

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

галузь знань 0501 «Інформатика та обчислювальна техніка»

спеціальність 6.050101 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ

на тему:

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОТИ

здобувача Романенка Максима Сергійовича _____

Науковий керівник: д.е.н., проф.

_____ Ріппа С.П.

**Випускний бакалаврський проект
допущений до захисту в Екзаменаційній
комісії з атестації здобувачів вищої
освіти**

Завідувач кафедри: к.е.н., проф.

_____ Тішков Б.О.

Київ - 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань
спеціальність

12 Інформаційні технології
122 Комп'ютерні науки

ПОГОДЖЕНО:

Керівник проектної групи (гарант)
освітньо-професійної програми

ЗАТВЕРДЖУЮ:

Завідувач кафедри інформаційних
систем в економіці

_____ Іванченко Г.Ф.
«__» _____ 202_ р.

_____ Тішков Б.О.
«__» _____ 202_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти Романенка Максима Сергійовича
(прізвище, ім'я, по батькові)

денної форми навчання
очної (денної), заочної, дистанційної

на підготовку кваліфікаційного бакалаврського проекту
на тему: «ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОТИ»

Тему затверджено наказом ректора Університету від «__» _____ 2024 р. №529-ст

**Кваліфікаційний бакалаврський проект виконується на матеріалах
усіх доступних публікацій та Інтернет джерел**

План кваліфікаційного бакалаврського проекту

Розділ I. Характеристика та аналіз предметної галузі
(назва розділу)

Розділ II. Розробка вимог і моделювання інформаційної системи
(назва розділу)

Розділ III. Проектування та реалізація компонентів системи
(назва розділу)

Об'єкт дослідження: процес організації і управління інформаційною системою для створення і обміну рекомендаціями для кіноспільноти, яка надає можливості враховувати описання сюжетів фільмів та емоційне забарвлення, що розширює можливості для рекомендацій і допомагає користувачам знайти цікаві фільми поза звичними рамками.

Предмет дослідження: методи та технології організації веб-застосунків для створення і обміну рекомендаціями щодо перегляду фільмів з можливостями відображення емоційних аспектів рекомендацій.

Мета кваліфікаційного бакалаврського проєкту: створення веб-застосунку для кіноспільноти з метою створення та обміну рекомендаціями з врахуванням сюжетних характеристик переглянутих фільмів та емоційних аспектів цих процесів.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I

1. Дослідження предметної області. Збір інформації та вивчення матеріалів за тематикою кваліфікаційного бакалаврського проєкту.
2. Аналіз літератури та існуючих підходів до організації обміну рекомендаціями в соціальних мережах, підкастах блогосфери в кіномистецтві.

У розділі II

1. Аналіз і специфікація вимог до інформаційної системи створення та обміну рекомендаціями кіноспільнот.
2. Формулювання функціональних і нефункціональних вимог веб-застосунків обміну рекомендаціями, визначення рейтингів та моніторингу в кіномистецтві.
3. Постановка задач і побудова алгоритмів обміну рекомендаціями в кіноспільнотах.
4. Побудова інформаційної моделі, діаграм відповідності і послідовностей в процесах обміну рекомендаціями в кіноспільнотах з врахуванням емоційних аспектів.

У розділі III

1. Проектування та реалізація компонентів веб-застосунку для обміну кінорекомендаціями.
2. Загальна характеристика інформаційного забезпечення процесів створення та обміну кінорекомендаціями.
3. Визначення архітектури бази даних рекомендацій для кіноспільнот.
4. Програмне, організаційне і технічне забезпечення ІС обміну рекомендаціями в кіноспільнотах.
5. Побудова прототипу і програмна реалізація веб-застосунку обміну рекомендаціями в кіномистецтві.

**Завдання підготував
науковий керівник**

_____ Ріппа Сергій Петрович
(підпис) (ініціали, прізвище)

« 06 » лютого 2024 р.

**Завдання одержав
здобувач**

_____ Романенко Максим Сергійович
(підпис) (ініціали, прізвище)

« 07 » лютого 2024 р.

Відгук
на кваліфікаційний бакалаврський проєкт *Романенка*
Максима Сергійовича на тему
«ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОТИ»
за спеціальністю 122 «Комп'ютерні науки»

Актуальність теми. Виконаний кваліфікаційний бакалаврський проєкт має досить актуальну тематику, яка пов'язана з розробкою та впровадженням інформаційних систем для інформаційного простору кіноспільноти. Якість створення, підтримки та управління подібними системами в значній мірі залежить від того, наскільки глибоко і повно зроблений аналіз предметної області та наскільки гнучкою та адаптивною буде реалізована комп'ютерна система для рекомендацій переглянутих фільмів. Вирішити ці задачі можна шляхом використання сучасних інструментів надання, оцінки та обміну кінорекомендаціями і в цьому контексті тема, обрана здобувачем для підготовки бакалаврського проєкту, є актуальною та такою, що має практичну орієнтацію.

Позитивні риси бакалаврського проєкту. Завдання на кваліфікаційний бакалаврський проєкт, які сформульовані здобувачем, орієнтовані на аналіз і організацію процесів надання і обміну рекомендаціями кіноспільноти, в досліджуваній предметній області та врахована специфіка інформатизації дейтамайнінгу при створенні і використанні механізмів персональних рекомендацій. В роботі виконано проєктні роботи, розробку та реалізацію основних компонентів інформаційної системи у вигляді веб-додатку з відповідними програмними кодами.

Наявність самостійних розробок автора. Запропоноване здобувачем інформаційно-програмне середовище надання рекомендацій в кіноспільноті має можливості використання емоційних аспектів у відгуках кіноманів і їх послідовного розвитку з врахуванням бажань користувачів та пропозиціями профільних соціальних мереж кіно-середовища, що можна віднести до авторських напрацювань і новизни.

Цінність теоретичних висновків та практичних рекомендацій. Інформаційно-управлінська та технологічна система рекомендацій, яка спроектована і представлена в дипломній роботі, за допомогою яких майбутні користувачі можуть створювати і обмінюватись кінорекомендаціями, є безперечно цікавою і практично орієнтованою розробкою в сфері кінопростору.

Наявність недоліків. в представленому бакалаврському проєкті мають місце окремі стилістичні неточності, несуттєві описки та погрішності в оформленні.

Загальна оцінка кваліфікаційного бакалаврського проєкту. Підготовлений кваліфікаційний бакалаврський проєкт на тему «ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОТИ» відповідає чинним вимогам методичних рекомендацій щодо його структури, обсягу та змісту, містить необхідний ілюстративний матеріал, що доповнює та в достатній мірі повно розкриває суть виконаних проєктних рішень. У цілому цей проєкт можна рекомендувати до захисту з *позитивною оцінкою*, а його автор *Романенко Максим Сергійович* заслуговує на присвоєння йому освітньо-кваліфікаційного рівня «бакалавр» в галузі знань 12 «Інформаційні технології» за спеціальністю 122 «Комп'ютерні науки».

Науковий керівник:
д.е.н., професор кафедри ІСЕ

“10” червня 2024 р.



С.П. Ріппа

РЕЦЕНЗІЯ

на кваліфікаційний бакалаврський проект
Романенка Максима Сергійовича на тему
«ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОСТІ»
 за спеціальністю 122 «Комп'ютерні науки»,
 ОПП «Інформаційні управляючі системи і технології»

Актуальність теми: Представлений на рецензування кваліфікаційний бакалаврський проект за своєю тематикою та актуальністю відповідає сьогодишнім викликам і загрозам, перед якими опинилась Україна у сфері кіномистецтва, оскільки в тяжких умовах тривалої повномасштабної війни з росією, коли кіно-галузь фінансується, як й більшість інших гуманітарних сфер, за остаточним принципом, кінематограф все рівно залишається дієвою зброєю в боротьбі з агресором в кіно-мистецтві та на інформаційному фронті. В цьому контексті дипломний проект і актуальний, і своєчасний.

Характеристика роботи. Проведене здобувачем дослідження складається традиційним чином із трьох розділів і додатків. Перший розділ присвячений характеристиці предметної області та об'єкта дослідження, а також аналізу профільної літератури, де здобувач компактно та інформативно представляє актуальну ситуацію і динаміку розвитку сфери систем і технологій рекомендацій в кіномистецтві на прикладі найвідоміших і найвагоміших гігантів в сфері кіно-БД і -соціальних мереж – це IMDb та Letterboxd. В другому розділі поданий детальний аналіз і формулювання специфікацій, функціональних і нефункціональних вимог до інформаційної моделі системи кіно-рекомендацій, описано математичне і алгоритмічне забезпечення, в яких емоційну складову рекомендацій для кіноспільноти можна віднести до авторської новизни. Третій розділ зосереджений на проектних рішеннях щодо БД, інфологічної і даталогічної моделі, програмного і технічного забезпечення реалізації системи кіно-рекомендацій на базі інструментів Flask. Змістовно і структурно виконаний проект відповідає обраній темі та сформульованим завданням, її оформлення також вкладається в нормативи.

Виконаний проект має окремі несуттєві недоліки: зокрема, представлені в проєкті рішення щодо програмного і технічного забезпечення викладені переважно в скороченому вигляді, хотілось би побачити в ключових аспектах більш розгорнуте представлення програмних кодів, орієнтовані на сучасний контекст медіа-простору кулінарних послуг, є несуттєві помилки і описки в оформленні дипломної роботи.

Загальний висновок. У цілому представлений на рецензування кваліфікаційний бакалаврський проект на здобуття ступеня бакалавра свідчить про достатню кваліфікацію автора щодо обраної тематики, проект виконаний на належному рівні, за змістом та практичною цінністю відповідає вимогам МОН і може бути оцінений позитивно, а її автор, Романенко Максим Сергійович заслуговує на присвоєння йому освітньо-кваліфікаційного рівня «бакалавр» за спеціальністю «Комп'ютерні науки».

Рецензент: _____
 (підпис)

Погореловська І.Д., к.е.н., доцент,
 доцент кафедри комп'ютерних та інформаційних
 технологій і систем факультету фінансів та
 цифрових технологій Державного
 податкового університету

«10» червня 2024 р.

АНОТАЦІЯ

випускного бакалаврського проекту студента 4 курсу
Навчально-наукового інституту «Інститут інформаційних технологій в економіці»
Романенка Максима Сергійовича, виконаної на тему:
«Інформаційна система для кіноспільноти»
Київ: кафедра інформаційних систем в економіці, 2024 р.

Випускний бакалаврський проект присвячений актуальній проблемі рекомендації фільмів, яку пропонується розв'язувати та удосконалювати з використанням сучасних інформаційних технологій. Випускний бакалаврський проект складається з трьох розділів, логічно пов'язаних між собою.

В першому розділі дана характеристика предметної галузі й об'єкта дослідження, наведено аналіз задач, що розв'язуються, а також наводиться перелік існуючих програмних засобів даної предметної галузі.

Другий розділ є проектним і присвячений обґрунтуванню методу проектування системи, розробленню її архітектури, виконанню постановки та розробленню алгоритму розв'язання задач.

Третій розділ конструктивний. Тут наведено інформаційне, технічне програмне та організаційне забезпечення для підсистеми рекомендації фільмів, авторизації користувачів та створення відгуків на фільми: розроблена структура інформаційного забезпечення, описані та наведені в додатках форми вхідних та вихідних документів. Вказані структури інформаційних масивів, які використовуються під час вирішення задачі.

Висновки містять рекомендації щодо доцільності розробки та впровадження системи, потенційні покращення в майбутньому, практичне застосування системи та підкреслюють унікальність цієї розробки.

РЕФЕРАТ

Дипломний проєкт містить 60 сторінок, 4 таблиці, 17 рисунки, перелік джерел посилань з 16 найменувань і 3 додатки.

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОТИ

Перелік ключових слів: *КІНОСПІЛЬНОТА, ОСОБИСТИЙ ОБЛІКОВИЙ ЗАПИС, ВІДГУКИ НА ФІЛЬМИ, ПЕРСОНАЛІЗОВАНІ РЕКОМЕНДАЦІЇ, АНАЛІЗ СЮЖЕТУ ФІЛЬМІВ, АНАЛІЗ ЕМОЦІЙ ФІЛЬМУ, ТЕХНОЛОГІЇ ВЕБ-ЗАСТОСУНКУ, ІНТЕРАКТИВНИЙ ІНТЕРФЕЙС*

Предметом дослідження є сукупність теоретичних та практичних підходів до рекомендації фільмів.

Об'єктом дослідження виступає процес проектування веб-застосунку.

Мета випускного бакалаврського проєкту полягає в розробці та впровадженні ефективного веб-застосунку, який використовує новий спосіб рекомендації фільмів.

Завданнями випускного бакалаврського проєкту є аналіз потреб аудиторії, проектування та реалізація програмного забезпечення, тестування та впровадження системи

Результати досягнуті в процесі роботи включають розроблений та впроваджений веб-застосунок для кіноспільноти. Новизна проєкту полягає в реалізації алгоритму рекомендацій на основі аналізу сюжетів та емоцій фільмів.

Одержані результати можуть бути використані для подальшого розвитку веб-застосунку та покращення користувацького досвіду

Рік виконання кваліфікаційного бакалаврського проєкту – 2024

Рік захисту кваліфікаційного бакалаврського проєкту - 2024

ANNOTATION

final bachelor's project of a 4th-year student
Educational and Scientific Institute "Institute of Information Technologies in the
Economy"

Romanenko Maksym Serhiiiovych, performed on the theme:
"Information system for the film community"

Kyiv: Department of Information Systems in Economics, 2024.

The final bachelor's project is devoted to the actual problem of film recommendation, which is proposed to be solved and improved using modern information technologies. The final bachelor's project consists of three logically connected sections.

In the first section, the characteristics of the subject area and the object of research are given, the analysis of the problems being solved is given, and the list of existing software tools of this subject area is given.

The second section is project-based and is devoted to the justification of the system design method, development of its architecture, implementation of the staging and development of the problem-solving algorithm.

The third section is constructive. Information, technical software, and organizational support for the movie recommendation subsystem, user authorization, and creation of movie reviews are provided here: the information support structure is developed, and the forms of input and output documents are described and given in the appendices. The structures of the information arrays used when solving the problem are indicated.

The conclusions include recommendations on the feasibility of system development and implementation, potential future improvements, and practical applications, and highlight the uniqueness of this development.

ЗМІСТ

АНОТАЦІЯ.....	4
ВСТУП	10
РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ.....	13
1.1. Характеристика предметної галузі та об'єкта дослідження	13
1.2. Аналіз літератури.....	19
РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ.....	25
2.1. Аналіз і специфікація вимог до інформаційної системи.....	25
2.2. Постановка та алгоритм розв'язування задач	28
2.3. Алгоритм розв'язання задачі.....	30
2.4. Моделювання інформаційної системи	39
РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ.....	48
3.1. Інформаційне забезпечення	48
3.2. Програмне забезпечення	50
3.3. Результат реалізації інформаційної системи	52
Висновок	55
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	56
ДОДАТКИ.....	57

ВСТУП

Сучасний розвиток інформаційних технологій неодмінно впливає на різноманітні аспекти нашого життя, включаючи сферу кінематографії. Спостерігається збільшення кількості інтернет-ресурсів, присвячених обговоренню фільмів та обміну враженнями про них. Однак, із зростанням потоку інформації, індивідуальний вибір фільмів, що відповідають особистим смакам, стає складнішим завданням для багатьох користувачів.

Переважає більшість сучасних систем рекомендації фільмів базується на соціальному аспекті [1]. Алгоритми аналізують популярні фільми та серіали за таким самим жанром та акторами, і хоч в багатьох випадках це призведе до корисних результатів, часто ці алгоритми не можуть надати список дійсно схожих фільмів. Прикладом цього феномену є фільм «Джокер», що вийшов у 2019 році. Сюжет фільму розповідає про життя Артура Флека, нещасного коміка, який живе у Готемі – місті, поглибленому в соціальні та економічні проблеми. З кожним днем він все більше занурюється у світ ментальної хаосу і стає символом анархії. Цей фільм досліджує теми соціальної несправедливості, психічного розладу та становлення Джокера, як персонажа цього всесвіту. Якщо шукати схожі фільми, використовуючи існуючі алгоритми рекомендації, список переважно заповнений фільмами у спільному всесвіті DC Comics з участю персонажу Джокер, що для тих глядачів, яких зацікавив персонаж це можуть бути хороші рекомендації, але для великої кількості глядачів, яким сподобався психологічний і соціальний аспект фільму, дані рекомендації не мають жодного сенсу.

Серед існуючих інформаційних систем, існують декілька, частиною яких є технології «рекомендацій фільмів». Найвідомішими та найвпливовішими з них є системи *IMDb* і *Letterboxd*, які за своєю сутністю є досить різними. Перша, *IMDb*, це перш за все, база величезна БД, а друга – соціальна мережа. Характеристика і хронологія цих веб-ресурсів є досить показовими для світової кіноспільноти та їх характеристика розглядатиметься в підрозділі 1.1.

IMDb використовує різноманітні методи для рекомендацій фільмів. Вони аналізують: Історію переглядів та оцінок користувачів, щоб зрозуміти їх вподобання та рекомендувати фільми з подібними характеристиками, популярність та рейтинги фільмів, беручи до уваги загальну думку користувачів про них, взаємозв'язки між фільмами на основі їхніх жанрів, режисерів, акторів та інших характеристик, щоб рекомендувати подібні фільми, соціальні взаємодії користувачів, такі як коментарі, рецензії та рекомендації, щоб зрозуміти їхні інтереси та рекомендувати фільми на основі спільних вподобань.

Ці методи дозволяють *IMDb* створювати персоналізовані рекомендації для кожного користувача, що покращує їхній досвід використання платформи та допомагає їм відкривати нові цікаві фільми.

Letterboxd аналізує вподобання користувачів на основі їх історії переглядів та оцінок, враховує спільні вподобання з аналогічними користувачами і використовує аналіз рецензій та списків, щоб створити персоналізовані рекомендації фільмів [3].

Завданням цього бакалаврського проекту є розробка сучасного веб-застосунку, спрямованого на покращення користувацького досвіду під час отримання рекомендацій фільмів. Для втілення даного проекту використано сучасні технології та інструменти розробки. Мова програмування *Python* вибрана як основний інструмент для створення функціоналу системи. Система управління даними *SQLite* обрана для зберігання та оптимізованого доступу до інформації про фільми та користувачів.

Мета цього проекту полягає в створенні ефективного способу рекомендації фільмів. Для даної задачі було обрано створення веб-застосунку, оскільки це досить доступний формат для більшості потенційних користувачів системи.

