НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	**Завідувач кафедри комп'ютерних наук**
_____ **Ігор БОЛБОТ**	_____ **Белла ГОЛУБ**
(підпис)	(підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмне забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів»

Спеціальність «121 Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення»

Гарант освітньої програми
к.т.н., доцент _____ **Ганна ВАЙГАНГ**
(підпис)

Керівник бакалаврської кваліфікаційної роботи
Старший викладач _____ **Світлана ВАСИЛЮК-ЗАЙЦЕВА**
(підпис)

Виконав _____ **Денис НЕСТЕРОВ**
(підпис)

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук

к.т.н., доцент _____Белла ГОЛУБ

«19__» грудня_____2025 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Нестерову Денису Юрійовичу

(прізвище, ім'я, по батькові)

Спеціальність «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення» Тема бакалаврської кваліфікаційної роботи

« **Програмне забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів**»

затверджена наказом від «19 грудня» 2025 р. № 3204«С»

Термін подання завершеної роботи на кафедру «22» травня 2026 р.

Вихідні дані до бакалаврської кваліфікаційної роботи:

Системи аналізу та оцінки відеоігор з рейтингами користувачів
Перелік питань, що підлягають дослідженню:

1. Аналіз існуючих аналогів та методів обробки користувацьких рейтингів у гейм-індустрії.
2. Обґрунтування алгоритмів агрегації оцінок та аналізу текстових відгуків.
3. Проєктування архітектури, бази даних та інтерфейсу програмного забезпечення.
4. Програмна реалізація, тестування та оцінка ефективності створеної системи.

Дата видачі завдання «19»грудня 2025 р.

Керівник бакалаврської кваліфікаційної робота _____ **Світлана ВАСИЛЮК-ЗАЙЦЕВА**
(підпис

Завдання взяв до виконання _____ **Денис НЕСТЕРОВ**
(підпис)

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API - прикладний програмний інтерфейс

БД - база даних

ВЗ - відеоігра

ГІК - графічний інтерфейс користувача

ДК - діаграма класів

ЗП - запит користувача

ІС - інформаційна система

КР - користувацький рейтинг

КС - комп’ютерна система

ПЗ - програмне забезпечення

ПК - персональний комп’ютер

СКБД - система керування базами даних

ТЗ - технічне завдання

UI - користувацький інтерфейс

UX - досвід користувача під час взаємодії з програмним забезпеченням

UML - уніфікована мова моделювання програмних систем

CRUD - базові операції створення, читання, оновлення та видалення даних

CSV - текстовий формат подання табличних даних

ER-діаграма - діаграма сутностей і зв’язків бази даних

ID - унікальний ідентифікатор запису в системі

JSON - текстовий формат обміну структурованими даними

MVC - архітектурний шаблон побудови програмного забезпечення

SQL - мова структурованих запитів до бази даних

HTML - мова гіпертекстової розмітки вебсторінок

CSS - мова опису зовнішнього вигляду вебсторінок

GUI - графічний інтерфейс користувача

HTTP - протокол передавання даних у мережі

HTTPS - захищений протокол передавання даних у мережі

ORM - технологія об’єктно-реляційного відображення даних

JWT - формат токена для автентифікації та авторизації користувачів

RBAC - модель керування доступом на основі ролей користувачів

ВСТУП

Сучасна індустрія відеоігор є однією з найбільш динамічних сфер цифрової економіки, що поєднує програмну інженерію, мультимедійні технології, інтерактивний дизайн, мережеві сервіси та елементи соціальної взаємодії користувачів. Відеоігри сьогодні розглядаються не лише як засіб розваги, а й як складний програмний продукт, якість якого визначається технічним виконанням, стабільністю роботи, графічним і звуковим оформленням, геймплейними механіками, сюжетною складовою, рівнем оптимізації, підтримкою багатокористувацьких режимів та загальним користувацьким досвідом. У зв’язку з постійним зростанням кількості ігрових продуктів виникає потреба у зручних програмних засобах, які дають змогу систематизувати інформацію про відеоігри, здійснювати їх оцінювання, аналізувати рейтинги та формувати обґрунтовані висновки щодо популярності й якості окремих проєктів.

Актуальність теми зумовлена тим, що користувачі часто стикаються з проблемою вибору відеоігор серед великої кількості доступних продуктів на різних платформах. Опис гри, рекламні матеріали або загальна оцінка у магазині не завжди дають повне уявлення про її якість, переваги та недоліки. Водночас користувацькі рейтинги, відгуки та аналітичні показники є важливим джерелом інформації, оскільки вони відображають реальний досвід гравців. Однак без належної систематизації такі дані можуть бути фрагментарними, суб’єктивними та складними для аналізу. Тому доцільним є створення програмного забезпечення, яке забезпечує збирання, зберігання, оброблення та візуалізацію оцінок користувачів, а також формування узагальнених рейтингів відеоігор.

Особливого значення набуває можливість не лише переглядати інформацію про ігри, а й виконувати її аналітичне опрацювання. Система аналізу та оцінки відеоігор має давати змогу порівнювати ігрові продукти за жанрами, платформами, роком випуску, середньою оцінкою, кількістю відгуків, рівнем

популярності та іншими характеристиками. Таке програмне забезпечення може бути корисним для звичайних користувачів, які обирають гру для придбання або проходження, для адміністраторів ігрових каталогів, а також для аналітиків, які досліджують динаміку інтересу до певних жанрів або продуктів.

Наявні ігрові платформи та онлайн-сервіси зазвичай мають власні механізми оцінювання, проте вони не завжди забезпечують гнучкий аналіз даних, зручне порівняння відеоігор і прозору структуру формування рейтингу. Часто оцінки користувачів подаються лише у вигляді середнього бала або відсотка позитивних відгуків, без детального поділу за критеріями. Це обмежує можливості глибшого аналізу якості ігрового продукту. Розроблення окремої інформаційної системи дає змогу реалізувати власну модель оцінювання, передбачити зручний інтерфейс користувача, сформувати базу даних відеоігор і забезпечити засоби фільтрації, сортування, пошуку та побудови аналітичних звітів.

Метою кваліфікаційної роботи є проєктування та розроблення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів, яке забезпечує ведення каталогу відеоігор, збирання користувацьких оцінок, оброблення рейтингів, аналіз відгуків і формування узагальненої інформації для підтримки вибору ігрових продуктів.

Для досягнення поставленої мети необхідно виконати такі **завдання:**

1. проаналізувати предметну область систем аналізу та оцінювання відеоігор;
2. розглянути існуючі програмні рішення, що використовуються для ведення ігрових каталогів, рейтингів і користувацьких відгуків;
3. визначити функціональні та нефункціональні вимоги до розроблюваного програмного забезпечення;
4. побудувати UML-моделі, які відображають основні сценарії роботи користувачів і взаємодію компонентів системи;
5. розробити логічну та фізичну модель бази даних для зберігання інформації про відеоігри, користувачів, оцінки, відгуки та жанри;

6. обґрунтувати вибір програмних засобів і технологій реалізації системи;

7. реалізувати основні програмні модулі системи аналізу та оцінки відеоігор;

8. провести тестування програмного забезпечення та оцінити коректність роботи основних функцій.

Об’єктом дослідження є процес аналізу, систематизації та оцінювання відеоігор на основі користувацьких рейтингів і відгуків.

Предметом дослідження є методи, моделі та програмні засоби створення інформаційної системи для зберігання, оброблення, аналізу й відображення даних про відеоігри та їх оцінки користувачами.

У роботі застосовуються методи системного аналізу для дослідження предметної області та визначення вимог до програмного забезпечення; методи об’єктно-орієнтованого проєктування для побудови структури програмної системи; UML-моделювання для опису функціональних сценаріїв і взаємодії компонентів; методи проєктування баз даних для формування інформаційної моделі системи; методи тестування програмного забезпечення для перевірки працездатності, стабільності та зручності використання розробленого застосунку.

Практичне значення роботи полягає у створенні програмного забезпечення, яке може використовуватися для організації електронного каталогу відеоігор, накопичення користувацьких оцінок, формування рейтингів і проведення аналітичного порівняння ігрових продуктів. Розроблена система дає змогу користувачам швидко знаходити потрібні відеоігри, переглядати їх характеристики, залишати оцінки й відгуки, а також аналізувати загальний рівень популярності та якості ігор. Для адміністратора система забезпечує можливість керування каталогом, редагування даних, контролю користувацького контенту та перегляду статистичних показників.

Науково-практична цінність розробки полягає в поєднанні функцій каталогу відеоігор, системи користувацького оцінювання та аналітичного

модуля в межах єдиного програмного середовища. Такий підхід дозволяє не лише зберігати інформацію про ігрові продукти, а й здійснювати її подальше опрацювання для формування рейтингів, порівняльних висновків і звітів. Завдяки цьому система може бути використана як основа для подальшого розширення, зокрема додавання рекомендаційного модуля, аналізу текстових відгуків, персоналізованих добірок і розширеної статистики за жанрами, платформами та активністю користувачів.

Розроблюване програмне забезпечення орієнтоване на підвищення зручності роботи з інформацією про відеоігри та забезпечення об’єктивнішого підходу до їх оцінювання. Використання рейтингової моделі, бази користувацьких відгуків і засобів аналітики створює умови для більш повного представлення якості ігрового продукту. Це особливо важливо в умовах великої кількості цифрового контенту, коли користувачеві потрібні не лише рекламні описи, а й структурована оцінка, сформована на основі реального досвіду інших гравців.

