зм.	Лист	№ докум.	Підпис	Дата		

- визначивши тип рекомендаційної системи, потрібно знайти дані, на які опиратиметься вибраний різновид рекомендаційної системи;
- виконати структурну та функціональну схеми з описом;
- вибрати інструменти реалізації, включно з мовою програмування (або мовами), базу даних тощо.

Потім, на основі цього, власне, потрібно буде реалізувати саму рекомендаційну систему з вебінтерфейсом.

Практичне значення даної системи полягатиме у зменшенні часу, витраченого на пошуки фільму, і, в результаті, проведення того вивільненого часу за переглядом вдало підібраного фільму.

Дипломний проєкт буде містити в собі наступні розділи: вступ; опис предметної області; аналіз існуючих рішень; постановка вимог; проєктування схем; вибір інструментів; робота з даними для рекомендаційної системи; реалізація рекомендаційної системи; вебзастосунок; висновок; перелік використаних джерел; додаток. Висновки до розділів є включені до кожного розділу окрім вступу, висновку, додатку, та переліку використаних джерел. Було створено чотири кресленики формату А3. Робота має обсяг у 61 сторінок.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Загальна характеристика рекомендаційних систем

Системи рекомендацій фільмів — це інтелектуальні алгоритми, що пропонують користувачам фільми на основі їхньої попередньої поведінки чи вподобань. Дані системи аналізують дані, такі як користувацькі оцінки, відгуки та історії переглядів заради створення персоналізованих рекомендацій [1].

Існує два головних типи рекомендаційних систем: колаборативна фільтрація (англ., collaborative filtering) та контентно-орієнтована фільтрація (англ., content-based filtering). На рисунку 1.1 зображено їхнє порівняння між собою.



Рисунок 1.1 — Порівняння головних типів рекомендаційних систем [2]

1.2 Характеристика колаборативної фільтрації

Колаборативна фільтрація — це тип фільтрації, що знаходить інших користувачів, подібних до конкретного користувача, і рекомендує тому

конкретному користувачу їхні (інших користувачів) вибори. Важливий момент: при колаборативній фільтрації, не враховуються особливості продукту.

Іншими словами, система дивиться на фільми які сподобалися користувачу А, і знаходить інших користувачів яким теж сподобалися ті фільми. Потім система рекомендує користувачу А фільми, які сподобалися вищезгаданим користувачам. Даний принцип роботи показано на рисунку 1.2.

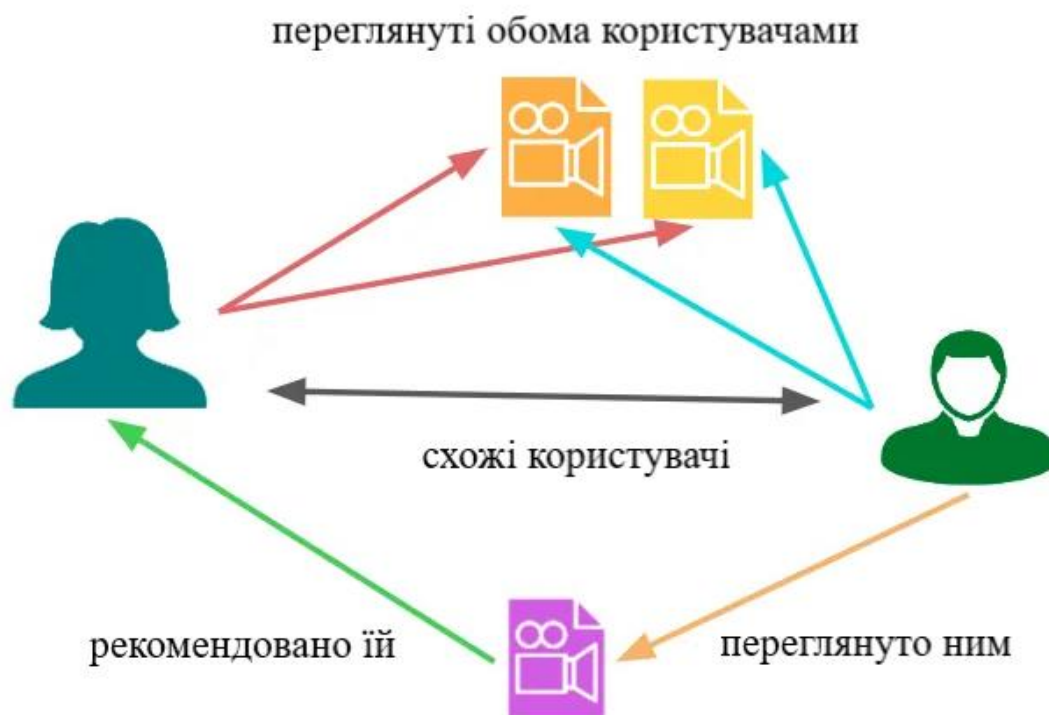


Рисунок 1.2 — Як працює колаборативна фільтрація [3]

Наприклад, уявимо, що певному користувачеві А разом з його подругою сподобався фільм «Jumanji: Welcome to the Jungle» (2017). Його подрузі також сподобався «Dune: Part Two» (2024). На основі цього вподобання подруги, система також рекомендуватиме користувачеві А «Dune: Part Two», вважаючи, що даному користувачеві подобатимуться ті ж фільми, що й його подрузі.

Переваги колаборативної фільтрації:

- модель надає високо персоналізовані рекомендації на основі поведінки користувача;
- модель здатна рекомендувати широкий спектр фільмів, які користувач, можливо, не знайшов би самотужки;

- використовуються колективні вподобання користувачів, що часто забезпечує більш точні рекомендації відносно контентно-орієнтованої фільтрації;
- модель постійно оновлюється на основі дій користувачів, що призводить до релевантних та up-to-date рекомендацій.

Недоліки колаборативної фільтрації:

- проблема «холодного старту» (англ., cold start problem) – при додаванні нових фільмів чи користувачів, система не здатна надавати точні рекомендації через відсутність достатньої кількості даних про нові елементи;
- popularity bias - популярні фільми рекомендуються частіше за менш відомі;
- проблеми з масштабуванням при великих наборах даних [4].

1.3 Характеристика контентно-орієнтованої фільтрації

Контентно-орієнтована фільтрація - це підхід, що використовує атрибути та метадані фільму для генерації рекомендацій, що мають подібні властивості. Прикладами метаданих для створення рекомендацій можуть бути жанри фільму, кінорежисер, актори, сюжет, теми, ключові слова тощо [1]. З врахуванням цих (або інших) даних, система виконує рекомендацію схожих фільмів.

Іншими словами, система рекомендує новий контент на основі схожості з тим контентом, що користувачеві вже сподобався.

Наприклад, користувач вподобав фільм «Casino» (1995). Система рекомендації визначає, що даний фільм має наступні характеристики: жанр – Crime; кінорежисер - Martin Scorsese; актор – Robert De Niro. На основі цих даних, система рекомендує схожі фільми про кримінал кінорежисера Martin Scorsese з актором Robert De Niro, а саме «GoodFellas» (1990) та «Raging Bull» (1980).

Переваги контентно-орієнтованої фільтрації:

- рекомендації не залежать від активності інших користувачів, що робить систему стійкою до змін у поведінці користувачів;
- система здатна рекомендувати нові або мало відомі фільми, якщо їх характеристики схожі на ті, що вже сподобалися користувачу;

– система надає рекомендації на основі індивідуальних вподобань користувача.

Недоліки контентно-орієнтованої фільтрації:

– система може пропонувати надто схожі фільми, що робить рекомендації менш різноманітними;

– якщо метадані фільму позначені не правильно, то якість контентної фільтрації буде погіршуватися.

Принцип роботи контентно-орієнтованої фільтрації показано на рисунку 1.3.

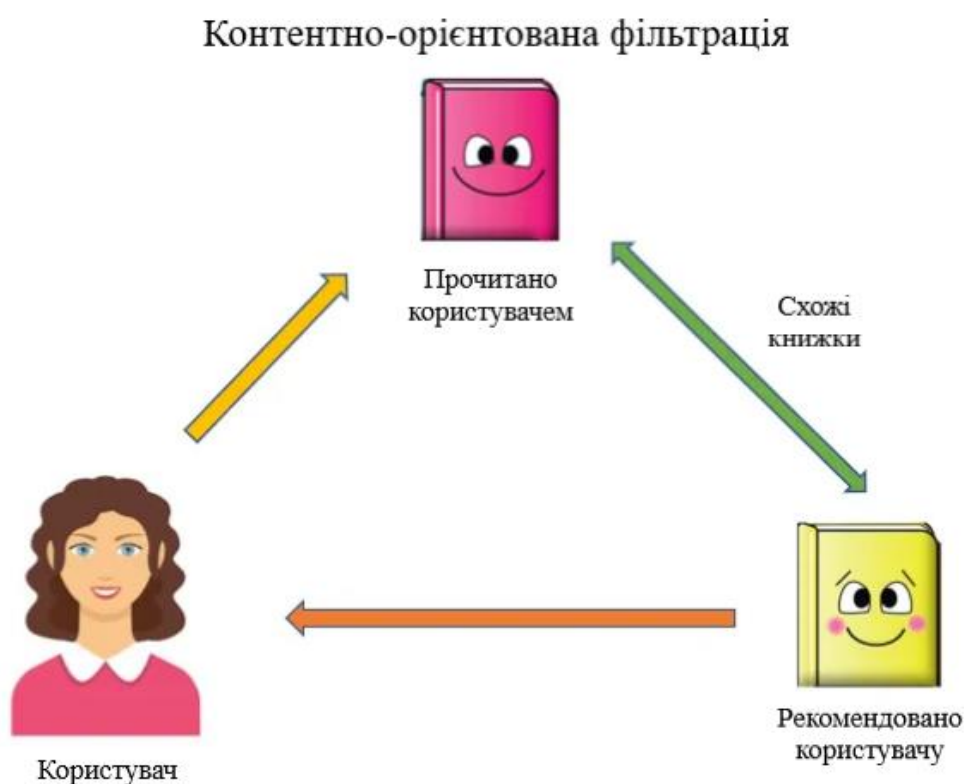


Рисунок 1.3 — Як працює контентно-орієнтована фільтрація [5]

1.4 Вибір типу рекомендаційної системи

Врахувавши всі плюси і мінуси двох основних підходів, було вирішено реалізовувати рекомендаційну систему на основі контентно-орієнтованої фільтрації.

Причини вибору були наступними:

1) навідміну від колаборативної фільтрації, контентно-орієнтована фільтрація не має проблеми холодного старту. Іншими словами, в цьому підході відсутня проблема збору перших відгуків - для реалізації даної системи потрібно мати лише добре структуровані метадані про фільми;

2) рекомендації на основі контентно-орієнтованої фільтрації будуть повністю будуватися на характеристиках фільму замість того, щоб покладатися на оцінку схожих користувачів. Прикладом таких характеристик є: головні актори, кінорежисер, жанри тощо. Таким чином буде підвищений шанс на отримання рекомендацій менш популярних, але більш схожих фільмів до переглянутого.

Висновки до розділу 1

В даному розділі було надане визначення рекомендаційних систем. Після цього було наведено два головних їхніх різновиди: контентно-орієнтована фільтрація та колаборативна. Для кожного різновиду, було вказано його коротку характеристику, переваги і недоліки.

В підсумку, було вирішено реалізовувати рекомендаційну систему на основі контентно-орієнтованої фільтрації, тому що:

- даних підхід не має проблеми холодного старту;
- рекомендації фільмів будуть базуватися тільки на основі їх характеристик (жанр, кінорежисер тощо).

2 АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

В даному розділі буде розглянуто декілька існуючих рішень в області рекомендаційних систем. Для кожного рішення будуть вказані його переваги і недоліки.

2.1 PickAMovieForMe

PickAMovieForMe — це онлайн-сервіс, що допомагає користувачам знайти фільм для перегляду на основі їхніх відповідей на декілька питань. На рисунку 2.1 показано результат роботи даного сервісу.

