КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики
Кафедра інтелектуальних програмних систем

Кваліфікаційна робота

на здобуття рівня бакалавра

за спеціальністю 121 Інженерія програмного забезпечення

на тему:

РОЗРОБКА ВЕБ-ОРІЄНТОВАНОЇ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ ДЛЯ ПОШУКУ СХОЖОЇ МУЗИКИ

Виконав студент 4-го курсу
Євгеній ВОРОБЙОВ _____
(підпис)

Науковий керівник:
доцент, кандидат фіз.-мат. наук
Лариса КАТЕРИНИЧ _____
(підпис)

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Роботу розглянуто й допущено до захисту на засіданні кафедри інтелектуальних програмних систем
« 29 » травня 2023 р.,
протокол № 11
Завідувач кафедри
Олександр ПРОВОТАР _____
(підпис)

Київ – 2022

РЕФЕРАТ

Обсяг роботи 41 сторінка, 9 ілюстрацій, 1 таблиця, 18 джерел посилань.

РЕКОМЕДАЦІЙНІ СИСТЕМИ, МАШИНЕ НАВЧАННЯ, ПЕРСОНАЛІЗОВАНІ РЕКОМЕНДАЦІЇ, ІНТЕРФЕЙС КОРИСТУВАЧА, ВЕБ ДОДАТОК, МОВА PYTHON.

Об’єктом роботи є використання та ефективне застосування API стрімінгових сервісів та інших допоміжних засобів для розробки веб додатку. Предметом роботи є веб додаток, що реалізовує в собі зручний пошук та рекомендації нових пісень.

Метою роботи є аналіз існуючих аналогів вирішення цієї проблеми, ознайомлення з розробкою методів реалізації персоналізованих рекомендацій на різних платформах та створення на основі аналізу веб застосунку «Рекомендація музики» для пошуку нових пісень, якими може зацікавиться споживач, що може набути широкого застосування серед користувачів що полюбляють музику.

Методи розроблення: вільно поширюване інтегроване середовище розробки WebStorm та PyCharm, мова програмування Python, середовище розробки Jupyter Notebook від Google Colaboratory, бібліотека Vue.js для мови програмування JavaScript, сервіс для API запитів від Spotify.

Результати роботи: виконано загальний огляд методів створення системи рекомендації, проаналізовано переваги та недоліки використання цих засобів під час розробки, розроблено мобільний додаток «Рекомендація музики», який у подальшому може набути практичного характеру.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

API - Application Programming Interface, прикладний програмний інтерфейс;

CF - Collaborative filtering, колаборативне фільтрування;

HTML – Hyper Text Markup Language, мова розмітки гіпертексту;

IT - Інформаційні технології

JSON – JavaScript Object Notation;

ОС - Операційна система;

RS - Recommender Systems, рекомендаційні системи є

UI - User Interface;

БД - База даних;

МН - машинне навчання;

ВСТУП

Оцінка сучасного стану об’єкта розробки. Підбір персоналізованих рекомендацій є активно розвиваючимся напрямком у галузі інформаційних технологій і машинного навчання. Він зосереджений на розробці алгоритмів та систем, які можуть аналізувати, прогнозувати та рекомендувати вміст, товари, послуги або дії користувачам з урахуванням їхніх індивідуальних потреб, вподобань та контексту.

Сучасні системи персоналізованих рекомендацій використовують широкий спектр технік, включаючи колаборативне фільтрування, фільтрування контенту, глибинне навчання, рекомендації засновані на контексті, змішані підходи та багато інших. Вони опираються на обробку великого обсягу даних про користувачів, вміст та контекст, щоб надавати рекомендації, які максимально задовольняють потреби кожного окремого користувача.

Актуальність роботи та підстави для її виконання. Існує кілька ключових підстав для дослідження персоналізованих рекомендацій:

1. Зростання обсягу даних: Завдяки постійному збільшенню кількості доступної інформації і користувацьких даних, виникає потреба у вдосконаленні алгоритмів рекомендацій для ефективнішої обробки цих даних та надання користувачам доречного вмісту.
2. Потреба у персоналізації: Користувачі сьогодні вимагають персоналізованого досвіду в різних сферах свого життя, включаючи рекомендації. Ефективні системи персоналізованих рекомендацій можуть значно покращити задоволення користувачів та збільшити залученість до вмісту.
3. Розвиток нових технологій: Постійний розвиток машинного навчання, глибинного навчання та інших технологій дозволяє вдосконалювати алгоритми рекомендацій та створювати більш точні та ефективні системи.
4. Ефективні системи персоналізованих рекомендацій також мають великий потенціал для бізнесу. Вони можуть сприяти збільшенню продажів,

покращенню взаємодії з клієнтами та виявленню нових можливостей у сфері маркетингу та реклами.

Однак, існують деякі виклики, з якими стикаються дослідники та практики у цій галузі:

1. Проблема холодного старту: Якщо про користувача або предмет рекомендації відомо дуже мало, складно зробити адекватну персоналізовану рекомендацію.
2. Проблема розрідженості: З великою кількістю користувачів та предметів, важко підібрати рекомендації, які б задовольняли кожного користувача.
3. Проблема інформаційного перевантаження: У сучасному світі користувачі зіткнулися з величезним обсягом інформації, тому рекомендаційні системи повинні знаходити баланс між наданням релевантного контенту та уникненням перевантаження користувачів надмірною кількістю рекомендацій.
4. Проблема забезпечення конфіденційності та приватності: Оскільки для надання персоналізованих рекомендацій необхідно зібрати та аналізувати велику кількість персональних даних, необхідно забезпечити їхню конфіденційність та приватність.

Вирішення цих проблем може призвести до підвищення рівня задоволеності користувачів, збільшення залученості та покращення ефективності бізнесу.

Мета й завдання роботи. Метою роботи є розробка веб додатку, який був б корисним для користувачів та заохочував б їх до ознайомлення з іншими музичними композиціями. Для досягнення цієї мети поставлено такі завдання:

- розглянути існуючі додатки, що вирішують цю проблему, визначити їх перевагу та недоліки;
- розглянути існуючі версії реалізації персоналізованих рекомендацій та визначити перевагу та недоліки цих рішеннь;
- створити систему рекомендацій;

- обрати архітектуру веб додатку для створення інтерфейсу користувача;
- спроектувати, розробити додаток та описати процес його створення.

Об'єкт, методи й засоби розроблення. В якості інструментів реалізації програмного засобу було обрано PyCharm та WebStorm – інтегровані середовища розробки (IDE) для мови програмування Python та JavaScript, відповідно, що надає ліцензований доступ до своєї продукції для студентів вищих навчальних закладів всього світу. Також було скориставшись Google Colaboratory - безкоштовним хмарним середовищем для роботи з блокнотами Jupyter, було прискорено та спрощено процес тренування та створення рекомендаційної моделі.

Реалізація здійснена мовою Python, з використанням бібліотеки для машинного навчання Scikit-learn — для розробки алгоритму машинного навчання; програмне забезпечення, що є розширенням мови Python, NumPy — для зручності підрахунків; бібліотека високоякісних наукових інструментів для мови Python SciPy — для використання традиційних числових алгоритмів; бібліотека для візуалізації даних двовимірною 2D графікою matplotlib.

Можливі сфери застосування. Веб додаток «Рекомендація музики» може мати широке практичне застосування серед великої кількості поціновувачів музики різних жанрів.

РОЗДІЛ 1. АНАЛІЗ РЕКОМЕНДАЦІЙНИХ СИСТЕМ

1.1. Головна задача рекмонедаційної системи

Рекомендаційні системи (РС) - це програмні інструменти та методи, що надають пропозиції щодо предметів, які можуть бути корисними для користувача. Пропозиції стосуються різних процесів прийняття рішень, наприклад, які товари купувати, яку музику слухати або які новини читати в Інтернеті. "Item" - це загальний термін, який використовується для позначення того, що система рекомендує користувачам [1].

РС зазвичай фокусується на конкретному типі товару (наприклад, компакт-диски або новини) і основна задача системи, що використовується для генерування рекомендацій, надавати корисні та ефективні пропозиції для цього конкретного типу товару. РС в першу чергу орієнтовані на людей, які не мають достатнього особистого досвіду або компетенції, щоб оцінити потенційно величезну кількість альтернативних товарів, які, наприклад, може запропонувати вебсайт. Прикладом може слугувати система книжкових рекомендацій, яка допомагає користувачам вибрати книгу для читання. На популярному веб-сайті Amazon.com використовується РС для персоналізації інтернет-магазину для кожного клієнта. Оскільки рекомендації зазвичай персоналізовані, різні користувачі або групи користувачів отримують різноманітні пропозиції.