Кваліфікаційна робота складається зі вступу, основних розділів, висновків, списку використаних джерел і додатків. У першому розділі розглядається предметна область, аналізуються існуючі рішення та формулюються вимоги до програмного забезпечення. У другому розділі виконується проєктування інформаційного та програмного забезпечення системи, зокрема розробляються моделі даних, діаграми класів, компонентів і пакетів. У третьому розділі обґрунтовується вибір технологій, описується архітектура системи та реалізація основних програмних модулів. У четвертому розділі проводиться тестування системи, аналізуються результати перевірки та оцінюється ефективність розробленого програмного забезпечення.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Предметна область оцінювання відеоігор за користувацькими рейтингами та показниками цифрової активності

Сучасна індустрія відеоігор є однією з найбільш динамічних сфер цифрового ринку, у якій щороку з’являється значна кількість нових програмних продуктів для різних платформ, жанрів і цільових аудиторій. Відеоігри сьогодні виступають не лише засобом розваги, а й складним цифровим продуктом, якість якого визначається технічним виконанням, стабільністю роботи, графічним оформленням, геймплейними механіками, сюжетною складовою, зручністю інтерфейсу, оптимізацією та загальним користувацьким досвідом. У зв’язку з великою кількістю доступних ігор виникає потреба у програмних засобах, які дають змогу систематизувати інформацію про відеоігри, накопичувати користувацькі оцінки, аналізувати відгуки та формувати рейтинги на основі реальної активності гравців.

Предметна область системи аналізу та оцінки відеоігор охоплює процеси збирання, зберігання, оброблення, аналізу та візуалізації даних про ігрові продукти. Основними джерелами інформації можуть бути локальна база даних системи, користувацькі оцінки, текстові відгуки, відкриті ігрові каталоги, цифрові платформи, тематичні спільноти та сервіси, які надають інформацію про популярність відеоігор. У межах розроблюваного програмного забезпечення такі дані використовуються для формування каталогу ігор, розрахунку середніх рейтингів, визначення популярності за жанрами та платформами, а також підготовки аналітичних звітів.

Відеоігра в межах розроблюваної системи розглядається як основний об’єкт обліку й аналізу. Для кожної гри зберігаються назва, жанр, платформа, рік випуску, розробник, видавець, короткий опис, середній рейтинг, кількість оцінок і відгуків. Такий підхід дає змогу не лише переглядати загальну інформацію про

гру, а й порівнювати її з іншими ігровими продуктами за визначеними критеріями. Користувацький рейтинг при цьому виступає одним із головних показників, оскільки він формується на основі оцінок реальних користувачів і відображає їхній практичний досвід взаємодії з грою.

Основні складові предметної області системи аналізу та оцінки відеоігор наведено в таблиці 1.1.

Таблиця 1.1

Основні складові предметної області системи аналізу та оцінки відеоігор

Складова предметної області	Призначення в системі	Основні дані
Відеоігра	Основний об’єкт обліку, аналізу та оцінювання	Назва, жанр, платформа, рік випуску, розробник, видавець, опис
Користувач	Учасник системи, який переглядає ігри, залишає оцінки та відгуки	Логін, роль, електронна пошта, історія оцінювання
Рейтинг	Узагальнений числовий показник оцінки гри користувачами	Середня оцінка, кількість оцінок, динаміка зміни рейтингу
Відгук	Текстове повідомлення користувача щодо переваг і недоліків гри	Автор, текст відгуку, дата створення, прив’язка до гри
Жанр	Класифікаційна ознака відеоігри	Назва жанру, опис, перелік пов’язаних ігор
Платформа	Середовище, для якого доступна відеоігра	PC, PlayStation, Xbox, Nintendo Switch, мобільні платформи
Джерело даних	Канал отримання інформації про ігри та активність користувачів	API, локальна база даних, відкриті каталоги, користувацькі записи
Аналітичний модуль	Компонент оброблення оцінок, відгуків і статистичних показників	Середній рейтинг, кількість відгуків, популярність за жанрами
Звіт	Узагальнене подання результатів аналізу	Найрейтинговіші ігри, статистика оцінок, активність користувачів
Адміністратор	Користувач із розширеними правами керування системою	Керування іграми, користувачами, відгуками та довідниками

Як видно з таблиці 1.1, предметна область системи охоплює не лише зберігання інформації про відеоігри, а й роботу з користувацькими оцінками,

відгуками, жанрами, платформами та аналітичними результатами. Це створює основу для подальшого проєктування бази даних, побудови UML-моделей і розроблення програмних модулів системи.

Узагальнений процес аналізу популярності відеоігор передбачає проходження даних через декілька послідовних етапів. Спочатку інформація надходить із джерел даних, до яких можуть належати Steam API, Twitch, Reddit, локальна база даних або внутрішні записи користувачів. Далі виконується збирання та нормалізація даних, тобто приведення їх до єдиної структури, перевірка коректності, усунення дублювання та підготовка до зберігання. Після цього інформація розміщується у сховищі даних, де вона стає доступною для подальшого аналізу.

Узагальнену схему процесу аналізу популярності відеоігор наведено на рисунку 1.1.

Рис. 1.1. Узагальнена схема процесу аналізу популярності відеоігор за користувацькими рейтингами та цифровою активністю

Відповідно до рисунка 1.1, центральним елементом оброблення є аналітичне ядро системи. Воно виконує розрахунок основних показників, зокрема середнього рейтингу, кількості оцінок, частки позитивних і негативних відгуків, динаміки зміни популярності та активності користувачів. У розширеному варіанті аналітичний модуль може використовувати методи машинного навчання або оброблення природної мови для аналізу текстових відгуків, визначення їх тональності та виявлення загального ставлення користувачів до певної гри. Проте для базової версії системи достатньо реалізувати обчислення рейтингових і статистичних показників, які безпосередньо використовуються у звітах і на інформаційній панелі.

Важливою характеристикою предметної області є розмежування понять якості та популярності відеоігри. Якість гри доцільно оцінювати на основі

середнього рейтингу, змісту відгуків, стабільності роботи, зручності ігрового процесу та загального задоволення користувачів. Популярність, у свою чергу, може визначатися кількістю оцінок, частотою згадок, кількістю переглядів, активністю обговорень і динамікою додавання нових відгуків. Тому система має забезпечувати не лише загальну оцінку гри, а й окреме відображення показників користувацької активності.

Користувачами розроблюваної системи можуть бути гравці, адміністратори та аналітики. Звичайний користувач переглядає каталог відеоігор, використовує пошук і фільтри, ознайомлюється з рейтингами, залишає власні оцінки та текстові відгуки. Адміністратор відповідає за додавання й редагування ігор, керування жанрами, платформами, користувачами та модерацію відгуків. Аналітик або адміністратор із розширеними правами може переглядати статистику, формувати звіти, аналізувати динаміку рейтингів і визначати найбільш популярні ігри за окремими критеріями.

Оцінювання відеоігор має суб’єктивний характер, оскільки користувачі можуть по-різному сприймати один і той самий ігровий продукт залежно від власних жанрових уподобань, досвіду, технічних характеристик пристрою та очікувань від гри. Саме тому система не повинна обмежуватися лише розрахунком середньої оцінки. Доцільно враховувати кількість оцінок, розподіл балів, текстові відгуки, дату створення оцінки та загальну активність користувачів. Це забезпечує більш повне представлення якості відеоігри та зменшує вплив випадкових або поодиноких оцінок.

Для забезпечення коректної роботи системи необхідно передбачити структуроване зберігання даних. Основними інформаційними сутностями є користувач, відеоігра, жанр, платформа, оцінка, відгук, рейтинг і звіт. Між ними існують логічні зв’язки: один користувач може залишати багато оцінок і відгуків, одна відеоігра може мати багато оцінок, належати до певного жанру та бути доступною на декількох платформах, а рейтинг формується на основі сукупності користувацьких оцінок. Така структура забезпечує можливість пошуку, фільтрації, сортування, порівняння ігор і формування аналітичних результатів.

Результати аналізу доцільно подавати у вигляді інформаційної панелі та звітів. На інформаційній панелі можуть відображатися найрейтинговіші ігри, найпопулярніші жанри, кількість оцінок за певний період, ігри з найбільшою кількістю відгуків і загальна активність користувачів. Звітний модуль може формувати підсумкові таблиці або візуалізації, які використовуються для оцінювання стану каталогу, аналізу інтересу користувачів і порівняння ігрових продуктів між собою.

Отже, предметна область програмного забезпечення системи аналізу та оцінки відеоігор охоплює комплекс процесів, пов’язаних із веденням каталогу ігор, збиранням користувацьких оцінок, обробленням відгуків, формуванням рейтингів і візуалізацією аналітичних результатів. Розроблення такої системи є доцільним, оскільки вона забезпечує впорядковане зберігання інформації про відеоігри, підвищує зручність порівняння ігрових продуктів і створює основу для об’єктивнішого аналізу їх популярності та якості на основі користувацької активності.

1.2 Аналіз існуючих рішень для оцінювання відеоігор та відстеження користувацької активності

Для обґрунтування доцільності розроблення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів необхідно розглянути існуючі рішення, які використовуються для збирання, відображення та аналізу інформації про ігрові продукти. Сучасні сервіси у цій сфері можна умовно поділити на кілька груп: бази даних і каталогізатори відеоігор, сервіси статистики активності гравців, платформи аналізу стримінгової популярності, аналітичні панелі для дослідження ринку та інструменти оцінювання пошукового інтересу. Кожна з таких систем розв’язує окрему частину задачі, однак не завжди забезпечує комплексне поєднання каталогу ігор, користувацьких рейтингів, відгуків, аналітики популярності та звітності в межах одного програмного середовища.

Одним із найбільш відомих сервісів для аналізу ігор у середовищі Steam є SteamDB. Його інтерфейс наведено на рисунку 1.2. Сервіс орієнтований на збирання та відображення даних про ігри, що розміщені у Steam, зокрема інформації про кількість активних гравців, пікові значення онлайну, популярні релізи, ціни, знижки, історію оновлень і тенденції за окремими ігровими продуктами. Перевагою SteamDB є значний обсяг даних, зручне представлення статистики та можливість швидко оцінити поточну популярність гри серед користувачів платформи Steam.