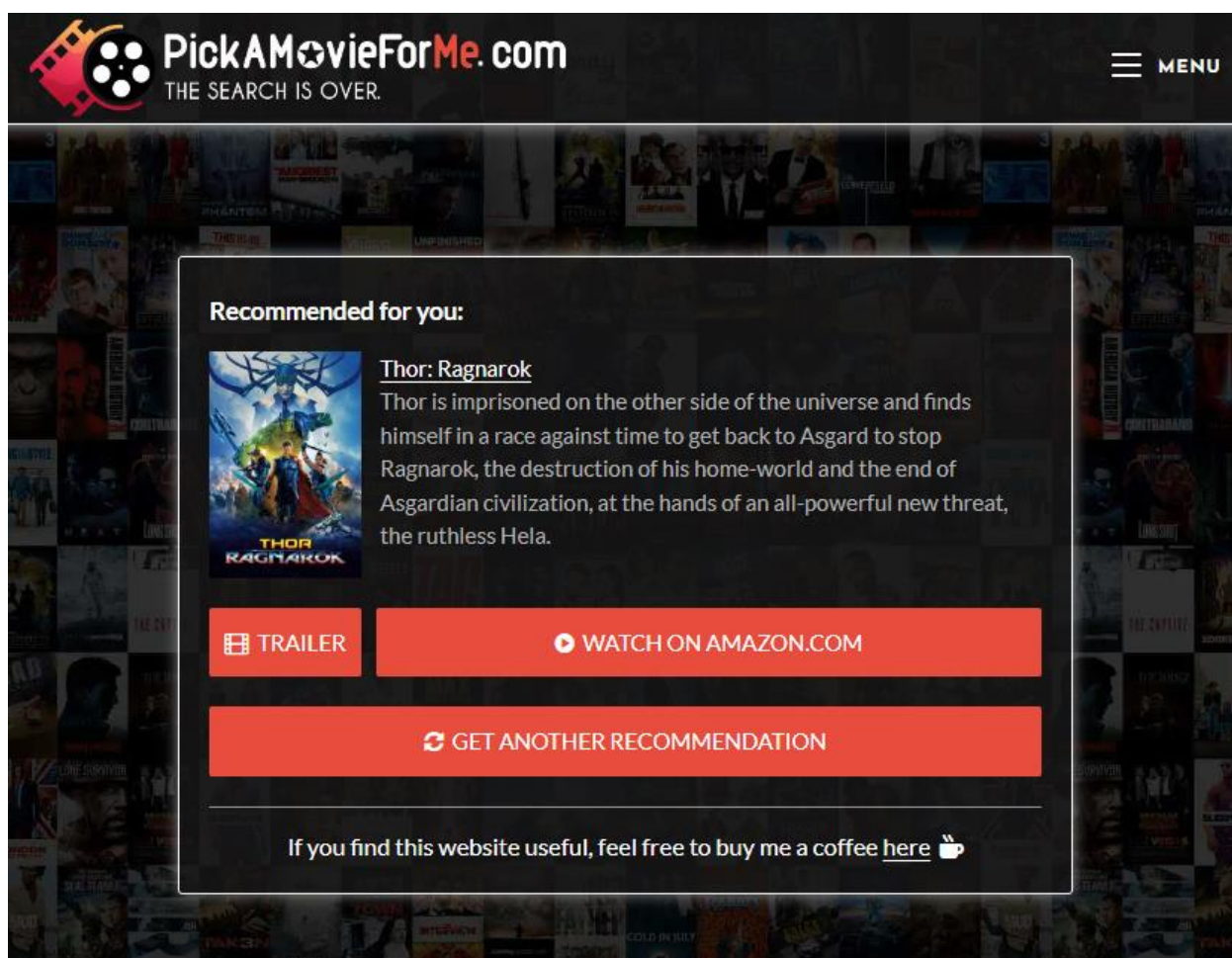


Рисунок 2.1 — Рекомендація від PickAMovieForMe після надання відповідей на шість питань [6]

Даний сервіс задає наступні питання: який настрій користувача сьогодні; з ким користувач дивитиметься фільм; жанри, в який користувач зацікавлений; наскільки старі фільми користувачеві прийнятні; чи подавати фільми з обмеженням по віку; чи в користувача є певні додаткові вподобання

Плюси:

- повністю безкоштовний;
- відсутність потреби в реєстрації чи надання своєї пошти – відразу можна користуватися.

Мінуси:

- проходити через шість меню з опитуваннями на багато варіантів – це надто довго;
- відсутність можливості пропустити не цікаві питання;
- після проходження вікторини, надається тільки одна рекомендація фільму.

2.2 Netflix

Netflix — це стрімінговий сервіс з широким вибором телевізійних шоу, фільмів, аніме, документальних фільмів та інших програм на тисячах, підключених до інтернету, пристроях.

Алгоритми рекомендацій та пошуку є центральними в роботі сервісу Netflix. Вони відіграють ключову роль у наданні персоналізованих рекомендацій користувачам, що відповідають вподобанням користувачів у будь-який момент. Враховуючи важливість цих рекомендацій, постійно проводиться робота над їх вдосконаленням та розвитком.

Персоналізація розпочинається з головної сторінки і розповсюджується на всі інші аспекти продукту. Це включає персоналізовані повідомлення та сповіщення в додатках для утримання користувачів в курсі та підтримання їхньої зацікавленості. З урахуванням різноманітних потреб клієнтів, було розроблено розширені функції пошуку, що дозволяють ефективно орієнтуватися в каталозі та знаходити необхідні відео та ігри. Це також включає вирішення проблем,

					IA01.020БАК.006 ПЗ	Арк.
						12
Зм.	Лист	№ докум.	Підпис	Дата		

пов'язаних з підтримкою багатьох мов і обробкою механізмів введення з найрізноманітніших пристроїв, таких як телевізійні пульти та пристрої з підтримкою голосових команд. Крім того, розглядаються нові способи презентації рекомендацій, їх пояснення та взаємодії з системами. Мета полягає в тому, щоб мінімізувати час на перегляд і пошук, одночасно максимізуючи задоволення від використання даного сервісу [7].

Плюси:

- платформа пропонує широкий вибір контенту;
- постійна адаптація до вподобань користувачів, що забезпечує індивідуальний досвід кожному користувачеві;
- можливість використовувати даний сервіс на багатьох платформах.

Мінуси:

- платний сервіс;
- потрібно виконати відносно великий перелік кроків перед тим як буде можливість отримувати рекомендації;
- можливі проблеми з конфіденційністю оскільки система збирає дані про користувачів.

Висновки до розділу 2

У цьому розділі було розглянуто два популярних сервіси з надання рекомендацій фільмів: PickAMovieForMe та Netflix. Було проаналізовано їхні переваги та недоліки, що дозволило краще зрозуміти їхні особливості та ефективність у різних контекстах.

3 ПОСТАНОВКА ВИМОГ

3.1 Функціональні вимоги

Функціональні вимоги визначають, які конкретні функції має виконувати система. Система рекомендації фільмів на основі контентно-орієнтованої фільтрації з веб-інтерфейсом має наступні функціональні вимоги:

- надані рекомендації фільмів повинні бути адекватними і доречними;
- взаємодія користувача з рекомендаційною системою повинна бути реалізована через веб інтерфейс;
- першою сторінкою системи повинна бути сторінка входу в обліковий запис з можливістю зареєструватися. Якщо користувач спробує потрапити на інші сторінки, не пройшовши автентифікації, то користувач має бути автоматично переправленим на сторінку автентифікації;
- повинна бути можливість увійти в систему, володіючи вірною комбінацією пароля та електронної пошти;
- користувачі повинні бути поділеними на гостей (незареєстровані користувачі), звичайних (зареєстрованих) користувачів, та адміністраторів;
- гості (незареєстровані користувачі) повинні мати можливість зареєструватися;
- звичайні (зареєстровані) користувачі повинні мати функціонал гостя. Окрім цього, звичайні користувачі повинні мати наступні можливості: можливість вводити назву фільму і отримувати рекомендацію; можливість шукати фільм за його назвою; можливість переглядати список 100 останніх фільмів з назвою, акторами, кінорежисером, роком випуску, жанром тощо; можливість переглядати дані свого облікового запису (користувацьке ім'я, електронна пошта, пароль) з можливістю їх змінити;
- адміністратори повинні мати функціонал звичайних (зареєстрованих) користувачів. Окрім цього, адміністратори повинні мати наступні додаткові можливості: можливість додавати новий фільм і редагувати інформацію про нього, або за потреби видалити його; можливість переглядати дані всіх зареєстрованих

					ІА01.020БАК.006 ПЗ	Арк.
						14
Зм.	Лист	№ докум.	Підпис	Дата		

користувачів і мати можливість змінювати дані їхніх облікових записів (користувацьке ім'я, електронна пошта, пароль), і, за потреби, видаляти користувачів; можливість створювати нових користувачів.

3.2 Нефункціональні вимоги

Нефункціональні вимоги описують якісні аспекти системи, такі як продуктивність, безпека, надійність тощо. Дана рекомендаційна система матиме наступні нефункціональні вимоги:

- веб інтерфейс має бути зрозумілим і простим у використанні користувачів;
- рекомендації повинні надаватися достатньо швидко (в межах декількох секунд);
- система повинна бути надійною, забезпечуючи безперервну роботу без збоїв.

Висновки до розділу 3

У даному розділі було визначено функціональні та нефункціональні вимоги до системи рекомендацій фільмів на основі контентно-орієнтованої фільтрації. Запропоновані вимоги дозволять створити ефективну, зручну та економічно вигідну систему, яка задовольнить широке коло користувачів.

4 ПРОЄКТУВАННЯ СХЕМ

4.1 Архітектура клієнт-сервер

Архітектура клієнт-сервер є моделлю обчислення, у якій сервер зберігає, надає та керує більшістю ресурсів та послуг клієнта. Ця модель також відома як мережева обчислювальна модель або клієнт-серверна мережа, оскільки всі запити та послуги надаються через мережу. Архітектура або модель клієнт-сервер має інші системи, підключені через мережу, де ресурси спільно використовуються між різними комп'ютерами. На рисунку 4.1 показано дану архітектуру.

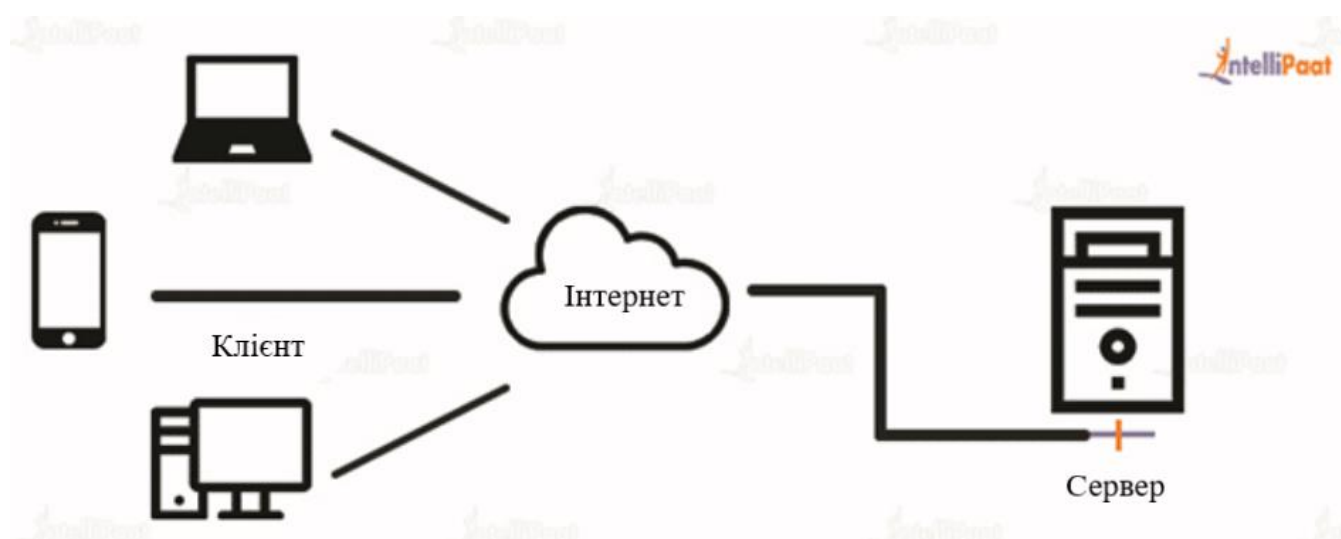


Рисунок 4.1 — Архітектура клієнт-сервер [8]

Зазвичай архітектура клієнт-сервер організована таким чином, що клієнти часто знаходяться на робочих станціях або на персональних комп'ютерах, в той час як сервери розташовані в іншому місці у мережі, зазвичай на більш потужних машинах. Така модель стає особливо корисною, коли клієнти та сервер виконують монотонні завдання. Наприклад, у обробці даних у лікарнях комп'ютер клієнта може бути зайнятий виконанням програми для введення інформації про пацієнта, тим часом як серверний комп'ютер може виконувати іншу програму для витягування та керування базою даних, в якій ця інформація постійно зберігається [8].