Цей проект в своїй основі має на меті побудувати ІС, яка буде надавати широкі можливості формулювати і обмінюватись рекомендаціями щодо фільмів між користувачами. Саме з цією метою для аналізу обирається опис сюжету фільму [4]. Фільми одного жанру можуть суттєво відрізнятися за своїм змістом та стилем. Аналіз сюжету дозволяє виявити фільми з подібними сюжетними

елементами навіть у різних жанрах, що розширює можливості для рекомендацій та допомагає користувачам знайти цікаві фільми поза звичними рамками.

Цей проект намагається мінімізувати людський фактор при отриманні рекомендацій фільмів. Саме для цього, на додаток до аналізу сюжету фільму, використовується його емоційний аналіз [5]. Рекомендації, що базуються на емоційному аналізі, менше піддаються впливу популярних трендів та соціальних упереджень. Це дозволяє відкривати користувачам менш відомі, але потенційно цікаві фільми, що відповідають їхнім емоційним уподобанням. Користувачі можуть знайти фільми, які не просто популярні, а які резонують з бажаними емоціями та настроями. Це підвищує загальне задоволення від користування веб-застосунком та стимулює довготривалу взаємодію з платформою.

Емоційний аналіз дозволяє враховувати не лише жанрові характеристики, але й тональність та атмосферу фільму, що часто є вирішальними факторами у виборі кіно.

Цей проект має потенціал дозволити в майбутньому розкрити альтернативний вектор в сфері рекомендації фільмів.

РОЗДІЛ 1. ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1. Характеристика предметної галузі та об'єкта дослідження

В рамках контексту даної роботи можна виділити наступні об'єкти предметної галузі:

1. Фільми (опис сюжету, жанр, режисер, актори, емоційний профіль);
2. Відгуки (назва фільму, оцінка, текстовий відгук, користувач);
3. Алгоритм рекомендації (порівняння фільмів за їх параметрами, оцінка емоційного профілю).

У предметній галузі розробки веб-застосунку для кіно-спільноти, приймаються рішення, спрямовані на досягнення вдосконалення алгоритму рекомендації та підвищення залучення користувачів, а саме на імплементацію методів аналізу природних мов [13,14] та аналізу настроїв [12].

Особи, що приймають рішення в цій галузі, включають розробників програмного забезпечення, дизайнерів інтерфейсу, аналітиків даних та менеджерів продукту.

Фактори, що впливають на середовище прийняття рішень, включають технічні можливості платформи, ринкові умови та поведінку користувачів.

Вищезгадані у Вступі веб-ресурси *IMDb* та *Letterboxd* потребують більшої уваги, оскільки вони, згідно з думкою багатьох експертів, є не тільки найвідомішими та найвпливовішими веб-ресурсами у світі кіномистецтва, але й вони певним чином висвітлюють основні тенденції в розвитку світового кіноринку і презентують різнобарвну картину усієї кіноспільноти. Розглянемо їх більш прискіпливо.

IMDb є одночасно найбільшою базою даних та веб-сайтом про кінематограф. База даних переважно заповнюється добровольцями і це чимось нагадує концепцію вікі. За актуальними даними у базі зібрана інформація про понад 10 млн фільмів і телесеріалів, також є інформація майже про 11,5 млн. акторів, режисерів та інших

професіоналів кіно зі всього світу. Починаючи від 1998 року базою даних *IMDb* володіє компанія «*Amazon.com*». Більшість інформації *IMDb* перебуває в безплатному доступі, сайт *IMDb.com* функціонує на основі безплатного програмного забезпечення (*Apache*, *Perl* та ін.). Існує можливість й платного доступу, який дозволяє отримати інформацію про контакти акторів, продюсерів, режисерів тощо.

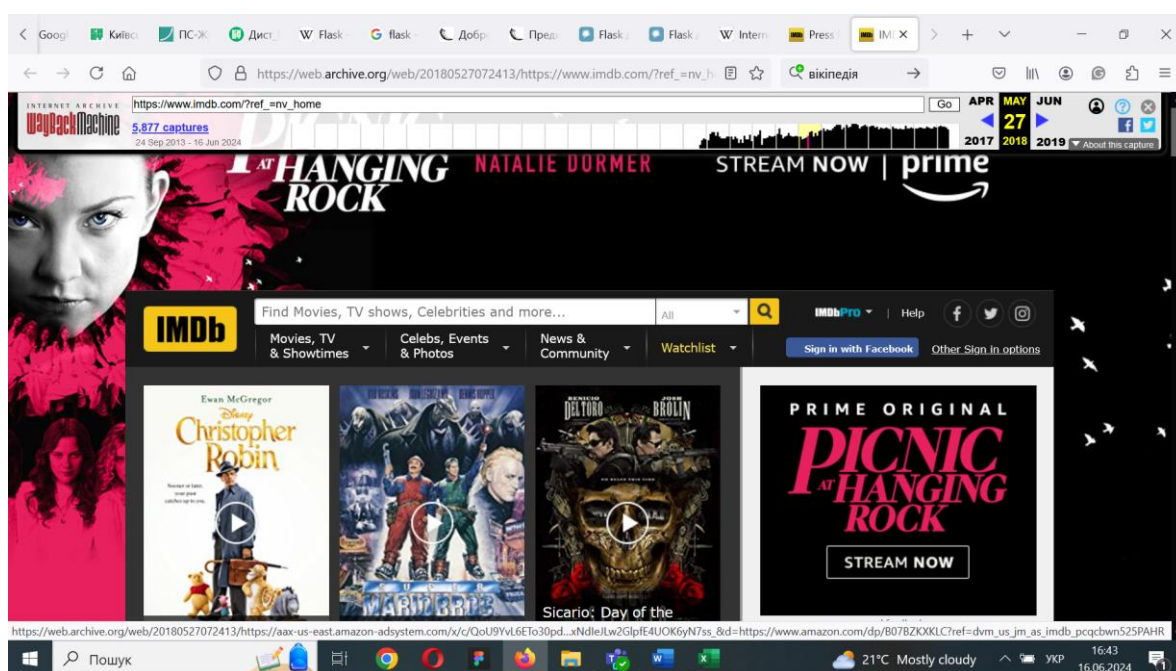


Рисунок. 1.4 – Титульна сторінка веб-сайту *IMDb.com*

За даними рейтингового агентства «*Alexa*», сайт *imdb.com* постійно входить до сотні найпопулярніших сайтів Інтернету. Основна мова інтерфейсу на сайті — англійська, українського інтерфейсу сайт не має, проте назви фільмів і мінімальна довідкова інформація на різних суміжних сервісах поступово перекладаються.

База *IMDb* офіційно виникла 17 жовтня 1990 року, коли кілька любителів кіно із групи новин *Usenet rec.arts.movies* вирішили проіндексувати свій *FAQ*, що містив до того часу понад 23 тисячі статей про приблизно 10 тисяч фільмів та телесеріалів. Для цієї індексації один із засновників *IMDb* (та теперішній її виконавчий директор) Кол Нідем (англ. *Col Needham*) написав потрібні *UNIX-скрипти* для організації бази даних кінофільмів. Особливістю цього проекту є те,

що *IMDb* завжди (і зараз) суворо дотримується вимог в програмному забезпеченні як до систем з відкритим вихідним кодом. У той час база ще мала назву «*rec.arts.movies movie database*», назва «*Internet Movie Database*» виникла пізніше.

З часом поступово база росла й технічно удосконалювалась. У 1993 році база дебютувала у Всесвітній павутині на сервері Кардіффського університету в Уельсі. Вже тоді база даних була дуже популярною в Інтернеті, до неї надходило в середньому до 65 мільйонів запитів на місяць. У січні 1996 року була утворена компанія «*Internet Movie Database, Ltd*», акціонерами якої й стали розробники бази. Попри величезну популярність, база продовжувала залишатися як хобі – більшість розробників бази все ще мали інше постійне місце роботи.

Незабаром база вичерпала ресурси університетського серверу і переїхала на виділені сервери в США та Велику Британію. Через рік виділених серверів було вже 9, але й цієї потужності виявилося замало. Підтримувати базу було дедалі складніше, для повноцінного розвитку не вистачало ресурсів. Декілька компаній пропонували купити *IMDb*, але ентузіасти бази не довіряли їм – ці компанії вбачали в *IMDb* лише маркетингову платформу, а не базу знань на благо кіноманів.

Тоді в 1998 році на горизонті *IMDb* з'явився *Джеф Безос* (англ. *Jeff Bezos*) з компанії «*Amazon.com*». Ця людина дійсно розуміла інтернет-спільноту та альтруїстичні принципи *IMDb*. Він запропонував купити частину акцій *IMDb* за умови повного збереження загального безплатного доступу до ресурсів *IMDb*. В обмін на інвестиції він хотів для «*Amazon.com*» лише право продавати кінопродукцію через сайт *IMDb.com*. Це була чудова пропозиція, від якої було неможливо відмовитися і з того часу співробітництво *IMDb* та «*Amazon.com*» продовжується. Були значно розширені серверні потужності, акціонерам «*IMDb Ltd*» виплачуються дивіденди, багато ентузіастів *IMDb* було прийнято в штат компанії, найнято й новий персонал. Компанія «*Amazon.com*» також не залишилася в накладі. Співробітники обох компаній називають цей союз благословенним. Нині це один із небагатьох прикладів вдалої взаємодії великого бізнесу й так званої спільноти ентузіастів відкритого ПЗ (англ. *opensource community*).

Характерно, що окрім основної англійської версії сайт у минулому мав

локалізації ще п'ятьма мовами: німецькою, італійською, іспанською, французькою та португальською. Але з 18 лютого 2013 року підтримку іншомовних версій *IMDb* було припинено, а їх відповідні сайти *IMDb.de*, *IMDb.fr*, *IMDb.it*, *IMDb.pt* та *IMDb.es* стали автоматично перенаправляти на *IMDb.com*.

Досить цікавим явищем на *IMDb* є онлайн-голосування, де кожен відвідувач сайту може проголосувати за свої улюблені фільми, і навпаки, проти не улюблених. Таким чином на *IMDb* виявлені 250 найкращих фільмів всіх часів, а також 100 найгірших фільмів. Є окремі рейтинги фільмів за жанрами, за десятиліттям та ін. Жанри в *IMDb* класифіковані наступним чином (табл.1.3):

Таблиця 1.3 – Жанри голосування кіноманів на *IMDb*

Action	Documentary	Horror	Sci-Fi
Adventure	Drama	Independent	Thriller
Animation	Family	Musical	War
Comedy	Fantasy	Mystery	Western
Crime	Film-Noi	Romance	

Джерело: [14]

Досить показовою є статистика [15] наповнення інформацією БД *IMDb* з сайту Press Room (рис.1.5) на початок 2020 року (див. табл.1.5):

Таблиця 1.5 – Статистика *IMDb*

7 548 383	Статей про фільми і телесеріали, з них:
5 338 520	телеепізодів;
560 232	кінофільмів (що вийшли в прокат та були показані в кінотеатрах);
196 972	телесеріалів;
34 590	мінісеріалів;
287 977	відеофільмів (що поширювалися тільки на відео, без демонстрації в кінотеатрах);
111 667	телефільмів (показані тільки по телебаченню, без демонстрації в кінотеатрах);
781 188	короткометражних фільмів;
26 915	відеоігор.
11 214 704	Статей про персоналії, з яких:
29 412 693	про акторів;

18 949 421	про актрис;
12 201 834	про продюсерів;
8 681 358	про сценаристів;
5 419 175	про режисерів;
2 568 752	про операторів;
2 111 393	про композиторів;
3 348 611	про монтажерів.

Джерело: [15]

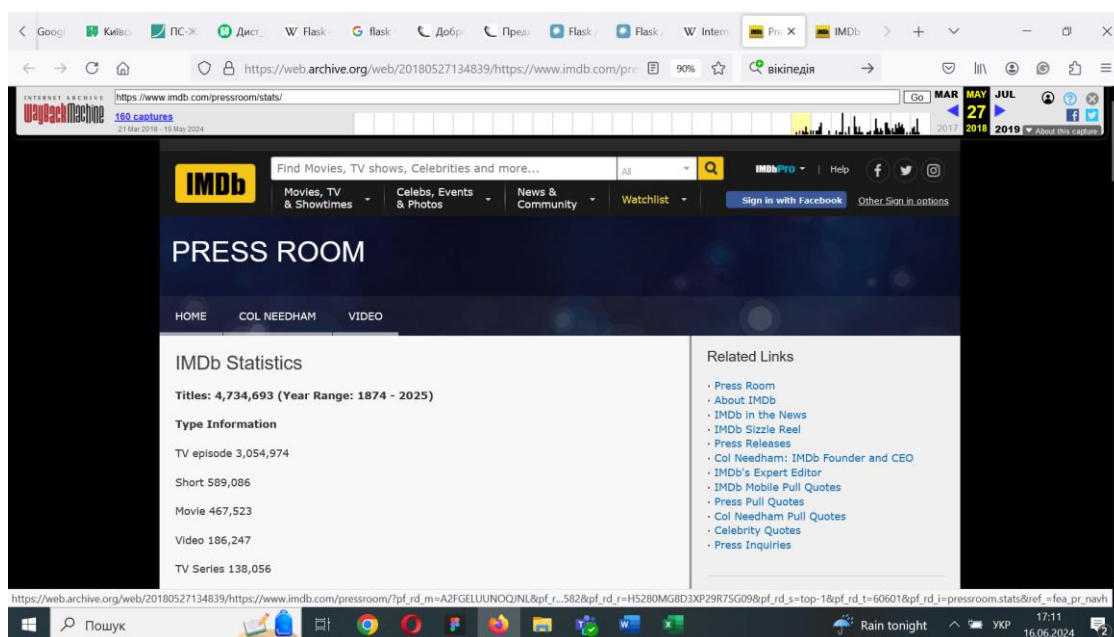


Рисунок. 1.4 – Титульна сторінка веб-сайту *Press Room IMDb.com*

Letterboxd [16] – це соціальна мережа, заснована Метью Б'юкененом і Карлом фон Рандоу у 2011 році в Окленді, Нова Зеландія. Головна мета сайту – поширювати любов до кіно. *Letterboxd* дозволяє людям ділитися своєю пристрастю до кіно, пишучи рецензії, взаємодіючи з іншими користувачами, створюючи списки фільмів та журнали переглядів. 15 травня 2017 року 100 мільйонів фільмів були позначені як переглянуті, а 20 серпня 2022 року понад один мільярд.

Хоча *Letterboxd* нечасто з'являється в заголовках новин як спільнота чи сервіс, за ці роки він набув стабільної популярності. У статті в *The Ringer* кінокритик Скотт Тобіас назвав *Letterboxd* «найбезпечнішим простором для обговорення фільмів, який ми маємо» завдяки його спільноті та дискусійній моделі в розпал пандемії *COVID-19*.

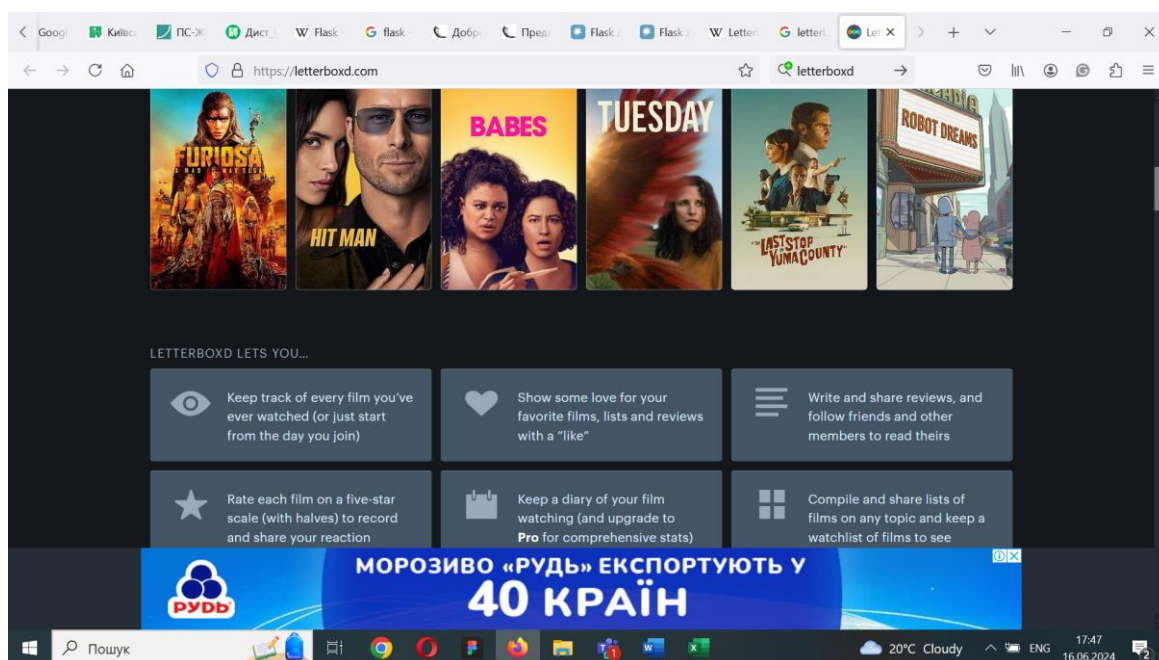


Рисунок. 1.4 – Титульна сторінка веб-сайту *Letterboxd.com*

Сайт був запущений Бруклінською бета-версією у жовтні 2011 року і протягом наступних шести місяців набрав понад 17 000 бета-тестерів. 24 квітня 2012 року він перестав бути приватним сайтом і перейшов у режим публічної бета-версії. Всі сторінки стали загальнодоступними, але спочатку там було лише 23 відео, а членство обмежувалося лише запрошеними. 8 лютого 2013 року сайт було відкрито для громадськості. *Letterboxd* запровадив багаторівневу структуру членства з диференційованими можливостями для платної та безкоштовної версій. Платна версія, яка називається *Pro*, надає доступ до додаткових функцій, включаючи персоналізовані сторінки з підсумками року.

У вересні 2020 року *Letterboxd* оголосив про новий рівень членства для кіноорганізацій та підприємств, таких як фестивалі, інтимні кінотеатри та навіть подкасти. Ця нова функція, що отримала назву *Letterboxd HQ*, спричинила сплеск популярності серед нової аудиторії, а такі організації, як *IndieWire*, *TIFF* та *The Criterion Collection*, почали взаємодіяти зі спільнотою.