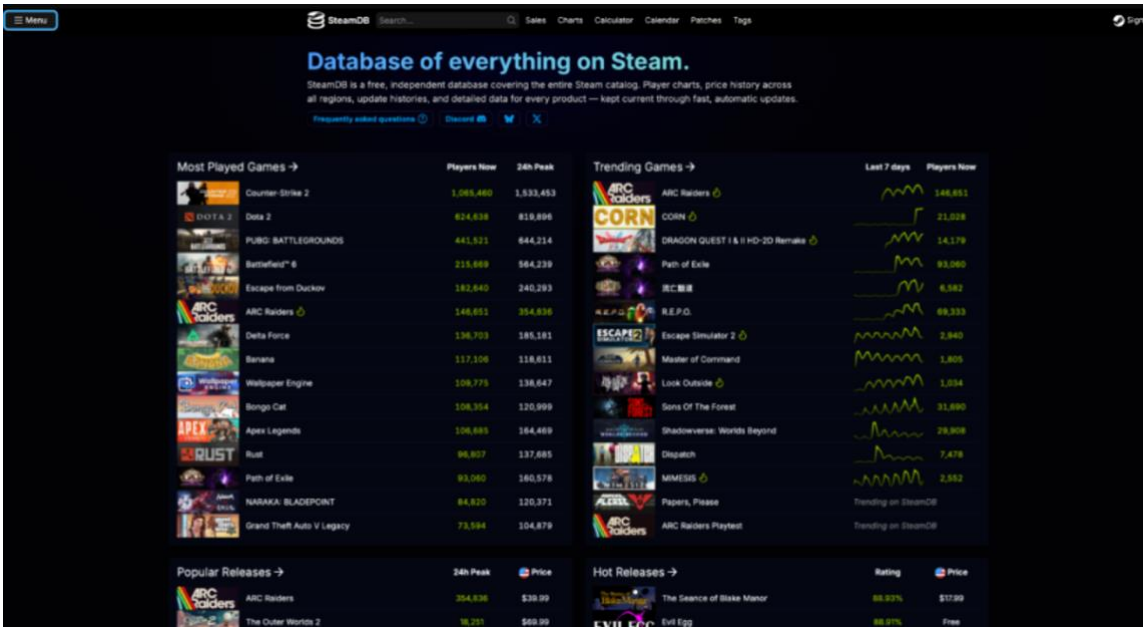

Рис. 1.2. Інтерфейс сервісу SteamDB для перегляду статистики ігор платформи Steam

Разом із тим SteamDB переважно орієнтований на статистику платформи Steam і не є універсальною системою користувацького оцінювання. У ньому основний акцент зроблено на технічних і ринкових показниках: кількості гравців, динаміці активності, цінах, оновленнях і релізах. Користувацькі відгуки та оцінки не виступають центральним елементом аналізу, а можливості формування власних рейтингів, розширеної модерації відгуків або локального аналітичного звіту є обмеженими. Тому SteamDB доцільно розглядати як потужне джерело статистики, але не як повноцінну систему аналізу відеоігор за користувацькими рейтингами.

Іншим напрямом є сервіси аналізу стримінгової активності, зокрема Twitch Games Statistics. Приклад такого інтерфейсу наведено на рисунку 1.3. Подібні рішення дають змогу аналізувати популярність ігор за кількістю переглядів, активних трансляцій, каналів, часу перегляду та часткою аудиторії. Такі показники є важливими для визначення поточного інтересу до гри, оскільки стримінгові платформи часто відображають реальну динаміку популярності ігрового продукту серед глядачів і гравців.

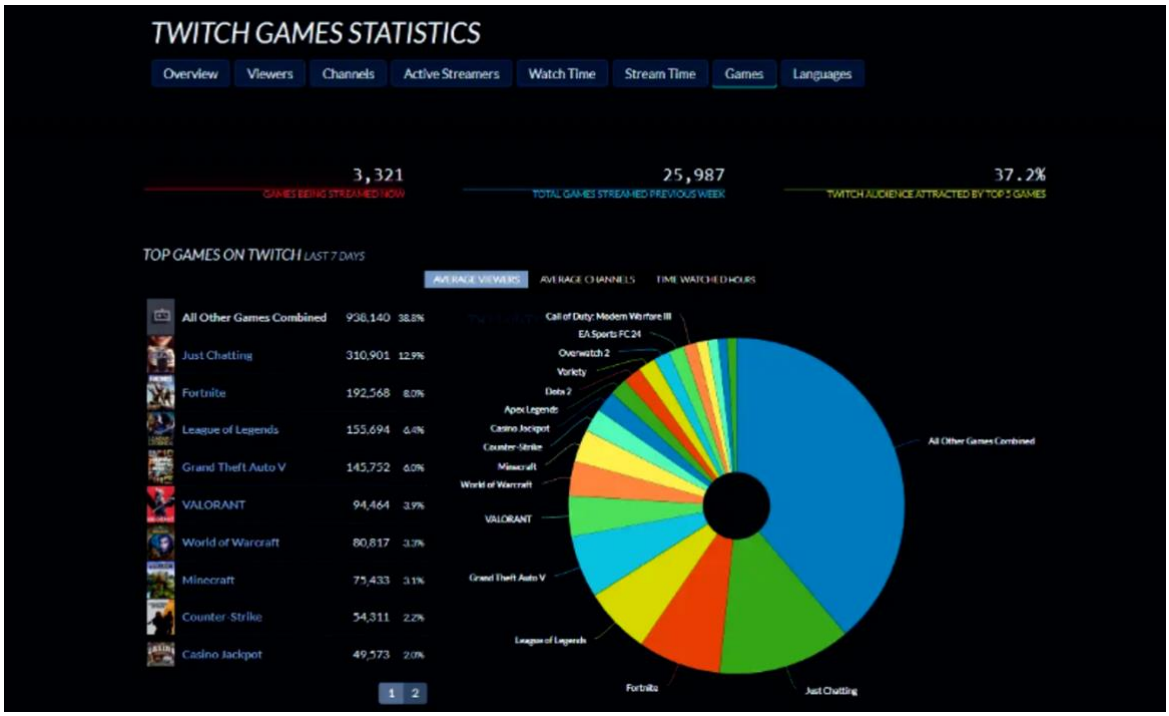

Рис. 1.3. Приклад сервісу аналізу статистики ігор на платформі Twitch

Перевагою таких сервісів є наочна візуалізація даних, можливість порівняння ігор за активністю переглядів і визначення найбільш популярних продуктів за певний період. Однак стримінгова популярність не завжди повністю відображає якість гри. Деякі ігри можуть мати високі перегляди через кіберспортивні події, рекламні кампанії або активність відомих стримерів, але водночас отримувати неоднозначні оцінки від звичайних користувачів. Через це дані Twitch доцільно використовувати як додатковий показник популярності, а не як основний критерій оцінювання якості відеоігри.

Окрему групу становлять аналітичні панелі, які використовуються для візуалізації маркетингових і користувацьких показників. Приклад такої панелі

наведено на рисунку 1.4. У подібних системах зазвичай відображаються ключові метрики, графіки, діаграми, порівняльні показники та звітна інформація. Для сфери відеоігор такий підхід є корисним, оскільки дає змогу в одному інтерфейсі переглядати кількість користувачів, динаміку оцінок, активність за період, частку позитивних відгуків і популярність окремих категорій.

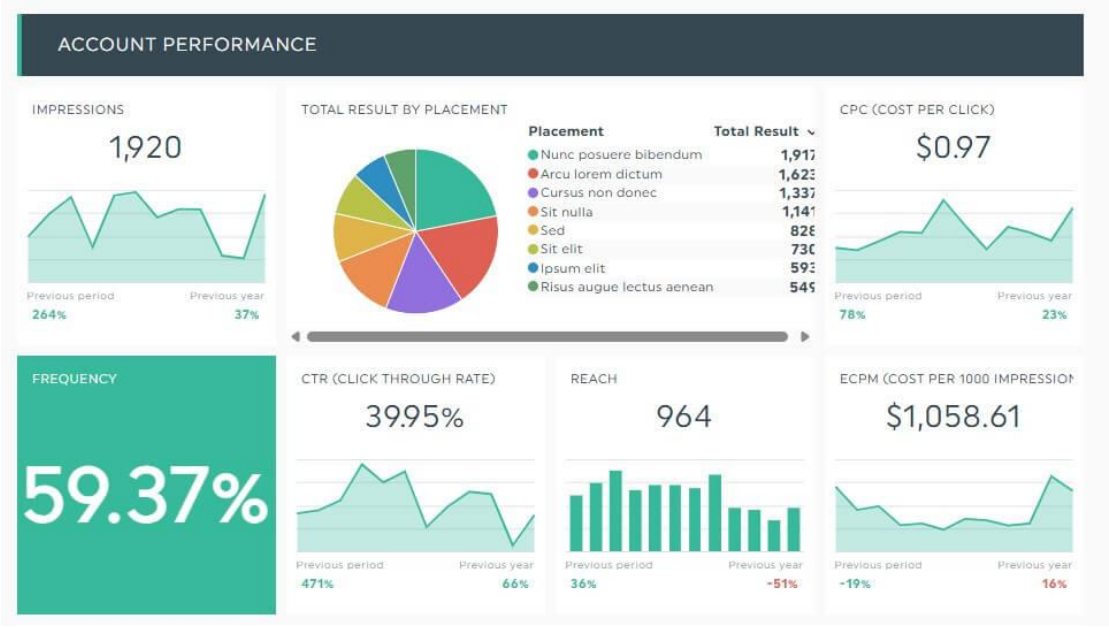

Рис. 1.4 - Приклад аналітичної панелі для візуалізації показників користувацької активності

Основною перевагою аналітичних панелей є компактне та наочне подання великого обсягу даних. Вони спрощують сприйняття статистики та дозволяють швидко оцінити стан об’єкта аналізу. Однак універсальні панелі зазвичай не враховують специфіку відеоігор, зокрема жанри, платформи, релізи, користувацькі оцінки, текстові відгуки та структуру ігрового каталогу. Тому для розроблюваної системи доцільно використати ідею дашборду, але адаптувати її саме до предметної області аналізу відеоігор.

Більш спеціалізованими є платформи ринкової аналітики ігор, які відображають дані про окремі ігрові продукти, їх аудиторію, платформи, показники активності та ринкове позиціонування. Приклад такого рішення наведено на рисунку 1.5. У таких системах можуть відображатися дані про щоденну або щомісячну активну аудиторію, середній час гри, охоплення різних

платформ, схожі ігри, видавців і розробників. Такі інструменти є корисними для професійного аналізу ігрового ринку та прийняття рішень у сфері розроблення, просування й підтримки відеоігор.