Також існують неперсоналізовані рекомендації. Їх набагато простіше створювати, і зазвичай вони публікуються в журналах або газетах. Типовими прикладами є десятка найкращих книжок, компакт-дисків тощо. Хоча вони можуть бути корисними та ефективними в певних ситуаціях, ці типи неперсоналізованих рекомендацій, як правило, не розглядаються в дослідженнях РС [2].

У найпростішій формі персоналізовані рекомендації пропонуються у вигляді списків. РС намагаються передбачити, які продукти чи послуги є

найбільш підходящими, виходячи з уподобань та обмежень користувача. Для того, щоб виконати таке обчислювальне завдання, РС збирає від користувачів їхні вподобання, які або явно виражені, наприклад, у вигляді оцінок продуктів, або виводяться шляхом інтерпретації дій користувача.

Наприклад, РС може розглядати перехід на сторінку певного продукту як неявну ознаку вподобань щодо товарів, представлених на цій сторінці. Розробка РС була ініційована досить простим спостереженням: люди часто покладаються на рекомендації, надані іншими, при прийнятті рутинних, повсякденних рішень. Наприклад, при виборі книжки для читання зазвичай покладаються на те, що рекомендують колеги; роботодавці покладаються на рекомендаційні листи при прийнятті рішень про найм на роботу; а при виборі фільму для перегляду люди схильні читати і покладатися на рецензії на фільми, написані кінокритиками, які з'являються в газетах чи інтернет блогах, які вони читають.

Вибуховий ріст і різноманітність інформації, доступної в Інтернеті, а також швидке впровадження нових послуг електронного бізнесу (купівля товарів, порівняння товарів, аукціони тощо) часто путали користувачів, що призводило до прийняття ними неправильних рішень [3]. Наявність вибору, замість того, щоб просто отримати вигоду, почала знижувати задоволеність користувачів. Було зрозуміло, що хоч вибір - це і добре, але величезний вибір - не завжди краще. Останні роки РС виявилися цінним засобом для подолання проблеми інформаційного перевантаження. Зрештою, РС вирішує цю проблему, вказуючи користувачеві на нові, ще не знайомі йому об'єкти, які можуть мати відношення до поточної задачі користувача.

На запит користувача, який може бути сформульований, залежно від рекомендаційного підходу, контекстом і потребами користувача, РС генерують рекомендації, використовуючи різні типи знань і даних про користувачів, доступні товари і попередні транзакції, що зберігаються в спеціальних базах даних. Користувач може переглянути рекомендації. Він може прийняти їх або ні, і може негайно або на наступному етапі надати явний або неявний зворотний

зв'язок. Всі ці дії користувача і його відгуки можуть зберігатися в базі даних рекомендацій і використовуватися для генерації нових рекомендацій при наступних взаємодіях користувача з системою.

1.2 Рекомендації музики

Системи рекомендацій все більше і більше використовуються в багатьох сферах: готелі, подорожі, товари. Але музична сфера має деякі особливості, які слід враховувати до уваги. Перший фактор, який слід враховувати - це **тривалість музичного треку**. Оскільки трек короткий, менш критично дати погану рекомендацію, ніж у випадку з фільмом, книгою чи книгою. Користувач також може швидко прослухати музику, щоб швидко зрозуміти, чи відповідає вона його смаку, чи ні. Другою особливістю є **кількість доступних треків**, дійсно, вибір дуже широкий, за оцінками, щонайменше десятки мільйонів пісень доступні в Інтернеті. У той час як для поїздок або фільмів користувач шукає різноманітності, він може любити слухати ту саму музику знову і знову. Більше того, можливо, що під час першого прослуховування користувач не був уважним, оскільки прослуховування музики часто відбувається паралельно з іншою діяльністю (спортом, роботою...). Уважне прослуховування вимагає
якісної апаратури, відповідного настрою та уваги. Крім того, з одного музичного твору досить легко виокремити набір характеристик. Інформація може бути вилучена через обробку сигналу, завдяки музичним
знанням, текстам пісень або просто за допомогою зворотного зв'язку.

Стара музика так само актуальна, як і нова: нещодавня музика, а також музика, написана кілька десятиліть тому, або класична музика можуть бути однаково приємними. Задача в тому, щоб правильно зрозуміти смаки користувача. Також потрібно враховувати, що прослуховування музики часто є пасивним: слухач не обов'язково слухає її уважно: в магазинах, барах, під час роботи...

Останній момент, який відрізняє музику від інших предметів, які можна рекомендувати, - це те, що музику часто відтворюють у певній послідовності. Оскільки пісні короткі, їх часто об'єднують у вигляді плейлиста.

Беручи до уваги особливості музики, тепер необхідно скласти адекватну рекомендацію. Звичайно, головна мета - досягти хорошого рівня точності, тобто передбачити музику, яка сподобається користувачеві і яку він буде слухати. Чим більше користувач довіряє системі рекомендацій і знає, як вона працює, тим ефективнішою вона буде.

Успішна рекомендація передбачає компроміс між експлуатацією та дослідженням: З одного боку, експлуатація полягає у відтворенні безпечної музики, музики, яка, як відомо рекомендатору, подобається користувачеві. Це називається "звичним досвідом", і він приносить короткострокову вигоду. З іншого боку, дослідження - це відтворення нової музики, нові відкриття. Це називається "зануренням", і воно приносить довгострокову користь. Якщо його правильно оцінити, може статися так, що з'явиться невеличка несподіванка. Це означає, що нам потрібно знайти відповідний баланс між новизною і знайомістю, різноманітністю і схожістю, а також популярністю і персоналізацією.

Відповідна рекомендація повинна також відображати контекст слухача. Вона не може грунтуватися лише на музиці та властивостях слухача, потрібно брати до уваги настрій, активність тощо. Наприклад, людина, яка працює, не захоче слухати ту саму музику, що й під час бігу. Нарешті, важливим моментом є прозорість з користувачами. Доведено, що пояснення того, як працює алгоритм, підвищує довіру користувача, а отже, і час, який він проведе на платформі, по-перше, щоб вдосконалити свій профіль, а по-друге, тому що він отримає кращі рекомендації.

1.3 Аналіз існуючих типів РС

Розглянемо три основні типи рекомендаційних систем - Рекомендації на основі популярності (Popularity-based recommendations); Рекомендації на основі колаборативного фільтрування (Collaborative Filtering); Рекомендації на основі контенту (Content-based recommendations). Для кожного з них розберемо, як він працює, та подивимось на сильні та слабкі сторони і приклади використання. Перш ніж заглиблюватися в деталі конкретних алгоритмів треба розуміти основні відмінності між ними:

- Рекомендації на основі популярності (Popularity-based recommendations):
    - Цей тип системи рекомендує користувачам найпопулярніші або найбільш популярні елементи. Рекомендації засновані на загальній популярності або популярності серед певної групи користувачів [3].
    - Переваги: Простота в реалізації, відносна легкість у використанні.
    - Недоліки: Неперсоналізовані рекомендації, не враховують індивідуальні смаки та уподобання користувачів. Можуть бути непридатними для специфічних або нішевих ринків.
    - Приклади використання: Список популярних товарів на основних сторінках електронних магазинів, таких як Amazon або Netflix.
- Рекомендації на основі колаборативного фільтрування (Collaborative Filtering):
    - Цей тип системи заснований на зіставленні вподобань користувачів із схожими історіями. Інформація про попередні вибори користувачів використовується для рекомендацій нових елементів [4].
    - Переваги: Персоналізовані рекомендації, здатні враховувати смаки користувачів та розпізнавати схожість між ними. Працюють добре для нових користувачів, оскільки не вимагають великого обсягу особистої інформації.

    - Недоліки: Проблеми з холодним стартом (cold-start problem), коли немає достатньо даних про нових користувачів або елементи. Також можуть виникати проблеми зі значною кількістю користувачів або елементів, що призводить до розрідженості матриці взаємодії.
    - Приклади використання: Рекомендації фільмів на платформі Netflix, товарів на Amazon, друзів в соціальних мережах.
- Рекомендації на основі контенту (Content-based recommendations):
    - Використовує властивості або характеристики елементів для рекомендацій, заснованих на схожості контенту. Інформація про вподобання користувачів може використовуватися для врахування особистих уподобань [5].
    - Переваги: Персоналізовані рекомендації, здатні враховувати індивідуальні смаки користувачів та враховувати особистий контекст. Можуть бути корисними для вузькоспеціалізованих ринків або товарів.
    - Недоліки: Обмежена можливість виявлення нових елементів або унікальних рекомендацій, якщо вони суттєво відрізняються від існуючих контентів. Залежність від точності або якості описів контенту.
    - Приклади використання: Рекомендації новинних статей на платформі Google News, пісні на музичних стрімінгових сервісах, книг на сайтах книгарень.