4.2 Розроблення діаграми прецедентів

Діаграма прецедентів, або діаграма варіантів використання (англ., use case diagram) використовується для представлення динамічної поведінки системи. Вона уособлює функціональність системи, включаючи в себе випадки використання, акторів та їхні зв'язки. Діаграма моделює завдання, сервіси та функції, необхідні для системи/підсистеми додатка. Вона зображує високорівневий функціонал системи та також показує, як користувач взаємодіє з системою.

Мета діаграм прецедентів: основною метою діаграм прецедентів є зображення динамічного аспекту системи. Вона накопичує потреби системи, які включають як внутрішні, так і зовнішні впливи. Вона викликає осіб, випадки використання та кілька речей, що викликають акторів та елементи, відповідальні за впровадження діаграм прецедентів. Вона представляє, як сутність від зовнішнього середовища може взаємодіяти з частиною системи.

Нижче було наведено цілі діаграм прецедентів:

- вона збирає потреби системи;
- вона зображує зовнішній вигляд системи;
- вона визначає внутрішні та зовнішні фактори, які впливають на систему;
- вона представляє взаємодію між акторами [9].

Діаграма прецедентів для системи рекомендації фільмів зображена на кресленику ІА01.020БАК.006 Д2. Дана діаграма складається з трьох акторів: Гість, Користувач та Адміністратор.

Гість має можливість зареєструватися і увійти в аккаунт, володіючи вірною комбінацією електронної пошти і пароля.

Користувач (або звичайний чи зареєстрований користувач) наслідуює можливості гостя. Крім того, користувач може виконувати наступні дії:

- отримати рекомендації фільмів;
- переглянути список фільмів;
- шукати фільм за назвою;

- переглядати дані власного облікового аккаунта з можливістю змінити ці дані чи взагалі видалити свій аккаунт;
- вийти з свого аккаунта.

Адміністратор (або користувач з адміністративними привілеями) наслідуює користувача, і, крім цього, має наступні можливості:

- переглянути список всіх користувачів вебзастосунку і, при потребі, додавати, редагувати, чи видаляти користувачів
- додавати, редагувати, та видаляти фільми.

4.3 Розроблення діаграми послідовності

Діаграма послідовності (англ., sequence diagram) — діаграма, на якій показані взаємодії об'єктів, упорядковані за часом їхнього прояву.

На діаграмі послідовності неявно присутня вісь часу, що дозволяє візуалізувати тимчасові відношення між переданими повідомленнями. За допомогою діаграми послідовності можна представити взаємодію елементів моделі як своєрідний часовий графік «життя» всієї сукупності об'єктів, зв'язаних між собою для реалізації варіанта використання програмної системи, досягнення бізнес-мети або виконання якого-небудь завдання.

На діаграмі послідовності зображуються об'єкти, які безпосередньо беруть участь у взаємодії, при цьому ніякі статичні зв'язки з іншими об'єктами не візуалізуються. Для діаграми послідовності ключовим моментом є саме динаміка взаємодії об'єктів у часі. При цьому діаграма послідовності має як би два виміри. Один – простягається зліва направо у вигляді вертикальних ліній, кожна з яких зображує лінію життя окремого об'єкту, що бере участь у взаємодії. Другий вимір діаграми послідовності – вертикальна тимчасова вісь, спрямована зверху вниз [10].

Діаграма послідовності для системи рекомендації фільмів показана на кресленнику IA01.020БАК.006 ДЗ. Дана діаграма показує покрокову роботу системи рекомендації фільмів:

- 1) користувач вводить назву фільму;

					IA01.020БАК.006 ПЗ	Арк.
						18
Зм.	Лист	№ докум.	Підпис	Дата		

- 2) до бази даних відправляється запит на отримання інформації про фільми;
- 3) від бази даних іде інформація;
- 4) фільм знаходиться за введеною користувачем назвою;
- 5) проходить вибір стовпців для рекомендаційної системи і їх подальша обробка;
- 6) відбувається обчислення cosine similarity matrix на основі оброблених стовпців;
- 7) дані сортуються у порядку спадання, щоб найбільш схожі фільми були першими;
- 8) найбільш схожі фільми показуються користувачеві.

4.4 Розроблення діаграми діяльності

Діаграма діяльності (Activity Diagram) візуалізує процес використання та ілюструє потік повідомлень від однієї дії до іншої. Показує цілісну роботу системи.

Діаграма діяльності вважається розширеним варіантом блок-схем. Проте блок-схема не має стандартизованої нотації та не містить паралелізмів (паралельне виконання дій) та синхронізації.

Також часто виникає питання, у чому різниця між діаграмою діяльності та послідовності. Головна мета діаграми послідовності — показати порядок виконання, або послідовність дій. Водночас діаграма діяльності потрібна для опису роботи всієї системи, вона показує перехід від однієї дії до іншої.

Ці дії можуть виконуватися людьми, програмними компонентами або комп'ютерами. Потік керування (порядок виконання) на діаграмі діяльності переходить від однієї операції до іншої. Цей потік може бути послідовним, розгалуженим або одночасним [11].

Діаграма діяльності рекомендаційної системи фільмів зображена на кресленику IA01.020БАК.006 Д4. На даній діаграмі показано які дії може виконати звичайний, зареєстрований користувач без адміністративних привілеїв після успішної аутентифікації. Можливості є наступними:

					IA01.020БАК.006 ПЗ	Арк.
						19
Зм.	Лист	№ докум.	Підпис	Дата		

- перейти у вкладку «Recommender»;
- перейти у вкладку «Movies»;
- перейти у вкладку «Search for a Movie»;
- перейти у вкладку «User Settings»;
- вийти з аккаунта.

4.5 Сценарії використання системи

На таблиці 4.1 буде показано коротко описані варіанти або сценарії використання системи.

Таблиця 4.1 — Сценарії використання системи

Номер сценарію	Назва сценарію
1	Перебування в меню автентифікації
1.1	Автентифікація
1.2	Реєстрація
2	Перебування на сторінці «Dashboard»
3	Використання рекомендаційної системи
4	Перегляд списку фільмів
4.1	Додати новий фільм
4.2	Редагувати фільм
4.3	Видалити фільм
5	Пошук інформації про фільм за введеною назвою
6	Перегляд користувацького профіля
6.1	Редагувати дані користувача (ім'я, пароль, та пошта)
6.2	Видалити аккаунт користувача (якщо не адміністратор)
7	Перегляд списку користувачів
7.1	Додати нового користувача
7.2	Редагувати користувача
7.3	Видалити користувача

8	Вийти з свого аккаунта
---	------------------------

У таблицях 4.2-4.8 буде наведено більш детальний опис найважливіших сценаріїв використання.

Таблиця 4.2 — Сценарій використання 1.1

Назва	Автентифікація
ID	1.1
Опис	Користувач намагається увійти в свій аккаунт
Актори	неавтентифікований користувач (звичайний користувач чи адміністратор)
Частота користування	Висока
Тригери	Користувач натискає на кнопку «Log In»
Передумови	Користувач знаходиться на сторінці автентифікації
Постумови	Користувач успішно увійшов у свій аккаунт
Основний розвиток	1) Користувач перебуває на сторінці автентифікації. 2) Користувач вводить вірну комбінацію електронної пошти і пароля. 3) Користувач натискає кнопку «Log In». 4) Користувач успішно автентифікується
Альтернативні розвитку	Перехід на сторінку створення нового аккаунта через натискання «Don't have an account? Sign up»
Виняткові розвитку	Не вірна комбінація електронної пошти і пароля

Таблиця 4.3 — Сценарій використання 1.2

Назва	Реєстрація
ID	1.2
Опис	Гість вирішив створити новий обліковий запис для себе
Актори	Гість
Частота користування	Низька

Тригери	Гість натискає на кнопку «Create»
Передумови	На сторінці автентифікації, гість натиснув на «Don't have an account? Sign up»
Постумови	1) Створюється новий обліковий акаунт з введеними гостем даними. 2) Гість переноситься до сторінки автентифікації
Основний розвиток	1) Користувач вводить електронну пошту і пароль для нового акаунта 2) Користувач натискає кнопку «Create» 3) Новий користувач створюється системою
Альтернативні розвитку	Гість вирішив не створювати новий обліковий запис і повернувся на сторінку автентифікації через натискання «Back to log in page»
Виняткові розвитку	1) Гість вводить електронну пошту яка вже зайнята. 2) Гість вводить електронну пошту у невірному форматі.

Таблиця 4.4 — Сценарій використання 3

Назва	Використання рекомендаційної системи
ID	3
Опис	Користувач вводить назву фільму для отримання декількох схожих фільмів на основі введеної назви
Актори	Звичайний користувач, адміністратор
Частота користування	Висока
Тригери	Натискання на опцію «Recommender» в лівому бічному меню
Передумови	Успішна автентифікація
Постумови	Список рекомендованих фільмів
Основний розвиток	1) Користувач вводить назву фільму.

	<p>2) Система знаходить фільми для надання рекомендації.</p> <p>3) Система надає рекомендовані фільми користувачеві.</p>
Альтернативні розвідки	<p>Користувач переходить до однієї з наступних сторінок:</p> <ul style="list-style-type: none"> - список фільмів на сторінці «Movies»; - пошук фільму за назвою на сторінці «Search for a Movie»; - налаштування користувацького профіля на сторінці «User Settings»; - (якщо адміністратор) керування користувачами на сторінці «Admin»; - вихід з облікового запису; - покидання сайту.
Виняткові розвідки	-

Таблиця 4.5 — Сценарій використання 4

Назва	Перегляд списку фільмів
ID	4
Опис	Користувач переглядає список останніх доданих фільмів з характеристиками такі як title, year, genres, vote average, vote count, director, та top actors
Актори	Звичайний користувач, адміністратор
Частота користування	Середня
Тригери	Натискання на опцію «Movies» в лівому бічному меню
Передумови	Успішна автентифікація
Постумови	Список фільмів завантажується на сторінку

Основний розвиток	Користувач переглядає список фільмів
Альтернативні розвитку	<p>Користувач переходить до однієї з наступних сторінок:</p> <ul style="list-style-type: none"> - рекомендаційна система на сторінці «Recommender»; - пошук фільму за назвою на сторінці «Search for a Movie»; - налаштування користувацького профіля на сторінці «User Settings»; - (якщо адміністратор) керування користувачами на сторінці «Admin»; - вихід з облікового запису; - покидання сайту. <p>Якщо користувач має привілеї адміністратора, то він також має доступ до наступних операцій:</p> <ul style="list-style-type: none"> - додавання нового фільму; - редагування даних фільму; - видалення фільму.
Виняткові розвитку	-

Таблиця 4.6 — Сценарій використання 5

Назва	Пошук інформації про фільм за введеною назвою
ID	5
Опис	Всі дані фільму знаходяться за його назвою, де певна кількість орфографічних помилок опрацьовується. Навідміну від списку фільмів у вкладці «Movies», знайдені фільми за назвою також показують стовпці «Franchise» та «Keywords»
Актори	Звичайний користувач, адміністратор

Частота користування	Середня
Тригери	Натискання на опцію «Search for a Movie» в лівому бічному меню
Передумови	Успішна аутентифікація
Постумови	Отримуються дані про фільм в якому назва найбільше збігається з назвою, яку ввів користувач
Основний розвиток	1) Користувач вводить назву фільму. 2) Користувач натискає кнопку «Search». 3) Система шукає фільм за назвою. 4) Система повертає найбільш схожий за назвою фільм.
Альтернативні розвитку	Користувач переходить до однієї з наступних сторінок: <ul style="list-style-type: none"> - рекомендаційна система на сторінці «Recommender»; - список фільмів на сторінці «Movies»; - налаштування користувацького профіля на сторінці «User Settings»; - (якщо адміністратор) керування користувачами на сторінці «Admin»; - вихід з облікового запису; - покидання сайту.
Виняткові розвитку	-

Таблиця 4.7 — Сценарій використання 6

Назва	Перегляд користувацького профіля
ID	6
Опис	Користувач дивиться на дані свого профіля
Актори	Звичайний користувач, адміністратор

Частота користування	Низька
Тригери	Натискання на опцію «User Settings» в лівому бічному меню
Передумови	Успішна аутентифікація
Постумови	Користувач бачить дані свого облікового запису
Основний розвиток	Користувач переглядає інформацію про себе
Альтернативні розвитку	<p>Користувач вирішив редагувати свої облікові дані (пошта, ім'я, пароль). Або, якщо не адміністратор, користувач вирішив видалити свій аккаунт. Або користувач переходить до однієї з наступних сторінок:</p> <ul style="list-style-type: none"> - рекомендаційна система на сторінці «Recommender»; - список фільмів на сторінці «Movies»; - пошук фільму за назвою на сторінці «Search for a Movie»; - (якщо адміністратор) керування користувачами на сторінці «Admin»; - вихід з облікового запису; - покидання сайту.
Виняткові розвитку	-

Таблиця 4.8 — Сценарій використання 7

Назва	Перегляд списку користувачів
ID	7
Опис	Користувач з адміністративними привілеями переглядає список всіх користувачів
Актори	Адміністратор
Частота користування	Середня

Тригери	Натискання на опцію «Admin» в лівому бічному меню
Передумови	Успішна аутентифікація
Постумови	Адміністратор отримує список користувачів
Основний розвиток	Адміністратор переглядає список користувачів
Альтернативні розвитку	<p>Адміністратор вирішив додати нового користувача. Або адміністратор вирішив відредагувати дані певного користувача (пошта, ім'я, пароль). Або адміністратор вирішив видалити певного користувача. Або користувач переходить до однієї з наступних сторінок:</p> <ul style="list-style-type: none"> - рекомендаційна система на сторінці «Recommender»; - список фільмів на сторінці «Movies»; - пошук фільму за назвою на сторінці «Search for a Movie»; - налаштування користувацького профіля на сторінці «User Settings»; - вихід з облікового запису; - покидання сайту.
Виняткові розвитку	-

Висновки до розділу 4

В даному розділі було виконано проектування схем для дипломного проекту. Було створено чотири кресленники, що містять наступні діаграми: компонентів, прецедентів, діяльності, та послідовності. Для деяких з кресленників було наведено опис.