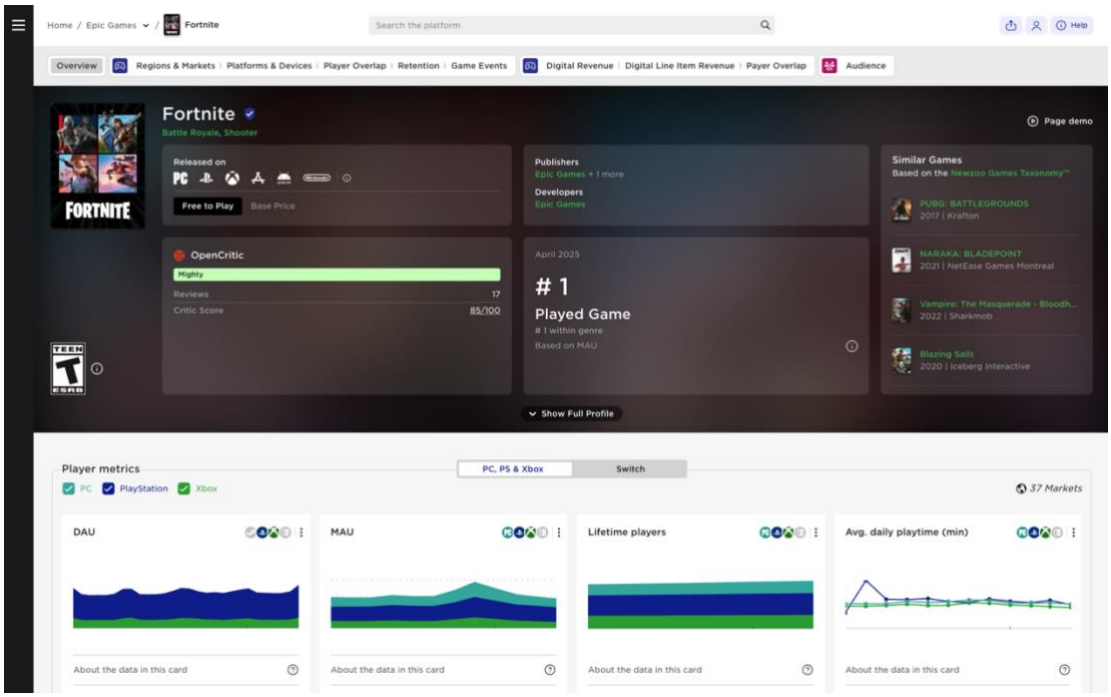

Рис. 1.5 - Приклад платформи ринкової аналітики відеоігор

Недоліком таких рішень є їх складність, комерційна спрямованість і орієнтація на професійних аналітиків або компанії. Звичайному користувачеві не завжди потрібен великий набір ринкових показників, оскільки основною потребою є швидке порівняння ігор, перегляд рейтингів, відгуків і загальної оцінки. Крім того, частина професійних сервісів має обмежений доступ до даних або потребує платної підписки. Це знижує доступність таких інструментів для навчальних, дослідницьких або локальних програмних систем.

Ще одним корисним інструментом для аналізу інтересу до ігор є Google Trends, інтерфейс якого наведено на рисунку 1.6. За допомогою цього сервісу можна оцінювати динаміку пошукового інтересу до певної гри, жанру, платформи або пов’язаного поняття. Такий підхід дає змогу визначити, як змінюється увага користувачів до ігрового продукту в часі, порівняти декілька ігор між собою та виявити регіональні особливості популярності.

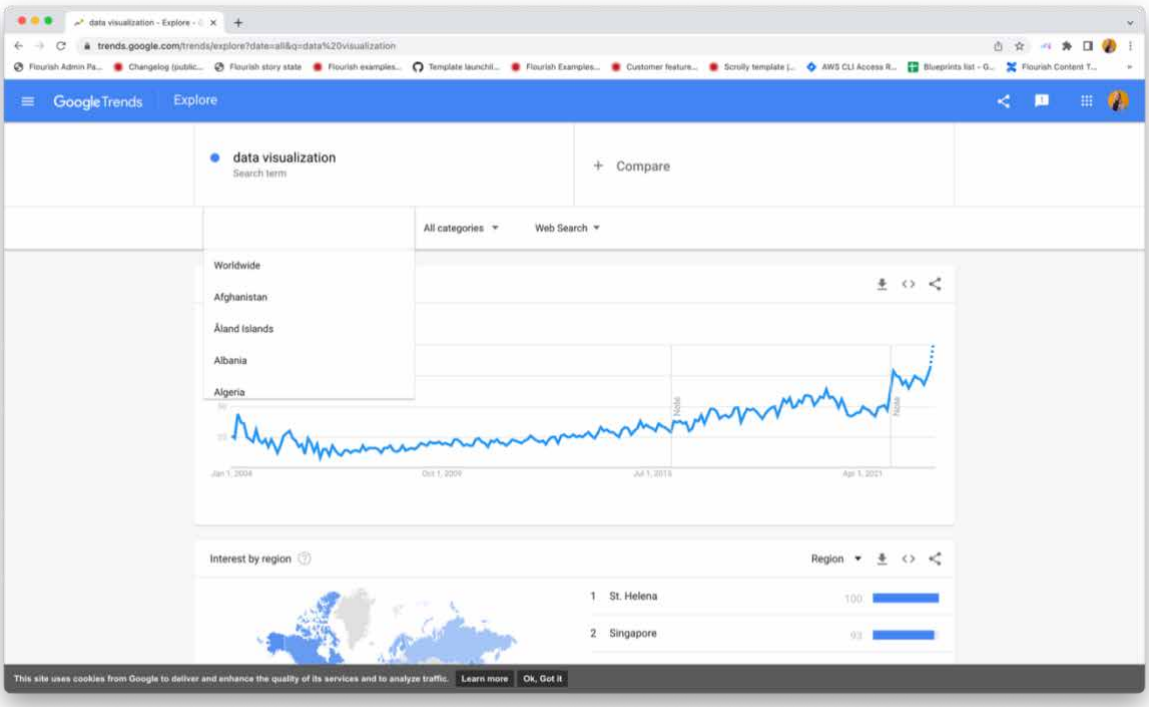

Рис.1.6 - Приклад використання Google Trends для аналізу пошукового інтересу

Перевагою Google Trends є доступність, простота використання та можливість аналізу динаміки популярності за регіонами й періодами. Водночас цей сервіс не містить повної інформації про самі відеоігри, не зберігає користувацькі рейтинги, не надає текстових відгуків і не дозволяє формувати внутрішній каталог ігрових продуктів. Тому Google Trends може бути використаний лише як допоміжне джерело для оцінки зовнішнього інтересу, але не як основна система аналізу та оцінювання відеоігор.

Порівняльну характеристику розглянутих рішень наведено в таблиці 1.2.

Таблиця 1.2

Порівняльна характеристика існуючих рішень для аналізу відеоігор

Рішення	Основне призначення	Переваги	Недоліки
SteamDB	Перегляд статистики ігор платформи Steam	Велика база ігор, дані про онлайн, ціни, релізи та оновлення	Орієнтація лише на Steam, обмежена робота з власними рейтингами та відгуками
Twitch Games Statistics	Аналіз популярності ігор на основі стримінгової активності	Відображення переглядів, каналів, активних трансляцій і динаміки аудиторії	Стримінгова популярність не завжди відповідає якості гри

Продовження таблиці 1.2

Аналітична панель показників	Візуалізація статистичних і маркетингових метрик	Зручні графіки, діаграми, компактне подання ключових показників	Не враховує специфіку ігрового каталогу та користувацьких оцінок
Платформа ринкової аналітики ігор	Професійний аналіз аудиторії, платформ і ринкових показників	Детальна статистика, порівняння ігор, аналіз аудиторії та ринку	Складність використання, комерційна спрямованість, обмежений доступ
Google Trends	Аналіз пошукового інтересу до ігор і пов'язаних запитів	Динаміка інтересу в часі, регіональний аналіз, простота використання	Відсутність рейтингової моделі, відгуків і внутрішнього каталогу ігор
Розроблювана система	Аналіз та оцінка відеоігор за рейтингами користувачів	Поєднання каталогу, оцінок, відгуків, рейтингів, фільтрів і звітності	Потребує наповнення бази даних і подальшого розширення джерел інформації

Як видно з таблиці 1.2, кожне з розглянутих рішень має власну сферу застосування та сильні сторони. SteamDB ефективно відображає статистику Steam, сервіси Twitch-статистики показують стримінгову популярність, аналітичні панелі забезпечують візуалізацію показників, професійні платформи дають змогу досліджувати ринок, а Google Trends відображає пошуковий інтерес. Проте жодне з цих рішень у повному обсязі не поєднує каталог відеоігор, користувацькі рейтинги, текстові відгуки, фільтрацію, локальне адміністрування та формування звітів у межах простої програмної системи.

На основі проведеного аналізу можна визначити основні вимоги до розроблюваного програмного забезпечення. Система має забезпечувати ведення каталогу відеоігор, зберігання інформації про жанри та платформи, реєстрацію користувачів, додавання оцінок і відгуків, автоматичний розрахунок середнього рейтингу, пошук і фільтрацію ігор, відображення статистики та формування звітної інформації. Важливо також передбачити роль адміністратора, який зможе керувати даними, контролювати коректність відгуків і підтримувати актуальність каталогу.

Отже, аналіз існуючих рішень показав, що сучасні сервіси надають значні можливості для перегляду статистики, оцінки популярності та візуалізації

ігрових показників, однак мають певні обмеження щодо комплексного користувацького оцінювання відеоігор. Саме тому розроблення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів є доцільним. Запропонована система має поєднати основні переваги розглянутих рішень, але бути орієнтованою на просте ведення каталогу ігор, накопичення оцінок, аналіз відгуків, формування рейтингів і подання результатів у зручному для користувача вигляді.

1.3 Моделювання інформаційної системи

Моделювання предметної області є важливим етапом проєктування програмного забезпечення, оскільки дає змогу формалізувати основні функції системи, визначити ролі користувачів, описати послідовність взаємодії між учасниками та відобразити загальну логіку виконання бізнес-процесів. Для програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів доцільно використати UML-моделі, які відображають функціональні можливості системи на рівні прецедентів, сценаріїв взаємодії та алгоритмічної послідовності дій.