Далі розглянемо деталі реалізації кожної:

1. Рекомендаційна система на основі популярності. Оскільки вона є доволі простою в реалізації вона не потребує складних алгоритмів аналізу даних або особистої інформації про користувачів, тож і реалізація не вимагає багато кроків для створення.

- Необхідно зібрати дані про популярність елементів. Це може бути кількість переглядів, продажів, оцінок або інші метрики, що відображають інтерес користувачів до елементів [6].
- На основі зібраних даних елементи сортуються в порядку спадання популярності. Це може включати створення рейтингу або рангу для кожного елемента.
- Залежно від контексту рекомендаційної системи може бути застосовано додаткові фільтри для вибору певної групи елементів або підгрупи користувачів, на основі якої буде здійснюватися рекомендація.
- Найпопулярніші елементи можуть бути представлені у вигляді списку на головній сторінці або на сторінці з рекомендаціями для користувача.

2. Рекомендації на основі колаборативного фільтрування. Реалізація вимагає аналізу та використання відносностей та подібностей між користувачами та їхніми виборами. Основні кроки для реалізації такої системи включають:
    - Збір даних: Необхідно постійно збирати дані про вибори користувачів, такі як оцінки, рейтинги, історія покупок або переглядів. Ці дані використовуються для побудови матриці взаємодії, де рядки відповідають користувачам, а стовпці - елементам.

	Item 1	Item 2	Item 3	Item 4	Item 5
User 1	0,88	0,29	0,43	0,08	
User 2		0,01	0,37		0,33
User 3	0,32			0,22	0,88
User 4		0,16	0,48		0,19
User 5	0,87	0,87		0,03	0,62

Рисунок 1.1 – приклад матриці взаємодії.

- Вимірювання подібності: На основі зібраних даних розраховуються міри подібності між користувачами або між елементами. Зазвичай використовуються метрики, такі як косинусна схожість, Евклідова відстань або кореляція Пірсона.
- Знаходження сусідів: На основі вимірювання подібності обираються найбільш схожі користувачі або елементи, які стануть сусідами. Це може бути обмеженням за фіксованою кількістю сусідів або за деяким пороговим значенням подібності.
- Прогнозування рейтингів: За допомогою інформації про сусідів розраховуються прогнози рейтингів для елементів, які ще не взаємодіяли з конкретним користувачем. Це може бути зроблено, наприклад, шляхом взваженого середнього рейтингу або застосування алгоритму рекомендації, такого як Singular Value Decomposition (SVD) або Matrix Factorization.
- Представлення рекомендацій: Найвищі прогнози рейтингів використовуються для ранжування та відображення конкретних рекомендованих елементів для користувача.

3. Реалізація рекомендаційної системи на основі контенту передбачає аналіз та використання характеристик або властивостей елементів для здійснення рекомендацій, заснованих на їх схожості до вже вподобаних або

переглянутих користувачами елементів [7]. Нижче наведено кроки, необхідні для реалізації такої системи:

- Збір даних: Необхідно зібрати дані про елементи та їх характеристики. Це можуть бути описи, ключові слова, категорії, автори, жанри або будь-які інші властивості, які можуть бути корисними для визначення схожості між елементами.
- Векторизація контенту: Зібрані дані про елементи потрібно перетворити на числові вектори, що можуть бути використані для обчислення схожості. Це можна зробити за допомогою моделей векторизації слів, наприклад “Bag-of-Words” або “Word2Vec”.
- Розрахунок схожості: На основі числових векторів елементів можна обчислити міру схожості між ними. Це може бути зроблено, наприклад, за допомогою “косинусу подібності” або “Евклідової метрики” між векторами.
- Відбір рекомендацій: На основі розрахованої схожості вибираються найбільш схожі елементи до тих, які сподобалися або були взаємодії з користувачем. Ці елементи стають рекомендаціями для користувача.
- Представлення рекомендацій: Рекомендовані елементи можуть бути відображені у вигляді списку або каруселі на сторінці з рекомендаціями для користувача.

РОЗДІЛ 2. СТРІМІНГОВІ СЕРВІСИ

2.1 Аналіз стрімінгових сервісів

Музичні стрімінгові сервіси надають користувачам можливість насолоджуватися необмеженим доступом до величезної бібліотеки музики, безпосередньо з їхніх пристроїв, незалежно від місця перебування. Функції цих сервісів включають:

- Підписка та особистий профіль: Користувачі можуть створювати власні облікові записи і підписуватися на платні абонементи для доступу до розширених функцій. Кожен користувач має власний профіль, де він може зберігати улюблену музику, створювати плейлисти та ділитися ними з іншими.
- Пошук та рекомендації: Стрімінгові сервіси надають потужні інструменти пошуку, які дозволяють користувачам знаходити музику за жанром, виконавцем, альбомом, піснею тощо. А також використовують алгоритми зазначенні вище для надання персоналізованих рекомендацій, враховуючи музичні вподобання користувача та його попередні прослуховування.
- Плейлисти та радіостанції: Користувачі можуть створювати власні плейлисти, додавати улюблені пісні та альбоми, редагувати їх і ділитися з друзями. Більшість сервісів також пропонує готові плейлисти, засновані на різних настроях, жанрах або подіях. Крім того, користувачі можуть слухати онлайн-радіостанції, засновані на певних жанрах, виконавцях або настроях.
- Офлайн-режим: Багато музичних стрімінгових сервісів дозволяють користувачам завантажувати музику для прослуховування в офлайн-режимі. Це особливо корисно для людей, які часто подорожують або знаходяться у місцях з обмеженим інтернет-з'єднанням.

- Соціальна взаємодія: Користувачі можуть спілкуватися з друзями через сервіс, ділитися плейлистами та піснями, залишати коментарі та відгуки про виконавців та альбоми. Деякі сервіси також надають можливість спільного прослуховування, де користувачі можуть одночасно слухати музику зі своїми друзями та обговорювати її в реальному часі.
- Висока якість звуку: Багато стрімінгових сервісів пропонують підтримку високої якості звуку, такої як “FLAC” або “Hi-Res Audio”. Це дозволяє користувачам насолоджуватися музикою з кращою чіткістю та деталізацією звучання.
- Інтеграція з іншими платформами: Багато стрімінгових сервісів інтегруються з іншими платформами, такими як соціальні мережі, медіацентри, автомобільні системи тощо. Це дає змогу користувачам зручно слухати музику на різних пристроях та платформах.

Це лише загальний опис функцій, які можна зустріти в багатьох музичних стрімінгових сервісах. Приклад популярних сервісів і чим вони виділяються на фоні інших:

- Spotify: Spotify є одним з найпопулярніших музичних стрімінгових сервісів у світі. Він пропонує величезну бібліотеку музики з різних жанрів та виконавців. Одним з головних переваг Spotify є його, як прийнято вважати - найкраща система рекомендацій.
- Apple Music: Apple Music є конкурентом Spotify та володіє великою аудиторією, особливо серед користувачів Apple. Вона пропонує доступ до великої бібліотеки музики, включаючи ексклюзивні альбоми та виступи, яких немає на інших платформах. Apple Music також відрізняється своїми високоякісними музичними кліпами, а також радіостанцією Beats-1, де ведуться ексклюзивні інтерв'ю та музичні програми. Інтеграція з екосистемою Apple дозволяє легко синхронізувати та слухати музику на пристроях Apple.

- YouTube Music: YouTube Music використовує велику бібліотеку відеокліпів з YouTube для стрімінгу музики. Вона пропонує не тільки аудіо, але й відеоверсії пісень та концертів.. Однією з переваг YouTube Music є можливість відтворювати не тільки музичні композиції, а й відеоролики у фоновому режимі на мобільних пристроях.

Вибір найкращого сервісу залежить від вашої пристрасті до музики, платформи, яку ви використовуєте, та наявності специфічних функцій, які вам потрібні.

2.2 Використання рекомендаційних систем популярними стрімінговими сервісами

У сфері музики існує кілька популярних сервісів, які використовують рекомендаційні системи для забезпечення користувачам персоналізованого музичного досвіду. Ось кілька з них:

- Spotify: Використовує методи, колаборативного фільтрінг (collaborative filtering) а також ті що базуються на контенті. Spotify також використовує контекстуальну інформацію, таку як час дня, день тижня та місцезнаходження, для покращення рекомендацій.
- Apple Music: Apple Music використовують комбінацію методів, включаючи аналіз поведінки користувачів, колаборативний фільтрінг, алгоритми засновані на контенті та машинне навчання для забезпечення рекомендацій, які враховують смаки та уподобання користувача.
- YouTube Music: YouTube Music використовує рекомендаційну систему, що базується на широкому спектрі факторів для надання персоналізованих рекомендацій користувачам. Вона враховує історію перегляду відео на YouTube, популярні треки, схожість на інших користувачів та контекстуальні фактори, такі як час дня та місцезнаходження. YouTube

Music також використовує розширену семантичну аналітику для розуміння зв'язків між треками та жанрами музики.