У статті *New York Times*, опублікованій у січні 2021 року, зазначалося, що з початку пандемії кількість користувачів *Letterboxd* «подвоїлася», налічуючи понад 3 мільйони учасників, порівняно з 1,7 мільйонами попереднього року. До 20 липня 2022 року користувачі позначили понад 1 мільярд фільмів як переглянуті. У 2022

році користувачі залишили 227 мільйонів рецензій.

Будь-хто може читати вміст сайту. Для участі потрібен безкоштовний обліковий запис користувача. Усі користувачі можуть оцінювати, переглядати та позначати фільми відповідними ключовими словами. Вони можуть вести списки фільмів, які вони бачили або хочуть побачити, і взаємодіяти з іншими користувачами. Списки можуть бути загальнодоступними або приватними. Рейтинги ґрунтуються на п'ятизірковій системі, з можливістю виставлення півзірок. Модель *фоловерів* дозволяє користувачам стежити за діяльністю інших користувачів на сайті й отримувати повідомлення про неї. Ці функції ведуть до іншого інтерактивного рівня, окрім перегляду фільмів.

Преміум-учасники можуть, серед іншого, отримувати сповіщення, коли фільм з'являється в їхньому списку у провайдера відео на вимогу. *Letterboxd* також доступний у вигляді мобільного додатку для *Android* та *iOS*.

Усі метадані, пов'язані з фільмами, що використовуються на сайті, походять з бази даних фільмів (*TMDB*), бази даних з відкритим вихідним кодом. У вересні 2019 року сайт уклав партнерство з *JustWatch*, щоб дозволити користувачам дивитися фільми у різних провайдерів, залежно від наявності. У березні 2022 року сайт уклав партнерство з *Nanocrowd*, щоб показувати користувачам «наножанри» та рекомендації щодо схожих фільмів.

1.2. Аналіз літератури

"*Recommender Systems: The Textbook*" - це важливий ресурс для тих, хто цікавиться рекомендаційними системами та бажає зрозуміти їх теоретичні основи та практичні аспекти. Автори *Паоло Массе* та *Альєсандро Росетті* представляють глибокий аналіз різних аспектів рекомендаційних систем, від базових концепцій до передових технологій. Автори розглядають різні підходи до побудови рекомендаційних систем, включаючи колаборативне фільтрування, факторизацію матриці, контентне фільтрування та гібридні методи. Книга детально розглядає

метрики та методи оцінки ефективності рекомендаційних систем, що допомагає розробникам та дослідникам оцінювати та порівнювати різні алгоритми. Автори описують різні алгоритми та моделі, що використовуються в рекомендаційних системах, включаючи розрахунок схожості, ранжування та прогнозування. Книга розглядає приклади застосування рекомендаційних систем у різних галузях, таких як електронна комерція, соціальні медіа та інтернет-платформи. В системі цієї роботи використаний саме контентний підхід до побудови системи рекомендації. Контентний підхід використовує аналіз властивостей об'єктів (у цьому випадку - фільмів), щоб зробити рекомендації. В даному випадку система аналізує схожість сюжетів фільмів за допомогою методу косинусної схожості, а також проводить аналіз сюжетів на емоції. Таким чином, система фокусується на аналізі самого контенту (сюжетів та емоційного тону), а не на врахуванні дій користувачів або їхніх відгуків про фільми. Цей підхід забезпечує більш об'єктивні та незалежні рекомендації, які не базуються на попередніх діях і відгуках інших користувачів.

Ще одна публікація за тематикою дипломної роботи – це книга «Recommender Systems: The Textbook», автор – Charu C. Aggarwal, (видавництво Springer, 2016. 519 p) – є широко відомою публікацією за тематикою систем рекомендацій і цю роботу не можна залишити поза увагою.

Категорії книги Charu C. Aggarwal – це AI & Machine Learning, Data Mining, Databases & Big Data, Computer Science. Представлені в ній матеріали всебічно висвітлюють тематику систем рекомендацій, які надають персоналізовані пропозиції щодо продуктів або послуг користувачам на основі їхніх попередніх пошуків або покупок. Більшість методів систем *Recommender* були адаптовані для різноманітних застосувань, включаючи аналіз журналів запитів, соціальні мережі, рекомендації новин і обчислювальну рекламу. Дана робота синтезує як фундаментальні, так і передові теми наукової галузі, яка зараз досягла зрілості.

Основні розділи цієї книги впорядковано в три категорії. Перша - *алгоритми та оцінка*: у цих розділах розглядаються фундаментальні алгоритми в системах рекомендацій, включаючи методи спільної фільтрації, методи на основі контенту, методи на основі знань, методи на основі ансамблю та оцінювання. Друга -

рекомендації в певних областях і контекстах: контекст рекомендації можна розглядати як важливу побічну інформацію, яка впливає на цілі рекомендації. Досліджуються різні типи контексту, такі як часові дані, просторові дані, соціальні дані, дані тегів і надійність.

Третя категорія – *розширені теми та застосування*: обговорюються різні аспекти стійкості систем рекомендацій, наприклад системи шилінгу, моделі атак та їхній захист. Крім того, новітні теми, такі як навчання ранжування, багаторуки бандити, групові системи, багатокритеріальні системи та системи активного навчання, представлені разом із програмами. Незважаючи на те, що вказана книга в першу чергу служить підручником, вона також може бути корисною промисловим практикам і дослідникам завдяки своїй зосередженості на застосуваннях і довідках. Також в роботі були надані численні приклади та вправи, а для інструкторів є доступним посібник з рішеннями.

Книга "*Flask Web Development: Developing Web Applications with Python, 1st Edition*" є важливим джерелом для тих, хто цікавиться розробкою веб-додатків з використанням мови програмування *Python* та фреймворку *Flask*. Автор пояснює основні концепції фреймворку *Flask* та його використання для розробки веб-додатків. Він розглядає маршрутизацію, створення шаблонів, роботу з формами та інші аспекти розробки з *Flask*. Книга пропонує крок за кроком інструкції та приклади коду для розробки повноцінного веб-додатку з використанням *Flask*. Це включає створення бази даних, автентифікацію користувачів, роботу з *AJAX* запитами та інші функціональні можливості. Книга також описує, як розширювати функціональність веб-додатків з використанням різних розширень та бібліотек, які доступні для *Flask*.

Ця книга має, в певному розумінні, особливий статус, оскільки *Flask-концепція* уособлює на сьогодні одне із найвпливовіших напрямів сучасного проектного менеджменту в світі веб-додатків. *Flask* є багатогранним, але розглянемо його основні проектно-орієнтовані інструменти, які особливо є важливими в контексті тематики даної дипломної роботи.

Flask як «мікрофреймворк». "*Мікро*" не означає, що веб-додаток, що

розробляється, цілком повинен поміщатися в один файл з кодом на *Python*, хоча, звичайно ж, це може бути і так. Також, це не означає, що *Flask* відчуває нестачу функціональності. "Мікро" в слові "мікрофреймворк" означає, що *Flask* прагне дотримуватися простого, але розширюваного ядра. *Flask* не буде вирішувати за розробника багато речей, наприклад, яку базу даних використовувати. А ті рішення, які він може прийняти, наприклад, який із двигунів для роботи з шаблонами використовувати, легко змінити. Решта залежить від розробника, таким чином, може виявитися, що *Flask* - все, що розробнику потрібно, і нічого зайвого.

За замовчуванням, *Flask* не включає рівень абстракції баз даних, валідації форм або якихось інших, для чого вже існують різні бібліотеки, що займаються цим. Замість цього, *Flask* підтримує розширення для додавання подібної функціональності у цільову програму, таким чином, якби це було реалізовано в самому *Flask*. Численні розширення забезпечують інтеграцію з базами даних, валідацію форм, обробку завантажень на сервер, різні відкриті технології аутентифікації тощо. *Flask* може бути "мікро", але при цьому він готовий для використання в реальних завданнях для найрізноманітніших потреб.

Конфігурація та Угоди. *Flask* має багато параметрів конфігурації з розумними значеннями за замовчуванням і мало попередніх угод. За згодою, шаблони та статичні файли зберігаються в піддиректоріях всередині дерева вихідних текстів на *Python*, з назвами *templates* і *static* відповідно. Хоча це можна і змінити, зазвичай цього робити не варто, особливо на початку роботи.

Зростає разом з Flask. В процесі завантаження і використання *Flask*, розробники можуть знайти в співтоваристві безліч доступних його розширень, для їх інтеграції у потрібний проект, що реально працює. Команда розробки ядра *Flask* розглядає розширення та гарантує, що схвалені розширення не перестануть працювати з майбутніми релізами.

Поки кодова база розробників зростає, можна приймати доречні для цільового проекту рішення. *Flask* забезпечує найпростіший шар для склеювання, виконаний настільки якісно, наскільки це дозволяє зробити *Python*. Можна реалізувати розширені шаблони в *SQLAlchemy* або в іншому засобі для роботи з БД,

у необхідних випадках застосувати надійні нереляційні бази даних і використовувати переваги не призначених для роботи з фреймворками засобів, побудованих для роботи з WSGI - веб інтерфейсом для мови *Python*.

Flask для налаштування поведінки містить безліч гачків (хуків). Якщо розробникам потрібні додаткові параметри, клас *Flask* побудований, щоб бути готовим до створення підкласів.

Внутрішньо-потоківі об'єкти (Thread-Locals) у Flask. Одним із проектних рішень для *Flask* було те, що прості завдання мають бути простими; вони не повинні вимагати великого обсягу коду і не повинні обмежувати розробників. Тому при створенні *Flask* було обрано кілька рішень, які, на перший погляд, можуть здатися несподіваними чи оригінальними. Наприклад, у собі *Flask* використовує внутрішньо-потоківі об'єкти, тому розробникам у запиті, щоб зберігати потоко-безпечність (*thread-safe*), необов'язково передавати об'єкти від функції до функції. Такий підхід зручний, але вимагає дійсний контекст запиту для впровадження залежностей або спроби повторного використання коду, який застосовує значення, прив'язане до запиту. *Flask* до внутрішньо-потоківих змінних відноситься чесно, не приховує їх і завжди явно вказує в коді та документації, де б вони не використовувалися.

Розробка для Інтернет з обережністю. Під час створення веб-застосунків завжди потрібно рахуватись з безпекою. Якщо розробник пише веб-додаток, і, ймовірно, дозволяє користувачам реєструватися та залишати їх дані на сервері. Користувачі довіряють розробнику свої дані. І навіть якщо розробник єдиний користувач, який може ввести дані в додаток, все одно необхідно надійно їх зберегти. На жаль, існує безліч способів, щоб скомпрометувати захист з точки зору безпеки веб-програми. *Flask* захищає розробників від однієї з найпоширеніших проблем безпеки сучасних веб-додатків: від між-сайтового скриптингу (*cross-site scripting* - *XSS*). Якщо тільки розробник не бачить небезпечний *HTML-код* як безпечний, його прикриє *Flask* і шаблонізатор *Jinja2*, що розташований на нижніх рівнях. Також є й безліч інших способів зустріти проблеми з безпекою.

Документація попереджає розробників про аспекти веб-розробки, які

вимагають уваги до безпеки. Деякі з цих міркувань набагато складніші, ніж можна було б подумати, і іноді усі недооцінюють ймовірність того, що вразливість буде використана – а в цей час розумний зловмисник обчислює способи для злому наших додатків. Досить часто розробники сподіваються, що їх додаток не настільки важливий, щоб зацікавити зловмисника. Але при деяких видах атак є ймовірність, що автоматизовані боти зондують різні способи заповнення вашої бази даних спамом, посиленнями на шкідливі програми і т.п.

У тому, що розробник, в процесі розробки відповідно до постановки завдань, повинен діяти з обережністю, і при цьому стежити за спробами злому, *Flask* нічим не відрізняється від інших фреймворків.

Статус Python 3. В сучасних умовах спільнота *Python* знаходиться у процесі вдосконалення бібліотек для підтримки нової ітерації мови програмування *Python*. Хоча ситуація і різко покращується, є ще деякі питання, через які користувачам важко перейти на *Python 3* прямо зараз. Ці проблеми частково викликані змінами в мові, що не пройшли тривалого випробування, частково тим, що не усі розробники працювали з низько-рівневими змінами рівня *API*, внесеними *Python 3* в роботу з *Unicode*.

Виходячи із досвіду рекомендується використовувати *Python 2.7* з активованими попередженнями про несумісності з *Python 3*. Якщо в найближчому майбутньому розробники планують оновитися до *Python 3*, можна наполегливо рекомендувати з особливою увагою відноситись до написання коду, сумісного з наступними версіями *Python*.

РОЗДІЛ 2. РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Аналіз і специфікація вимог до інформаційної системи

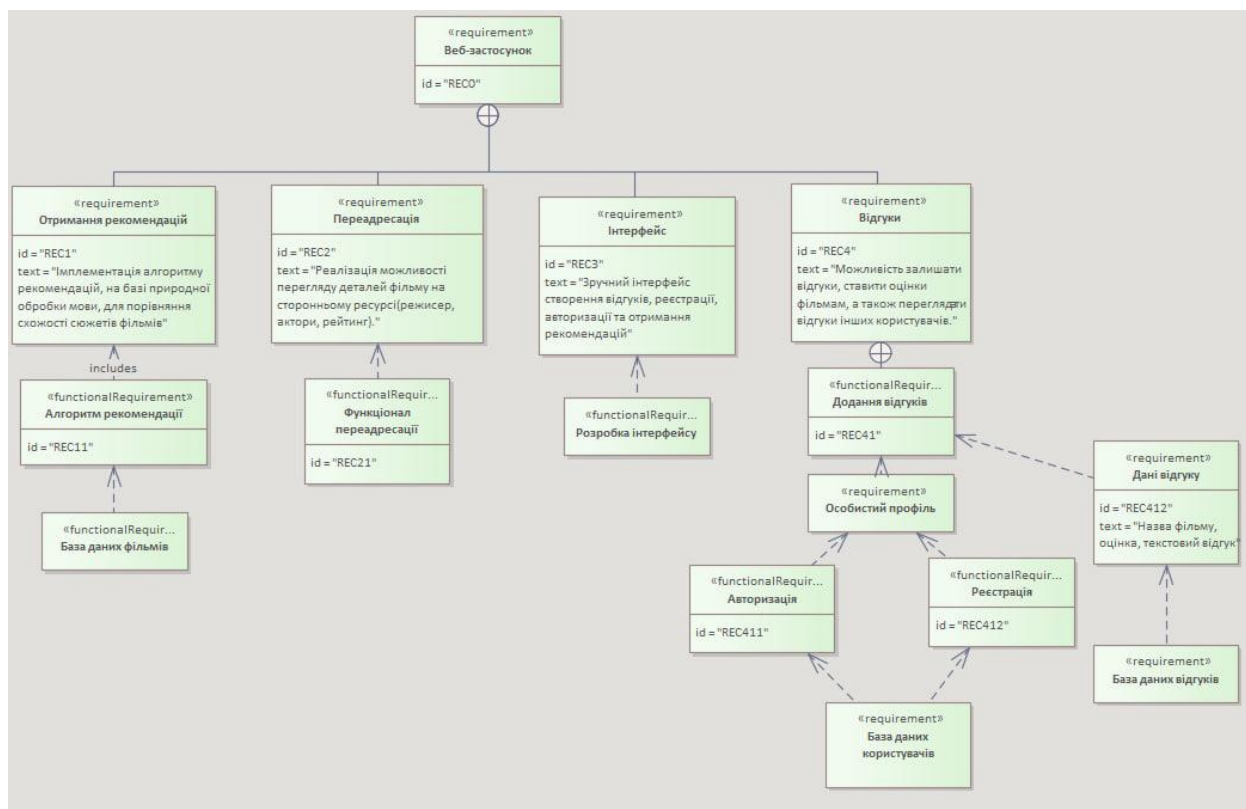


Рисунок 2.1 – Діаграма ієрархії функціональних вимог

Алгоритм рекомендації. Ця вимога передбачає розробку алгоритму яка на основі даних про фільми забезпечує систему рекомендацій. Цей алгоритм порівнює схожість фільмів на основі сюжету, акторів, жанру, режисеру та емоційного профілю. На основі цього аналізу алгоритм повинен здійснювати рекомендації щодо нових фільмів для перегляду користувачами [10, 11].

База даних фільмів передбачає створення та підтримку бази даних, яка матиме дві таблиці: перша таблиця зберігає назву фільму та посилання на сторонній ресурс, для переадресації. Друга таблиця зберігає необхідну інформацію для алгоритму, а саме набір тегів, завдяки яким відбувається підбір фільмів для рекомендації.

Функціонал переадресації. Ця вимога передбачає використання таблиці бази даних фільмів в якій знаходяться посилання на сторонній ресурс з інформацією про обраний фільм задля здійснення переадресації користувача для отримання ним повної інформації про рекомендований фільм.

Розробка інтерфейсу. Ця вимога передбачає створення користувацького інтерфейсу, який буде забезпечувати зручний доступ користувачів до функціональності системи, а саме: Створення профілю, Рекомендація фільмів і Залишення відгуку.

Створення профілю. Система повинна надавати форму реєстрації з необхідними полями для введення основної інформації (ім'я, електронна пошта, пароль).

Відображення повідомлення або переадресація на сторінку входу після успішної реєстрації.

Рекомендація фільмів. Забезпечення текстового поля для введення назви фільму.

Виведення результатів у зручному форматі, наприклад, у вигляді списку карток фільмів.

Залишення відгуку. Забезпечення інтерфейсу для введення текстового відгуку.

Забезпечення кнопки або іншого елемента для виправлення чи зміни тексту відгуку.

Забезпечення кнопки або іншого елемента для видалення відгуку під час перегляду власних відгуків.

Авторизація. Забезпечення інтерфейсу для введення електронної пошти або імені користувача та пароля.

Забезпечення кнопки або іншого елемента для виходу з системи.

Додавання відгуків. Ця вимога передбачає можливість користувачів системи залишати свої відгуки та коментарі щодо переглянутих фільмів. Ця функціональність є важливою для створення спільноти користувачів та обміну думками про фільми.

Дані відгуку. Ця вимога передбачає перевірку цілісності відгуку та відповідність вимогам, таким як наявність фільму в базі даних, числова оцінка в межах від 1 до 10 та неов'язковий текстовий відгук, розмір якого не перебільшує певну кількість символів.

База даних відгуків. Ця вимога передбачає наявність структурованої бази даних, в якій зберігатимуться відгуки користувачів про переглянуті фільми. Структура бази даних складається з наступних елементів: назва фільму, числова оцінка, текстовий відгук та користувач.

