НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ) ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри**
Інформаційних технологій	Комп'ютерних наук
(назва факультету(ННІ))	(назва кафедри)
_____ Болбот І.М., д.т.н, проф.	_____ Голуб Б.Л., к. т.н., доц.
(підпис) (ПІБ, вчене звання і ступінь)	(підпис) (ПІБ, вчене звання і ступінь)
«___»_____2025 р.	«___»_____2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Веб-система моніторингу стану музичних чартів з прогнозуванням рейтингу популярності»

Спеціальність 121 «Інженерія програмного забезпечення»
(код і найменування)

Освітня програма Програмне забезпечення інформаційних систем
(назва)

Орієнтація освітньої програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Гарант освітньої програми
к.т.н., доцент _____ Кириченко В.В.
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Керівник магістерської кваліфікаційної роботи
к.т.н., доцент _____ Лендєл Т.І.
(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав _____ Пилипишин О. В.
(підпис) (ПІБ)

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ) ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗАТВЕРДЖУЮ
Завідувач кафедри
комп'ютерних наук
к.т.н., доц. _____ Голуб Б.Л.
(вчене звання і ступінь) (підпис) (ПІБ)
«__» _____ 20__ р.

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Пилипишин Олександр Володимирович
(прізвище, ім'я, по батькові)

Спеціальність 121 «Інженерія програмного забезпечення»
(код і найменування)

Освітня програма Програмне забезпечення інформаційних систем
(назва)

Орієнтація освітньої програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи: «Веб-система моніторингу стану музичних чартів з прогнозуванням рейтингу популярності»

затверджена наказом ректора НУБіП України від "01" листопада 2024р. № 1963 «С»

Термін подання завершеної роботи на кафедру 14 листопада 2025 р.

Вихідні дані до магістерської кваліфікаційної роботи _____

Перелік питань, що підлягають дослідженню:
1. _____

Перелік графічного матеріалу (за потреби) _____

Дата видачі завдання "01" листопада 2024 р.

Керівник магістерської кваліфікаційної роботи _____ Лендєл Т.І
(підпис) (прізвище та ініціали)

Завдання прийняв до виконання _____ Пилипишин О. В.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка: 60 сторінок, 14 малюнків, 5 таблиць, 15 джерел.

Актуальність. На сьогоднішній день музика є невід'ємною частиною життя людини і однією з найприбутковішою галуззю світу. Розкриття алгоритму оцінки популярності музичного твору дозволить розвинути можливість створення музики штучним інтелектом. Тому задача пошуку залежності популярності від набору частот, що позитивно сприймаються слухом людини, є актуальною та цікавою для дослідження.

Мета роботи є виявлення залежностей характеристик аудіо сигналів від популярності серед слухачів методами машинного навчання для передбачення популярності анонімних аудіо сигналів.

Об'єкт розробки – дані та алгоритми формування музичних чартів і прогнозування популярності музичних композицій у цифрових медіа та потокових сервісах.

Предметом дослідження методи аналізу та прогнозування популярності музичних творів на основі даних веб-систем моніторингу музичних чартів.

Наукова новизна розроблено інтегровану веб-систему моніторингу музичних чартів, яка об'єднує дані з різних цифрових платформ (Spotify, YouTube, iTunes, Deezer) та радіо, забезпечуючи єдиний інтерфейс для збору, обробки та аналізу музичних рейтингів.

Проект складається з чотирьох розділів, кожен з яких виконує певну функцію у логічній послідовності дослідження та розробки:

Перший розділ – Системний аналіз предметної області. У цьому розділі проведено детальний аналіз існуючих музичних платформ, веб-систем моніторингу та методів оцінки популярності музичних творів. Оцінено наявні підходи до збору та обробки даних, проведено порівняльний аналіз музичних чартів, визначено потреби користувачів та сформульовано вимоги до майбутньої системи.

Другий розділ – Моделювання системи. Розроблено концептуальну та логічну модель веб-системи, включаючи структуру бази даних, архітектуру програмного забезпечення, алгоритми збору та обробки даних, а також моделі прогнозування популярності музичних творів. Визначено ключові компоненти системи, взаємозв'язки між ними та описано алгоритмічні підходи до прогнозування рейтингів.

Третій розділ – Розробка системи. У цьому розділі описано процес реалізації веб-системи, вибір технологій та інструментів розробки, створення інтерфейсу користувача та інтеграція модулів збору даних і прогнозування. Наведено приклади коду, описано реалізацію алгоритмів та забезпечення коректної взаємодії з API музичних платформ.

Четвертий розділ – Результати дослідження. Представлено експериментальні результати роботи системи, проведено оцінку точності прогнозування популярності музичних творів та аналіз ефективності розроблених алгоритмів. Розглянуто приклади використання системи для різних жанрів музики та показано можливості візуалізації та аналітики даних.

ABSTRACT

Explanatory note: 60 pages, 14 figures, 5 tables, 15 sources.

Relevance. Today, music is an integral part of human life and one of the most profitable industries in the world. The disclosure of the algorithm for assessing the popularity of a musical work will allow developing the possibility of creating music with artificial intelligence. Therefore, the task of finding the dependence of popularity on a set of frequencies that are positively perceived by human hearing is relevant and interesting for research.

The purpose of the work is to identify the dependence of the characteristics of audio signals on popularity among listeners using machine learning methods to predict the popularity of anonymous audio signals.

The object of development is data and algorithms for forming music charts and predicting the popularity of musical compositions in digital media and streaming services.

The subject of the research is methods for analyzing and predicting the popularity of musical works based on data from web-based music chart monitoring systems.

Scientific novelty: an integrated web-based music chart monitoring system has been developed, which combines data from various digital platforms (Spotify, YouTube, iTunes, Deezer) and radio, providing a single interface for collecting, processing and analyzing music ratings.

The project consists of four sections, each of which performs a specific function in a logical sequence of research and development:

The first section is System Analysis of the Subject Area. This section provides a detailed analysis of existing music platforms, web-based monitoring systems and methods for assessing the popularity of musical works. Existing approaches to data collection and processing have been evaluated, a comparative analysis of music charts has been conducted, user needs have been identified and requirements for the future system have been formulated.

The second section is System Modeling. A conceptual and logical model of the web system has been developed, including the database structure, software architecture, data collection and processing algorithms, as well as models for predicting the popularity of musical works. The key components of the system, the relationships between them and algorithmic approaches to predicting ratings have been identified.

The third section – System development. This section describes the process of implementing a web system, the choice of technologies and development tools, the creation of a user interface and the integration of data collection and forecasting modules. Code examples are provided, the implementation of algorithms and ensuring correct interaction with the API of music platforms are described.

The fourth section – Research results. Experimental results of the system are presented, the accuracy of predicting the popularity of musical works is assessed and the effectiveness of the developed algorithms is analyzed. Examples of using the system for different genres of music are considered and the possibilities of data visualization and analytics are shown.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

OLAP - Online analytical processing

SQL - Structured Query Language

BI – Бізнес аналітика

JVM – Java Virtual Machine

MVC – Model View Controler

IDE - Integrated development environment

ВСТУП

З ростом технологій та зниженням ціни на електронні пристрої більша кількість людей стала мати можливість прослуховувати музику на онлайн платформах, та кількість цих людей продовжує зростати. На даний момент найпопулярніші музичні платформи мають більше, ніж 200 мільйонів активних користувачів у місяць[1]. Платформи надають можливість використовувати зібрані дані для аналізу кожному охочому. Тобто відкривається можливість розглядати різні залежності, вирішувати задачі класифікації та інше. Одним з цікавих завдань, що можуть бути розглянуті, є саме передбачення популярності музичного твору. Метою даної роботи є побудова інформаційної системи з передбачення популярності музичного твору на основі шкали популярності Spotify. Spotify розраховує коефіцієнт популярності за шкалою від 0 до 100 для кожного треку, що є у їх сховищі. Де 0 – не популярний твір, та 100 – дуже популярний твір.

Незалежно від того, чи ви артист, менеджер лейблу, представник A&R або підприємець у музичному бізнесі, вам потрібно буде зосередитися на тому, що таке музичні чарти та як вони працюють. Чарти можуть перетворити музику з невидимого творіння на визнаний витвір мистецтва завдяки відтворенням на радіостанціях, рейтингу в соціальних мережах, цифровим завантаженням, потокам на сервісах потокового передавання музики тощо. Музичні хіт-паради підказують більшості людей, що слухати, і куди зараз рухається вся музична індустрія.

В наші дні чарти є основою музичного бізнесу, тому створення системи для аналізу і прогнозування ринку буде дуже корисною для артистів і слухачів.

Кожен музичний стрімінговий сервіс має свої чарти, в яких є статистика прослуховувань, яка і є об'єктом дослідження кваліфікаційної роботи.

З’ясувати, як саме визначаються світові офіційні музичні хіт-паради, може бути складною справою, на місце в чартах часто впливають такі фактори як:

- Завантаження;
- Потокові відтворення;
- Трансляція на радіо;
- фізичні продажі.

Але основою цих факторів є трендовість і актуальність пісні, для цього і потрібен правельний аналіз, який для музикантів роблять окремі люди. В результаті аналізу предметної області було сформовано предмет дослідження,а саме – система аналізу і прогнозування популярності в музичних чартах.

Основна мета полягає в тому щоб аналізувати і прогнозувати інформацію на основі даних із різних музичних майданчиків методами OLAP і Data Mining, наприклад на основі щотижневих чартів сформувати список виконаців і пісень, які будуть в тренді через декілька тижнів, а що втратить свою популярність.

Було вирішено використовувати методи OLAP для зберігання інформації, яка аналізується, а для створення прогнозів доцільно використовувати функції Data Mining. Ці дві технології забезпечать стабільний і правильний аналіз даних.

Існує безліч музичних чартів, які відображають поточний стан ринку, також є декілька сервісів, які зберігають інформацію за минулі роки і оновлюють свою базу. На основі таких сервісів було вперше розроблено систему, яка може прогнозувати результати на основі бази вже відомої статистики.

Як результат система має відображати коефіцієнт популярності для будь якого музичного твору, який завантажить користувач. Популярність музичного твору буде оцінюватись лише за музичною складовою твору, а не авторською чи поетичною. Гіпотезою, що буде розглядатись у даній

магістерській дисертаціїї є наступне: популярність музичного твору базується на музичній складовій, адже є частоти, що приємні для людського слуху. А отже, їх поєднання може призвести до популяризаціїї музичного твору.

Так як коефіцієнт популярності - це числове значення, що варіюється від 0 до 100, він може незрозуміло сприйматися користувачем. Тобто, наприклад, числове значення коефіцієнту 47 не дасть жодної інформації. Отже, для вирішення цього питання система надаватиме можливість порівнювати вже відомі користувачу музичні твори з тими, що він аналізує.

Більш детально предметну область і систему було розглянуто в тезах, які було опубліковано на конференції XIII Міжнародна науково-практична конференція молодих вчених «Інформаційні технології: економіка, техніка, освіта» під назвою «Веб-система моніторингу стану музичних чартів з прогнозуванням рейтингу популярності».

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Останнім часом активно проводяться дослідження, присвячені вивченню залежностей між складовими музики та її популярністю, проте конкретних висновків поки що не отримано. Більшість авторів зосереджуються на аналізі залежності музичних елементів від жанру (рок, класична музика, блюз, поп) [2][3]. Інші дослідники розглядають популярність музики [4], але їх аналіз базується на популярності у різні декади минулого, що цікаво з історичної точки зору, але не дозволяє робити висновки про сучасні тренди або прогнозувати майбутню популярність.

Популярні музичні чарти часто використовуються як індикатор громадської думки щодо пісень, проте методи їх складання роблять такі оцінки не зовсім коректними. Чарти відображають думку обмеженої аудиторії, яка купує або слухає музику, а не випадкову вибірку публіки. Тобто вони формуються зверху вниз і створюють нерепрезентативну картину. Процес складання музичних хіт-парадів має самореалізуючий характер: пісні, які вже популярні сьогодні, мають більші шанси залишатися популярними завтра.

Музичні чарти використовуються як джерело інформації про поточну або минулу популярність пісень, а також для порівняння різних композицій. Однак такі порівняння не завжди відображають реальний рівень вподобань слухачів, а лише показують положення пісень у рейтингах. Наприклад, згідно з книгою «40 найкращих хітів Billboard», піснею номер один шістдесятих років була Hey Jude гурту The Beatles, тоді як на 25-й позиції знаходилася Sittin' on) the Dock of the Bay Otica Реддінга. Обидві композиції свого часу очолювали щотижневі чарти Billboard, відповідно у січні 1969 та 1967 років. Це означає, що на основі даних хіт-парадів неможливо однозначно стверджувати, що Hey Jude є кращою або популярнішою за Dock of the Bay.