Лідером серед цих сервісів є Spotify. Spotify має широкий обсяг контенту, найбільшу базу користувачів та довгий час роботи над вдосконаленням своїх рекомендаційних систем. Вони інтегрують різноманітні методи, враховують різні фактори та використовують машинне навчання для постійного поліпшення якості рекомендацій. Крім того, їхня здатність адаптуватися до особливостей користувачів і змінюваних уподобань допомагає їм залишатися попереду конкурентів.

Варто зазначити, що рекомендаційні системи цих сервісів постійно розвиваються, і можуть використовувати інші методи та підходи, ніж ті, які зазначені в інтернеті.

Після аналізу різних музичних стрімінгових сервісів та порівняння їх функціональності, репутації та наявних інтеграційних можливостей, обрання Spotify для створення системи рекомендацій є обґрунтованим. Spotify відрізняється широким каталогом музики, потужним алгоритмом рекомендацій та наявністю високоякісного API для розробників. Крім того, популярність та широка база користувачів Spotify створюють потенціал для досягнення більшої аудиторії та популяризації веб-додатку.

2.3 Практичне дослідження PC на стрімінгових сервісах

Сайтом для розбору обрано Spotify.com (Рисунок. 2.1). На момент входу Spotify пропонує користувачеві одразу зареєструватись і попереджає, що прослуховування буде з рекламою. На головній сторінці одразу пропопнує неавторизованому користувачу набір плейлістів згрупованих за жанрами, роками, настроєм чи популярністю.

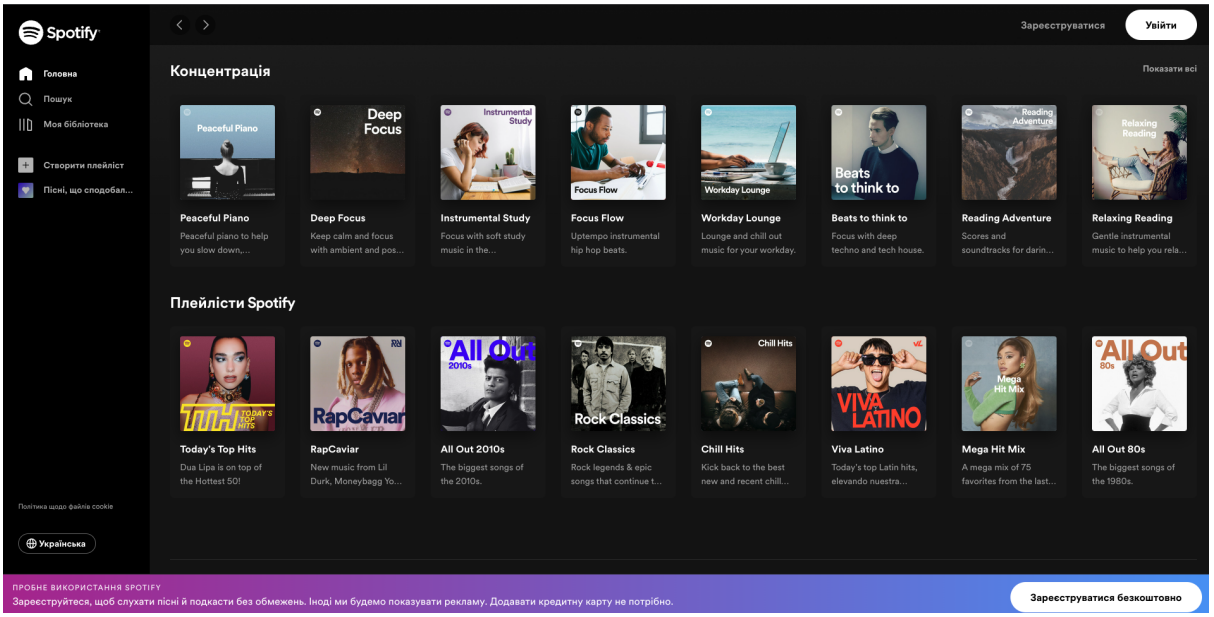

Рисунок. 2.1 - головна сторінка Spotify.com

Оскільки Spotify популярний саме завдякими своїм персоналізованим рекомендаціям - щоб побачити їх роботу на практиці треба авторизуватись. Після авторизації сервіс одразу пропонує створити перший плейліст - для того щоб отримати хоч якусь інформацію про користувача і вирішити проблему “cold-start”. Скориставшись безкоштовними функціями що пропонує цей сервіс - з рекомендацій вдалося знайти лише генерацію схожого до персонального плейліста.

З тих функцій що з’являються пілся оформлення платної підписки можна виділити окремо згенерований плейліст від Spotify під назвою “Discover Weekly - Ваш тижневий мікс свіжої музики. Насолоджуйтеся новинками й цікавинками створеними саме для вас. Оновлення щопонеділка.” Він бере до уваги всю інформацію про користувача і загалом доволі часто рекомендує непогані пісні. А також окремі рекомендації для кожного власного плейлиста (Рисунок 2.2). При вмиканні такої функції автоматично додає пісні за схожими жанрами і артистами - і робить це цілком доречно.

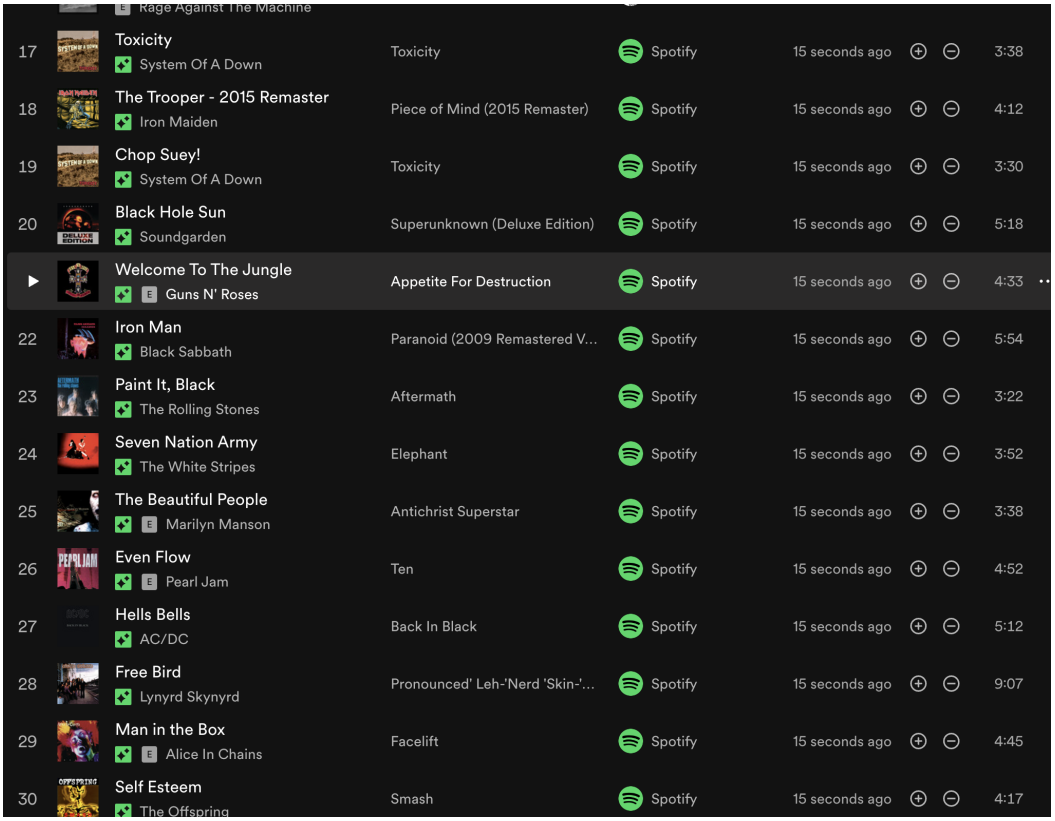

Рисунок 2.2 - рекомендації для жанру Рок

РОЗДІЛ 3. ТЕХНОЛОГІЇ ДЛЯ РОЗРОБКИ ДОДАТКУ

3.1 Концепція додатку

Метою створення окремого додатку було бажання зробити зручний та простий у використанні інструмент, що буде допомагати користувачам знаходити для себе щось нове, але схоже на те що йому подобається серед великої купи пісень що існують на сьогоднішній день.