Авторизація. Ця вимога передбачає функціонал перевірки даних, введених користувачем, для авторизації в систему.

Реєстрація. Ця вимога передбачає функціонал перевірки даних, введених користувачем, а саме: вимоги до мінімальної довжини та складності паролю, неповторність імені користувача та валідність електронної пошти.

База даних користувачів. Ця вимога передбачає створення та управління базою даних, в якій зберігатимуться дані про користувачів системи, а саме: унікальний ID, ім'я користувача, електронна пошта, пароль, та відгуки, які йому належать.

Стейкхолдери. IT розробники, менеджери, адміністратори.

Бізнес-цілі. IT Розробники:

- Оптимізація алгоритму рекомендації фільмів;
- Покращення інтерфейсу застосунку.

Адміністратори. Покращення задоволення користувачів під час користування застосунком

Менеджери. Оптимізація процесу залучення в систему інформації стосовно нових фільмів

Бізнес-вимоги до ІС. Імплементація алгоритму рекомендацій, на базі природної обробки мови, для порівняння схожості сюжетів фільмів.

Зручний інтерфейс створення відгуків, реєстрації, авторизації та отримання рекомендацій.

Реалізація можливості перегляду деталей фільму на сторонньому

ресурсі(режисер, актори, рейтинг).

Можливість залишати відгуки, ставити оцінки фільмам, а також переглядати відгуки інших користувачів.

2.2. Постановка та алгоритм розв'язування задач

Задача має на меті розробку веб-застосунку для кіно-спільноти, що включає створення особистого профілю користувача, можливість додавання відгуків про фільми та основну функціональність - алгоритм рекомендації фільмів на основі порівняння сюжетів та емоційного тону фільмів.

Розробка цього веб-застосунку може сприяти залученню більшої аудиторії до онлайн-платформи кіноспільноти, підвищити її конкурентоспроможність та забезпечити більш ефективну взаємодію між користувачами та платформою.

Потреба у розв'язанні цієї задачі за допомогою електронно-обчислювальних машин (ЕОМ) обумовлена потребою в обробці великої кількості даних, що стосуються інформації про фільми та користувачів. ЕОМ можуть ефективно аналізувати ці дані, застосовуючи алгоритми машинного навчання та обробки природної мови для створення персоналізованих рекомендацій, що відповідають індивідуальним потребам та уподобанням користувачів. Такий підхід дозволить забезпечити якісний та ефективний сервіс для кіноспільноти, що використовується за допомогою веб-застосунку.

Об'єкти управління. Користувачі, фільми, відгуки, алгоритм рекомендації.

Таблиця 2.1 – Перелік та опис вихідних повідомлень

<i>Назва вихідного повідомлення</i>	<i>Ідентифікатор</i>	<i>Форма подання і вимоги до неї</i>	<i>Користувача інформації</i>
Масив рекомендованих фільмів	RECFILMS	Файл, екранна форма	Клієнт
Особистий профіль	PERSONAL_PROFILE	Файл, екранна форма	Клієнт

Масив рекомендованих фільмів:

Призначення. Ця вихідна інформація призначена для надання користувачам персоналізованих рекомендацій щодо фільмів, які вони, ймовірно, знайдуть цікавими.

Використання. Користувачі можуть переглядати цей масив рекомендацій та отримувати персоналізовані пропозиції щодо фільмів, які вони можуть бажати переглянути.

Особистий профіль.

Призначення: Користувач має опцію створити особистий профіль який містить інформацію про їхні відгуки.

Використання: Він дає можливість користувачам вести облік своїх переглядів, ділитися відгуками та взаємодіяти з іншими користувачами.

Вхідним повідомленнями є набір електронних форм, які використовуються в подальших функціях ІС.

Таблиця 2.2 – Набір електронних форм для відображення характеристик фільмів

<i>Назва вхідного повідомлення</i>	<i>Ідентифікатор</i>	<i>Форма подання</i>	<i>Джерело</i>
Назва фільму	FILMN	Електронна форма	Ручне введення користувачем
Оцінка фільму	FILMS	Електронна форма	Ручне введення користувачем
Текстовий відгук до фільму	FILMT	Електронна форма	Ручне введення користувачем
Електронна пошта користувача	EMAIL	Електронна форма	Ручне введення користувачем
Пароль користувача	PASS	Електронна форма	Ручне введення користувачем

Назва фільму – назва фільму, до якого користувач хоче отримати рекомендації.

Оцінка фільму – числове значення від 1 до 10, яке відповідає оцінці фільму від користувача.

Текстовий відгук до фільму – необов’язкова частина відгуку з текстовою рецензією на фільм.

Електронна пошта користувача – адреса електронної пошти, яка використовується користувачем для реєстрації та подальшої авторизації.

Пароль користувача – пароль користувача, який використовується користувачем для реєстрації та подальшої авторизації.

Інформація повинна відображатися після введення даних користувачем.

Автоматизоване розв'язання може припинитися у разі виникнення технічних проблем, збоїв в мережі, або якщо користувач вирішить вийти з системи.

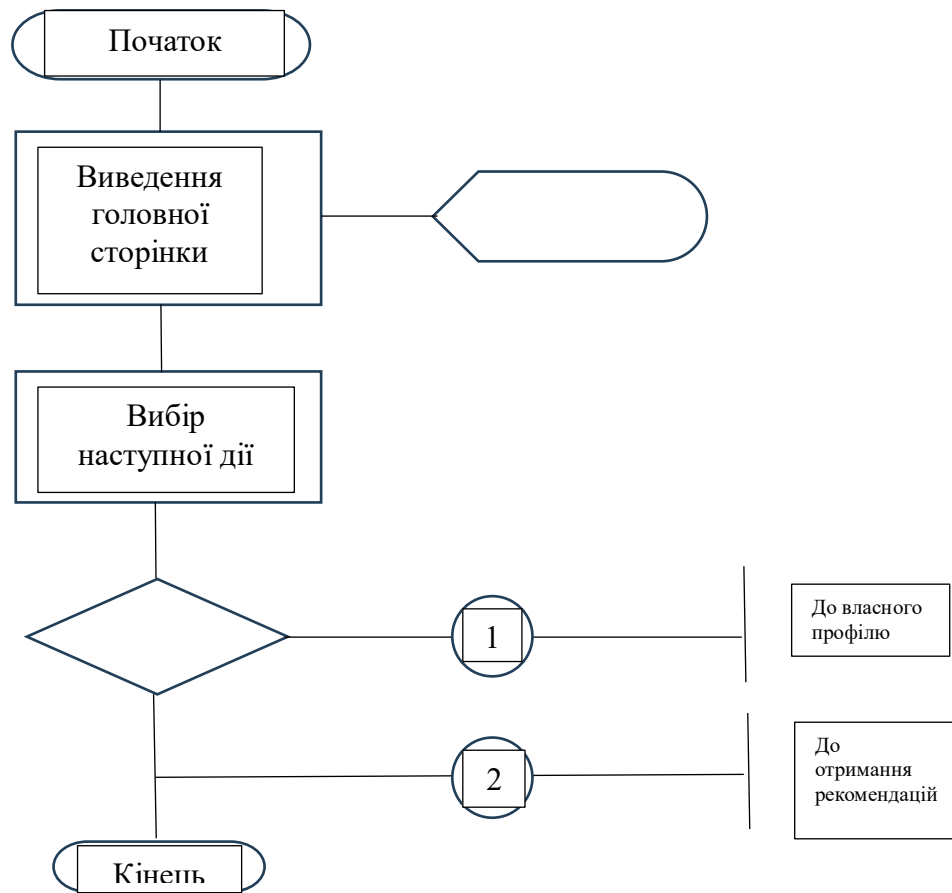


Рисунок 2.2 – Алгоритм розв'язання задачі Лист 1

2.3. Алгоритм розв'язання задачі

Економіко-математична модель та алгоритм розв'язання

Математичний опис:

$$S = \begin{vmatrix} S_{11} & S_{12} & \cdots & S_{1n} \\ S_{21} & S_{22} & \cdots & S_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ S_{n1} & S_{n2} & \vdots & S_{nn} \end{vmatrix}, \quad (2.1)$$

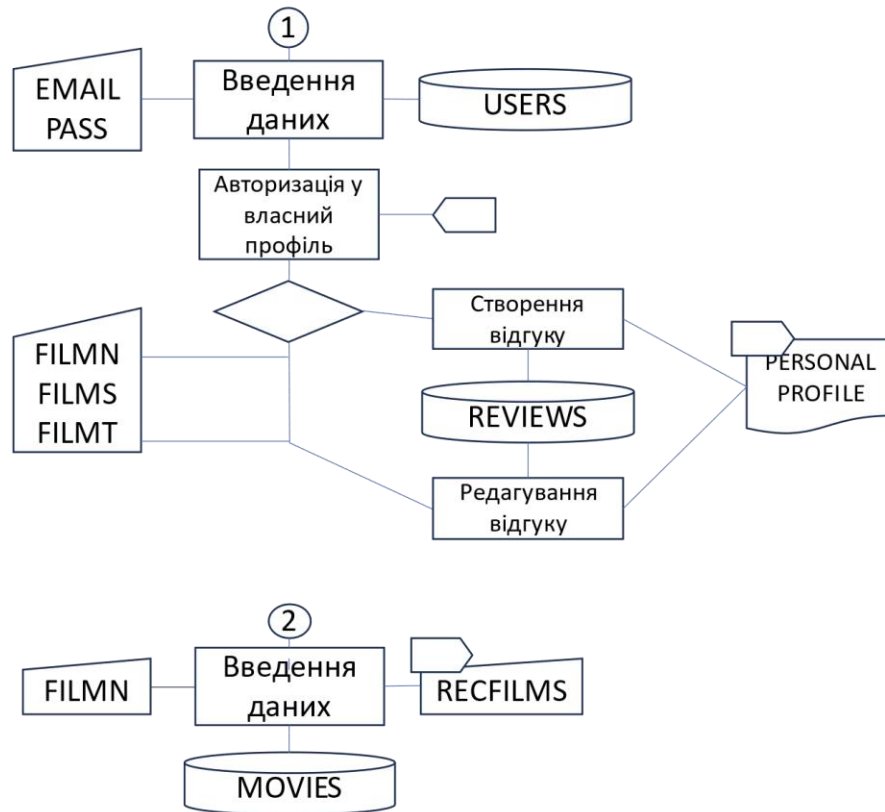


Рисунок 2.2 – Лист 2

$$\begin{aligned}
 S_{nn} &= cosine_{similarity}(Film1, Film2) = \\
 &= \frac{\sum_{i=1}^i Film1_i Film2_i}{\sqrt{\sum_{i=1}^n Film1_i^2} \cdot \sqrt{\sum_{i=1}^n Film2_i^2}},
 \end{aligned}
 \tag{2.2}$$

де: S – матриця порівнянь кожного фільму до всіх інших;

S_{nn} – числовий показник функції *cosine_similarity* між сюжетами двох фільмів, оброблених за допомогою алгоритмів природної мови;

$Film1/2$ – сюжет фільму, оброблений за допомогою алгоритмів природної мови.

$$T = \{w_1, w_1, \dots, w_n\} \tag{2.3}$$

$$C(e) = \sum_{i=1}^n L(e, w_i),$$

$$P(e) = \frac{C(e)}{n},$$

$$E = [P(e_1), P(e_2), \dots, P(e_m)],$$

де: T – набір слів (w) із тексту;

$L(e, w_i)$ – функція пошуку, яка повертає 1 , якщо слово w пов'язане з емоцією e , і 0 в іншому випадку;

$C(e)$ – обчислення частоти для кожної емоції (e);

$P(e)$ – нормалізація частоти емоції, де n – загальна кількість емоцій;

E – репрезентація емоцій як вектор.

Приклад виконання роботи алгоритму отримання емоційного тону тексту - в даному прикладі, аналізуються теги фільму «Аватар»:

[*'In', 'the', '22nd', 'century', 'a', 'paraplegic', 'Marine', 'is', 'dispatched', 'to', 'the', 'moon', 'Pandora', 'on', 'a', 'unique', 'mission', 'but', 'becomes', 'torn', 'between', 'following', 'orders', 'and', 'protecting', 'an', 'alien', 'civilization.', 'Action', 'Adventure', 'Fantasy', 'ScienceFiction', 'culture clash', 'future', 'spacewar', 'spacecolony', 'society', 'spacetravel', 'futuristic', 'romance', 'space', 'alien', 'tribe', 'alienplanet', 'cgi', 'marine', 'soldier', 'battle', 'loveaffair', 'antiwar', 'powerrelations', 'mindandsoul', '3d', 'SamWorthington', 'ZoeSaldana', 'SigourneyWeaver', 'JamesCameron'*].

$T = \{ \dots \text{torn, between, following, orders, and, protecting, an, alien, civilization} \dots \}$

$$L(\text{negative}, \text{torn}) = 1$$

$$L(\text{positive}, \text{protecting}) = 1;$$

$$L(\text{trust}, \text{protecting}) = 1;$$

$$P(\text{positive}) = \frac{5}{27} = 0.1851851852$$

$$E = [(\text{'positive'}, 18.51851851851852), (\text{'trust'}, 18.51851851851852), (\text{'negative'}, 14.814814814814813), (\text{'fear'}, 11.111111111111111), (\text{'surprise'}, 7.4074074074074066), (\text{'disgust'}, 7.4074074074074066), (\text{'sadness'}, 7.4074074074074066), (\text{'anger'}, 7.4074074074074066), (\text{'anticipation'}, 3.7037037037037033), (\text{'joy'}, 3.7037037037037033)]$$

Як видно з прикладу, кожне слово оцінюється на емоцію, яка з ним корелює. У деяких слів може бути декілька емоцій. Даний процес повторюється для кожного слова тексту. Після чого, вираховується загальна сума отриманих емоцій та відсоток кожної окремої емоції від загальної суми. В результаті виконання алгоритму отримуємо вектор відсотковим значенням присутності емоцій. Всього доступно 8 емоцій: anger, anticipation, disgust, fear, joy, sadness, surprise, trust (гнів, очікування, огида, страх, радість, сум, здивування, довіра).

Фільми одного жанру можуть кардинально відрізнятися за своїм емоційним тоном. Для наступного прикладу було використано два фільми “The Amazing Spider-Man” та “Pirates of the Caribbean: At World's End”.

Ці фільми можуть викликати різні емоції під час перегляду, незважаючи на те що обидва фільми належать до жанру пригодницьких екшенів. Ці емоції виникають через різні теми, сюжетні лінії, персонажі та візуальні стилі фільмів.

Пітер Паркер – герой з добрим серцем, який намагається знайти баланс між своїм особистим життям та обов'язками супер героя. Глядачі можуть відчувати симпатію та співчуття до нього, особливо через його особисті втрати та боротьбу з внутрішніми конфліктами.

Джек Горобець – харизматичний і не зовсім передбачуваний капітан піратського корабля, який постійно потрапляє у різні неприємності. Його пригоди та гумор можуть викликати сміх і захоплення, але також тривогу через небезпеки, з якими він стикається.

The Amazing Spider-Man має сучасний урбаністичний стиль з багатьма сценами нічного міста і динамічними бойовими сценами. Цей стиль підкреслює відчуття сучасності, реальності та напруги.

Візуальний стиль фільму Pirates of the Caribbean: At World's End більш фентезійний з елементами морських пригод, старовинних кораблів і екзотичних локацій. Це створює атмосферу загадковості, пригод та епічних подій.

В The Amazing Spider-Man глядачі можуть відчувати сильну емоційну залученість через ідентифікацію з головним героєм та його проблемами.

Напрочуд, в фільмі Pirates of the Caribbean: At World's End глядачі можуть

відчувати більше захоплення та адреналіну через непередбачувані пригоди та гумор фільму.

Аналіз емоцій фільму The Amazing Spider-Man:

['Peter', 'Parker', 'is', 'an', 'outcast', 'high', 'schooler', 'abandoned', 'by', 'his', 'parents', 'as', 'a', 'boy', 'leaving', 'him', 'to', 'be', 'raised', 'by', 'his', 'Uncle', 'Ben', 'and', 'Aunt', 'May.', 'Like', 'most', 'teenagers', 'Peter', 'is', 'trying', 'to', 'figure', 'out', 'who', 'he', 'is', 'and', 'how', 'he', 'got', 'to', 'be', 'the', 'person', 'he', 'is', 'today.', 'As', 'Peter', 'discovers', 'a', 'mysterious', 'briefcase', 'that', 'belonged', 'to', 'his', 'father', 'he', 'begins', 'a', 'quest', 'to', 'understand', 'his', "parents'", 'disappearance', '—', 'leading', 'him', 'directly', 'to', 'Oscorp', 'and', 'the', 'lab', 'of', 'Dr.', 'Curt', 'Connors', 'his', "father's", 'former', 'partner.', 'As', 'Spider-Man', 'is', 'set', 'on', 'a', 'collision', 'course', 'with', "Connors'", 'alter', 'ego', 'The', 'Lizard', 'Peter', 'will', 'make', 'life-altering', 'choices', 'to', 'use', 'his', 'powers', 'and', 'shape', 'his', 'destiny', 'to', 'become', 'a', 'hero.', 'Action', 'Adventure', 'Fantasy', 'lossoffather', 'vigilante', 'serum', 'marvelcomic', 'scientificexperiment', 'spiderbite', 'maskedvigilante', 'reboot', 'superpowers', 'geneticengineering', 'socialoutcast', 'duringcreditsstinger', 'AndrewGarfield', 'EmmaStone', 'RhysIfans', 'MarcWebb']

$E = [(negative, 20.0), (fear, 20.0), (anger, 15.0), (sadness, 15.0), (disgust, 10.0), (surprise, 10.0), (positive, 5.0), (trust, 5.0)]$

Аналіз емоцій фільму Pirates of the Caribbean: At World's End:

['Captain', 'Barbossa', 'long', 'believed', 'to', 'be', 'dead', 'has', 'come', 'back', 'to', 'life', 'and', 'is', 'headed', 'to', 'the', 'edge', 'of', 'the', 'Earth', 'with', 'Will', 'Turner', 'and', 'Elizabeth', 'Swann.', 'But', 'nothing', 'is', 'quite', 'as', 'it', 'seems.', 'Adventure', 'Fantasy', 'Action', 'ocean', 'drugabuse', 'exoticisland', 'eastindiatradingcompany', "loveofone'slife", 'traitor', 'shipwreck', 'strongwoman', 'ship', 'alliance', 'calypso', 'afterlife', 'fighter', 'pirate', 'swashbuckler', 'aftercreditsstinger', 'JohnnyDepp', 'OrlandoBloom', 'KeiraKnightley', 'GoreVerbinski']

$E = [(negative, 21.428571428571427), (anticipation, 14.285714285714285), (trust, 14.285714285714285), (anger, 14.285714285714285), (fear, 14.285714285714285), (sadness, 14.285714285714285), (disgust, 7.142857142857142)]$

Застосування алгоритму аналізу емоцій до обох фільмів підтверджує цю теорію, адже для *The Amazing Spider-Man* найчастішими емоціями є: негативні, страх та злість, в той час як для *Pirates of the Caribbean: At World's End* найчастішими емоціями, які він викликає, є: негативні, очікування та довіра.