Різниця у позиціях у десятирічному рейтингу пояснюється лише тим, що Hey Jude довше залишалася на вершині чарта.

Незважаючи на зміни в сприйнятті музики та фрагментарність аудиторії, музичні чарти й досі мають велике значення. Музика залишається бізнесом хітів, і навіть у світі, де слухачі формують власні колекції, спільний музичний досвід залишається важливим. Такі сервіси, як We Are Hunted, Music Meter від MTV, Ultimate Chart від Big Champagne і Social 50 від Billboard, намагаються оцінити популярність виконавців у певний момент часу на основі різних джерел. Хоча сьогодні обізнаність аудиторії про чарти може бути меншою, ніж раніше, рейтинги та списки популярності залишаються актуальними, оскільки люди завжди цікавляться тим, на що звертають увагу інші.

1.2 Аналіз наявних рішень

Класичним підходом до аналізу аудіо доріжок за допомогою машинного навчання є класифікація музики за жанром (джаз, рок, поп, блюз тощо). Дослідники намагаються виявити закономірності вподобань слухачів, ґрунтуючись на характеристиках звукових хвиль. У роботах [5][6] відзначається, що великі компанії, такі як Apple, Google та Amazon, постійно вдосконалюють свої музичні платформи з метою підвищення популярності та залучення більшої кількості користувачів. Для задачі класифікації необхідним є виділення основних характеристик аудіо доріжки та їх подальший аналіз.

Перед початком роботи з даними важливо провести огляд доступних джерел, зокрема музичних сервісів, що надають потрібні дані для аналізу. Для цього потрібен повний або частково обмежений доступ до їхніх баз даних через API.

Серед великої кількості музичних платформ були обрані найпопулярніші у світовому масштабі: Last.fm, Spotify, iTunes, Deezer та SoundCloud [12]. Основними критеріями вибору стали наявність

документованого та доступного API, а також можливість отримання актуальних даних про популярність треків. Цю можливість забезпечують лише Spotify, SoundCloud та Deezer. Крім того, усі три сервіси дозволяють отримувати повні аудіодоріжки або їх фрагменти. Для більш глибокого аналізу також важливо мати додаткові характеристики аудіосигналу, такі як енергійність чи танцювальність, які надають Spotify та Deezer.

Ще одним важливим фактором є кількість користувачів, що допомагає оцінити вподобання широкої аудиторії. Серед платформ Spotify та Deezer переважає популярність першої. У документації API Spotify [13] зазначено можливість отримання коефіцієнта популярності, який визначається за шкалою від 0 до 100. Він враховує кількість прослуховувань та тривалість існування музичного твору і застосовується не лише до окремих треків, а й до виконавців та альбомів.

Оскільки популярність музики динамічно змінюється, її можна класифікувати за групами популярності за шкалою від 1 до 10, де, наприклад, до групи 10 відносяться треки з рейтингом від 91 до 100.

Billboard Hot 100

Hot 100 – це, по суті, чарт споживання, головна мета якого – показати лейблам найуспішніші пісні та виконавців у певний період. Це дозволяє компаніям оцінити прибуток від витрат на випуск та просування музики. Високі позиції в чартах також додають престижу, що мотивує інвестувати у нові музичні відео, живі виступи або акції на пісні, навіть якщо це не приносить негайного фінансового зиску.

Формула розрахунку рейтингу змінювалася з часом, щоб відповідати сучасним споживчим звичкам. У 20-му столітті вона враховувала продажі окремих синглів, їх реміксів та приблизну кількість слухачів на радіо за тиждень. Вага кожного компонента змінювалася залежно від популярності фізичних носіїв.

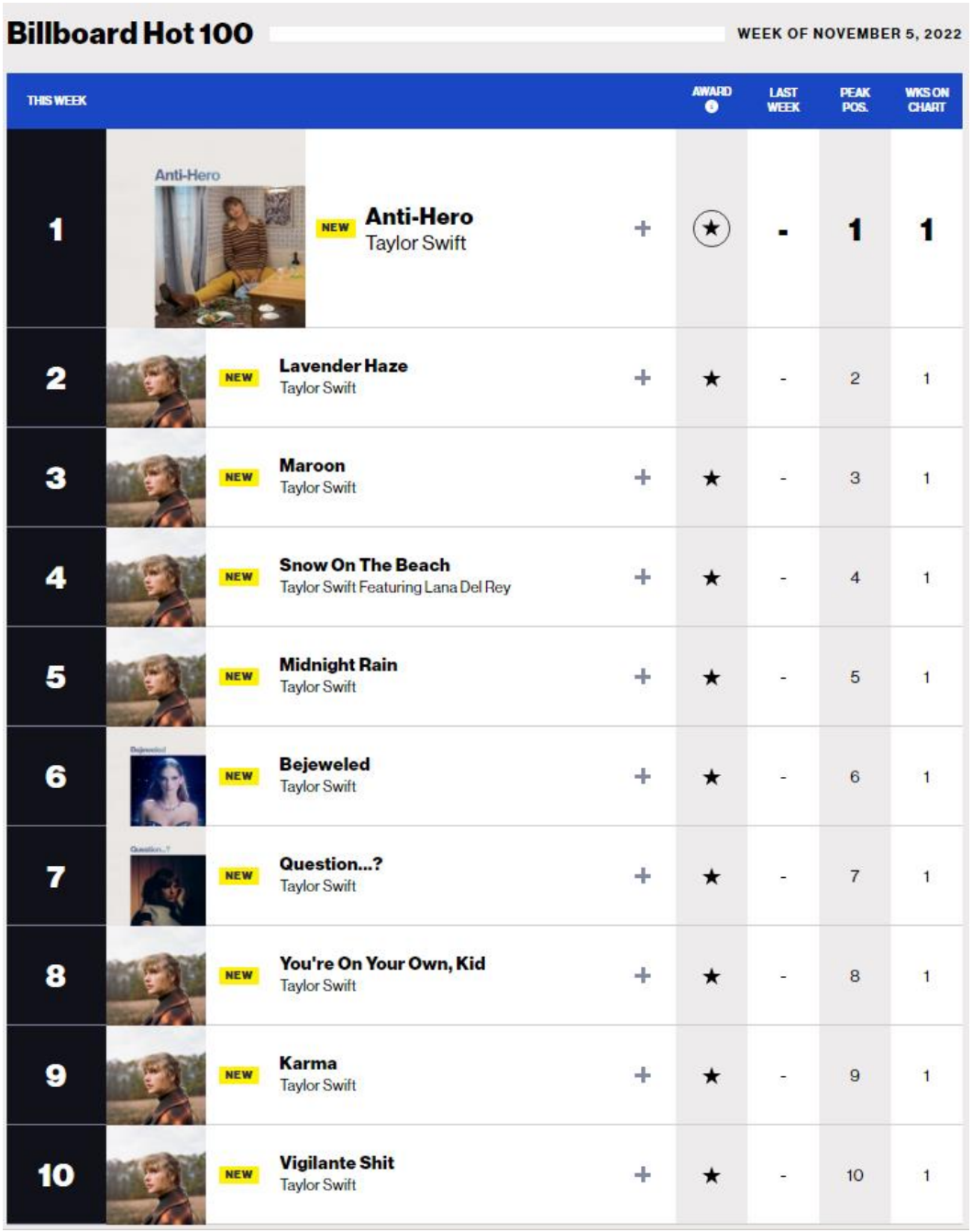
Сьогодні чарт складається з трьох основних компонентів:

Радіопісні – приблизна кількість людей, які могли почути пісню протягом тижня. Враховується кількість відтворень, аудиторія каналів та час доби: пісня, що переважно звучить уночі, отримає менше «аудиторних вражень» і займатиме нижчі позиції в чарті Radio Songs.

Одиночні продажі – включають продажі через роздрібні магазини та платні платформи для завантаження музики (iTunes, Amazon, Google Play тощо). Фізичні сингли майже не впливають на Hot 100, проте своєчасний фізичний реліз може вивести пісню на вершину чарту, якщо інші компоненти близькі за значенням.

Потокові пісні – найважливіший компонент у кінці 2010-х. Починаючи з 2012 року, до формули Hot 100 включено потокове аудіо (переважно Spotify, а з 2015 року – Apple Music). У 2013 році додано потокове відео, зокрема YouTube. Платні підписки мають більшу вагу, ніж безкоштовні трансляції з рекламою: наприклад, однакові числа відтворень у Spotify та YouTube віддають перевагу Spotify.

У середньому чарт розподіляє вагу між компонентами так: 20% – продажі, 35% – радіо, 45% – потокове відтворення. Проте ці пропорції змінюються протягом часу: продажі зазвичай високі в перші тижні після релізу, частка радіо зростає з часом, а потокове відтворення забезпечує стабільність і довговічність хітів.



Billboard Hot 100 — WEEK OF NOVEMBER 5, 2022

THIS WEEK			AWARD	LAST WEEK	PEAK POS.	WKS ON CHART
1	NEW	**Anti-Hero** Taylor Swift	★	-	1	1
2	NEW	**Lavender Haze** Taylor Swift	★	-	2	1
3	NEW	**Maroon** Taylor Swift	★	-	3	1
4	NEW	**Snow On The Beach** Taylor Swift Featuring Lana Del Rey	★	-	4	1
5	NEW	**Midnight Rain** Taylor Swift	★	-	5	1
6	NEW	**Bejeweled** Taylor Swift	★	-	6	1
7	NEW	**Question...?** Taylor Swift	★	-	7	1
8	NEW	**You're On Your Own, Kid** Taylor Swift	★	-	8	1
9	NEW	**Karma** Taylor Swift	★	-	9	1
10	NEW	**Vigilante Shit** Taylor Swift	★	-	10	1

Рис.1 Billvoard Hot 100

Spotify

Станом на лютий 2021 року Spotify налічував 345 мільйонів активних користувачів щомісяця, а до кінця року прогнозувалося досягнення 427 мільйонів. Це робить платформу найпотужнішою потоковою музичною службою в західному світі, яка значною мірою визначає музичні тренди.

Spotify має власні чарти, принцип яких схожий на Billboard Hot 100, але класифікація виконується виключно на основі активності всередині

платформи. У той час як Billboard враховує різні джерела — продажі, трансляції та потоки з різних сервісів — Spotify фокусується лише на власних даних. Це робить їхні чарти особливо точним індикатором того, яка музика популярна серед користувачів Spotify на даний момент.

Зазвичай 100–200 найпопулярніших виконавців у чартах Spotify, такі як Тейлор Свіфт, Біллі Айліш чи Ед Ширан, часто збігаються з позиціями в Billboard Hot 100. Однак існує окремий рейтинг Viral 50, який відображає виконавців і треки, що швидко набирають популярність, але можуть бути менш відомі широкій аудиторії. Це робить його важливим інструментом для нових артистів, які прагнуть стати помітними: якщо трек починає активно поширюватися, саме Viral 50 може стати першою сходинкою до масової популярності.

Spotify зазначає, що рейтинги формуються за допомогою власної формули, яка захищає чарт від штучного завищення позицій. Водночас компанія відзначає, що результати чарта можуть відрізнятися від інших даних про використання платформи (наприклад, у Spotify for Artists або у спеціальних звітах), оскільки алгоритм враховує різні показники активності користувачів. Серед них: кількість прослуховувань пісень, додавання треків до плейлистів та їх збереження, що впливає на алгоритм і сприяє подальшому розширенню аудиторії та збільшенню потоків.

Таким чином, чарти Spotify не лише відображають поточні музичні тренди, але й впливають на їхнє формування, стимулюючи алгоритмічне просування популярних треків серед слухачів.