У межах предметної області виділено три основні ролі користувачів: користувач, аналітик та адміністратор. Користувач взаємодіє із системою для реєстрації, входу, пошуку відеоігор, перегляду рейтингів і відгуків, а також оцінювання ігрових продуктів. Аналітик працює з аналітичними функціями системи, формує звіти, аналізує тренди популярності та за потреби експортує результати. Адміністратор забезпечує підтримку інформаційного наповнення системи, керує каталогом відеоігор і обліковими записами користувачів.

Для узагальнення функціональних можливостей системи побудовано діаграму прецедентів, наведену на рисунку 1.7.

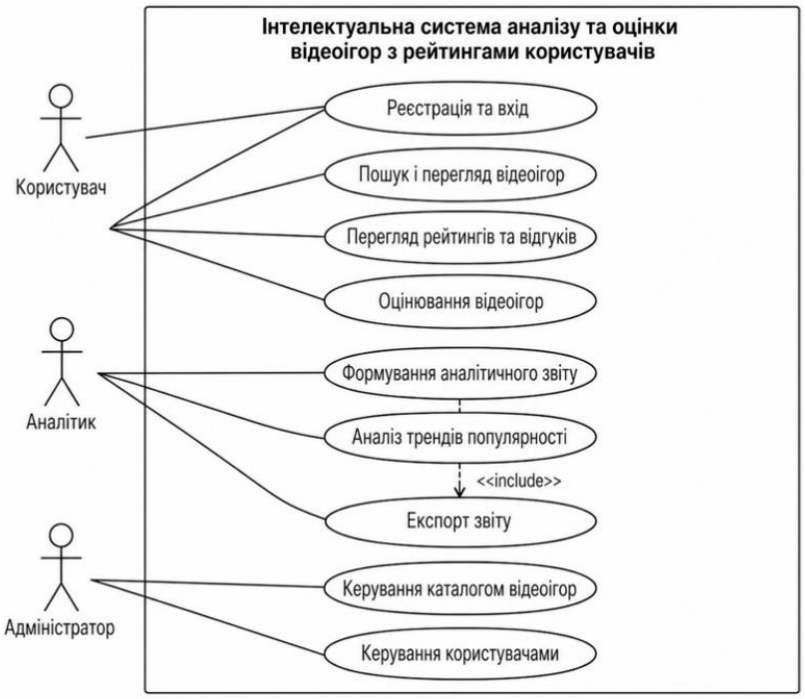

Рис. 1.7.Діаграма прецедентів системи аналізу та оцінки відеоігор з рейтингами користувачів

Як видно з рисунка 1.7, діаграма прецедентів відображає основні функції програмного забезпечення та розподіл доступу між різними ролями. Користувач має доступ до базових функцій перегляду й оцінювання відеоігор, аналітик працює з функціями формування звітів та аналізу популярності, а адміністратор виконує керування каталогом і користувачами. Такий розподіл ролей забезпечує логічне відокремлення звичайних операцій користувача від адміністративних та аналітичних функцій.

Центральною частиною системи є каталог відеоігор, який об’єднує інформацію про назви ігор, жанри, платформи, описи, рейтинги та відгуки. Користувач, працюючи з каталогом, може здійснювати пошук, переглядати детальну інформацію про гру, аналізувати оцінки інших користувачів і залишати власну оцінку. Після збереження оцінки система оновлює рейтингові показники, що надалі використовуються у загальній статистиці та звітності.

Аналітична частина системи призначена для формування узагальнених висновків щодо популярності відеоігор. До її функцій належать створення аналітичного звіту, аналіз трендів популярності та експорт результатів. На

діаграмі прецедентів зв’язок між аналізом трендів і експортом звіту подано як включення, оскільки експорт є додатковою дією, що виконується після отримання результатів аналізу. Такий підхід дає змогу розглядати формування звіту як окремий функціональний процес, який може завершуватися переглядом результатів або їх збереженням у файлі.

Адміністративний блок забезпечує коректність і актуальність даних у системі. Адміністратор може додавати, редагувати та видаляти записи про відеоігри, змінювати довідкову інформацію, керувати користувачами та підтримувати порядок у каталозі. Наявність адміністративної ролі є необхідною, оскільки якість аналітичних результатів залежить від повноти, правильності та структурованості даних.

Для деталізації взаємодії учасників із програмною системою побудовано діаграму послідовності, наведену на рисунку 1.8.

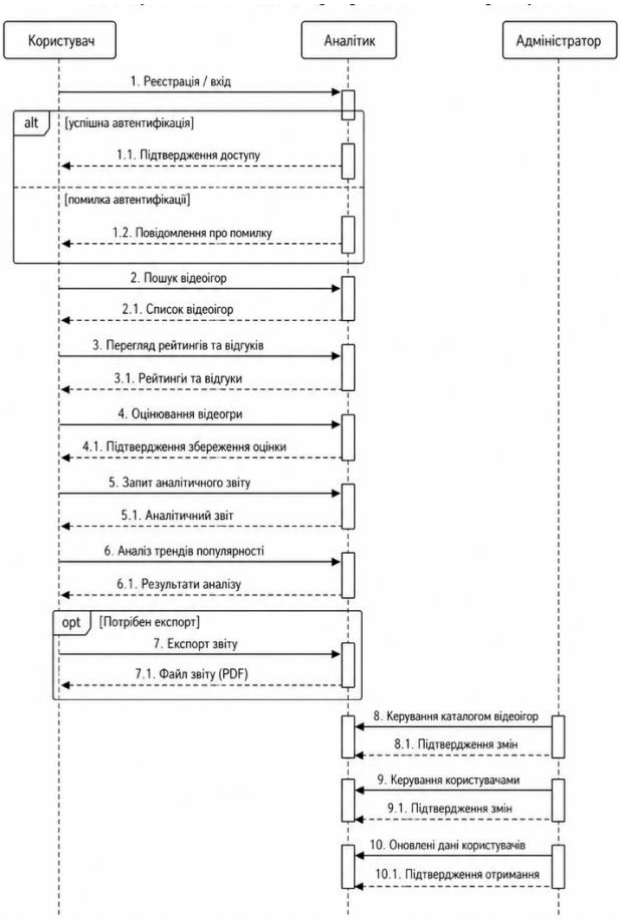

Рис. 1.8. Діаграма послідовності взаємодії користувача, аналітика та адміністратора із системою

Діаграма послідовності на рисунку 1.8 показує порядок виконання основних операцій у системі. Спочатку користувач проходить реєстрацію або вхід. У межах цього сценарію передбачено альтернативні варіанти: успішна автентифікація з наданням доступу або помилка автентифікації з виведенням відповідного повідомлення. Після успішного входу користувач отримує можливість виконувати пошук відеоігор, переглядати рейтинги та відгуки, а також залишати власну оцінку.

Наступний блок взаємодії пов’язаний з аналітичними функціями. Користувач із правами аналітика формує запит на створення звіту, отримує аналітичні результати, переглядає тренди популярності й за потреби експортує звіт у файл. У діаграмі цей процес подано через опціональний фрагмент, оскільки експорт не є обов’язковим для кожного сценарію аналізу. Така структура відповідає логіці роботи аналітичного модуля: спочатку система формує дані, а потім користувач вирішує, чи потрібно зберігати їх у зовнішньому форматі.

Окремо на діаграмі послідовності відображено адміністративні операції. Адміністратор виконує керування каталогом відеоігор і користувачами, після чого отримує підтвердження збереження змін. Така взаємодія демонструє, що адміністративні функції не змішуються з користувацьким оцінюванням і аналітичним переглядом, а виконуються в окремому контурі керування даними.

Для опису загальної логіки виконання процесів у системі побудовано діаграму активності, наведену на рисунку 1.9.

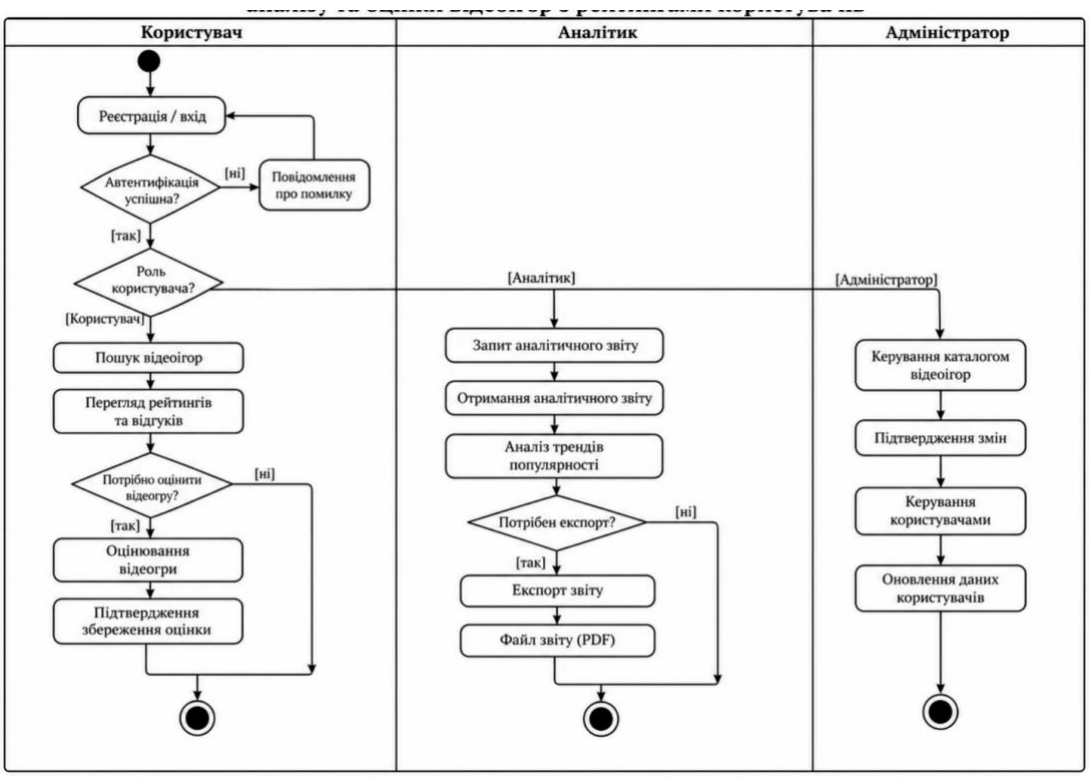

Рис. 1.9. Діаграма активності системи аналізу та оцінки відеоігор

Діаграма активності на рисунку 1.9 відображає послідовність дій у межах трьох ролей: користувача, аналітика та адміністратора. Робота починається з реєстрації або входу, після чого виконується перевірка автентифікації. У разі помилки користувач отримує повідомлення, а в разі успішної перевірки система визначає роль облікового запису та спрямовує подальші дії відповідно до прав доступу.