Для проекту було обрано клієнт-серверну архітектуру, де сервер зберігає, надає та керує ресурсами та послугами, а клієнти взаємодіють з системою через мережу.

Також було наведено всі сценарії використання системи. Для найважливіших сценаріїв було відвідено окрему таблицю з детальним описом. Цими важливими сценаріями було визначено аутентифікацію, реєстрацію, використання рекомендаційної системи, перегляд списку фільмів, пошук інформації про фільм за введеною назвою, перегляд користувацького профіля, та перегляд списку користувачів.

					ІА01.020БАК.006 ПЗ	Арк.
						28
Зм.	Лист	№ докум.	Підпис	Дата		

5 ВИБІР ІНСТРУМЕНТІВ

Для розробки системи рекомендації фільмів було обрано наступні інструменти.

5.1 Вибір мов програмування

Python — це інтерпретована об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою. Високорівневі вбудовані структури даних у поєднанні з динамічною типізацією та динамічним зв'язуванням роблять дану мову дуже привабливою для швидкої розробки додатків, а також для використання як мови сценаріїв або з'єднувальної мови для поєднання існуючих компонентів. Простий, легкий для вивчення синтаксис Python підкреслює читабельність і, через це, знижує вартість обслуговування програми. Python підтримує модулі та пакети, чим заохочує модульність програми та повторне використання коду. Інтерпретатор Python і обширна стандартна бібліотека доступні у вихідному або бінарному вигляді безкоштовно для всіх основних платформ і можуть вільно поширюватися [12].

Python буде слугувати як головна мова програмування даного проєкту, якою буде виконана очистка даних, реалізація рекомендаційної системи та backend вебзастосунку з використанням фреймворку FastAPI.

Причини вибору:

- попередній досвід роботи;
- наявність найрізноманітніших високоякісних бібліотек та фреймворків для виконання поставлених задач.

TypeScript — це строго типізована мова програмування, яка ґрунтується на JavaScript і надає кращі інструменти для потреб будь-якого масштабу. Окрім цього, дана мова додає до JavaScript додатковий синтаксис для підтримки більш тісної інтеграції з вибраним середовищем розробки, що дозволяє ловити помилки на ранніх стадіях розробки. Код TypeScript перетворюється на JavaScript, який

					IA01.020БАК.006 ПЗ	Арк.
						29
Зм.	Лист	№ докум.	Підпис	Дата		

виконується будь-де, де працює JavaScript: у браузері, на Node.js або Deno тощо [13].

За допомоги TypeScript буде реалізовано frontend вебзастосунку.

Причини вибору:

- популярність з якої виходить достатня кількість відео-пояснень та інших навчальних матеріалів;
- навідміну від JavaScript, TypeScript це строго типізована мова програмування.

5.2 Перелік головних вебфреймворків та бібліотек проєкту

FastAPI — це сучасний, швидкий (високопродуктивний) вебфреймворк для створення API з допомогою Python на основі стандартних Python type hints [14].

Через FastAPI буде реалізовано backend вебзастосунку.

Причини вибору:

- попередній досвід роботи;
- автоматично згенерована API документація за шляхом <http://localhost:8000/docs#/> через Swagger UI;
- достатня популярність і тому задовільна кількість навчальних матеріалів.

Pandas — це швидкий, потужний, гнучкий та простий у використанні інструмент аналізу та обробки даних з відкритим кодом, побудований на основі мови програмування Python [15].

З допомогою Pandas, буде проведена очистка та перетворення даних. Потім на основі підготовлених даних і теж з використанням Pandas, буде створено рекомендаційну систему.

Причини вибору:

- дані для рекомендаційної системи знаходяться у табличному вигляді і Pandas є головним інструментом для роботи з даними в такому вигляді для мови програмування Python;
- попередній досвід роботи.

					ІА01.020БАК.006 ПЗ	Арк.
						30
Зм.	Лист	№ докум.	Підпис	Дата		

React — це декларативна, ефективна і гнучка JavaScript-бібліотека, призначена для створення інтерфейсів користувача. Вона дозволяє компонувати складні інтерфейси з невеликих окремих частин коду — “компонентів” [16].

Frontend вебзастосунку буде реалізований через дану бібліотеку.

Причина вибору: висока популярність і, що слідує, велика кількість навчальних матеріалів.

5.3 Вибір РСУБД

PostgreSQL — це потужна об’єктно-реляційна база даних з відкритим вихідним кодом, що використовує та розширює мову SQL у поєднанні з багатьма функціями, які безпечно зберігають і масштабують найскладніші дані. PostgreSQL почався у 1986 році як частина проєкту POSTGRES в Каліфорнійському університеті та Берклі і, відтоді, активно розвивається на основній платформі. PostgreSQL заслужив міцну репутацію завдяки своїй перевірній архітектурі, надійності, цілісності даних, перевіреному набору функцій, та відданості його спільноти, що регулярно створює продуктивні та іноваційні рішення. PostgreSQL працює на всіх поширених операційних системах, є сумісним з ACID з 2001 року, та має потужні додаткові модулі, такі як розширювач геопросторових баз даних PostGIS [17].

Дана РСУБД буде використовуватися для зберігання даних про фільми та користувачів.

Причини вибору:

- попередній досвід роботи;
- велика популярність і, як результат, наявність багатьох навчальних матеріалів.

5.4 Вибір середовища розроблення

Сама ж розробка дипломного проєкту буде відбуватися на VS Code. VS Code — це легкий, але потужний редактор вихідного коду, який доступний для Windows, macOS і Linux. Даний редактор має вбудовану підтримку JavaScript, TypeScript і Node.js. Також VS Code має багату екосистему розширень для інших мов і середовищ виконання (таких як C++, C#, Java, Python, PHP, Go, .NET тощо) [18].

Причина вибору: попередній досвід використання.

5.5 Додаткові інструменти

Jupyter Notebook (або IPython Notebook) – це інтерактивне обчислювальне середовище, в якому можна поєднувати виконання коду, форматований текст, математику, графіки, та мультимедіа [19].

Даний інструмент використовується для обробки даних та реалізації рекомендаційної функції.

Причини вибору:

- попередній досвід роботи;
- зручність у використанні у процесах очистки та перетворення даних через його покроковість.

Висновки до розділу 5

В даному розділі було обрано інструменти для реалізації проєкту та названо короткі характеристики цих інструментів, і причини їх вибору.

Мовами програмування проєкту будуть Python та TypeScript. Головними бібліотеками і фреймворками будуть FastAPI, Pandas, та React. РСУБД – PostgreSQL. Середовищем розробки – VS Code. Також для підвищення комфорту виконання обробки даних та створення рекомендаційної системи було вибрано Jupyter Notebook.

6 РОБОТА З ДАНИМИ ДЛЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

6.1 Характеристика обраного набору даних

Рекомендації будуть виконуватися на основі набору даних MovieLens [20]. Цей набір даних містить інформацію про 45,000 фільмів, які вийшли до липня 2017 року.

Даний набір даних складається з семи файлів на майже 1 ГБ пам'яті, проте буде використано тільки три файли з семи, що мають наступні назви і характеристики:

- movies_metadata.csv – головний файл, що містить стовпці з інформацією про бюджет; дохід; оригінальну мову; мови, якими говорили у фільмі; оригінальну назву фільму; теперішню назву фільму; TMDb popularity score; дату релізу фільма; poster path; tagline; країну виробництва і тд. Загалом, маємо двадцять чотири стовпці, більшість з яких буде пізніше видалено, оскільки вони не містять важливої інформації для виконання поставленої задачі;

- keywords.csv – файл, що містить ключові слова, що характеризують фільм. Наприклад, «Star Wars: Episode IV - A New Hope» має наступні ключові слова: android; rescue mission; rebellion; planet; space opera. Різні фільми мають різну кількість ключових слів. Також є фільми з унікальними ключовими словами, які дуже рідко зустрічаються. Ці проблеми будуть адресовані в наступному підрозділі;

- credits.csv – файл, що містить інформацію про акторів і знімальну групу фільмів. Займає найбільший обсяг пам'яті (190 МБ) порівняно з двома попередніми файлами: movies_metadata.csv (34 МБ) та keywords.csv (6 МБ). Як і попередні два файли, над цим файлом теж будуть виконані певні перетворення.

6.2 Обробка даних

З допомогою бібліотеки Pandas [15], обробляємо дані. Робота з даними розпочнеться з їх завантаження і попереднього огляду: В результаті матимемо дані у вигляді, показаному на рисунку 6.1.

					ІА01.020БАК.006 ПЗ	Арк.
						33
Зм.	Лист	№ докум.	Підпис	Дата		

	0	1
adult	False	False
belongs_to_collection	{'id': 10194, 'name': 'Toy Story Collection', ...}	NaN
budget	30000000	65000000
genres	[{'id': 16, 'name': 'Animation'}, {'id': 35, 'name': 'Comedy'}]	[{'id': 12, 'name': 'Adventure'}, {'id': 14, 'name': 'Fantasy'}]
homepage	http://toystory.disney.com/toy-story	NaN
id	862	8844
imdb_id	tt0114709	tt0113497
original_language	en	en
original_title	Toy Story	Jumanji
overview	Led by Woody, Andy's toys live happily in his room. When Woody finds out that his owner is moving, he and the other toys decide to go on an adventure to stay with him.	When siblings Judy and Peter discover an enchanted door in their attic, they are drawn into a world of adventure and danger.
popularity	21.946943	17.015539
poster_path	/rhIRbceoE9IR4veEXuwCC2wARtG.jpg	/vzmL6fP7aPKNKPRTFnZmiUfcyV.jpg
production_companies	[{'name': 'Pixar Animation Studios', 'id': 3}]	[{'name': 'TriStar Pictures', 'id': 559}, {'name': 'Columbia Pictures', 'id': 1}]
production_countries	[{'iso_3166_1': 'US', 'name': 'United States of America'}]	[{'iso_3166_1': 'US', 'name': 'United States of America'}]
release_date	1995-10-30	1995-12-15
revenue	373554033.0	262797249.0
runtime	81.0	104.0
spoken_languages	[{'iso_639_1': 'en', 'name': 'English'}]	[{'iso_639_1': 'en', 'name': 'English'}, {'iso_639_1': 'es', 'name': 'Spanish'}]
status	Released	Released
tagline	NaN	Roll the dice and unleash the excitement!
title	Toy Story	Jumanji
video	False	False
vote_average	7.7	6.9
vote_count	5415.0	2413.0

Рисунок 6.1 — Перші два фільми набору даних

На рисунку 6.2 показана коротка характеристика кожного стовпця набору даних, в якій вказується кількість not-null значень та тип даних кожного стовпця.