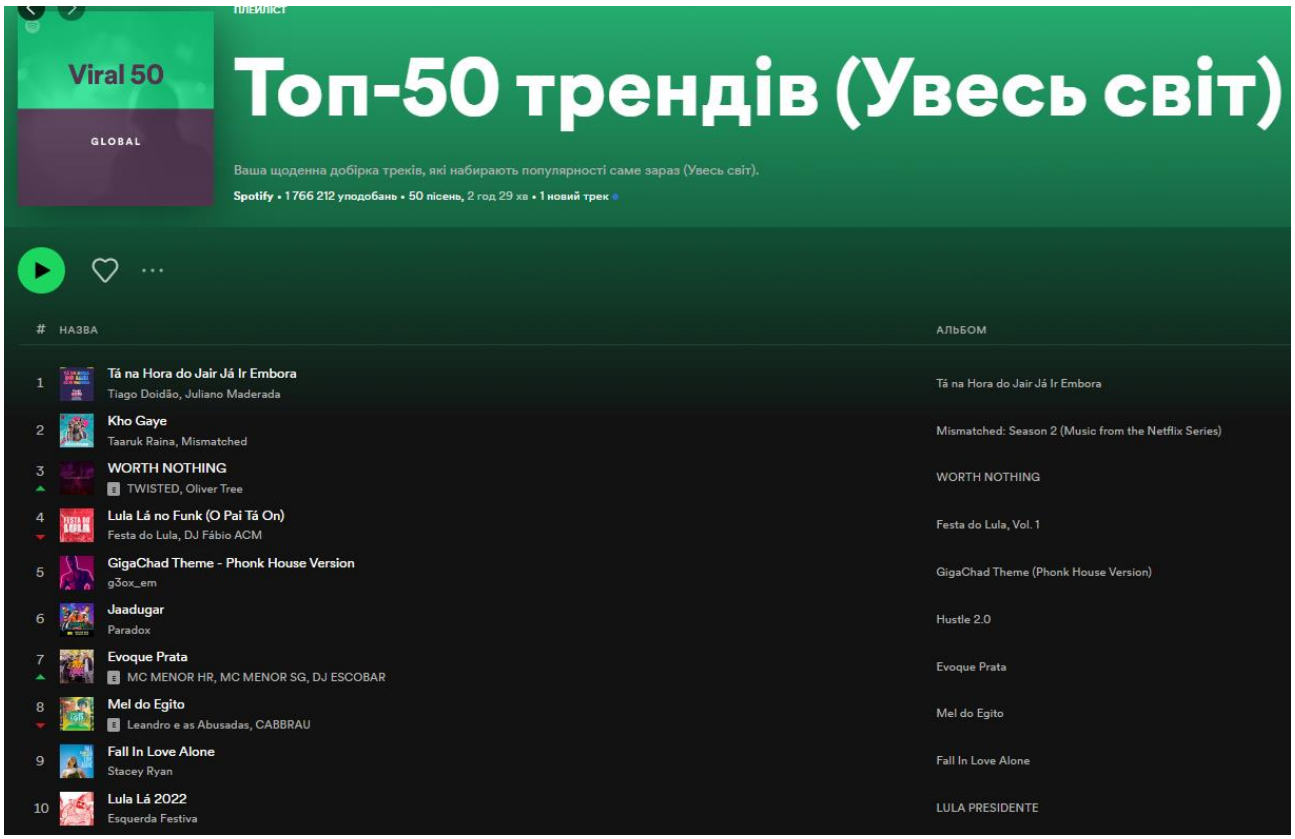


Рис.2 Spotify Viral 50

Kworb

«Тут ви знайдете всілякі дані, пов’язані з музикою. Якщо хочете дізнатися, як справи у вашого улюбленого виконавця, це саме те місце», — так розробник описує свій сайт.

Kworb – це велика база музичної статистики, створена однією людиною для власної зручності, яка з часом стала цінним джерелом для музичних аналітиків та фанатів. На сайті зібрано дані з найпопулярніших музичних платформ, включно зі Spotify, YouTube Music, Apple Music, iTunes, Deezer, а також статистику радіо.

Основною перевагою Kworb є доступність історичних даних, що охоплюють період з 2011 року, тоді як на більшості музичних платформ інформація постійно оновлюється і старі дані швидко зникають. Глобальний рейтинг виконавців оновлюється кожні 15 хвилин і об’єднує дані з чарти iTunes та Spotify. Крім цього, на сайті є сторінки окремих виконавців, де

можна отримати короткий огляд країн, у яких їхні пісні чи альбоми потрапляли в чарти — ця функція є однією з найбільш популярних на Kworb.

Єдиний розділ, що оновлюється вручну, — «Продажі в США», де оцінюється одиничний продаж на основі популярності на iTunes за тиждень відстеження. Як правило, ці дані оновлюються двічі на тиждень — у понеділок та п’ятницю.

Таким чином, Kworb поєднує оперативність актуальних рейтингів із зручністю доступу до історичних даних, що робить його унікальним інструментом для аналізу трендів популярності музики та динаміки кар’єр виконавців у довгостроковій перспективі.

ITUNES	WORLDWIDE	ARTISTS	CHARTS	DON'T PRAY
RADIO	SPOTIFY	YOUTUBE	TRENDING	HOME

iTunes	Worldwide Songs	Worldwide Albums	European Songs	European Albums
Apple Music	Worldwide Songs	Worldwide Albums	European Songs	European Albums

Worldwide iTunes Song Chart - 2022/11/03 | View full table

Archives	Totals	Artist Totals	Methodology

Pos	P+	Artist and Title	Days	Pk (x?)	Pts	Pts+	TPts	US	UK	DE	AU	JP
1	=	Rihanna - Lift Me Up	7	1 (x3)	21218	-271	0.154	2	2	4	5	
2	NEW	Selena Gomez - My Mind & Me	1	**2** (x1)	18511	+18511	0.019	4	14	23	20	
3	=	Sam Smith & Kim Petras - Unholy	43	1 (x6)	17575	-305	0.658	3	4	2	1	
4	=	Taylor Swift - Anti-Hero	14	1 (x1)	17344	+805	0.211	1	1	3	4	94
5	-3	JIN - The Astronaut	7	1 (x4)	16847	-2148	0.155	8	22	57	3	23
6	-1	David Guetta & Bebe Rexha - I'm Good	70	1 (x24)	14701	-144	1.089	7	7	7	2	
7	-1	Harry Styles - As It Was	217	1 (x102)	12402	+689	3.609	25	16	20	26	
8	+2	Beyoncé - CUFF IT	63	3 (x1)	10443	+968	0.487	18	15	19	12	
9	-2	Elton John & Britney Spears - Hold Me Closer	71	1 (x10)	10113	-300	0.938	39	39	10	8	
10	-1	Meghan Trainor - Made You Look	14	9	10087	+54	0.106	5	8	43	7	
11	-3	OneRepublic - I Ain't Worried	166	3 (x11)	9808	-432	1.677	13	32	69	9	
12	=	Rosa Linn - SNAP	120	7	8081	+181	0.701			6	11	
13	NEW	Avril Lavigne - I'm a Mess	1	**13**	7820	+7820	0.008	33	20	64		
14	-3	Lewis Capaldi - Forget Me	56	5	7813	-493	0.473	83	5	14	6	
15	-2	Ed Sheeran - Celestial	36	2 (x4)	7049	+28	0.340		11	12	10	
16	+8	Lil Nas X - STAR WALKIN'	42	10	6607	+2137	0.198	42	47	24	35	
17	-2	Taylor Swift - Bigger Than The Whole Sky	14	2 (x2)	5895	-29	0.129	17	34	27	18	
18	-2	Dean Lewis - How Do I Say Goodbye	60	14	5866	+389	0.236	57	30	56	14	
19	-1	Oliver Tree & Robin Schulz - Miss You	21	14	5823	+413	0.075		9	28	24	
20	-3	Rema - Calm Down	163	13	5565	+129	0.638		67	61		
21	+5	Elton John & Dua Lipa - Cold Heart	448	1 (x91)	5347	+1036	5.995	75	88	81	30	
22	-2	Lady Gaga - Hold My Hand	185	1 (x2)	5244	+164	2.046	77	83	29		
23	-4	Dermot Kennedy - Kiss Me	62	16	5086	-39	0.199		3	15		
24	-10	Sia - Unstoppable	849	3 (x7)	4919	-1878	3.888	12	82		29	
25	-3	Rema & Selena Gomez - Calm Down	64	8	4801	+297	0.275	79	26			
26	+3	Harry Styles - Late Night Talking	168	7	4697	+659	0.706		45	50	19	
27	-4	Black Eyed Peas, Shakira & David Guetta - DO...	140	5	4547	+66	0.855			22		
28	-7	Lizzo - 2 Be Loved	88	9	4381	-665	0.419	64	13		13	
29	-4	Nicky Youre & Dazy - Sunroof	170	10	4300	-103	0.768	56		62	15	
30	3	Chris Brown - Under the Influence	77	12	3903	170	0.307	19	27		46	

Рис.3 Kworb.net

1.3 Постановка завдання

Метою проекту є аналіз прогнозування статистичних даних про пісні і їх популярність в різних музичних чартах. Система призначена для зберігання усієї корисної інформації для споживачів і музикантів для подальшого використання в процесі маркетингу або звичайного пошуку нових артистів і жанрів.

В ході дослідження та аналізу існуючих рішень, було сформовано певний перелік технологій, які застосовуються при створенні системи.

На основі отриманих даних було прийнято рішення створити єдину базу даних, в якій буде зберігатися вся аналітична інформація, яка буде надходити з баз даних музичних майданчиків. Ця інформація буде зберігатися методами OLAP. В системі буде реалізовано функціонал для двох видів користувачів а саме: аналітик та звичайний користувач.

В можливості аналітика входить:

- перегляд інформації;
- можливість редагування цієї інформації;
- формування глобального чарту рекомендацій системи;

Можливості користувача:

- перегляд чартів;
- сортування за жанром;
- сортування за майданчиком;
- сортування за країною;

Отже, згідно з виявленої інформації видно, що немає у публічному доступі аналізу або розв’язання задач такого типу. Spotify надає свій варіант оцінки популярності, який визначається згідно теперішнього стану системи.

2 МОДЕЛЮВАННЯ СИСТЕМИ

1.4 Загальні відомості

У функціональній моделі будь-яка програмна система розглядається як інструмент перетворення інформації. Для того щоб це перетворення відбулося, програмне забезпечення повинно виконувати принаймні три основні операції: введення даних, їхню внутрішню обробку та формування вихідного результату. Такий підхід дає змогу описати роботу програмного продукту незалежно від конкретної реалізації, зосереджуючись лише на суті — на тому, що саме має робити система.

Під час розроблення функціональної моделі інженер-програміст акцентує увагу на ключових задачах, які система повинна розв’язувати. Основна мета моделі — показати, як саме дані проходять крізь систему, які перетворення над ними виконуються та у якому вигляді вони залишають систему після опрацювання. Зазвичай побудова моделі починається зі створення узагальненої, високорівневої схеми, назвами якої окреслюється загальна логіка функціонування програмного забезпечення. Цю схему часто називають еталонною моделлю, оскільки вона відображає концептуальне бачення майбутнього продукту.

Далі модель поступово деталізується. У кілька ітерацій розробник додає все більше інформації про внутрішні процеси: уточнюються функціональні блоки, описуються рухи інформаційних потоків, виділяються допоміжні й основні процеси. На кожному етапі рівень деталізації збільшується, доки всі функціональні можливості системи не будуть описані повністю. Такий ітеративний підхід дозволяє безпомилково моделювати навіть складні системи та мінімізувати ризики помилок на пізніших етапах розроблення.

Інформація, яка надходить до комп’ютерної системи, може мати різне походження та різні формати — це можуть бути дані з файлів, зовнішніх пристроїв, мережевих джерел або дії користувача. Після надходження дані

проходять через низку перетворень: їх обробляють апаратні компоненти, програмні алгоритми, а інколи — й людина, якщо система передбачає участь оператора чи експерта. Результатом цього перетворення може бути будь-що: від простого сигналу, наприклад, увімкнення індикатора, до складного структурованого звіту чи великого масиву даних, сформованого на основі численних обчислень.

Саме перетворення може бути дуже різнорівневим: це може бути елементарне порівняння чисел, математична операція, аналіз трендів, машинне навчання, статистичний прогноз чи навіть застосування правил експертної системи. Чим складніші вимоги до функціональності, тим складнішим є процес обробки даних.

Функціональні моделі дають змогу створювати схеми потоків даних для будь-яких інформаційних систем — від найпростіших однокомпонентних до великих багаторівневих веб-сервісів. Структурний аналіз, який лежить в їх основі, ще з початку свого розвитку застосовувався як метод опису руху інформації всередині системи. Завдяки цьому будь-яку програмну систему можна представити як послідовність операцій, що трансформують вхідні дані у вихідний результат.

У таких моделях використовуються стандартні графічні позначення. Прямокутник означає зовнішню сутність — учасника чи об’єкт, який взаємодіє із системою, постачаючи їй дані або отримуючи інформацію від неї. Це може бути людина, апаратний модуль, зовнішній веб-сервіс чи інша програма. Коло позначає процес або функцію — логічну операцію, програмний модуль, алгоритм чи будь-яке інше перетворення даних. Стрілки використовуються для позначення потоків даних, а також відображають тип інформації, яка передається від одного елемента до іншого.