Не зважаючи на такі явні відмінності, типовий алгоритм рекомендації ймовірно буде вважати ці два фільми схожими між собою, адже вони підпадають під один жанр.

Одна з найбільших проблем, яка з'являється під час типової рекомендації фільмів, постає під час пошуку фільмів в жанрі горрор. Проблема полягає в тому, що жанр горрор є дуже широким і має різноманітні піджанри та підходи до створення емоційного напруження у глядачів. Наприклад, фільми, як *"Hereditary"* і *"The Conjuring"*, є частиною цього жанру, але їх підходи до створення страху суттєво відрізняються.

"Hereditary" розміщується в жанрі психологічного жаху, де більша увага приділяється побудові атмосфери, психологічним травмам персонажів і внутрішнім конфліктам. Фільм викликає страх за рахунок поглибленого використання напруження та внутрішньої напруженості.

"The Conjuring" більше схильний до традиційного жаху з акцентом на неочікуваності та стрибках у сценах.

Аналіз емоцій фільму Hereditary:

['a', 'miniatures', 'artist', 'who', 'struggles', 'with', 'the', 'recent', 'loss', 'of', 'her', 'mother,', 'Ellen.', 'As', 'Annie', 'tries', 'to', 'cope', 'with', 'her', 'grief,', 'her', 'family,', 'including', 'her', 'husband', 'Steve,', 'son', 'Peter,', 'and', 'daughter', 'Charlie,', 'grapple', 'with', 'increasingly', 'disturbing', 'occurrences', 'that', 'suggest', 'a', 'sinister', 'and', 'supernatural', 'presence', 'haunting', 'their', 'lives.']

$$E = [('negative', 19.51219512195122), ('sadness', 17.073170731707318),$$

$$('positive', 17.073170731707318), ('anger', 9.75609756097561),$$

$$('fear', 9.75609756097561), ('disgust', 7.317073170731707),$$

$$('anticipation', 4.878048780487805), ('surprise', 4.878048780487805),$$

$$('joy', 4.878048780487805), ('trust', 4.878048780487805)]$$

Аналіз емоцій фільму The conjuring:

['Paranormal', 'investigators', 'Ed', 'and', 'Lorraine', 'Warren', 'work', 'to', 'help', 'a', 'family', 'terrorized', 'by', 'a', 'dark', 'presence', 'in', 'their', 'farmhouse.', 'Forced', 'to', 'confront', 'a', 'powerful', 'entity,', 'the', 'Warrens', 'find', 'themselves', 'caught', 'in', 'the', 'most', 'terrifying', 'case', 'of', 'their', 'lives.', 'Horror', 'Thriller', 'sistersisterrelationship', 'exorcism', 'rhodeisland', 'basedontruestory', 'farmhouse', 'paranormalinvestigation', 'ghost', 'supernaturalpower', 'paranormalinvestigator', 'annabelle', 'PatrickWilson', 'VeraFarmiga', 'LiliTaylor', 'JamesWan']

$$E = [('fear', 23.52941176470588), ('sadness', 17.647058823529413),$$

$$('positive', 11.76470588235294), ('anger', 11.76470588235294),$$

$$('negative', 11.76470588235294), ('anticipation', 5.88235294117647),$$

$$('disgust', 5.88235294117647), ('joy', 5.88235294117647),$$

$$('trust', 5.88235294117647)]$$

Як видно з аналізів емоцій фільмів, хоч вони виглядають досить схожими, але є одна ключова відмінність. В фільмі, який більше орієнтується на психологічний жах та атмосферу, переважною емоцією є «негативні», проте як в фільмі, який розраховує на неочікувані жахи, переважною є «страх». Таким чином видно, що завдяки аналізу емоцій можна відрізнити ці два під жанри фільмів жахів, коли як традиційна система рекомендацій вважатиме їх схожими.

Робота алгоритму поділяється на дві частини: очистка даних та, безпосередньо, рекомендація фільмів.

Очистка даних. На початку роботи, програма зчитує два CSV-файли: "credits.csv" і "movies.csv", що містять інформацію про людський склад фільмів (акторів, режисерів) та загальну інформацію про фільми відповідно. Відбирає необхідні стовпці зі структури "movies.csv", які включають унікальний ідентифікатор фільму, назву, опис, жанри, ключові слова, акторів та екіпаж.

Наступним етапом в роботі є обробка отриманих даних.

Для стовпців "genres" та "keywords", які містять дані у форматі JSON-масивів, виконується функція "convert", яка перетворює ці дані у буквальні значення.

Такому ж процесу підлягають актори та режисер, тільки кількість акторів обмежується трьома.

Далі, створюється датафрейм “df”, стовпцем “tags” якого є об’єднання всіх тегів фільму, таких як жанр, ключові слова, актори та режисер.

Далі йде отримання емоційного профілю фільму.

В рамках прототипу, використовується бібліотека NRCLex для аналізу емоційного тону тексту. Ця бібліотека містить словник, який мапує слова на їхні емоційні оцінки. Вхідний рядок tags передається у конструктор NRCLex, який створює об’єкт для подальшого аналізу емоційного тону. Згодом, викликається метод raw_emotion_scores, який повертає словник з емоційними оцінками. Ключі цього словника відповідають різним емоціям, таким як 'anger', 'fear', 'joy', 'sadness' тощо, а значення - кількостям, що відображають інтенсивність кожної емоції у тексті. Результатом функції є словник з емоційними оцінками тегів тексту, тобто, емоційний профіль.

Наступним кроком роботи є додавання емоцій до “df”.

Спочатку обчислюється загальна сума всіх емоційних оцінок. Потім для кожної емоції обчислюється відсоток від загальної суми. Емоції сортуються в порядку спадання їх відсоткових значень. Перші три емоції додаються до списку result. Якщо список result вже містить три емоції, додавання припиняється. Рядок у датафреймі “df” за індексом row оновлюється, додаючи туди рядок із топ-3 емоцій, об’єднаних у строку, за допомогою пробілу. Обробляються можливі помилки, які можуть виникнути при збереженні результатів. Це допомагає уникнути викидання виключень у випадку, якщо збереження не вдалося.

Отримання рекомендацій. Після вдалої обробки даних, алгоритм готовий до видачі рекомендацій.

Алгоритм використовує модуль CountVectorizer з бібліотеки sklearn.feature_extraction.text для перетворення текстових даних у вектори.

Створюється об’єкт cv типу CountVectorizer. Параметр max_features вказує максимальну кількість ознак, які будуть використовуватися для кодування тексту. У цьому випадку обмеженням є 5000 ознак. Параметр stop_words='english' вказує, що

будуть використані стандартні стоп-слова англійської мови, які будуть вилючені під час перетворення тексту в вектори. Згодом, викликається метод `fit_transform()` для об'єкта `cv`, який перетворює текст у векторний формат. Результат перетворення перетворюється в масив методом `toarray()`. Отримані вектори зберігаються у змінній `vectors`.

Далі, програма ініціалізує об'єкт `PorterStemmer` з бібліотеки `nlTK.stem` для використання алгоритму стемінгу Портера. Основне призначення цього об'єкта - проведення стемінгу, тобто отримання їх кореня. Стемінг є одним із способів підготовки текстових даних перед подальшою обробкою, такою як векторизація або класифікація. Він допомагає зменшити розмірність словника та уніфікувати слова, які мають спільний корінь, що полегшує аналіз тексту та покращує ефективність моделей машинного навчання.

Ця функція застосовується до всіх тегів кожного фільму.

Далі, необхідна матриця схожості, оскільки дані зберігаються у вигляді векторів. Для цього, алгоритм використовує функцію `cosine_similarity` з бібліотеки `sklearn.metrics.pairwise` для обчислення схожості між векторами, які були отримані попередньо за допомогою методу векторизації, наприклад, за допомогою `CountVectorizer`. Схожість косинуса використовується для обчислення кута між двома векторами в просторі ознак. Вона дозволяє визначити схожість між документами або об'єктами на основі їхніх ознак. Чим ближче значення схожості косинуса до 1, тим більш схожі об'єкти, а значення ближче до 0 вказує на меншу схожість. Отримана матриця схожості `similarity` містить попарні значення схожості між кожною парою векторів у вихідному наборі даних.

В кінці, застосовується функція, яка використовує усі створені до цього змінні та рекомендує фільми. На початку її роботи, здійснюється пошук індексу фільму у датафреймі `"df"`, який відповідає заданому назві фільму `movie`. За допомогою індексу фільму визначається рядок у матриці схожості `similarity`, що відповідає цьому фільму. Отримується список відстаней від цього фільму до всіх інших фільмів у наборі даних. Фільми сортуються за спаданням схожості, отриманої з матриці схожості. Для кожного фільму у відсортованому списку

перевіряється, чи співпадають його емоції з емоціями вхідного фільму. Якщо емоції співпадають, фільм додається до списку рекомендацій.

В результаті, повертається список рекомендованих фільмів, обмежений до перших 5 фільмів.

2.4. Моделювання інформаційної системи

Діаграма прецедентів (див рис. 2.3) показує взаємодію користувачів з веб-застосунком для рекомендацій фільмів. Вона містить кілька ключових прецедентів, зокрема авторизацію, реєстрацію, перегляд особистих профілів інших користувачів, отримання рекомендацій, перегляд особистого профілю та додавання відгуків.

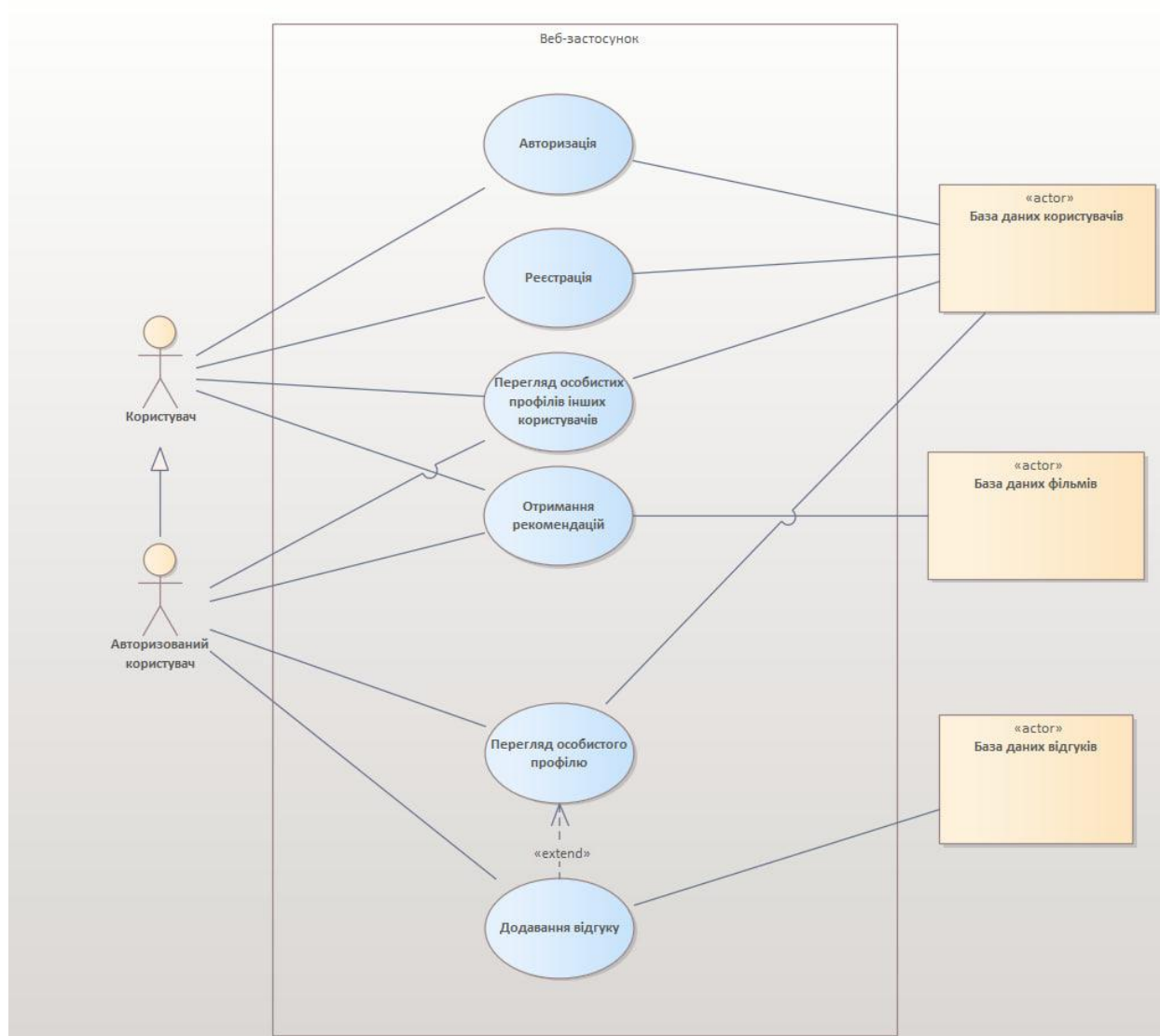


Рисунок 2.3 – Діаграма прецедентів

Користувач починає взаємодію з веб-застосунком як звичайний користувач та має доступ до отримання рекомендацій та перегляду особистих профілів інших користувачів. Також, він має змогу зареєструватися та авторизуватися для отримання доступу до розширеного функціоналу веб-застосунку, а саме перегляд особистого профілю та створення відгуків на фільми.

Система взаємодіє з трьома базами даних: користувачів, фільмів та відгуків.

Дана діаграма послідовності описує процес отримання рекомендацій фільмів користувачем у веб-застосунку. Вона містить взаємодії між чотирма основними об'єктами: User (Користувач), MovieFunc (Функція отримання рекомендації), DatabaseManager (Менеджер бази даних), і WebApplication (Веб-застосунок).

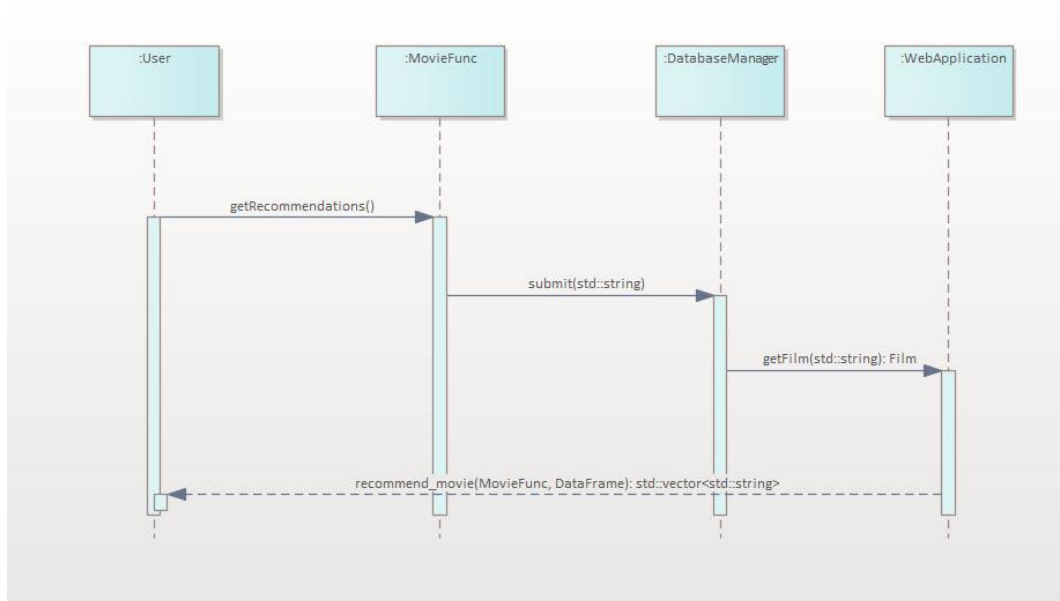


Рисунок 2.5 - Діаграма послідовності отримання рекомендацій

Процес починається, коли користувач викликає метод `getRecommendations()`. Цей запит передається до об'єкта `MovieFunc`, який виконує метод `submit(std::string)`, звертаючись до `DatabaseManager` для отримання фільму. `DatabaseManager` виконує метод `getFilm(std::string)` і повертає об'єкт фільму до `MovieFunc`. Функція `recommend_movie` обробляє отримані дані і повертає список рекомендацій користувачу у вигляді рядку та оновлює сторінку, показуючи рекомендовані фільми.

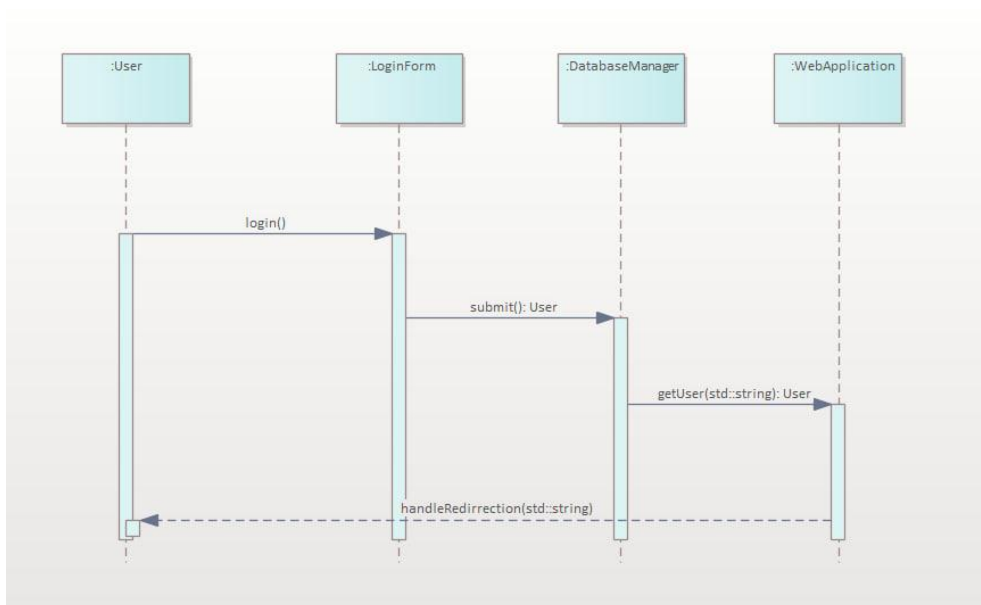


Рисунок 2.6 – Діаграма послідовності авторизації

Дана діаграма описує процес авторизації користувача до веб-застосунку. Вона містить взаємодії між чотирма основними об'єктами: User (Користувач), LoginForm (Форма обробки авторизації), DatabaseManager (Менеджер бази даних), і WebApplication (Веб-застосунок).