					ІА01.020БАК.006 ПЗ	Арк.
						34
Зм.	Лист	№ докум.	Підпис	Дата		


```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45466 entries, 0 to 45465
Data columns (total 24 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   adult                                45466 non-null  object
1   belongs_to_collection               4494 non-null   object
2   budget                              45466 non-null  object
3   genres                              45466 non-null  object
4   homepage                            7782 non-null   object
5   id                                  45466 non-null  object
6   imdb_id                             45449 non-null  object
7   original_language                   45455 non-null  object
8   original_title                       45466 non-null  object
9   overview                             44512 non-null  object
10  popularity                           45461 non-null  object
11  poster_path                         45080 non-null  object
12  production_companies                45463 non-null  object
13  production_countries                 45463 non-null  object
14  release_date                        45379 non-null  object
15  revenue                             45460 non-null  float64
16  runtime                             45203 non-null  float64
17  spoken_languages                    45460 non-null  object
18  status                              45379 non-null  object
19  tagline                             20412 non-null  object
20  title                               45460 non-null  object
21  video                               45460 non-null  object
22  vote_average                         45460 non-null  float64
23  vote_count                           45460 non-null  float64
dtypes: float64(4), object(20)
memory usage: 8.3+ MB

```

Рисунок 6.2 — Характеристика набору даних

Лишати таку кількість фільмів з 24 стовпцями-характеристиками не є доцільно оскільки більшість фільмів з даного набору є маловідомими і їхня наявність тільки сповільнить роботу програми, не приносячи великої користі.

Тому першим ділом буде проведена фільтрація. Критеріями фільтрації поставимо наявність менше 40 голосів або середню оцінку, що має значення нижче 4:

```
# Calculate the count of films that have vote_average < 4 or
len(df[(df['vote_average'] < 4) | (df['vote_count'] < 40)])

35109
```

Рисунок 6.3 — Кількість фільмів, що відповідають умовам вище

Таким чином було відкинуто 77% фільмів набору даних. Тепер буде проведено аналіз стовпців. З них теж більшість не є важливими для надання рекомендацій. Наприклад, стовпці `spoken_languages` та `original_language` будуть видаленими тому що вони здебільшого містять англійську мову.

В свою чергу, стовпці `adult`, `status`, та `video` у 99% випадків мають одне домінуюче значення. На рисунку 6.4 показано підрахунок даних для стовпця `video`. Через це, дані стовпці теж варто видалити.

```
df["video"].value_counts()
```

✓ 0.0s

```
video
False    10348
True         3
Name: count, dtype: int64
```

Рисунок 6.4 — Приклад стовпця з одним домінуючим значенням

Застосувавши подібну логіку до решти стовпців, в результаті було отримано набір даних, де фільми мають наступні 7 характеристик:

– `belongs_to_collection` – чи фільм належить до певної франшизи. Наприклад, фільм «Star Wars: Episode IV - A New Hope» належить до франшизи «Star Wars»;

– genres – до яких жанрів належить фільм. Приміром, фільм «Star Wars: Episode IV - A New Hope» - Action, Adventure та Science Fiction;

– id – унікальний ідентифікатор, завдяки якому пізніше буде додані дані з credits.csv та keywords.csv;

– release_date – дата, коли вийшов фільм;

– title – назва фільму;

– vote_average – рейтинг фільму від 4 до 10;

– vote_count – кількість оцінок фільму. Наприклад, «Star Wars: Episode IV - A New Hope» має 6778 голосів.

На рисунку 6.5 показано стовпці, що лишилися.

	0	1
belongs_to_collection	{'id': 10194, 'name': 'Toy Story Collection', ...}	NaN
genres	[{'id': 16, 'name': 'Animation'}, {'id': 35, 'name': 'Adventure'}, {'id': 14, 'name': 'Fantasy'}, ...]	[{'id': 12, 'name': 'Adventure'}, {'id': 14, 'name': 'Fantasy'}, ...]
id	862	8844
release_date	1995-10-30	1995-12-15
title	Toy Story	Jumanji
vote_average	7.7	6.9
vote_count	5415.0	2413.0

Рисунок 6.5 — Стовпці, які лишилися після очистки

Дані стовпці мають наступні типи даних:

– belongs_to_collection – object;

– genres – object;

– id – object;

– release_date – object;

– title – object;

– vote_average – float64;

– vote_count – float64.

Стовпці belongs_to_collection та genres будуть опрацьовані пізніше. Першим опрацьованим стовпцем стане release_date. Його тип даних буде змінено з object на datetime, де в поля з помилками буде ставитися NaT (Not a Time). Потім буде пораховано скільки рядків містять NaT. На рисунку 6.6 показано очистку даного стовпця:

```
# Convert 'release_date' column to datetime type
df['release_date'] = pd.to_datetime(df['release_date'], errors='coerce')
# Count the number of rows with bad date values
bad_date_count = df['release_date'].isnull().sum()
print(f"Number of rows with bad date values: {bad_date_count}")
```

Py

Number of rows with bad date values: 2

Рисунок 6.6 — Обробка стовпця «release_date»

На рисунку 6.6 показана кількість рядків з NaT. Оскільки 2 рядки з 10,000 становлять дуже незначну кількість, їх було відкинуто. Після цього було змінено типи даних стовпців vote_count та id на integer, перед тим замінивши NaN (Not a Number) на 0

Розпочато роботу зі стовпцями belongs_to_collection та genres. Ці стовпці містять дані у вигляді словників та масивів словників, відповідно. На рисунку 6.7 можна побачити як виглядають дані з стовпця genres.

Genres	
[{'id': 16, 'name': 'Animation'}, {'id': 35, 'name': 'Comedy'}, {'id': 10751, 'name': 'Family'}]	
[{'id': 12, 'name': 'Adventure'}, {'id': 14, 'name': 'Fantasy'}, {'id': 10751, 'name': 'Family'}]	
[{'id': 10749, 'name': 'Romance'}, {'id': 35, 'name': 'Comedy'}]	
[{'id': 35, 'name': 'Comedy'}, {'id': 18, 'name': 'Drama'}, {'id': 10749, 'name': 'Romance'}]	
[{'id': 35, 'name': 'Comedy'}]	

Рисунок 6.7 — Вид даних стовпця «genres»

Обробляємо стовпець genres функцією literal_eval для перетворення рядків у форматі JSON на масиви словників. Одинарні лапки було замінено на подвійні для забезпечення коректного формату. Потім з кожного словника масиву було отримано значення ключа name і створено масив з назвами жанрів, який було відсортовано в алфавітному порядку.

Далі перевіряється наявність непридатних значень у стовпці genres. Виявлено погані значення: Music, TV Movie і тд. Для вирішення проблеми, було визначено перелік відомих жанрів (Action тощо), а решту відкинемо. До переліку відомих жанрів фільмів було обрано наступні жанри: Action; Adventure; Animation; Comedy; Crime; Documentary; Drama; Family; Fantasy; History; Horror; Mystery; Romance; Science Fiction; Thriller; War; Western.

Наприкінці, масив жанрів перетворюється в тип даних STRING, де кома буде слугувати сепаратором:

Після цього відбувається обробка стовпця belongs_to_collection. Для цього визначається функція extract_franchise_name, що перетворює рядок у словник за допомогою функції literal_eval і витягує значення ключа name. При виникненні помилки ValueError чи TypeError, буде повертатися значення None.

За допомогою функції apply, створена вище функція extract_franchise_name застосовується до стовпця belongs_to_collection, а результат зберігається у новому стовпці franchise. Після цього, назви франшиз обробляються командою str.replace() для видалення слова collection з кінця назв.

Після обробки стовпця belongs_to_collection, було змінено порядок стовпців. Тепер стовпець id буде першим, title – другим, franchise – третім, release_date – четвертим, genres – п'ятим, vote_average – шостим, а vote_count – сьомим.

Всі наявні стовпці було оброблено, проте робота з даними ще не закінчена. Дані з файлів credits.csv та keywords.csv було завантажено та об'єднано з існуючим DataFrame за спільним стовпцем id, використовуючи метод merge.

Як результат, було отримано доступ до трьох нових стовпців: cast, crew, та keywords, над якими буде виконано певні перетворення.

Робота з новими даними розпочнеться зі стовпця crew. Даний стовпець містить інформацію про знімальну групу фільму. З цього стовпця потрібно буде отримати лише ім'я кінорежисера. Для цього визначається схожа до попередніх функція, що ітерує через список членів знімальної групи доки не знайде кінорежисера, і далі повертає його ім'я в новий стовпець з назвою director.

На рисунку 6.8 показано як було оброблено даний стовпець.

```
def extract_director(crew_list):
    for crew_member in crew_list:
        if crew_member["job"] == "Director":
            return crew_member["name"]
    return None

# For function get_director to work, convert
# the string representations to actual dictionaries
df["crew"] = df["crew"].apply(literal_eval)
# Extract the director's name for each movie
df["director"] = df["crew"].apply(extract_director)
```

Рисунок 6.8 — Обробка стовпця «crew»

Далі буде оброблено стовпець cast. Даний стовпець містить дані про акторів фільму. Проте не має сенсу брати всіх акторів, оскільки маловідомі актори на другорядних ролях рідко мають великий вплив на успіх фільму. Через це, було обрано лише перших трьох акторів на головних ролях тому що високий шанс, що найкращі чи найвідоміші актори будуть перебувати саме там.

Залишилося опрацювати тільки стовпець keywords. Його буде опрацьовано подібно до попередніх стовпців з словниками з тою відмінністю, що буде підраховано скільки зустрічається кожне ключове слово. Після цього будуть відкинуто ключові слова з підрахунком менше 10. Таким чином буде підвищена ефективність і швидкість роботи рекомендаційної системи з мізерною втратою точності. На рисунку 6.9 можна помітити скільки було відкинуто ключових слів.

```
# Print the number of unique keywords
num_unique_keywords_before = len(keyword_counts)
print(f"Number of unique keywords before filtering: {num_unique_keywords_before}")
num_unique_keywords_after = len(sorted_keywords_filtered)
print(f"Number of unique keywords after filtering: {num_unique_keywords_after}")
```

```
Number of unique keywords before filtering: 13953
Number of unique keywords after filtering: 1553
```

Рисунок 6.9 — Підрахунок ключових слів до і після фільтрування

Зберігаємо відфільтровані ключові слова в окремій множині, на яку буде здійснено посилання при ітерації даних в стовпці keywords. Тільки ті ключові слова, що лежать в тій множині, будуть прийняті. Окрім того, встановлюється обмеження на максимальну кількість ключових слів - не більше п'яти.

```
<class 'pandas.core.frame.DataFrame'>
Index: 10342 entries, 0 to 10512
Data columns (total 10 columns):
#   Column          Non-Null Count  Dtype
---  -
0   id               10342 non-null  int64
1   title            10342 non-null  object
2   franchise        2134 non-null   object
3   release_year     10342 non-null  object
4   genres           10325 non-null  object
5   vote_average     10342 non-null  float64
6   vote_count       10342 non-null  int32
7   director         10325 non-null  object
8   top_actors       10342 non-null  object
9   keywords         10342 non-null  object
dtypes: float64(1), int32(1), int64(1), object(7)
memory usage: 848.4+ KB
```

Рисунок 6.10 — Характеристика обробленого набору даних

Всі дані було оброблено і тепер настав час на завершальні дії, а саме: відкидання непотрібних вже стовпців id, cast, crew та keywords, отримання року з

дати релізу, перейменування стовпців `filtered_keywords` та `release_date`, видалення рядків-дублікатів, та додавання нового індекса першим стовпцем.

Старий індексний стовпець було видалено оскільки значення в ньому містили багато пропусків. Тобто міг бути рядок з індексом 862, за яким слідував рядок з індексом 8844.

Як результат, отримано набір даних з характеристиками, показаними на рисунку 6.10:

Оброблені дані було збережено в .csv файлі з назвою `recommender_data`. На основі даних з цього файла будуть надаватися рекомендації системою, яка буде розроблена в наступному розділі.

Висновки до розділу 6

В даному розділі було охарактеризовано набір даних на основі якого будуть надаватися рекомендації, з вказанням які його частини будуть використовуватися і чому.

Після цього дані з набору вище було оброблено і підготовлено для рекомендаційної системи. Було відкинуто лишні стовпці та рядки, змінено типи даних на потрібні, виокремлено та відфільтровано бажані дані з масивів словників, видалено рядки-дублікати, змінено порядок стовпців тощо.

В результаті усіх цих перетворень, було отримано набір даних для рекомендаційної системи з задовільною якістю і повнотою даних для рекомендаційної системи.

7 РЕАЛІЗАЦІЯ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

У даному розділі буде створена рекомендаційна система фільмів, що будуватиме свої рекомендації на основі даних попереднього розділу. Дану систему можна розбити на 4 основних компоненти:

- функція `find_movie`;
- функція `preprocess`;
- функція `calculate_cosine_sim`;
- функція `recommender`.