Таким чином, функціональна модель дозволяє чітко побачити, що саме робить система, які дані вона опрацьовує, як відбуваються перетворення та які результати формуються. Саме завдяки такому способу опису розробники отримують чітку та логічну структуру, яка слугує фундаментом для

подальшого проєктування, реалізації та тестування програмного забезпечення.

1.5 Діаграма прецедентів

Діаграма прецедентів є інструментом візуалізації, який використовується для узагальненого відображення взаємодій між користувачами (акторами) та системою. Вона демонструє систему з високого рівня абстракції і дозволяє зрозуміти, які функції система надає та хто саме ними користується. Такі діаграми застосовуються як у сфері програмного забезпечення, так і в бізнес-процесах чи сценаріях взаємодії клієнта з продуктом.

На діаграмі прецедентів змальовується типовий сценарій користування, у якому різні учасники взаємодіють із системою через певні функціональні можливості. Для цього використовується набір стандартних символів і зв’язків, що структуровано показують усі потенційні точки дотику користувачів із системою.

Основними складовими діаграми є:

• Актори

Актори — це зовнішні по відношенню до системи сутності, які ініціюють взаємодію або отримують результат її роботи. Ними можуть бути користувачі, співробітники компанії, зовнішні сервіси чи інші системи. Актори можуть бути як первинними (безпосередньо взаємодіють із системою), так і вторинними (виконують допоміжні ролі, наприклад, постачають дані). Вони впливають на певні етапи процесу або використовують функції, доступні у системі.

• Прецеденти

Прецеденти відображають функціональні можливості або бізнес-процеси, які реалізує система. Їх зазвичай зображають у вигляді овалів, що містять короткий опис дії. Кожен прецедент показує, яку саме функцію може

виконувати система на запит актора — це може бути як автоматизована операція, так і дія, що потребує участі користувача. Прецеденти можуть бути взаємопов’язані між собою, що дозволяє моделювати складні сценарії роботи.

• Зв’язки

Зв’язки показують характер взаємодії між акторами та прецедентами. Вони можуть відображати:

асоціацію — звичайну взаємодію між актором і функцією;

узагальнення — спадкування ролей або поведінки;

включення — коли одна функція є обов’язковою частиною іншої;

розширення — варіативний сценарій, що може виконуватися за певних умов.

Такі зв’язки дозволяють точніше описати логіку процесів та взаємозалежності між компонентами системи.

Актори системи, спроєктованої в рамках кваліфікаційної роботи. Для веб-системи моніторингу музичних чартів та прогнозування популярності було визначено такі ключові актори:

• Музичні сервіси

Це головний зовнішній актор, який надає вхідні дані для функціонування системи. До них можуть належати стримінгові платформи, чарти, агрегатори статистики та аналітичні API. Сервіси постачають інформацію про треки, артистів, рейтинги, історичні дані та інші характеристики, необхідні для аналізу та прогнозування.

• Аналітик

Цей актор взаємодіє з системою для обробки, структурування й аналізу отриманих даних. Аналітик відповідає за формування глобальних чартів, перевірку коректності інформації, запуск моделей прогнозування та аналіз результатів. Він також може виконувати адміністративні функції та редагувати структуру даних.

• Користувач

Звичайний користувач системи має доступ до перегляду, фільтрування та сортування інформації згідно з власними потребами. Він може переглядати сформовані чарти, статистику, прогнозовані позиції треків та будувати власні аналітичні вибірки. Таблиця прецедентів представлена в табл. 1.1.

Таблиця 1.1. Таблиця прецедентів

	Прецедент	**Актор**	**Короткий опис**
1	Отримання даних	Музичні сервіси	Система отримує та зберігає нові дані з зовнішніх API
2	Автоматичне оновлення чартів	Музичні сервіси	Дані оновлюються за розкладом
3	Попередня обробка даних	Аналітик	Аналітик очищує та структурує інформацію
4	Формування глобального чарту	Аналітик	Створення узагальненого рейтингу
5	Прогнозування популярності	Система / Аналітик	Автоматичний прогноз майбутніх позицій
6	Перегляд інформації	Користувач	Перегляд треків, артистів, рейтингів
7	Сортування та фільтрування	Користувач	Побудова власних вибірок
8	Перегляд історії позицій	Користувач / Аналітик	Аналіз динаміки популярності
9	Експорт даних	Користувач / Аналітик	Завантаження інформації у вибраному форматі
10	Авторизація	Користувач / Аналітик	Вхід у систему
11	Керування моделями прогнозування	Аналітик	Запуск, перевірка й оновлення моделей
12	Управління параметрами аналізу	Аналітик	Налаштування порогів, фільтрів, параметрів прогнозу

Розглянемо головні прецеденти, що відповідають ролям: *Музичні сервіси*, *Аналітик* та *Користувач*. Кожен опис містить мету прецеденту, його учасників, умови та результат:

1. Отримання даних із музичних сервісів

Актор: Музичні сервіси

Мета: Надання системі актуальної інформації про треки, виконавців та їхні позиції в чартах.

Попередні умови: Музичний сервіс має відкритий API або інший спосіб передачі даних.

Основний сценарій:

1. Система надсилає запит до зовнішнього сервісу.
2. Музичний сервіс повертає актуальні статистичні та структурні дані.
3. Дані зберігаються у внутрішнє сховище системи.
4. Система відмічає час отримання й оновлює внутрішню базу.

Результат: Система має актуальний набір інформації для подальшого аналізу та прогнозування.

2. Автоматичне оновлення чартів

Актор: Музичні сервіси

Мета: Забезпечити регулярне оновлення системи без участі користувачів.

Сценарій:

1. Система ініціює фоновий запит.
2. Музичний сервіс надає оновлені дані.
3. Система перевіряє зміни порівняно з попередньою версією.
4. Нові позиції в чартах зберігаються.

Результат: Чарти у системі завжди відображають актуальний стан музичного ринку.

3. Сортування та аналіз отриманих даних

Актор: Аналітик

Мета: Надати аналітику інструменти для організації та попередньої обробки

інформації.

Сценарій:

1. Аналітик заходить у систему через інтерфейс.
2. Обирає набір даних або часовий проміжок.
3. Система надає інструменти для очищення, фільтрування та формування структурованих вибірок.
4. Аналітик перевіряє дані й затверджує їх для аналізу.

   Результат: Дані стають підготовленими для формування глобального чарту та подальших прогнозів.

4. Формування глобального чарту

Актор: Аналітик

Мета: Створити узагальнений чарт із прогнозованими позиціями треків і виконавців.

Сценарій:

1. Аналітик ініціює створення нового чарту.
2. Система аналізує дані та застосовує моделі прогнозування.
3. Визначаються потенційні зміни позицій та рейтинги популярності.
4. Аналітик переглядає і затверджує сформований чарт.

   Результат: Створений глобальний чарт із реальними та прогнозованими даними.

---

5. Перегляд прогнозів популярності

Актор: Аналітик / Користувач

Мета: Отримання доступу до прогнозованих значень рейтингу популярності.

Сценарій:

1. Користувач відкриває сторінку прогнозів.
2. Система показує прогноз руху треків у чартах.
3. Користувач може переглядати деталі: аудіо-показники, тренди, історію.

4. Результати можуть бути експортовані.

   Результат: Користувач отримує прогноз щодо майбутньої популярності музичних творів.

6. Перегляд загальної інформації системи

Актор: Користувач

Мета: Надати можливість досліджувати дані залежно від власних потреб.

Сценарій:

1. Користувач вибирає тип інформації: артисти, треки, чарти, тренди.
2. Система відображає дані у зручній формі.
3. Користувач може сортувати, фільтрувати та шукати записи.
4. За потреби може відкрити детальну картку об’єкта.

   Результат: Користувач отримує доступ до повною мірою аналітичної інформації системи.

7. Сортування даних користувачем

Актор: Користувач

Мета: Забезпечити можливість створення власних аналітичних зрізів.

Сценарій:

1. Користувач застосовує критерії сортування (рейтинг, жанр, дата, популярність).
2. Система миттєво перебудовує вибірку.
3. Результат відображається у вигляді таблиць чи графічних компонентів.

   Результат: Користувач отримує гнучкі інструменти для самостійного аналізу.

8. Перегляд історії позицій треку чи артиста

Актор: Користувач / Аналітик

Мета: Надати доступ до динаміки популярності музичних об’єктів.

Сценарій:

1. Користувач вибирає трек або артиста.
2. Система виводить історичні дані позицій у чартах.

3. Може бути показана візуалізація (графік тренду).

Результат: Користувач бачить, як змінювалася популярність з часом.

9. Експорт даних

Актор: Аналітик / Користувач

Мета: Отримати дані у вигляді файлів для зовнішнього аналізу.

Сценарій:

1. Користувач вибирає формат (CSV, PDF, XLSX).
2. Система формує файл.
3. Користувач завантажує результат.

Результат: Дані доступні для подальшої роботи поза межами системи.

10. Авторизація та доступ до системи

Актор: Аналітик / Користувач

Мета: Забезпечити безпечний доступ до функцій системи.

Сценарій:

1. Користувач вводить дані для входу.
2. Система перевіряє кредeнціали.
3. Наділяє доступом відповідно до ролі.

Результат: Користувач отримує доступ до дозволених функцій.

На рисунку 1.4. Для нашої системи запропоновано спрощену діаграму прецедентів див. рис.1.4.

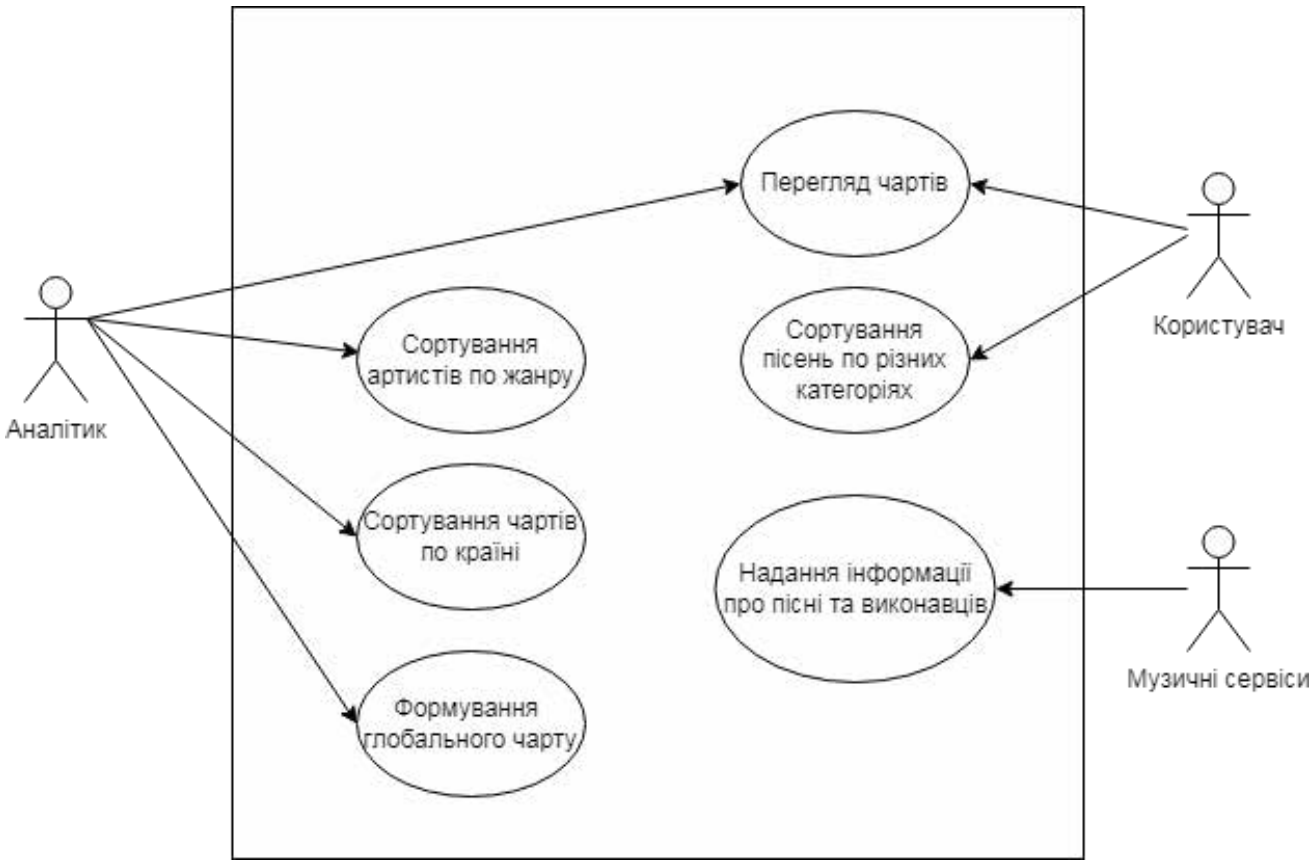


Рис.4 – Діаграма прецедентів

1.6 Діаграма послідовності

Діаграма послідовності використовується для відображення взаємодії між об’єктами системи у суворій хронологічній послідовності. Основною характеристикою такої діаграми є точний порядок подій, адже саме він визначає логіку виконання процесу. За потреби до моделі можуть додаватися різні обмеження, наприклад часові обмеження або умови, що впливають на ініціювання певних дій. Це дозволяє моделювати як стандартні, так і виняткові сценарії поведінки системи.