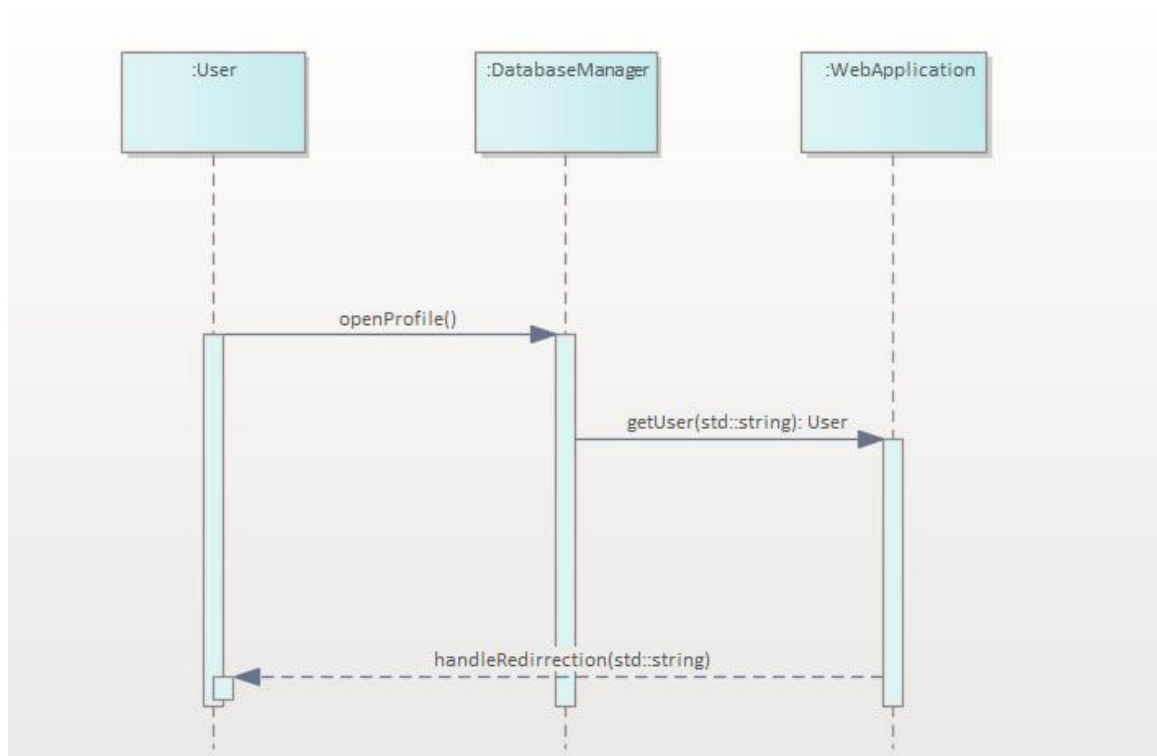


Рисунок 2.7 – Діаграма послідовності відкриття профілю

Користувач починає взаємодію з веб-застосунком як звичайний користувач та має доступ до отримання рекомендацій та перегляду особистих профілів інших користувачів. Також, він має змогу зареєструватися та авторизуватися для отримання доступу до розширеного функціоналу веб-застосунку, а саме перегляд особистого профілю та створення відгуків на фільми.

Дана діаграма описує процес перегляду особистого профілю іншого користувача веб-застосунку. Вона містить взаємодії між чотирма основними об'єктами: : User (Користувач), DatabaseManager (Менеджер бази даних), і WebApplication (Веб-застосунок).

Процес починається, коли користувач робить запит на перегляд особистого профілю іншого користувача. Запит надсилається до DatabaseManager, який перевіряє наявність даного користувача в базі даних. Якщо такий користувач існує, надсилається запит до WebApplication, який в свою чергу переадресовує користувача на запитуваний профіль.

Дана діаграма описує процес створення відгуку на фільм. Вона містить взаємодії між чотирма основними об'єктами: : User (Користувач), DatabaseManager (Менеджер бази даних), ReviewForm(Форма обробки відгуків) і WebApplication (Веб-застосунок).

Процес починається, коли користувач робить запит створення відгуку. Першочергово, цей запит оброблює ReviewForm, форма, яка перевіряє отримані дані відгуку на правильність, а саме чи існує даний цей фільм, на який користувач хоче зробити відгук та чи присутня числова оцінка фільму. Всі інші параметри відгуку вважаються необов'язковими. Якщо перевірка пройдена, дані відгуку далі відправляються до DatabaseManager, який перевіряє, чи робив вже користувач відгук на цей фільм. Якщо ні, DatabaseManager створює відгук в базі даних та додає його до особистого профілю користувача. Після чого, відбувається запит до WebApplication, який оновлює сторінку особистого профілю користувача з доданим новим відгуком.

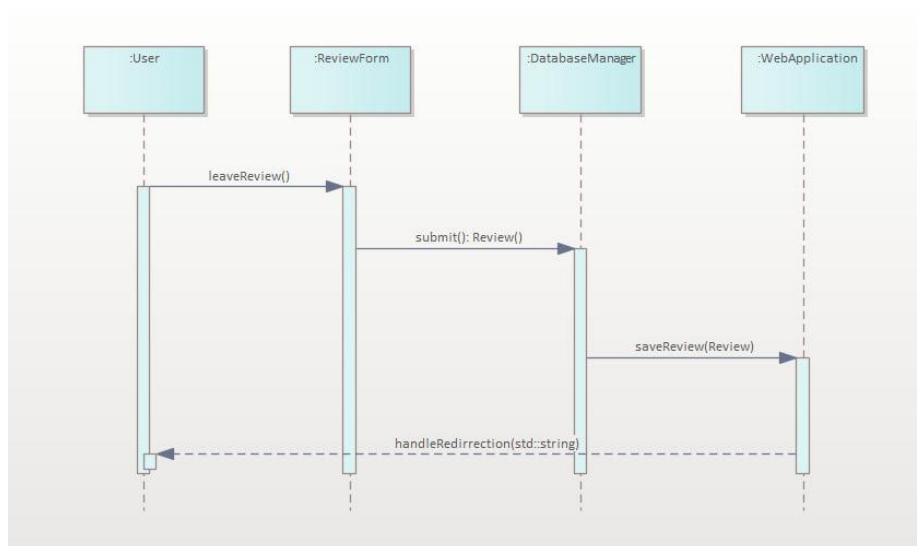


Рисунок 2.8 – Діаграма послідовності створення відгуку

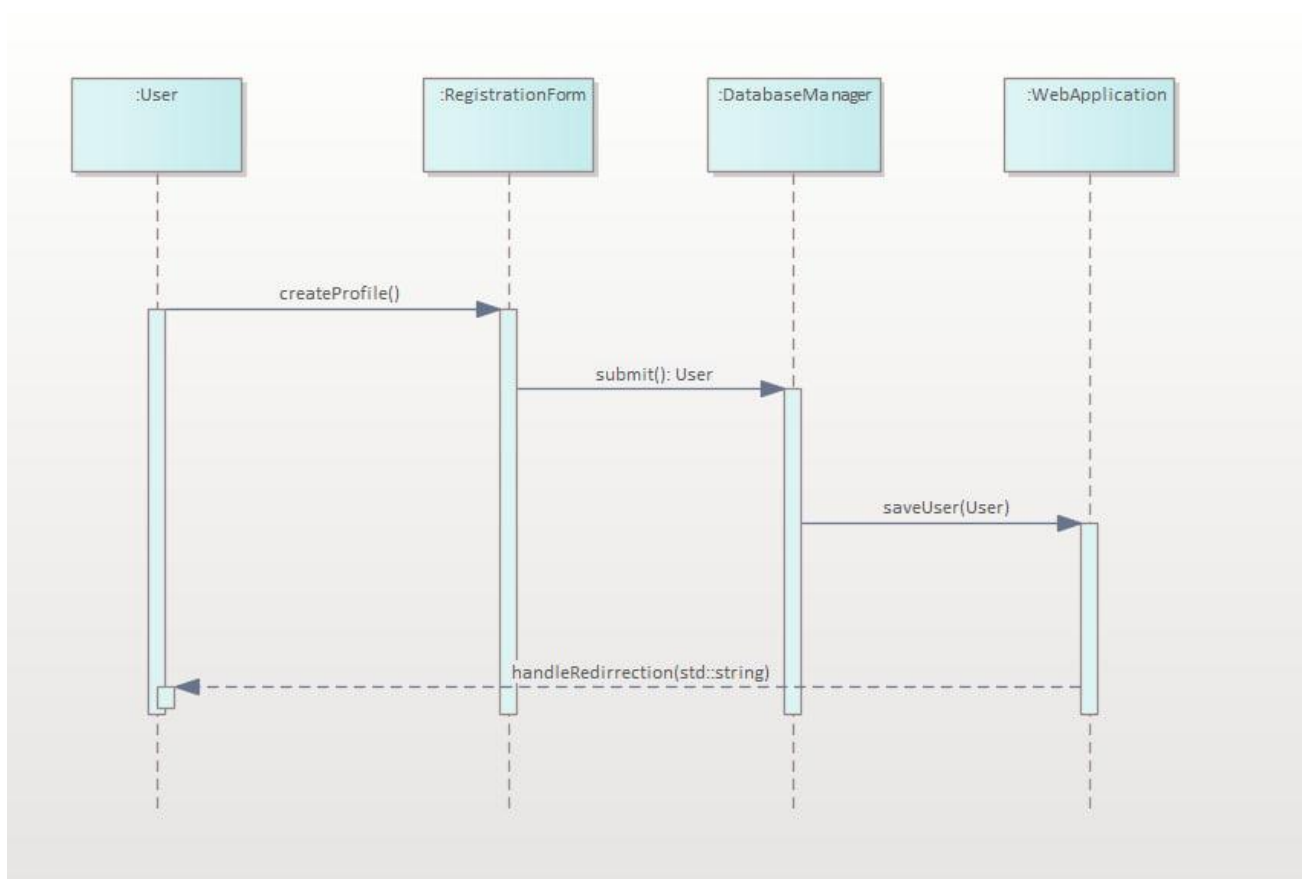


Рисунок 2.9 – Діаграма послідовності реєстрації

Дана діаграма описує процес реєстрації користувача до веб-застосунку. Вона містить взаємодії між чотирма основними об'єктами: : User (Користувач), RegistrationForm (Форма обробки реєстрації), DatabaseManager (Менеджер бази

даних), і WebApplication (Веб-застосунок).

Процес починається коли користувач викликає функцію login() при спробі авторизації та вводить свої дані для авторизації. Цей запит передається до RegistrationForm, яка, за допомогою вбудованих фільтрів, перевіряє отримані дані на відповідність правилам. Після цього, якщо перевірка успішна, надсилається запит до DatabaseManager, який в свою чергу перевіряє наявність даного користувача в базі даних. Якщо користувач в базі даних відсутній, DatabaseManager створює особистий профіль новому користувачу та WebApplication оновлює сторінку з особистим профілем даного користувача.

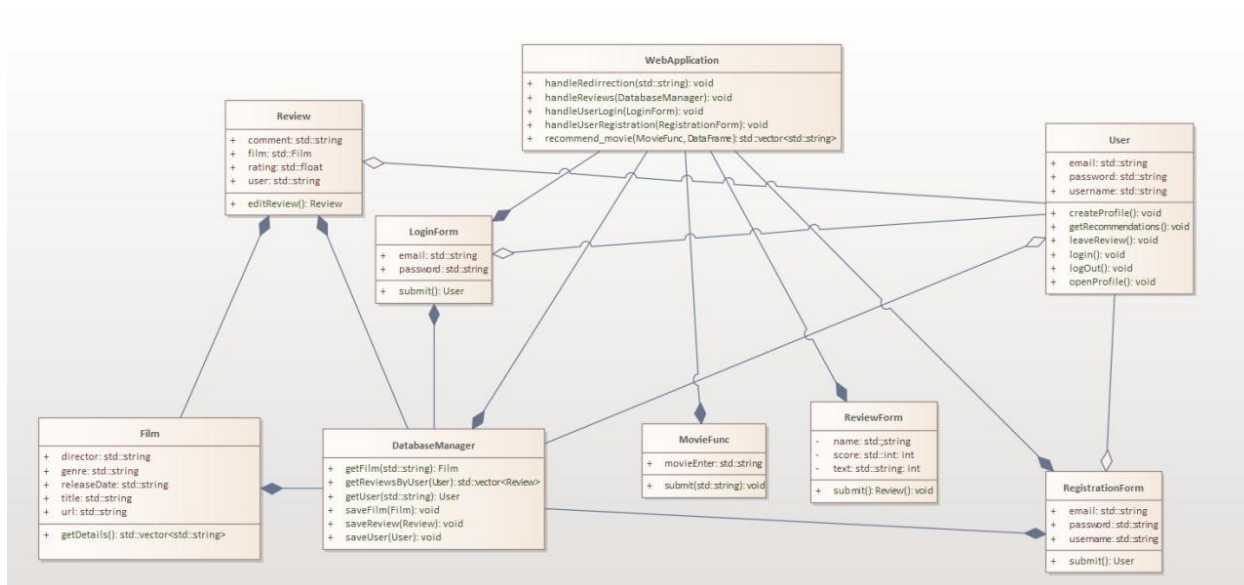


Рисунок 2.10 – Діаграма класів

Дана діаграма класів представляє структуру веб-застосунку для рекомендацій фільмів. Вона складається з кількох основних класів, кожен з яких має свої атрибути та методи.

Метод createProfile() у класі User взаємодіє з методом getUser(User) у класі DatabaseManager, який в свою чергу взаємодіє з методом submit() класу RegistrationForm та handleUserRegistration(RegistrationForm) класу WebApplication щоб обробити нову реєстрацію користувача.

Метод getRecommendations() у класі User взаємодіє з методом

`recommend_movie(MovieFunc, DataFrame)` у класі `WebApplication` для отримання рекомендацій фільмів.

Метод `leaveReview()` у класі `User` взаємодіє з методом `getReviews(Review)` у класі `DatabaseManager`, який в свою чергу взаємодіє з методом `submit()` класу `ReviewForm`, класом `Review` та `handleReviews(DatabaseManager)` у класі `WebApplication`, щоб обробити новий відгук.

Метод `logOut()` у класі `User` взаємодіє з методом `handleRedirection(std::string)` у класі `WebApplication`, щоб обробити вихід користувача.

Метод `openProfile()` у класі `User` взаємодіє з методом `handleRedirection(std::string)` у класі `WebApplication` та `getUser(User)` у класі `DatabaseManager` для перегляду профілю користувача.

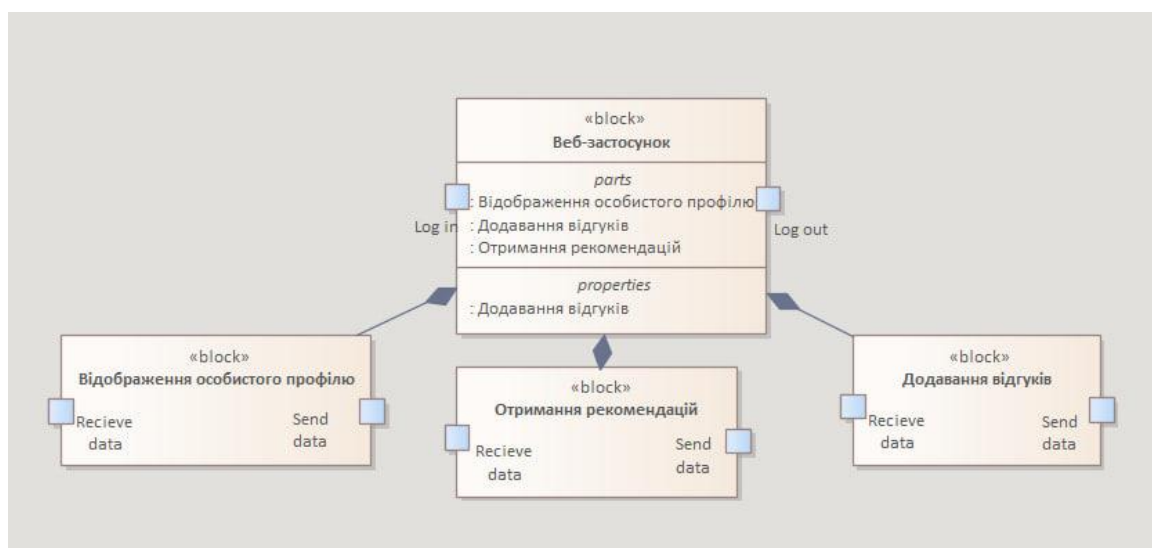


Рисунок 2.11 - Діаграма визначених блоків

Метод `getDetails()` у класі `Film` взаємодіє з методом `getFilm(std::string)` у класі `DatabaseManager`, щоб отримати деталі про фільм.

`DatabaseManager` має допоміжні методи, такі як `saveFilm()` для майбутнього поповнення бази даних фільмів.

Дана діаграма описує визначені блоки системи, а саме: Відображення особистого профілю, отримання рекомендацій та додання відгуків. Кожен блок має чітко визначені точки входу і виходу для даних.