Основною концепцією додатку - буде створення музичного сервіса, який буде рекомендувати користувачам нову музику, спираючись на ті пісні які будуть введені ним.

Користувачу не потрібно авторизуватись для того, щоб отримати рекомендації, принцип потреби у використанні додатку простий:

1. Користувач знаходить для себе пісню/пісні, що йому до вподоби;
2. Користувач хоче знайти пісні що схожі на ту що йому сподобалась/сподобались;
3. Користувач використовуючи додаток вводить в пошуку пісні/пісню і в результаті отримує плейліст з новою мизукою, яку відразу може додати собі в акаунт на стрімінговій платформі.

Для реалізації пошуку пісень та можливості додавання створеного результату собі в бібліотеку буде використовуватись API стрімінгово сервісу - Spotify [10].

Для реалізації генерації рекомендацій - буде створена відповідна рекомендаційна модель, користуючись якою, за допомогою UI користувач отримує результат.

Вибір такого простого, здавалось би, функціоналу був зроблений як раз через те, що потреба в ньому виникає саме через перевантаженість інофрмації на стрімінгових сервісах. Не зважаючи на те, що рекомендаційні системи і так інтегровані на популярних музичних платформах, зазвичай - вони не надають

рекомендації по конкретним пісням, навпаки - вони аналізують всю інформацію про користувача, всі його вподобані пісні, жанри і інші фактори - і на основі цього видають список рекомендацій, що не завжди може збігатись з тим, чого очікував користувач. В випадку ж цього додатку будуть більш конкретні запити, і відповідно більш конкретні результати: пісні що подобаються - пісні, що на них схожі.

3.2 Архітектура додатку

Архітектура веб-додатку охоплює організацію та структуру компонентів додатку, взаємодію між ними та принципи, які лежать в основі його функціонування. Основна мета архітектури полягає в створенні розширюваного, ефективного та легко підтримуваного веб-додатку.

Однією з найпоширеніших архітектур веб-додатків є модель клієнт-сервер. У цій моделі клієнт (зазвичай веб-браузер) здійснює запити до сервера, який обробляє ці запити і надсилає відповіді назад до клієнта. Ця взаємодія зазвичай відбувається за допомогою протоколу HTTP.

Основні компоненти архітектури веб-додатку включають:

1. Клієнтська сторона: Це частина додатку, що виконується на стороні клієнта (веб-браузер). Вона складається з HTML, CSS та JavaScript коду. HTML визначає структуру сторінки, CSS відповідає за її оформлення, а JavaScript забезпечує інтерактивність та динаміку.
2. Серверна сторона: Це частина додатку, що виконується на сервері. Вона забезпечує обробку запитів від клієнта, взаємодію з базою даних, обробку бізнес-логіки та генерацію відповідей. Серверні технології можуть включати такі інструменти, як Node.js, PHP, Python, Ruby, Java та інші.
3. База даних: Деякі веб-додатки використовують базу даних для зберігання та організації даних. База даних може бути використана для зберігання користувацьких облікових записів, інформації про товари, даних про

замовлення тощо. Поширені системи управління базами даних (СУБД) включають MySQL, PostgreSQL, MongoDB та інші.

4. API (Application Programming Interface): Інтерфейс програмування додатків визначає набір правил і протоколів, за допомогою яких різні компоненти додатку можуть взаємодіяти між собою. API дозволяє зовнішнім службам або іншим додаткам комунікувати з вашим веб-додатком та виконувати певні дії, такі як отримання даних або виконання дії замовлення.
5. Безпека: Веб-додатки повинні приділяти особливу увагу питанням безпеки. Це включає захист від атак, таких як внедрення коду (injection attacks), перехоплення сесій (session hijacking), злам паролів та інші. Застосування механізмів аутентифікації, авторизації та шифрування даних є важливим аспектом розробки веб-додатків.

Відповідно в реалізації поставленої задачі будуть використані такі технології (рисунок 3.1):

- Клієнтська сторона - JavaScript framework Vue.js, SCSS
- Серверна сторона - Python, Fast API, scikit-learn, numpy, pandas, spotipy
- API - Spotify API

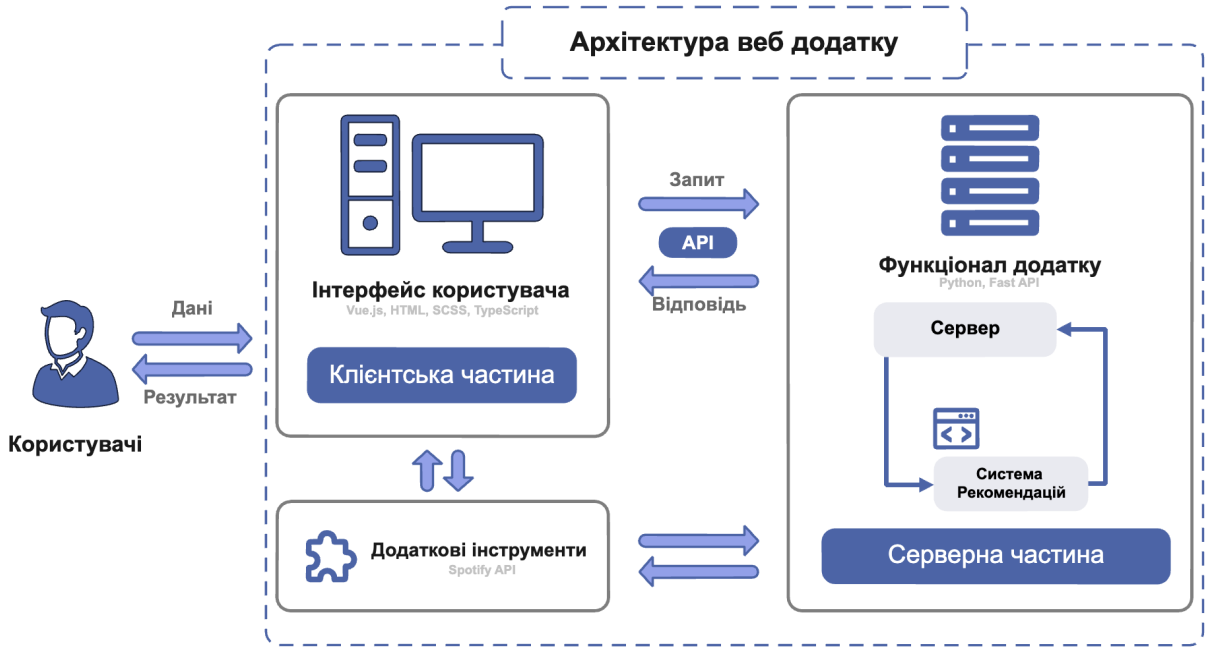

Рисунок 3.1 - архітектура веб додатку

3.3 Технології клієнтської сторони додатку

Процес розробки клієнтської сторони веб-додатку включає ряд етапів, які потрібно пройти, починаючи від планування та проектування до реалізації та тестування. Важливо відповідально підходити до розробки клієнтської сторони, оскільки це є основою веб-додатку, з яким користувачі будуть взаємодіяти. Від якості розробки залежить користувацький досвід, продуктивність та масштабованість додатку.

Обґрунтування вибору технологій:

1. Vue.js - це популярний фреймворк JavaScript для розробки клієнтської сторони веб-додатків. Використання Vue.js полегшує створення інтерактивних та динамічних інтерфейсів користувача. Class-based підхід дозволяє використовувати класи ES6 для опису компонентів, що полегшує організацію та управління кодом. Vue.js забезпечує широкий набір

функціональності, таких як двостороннє зв'язування даних, реактивність, компонентна архітектура, маршрутизація тощо. Він також має велику спільноту розробників і багато документації, що полегшує процес навчання та підтримки [8].

2. SCSS (Sass CSS) - це препроцесор CSS, який надає розширений набір можливостей та зручність у роботі з CSS. SCSS дозволяє використовувати змінні, міксіни, вкладені стилі, імпортування та багато іншого. Він полегшує організацію та управління стилями, дозволяючи зручно розбити код на модулі і перевикористовувати стилі. Крім того, SCSS дозволяє писати код більш ефективно та зручно, спрощуючи процес розробки та підтримки [9].

Переваги використання обраних технологій:

1. Продуктивність: Vue.js має легку та ефективну віртуальну DOM (Document Object Model), що призводить до швидкого рендерингу сторінок. SCSS дозволяє писати більш структурований та організований CSS код, що полегшує розробку та підтримку.
2. Розширюваність: Vue.js має широкий набір розширень та плагінів, що полегшує розширення функціональності додатку. SCSS дозволяє писати багатошаровий CSS, що дозволяє створювати складні та високо організовані стилі.
3. Спільнота та підтримка: ці технології мають велику та активну спільноту розробників. Існує багато ресурсів, документації, статей, форумів та інших джерел, що полегшує навчання та вирішення проблем.