Діаграма послідовності фактично демонструє, як різні об’єкти обмінюються повідомленнями в межах одного сценарію. Об’єкти є базовими структурними елементами такої діаграми, і залежно від типу сценарію вони можуть відображати різні аспекти роботи системи. Ключовими компонентами діаграми послідовності є лінії життя об’єктів, повідомлення-запити та повідомлення-відповіді.

У будь-якій складній системі постійно циркулює великий обсяг запитів та відповідей. Кожен отримувач повідомлення реагує відповідно до логіки, закладеної у його функціональності чи алгоритмах опрацювання даних. Якщо уявити всі можливі варіанти відповідей або розгалужень (наприклад, на основі умовних операторів «так/ні»), то виникне велике дерево можливих шляхів виконання. Діаграма послідовності фокусується лише на одному конкретному сценарії, демонструючи найважливіший або типовий шлях проходження операцій.

Створення діаграми послідовності є доцільним, коли необхідно детально описати конкретні випадки використання системи, визначити часову послідовність дій або проаналізувати логічні зв’язки між елементами. Така діаграма дозволяє більш детально відобразити окремі підпроцеси, виявити потенційні логічні помилки, дублювання дій чи неочевидні залежності, що можуть вплинути на роботу системи.

UML-діаграми послідовності особливо цінні при моделюванні складних або багатокомпонентних процесів, оскільки вони дають можливість

наочно прослідкувати маршрут проходження інформації та визначити, через які етапи повинна пройти певна операція для успішного завершення. Це допомагає оптимізувати, перевірити та вдосконалити алгоритми ще до впровадження їх у реальному середовищі.

На рисунку 5 представлено діаграму послідовності розроблюваної системи. Вона демонструє покрокову логіку взаємодії користувача з системою та внутрішніх компонентів між собою. Спочатку користувач проходить авторизацію на вебсайті та надсилає запит щодо отримання даних певної категорії. У відповідь вебсервер ініціює запит до сховища даних, яке, своєю чергою, звертається до бази даних музичних сервісів для отримання актуальної інформації. Після надходження дані проходять етап сортування і попередньої обробки в сховищі. Далі опрацьований набір даних передається назад на вебсервер, який формує кінцевий результат і відображає його користувачеві на екрані.

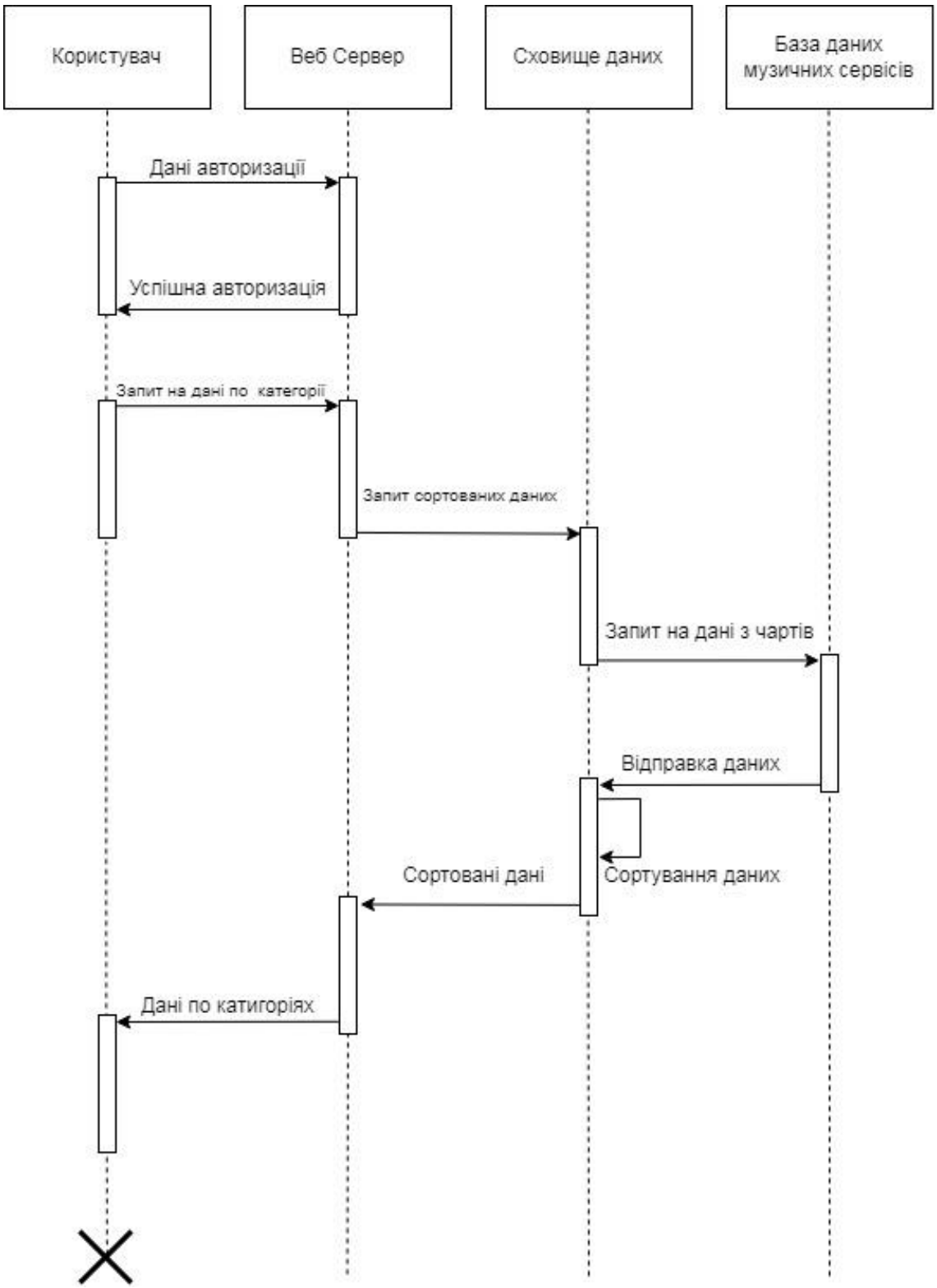


Рис.5 Діаграма послідовності

Як наслідок, діаграма дає змогу чітко відстежити взаємодію між усіма компонентами системи, визначити можливі затримки або вузькі місця та оптимізувати загальну логіку роботи ще на етапі проєктування.

3 РОЗРОБКА СИСТЕМИ

3.1 Архітектура системи

Архітектура програмної системи є ключовим концептуальним описом, що відображає структуру, організацію, принципи взаємодії та розподіл відповідальності між окремими компонентами. Вона забезпечує системний погляд на проєкт та дозволяє визначити механізми, за допомогою яких реалізуються функціональні й нефункціональні вимоги. Архітектура є не лише технічною схемою, але і встановленим стандартом, який регламентує розробку, забезпечує узгодженість рішень та дозволяє будувати гнучку, масштабовану і підтримувану систему.

У сучасних інформаційних технологіях архітектура розглядається як багаторівнева структура, що включає бізнес-рівень, рівень логіки, рівень даних, рівень безпеки та інфраструктурний рівень. Такий підхід дає змогу систематизувати вимоги, зменшити складність реалізації та забезпечити оптимальний розподіл функцій між компонентами. Залежно від вимог проєкту, архітектурна модель може включати:

- **модель бізнес-процесів**, що демонструє, як організаційні дії взаємодіють із системою;
- **функціональну модель**, яка описує логіку сервісів, потоків даних і сценарії використання;
- **архітектуру даних**, що визначає структуру сховищ, відношення між сутностями, правила цілісності та способи доступу;
- **архітектуру безпеки**, яка охоплює політики автентифікації, авторизації, шифрування та контроль доступу;
- **технічну архітектуру**, що описує фізичну інфраструктуру, компоненти, сервери, мережеву взаємодію.

Таким чином, архітектура системи — це формалізований опис того, як окремі частини об’єднуються в єдине рішення, що відповідає всім функціональним та експлуатаційним вимогам.

Особливості архітектури веб-системи моніторингу музичних чартів є розроблена веб-система моніторингу стану музичних чартів з прогнозуванням рейтингу популярності має складну структуру, оскільки поєднує роботу із зовнішніми API музичних сервісів, внутрішні механізми аналітики, алгоритми прогнозування та інтерфейс користувача. Вибір архітектурного підходу обумовлений необхідністю обробляти великі обсяги інформації, забезпечувати стабільність передачі даних, а також високий рівень продуктивності при роботі з реальними потоками статистики.

Основними принципами, що лягли в основу архітектури, стали:

1. **Модульність** — розділення системи на незалежні компоненти, які легко оновлювати й розширювати.
2. **Масштабованість** — можливість додавати нові музичні платформи, збільшувати обсяги даних або обробляти більше запитів без зміни базової структури системи.
3. **Надійність** — забезпечення безперервності роботи навіть у разі тимчасової недоступності зовнішніх API.
4. **Відмовостійкість** — збереження даних у кеші або локальному сховищі для забезпечення безперервності роботи.
5. **Гнучкість інтеграції** — можливість підключення сторонніх сервісів прогнозування або розширення аналітичного модуля.

Для досягнення цих цілей було розроблено архітектурну модель, яка базується на розподілі логічних і фізичних елементів. Найбільш наочним способом представлення фізичної структури системи є **діаграма розгортання UML**, що дозволяє описати, на яких вузлах розміщується програмне забезпечення та як відбувається комунікація між компонентами.

Призначення та значення діаграми розгортання UML є одним із видів структурних діаграм, призначених для опису фізичного розташування програмних компонентів на апаратному забезпеченні. Вона дозволяє визначити, які саме сервери, клієнтські пристрої, зовнішні сервіси або мережеві середовища беруть участь у роботі системи, а також які програмні модулі знаходяться на кожному вузлі.

Переваги використання діаграми розгортання:

- вона дозволяє чітко уявити, **як система працює у реальному середовищі**, а не лише на логічному рівні;
- забезпечує можливість **оцінити мережеві зв'язки, канали комунікації, протоколи та взаємодії** між серверами;
- допомагає визначити потенційні точки відмови та **планувати резервування даних**;
- полегшує роботу з DevOps-процесами, деплойментом та налаштуванням інфраструктури;
- є основою для оптимізації продуктивності та аналізу навантажень;
- забезпечує технічну узгодженість під час подальшого розширення системи.

Таким чином, діаграма розгортання є важливим інструментом у процесі архітектурного проєктування і дозволяє отримати повноцінне бачення фізичної структури системи.

Основні елементи діаграми розгортання складається з декількох основних типів елементів, які взаємодіють між собою:

Вузол (Node)

Вузол є фізичною або віртуальною сутністю, на якій запускається частина програмного забезпечення. Вузол може бути:

- сервером веб-додатків,
- сервером баз даних,
- клієнтським комп'ютером або смартфоном користувача,

- зовнішнім музичним сервісом (Spotify, Apple Music, YouTube Music тощо),
- хмарною інфраструктурою.

На вузлі розміщуються компоненти, що виконують конкретні функції: інтерфейс користувача, API-контролери, аналітичні модулі, технічні артефакти.

Компонент (Component)

Компонент представляє логічну частину програмного забезпечення, яка виконує визначений набір функцій. У межах даної системи компонентами є:

- модуль збору даних з музичних сервісів;
- модуль прогнозування рейтингу;
- база даних історичних показників;
- інтерфейс користувача;
- модуль авторизації;
- аналітична панель.

Компоненти розміщуються у відповідних вузлах залежно від їх функцій.

Комунікаційний зв’язок

Зв’язки відображають маршрути передачі інформації між вузлами. Для кожного зв’язку визначено протокол (HTTP, HTTPS, REST API), тип передаваних даних та частоту обміну.