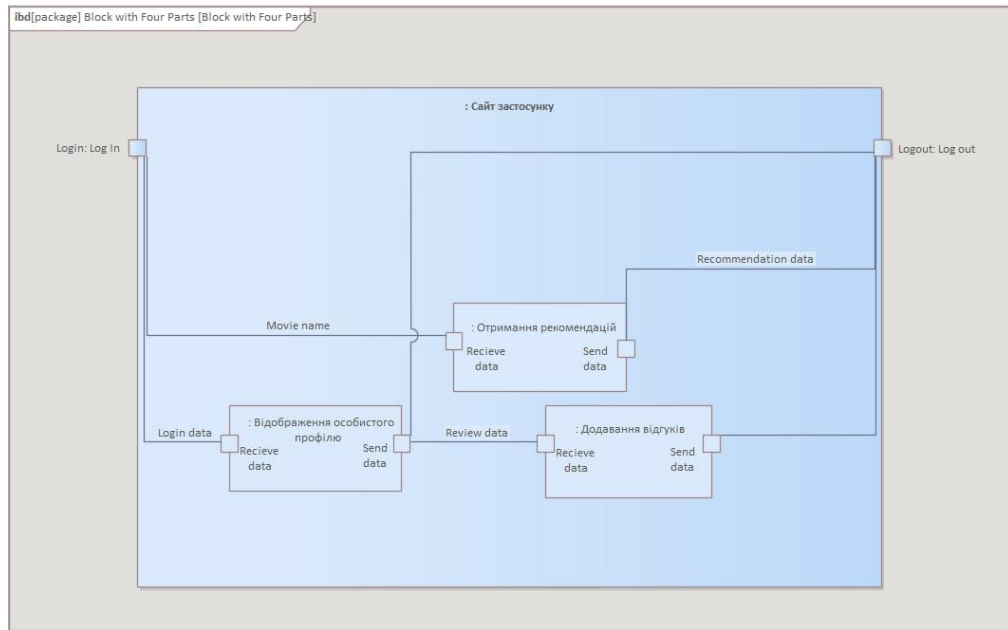


Рисунок 2.12 – Діаграма внутрішніх блоків

Дана діаграма показує структуру веб-застосунку що включає в себе функціональні блоки для роботи з даними користувача і рекомендаціями фільмів. На діаграмі відображено взаємодію між основними компонентами системи.

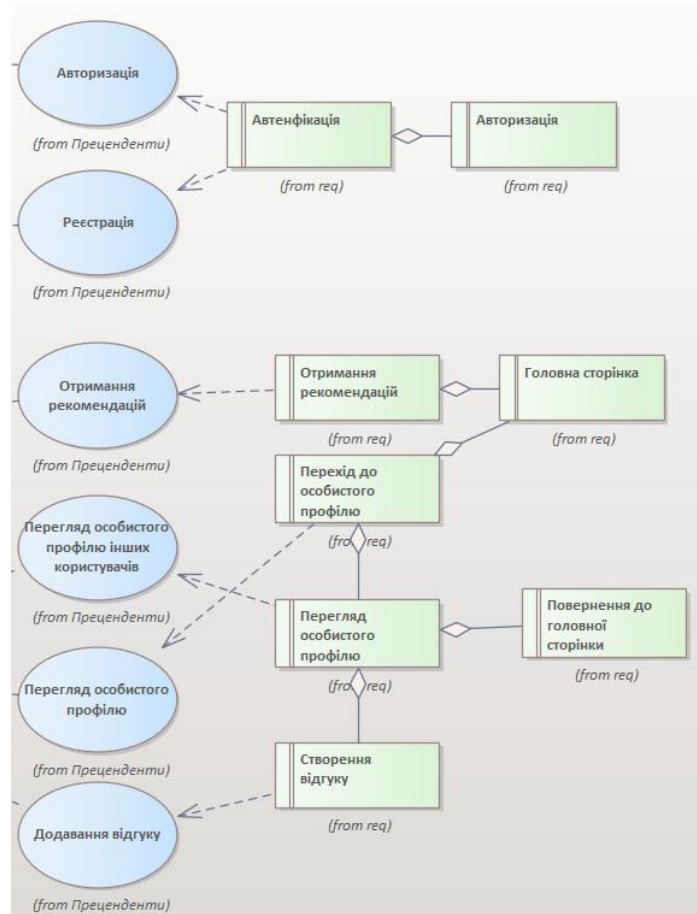


Рисунок 2.13 – Діаграма трасування

Користувач починає взаємодію з веб-застосунком як звичайний користувач та має доступ до отримання рекомендацій та перегляду особистих профілів інших користувачів. Також, він має змогу зареєструватися та авторизуватися для отримання доступу до розширеного функціоналу веб-застосунку, а саме перегляд особистого профілю та створення відгуків на фільми.

Кожен блок функціоналу отримує та передає дані: блок відображення профілю отримує і передає дані профілю, блок отримання рекомендацій отримує дані про фільми і передає рекомендації, а блок додавання відгуків отримує та передає дані відгуків.

Дані про фільми (Movie name), рекомендації (Recommendation data) та відгуки (Review data) використовуються для взаємодії між блоками системи.

Дана діаграма показує відношення прецедентів з вимогами системи. Як видно з діаграми, на кожен вимогу припадає прецедент системи. Для зображення взаємодій в табличному вигляді прикріплено матрицю взаємозв'язків (див рис. 2.14).

Target +	Преценденти::Авторизація	Преценденти::Додавання від	Преценденти::Отримання ре	Преценденти::Перегляд особ	Преценденти::Перегляд особ	Преценденти::Реєстрація
+ Source						
req::Автенфікація	↑					↑
req::Авторизація						
req::Головна сторінка						
req::Отримання рекомен...			↑			
req::Перегляд особистого...					↑	
req::Перехід до особистог...				↑		
req::Повернення до голов...						
req::Створення відгуку		↑				

Рисунок 2.14 – Матриця взаємозв'язків

РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1. Інформаційне забезпечення

Для забезпечення ефективного зберігання та доступу до інформації використовується база даних SQLite. Структура даних включає таблиці для зберігання інформації про користувачів, фільми, відгуки та інші важливі дані.

SQLite було обрано як СКБД через його легкість використання, швидкість та невеликі ресурси, необхідні для функціонування. Крім того, він забезпечує надійність та стабільність для операцій з даними.

Контроль за інформацією забезпечується за допомогою механізмів, вбудованих у SQLite, таких як права доступу та обмеження доступу до бази даних. Захист даних від несанкціонованого доступу та відновлення даних у випадку аварійного видалення або пошкодження також забезпечується за допомогою вбудованих засобів SQLite.

Загальна схема інформаційного забезпечення включає базу даних SQLite, що використовується для зберігання інформації, а також програмні інтерфейси для взаємодії з даними. Конкретні елементи, такі як таблиці бази даних та структура даних, були розроблені з урахуванням вимог та потреб системи.

В якості первинних даних був обраний публічно доступний датасет з фільмами. В ньому знаходяться дані про більш ніж 5000 фільмів, а саме:

Назва, ідентифікаційний номер в базі фільмів IMDb, жанр, дата випуску, актори, режисер, короткий опис сюжету.

Оскільки дані про наявні фільми потрібно постійно вносити, для оновлення інформації в майбутньому, використовуватиметься ручний внесок в систему даних про нові фільми з Інтернету.

Періодичність внеску даних варіюється, оскільки їх потрібно оновлювати щоразу коли з'являється новий фільм.

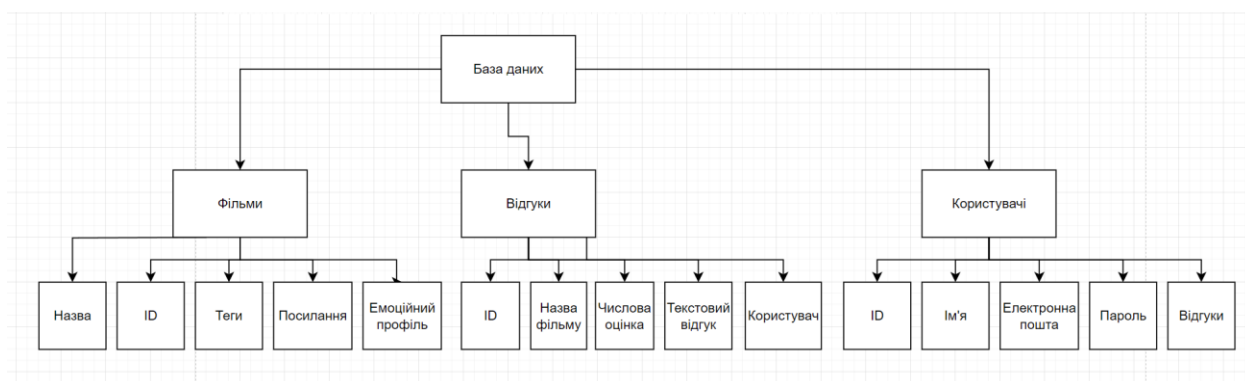


Рисунок 3.1 – Інфологічна модель бази даних

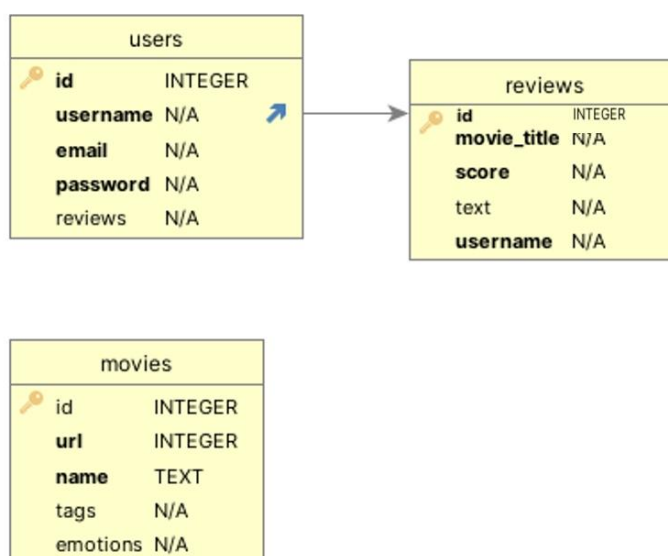


Рисунок 3.2 – Даталогічна модель бази даних

Ця даталогічна модель бази даних містить три таблиці: users, reviews, і movies. Таблиця users зберігає інформацію про користувачів із полями id (унікальний ідентифікатор), username (ім'я користувача), email (електронна пошта), password (пароль) та reviews (список відгуків користувача). Таблиця reviews містить інформацію про відгуки, що включає id (унікальний ідентифікатор відгуку), movie_title (назва фільму), score (оцінка), text (текст відгуку) та username (ім'я користувача, який залишив відгук). Таблиця movies зберігає дані про фільми з полями id (унікальний ідентифікатор фільму), url (посилання на фільм на сторонньому ресурсі), name (назва фільму), tags (теги) та emotions (емоційний профіль фільму). (див. рис. 3.1)

Зв'язок між таблицями відбувається за допомогою ключа username у таблицях users та reviews, що дозволяє зв'язати відгуки з конкретними користувачами.

3.2. Програмне забезпечення

Системне програмне забезпечення. Веб-застосунок розроблявся за допомогою фреймворку Flask на мові програмування Python. Flask є легким та простим у використанні фреймворком для розробки веб-додатків. Він має просту структуру та мінімалістичний дизайн, що дозволяє швидко створювати та вдосконалювати веб-додатки. Flask має велику кількість розширень та бібліотек, які полегшують розширення функціональності додатків. Це дозволяє використовувати готові рішення для реалізації різних завдань та швидко розвивати проект [6, 7, 8].

Прикладне програмне забезпечення. Функціонал даної інформаційної системи залежить від декількох розроблених програмних забезпечень.

Першочергово, для роботи системи рекомендацій необхідно дві частини: система очистки даних та, безпосередньо, алгоритм рекомендації

Система очистки даних є обов'язковою, оскільки для того щоб оновлювати базу даних новими фільмами, кожен новий фільм потрібно піддавати очистці автоматично.

Необхідно також розробити програмне забезпечення для взаємодії з базою даних. Дане програмне забезпечення має безпосередній зв'язок з вибраною СУБД, оскільки використовую бібліотеку sqlite3. Бібліотека sqlite3 для Python - це інструмент, який дозволяє взаємодіяти з базою даних SQLite з використанням мови програмування Python. Вона надає зручний інтерфейс для виконання різних операцій з базою даних SQLite, таких як створення та з'єднання з базою даних, виконання SQL-запитів (створення, читання, оновлення, видалення даних), управління транзакціями, тощо.

Програмна документація

Формуляр документації про програму. Цей проект - веб-застосунок, призначений для створення та підтримки кіноспільноти. Основна функціональність включає в себе можливість створення особистого профілю, додавання відгуків до фільмів та отримання рекомендаційних фільмів на основі порівняння сюжетів та емоційного тону фільмів.

Основні характеристики. Користувачі можуть створювати особисті профілі для взаємодії з іншими учасниками кіноспільноти та відстежування своїх вподобань.

Користувачі можуть додавати відгуки та оцінки до переглянутих фільмів, щоб ділитися своїми враженнями з іншими учасниками спільноти.

Система використовує алгоритм порівняння сюжетів та емоційного тону фільмів для надання користувачам персоналізованих рекомендацій фільмів, які відповідають їхнім інтересам та вподобанням.

Комплектність:

- Створення особистого профілю;
- Додавання відгуків;
- Алгоритм рекомендацій.

Ця програма призначена для покращення користувацького досвіду у сфері кіно. Розроблений алгоритм може використовуватись в багатьох різних системах, пов'язаних з фільмами.

Керівництво системного програміста

Кроки для перевірки та забезпечення функціонування програми:

- Перевірте вимоги до середовища:

Переконайтеся, що ваше середовище розробки відповідає вимогам програми щодо версій мов програмування, бібліотек та інших залежностей.

- Встановіть необхідні бібліотеки та інструменти:

Встановіть усі необхідні бібліотеки та інструменти, які використовуються у

програмі, щоб забезпечити правильне функціонування.

До необхідних сторонніх бібліотек та інструментів, які необхідно встановити, входять: flask, pandas, forms, sklearn, ast, SQLite3, numpy, re, nrclex, nltk, jupyter notebook.

- Перевірте конфігураційні файли:

Перевірте конфігураційні файли програми та впевніться, що всі налаштування відповідають вимогам вашого середовища та конкретним потребам програми.

Кроки для налаштування програми:

- Налаштуйте базу даних:

Налаштуйте з'єднання з базою даних та впевніться, що всі таблиці та схеми бази даних створені відповідно до потреб програми.

- Конфігуруйте веб-сервер:

Налаштуйте веб-сервер для правильного виконання та розгортання програми відповідно до вимог вашого серверного середовища.

- Контрольний приклад:

Опис контрольного прикладу наданий в Додатках А, Б і В.

3.3. Результат реалізації інформаційної системи

Суть проекту полягає в розробці веб-додатку, який використовує алгоритми обробки природної мови (NLP) та аналізу емоцій для надання персоналізованих рекомендацій фільмів користувачам. Використовуючи бібліотеку NLTK для обробки тексту, система аналізує схожість сюжетів фільмів та визначає емоційний тон кожного з них. Після цього вона рекомендує користувачам фільми, які мають схожий сюжет та співпадають за емоційним впливом. Такий підхід дозволяє надавати індивідуально налаштовані рекомендації, враховуючи смаки та емоційні переваги кожного користувача.

Оригінальність цього проекту полягає в його комбінації двох ключових аспектів: аналізу сюжетів фільмів з використанням методів природної мови (NLP) та аналізу емоційного впливу цих фільмів. Більшість існуючих систем рекомендацій фільмів обмежуються лише аналізом схожості сюжетів або враховують тільки попередні дії користувачів, такі як оцінки та відгуки. Проект поєднує в собі обидва підходи, що дозволяє, в теорії, надавати більш точні та персоналізовані рекомендації, враховуючи як схожість сюжетів, так і емоційні впливи фільмів. Такий підхід є інноваційним у контексті рекомендаційних систем для кіноспільнот, оскільки він надає новий рівень індивідуалізації та розуміння вподобань користувачів.

Потенційні переваги проекту полягають у наданні нового способу отримувати рекомендації для перегляду. Це може призвести до підвищення відвідуваності та зацікавленості користувачів у веб-сайті або додатку, що в свою чергу може призвести до зростання прибутку через збільшення рекламних показів.

Проект відрізняється від інших систем спроможністю надавати більш точні та персоналізовані рекомендації на основі аналізу сюжетів та емоційного впливу, що може привертати більше користувачів. Аналіз емоційного тону сюжету представляє собою інноваційний підхід у сфері рекомендаційних систем для фільмів, який значно відрізняється від традиційних методів, що базуються на соціальних факторах, таких як відгуки користувачів або оцінки. Унікальність підходу аналізу емоційного тону сюжету полягає у його здатності надавати більш точні, глибинні та персоналізовані рекомендації, що враховують емоційний вплив фільмів, забезпечуючи таким чином покращений користувацький досвід.

Для оцінки повноти та достовірності результатів, необхідно провести аналіз точності рекомендацій, їхню придатність для користувачів та відповідність їхніх вподобань. Також слід оцінити ефективність алгоритмів аналізу сюжетів та емоційного впливу на прикладі використання в реальних умовах.

Впровадження цієї розробки на даному етапі є доволі перспективним, тому що оцінка якості роботи алгоритму рекомендацій залежить від відгуків користувачів, які ним скористувались.

В майбутньому, можливе удосконалення алгоритму, а саме покращення вхідних даних. Оскільки рекомендація відбувається на аналізі сюжету, кращий його опис може послужити збільшенню точності алгоритму.

Висновок

В результаті виконання цього проекту було розроблено та впроваджено веб-застосунок для кіноспільноти, який надає користувачам можливість створювати особисті профілі, додавати відгуки до фільмів та отримувати персоналізовані рекомендації. За допомогою аналізу сюжетів та емоційних відгуків було реалізовано алгоритм рекомендацій, що дозволяє користувачам знаходити фільми, які відповідають їхнім інтересам та настрою.

Цей проект відкриває нові можливості для кіноспільноти, дозволяючи їм знаходити нові фільми та ділитися враженнями з іншими учасниками. Новаторський підхід до рекомендацій, що базується на аналізі сюжетів та емоцій, робить цей веб-застосунок унікальним.

Завдяки розробленому веб-застосунку користувачі можуть не лише знаходити нові фільми для перегляду, а й активно взаємодіяти з іншими учасниками спільноти, обмінюючись враженнями та рекомендаціями. Такий підхід стимулює спільноту до активного обговорення та взаємодії, сприяючи розвитку кіномистецтва та культури в цілому.

У майбутньому можливе розширення функціоналу цього веб-застосунку шляхом додавання нових функцій та покращення алгоритмів рекомендацій. Також важливим є подальший аналіз та оптимізація системи з метою підвищення ефективності та зручності використання для користувачів.