Для звичайного користувача основний сценарій передбачає пошук відеоігор, перегляд рейтингів і відгуків, а також можливість оцінювання відеоігри. Якщо користувач вирішує залишити оцінку, система зберігає її та підтверджує виконання операції. Якщо оцінювання не потрібне, сценарій завершується після перегляду інформації. Такий процес відповідає типовій поведінці користувача, який використовує систему для вибору ігрового продукту та ознайомлення з думками інших гравців.

Для аналітика діаграма активності передбачає запит аналітичного звіту, отримання результатів, аналіз трендів популярності та можливий експорт звіту. Цей сценарій відображає роботу з узагальненими даними, які формуються на

основі користувацьких оцінок, відгуків і показників активності. Експорт звіту подано як додаткову дію, оскільки аналітик може обмежитися переглядом результатів у системі або зберегти їх у вигляді окремого файлу.

Для адміністратора основний сценарій складається з керування каталогом відеоігор, підтвердження змін, керування користувачами та оновлення користувацьких даних. Цей процес забезпечує підтримку актуальності інформаційної бази та контроль за правильністю даних. Завдяки такому підходу система зберігає структурованість каталогу, коректність облікових записів і стабільність роботи основних функцій.

Узагальнену характеристику побудованих UML-моделей наведено в таблиці 1.3.

Таблиця 1.3

Характеристика UML-моделей предметної області

UML-модель	Призначення	Основні елементи
Діаграма прецедентів	Визначення функціональних можливостей системи та ролей користувачів	Користувач, аналітик, адміністратор, реєстрація, пошук, оцінювання, звіти, керування каталогом
Діаграма послідовності	Опис порядку взаємодії учасників із системою під час виконання основних сценаріїв	Вхід, автентифікація, пошук, перегляд рейтингів, оцінювання, формування та експорт звіту
Діаграма активності	Відображення загальної логіки виконання процесів залежно від ролі користувача	Перевірка входу, вибір ролі, користувацькі дії, аналітичні дії, адміністративне керування

Як видно з таблиці 1.3, кожна з побудованих UML-моделей описує предметну область з окремого погляду. Діаграма прецедентів визначає функції та ролі, діаграма послідовності показує порядок інформаційної взаємодії, а діаграма активності відображає загальну логіку виконання процесів. У сукупності ці моделі формують основу для подальшого проєктування структури програмного забезпечення, бази даних, інтерфейсу користувача та алгоритмів оброблення інформації.

Отже, моделювання предметної області системи аналізу та оцінки відеоігор дало змогу визначити основні ролі користувачів, функціональні можливості системи та послідовність виконання ключових процесів. Побудовані UML-діаграми підтверджують, що розроблюване програмне забезпечення має підтримувати три основні напрями роботи: користувацьке оцінювання відеоігор, аналітичне опрацювання рейтингів і адміністративне керування даними. Отримані моделі можуть бути використані як вихідна основа для формування вимог, проєктування бази даних і подальшої реалізації програмних модулів системи.

1.4 Визначення вимог системи аналізу ігор

Визначення вимог є одним із ключових етапів розроблення програмного забезпечення, оскільки саме на цьому етапі встановлюються основні функції майбутньої системи, умови її роботи, обмеження, правила доступу до даних і критерії якості. Для програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів вимоги формуються з урахуванням особливостей предметної області, результатів аналізу існуючих рішень і побудованих UML-моделей.

Розроблювана система орієнтована на роботу з каталогом відеоігор, користувацькими оцінками, текстовими відгуками, рейтинговими показниками та аналітичними звітами. Вона має забезпечувати зручну взаємодію звичайного користувача, аналітика й адміністратора, а також підтримувати структуроване зберігання даних про ігри, жанри, платформи, користувачів і результати оцінювання. Відповідно до цього вимоги до системи доцільно поділити на функціональні, нефункціональні, технічні та безпекові.

Функціональні вимоги визначають перелік основних можливостей, які реалізуються в програмному забезпеченні. Вони безпосередньо пов’язані зі сценаріями, описаними в діаграмах прецедентів, послідовності та активності. Основні функціональні вимоги до системи наведено в таблиці 1.4.

Таблиця 1.4

Функціональні вимоги до системи аналізу та оцінки відеоігор

Позначення	Вимога	Опис реалізації
F1	Реєстрація та вхід користувача	Передбачено створення облікового запису, вхід до системи та перевірку коректності введених даних
F2	Розмежування ролей користувачів	Реалізується робота ролей користувача, аналітика й адміністратора з різним рівнем доступу до функцій
F3	Ведення каталогу відеоігор	Забезпечується додавання, редагування, видалення та перегляд записів про відеоігри
F4	Пошук і фільтрація відеоігор	Передбачено пошук за назвою та фільтрацію за жанром, платформою, роком випуску й рейтингом
F5	Перегляд інформації про відеоігру	Користувач отримує доступ до опису гри, жанру, платформи, розробника, рейтингу та відгуків
F6	Оцінювання відеоігор	Зареєстрований користувач може залишати числову оцінку для вибраної відеоігри
F7	Додавання текстових відгуків	Передбачено можливість залишення коментаря з описом переваг і недоліків гри
F8	Автоматичний розрахунок рейтингу	Після додавання або зміни оцінки виконується оновлення середнього рейтингу відеоігри
F9	Формування аналітичних показників	Аналітичний модуль обчислює кількість оцінок, середній рейтинг, активність користувачів і популярність за жанрами
F10	Формування аналітичного звіту	Аналітик отримує узагальнений звіт про рейтинги, популярність і динаміку оцінювання відеоігор
F11	Експорт звітів	Передбачено збереження результатів аналізу у зовнішньому форматі, зокрема PDF або CSV
F12	Керування користувачами	Адміністратор може переглядати, редагувати та блокувати облікові записи користувачів
F13	Модерація відгуків	Адміністратор має можливість видаляти некоректні або помилкові відгуки
F14	Перегляд інформаційної панелі	Користувачі з відповідними правами можуть переглядати статистику, графіки та ключові показники системи

Як видно з таблиці 1.4, функціональні вимоги охоплюють основні напрями роботи системи: користувацьке оцінювання, адміністрування каталогу та аналітичне опрацювання даних. Особливу увагу приділено рейтинговій моделі, оскільки саме вона є основою для формування висновків про якість і популярність відеоігор. Реалізація цих вимог забезпечує повний цикл роботи з даними: від додавання гри до формування аналітичного звіту.

Нефункціональні вимоги визначають якісні характеристики програмного забезпечення. Вони не описують окремі функції системи, але впливають на

стабільність, зручність, продуктивність і надійність її роботи. Основні нефункціональні вимоги наведено в таблиці 1.5.

Таблиця 1.5

Нефункціональні вимоги до системи аналізу та оцінки відеоігор

Позначення	Вимога	Опис реалізації
NF1	Зручність інтерфейсу	Інтерфейс має бути зрозумілим, логічно структурованим і придатним для швидкого пошуку потрібної інформації
NF2	Швидкодія	Основні операції перегляду каталогу, пошуку, фільтрації та відкриття сторінки гри мають виконуватися без помітних затримок
NF3	Надійність збереження даних	Дані про користувачів, ігри, оцінки та відгуки мають зберігатися без втрати цілісності
NF4	Масштабованість	Структура системи має дозволяти подальше збільшення кількості ігор, користувачів, жанрів і відгуків
NF5	Модульність	Програмні компоненти мають бути поділені на логічні модулі: інтерфейс, оброблення даних, аналітика, звітність і доступ до БД
NF6	Супроводжуваність	Код і структура бази даних мають бути організовані так, щоб спростити подальше оновлення та розширення функціоналу
NF7	Коректність розрахунків	Рейтинги та статистичні показники мають формуватися за єдиними правилами з урахуванням усіх актуальних оцінок
NF8	Інформативність результатів	Аналітичні дані мають подаватися у зрозумілій формі за допомогою таблиць, графіків, рейтингів і звітів
NF9	Стійкість до помилок введення	Система має перевіряти коректність введених значень і повідомляти користувача про помилки
NF10	Актуальність даних	Після зміни оцінок, відгуків або даних каталогу відповідні показники мають оновлюватися в системі

Наведені в таблиці 1.5 нефункціональні вимоги визначають загальну якість роботи програмного забезпечення. Для системи аналізу відеоігор важливо не лише реалізувати набір функцій, а й забезпечити їх зручне та стабільне використання. Особливо важливими є швидкий пошук, правильний розрахунок рейтингів, зрозуміле подання статистики та надійне збереження інформації, оскільки від цього залежить практична цінність системи для користувачів.

Технічні вимоги визначають програмні засоби, структуру зберігання даних і загальні умови функціонування системи. Вони формуються з урахуванням того,

що розроблюване програмне забезпечення має бути придатним для навчального проєкту, тестування та подальшого розширення. Технічні вимоги наведено в таблиці 1.6.