7.1 Робота над рекомендаційною системою

На рисунку 7.1 показано зчитування даних, які було отримано в попередньому розділі.

```
df = pd.read_csv("data/recommender_data.csv", index_col=0)
df.head(2).transpose()
```

✓ 0.0s

id	1	2
title	Toy Story	Jumanji
franchise	Toy Story	NaN
release_year	1995	1995
genres	Animation, Comedy, Family	Adventure, Family, Fantasy
vote_average	7.7	6.9
vote_count	5415	2413
director	John Lasseter	Joe Johnston
top_actors	Tom Hanks, Tim Allen, Don Rickles	Robin Williams, Jonathan Hyde, Kirsten Dunst
keywords	jealousy, toy, boy, friendship, friends	disappearance, based on children's book

Рисунок 7.1 — Зчитування даних

Реалізацію рекомендаційної системи фільмів буде розпочато зі зчитування назви фільму, введеної користувачем, і перевірка її наявності в базі даних. У процесі введення користувачем назви фільму можуть виникати різні проблеми, зокрема орфографічні помилки або незначні розбіжності в написанні. Щоб забезпечити точне виявлення фільму в базі даних, навіть за наявності таких помилок, використовується бібліотека fuzzywuzzy.

Бібліотека fuzzywuzzy - це бібліотека для fuzzy string matching в Python. Вона дозволяє порівнювати рядки, які не повністю співпадають, і знаходити найбільш схожі. FuzzyWuzzy заснована на алгоритмах, які використовуються в прикладних програмах розпізнавання тексту та інформаційного пошуку [21-22].

У контексті даного дипломного проєкту, ця бібліотека використовує алгоритми нечіткого пошуку для знаходження найбільш схожих рядків у наборі даних. Це дозволяє системі ефективно обробляти орфографічні помилки та невідповідності у введених користувачем назвах фільмів.

Дану бібліотеку було використано у функції `find_movie`, яка приймає на вхід назву фільму, введenu користувачем, і повертає інформацію про фільм, який найбільш точно відповідає введеному рядку. Ця функція має наступну реалізацію:

1) зчитування всіх назв фільмів функцією `tolist()`;

2) нечіткий пошук найбільш схожої назви. Бібліотека fuzzywuzzy застосовується для пошуку найбільш схожої назви фільму в списку `all_titles`. Метод `process.extractOne(input, all_titles)` повертає кортеж, де першим елементом є найбільш схожа назва;

3) фільтрація `DataFrame` за знайденою назвою для отримання рядка з даними про фільм.

Наступним кроком реалізації рекомендаційної системи буде обробка тексту з використанням NLTK. NLTK – це провідна платформа для створення програм Python для роботи з даними людської мови. Дана платформа надає прості у використанні інтерфейси для більш ніж 50 корпусів і лексичних ресурсів, таких як WordNet, а також набору бібліотек для обробки тексту для класифікації, токенизації, тагування тощо [23]:

З використанням NLTK і теорії з книг [24-25] буде реалізована функція `preprocess`. Дана функція виконує обробку введеного тексту для подальшого аналізу та використання в системі рекомендацій фільмів. Даний процес включає в себе видалення небуквених символів, переведення тексту в нижній регістр, токенизацію та фільтрацію стоп-слів. Ці кроки забезпечують нормалізацію тексту, що в свою чергу покращує точність моделей машинного навчання та пошукових алгоритмів.

Функція `preprocess` використовується всередині іншої функції - `calculate_cosine_sim`. Для реалізації даної функції потрібно буде використати бібліотеку `Scikit-learn`. `Scikit-learn` – це бібліотека машинного навчання з відкритим кодом, яка підтримує навчання з і без вчителя Також ця бібліотека надає різні інструменти для роботи з моделями, попередньої обробки даних, вибору та оцінки. моделей і тд [26].

З використанням бібліотеки `Scikit-learn`, функція `calculate_cosine_sim` обчислює матрицю косинусної схожості для `DataFrame`, який містить дані про фільми. Косинусна схожість використовується для визначення ступеня подібності між фільмами на основі їхніх характеристик, що дозволяє системі рекомендацій знаходити фільми, схожі на введений користувачем фільм.

Стовпцями, на основі даних яких виконуватимуться рекомендації, було обрано "franchise", "director", "top_actors", "genres" та "keywords", оскільки вони містять важливі характеристики фільмів. Враховуючи велику роль кінорежисера під час створення фільмів, стовпець "director" було згадано двічі, заради надання йому більшої ваги при розрахунку схожості. Таким чином, було збільшено вплив кінорежисера на кінцевий результат. Решта ж стовпців згадується тільки один раз.

Всі вище реалізовані функції об'єднані разом дозволяють нарешті створити рекомендаційну функцію `recommender`.

Функція `recommender` призначена для надання рекомендацій схожих фільмів на основі введеної користувачем назви фільму. Вона знаходить найбільш схожі фільми та повертає інформацію про них у вигляді `DataFrame`.

Дана функція виконує наступні дії:

					IA01.020БАК.006 ПЗ	Арк.
						45
Зм.	Лист	№ докум.	Підпис	Дата		

1) пошук фільму за введеним заголовком користуючись функцією `find_movie()`;

2) отримання інформації про знайдений фільм;

3) визначення індексу фільму. Дані будуть отримуватися з бази даних, де індексація починається з 1. В свою чергу в Pandas DataFrame індекс починається з 0 [27]. Через це, одиниця буде віднята від індекса, заради того, щоб відповідати індексації у DataFrame;

4) обчислення схожості фільмів на основі попередньо обчисленої матриці косинусної схожості `cosine_sim`;

5) сортування оцінок схожості в порядку спадання;

6) повернення найкращих рекомендацій.

На рисунку 7.2 зображена робота рекомендаційної системи з вхідними даними «fellashi ring».

```
recommender("fellashi ring")
✓ 0.2s
```

For the input 'fellashi ring', the closest match is 'The Lord of the Rings: The Fellowship of the Ring' from 2001.

'The Lord of the Rings: The Fellowship of the Ring' from 2001 has the following characteristics used in the recommender system:

franchise: The Lord of the Rings.
director: Peter Jackson.
top actors: Elijah Wood, Ian McKellen, Cate Blanchett.
genres: Action, Adventure, Fantasy.
keywords: elves, dwarves, based on novel, mountain, fireworks.

Similar movies to 'The Lord of the Rings: The Fellowship of the Ring' from 2001:

	title	franchise	release_year	genres	vote_average	vote_count	director	top_actors
id								
3017	The Lord of the Rings: The Two Towers	The Lord of the Rings	2002	Action, Adventure, Fantasy	8.0	7641	Peter Jackson	Elijah Wood, Ian McKellen, Viggo Mortensen
3487	The Lord of the Rings: The Return of the King	The Lord of the Rings	2003	Action, Adventure, Fantasy	8.1	8226	Peter Jackson	Elijah Wood, Ian McKellen, Viggo Mortensen
7174	The Hobbit: An Unexpected Journey	The Hobbit	2012	Action, Adventure, Fantasy	7.0	8427	Peter Jackson	Ian McKellen, Martin Freeman, Richard Armitage

Рисунок 7.2 — Демонстрація роботи рекомендаційної системи в jupyter notebook

Висновки до розділу 7

У цьому розділі було розглянуто реалізацію ключових функцій рекомендаційної системи, а саме `find_movie`, `preprocess`, `calculate_cosine_sim`, і власне головну функцію `recommender`, що поєднує всі попередні функції в собі. Всі ці функції відіграють важливу роль у побудові рекомендаційної системи і разом дозволяють надавати рекомендації схожих фільмів на основі введеної користувачем назви фільму.

Функція `find_movie` за допомогою бібліотеки `FuzzyWuzzy` здатна обробляти орфографічні помилки та невідповідності у введених назвах. Попередня обробка тексту з використанням `NLTK` у функції `preprocess` забезпечує нормалізацію тексту для подальшого аналізу. Обчислення матриці косинусної схожості за допомогою `Scikit-learn` всередині функції `calculate_cosine_sim` дозволяє визначити ступінь подібності між фільмами на основі їхніх характеристик.

В порядку згадування, функція `recommender` використовує вищезгадані функції для видачі рекомендацій користувачам.

В результаті виконання даного розділу, було створено працюючу рекомендаційну систему, що здатна рекомендувати схожі фільми як це показано на рисунку 7.2.

8 ВЕБЗАСТОСУНОК

На рисунку 8.1 показано структуру дипломного проєкту. Головними папками є backend, frontend, та DA_skill_showcase. Зміст крайніх папок є зрозумілим, тоді як в середній папці знаходиться jupyter notebook в якому було оброблено дані і розроблено рекомендаційну систему. Також в цій папці зберігаються оброблені дані, які будуть завантажені в базу даних через спеціальний скрипт.

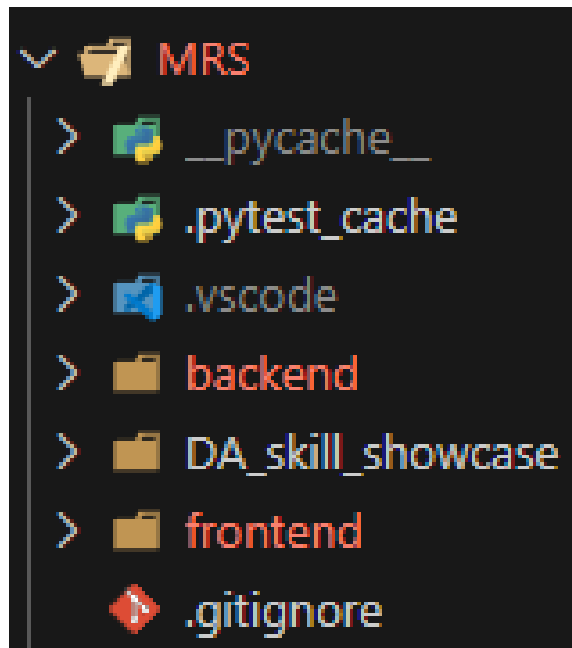


Рисунок 8.1 — Структура застосунку

У наступних двох підрозділах буде розглянуто вміст папок backend та frontend вебзастосунку більш детально. Огляд розпочнеться з папки backend.

8.1 Backend

Папка «backend» складається з двох підпапок: app та DB_table_creation. Папка DB_table_creation містить в собі скрипт, який створює таблиці бази даних і заповнює їх обробленими даними про фільми, отримані в 6 розділі. Окрім цього, також створюється аккаунт адміністратора. Цей скрипт корисний тим, що суттєво пришвидшує розробку застосунку, нівелюючи потребу самотужки повторювати

					ІА01.020БАК.006 ПЗ	Арк.
						48
Зм.	Лист	№ докум.	Підпис	Дата		

перелік одних і тих самих кроків кожного разу коли виникає потреба скинути дані до стандартних.

У другій папці `app` містяться інші папки, кожна з яких існує для виконання певної цілі. Папка `core` містить основні файли для налаштування та забезпечення роботи додатку, включаючи `config.py`, `db.py` та `security.py`. Папка `recommender` містить файли, необхідні для реалізації рекомендаційної системи, включаючи допоміжні функції (`utils` з файлами `calc_cosine_sim.py` та `find_movie.py`), а також основні файли для роботи рекомендаційної системи (`recommender.py`).

Така структура дозволяє організувати серверну частину додатку на фреймворку FastAPI із забезпеченням зручного управління конфігурацією, безпекою, базою даних та рекомендаційною системою.

База даних вебзастосунку складається з двох таблиць, що вміщують оброблені дані для рекомендаційної системи і дані користувачів. Ці дві таблиці матимуть наступну структуру, показану в таблицях 8.1— 8.2.

Таблиця 8.1 — Користувачі

Назва стовпця	Тип даних	Опис
<code>id</code>	Integer	Ідентифікатор
<code>email</code>	String	Електронна пошта
<code>full_name</code>	String	Ім'я
<code>is_active</code>	Bool	Чи аккаунт користувача активний. Якщо ні – то користувач не зможе зайти в свій аккаунт
<code>is_superuser</code>	Bool	Чи користувач володіє адміністративними привілеями

Таблиця 8.2 — Фільми

Назва стовпця	Тип даних	Опис
<code>id</code>	Integer	Ідентифікатор
<code>title</code>	String	Назва фільму
<code>franchise</code>	String	Франшиза, до якої належить фільм

release_year	String	Рік виходу фільму
genres	String	Жанри до яких відноситься фільм
vote_average	Float	Середнє арифметичне відгуків за фільм
vote_count	Integer	Кількість відгуків за фільм
director	String	Кінорежисер фільму
top_actors	String	Головні актори фільму
keywords	String	Ключові слова фільму

Для фільмів API маршрути знаходиться за `/api/v1/items/`. Шлях прописаний за `items/`, а не за `movies/` тому що це додає зручний рівень абстракції. Разом з CRUD для фільмів за цим маршрутом також знаходиться рекомендаційна система. На рисунку 8.3 показано вищезгадані шляхи:

Для користувачів маршрут має наступний вигляд: `/api/v1/users/`.