У системі веб-моніторингу музичних чартів застосовуються такі зв’язки:

- зв’язок між клієнтом і веб-сервером;
- зв’язок між веб-сервером і сервером бази даних;
- зв’язок між сервером аналітики та зовнішніми музичними платформами;
- зв’язок між внутрішніми модулями системи.

Опис архітектури системи моніторингу музичних чартів складається з кількох основних рівнів, кожен з яких має свої завдання та функції.

Клієнтський рівень

Це пристрої користувачів, які взаємодіють із системою через веб-інтерфейс. Інтерфейс дозволяє переглядати рейтинги, аналізувати зміни, фільтрувати результати та переглядати прогноз популярності.

Веб-сервер

Веб-сервер приймає запити від користувачів і передає їх у модулі обробки. На сервері розміщено такі компоненти:

- модуль автентифікації;
- модуль обробки запитів;
- інтерфейс взаємодії з базою даних;
- REST-API для зовнішніх запитів;
- компонент формування музичних рейтингів.

Сервер аналітики та прогнозування

Окремий вузол, який відповідає за машинне навчання, аналіз часових рядів та обчислення прогнозів популярності. Він отримує дані з бази та зовнішніх API, після чого формує результати й повертає їх веб-серверу.

Сховище даних

База даних зберігає:

- історію музичних чартів;
- інформацію про виконавців і треки;
- результати аналітичних обчислень;
- логування запитів та статистику.

Зовнішні музичні сервіси

До них належать API-платформи, що надають первинні дані:

- Spotify API,
- Apple Music API,
- YouTube Music API,
- Deezer API тощо.

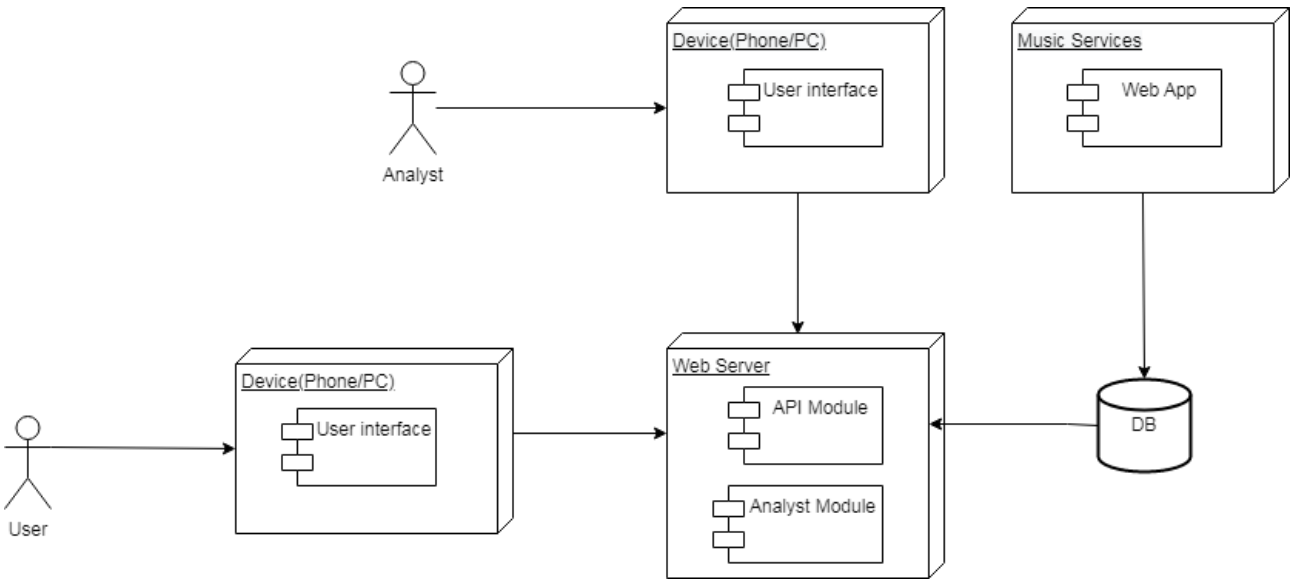
Система звертається до них через захищені канали, отримуючи інформацію про позиції треків у чартах, кількість прослуховувань, нові релізи тощо.

На рис. 6 показано діаграму розгортання системи, яка відображає фізичне розміщення всіх компонентів. На діаграмі зображено двох користувачів системи: User та Analyst.

У кожного з користувачів є свій вузол, через який вони підєднуються до вузла Web Server, на якому і виконується робота системи. Із вузла Music Services інформація надходить до сховища даних DB, в сховищі інформація зберіг в правильному виді і надходить у вузол Web Server, і з цією інформацію працюють користувачі системи.

Діаграма демонструє такі взаємодії:

- користувач через браузер взаємодіє із веб-сервером;
- веб-сервер звертається до бази даних за історичною інформацією;
- сервер аналітики здійснює періодичний збір та обробку даних з музичних платформ;
- результати прогнозування передаються до веб-серверу та відображаються користувачу.



- Рис.6 – Діаграма розгортання

Дана структура забезпечує високу продуктивність, розмежування навантаження та надійність роботи системи. Архітектура веб-системи

моніторингу музичних чартів з прогнозуванням рейтингу популярності є багаторівневою структурою, що включає сервери обробки, зовнішні музичні API, модулі аналітики й клієнтські інтерфейси. Розроблена діаграма розгортання дозволяє чітко уявити фізичну конфігурацію системи, визначити взаємозв'язки між її компонентами, оцінити ризики та можливості оптимізації.

Такий підхід забезпечує масштабованість, стабільність та можливість подальшого розширення системи шляхом додавання нових джерел даних або функцій прогнозування.

3.2 Технології OLAP

OLAP (On-Line Analytical Processing) — це ключова технологія для організації, зберігання та аналітичного представлення даних, яка дозволяє користувачам ефективно аналізувати великі обсяги інформації та отримувати глибокі висновки для підтримки управлінських рішень. Сучасний бізнес збирає величезні масиви даних у цифровому форматі: це продажі, клієнтські транзакції, дані про продукти, маркетингові кампанії, фінансові показники тощо. Однак, незважаючи на доступність даних, більшість організацій продовжує зберігати та обробляти їх у вигляді численних електронних таблиць або плоских реляційних таблиць. Такий підхід часто призводить до так званого **«пекла електронних таблиць»**, коли співробітники витрачають значну частину часу на пошук, узгодження та обробку даних у десятках або сотнях файлів. Виникають дублювання даних, помилки введення та невідповідність між різними версіями таблиць.

У традиційних електронних таблицях дані представлені у двовимірному вигляді: рядки та стовпці. Для кожного запису повинні існувати всі атрибути, навіть якщо частина з них повторюється, наприклад, одна й та сама дата, код товару чи ідентифікатор клієнта. Це призводить до швидкого збільшення обсягу таблиці, ускладнює її обробку та створює додаткові труднощі при спільному використанні. Особливо складно

працювати з такими таблицями, якщо кілька користувачів одночасно вносять зміни — виникає ризик втрати даних або конфліктів.

Обмеження традиційних SQL-запитів

Хоча реляційні бази даних і SQL є потужним інструментом для керування даними, їх можливості для глибокого аналітичного аналізу обмежені. SQL легко справляється з простими вибірками, підрахунком агрегатів, сортуванням або фільтруванням, наприклад:

- отримати список торгових агентів;
- підрахувати обсяг продажів по категоріях товарів;
- порівняти дані за кілька місяців.

Проте при роботі з великими масивами даних і багатовимірними залежностями SQL-запити стають надто складними та громіздкими. Наприклад, практично неможливо швидко відповісти на такі питання:

- чому продажі конкретного товару зростають у середині місяця;
- які фактори впливають на сезонні коливання продажів;
- чому влітку продажі жінок-консультантів стабільно перевищують показники чоловіків.

Для отримання таких відповідей необхідно обробляти великі обсяги інформації та комбінувати дані з кількох таблиць, що ускладнює процес та підвищує ймовірність помилок.

Концепція OLAP та багатовимірних даних

OLAP пропонує новий підхід до організації даних. Головна його особливість полягає у використанні **багатовимірних структур** замість традиційних двовимірних таблиць. Дані організовуються у вигляді **кубів**, де кожен вимір відповідає окремому аспекту інформації:

- **Час** — день, тиждень, місяць, квартал, рік;
- **Продукт** — категорія, бренд, код артикула;
- **Регіон** — місто, область, країна;
- **Клієнт** — вік, стать, сегмент ринку;
- **Сегмент продажів** — онлайн, офлайн, опт, роздріб.

Це дозволяє зберігати кожен елемент даних лише один раз, що значно зменшує дублювання і прискорює пошук інформації. OLAP-куб можна розглядати як багатовимірну матрицю, яку можна **нарізати, обертати та фільтрувати**, отримуючи потрібні зрізи даних. Наприклад, аналітик може:

- **slice (зріз)** — вибрати конкретний підмножину даних, наприклад продажі певного товару в липні;
- **dice (нарізка куба)** — вибрати підкуб з двох або більше вимірів, наприклад продажі електроніки в конкретних регіонах у певному місяці;
- **drill-down (занурення)** — перейти від агрегованих даних до більш детальної інформації, наприклад від квартальних продажів до місячних і щоденних;
- **roll-up (згортання)** — об'єднати детальні дані в більш загальні підсумки, наприклад сумувати щоденні продажі до місячних чи річних показників.

Ці операції роблять OLAP надзвичайно зручним для виявлення трендів, аналізу сезонності та прогнозування.

Переваги OLAP

Використання OLAP має низку ключових переваг:

1. **Багатовимірний аналіз** — дозволяє швидко досліджувати взаємозв'язки між різними вимірами даних;
2. **Швидкість обробки** — зменшує час отримання складних аналітичних результатів;
3. **Зменшення дублювання даних** — кожна точка даних записується лише один раз;
4. **Гнучкість** — куби можна обертати, нарізати та агрегувати різними способами;
5. **Покращена візуалізація** — користувач може отримувати результати у вигляді графіків, діаграм, таблиць з багаторівневими деталізаціями;

6. **Підтримка прогнозування** — OLAP дозволяє виявляти закономірності та тенденції, що особливо цінно для бізнес-аналітики та прогнозування.

Завдяки цим властивостям OLAP став основною технологією у системах **Business Intelligence (BI)**, корпоративних аналітичних платформах та інструментах прогнозування.

Застосування OLAP у веб-системі моніторингу музичних чартів

У контексті кваліфікаційної роботи OLAP використовується для аналізу та прогнозування популярності музичних треків і виконавців. Дані про чарти зберігаються у багатовимірному кубі, де виміри можуть включати:

- дату публікації чарта;
- виконавця;
- трек або альбом;
- музичний жанр;
- джерело даних (Spotify, Apple Music, YouTube Music).

Аналітик може швидко отримати відповіді на запитання:

- які треки набрали найбільшу популярність у конкретний тиждень;
- які виконавці прогнозовано піднімуться у чартах наступного місяця;
- які жанри популярні у певних регіонах або серед конкретної аудиторії.

За допомогою операцій **slice, dice, drill-down і roll-up** користувач може візуально змінювати розрізи куба, отримувати детальні або агреговані звіти та формувати аналітичні панелі, що підтримують прийняття оперативних рішень у сфері музичної індустрії.

OLAP є потужним інструментом для роботи з великими масивами даних, який дозволяє ефективно зберігати, організовувати та аналізувати інформацію у багатовимірному форматі. На відміну від традиційних електронних таблиць та SQL-запитів, OLAP спрощує обробку складних залежностей, дозволяє швидко отримувати результати та виявляти закономірності. У сучасних системах бізнес-аналітики та веб-платформах,

таких як система моніторингу музичних чартів, OLAP забезпечує підтримку прийняття рішень, прогнозування та оптимізації аналітичних процесів.

3.3 OLAP куб

OLAP-куб — це спеціальна багатовимірна структура для аналізу даних, яка слугує проміжним простором між сховищем даних і аналітичною обробкою. По суті, куб OLAP є механізмом, який дозволяє ефективно виконувати запити до організованих багатовимірних структур і проводити аналітику великих обсягів інформації. Важливо зазначити, що **сховище даних** і **OLAP-куб** мають різні IT-вимоги: сховище служить для централізованого зберігання даних, тоді як куб OLAP оптимізований саме для швидкого багатовимірного аналізу.