Таблиця 1.6

Технічні вимоги до програмного забезпечення

Позначення	Вимога	Опис реалізації
Т1	Використання бази даних	Для зберігання інформації про ігри, користувачів, оцінки, відгуки та звіти використовується реляційна база даних
Т2	Підтримка структурованих довідників	У базі даних передбачаються довідники жанрів, платформ, ролей користувачів і статусів записів
Т3	Реалізація графічного інтерфейсу	Користувач взаємодіє із системою через вікна, форми, таблиці, кнопки, поля введення та інформаційні панелі
Т4	Локальна або клієнт-серверна архітектура	Система може працювати як локальний застосунок або як веборієнтоване рішення з серверною частиною
Т5	Підтримка експорту даних	Результати аналізу та звітності можуть зберігатися у форматах PDF або CSV
Т6	Оброблення користувацьких дій	Програмні модулі мають реагувати на пошук, фільтрацію, додавання оцінки, створення відгуку та формування звіту
Т7	Збереження журналу змін	Для адміністративних дій доцільно передбачити фіксацію змін у каталозі та користувацьких записах
Т8	Можливість подальшої інтеграції	Архітектура має дозволяти підключення зовнішніх джерел даних, наприклад ігрових API або відкритих каталогів
Т9	Резервне копіювання даних	Для зниження ризику втрати інформації доцільно передбачити створення резервної копії бази даних
Т10	Підтримка аналітичних запитів	Структура БД має забезпечувати виконання вибірок за рейтингом, жанром, платформою, датою та активністю користувачів

Як показано в таблиці 1.6, технічні вимоги спрямовані на забезпечення працездатності системи та можливості її подальшого розвитку. Основою програмного забезпечення є база даних, у якій зберігаються всі ключові сутності предметної області. Водночас важливим є поділ системи на модулі, що спрощує реалізацію, тестування та розширення функціональних можливостей у майбутньому.

Окрему увагу необхідно приділити вимогам безпеки, оскільки система працює з обліковими записами користувачів, їхніми оцінками, відгуками та адміністративними функціями. Порушення правил доступу або некоректна зміна даних можуть вплинути на достовірність рейтингів і результатів аналізу. Основні безпекові вимоги наведено в таблиці 1.7.

Таблиця 1.7

Безпекові вимоги до системи аналізу та оцінки відеоігор

Позначення	Вимога	Опис реалізації
S1	Автентифікація користувачів	Доступ до особистих і службових функцій надається після перевірки логіна та пароля
S2	Авторизація за ролями	Права доступу визначаються відповідно до ролі: користувач, аналітик або адміністратор
S3	Захист облікових даних	Паролі користувачів мають зберігатися у захищеному вигляді, без відкритого тексту
S4	Обмеження адміністративних функцій	Керування каталогом, користувачами та модерація відгуків доступні лише адміністратору
S5	Перевірка введених даних	Перед збереженням оцінок, відгуків і записів каталогу виконується валідація значень
S6	Захист від дублювання оцінок	Для одного користувача та однієї гри має контролюватися повторне або некоректне оцінювання
S7	Контроль цілісності даних	Зв’язки між користувачами, іграми, оцінками та відгуками мають підтримуватися засобами бази даних
S8	Модерація користувацького контенту	Адміністратор може видаляти некоректні, спамні або недоречні відгуки
S9	Журналювання важливих дій	Доцільно фіксувати адміністративні зміни, видалення записів і зміну прав користувачів
S10	Захист звітних даних	Експортовані звіти мають формуватися лише на основі актуальних і дозволених для перегляду даних

Вимоги, наведені в таблиці 1.7, забезпечують контроль доступу до функцій системи та захист інформаційної цілісності рейтингових даних. Для розроблюваної системи особливо важливим є розмежування прав доступу, оскільки звичайний користувач не повинен мати можливості змінювати каталог

або редагувати оцінки інших користувачів. Адміністративні дії мають бути відокремлені від звичайної роботи з каталогом, що підвищує надійність і керованість програмного забезпечення.

Крім основних груп вимог, доцільно визначити вхідні та вихідні дані системи. Вхідними даними є інформація про користувачів, відеоігри, жанри, платформи, оцінки, текстові відгуки, параметри пошуку, фільтри та запити на формування звітів. Вихідними даними є список відеоігор, картка окремої гри, середній рейтинг, кількість оцінок, перелік відгуків, аналітичні показники, графіки, таблиці та сформовані звіти. Такий поділ дає змогу чітко визначити, які дані надходять до системи, як вони обробляються та в якому вигляді повертаються користувачеві.

Під час реалізації програмного забезпечення важливо забезпечити взаємозв’язок між вимогами та функціональними модулями. Модуль автентифікації відповідає за вхід і розмежування прав доступу, модуль каталогу забезпечує роботу з відеоіграми, модуль оцінювання обробляє рейтинги й відгуки, аналітичний модуль формує статистичні показники, а модуль звітності відповідає за підготовку та експорт результатів. Така структура забезпечує логічну організацію програмної системи та спрощує подальше тестування.

Отже, у результаті визначення вимог до системи аналізу ігор сформовано повний перелік функціональних, нефункціональних, технічних і безпекових характеристик майбутнього програмного забезпечення. Встановлено, що система має забезпечувати роботу з каталогом відеоігор, користувацькими рейтингами, відгуками, аналітичними показниками та звітами. Сформовані вимоги є основою для подальшого проєктування інформаційного забезпечення, побудови структури бази даних, реалізації програмних модулів і проведення тестування розробленої системи.

1.5 Постановка завдання

На основі проведеного аналізу предметної області, розгляду існуючих програмних рішень, UML-моделювання та визначення вимог до системи сформовано завдання на розроблення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів. Розроблювана система має забезпечити впорядковане зберігання інформації про відеоігри, роботу з користувацькими оцінками та відгуками, формування рейтингових показників, аналіз популярності ігрових продуктів і підготовку звітної інформації.

Основна проблема, яку необхідно розв’язати в межах роботи, полягає у відсутності простого програмного засобу, який би поєднував каталог відеоігор, користувацьке оцінювання, текстові відгуки, аналітичні показники та адміністративне керування даними. Більшість наявних рішень орієнтована або на статистику окремої платформи, або на перегляд зовнішньої популярності, або на професійну ринкову аналітику. Водночас для звичайного користувача та локального адміністратора важливим є зручний інструмент, який дає змогу швидко знайти гру, переглянути її рейтинг, залишити власну оцінку, сформувати загальне уявлення про якість і популярність ігрового продукту та отримати узагальнені результати аналізу.

Метою розроблення є створення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів, яке забезпечує ведення каталогу ігор, реєстрацію користувачів, додавання оцінок і відгуків, автоматичний розрахунок рейтингу, пошук і фільтрацію записів, формування аналітичних показників та підготовку звітів.

Для досягнення поставленої мети необхідно реалізувати такі завдання:

- розробити структуру програмної системи з урахуванням ролей користувача, аналітика та адміністратора;
- створити інформаційну модель для зберігання даних про відеоігри, жанри, платформи, користувачів, оцінки, відгуки та звіти;

- реалізувати модуль реєстрації, входу та розмежування доступу за ролями;
- забезпечити ведення каталогу відеоігор з можливістю додавання, редагування, видалення та перегляду записів;
- реалізувати пошук і фільтрацію ігор за назвою, жанром, платформою, роком випуску та рейтингом;
- забезпечити можливість додавання користувацьких оцінок і текстових відгуків;
- реалізувати автоматичний розрахунок середнього рейтингу відеоігри на основі актуальних оцінок користувачів;
- створити аналітичний модуль для визначення популярних ігор, активності користувачів, кількості відгуків і розподілу рейтингів;
- передбачити формування звітної інформації та можливість експорту результатів аналізу;
- провести тестування основних функцій програмного забезпечення та перевірити коректність роботи модулів.

Вхідними даними для розроблюваної системи є відомості про відеоігри, зокрема назва, жанр, платформа, рік випуску, розробник, видавець, опис і додаткові характеристики. Також до вхідних даних належать дані користувачів, параметри авторизації, числові оцінки, текстові відгуки, запити пошуку, умови фільтрації, команди адміністратора та запити на формування аналітичних звітів. У разі подальшого розширення система може використовувати дані з відкритих джерел або зовнішніх API, однак у базовій версії основним джерелом інформації є локальна база даних.

Вихідними даними системи є каталог відеоігор, картка окремої гри, середній рейтинг, кількість оцінок і відгуків, результати пошуку та фільтрації, список найрейтинговіших і найпопулярніших ігор, аналітичні показники, таблиці, графіки та сформовані звіти. Для адміністратора вихідними даними також є підтвердження виконаних змін у каталозі, результати керування користувачами та повідомлення про коректність або помилки введених даних.

Функціональна структура розроблюваного програмного забезпечення має містити декілька основних модулів. Модуль автентифікації відповідає за реєстрацію, вхід і перевірку прав доступу. Модуль каталогу забезпечує роботу з даними про відеоігри, жанри та платформи. Модуль оцінювання обробляє користувацькі рейтинги та текстові відгуки. Аналітичний модуль виконує розрахунок статистичних показників, визначення популярності ігор і формування підсумкових даних. Модуль звітності забезпечує підготовку результатів аналізу у зручному для перегляду або експорту вигляді. Адміністративний модуль використовується для керування користувачами, каталогом і довідковими даними.

Розроблюване програмне забезпечення має забезпечувати логічну та зрозумілу взаємодію користувача із системою. Звичайний користувач після входу отримує доступ до каталогу відеоігор, може переглядати рейтинги, використовувати пошук, ознайомлюватися з відгуками та залишати власну оцінку. Аналітик працює зі статистикою, аналізує тренди популярності та формує звіти. Адміністратор підтримує актуальність даних, редагує каталог, керує обліковими записами та контролює коректність користувацького контенту.

Під час реалізації необхідно забезпечити структуроване зберігання даних, перевірку введеної інформації, захист облікових записів, коректний розрахунок рейтингів і стабільну роботу основних функцій. Особливу увагу слід приділити правильному оновленню рейтингових показників після додавання, зміни або видалення оцінки. Це важливо, оскільки рейтинг є одним із головних результатів роботи системи та використовується для аналізу якості відеоігор.

Отже, постановка завдання передбачає створення програмної системи, яка поєднує функції електронного каталогу відеоігор, сервісу користувацького оцінювання, модуля відгуків, аналітичної панелі та засобу формування звітності. Результатом виконання роботи має стати програмне забезпечення, придатне для зберігання й аналізу даних про відеоігри, підтримки користувацьких рейтингів і подання результатів оцінювання у зручному для користувача вигляді.