8.2. Frontend

У даному підрозділі буде розглянуто вміст підпапки `frontend` з назвою `src`, в якій і знаходиться більшість коду `frontend` частини.

У папці `assets` знаходяться створені за допомогою онлайн інструменту [28] логотип з текстом «MRS» і іконка вебсторінки. Папка `client` містить в собі моделі даних, схеми та сервіси, які використовуються в додатку. Папка `components` містить компоненти для різних функцій додатку. Вона включає компоненти для управління користувачами адміністраторами, загальні компоненти, які використовуються в різних частинах додатку, компоненти для додавання та редагування елементів, а також компоненти для налаштування користувацьких облікових записів. Папка `routers` містить файли, що визначають маршрути додатку, включаючи `_layout`, `__root.tsx`, `_layout.tsx`, `login.tsx`, `admin.tsx`, `items.tsx`, `recommender.tsx` та інші.

8.3 Тестування вебзастосунку і рекомендаційної системи

Даний застосунок реалізований таким чином, щоб при першому входженні в додаток, користувач мусив пройти обов'язкову процедуру входження в свій аккаунт, як показано на рисунку 8.2.

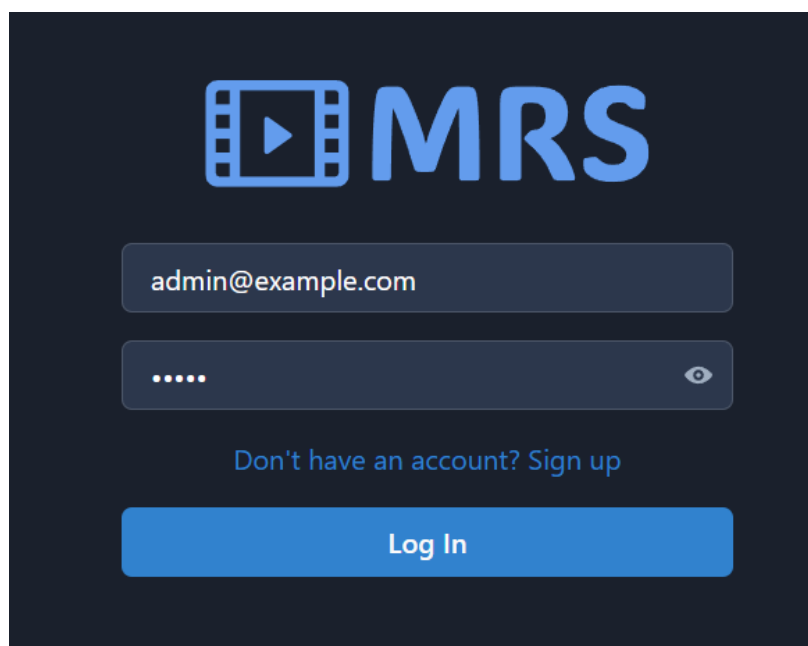


Рисунок 8.2 — Сторінка аутентифікації

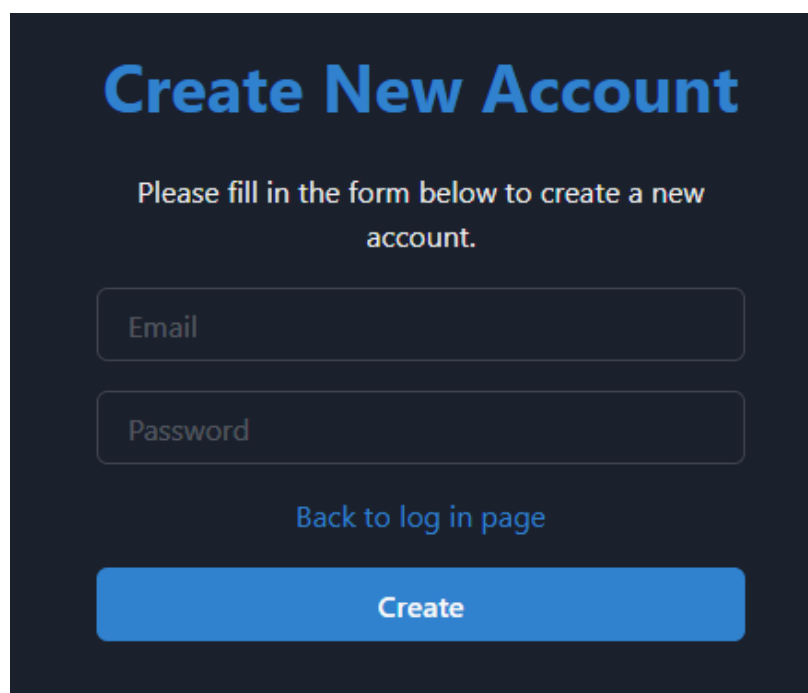


Рисунок 8.3 — Сторінка реєстрації

Якщо ж це перше використання застосунку, то існує опція створити новий аккаунт після натискання на «Don't have an account? Sign up», після натискання якої користувач потрапить на сторінку реєстрації як це показано на рисунку 8.3.

Після успішної аутентифікації, користувач бачить наступний, зображений на рисунку 8.4, інтерфейс.

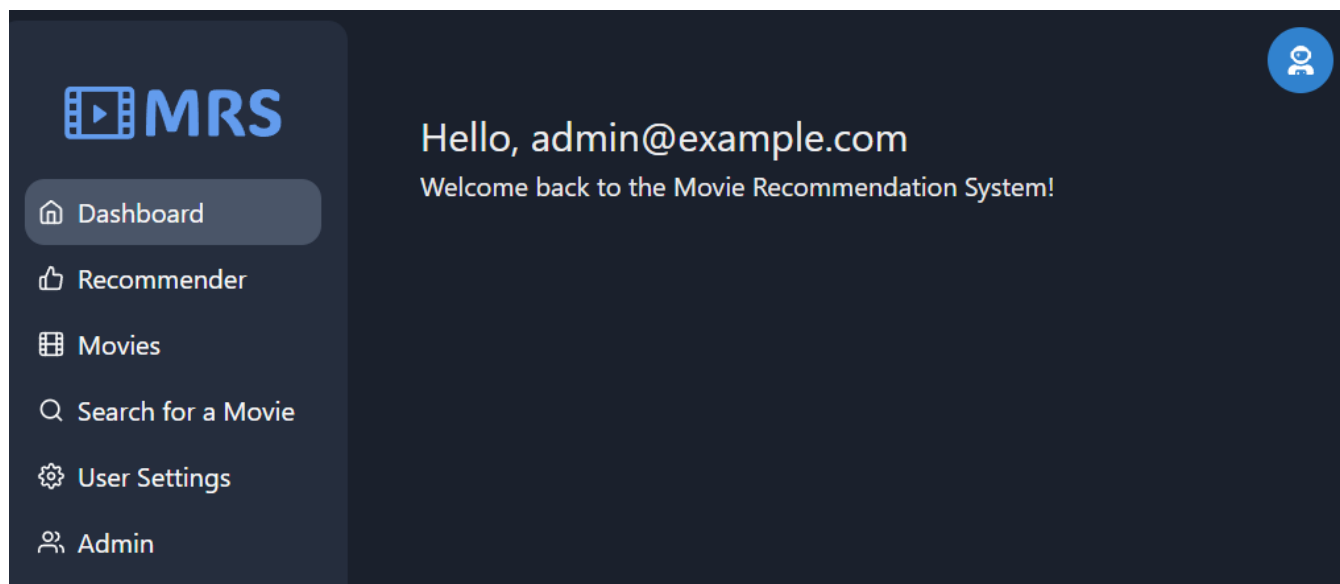


Рисунок 8.4 — Сторінка після login

Далі на вибір користувача є наступні дії:

- ввести назву фільму на основі якої отримати рекомендацію схожих фільмів у вкладці «Recommender»;
- переглянути список останніх доданих фільмів, і, якщо користувач володіє адміністративними привілеями, виконати операції додавання, редагування, та видалення окремих фільмів у вкладці «Movies»;
- пошук фільму за назвою, з отриманням більш детальної інформації порівняно до вкладки «Movies», у вкладці «Search for a Movie»;
- переглянути користувацький профіль з можливістю оновити власні дані включно з ім'ям, електронною поштою, та паролем у вкладці «User Settings». Також тут звичайний користувач (без адміністративних привілеїв) може видалити свій аккаунт;

					ІА01.020БАК.006 ПЗ	Арк.
						52
Зм.	Лист	№ докум.	Підпис	Дата		

— за наявністю адміністративних привілеїв, користувач може переглянути список всіх користувачів з можливістю додавання нових, редагування і видалення існуючих у вкладці «Admin».

На рисунку 8.5 показано, що знаходиться всередині вкладки «Movies». За наявністю адміністративних привілеїв, тут є можливість додати, редагувати, та видаляти фільми.

Movies Management						
+ Add Item						
TITLE	YEAR	GENRES	VOTE AVG	VOTE COUNT	DIRECTOR	TOP ACTORS
Cadet Kelly	2002	Comedy	5.2	145	Larry Shaw	Hilary Duff, Christy Carlson Romano, Gary Cole
In a Heartbeat	2017	Animation, Comedy, Family, Rom...	8.3	146	Beth David	N/A
The Visitors: Bastille Da...	2016	Comedy	4	167	Jean-Marie Poiré	Jean Reno, Christian Clavier, Franck Dubosc
With Open Arms	2017	Comedy	5.2	94	Philippe de Chauveron	Christian Clavier, Ary Abittan, Elsa Zylberstein
Good Guys Go to Heaven, B...	2016	Comedy	5.3	153	Franck Gastambide	Ramzy Bedia, Malik Bentalha, Franck Gastambide

Рисунок 8.5 — Вміст вкладки «Movies»

На рисунку 8.6 можна побачити, що знаходиться всередині вкладки «Search for a Movie». В даній вкладці можна знайти більш детальну інформацію про фільм, порівняно з інформацією про фільми у вкладці «Movies». Тобто тут також показують додаткові стовпці такі як keywords та franchise. Також функція пошуку реалізована таким чином, що деяка кількість помилок в написанні назви фільму опрацьовується.

На рисунку 8.7 показано вміст вкладки «Admin». Тут знаходиться перелік зареєстрованих користувачів системи. Дана вкладка доступна тільки адміністратору.