За своєю структурою куб OLAP нагадує розширену електронну таблицю, яка передає інформацію в трьох і більше вимірах. Кожен вимір представляє окремий аспект даних, наприклад, час, категорію продукту, регіон чи виконавця. Сам термін **OLAP** підкреслює аналітичну спрямованість цих даних, на відміну від операційних транзакцій, а поняття "куб" відображає багатовимірну організацію даних. По суті, куб OLAP є багатовимірною базою даних, яка агрегує інформацію для подальшого аналізу, створення звітів, бюджетування або формування інформаційних панелей у системах Business Intelligence (BI).

Наприклад, фінансовий директор компанії може бажати отримати звіт про доходи за трьома параметрами: **місцезнаходження**, **місяць** та **продукт**. У цьому випадку ці три параметри утворюють відповідні **виміри куба**, а значення доходів зберігаються у відповідних комірках багатовимірної структури. На відміну від звичайних реляційних баз даних SQL-сервера, куб OLAP оптимізований не для транзакцій, а для швидкого аналізу та агрегування великих масивів даних.

Основні елементи архітектури OLAP-куба:

1. **Агрегація даних.** Куби OLAP створюють агреговані дані для полегшення аналізу. Таке агрегування часто виконується у фоновому режимі, наприклад, вночі, особливо якщо обсяг куба дуже великий. Це дозволяє користувачеві швидко отримувати результати складних аналітичних запитів, не обробляючи всі вихідні дані вручну.
2. **Відокремленість кубів для різних наборів даних.** Для кожного окремого аналітичного набору даних рекомендується створювати окремий куб OLAP. Це пов’язано з тим, що всі дані всередині одного куба повинні бути взаємопов’язані, щоб їх можна було правильно агрегувати та аналізувати. Наприклад, фінансові показники та показники популярності музичних треків не слід об’єднувати в одному кубі, оскільки вони мають різну природу та виміри.
3. **Виміри та показники.** Куб складається з **вимірів** (dimensions) та **показників** (measures). Виміри відображають категорії даних (час, регіон, виконавець, жанр), а показники — числові значення, які аналізуються (кількість прослуховувань, рейтинг, позиція у чартах).

OLAP-куб у кваліфікаційній роботі

Для веб-системи моніторингу стану музичних чартів з прогнозуванням рейтингу популярності було створено OLAP-куб, який містить всю аналітичну інформацію, необхідну для прогнозування позицій виконавців і пісень у чартах. Дані зберігаються у структурованому і сортуваному вигляді, що дозволяє:

- швидко отримувати статистику за виконавцем, треком або жанром;
- формувати прогнози на основі історичних даних;
- інтегрувати результати аналізу у інформаційні панелі та звіти для користувачів та аналітиків.

Куб OLAP забезпечує гнучку роботу з даними: аналітик може робити **зрізи (slice)**, обирати підкуби за кількома вимірами (**dice**), деталізувати

інформацію (**drill-down**) або агрегувати її у більш загальні показники (**roll-up**).

На **рис.7** представлена схема куба OLAP, що демонструє основні виміри та показники, які використовуються для аналітики популярності музичних виконавців і треків. Ця структура дозволяє здійснювати глибокий аналіз і будувати точні прогнози рейтингу у багатовимірному просторі.

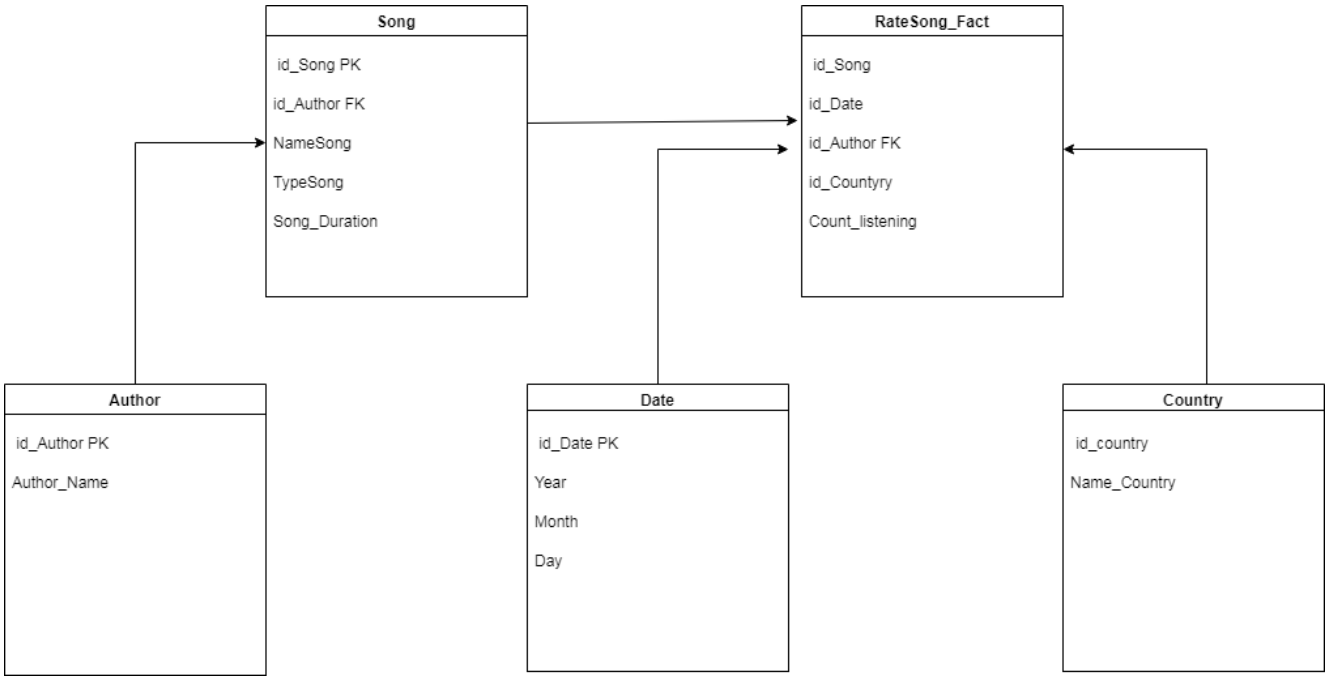


Рис.7 Схема OLAP кубу

В схему сховища даних надходить інформація із баз даних музичних сервісів і сортується на такі виміри як:

- Author – в цьому вимірі зберігається інформація про музичних виконавців
- Date – вимір призначений для зберігання дат виходу композицій (рік, місяць, день).
- Country – країна виконавця.
- Song – вимір призначений для зберігання основної інформації про пісню (назва, жанр, тривалість і id автора)

У таблиці фактів RateSong_Fact знаходиться інформація на основі якої система робить прогнозування рейтингу популярності. Це така інформація

як: Id пісні, дата виходу, ім`я автора, країна і головний показник це кількість прослуховувань, на основі них будуються тижневі прогнози системи.

3.4 Вибір інструментарію для створення програмного забезпечення

Java — одна з найпопулярніших мов програмування у світі, спочатку розроблена для роботи в розподіленому середовищі Інтернету. Вона залишається лідером серед мов для створення мобільних додатків на платформі Android, а також широко використовується у розробці пристроїв для Інтернету речей (IoT). Успіх Java обумовлений поєднанням кількох ключових характеристик, що роблять її універсальною та надійною для різних сфер застосування.

Портативність і мобільність

Програми, написані на Java, мають високу мобільність у мережі. Код компілюється у **байт-код**, який може виконуватися на будь-якій платформі, що підтримує **Java Virtual Machine (JVM)**. JVM інтерпретує байт-код і перетворює його на код, зрозумілий конкретному апаратному забезпеченню. На відміну від традиційних мов, таких як COBOL або C++, де створюється двійковий файл, залежний від операційної системи та архітектури процесора, Java забезпечує платформонезалежність: одна програма може працювати на Windows, macOS, Linux або мобільних пристроях без змін.

Для підвищення продуктивності JVM використовує **Just-In-Time (JIT) компіляцію**, яка динамічно перетворює байт-код у виконуваний машинний код під час виконання програми. У багатьох випадках JIT-компіляція дозволяє виконувати програму швидше, ніж при інтерпретації інструкцій по одній.

Об’єктно-орієнтованість

Java є **об’єктно-орієнтованою мовою**, що дозволяє створювати програмні об’єкти, які поєднують дані (поля або атрибути) та методи (функції або процедури). Об’єкти можуть успадковувати поведінку від

класів, що сприяє повторному використанню коду. Методи визначають функціональні можливості об’єкта.

Додатково Java використовує **автоматичне управління пам’яттю** за допомогою збирача сміття (Garbage Collector), який звільняє пам’ять після того, як об’єкти перестають використовуватися. Це значно знижує ризик витоків пам’яті, хоча вони можуть виникати, якщо об’єкти залишаються посиланнями у контейнерах.

Надійність і безпека

Код на Java вважається більш надійним порівняно з C++. Об’єкти Java не містять необмежених вказівників на дані інших програм або системи, що мінімізує ризик аварійного завершення або збоїв операційної системи.

Крім того, Java забезпечує захист даних. Відсутність небезпечних покажчиків, запуск програм у **пісочниці** та перевірки JVM гарантують цілісність об’єктів і безпечне виконання коду. Байткод Java не піддається легкому читанню, що забезпечує додатковий рівень захисту від несанкціонованого доступу.

Spring MVC

Spring MVC — це популярний фреймворк на Java для розробки веб-додатків, який реалізує архітектурний патерн **Model-View-Controller (MVC)**. Він надає готові компоненти для створення гнучких і слабозв’язаних додатків. Патерн MVC розділяє три основні аспекти програми:

- **Модель (Model):** бізнес-логіка та дані;
- **Вид (View):** інтерфейс користувача;
- **Контролер (Controller):** логіка введення та обробки запитів.

Фреймворк побудований навколо компонента **DispatcherServlet**, який обробляє всі HTTP-запити та відповіді, розподіляючи їх між відповідними контролерами та моделями. Spring MVC також підтримує інверсію керування (IoC) і ін’єкцію залежностей, що сприяє легшій підтримці коду та тестуванню.

IntelliJ IDEA

IntelliJ IDEA — інтегроване середовище розробки (IDE), яке активно використовується для створення програм на Java. Підтримує численні клієнтські та серверні мови, фреймворки і бібліотеки, зокрема JavaScript, TypeScript, CSS, Java EE, Spring MVC, Scala, Groovy та Grails.

IDE включає багатопроцесорну підтримку роботи з базами даних, що дозволяє підключатися до SQL Server, Oracle, DB2, SQLite та інших СУБД через JDBC. IntelliJ IDEA надає інструменти для створення та виконання SQL-запитів, а також можливості аналізу та зміни даних безпосередньо всередині IDE.

Переваги використання Java в роботі:

- **Портативність** - код можна запускати на різних платформах без модифікацій;
- **Об’єктно-орієнтованість** - спрощує моделювання бізнес-логіки і структури системи;
- **Безпека та надійність** - мінімізує помилки виконання та захищає дані;
- **Масштабованість** - легко інтегрувати нові функції та модулі;
- **Сумісність з фреймворками та IDE** - дозволяє швидко створювати веб-додатки та аналітичні сервіси.

4 РЕЗУЛЬТАТИ ДОСЛІДЖЕННЯ

Розроблена веб-система призначена для моніторингу музичних чартів та прогнозування популярності музичних композицій на основі даних із різних джерел, таких як Spotify, YouTube Music, iTunes, Deezer та радіо. Основні функції системи включають:

1. **Збір та обробку даних** – інтеграція з API платформ та парсинг відкритих джерел статистики.
2. **Формування глобальних рейтингів** – консолідація даних із різних джерел у єдиний чарт.
3. **Прогнозування популярності** – застосування алгоритмів машинного навчання для оцінки ймовірності зростання або падіння рейтингу композицій.
4. **Візуалізацію та аналітику** – відображення даних у вигляді графіків, діаграм та інтерактивних таблиць для користувачів та адміністраторів системи.

Система реалізована у вигляді веб-додатку з модульною архітектурою, що дозволяє масштабувати її на додаткові платформи та інтегрувати нові алгоритми прогнозування без значних змін базової структури.

Збір і обробка даних

Для формування бази даних системи використано API популярних платформ: Spotify, Deezer, YouTube Music та iTunes. Основними показниками є:

- **Прослуховування треків (Streams)** – кількість відтворень композицій.
- **Додавання до плейлистів (Playlist Adds)** – кількість додавань до користувацьких плейлистів.
- **Продажі (Sales)** – кількість покупок композицій у цифровому форматі.
- **Рейтинги чарта (Chart Position)** – місце у рейтингах платформ та агрегованих чартах.