Узагальнюючи, використання Vue.js 2 та SCSS дозволяє створювати ефективні, модульні та легко підтримувані клієнтські сторони веб-додатків. Ці технології надають зручність у розробці, покращують продуктивність та забезпечують багато можливостей для розширення функціональності.

3.4 Використання Spotify API

Spotify API є популярним і потужним інструментом для пошуку музики та отримання відповідної інформації про неї. Ось декілька вирішальних факторів, які роблять Spotify API привабливим вибором для розробки музичних додатків:

- Багатий функціонал: Spotify API надає доступ до широкого спектру функцій, включаючи пошук треків, альбомів та виконавців, отримання деталей про треки, альбоми та плейлисти, отримання аудіофайлів, отримання списку треків для певного виконавця або альбому, управління користувацькими плейлистами, отримання рекомендацій та багато іншого [10].
- Велика база даних: Spotify має одну з найбільших баз даних музичного контенту, що охоплює мільйони треків, альбомів та виконавців різних жанрів. Це дозволяє отримати доступ до широкого спектру музичного контенту для використання у власних додатках.
- Розширена функціональність плейлистів: Spotify API дозволяє не тільки отримувати дані про плейлисти користувачів, але й створювати нові плейлисти, додавати треки до існуючих плейлистів, управляти порядком треків тощо. Це відкриває можливості для створення власних музичних додатків, які можуть працювати з плейлистами користувачів.
- Розширений аутентифікаційний протокол: Spotify API підтримує OAuth 2.0, що дозволяє користувачам авторизуватись у своїх облікових записах Spotify та надавати доступ до їхньої музичної інформації. Це дозволяє розробникам створювати додатки, які отримують обмежений доступ до музичного контенту користувача за його згодою.

Для використання Spotify API нам знадобиться реєстрація в Spotify Developer Dashboard та створення додатка для отримання клієнтського ідентифікатора та секретного ключа. Авторизація та отримання токену доступу для використання API. Встановлення необхідних залежностей або бібліотек для

взаємодії з Spotify API (наприклад, використовуючи офіційну бібліотеку або HTTP-запити).

Для ознайомлення і подальшої зручної роботи був використаний додаток Postman, який дозволяє здійснювати REST-запити. Використання саме цього інструменту має кілька переваг, таких як “user-friendly” інтерфейс, легкість використання і зручний формат виводу JSON-відповідей. Розуміння формату відповіді відіграє важливу роль у розробці веб-додатків, особливо для інтернет-менеджера, який здійснює запити і обробляє відповіді у потрібному форматі.

Основні методи, що пропонує Spotify API, які знадобляться для реалізації:

Таблиця 3.1 – Функціональні методи Spotify API

Методи	Опис методу
GET - /search	Пошук музичного контенту за ключо словами, фільтрами та параметрами.
GET - /tracks/{id} GET - /albums/{id} GET - /artists/{id}	Отримання деталей про конкретний альбом або плейлист.
GET - /audio-features	Отримання аудіо-функції треків.
GET - /albums/{id}/tracks	Отримання списку треків для пе альбому.
POST - /users/{user_id}/playlists PUT - /playlists/{playlist_id}/tracks POST - /playlists/{playlist_id}/tracks DELETE - /playlists/{playlist_id}/track	Управління плейлистами, включ створення, редагування, додаванн видалення треків.

Ці методи дозволять взаємодіяти з базою даних Spotify та використовувати її контент у власних цілях.

3.5 Технології серверної сторони додатку

Вибір розробки серверної частини використовуючи мову Python та бібліотеки Fast API було з ряду причин:

- Простота та ефективність: Python є простою мовою програмування, що дозволяє розробляти серверну частину веб-додатку швидко та зручно [11]. FastAPI, з іншого боку, надає високошвидкісну роботу та ефективність завдяки використанню асинхронного підходу та оптимізаціям.
- Асинхронність та швидкодія: FastAPI базується на асинхронному фреймворку Starlette та використовує корутини для обробки багатьох запитів одночасно [12]. Це дозволяє досягти високої швидкодії та масштабованості веб-додатків, особливо при великому навантаженні.
- Документація та спільнота: Python має широку спільноту розробників, що забезпечує наявність багато документації, пакетів та ресурсів. FastAPI також має добре документовану та активну спільноту, що сприяє легкості вивчення та підтримці фреймворку.
- Вбудована підтримка стандартів: FastAPI надає вбудовану підтримку для стандартних специфікацій API, таких як OpenAPI та JSON Schema. Це спрощує розробку та документування API і забезпечує сумісність з іншими інструментами та сервісами.
- Широкий функціонал: FastAPI має багатий функціонал, включаючи маршрутизацію, автоматичну перевірку типів даних, валідацію запитів та багато іншого. Це спрощує розробку та підтримку серверної частини веб-додатку.

Переваги використання Python та FastAPI включають простоту розробки, швидкодію, асинхронність, добре документовану спільноту та вбудовану підтримку стандартів API. Ці технології роблять процес розробки серверної частини веб-додатку ефективним та зручним.

3.6 Технології для створення РС

Рекомендаційна система буде частиною серверної сторони усього веб-додатку, але для її створення і вдосконалення потрібно виконати ряд окремих кроків:

1. Збір даних: за допомогою Spotify API зібрати релевантну інформацію про треки та відповідні метадані. Зберегти ці дані у структурованому форматі для подальшого аналізу та навчання моделі [13].
2. Аналіз даних: покращити представлення даних. При потребі створити додаткові функції, які можуть відображати значущі патерни або взаємозв'язки. Розглянути можливість включення специфічних або зовнішніх наборів даних для збагачення даних [14].
3. Попередня обробка даних: Очистити зібрані дані, видаливши всі нерелевантні або повторювані записи. Витягніть з даних релевантні характеристики - атрибути пісень. Перетворення даних у формат, придатний для машинного навчання, наприклад, у числове або категоріальне представлення.
4. Навчання моделі: обрати відповідну модель машинного навчання для генерації плейлистів. Оптимізувати продуктивність моделі на основі метрик оцінки та методів перехресної перевірки.

Для відтворення всіх цих кроків буде використано бібліотеки Python Scikit-learn, Numpy, Pandas та Matplotlib - що є потужними інструментами для аналізу даних, машинного навчання, обробки числових обчислень та візуалізації даних. Ось що вони являють собою та для чого їх використовують:

1. Scikit-learn: є однією з найпопулярніших бібліотек для машинного навчання в Python. Вона надає широкий спектр алгоритмів для класифікації, регресії, кластеризації, зменшення розмірності та багатьох інших завдань машинного навчання [15]. Scikit-learn також містить інструменти для підготовки та передобробки даних, оцінки моделей,

валідації та підбору параметрів. Вона є дуже добре документованою та підтримується активною спільнотою, що робить її вибором номер один для багатьох машинно-навчальних завдань.

2. NumPy: є основною бібліотекою для наукових обчислень в Python [16]. Вона надає підтримку для многовимірних масивів, векторизації обчислень та багатьох математичних операцій. NumPy є основною складовою багатьох інших бібліотек, включаючи Scikit-learn та Pandas. Використання NumPy дозволяє виконувати швидкі та ефективні обчислення над великими об'ємами даних.
3. Pandas: є бібліотекою для обробки та аналізу даних [17]. Вона надає потужні структури даних, такі як DataFrame, що дозволяють легко виконувати операції з фільтрації, сортування, групування та об'єднання даних. Pandas також надає зручний спосіб роботи з великими та складними наборами даних, включаючи читання та запис даних у різних форматах, таких як CSV, Excel, SQL та інші.
4. Matplotlib: Matplotlib є бібліотекою для візуалізації даних. Вона надає широкий спектр функцій та інструментів для створення різноманітних графіків, діаграм, діагностичних зображень та інших візуалізаційних елементів [18]. Matplotlib дозволяє відтворювати дані у вигляді графіків, що допомагає виявляти залежності, тренди, аномалії та інші характеристики даних.

Загалом, Scikit-learn, NumPy, Pandas та Matplotlib є незамінними інструментами для роботи з даними, візуалізації та розв'язання завдань машинного навчання у мові програмування Python. Вони поєднуються між собою та доповнюють один одного, що дозволяє зручно та ефективно працювати зі складними завданнями аналізу та моделювання даних.

РОЗДІЛ 4. ПРАКТИЧНА ЧАСТИНА РОЗРОБКИ ДОДАТКУ

4.1 Створення РС

Проаналізувавши можливі варіанти було вирішено скористатись даними у відкритому доступі що надає Spotify API, і за допомогою інформації що вони збирають про пісні створювати модель.