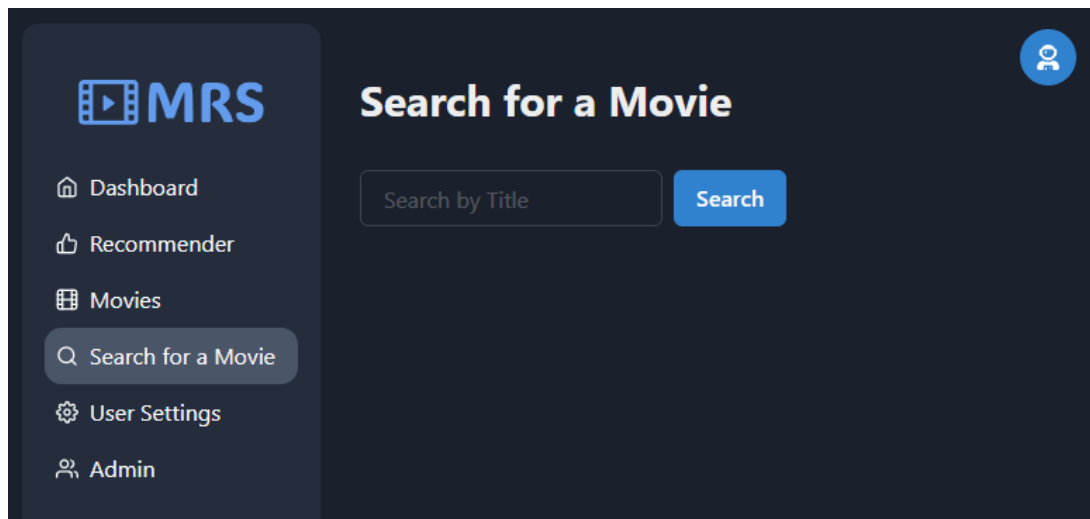


Рисунок 8.6 – Вміст вкладки «Search for a Movie»

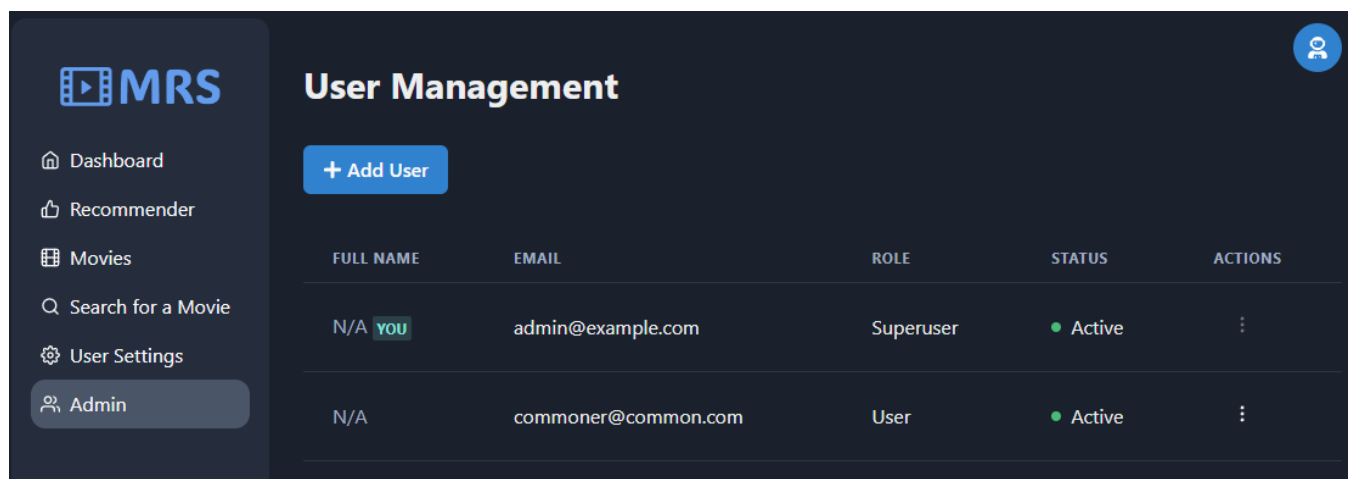


Рисунок 8.7 — Вміст вкладки «Admin»

На рисунку 8.8 показано меню створення нового користувача. Щоб потрапити до цього меню, потрібно натиснути на кнопку «Add User». Дану кнопку можна побачити на рисунку 8.7.

Тепер буде проведено тестування рекомендаційної системи. Для цього було натиснено на вкладку «Recommender», а у ввідне поле передано неповну назву з помилками «empare strke». Таким чином разом з рекомендаційною системою буде ще протестовано функцію `find_movie` і бібліотеку `fuzzywuzzy` на здатність знаходити найбільш схожий фільм за вхідними даними з помилками. На введені дані «empire strke» очікуваним результатом буде фільм 1980 року з назвою «The Empire Strikes Back».

Add User

✕

Email *

Email

Email is required

Full name

Full name

Set Password *

Password

Password is required

Confirm Password *

Password

☐ Is superuser?
☐ Is active?

Save

Cancel

Рисунок 8.8 — Створення нового користувача

На рисунку 8.9 можна побачити результат роботи рекомендаційної системи. Видно, що система правильно визначила, що під «empare strike» мається на увазі «The Empire Strikes Back».

Movie Recommender

empare strike

Recommend

For the typed in input title, the closest match is "The Empire Strikes Back":

TITLE	FRANCHISE	YEAR	GENRES	VOTE AVG	VOTE COUNT	DIRECTOR	TOP ACTORS
The Empire Strikes Back	Star Wars	1980	Action, Adventure, S...	8.2	5998	Irvin Kershner	Mark Hamill, Harrison Ford, Carrie Fische...

Here are recommended movies based on the movie title you entered:

TITLE	FRANCHISE	YEAR	GENRES	VOTE AVG	VOTE COUNT	DIRECTOR	TOP ACTORS
Star Wars	Star Wars	1977	Action, Adventure, S...	8.1	6778	George Lucas	Mark Hamill, Harrison Ford, Carrie Fische...
Return of the Jedi	Star Wars	1983	Action, Adventure, S...	7.9	4763	Richard Marquand	Mark Hamill, Harrison Ford, Carrie Fische...
RoboCop 2	RoboCop	1990	Action, Adventure, C...	5.6	498	Irvin Kershner	Peter Weller, Nancy Allen, Tom Noonan

Рисунок 8.9 — Результат роботи рекомендаційної системи на ввідні дані «empare strike»

Щодо наданих рекомендацій до фільму «The Empire Strikes Back», то вони теж є задовільними. Перші два фільми разом з введеним належать до оригінальної трилогії. Третій ж фільм, хоча й має найменшу схожість, проте також розділяє однакового кінорежисера та жанри.

Висновки до розділу 8

У цьому розділі було загально охарактеризовано вміст та функції папок backend, DA_skill_showcase, та frontend. Було показано, як організовано серверну частину додатку за допомогою FastAPI та інших бібліотек, а також розглянуто структуру бази даних та маршрути для роботи з користувачами і фільмами. У папці frontend було описано структуру коду, компоненти та маршрути, що, на основі React та допоміжних бібліотек, забезпечують функціональність користувацького інтерфейсу.

Після аутентифікації користувач має доступ до функцій перегляду та редагування фільмів, отримання рекомендацій, управління своїм профілем тощо.

Результати тестування показали задовільну точність роботи рекомендаційної системи та коректність реалізації усіх функцій, що свідчить про ефективність та надійність розробленого вебзастосунку.

ВИСНОВКИ

У даному дипломному проєкті на тему "Система Рекомендації Фільмів" було розглянуто та реалізовано комплексний підхід до створення рекомендаційної системи, яка допомагає користувачам знаходити схожі фільми до того фільму, назву якого вони ввели в систему.

Було розглянуто основні типи рекомендаційних систем, зокрема колаборативну фільтрацію та контентно-орієнтовану фільтрацію. Було проаналізовано принципи їхньої роботи, їхні переваги та недоліки.

Далі було проаналізовано існуючі рішення в області рекомендаційних систем, зокрема системи PickAMovieForMe та Netflix. Це дозволило сформулювати функціональні та нефункціональні вимоги до системи рекомендації фільмів.

Після цього було визначено інструменти та технології, які будуть використані для розробки системи. На основі аналізу різних варіантів було обрано мови програмування, фреймворки, бази даних та інші інструменти, що забезпечило ефективну реалізацію поставлених вимог.

Було обрано набір даних, який буде використовуватися для тренування та тестування системи. Проведено попередню обробку даних, включаючи очистку, нормалізацію та трансформацію даних. Це забезпечило задовільну якість вхідних даних та підвищило точність рекомендацій. Розроблено основні компоненти рекомендаційної системи, включаючи функції для пошуку фільмів, попередньої обробки тексту, розрахунку косинусної схожості та генерації рекомендацій. Далі було створено вебзастосунок для взаємодії користувачів з системою рекомендацій зі зручним та інтуїтивно зрозумілим інтерфейсом.

Підсумовуючи, у ході виконання дипломного проєкту було досягнуто поставлених цілей, а саме розроблено та реалізовано систему рекомендації фільмів, яка забезпечує задовільний рівень релевантних рекомендацій та зручність користування.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Про системи рекомендації фільмів. URL: <https://www.knowledgehut.com/blog/data-science/movie-recommendation-system> (дата звернення: 16.04.2024)
2. Стаття про рекомендаційні системи. URL: <https://labeledyourdata.com/articles/movie-recommendation-with-machine-learning> (дата звернення: 17.04.2024)
3. Стаття і рисунок про колаборативну фільтрацію. URL: https://medium.com/@ashmi_banerjee/understanding-collaborative-filtering-f1f496c673fd (дата звернення: 17.04.2024)
4. Стаття про рекомендаційні системи фільмів (здебільшого про колаборативну фільтрацію). URL: <https://www.freecodecamp.org/news/how-to-build-a-movie-recommendation-system-based-on-collaborative-filtering/> (дата звернення: 17.04.2024)
5. Стаття і рисунок про контентно-орієнтовану фільтрацію. URL: <https://medium.com/@beepabose/content-based-filtering-for-book-recommendation-using-pyspark-4369c4cbe006> (дата звернення: 17.04.2024)
6. Онлайн сервіс по рекомендації фільмів. URL: <https://pickamovieforme.com/> (дата звернення: 18.04.2024)
7. Нетфлікс. URL: <https://www.netflix.com/ua-en/> (дата звернення: 18.04.2024)
8. Клієнт-серверна архітектура. URL: <https://intellipaat.com/blog/what-is-client-server-architecture/> (дата звернення: 18.04.2024)
9. Про діаграму прецедентів. URL: <https://www.javatpoint.com/uml-use-case-diagram> (дата звернення: 18.04.2024)
10. Лекція про діаграму послідовності від ММСА ІПСА. URL: <http://mmsa.kpi.ua/sites/default/files/disciplines/%D0%A0%D0%BE%D0%B7%D1%80%D0%BE%D0%B1%D0%BA%D0%B0%20%D1%96%20%D1%82%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%20%D0%>

BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC/didkovska_m_v_testing_lecture_6.pdf (дата звернення: 18.04.2024)

11. Про побудову діаграм поведінки. URL: <https://dou.ua/forums/topic/40575/> (дата звернення: 18.04.2024)

12. Документація Python. URL: <https://docs.python.org/3/> (дата звернення: 19.04.2024)

13. Документація TypeScript. URL: <https://www.typescriptlang.org/> (дата звернення: 19.04.2024)

14. Документація веб фреймворка FastAPI мови програмування Python. URL: <https://fastapi.tiangolo.com/tutorial/> (дата звернення: 19.04.2024)

15. Документація бібліотеки Pandas мови програмування Python. URL: https://pandas.pydata.org/pandas-docs/stable/user_guide/index.html (дата звернення: 19.05.2024)

16. Документація бібліотеки React мови програмування JavaScript. URL: <https://uk.legacy.reactjs.org/tutorial/tutorial.html> (дата звернення: 19.04.2024)

17. Офіційна документація РСУБД PostgreSQL. URL: <https://www.postgresql.org/docs/> (дата звернення: 19.04.2024)

18. Документація VS Code. URL: <https://code.visualstudio.com/docs> (дата звернення: 19.04.2024)

19. IPython (Jupyter Notebook) документація. URL: <https://ipython.org/notebook.html> (дата звернення: 19.04.2024)

20. Використаний набір даних (MovieLens Dataset). URL: <https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset/data> (дата звернення: 05.02.2024)

21. Бібліотека FuzzyWuzzy мови програмування Python. URL: <https://pypi.org/project/fuzzywuzzy/> (дата звернення: 06.05.2024)

22. Стаття про бібліотеку FuzzyWuzzy. URL: <https://dou.ua/forums/topic/42948/> (дата звернення: 06.05.2024)

23. Набір інструментів по роботі з природніми мовами Natural Language Toolkit. URL: <https://www.nltk.org/> (дата звернення: 07.05.2024)

					ІА01.020БАК.006 ПЗ	Арк.
						59
Зм.	Лист	№ докум.	Підпис	Дата		

24. Benjamin Bengfort. Applied Text Analysis with Python. – O'Reilly Media, 2018. – 332p (дата звернення: 07.05.2024)

25. Aman Kedia. Hands-On Python Natural Language Processing. – Packt, 2020. – 304p (дата звернення: 08.05.2024)

26. Документації бібліотеки Scikit-learn мови програмування Python. URL: https://scikit-learn.org/stable/user_guide.html (дата звернення: 09.05.2024)

27. Сторінка документації Pandas про те, що індекси в даній бібліотеці починаються з нуля. URL: <https://pandas.pydata.org/pandas-docs/version/0.15.0/indexing.html#:~:text=pandas%20provides%20a%20suite%20of,These%20are%200%2Dbased%20indexing>. (дата звернення: 09.05.2024)

28. Безкоштовний інструмент для створення зображення бренду для вебзастосунку. URL: <https://logo.com/dashboard/your-logos> (дата звернення: 14.05.2024)

					ІА01.020БАК.006 ПЗ	Арк.
						60
Зм.	Лист	№ докум.	Підпис	Дата		

ДОДАТОК А

Система рекомендації фільмів

<https://github.com/SanAntonik/MRS>