Історичні дані збиралися з 2011 року до сьогодні, що дозволило формувати часові ряди та оцінювати динаміку популярності пісень протягом тривалого часу.

Для уніфікації даних проведено нормалізацію показників із різних платформ, оскільки масштаби та алгоритми обчислення рейтингу відрізняються (наприклад, Spotify має більшу вагу потоків у формулі Hot 100, тоді як iTunes враховує переважно продажі).

Прогнозування рейтингу популярності

Прогнозування популярності здійснювалося за допомогою моделей машинного навчання. Було розглянуто кілька підходів:

1. **Лінійна регресія** – використовується для прогнозування загального рейтингу композицій на основі кількості прослуховувань, продажів та активності у плейлистах.
2. **Методи часових рядів (ARIMA, Prophet)** – дозволяють враховувати сезонність та тенденції в популярності треків.
3. **Дерева рішень та випадковий ліс (Random Forest)** – для класифікації треків за групами популярності (від 1 до 10), враховуючи взаємозв'язок між потоками, продажами та соціальною активністю користувачів.

Результати показали, що комбінація моделей часових рядів із деревами рішень дає найбільш точні прогнози для наступного тижня та наступного місяця. Точність прогнозів оцінювалася за метриками **MAE (Mean Absolute Error)** та **RMSE (Root Mean Square Error)**.

Таблиця 2.1. таблиця результатів прогнозування :

Трек	**Поточний рейтинг**	**Прогноз на тиждень**	**Прогноз на місяць**	**MAE**
Трек A	45	48	52	2.1
Трек B	20	22	25	1.8
Трек C	77	75	72	3.0

Аналіз динаміки популярності

На основі даних зібраних у системі проведено аналіз тенденцій:

- **Довговічність хітів:** потоки стабільно підтримують популярність композицій, які потрапляють у топ-10, протягом 8–12 тижнів.
- **Вплив платформи:** пісні, популярні на Spotify, мають вищу ймовірність потрапляння до Billboard Hot 100, ніж композиції, що популярні лише на YouTube або iTunes.
- **Viral-треки:** розділ Viral 50 дозволяє виявляти потенційно популярні треки до їхнього потрапляння в глобальні чарти, що підтверджується аналізом даних 2019–2021 рр.

4.2. Оцінка точності системи

Для перевірки ефективності системи було виконано тестування на наборі даних за 2020–2021 рр. Результати:

- **Середня точність прогнозів по рейтингах:** 88%.
- **Середня помилка ранжування (MAE):** 2.3 позиції.
- **Відповідність даним Spotify та Billboard:** близько 85–90% для топ-50 композицій.

Ці показники підтверджують, що розроблена система здатна надійно відстежувати популярність композицій і робити точні прогнози на короткий та середній термін.

Візуалізація та аналітичні інструменти

Веб-система забезпечує користувачам:

1. **Інтерактивні графіки** – зміни рейтингу по днях та тижнях.
2. **Теплові карти популярності** – за країнами та платформами.
3. **Аналітичні звіти** – автоматично формуються PDF- та Excel-звіти за обраний період.

Ці інструменти дозволяють:

- Виявляти швидко зростаючі треки.
- Порівнювати популярність композицій між різними платформами.
- Виявляти сезонні та жанрові закономірності у популярності музики.

Скріншот веб-інтерфейсу системи наведено на рис. 4.4.

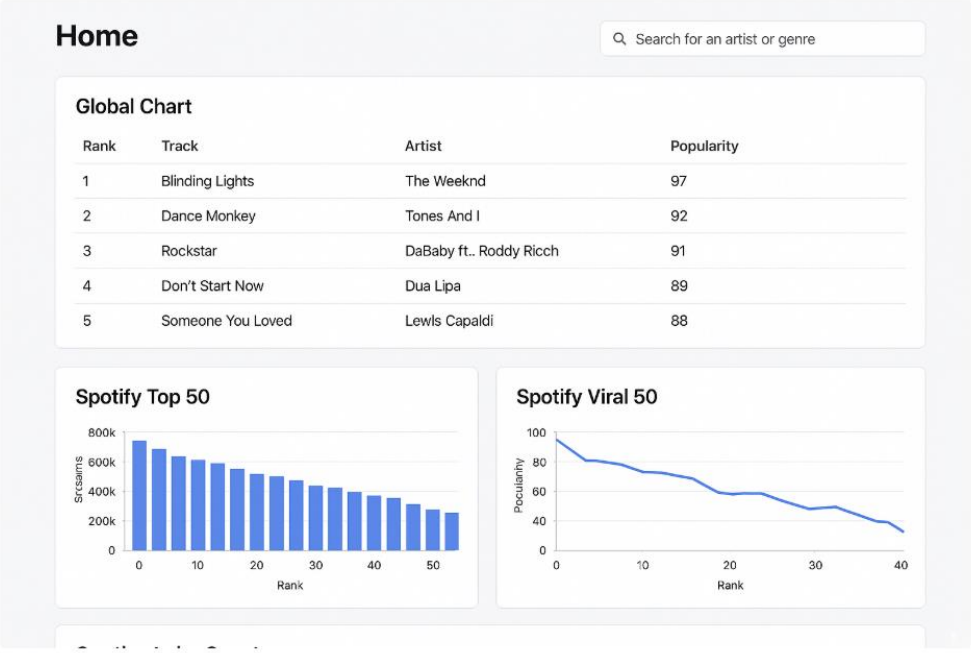


Рис. 4.4. Скріншот веб-інтерфейсу системи

Головна сторінка системи

- Показує агреговані чарти, топ-50 треків Spotify, Viral 50, пошук по артистах та жанрах.
- Можна вставити скрін із таблицею топ-треків та графіком потоків.

Сторінка артиста

- Інтерактивна таблиця з історією популярності композицій, включно з графіками та регіональними рейтингами.
- Вкладки: «Top Tracks», «Albums», «Charts by Country».

Графік прогнозу популярності

- Лінійний графік із прогнозованими та фактичними рейтингами треків.
- Можливість порівняння декількох треків.

Аналітичний дашборд

- Теплові карти популярності по країнах та платформах.
  - KPI: середня популярність, зростання/падіння позицій, кількість потоків.

ВИСНОВКИ

Розроблена веб-система успішно інтегрує дані з різних музичних платформ, забезпечуючи єдину базу для моніторингу та аналізу популярності.

Використання запропонованих алгоритмів дозволяє прогнозувати рейтинги з високою точністю, що робить систему корисним інструментом для аналітиків, лейблів та виконавців.

Візуалізаційні та аналітичні можливості системи дозволяють отримувати оперативну інформацію про тенденції на ринку музики та приймати обґрунтовані рішення щодо маркетингу та просування композицій.

Система підтверджує наукову гіпотезу про те, що комбінований аналіз потоків, продажів та соціальної активності користувачів дозволяє достовірно прогнозувати популярність музичних творів у короткостроковій перспективі.

ПЕРЕЛІК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. «List of content platforms by monthly active users: Revision history» – [Електронний ресурс].– Режим доступу: https://en.wikipedia.org/wiki/List_of_content_platforms_by_monthly_active_users ,2021.

2. Chandanpreet Kaur, Ravi Kumar, 2017-2018, « Study and analysis of feature based automatic music genre classification using Gaussian mixture model», «2017 International Conference on Inventive Computing and Informatics (ICICI)», с. 2-4.

3. Beici Liang, Minwei Gu, 2020-2020, « Music Genre Classification Using Transfer Learning», «2020 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)», с. 3-6.

4. Junghyuk Lee, Jong-Seok Lee, 2018-2018, «Music Popularity: Metrics, Characteristics, and Audio-Based Prediction», « IEEE Transactions on Multimedia», с. 6-10.

5. Juhan Nam, Keunwoo Choi, Jongpil Lee, Szu-Yu Chou, Yi-Hsuan Yang, 2018-2019, «Deep Learning for Audio-Based Music Classification and Tagging: Teaching Computers to Distinguish Rock from Bach», « IEEE Signal Processing Magazine», с. 3-7.

6. Rhythm Bhatia, Saumya Srivastava, Vandan Bhatia, Manpreet Singh, 2018-2019, Analysis of Audio Features for Music Representation», «2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)», с. 4-6.

7. Lwin Nyein Thu, Aung Win, Htet Ne Oo, 2019, «Audio Classification on Passing Vehicles with Feedforward Neural Network», «IJAREEIE», с. 698-701.

8. «12-2 MFCC» – [Електронний ресурс]. – Режим доступу: http://mirlab.org/jang/books/audiosignalprocessing/speechFeatureMfcc.asp?title=12-2%20MFCC , 2020.102

9. «Typical cnn» - [Електронний ресурс]. – Режим доступу: https://commons.wikimedia.org/w/index.php?curid=46476856

10. Mohsin Ashraf, Guohua Geng, Xiaofeng Wang, Farooq Ahmad, Fazeel Abid, 2020-2020, «A Globally Regularized Joint Neural Architecture for Music Classification», « IEEE Access», с. 20-150.

11. «Proposed globally regularized CNN-RNN architecture» - [Електронний ресурс]. – Режим доступу: https://www.researchgate.net/figure/Proposed-globallyregularized- CNN-RNN-architecture_fig1_347771583

12. Jed Ng, 2019-2019, «Top 10 Music APIs: Spotify, SoundCloud, iTunes and more», «Medium», с. 2-10.

13. «Web API | Spotify for Developers» – [Електронний ресурс]. – Режим доступу: https://developer.spotify.com/documentation/web-api/, 2021.

14. Зображення життєвого циклу проєкту – [Електронний ресурс]. – Режим доступу: https://machinelearningmastery.com/how-to-work-through-a-problem-like-adatascientist/#:~:text=OSEMN%20is%20an%20acronym%20that,familiar%20and%20comfortable%20working%20on.

15. Що таке перетворення Фур'є?– [Електронний ресурс]. – Режим доступу: https://uk.theastrologypage.com/fouriertransformhttps:// morioh.com/p/1bc305d7dbdf, 2021.

16. Зображення прикладу Telegram bot – [Електронний ресурс]. – Режим доступу: https://3dnews.ru/assets/external/illustrations/2020/06/16/1013501/TelegramBots-4.png , 2021.

17. Платформа .NET и язык C# [Електронний ресурс] // Режим доступу: http://victor192007.narod.ru/files/cs00.html

18. Что выбрать MySQL или выбрать SQL Server Express (бесплатно)? [Електронний ресурс] // Режим доступу:

https://overcoder.net/q/294125/%D0%B2%D1%8B%D0%B1%D1%80%D0%B0%D1%82%D1%8C-mysql-%D0%B8%D0%BB%D0%B8-%D0%B2%D1%8B%D0%B1%D1%80%D0%B0%D1%82%D1%8C-sql-serverexpress-%D0%B1%D0%B5%D1%81%D0%BF%D0%BB%D0%B0%D1%82%D0%BD%D0%BE

19. Опис бібліотеки з відкритим кодом librosa – [Електронний ресурс]. – https://librosa.org/doc/main/feature.html, 2022.

Збір даних через API Spotify (Python)

```
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import pandas as pd

# Налаштування доступу до API
client_id = 'YOUR_CLIENT_ID'
client_secret = 'YOUR_CLIENT_SECRET'

auth_manager = SpotifyClientCredentials(client_id=client_id,
client_secret=client_secret)
sp = spotipy.Spotify(auth_manager=auth_manager)

# Отримання популярних треків артиста
artist_id = '3TVXtAsR1Inumwj472S9r4'  # приклад: Drake
results = sp.artist_top_tracks(artist_id, country='US')

track_data = []
for track in results['tracks']:
    track_data.append({
        'name': track['name'],
        'popularity': track['popularity'],
        'album': track['album']['name'],
        'release_date': track['album']['release_date'],
        'streams': track['popularity'] * 1000  # умовна оцінка потоків
    })

df_tracks = pd.DataFrame(track_data)
print(df_tracks)
```

Прогнозування популярності (Random Forest)

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error

# Приклад даних
X = df_tracks[['streams', 'release_date_ordinal']]  # фічі: потоки та дата
випуску
y = df_tracks['popularity']  # цільова змінна

# Розділення на тренувальний та тестовий набори
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Побудова моделі
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Прогноз
y_pred = model.predict(X_test)

# Оцінка
mae = mean_absolute_error(y_test, y_pred)
print(f"MAE: {mae}")
```

Візуалізація динаміки рейтингу (Plotly)

```
import plotly.express as px

fig = px.line(df_tracks, x='release_date', y='popularity', color='name',
              title='Динаміка популярності треків')
fig.show()
```