Для того щоб зібрати достатню кілкість даних треба було обійти граничну кількість результатів що надає результат запитів від Spotify API. Для цього було створено алгоритм:

1. Вивантажити список альбомів з 1924 по 2023 рік;
2. Для кожного альбому вивантажити пісні;
3. Для кожної пісні отримати додаткову інформацію, на основі якої буде будуватись модель.

В результаті збору даних було отримано майже мілліон пісень, і відповідні їм характеристики (рисунок 4.1).

```
RangeIndex: 981527 entries, 0 to 981526
Data columns (total 19 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   danceability      981527 non-null  float64
 1   energy            981527 non-null  float64
 2   key               981527 non-null  int64
 3   loudness          981527 non-null  float64
 4   mode              981527 non-null  int64
 5   speechiness       981527 non-null  float64
 6   acousticness      981527 non-null  float64
 7   instrumentalness  981527 non-null  float64
 8   liveness          981527 non-null  float64
 9   valence           981527 non-null  float64
 10  tempo             981527 non-null  float64
 11  id                981527 non-null  object
 12  duration_ms       981527 non-null  int64
 13  name              981523 non-null  object
 14  artists           981527 non-null  object
 15  type              981527 non-null  object
 16  explicit          981527 non-null  int64
 17  year              981527 non-null  int64
 18  release_date      981527 non-null  object
```

Рисунок 4.1 - дані отримані в результаті створеного алгоритму

Далі можемо приступити до аналізу отриманих результатів щоб отримати повне уявлення про набір даних, з яким доведеться працювати. Треба вивчети структуру даних, типів змінних, наявності пропущених значень та викидів. Таке розуміння допоможе прийняти обґрунтовані рішення щодо попередньої обробки та очищення даних. Також варто звернути увагу на аномалії, які можуть спотворити результати моделі. Виявлення і обробка цих аномалій є важливим кроком для забезпечення якості та надійності моделі машинного навчання. Обрати релевантні ознаки. Це допоможе зменшити розмірність даних, виключити незначущі ознаки і зосередитися на ключових факторах. Для цього спершу подивимось на кореляцію між параметрами за допомогою інструменту “Heat Map” (рисунок 4.2).

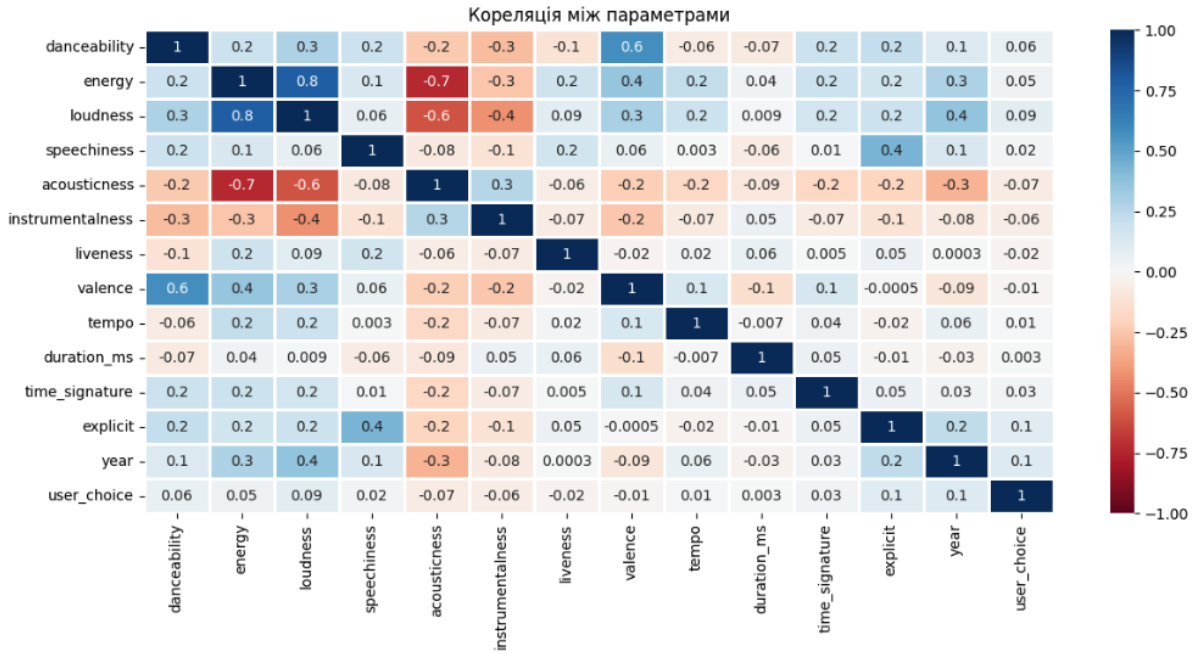

Рисунок 4.2 - кореляція між праметрами

Можна помітити, що існує слабкий взаємозв'язок між енергійністю та акустичністю музичних композицій. З іншого боку, було встановлено, що існує сильний позитивний зв'язок між енергійністю та гучністю. Таким чином, треки, які відзначаються високою енергійністю, зазвичай також відрізняються значною гучністю (рисунок 4.2).

Після аналізу відфільтровуємо ті параметри що нам не знадобляться,

і переходимо до стандартизації набору данних за допомогою “Standard Scaler”, маючи 14 числових параметрів отримаємо набір точок в 14 вимірному просторі, що буде відповідати кожній з наших пісень - а далі щоб отримати рекомендації будемо шукати наближчі точки до вхідної, обраховуючи дистанцію між ними за допомогою Евклідової метрики:

$$\sqrt{\sum_{i=1}^{n}(p_i - q_i)^2},$$

де p_i та q_i - координати точок,

n - розмірність простору

Можемо перевірити результат роботи моделі - для прикладу введемо такі тестові пісні: 'House of cards - Radiohead', 'Cherry - Jungle', 'Every Other Freckle - alt-J', 'Mountain at My Gates - Foals', 'Give Me the Night - George Benson', отримали такий результат (рисунок 4.3).

	name	year	artists
0	Lovely Day	2006	['Peter White']
1	The Secret of Life	1998	['Faith Hill']
2	You Don't Know What You're Missing	2013	['George Strait']
3	One Night At A Time	2011	['George Strait']
4	Rainbow	2009	['Colbie Caillat']
5	Scarlet Begonias - 1994/Live On KUCI, Irvine	2006	['Sublime']
6	Cigarettes And Chocolate Milk - Reprise	2001	['Rufus Wainwright']
7	Olvidala	2002	['Beto Y Sus Canarios']
8	El Amor De Mi Vida	2002	['Chon Arauza Y Su Furia Colombiana']
9	Just A Closer Walk With Thee/Take My Hand Lord...	1999	['Anne Murray', 'Tommy West']
10	The Secret of Life - 2007 Remaster	2007	['Faith Hill']
11	Moving On	2015	['Mat Kearney']
12	The Whole Of The Moon	2016	['Celtic Woman']
13	Let's Go	2012	['Matt and Kim']
14	Tu Pasado	2001	['Viento Y Sol']
15	The Night Squall	2003	['Lamp']
16	Tù estàs aquì	2004	['Nek']
17	So Much To Do	2007	['Loudon Wainwright III']
18	La Libertad	2004	['Vicentico']
19	Before This Night Is Through	2009	['Bonnie Tyler']
20	Keep On Walking	2012	['Passenger']
21	Beyond	2002	['Mr. Scruff', 'Seaming To']
22	Sunshine	2001	['S Club 7']
23	Under Pressure - Remastered 2011 / New Master	2011	['Queen', 'David Bowie']
24	Keep On Walking	2013	['Passenger']
25	Chariot	2004	['Gretchen Wilson']
26	I Got You	2005	['Craig Morgan']
27	精彩	2003	['Tanya Chua']
28	Vitamina	2015	['Sonora Carruseles', 'Marinho Paz']
29	Undertow (Featuring The Fray & Esthero)	2009	['Timbaland', 'The Fray', 'Esthero']

Рисунок 4.3 - результат роботи створенної РС

4.2 Створення веб-додатку

Як можна побачити - користувач опиняється на головній сторінці веб-додатку - де все що йому треба це ввести назви пісень (рисунок 4.4).