Аналіз емоційного тону демонструє високу ефективність у створенні персоналізованих та релевантних рекомендацій для користувачів. Завдяки врахуванню емоцій, які викликають фільми, система може запропонувати ті, що відповідають поточному емоційному стану чи вподобанням глядача. Це сприяє глибшій емоційній інтеграції користувачів з платформою, підвищує їхнє задоволення та довіру до рекомендаційної системи, а також відкриває нові можливості для відкриття менш відомих, але емоційно резонуючих фільмів. Такий підхід забезпечує комплексне розуміння потреб користувачів і допомагає створити більш цінний та захоплюючий досвід перегляду.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

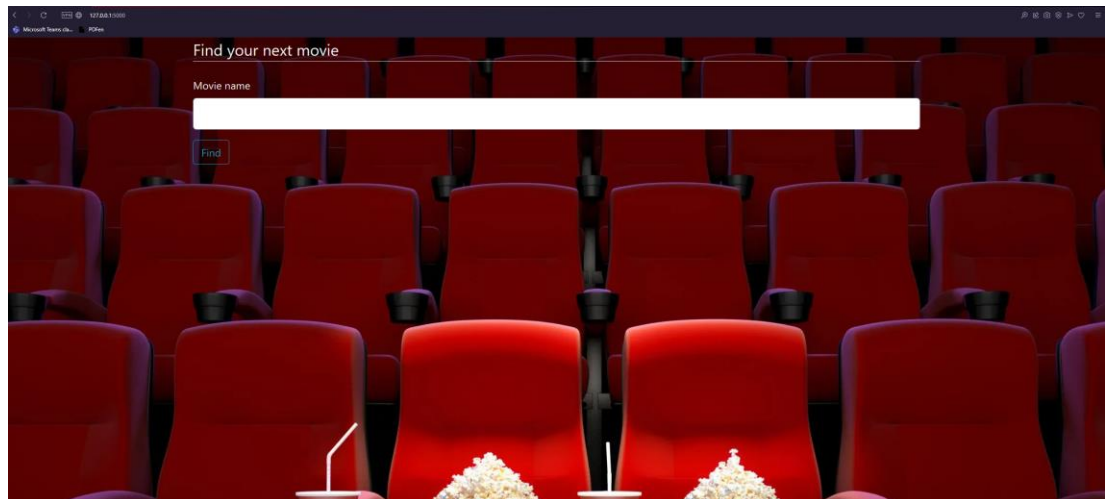
1. Flask Web Development: Developing Web Applications with Python. Miguel Grinberg
2. Recommender Systems: The Textbook. Charu C. Aggarwal. 518 p.
3. Pattern Recognition and Machine Learning. Christopher M. Bishop. 758 p.
4. Machine Learning Yearning. Andrew Ng. 118 p.
5. Sentiment Analysis: Mining Opinions, Sentiments, and Emotions. Bing Liu. 451 p.
6. Content-Based Recommender Systems. Michael J. Pazzani. Daniel Billsus.
7. Recommender Systems Handbook. Francesco Ricci, Lior Rokach, Bracha Shapira. 845 p.
8. Collaborative Filtering Recommender Systems. Michael D. Ekstrand, John T. Riedl, Joseph A. Konstan. 85 p.
9. Introduction to Data Mining. Pang-Ning Tan, Michael Steinbach, Vipin Kumar. 719 p.
10. Natural Language Processing with Python. Steven Bird, Ewan Klein, Edward Loper. 504 p.
11. Deep Learning for Natural Language Processing. Palash Goyal, Sumit Pandey, Karan Jain. 290 p.
12. Pazzani, M.J., Billsus, D. (2007). Content-Based Recommendation Systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. Lecture Notes in Computer Science, vol 4321. Springer, Berlin, Heidelberg. pp 325–341
13. Hybrid Recommender Systems: Survey and Experiments. Robin Burk. 29 p.
14. IMDb. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Internet_Movie_Database (дата звернення: 24.05.2024).
15. Press Room IMDb. <https://web.archive.org/web/20180527134839/https://www.imdb.com/pressroom/stats/> (дата звернення: 22.05.2024).
16. Letterboxd. *Вікіпедія*. URL: <https://uk.wikipedia.org/wiki/Letterboxd> (дата звернення: 22.05.2024).

ДОДАТКИ

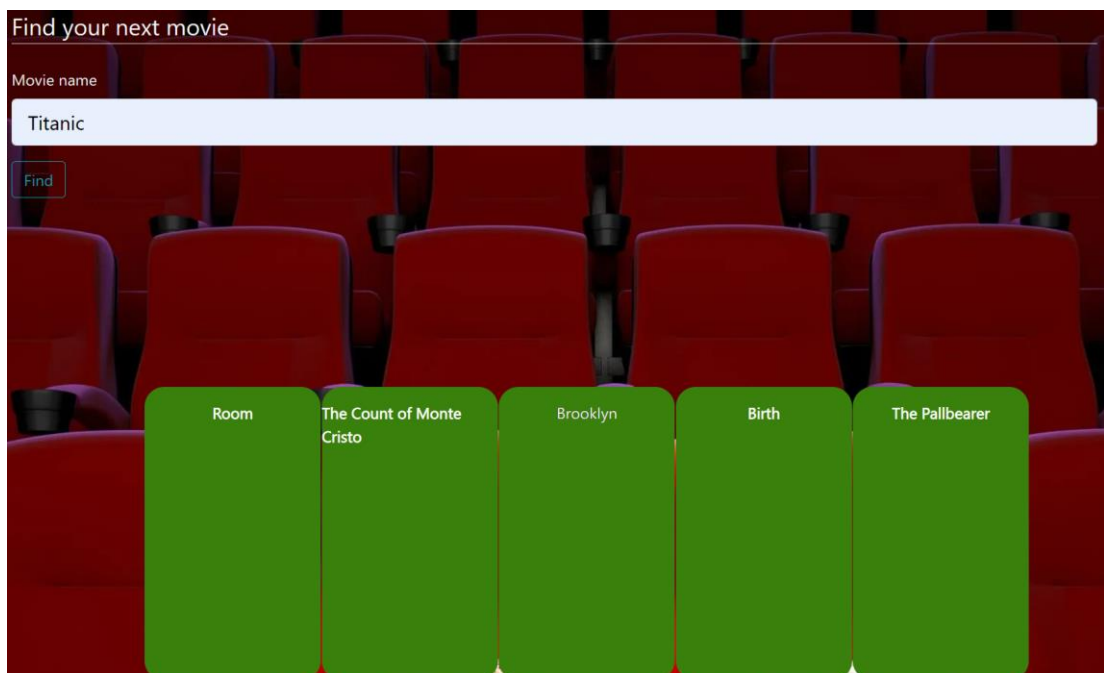
Додаток А

Отримання рекомендацій:

На головній сторінці знаходяться форма для вводу назви фільму та кнопка підтвердження



Після вводу назви фільму та підтвердження дії, видається рекомендацій фільмів:



Посилання веде на сторінку сайту з рекомендованим фільмом:

TMDB

MoviesTV ShowsPeopleMore

+ENLoginJoin TMDB

OverviewMediaFandomShare



Room (2015)

R 2015-10-16 (US) • Drama, Thriller • 1h 58m

80% User Score

What's your Vibe?

Grid

Heart

Bookmark

▶ Play Trailer

Love knows no boundaries

Overview

Held captive for 7 years in an enclosed space, a woman and her young son finally gain their freedom, allowing the boy to experience the outside world for the first time.

Emma Donoghue

Novel, Screenplay

Lenny Abrahamson

Director

Top Billed Cast



Brie Larson

Ma



Jacob Tremblay

Jack



Joan Allen

Nancy



Sean Bridgers

Old Nick



Tom McCamus

Leo



William H. Macy

Robert



Cas Anvar

Dr. Mittal

Facebook

Twitter

Status

Released

Original Language

English

Budget

\$13,000,000.00

Revenue

\$35,401,758.00

Приклад SQL запиту

```
class UserDatabase:
    #user_creation
    def __init__(self, db_name):
        self.conn = sqlite3.connect(db_name, check_same_thread=False)
        self.cursor = self.conn.cursor()
```

```
2 usages
def add_user(self, username, email, password):
    user = (str(username), str(email), str(password))
    self.cursor.execute("INSERT INTO users (username, email, password, reviews) VALUES (?, ?, ?, Null)", user)
    self.conn.commit()
```

```
2 usages
def get_user(self, email):
    self.cursor.execute("SELECT * FROM users WHERE email=?", (email,))
    user = self.cursor.fetchone()
    return user
```

```
1 from DatabaseManager import UserDatabase
2
3 db = UserDatabase("users.db")
4
5 db.add_user("sql_test", "testmail@gmail.com", "123123123")
6
7 print(db.get_user("testmail@gmail.com"))
8
```

sql_requests x

```

:
C:\Users\user\PycharmProjects\movieRecommender\venv\Scripts\python.exe
(7, 'sql_test', 'testmail@gmail.com', '123123123', None)

Process finished with exit code 0
```

Для прикладу, було обрано процес створення нового користувача в базі даних. Кожна представлена функція належить класу UserDatabase, який на етапі ініціалізації, за допомогою команди connect бібліотеки sqlite3, під'єднується до бази даних користувачів. Далі, функція add_user імітує sql-запит до бази даних який вставляє в таблицю отримані дані про користувача. Для перевірки успішності, використовується функція get_user, яка, використовуючи електронну пошту, може знайти користувача в базі даних.

Як видно з результатів виконання запитів, програма успішно записує та зчитує дані з бази даних.

Програмні коди:

Імпорт необхідних бібліотек

```
[1]: import pandas as pd
import numpy as np
```

```
[2]: import re
```

```
[3]: from nrclex import NRCLex
```

Зчитування даних з первинних джерел інформації

```
[4]: credits = pd.read_csv("credits.csv")
movies = pd.read_csv("movies.csv")
```

```
[5]: credits
```

```
[5]:
```

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...
...	...	...	...	...
4798	9367	El Mariachi	[{"cast_id": 1, "character": "El Mariachi", "c...	[{"credit_id": "52fe44eec3a36847f80b280b", "de...
4799	72766	Newlyweds	[{"cast_id": 1, "character": "Buzzy", "credit_...	[{"credit_id": "52fe487dc3a368484e0fb013", "de...
4800	231617	Signed, Sealed, Delivered	[{"cast_id": 8, "character": "Oliver O\u2019To...	[{"credit_id": "52fe4df3c3a36847f8275ecf", "de...
4801	126186	Shanghai Calling	[{"cast_id": 3, "character": "Sam", "credit_id...	[{"credit_id": "52fe4ad9c3a368484e16a36b", "de...
4802	25975	My Date with Drew	[{"cast_id": 3, "character": "Herself", "credi...	[{"credit_id": "58ce021b9251415a390165d9", "de...

4803 rows × 4 columns

```
[ ]: movies
```

Поєднання зчитаних файлів

```
[7]: movies = movies.merge(credits, on="title")
```

Виокремлення виключно необхідний стовпців даних

```
[9]: movies = movies[['movie_id', 'title', 'overview', 'genres', 'keywords', 'cast', 'crew']]
```

```
[11]: movies = movies.dropna()
```

Отримання буквальних значень

```
[13]: import ast
```

```
•[14]: def convert(obj):
result = []
for i in ast.literal_eval(obj):
result.append(i['name'])
return result
```

```
[ ]: movies['genres'] = movies['genres'].apply(convert)
movies['keywords'] = movies['keywords'].apply(convert)
movies.head
```

```
•[16]: def convert3(obj):
result = []
counter = 0
for i in ast.literal_eval(obj):
if counter == 3:
break
result.append(i['name'])
counter += 1
return result
```

```
[ ]: movies['cast'] = movies['cast'].apply(convert3)
```

```
•[18]: def director(obj):
result = []
for i in ast.literal_eval(obj):
if i['job'] == 'Director':
result.append(i['name'])
return result
```

```
[ ]: movies['crew'] = movies['crew'].apply(director)
movies.head
```

Продовження Додатку В

Видалення роздільних символів

```
[ ]: movies['overview'] = movies['overview'].apply(lambda x:x.split())

21: movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ", "") for i in x])
    movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ", "") for i in x])
    movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ", "") for i in x])
    movies['crew'] = movies['crew'].apply(lambda x:[i.replace(" ", "") for i in x])

[ ]: for i in movies.columns:
    movies[i] = movies[i].apply(lambda x:[i.replace(" ", "") for i in x])
```

Формування тегів фільму

```
[23]: movies['tags'] = movies['overview']+movies['genres']+movies['keywords']+movies['cast']+movies['crew']
    movies['tags']
```

```
[23]: 0      [In, the, 22nd, century,, a, paraplegic, Marin...
      1      [Captain, Barbossa,, long, believed, to, be, d...
      2      [A, cryptic, message, from, Bond's, past, send...
      3      [Following, the, death, of, District, Attorney...
      4      [John, Carter, is, a, war-weary,, former, mili...
      ...
      4803     [El, Mariachi, just, wants, to, play, his, gui...
      4804     [A, newlywed, couple's, honeymoon, is, upended...
      4805     [Signed,, Sealed,, Delivered", introduces, a,...
      4806     [When, ambitious, New, York, attorney, Sam, is...
      4807     [Ever, since, the, second, grade, when, he, fi...
      Name: tags, Length: 4805, dtype: object
```

Створення датафрейму для алгоритму рекомендації

```
[26]: df = movies[['movie_id', 'title', 'tags']]
    df.head
```

```
[26]: <bound method NDFrame.head of      movie_id      title \
0      19995      Avatar
1      285      Pirates of the Caribbean: At World's End
2      206647      Spectre
3      49026      The Dark Knight Rises
4      49529      John Carter
...      ...
4803      9367      El Mariachi
4804      72766      Newlyweds
4805      231617      Signed, Sealed, Delivered
4806      126186      Shanghai Calling
4807      25975      My Date with Drew

      tags
0      [In, the, 22nd, century,, a, paraplegic, Marin...
1      [Captain, Barbossa,, long, believed, to, be, d...
2      [A, cryptic, message, from, Bond's, past, send...
3      [Following, the, death, of, District, Attorney...
```

```
[ ]: df['tags'] = df['tags'].apply(lambda x: ' '.join(x))
    df.head
```

```
[ ]: df['tags'] = df['tags'].apply(lambda x:x.lower())
    df.head
```

Імпорт необхідних бібліотек та перетворення даних на вектори

```
[29]: from sklearn.feature_extraction.text import CountVectorizer

[30]: cv = CountVectorizer(max_features=5000, stop_words='english')

[31]: cv.fit_transform(df['tags']).toarray().shape

[31]: (4805, 5000)

[32]: vectors = cv.fit_transform(df['tags']).toarray()

[33]: vectors = cv.fit_transform(df['tags']).toarray()

[34]: vectors[0]
```

```
[34]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

Імпорт необхідних бібліотек та отримання коренів слів для подальшої обробки

```
[35]: import nltk

[36]: from nltk.stem.porter import PorterStemmer

[37]: ps = PorterStemmer()

[38]: def stem(text):
    y = []
    for i in text.split():
        y.append(ps.stem(i))
    return ' '.join(y)

[ ]: df['tags'] = df['tags'].apply(stem)
```

Продовження Додатку В

Створення матриці косинусного порівняння

```
[40]: from sklearn.metrics.pairwise import cosine_similarity
[41]: similarity = cosine_similarity(vectors)
```

Створення функції для обчислення емоцій даного тексту

```
*[46]: def mood(tags):
        return(NRCLex(tags).raw_emotion_scores)#пошук відповідної емоції до кожного слова тексту
```

Функція додавання емоційного профілю до датафрейму на основі сюжету фільму

```
*[49]: def add_emotions(title, row):
        result = []
        emote = mood(df.loc[df['title'] == title].values[0][2])#застосування функції отримання емоцій
        total_sum = sum(emote.values())#розрахунок загальної суми емоцій
        data_percentages = {emotion: (score / total_sum) * 100 for emotion, score in emote.items()}#нормалізація емоцій в формат відсотків
        sorted_data_percentages = sorted(data_percentages.items(), key=lambda x: x[1], reverse=True)#сортування емоцій
        for i in sorted_data_percentages:
            if len(result) != 3:#обмеження в 3 емоції
                result.append(i[0])#створення списку емоцій
        try:
            df.loc[row, 'emotions'] = ' '.join(result)#додавання емоцій до відповідних фільмів
        except:
            pass
```

Функція рекомендації фільмів:

```
def recommend_movie(movie, df, similarity):
    result = []
    movie_index = df[df['title']==movie].index[0]#отримання індексу фільму в матриці за назвою
    distances = similarity[movie_index] # розрахунок "відстаней" до інших фільмів, тобто їх схожість
    movie_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x:x[1])
    orig_list = ast.literal_eval(df.iloc[movie_index].emotions)
    orig_list.sort()
    for i in movie_list:
        try:
            liste = ast.literal_eval(df.iloc[i[0]].emotions)#отримання списку емоцій початкового фільму
            liste.sort()#сортування результатів за відповідними емоціями
            if orig_list == liste:
                result.append(df.iloc[i[0]].title) #якщо емоції співпадають, фільм додається до масиву рекомендацій
        except:
            pass
    return result[1:6]#повернення списку з п'яти фільмів
```

Имя пользователя:
Інформаційних систем в економіці Шкуратовська Те...

ID проверки:
1016368671

Дата проверки:
17.06.2024 16:24:06 EEST

Тип проверки:
Doc vs Internet + Library

Дата отчета:
17.06.2024 17:32:39 EEST

ID пользователя:
100005745

Название файла: Романенко М_IH-401

Количество страниц: 52 Количество слов: 9078 Количество символов: 73078 Размер файла: 5.08 MB ID файла: 10161753

18.9% Совпадения

Наибольшее совпадение: 5.56% с Интернет-источником (https://www.limsforum.com/knowledge-base-wiki/?rdp_we_res).

16% Источники из Интернета

379

Страница 54

5.39% Источники из Библиотеки

337

Страница 56

0% Цитат

Исключение цитат выключено

Исключение списка библиографических ссылок выключено

0% Исключений

Нет исключенных источников

Название файла: Романенко М_ІН-401

ID файла: 1016175388

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАЛИМА ГЕТЬМАНА**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»

спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЕКТ

на тему:

ІНФОРМАЦІЙНА СИСТЕМА ДЛЯ КІНОСПІЛЬНОСТІ

здобувача Романенка Максима Сергійовича _____

Науковий керівник: д.с.н., проф.

Ріппа С.П.

**Випускний бакалаврський проект
допущений до захисту в Екзаменаційній
комісії з атестації здобувачів вищої
ОСВІТИ**

Завідувач кафедри: к.е.н., проф.

Тішков Б.О.

Київ - 2024