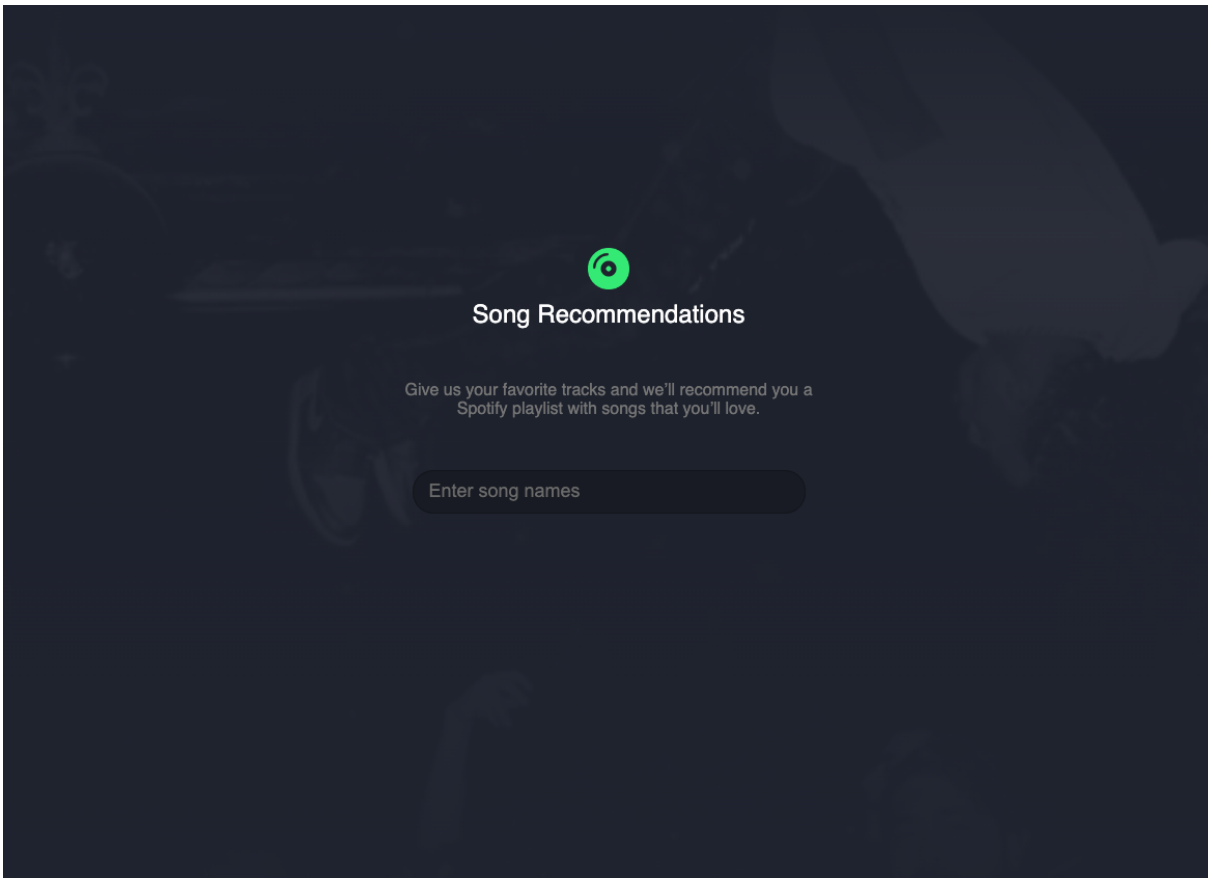

Рисунок 4.4 - головна сторінка веб-додатку

Якщо користувач не впевнений в правильності введеної пісні у нього буде можливість обрати одну з підказок нижче під полем вводу, для того щоб точно ввести правильу пісню.

Після натискання кнопки “Reccomend” - користувач опиниться на іншій сторінці де вже побачить свій згенерований плейліст (рисунок 4.5), зможе одразу його вивчити, зберегти або ж знову почати пошук за іншими піснями які йому до вподоби.

Рисунок 4.5 - результуючий плейліст з рекомендаціями

ВИСНОВКИ

В рамках огляду літератури було проведено дослідження із збору наукових джерел та інформації, пов'язаної з темою створення спеціального веб-додатку для пошуку схожої музики. Були розглянуті різні аспекти, включаючи аналіз музичних стрімінгових сервісів, рекомендаційні системи та інтеграцію з веб-додатками.

Рекомендаційні системи відіграють важливу роль у сучасному цифровому середовищі, де об'єм і розмаїття доступної інформації є великими. Вони стають незамінними інструментами для забезпечення персоналізованого та зручного досвіду споживачів у різних сферах, таких як музика, фільми, книги, товари тощо.

Було обрано Spotify як основний музичний стрімінговий сервіс для інтеграції з веб-додатком. Проведено детальний аналіз його функцій, особливостей, репутації та API. Spotify виділяється своїм великим каталогом музики, потужним алгоритмом рекомендацій та широкою базою активних користувачів.

Було проведено аналіз великого обсягу можливих варіантів даних, які можна використовувати для створення PC у створеному веб-додатку. Під час цього аналізу розглядалися різні джерела даних, які можуть бути корисними для отримання інформації про користувачів та їхні музичні вподобання. Аналіз джерел даних дав змогу визначити найбільш відповідні та корисні дані для системи рекомендацій, забезпечуючи зручні, персоналізовані та актуальні рекомендації для користувачів.

Було проведено дослідження можливих варіантів створення системи рекомендацій для спеціального веб-додатку. Під час аналізу було розглянуто різні підходи до рекомендаційних систем, зокрема системи на основі контенту, колаборативного фільтрування та гібридні підходи. Після ретельного вивчення літератури та наукових джерел було прийнято рішення обрати систему рекомендацій на основі контенту для веб-додатку. Цей підхід базується на аналізі

характеристик музичних треків, таких як жанр, темп, інструментальність тощо, і надає рекомендації на основі схожості контенту між треками.

Загалом, розроблена рекомендаційна система забезпечує зручний та персоналізований досвід для користувачів, допомагаючи їм знаходити нову музику, яка відповідає їхнім вподобанням і створює неповторну атмосферу. Вона сприяє розширенню музичного репертуару та забезпечує задоволення музичних потреб у сучасному, розмаїтому світі музики.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Recommender Systems Handbook. 10.1007/978-0-387-85820-3. Ricci, Francesco & Rokach, Lior & Shapira, Bracha & Kantor, Paul. (2011). https://www.researchgate.net/publication/234794211_Recommender_Systems_Handbook
2. Music recommender systems. Recommender systems handbook, 453-492. Schedl, M., Knees, P., McFee, B., Bogdanov, D., & Kaminskas. (2015). https://link.springer.com/chapter/10.1007/978-1-4899-7637-6_13
3. Trust in recommender systems. In Proceedings of the 10th international conference on Intelligent user interfaces (pp. 167-174). O'Donovan, J., & Smyth, B. (2005, January). https://dl.acm.org/doi/abs/10.1145/1040830.1040870
4. Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53. Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). https://dl.acm.org/doi/abs/10.1145/963770.963772
5. Content-based music recommendation system: A comparison of supervised Machine Learning models and music features. Chemeque Rabel, M. (2020). https://www.diva-portal.org/smash/record.jsf?pid=diva2%3A1515217&dswid=-9801
6. Hybrid recommender systems: Survey and experiments. User modeling and user-adapted interaction, 12, 331-370. Burke, R. (2002). https://link.springer.com/article/10.1023/A:1021240730564
7. Group recommender systems: Combining individual models. In Recommender systems handbook (pp. 677-702). Boston, MA: Springer US. Masthoff, J. (2010). https://link.springer.com/chapter/10.1007/978-0-387-85820-3_21
8. Vue.js 2 - https://v2.vuejs.org/v2/guide/
9. SCSS - https://sass-lang.com/documentation/
10. Spotify API - https://developer.spotify.com/documentation/web-api

11. Сторінка Python – https://www.python.org/
12. Fast API - https://fastapi.tiangolo.com/
13. Data mining methods for recommender systems. In Recommender systems handbook (pp. 39-71). Boston, MA: Springer US. Amatriain, X., Jaimes*, A., Oliver, N., & Pujol, J. M. (2010). https://link.springer.com/chapter/10.1007/978-0-387-85820-3_2
14. Mining gene expression data with pattern structures in formal concept analysis. Inf. Sci.. 181. 1989-2001. 10.1016/j.ins.2010.07.007. Kaytoue, Mehdi & Kuznetsov, Sergei & Napoli, Amedeo & Duplessis, Sebastien. (2011). https://www.researchgate.net/publication/220313276_Mining_gene_expression_data_with_pattern_structures_in_formal_concept_analysis
15. Бібліотека Scikit-learn - https://scikit-learn.org/stable/index.html
16. Бібліотека NumPy - https://numpy.org/doc/stable/
17. Бібліотека Pandas - https://pandas.pydata.org/docs/
18. Бібліотека Matplotlib - https://matplotlib.org/stable/index.html