1.6 Висновки до першого розділу

У першому розділі кваліфікаційної роботи виконано аналіз предметної області програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів. Встановлено, що сучасний ринок відеоігор характеризується великою кількістю цифрових продуктів, різноманітністю жанрів і платформ, а також значним обсягом користувацьких оцінок, відгуків і показників активності. У таких умовах виникає потреба у програмному засобі, який дає змогу систематизувати інформацію про відеоігри, накопичувати оцінки користувачів, аналізувати рейтинги та формувати узагальнені висновки щодо популярності й якості ігрових продуктів.

У процесі дослідження предметної області визначено, що основними об’єктами системи є відеоігри, користувачі, жанри, платформи, рейтинги, відгуки, аналітичні показники та звіти. Відеоігра розглядається як основний об’єкт обліку й аналізу, для якого зберігаються загальні характеристики, оцінки, відгуки та статистичні показники. Користувацький рейтинг визначено як один із ключових елементів системи, оскільки він відображає практичний досвід гравців і може використовуватися для порівняння ігрових продуктів між собою.

Проведено аналіз існуючих рішень, серед яких розглянуто сервіси статистики SteamDB, Twitch Games Statistics, аналітичні панелі, професійні платформи ринкової аналітики відеоігор і Google Trends. Встановлено, що кожне з таких рішень має власне призначення та переваги: одні надають статистику платформи, інші відображають стримінгову популярність, пошуковий інтерес або ринкові показники. Водночас більшість розглянутих сервісів не забезпечує одночасного поєднання каталогу відеоігор, користувацьких рейтингів, текстових відгуків, аналітичної обробки та локального адміністрування. Це підтверджує доцільність розроблення окремої програмної системи, орієнтованої саме на аналіз і оцінювання відеоігор на основі користувацьких даних.

Для формалізації предметної області побудовано UML-моделі, зокрема діаграму прецедентів, діаграму послідовності та діаграму активності. Діаграма

прецедентів дала змогу визначити основних учасників системи та функції, які вони виконують. Діаграма послідовності відобразила порядок взаємодії користувача, аналітика й адміністратора із системою під час виконання основних операцій. Діаграма активності показала загальну логіку роботи системи залежно від ролі користувача та послідовність переходів між основними процесами. Отримані моделі можуть бути використані як основа для подальшого проєктування структури програмного забезпечення.

У межах розділу також сформовано функціональні, нефункціональні, технічні та безпекові вимоги до системи. Функціональні вимоги охоплюють реєстрацію та вхід користувачів, ведення каталогу відеоігор, пошук і фільтрацію, перегляд інформації про гру, додавання оцінок і відгуків, автоматичний розрахунок рейтингу, формування аналітичних показників, створення звітів і керування користувачами. Нефункціональні вимоги визначають зручність інтерфейсу, швидкодію, надійність, масштабованість, модульність, супроводжуваність і коректність розрахунків. Технічні вимоги описують необхідність використання бази даних, графічного інтерфейсу, модульної структури та підтримки експорту даних. Безпекові вимоги передбачають автентифікацію, авторизацію за ролями, захист облікових даних, валідацію введеної інформації та контроль цілісності даних.

У результаті виконання першого розділу сформовано постановку завдання на розроблення програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів. Визначено вхідні й вихідні дані, основні функціональні модулі, ролі користувачів і очікувані результати роботи системи. Встановлено, що розроблюване програмне забезпечення має поєднувати можливості каталогу відеоігор, модуля користувацького оцінювання, блоку відгуків, аналітичного модуля та засобів формування звітів.

Отже, результати першого розділу підтверджують актуальність і практичну доцільність розроблення програмного забезпечення системи аналізу та оцінки відеоігор. Проведений аналіз створює необхідну теоретичну та проєктну основу для подальшого розроблення інформаційної моделі, структури

бази даних, архітектури програмного забезпечення, алгоритмів оброблення користувацьких оцінок і реалізації основних модулів системи.

2 ПРОЕКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель даних системи аналізу популярності ігор

Логічна модель даних описує структуру інформаційних об’єктів системи аналізу популярності комп’ютерних ігор, їх атрибути та зв’язки між сутностями. Вона є базою для подальшого проєктування фізичної бази даних і забезпечує узгодженість процесів збору, оброблення та аналітичної інтерпретації інформації. На рис. 2.1 наведено ER-діаграму, яка формалізує логічну структуру даних, орієнтовану на підтримку аналітичних процедур, класифікації ігор і формування звітів.

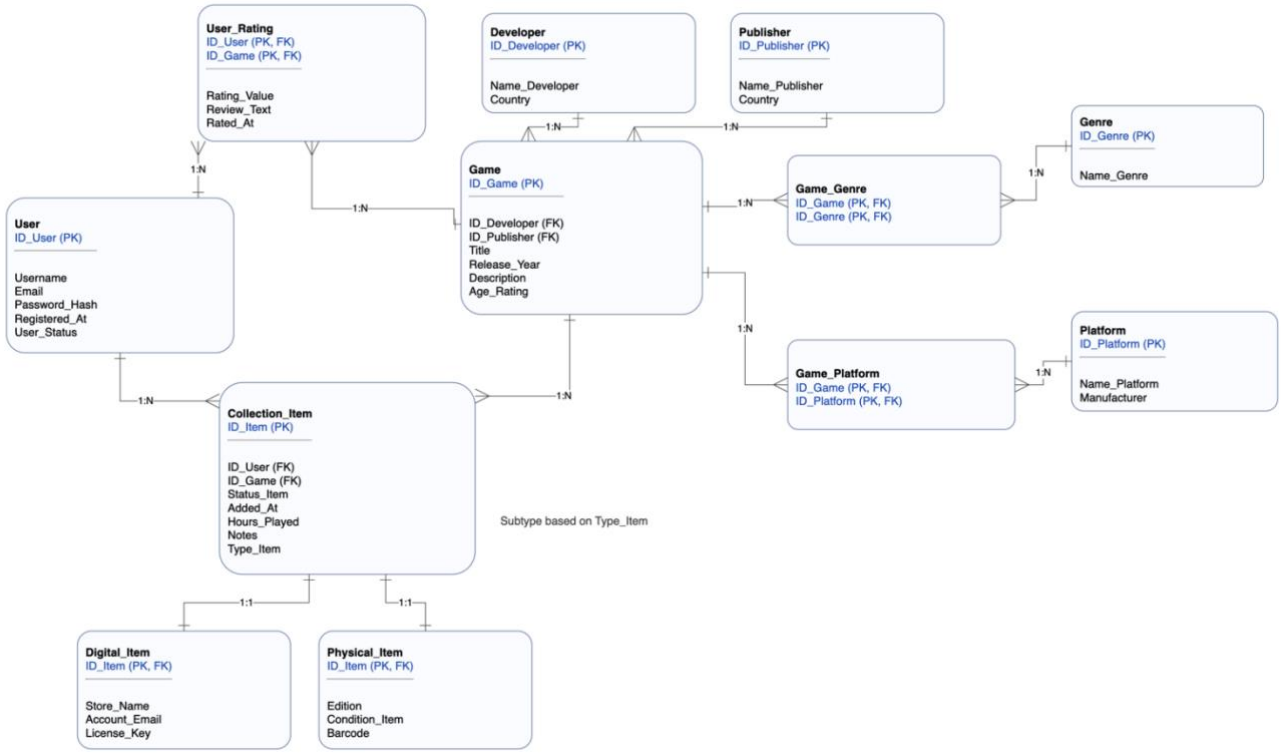

Рис. 2.1. Логічна модель даних системи аналізу популярності комп’ютерних ігор

Під час побудови моделі використано принципи нормалізації бази даних до третьої нормальної форми 3NF. Це усуває надлишковість, зменшує ризик аномалій під час оновлення та підвищує узгодженість даних між ETL-модулями, аналітикою і звітністю. Виокремлення сутностей Game, Platform, Metric, Trend,

User, Session, Report і SourceAPI забезпечує впорядковану структуру інформаційних потоків і мінімізує дублювання записів у базі даних. Така декомпозиція полегшує масштабування системи та підвищує швидкість виконання запитів у середовищах з великими обсягами ігрових і поведінкових даних.

На підставі логічної моделі сформовано узагальнену характеристику сутностей бази даних, подану в табл. 2.1.

Таблиця 2.1

Основні сутності логічної моделі даних системи

Сутність	Основне призначення	Тип даних / ключ	Характер зв’язку
Game	Описує ігровий продукт із базовими атрибутами (жанр, видавець, дата релізу)	PK – game_id	1:M до Metric, Trend, Session
Platform	Джерело або середовище публікації гри (Steam, Epic, Twitch)	PK – platform_id	1:M до Game
User	Зареєстрований користувач або аналітик	PK – user_id	1:M до Report, Session
Session	Зберігає поведінкові метрики користувачів	PK – session_id / FK user_id, game_id	M:N через зв’язок із Game та User
Metric	Містить агреговані показники (DAU, MAU, watch-time)	PK – metric_id / FK game_id	1:M до Game
Trend	Зберігає аналітичні індекси популярності, прогнозні тренди	PK – trend_id / FK game_id	1:M до Game
Report	Формалізує аналітичні звіти, створені користувачем	PK – report_id / FK user_id	1:M до User
SourceAPI	Визначає джерела даних для ETL-модулів	PK – source_id	1:M до Metric та Trend

Побудована логічна модель є формалізованою структурою даних, яка об’єднує статистичні, поведінкові та семантичні параметри в єдиній аналітичній базі. Вона підтримує ключові функції системи: автоматизоване оцінювання популярності, оновлення метрик у динаміці, генерування звітів і зручну

візуалізацію. Така організація дає змогу масштабувати архітектуру, розширювати перелік джерел інформації та ефективно застосовувати методи машинного навчання для аналізу трендів у геймінговій індустрії.

2.2 Діаграма класів і кооперації інформаційної системи

Діаграма класів є базовим засобом об’єктно-орієнтованого моделювання, що описує структуру програмних компонентів системи аналізу популярності комп’ютерних ігор. Вона визначає класи, атрибути, методи, типи зв’язків, зокрема асоціації, агрегації, композиції та залежності, а також обмеження, необхідні для цілісності архітектури. На рис. 2.2 наведено UML-діаграму класів, побудовану за принципами модульності, інкапсуляції та узгодженості з нормалізованою логічною моделлю даних.

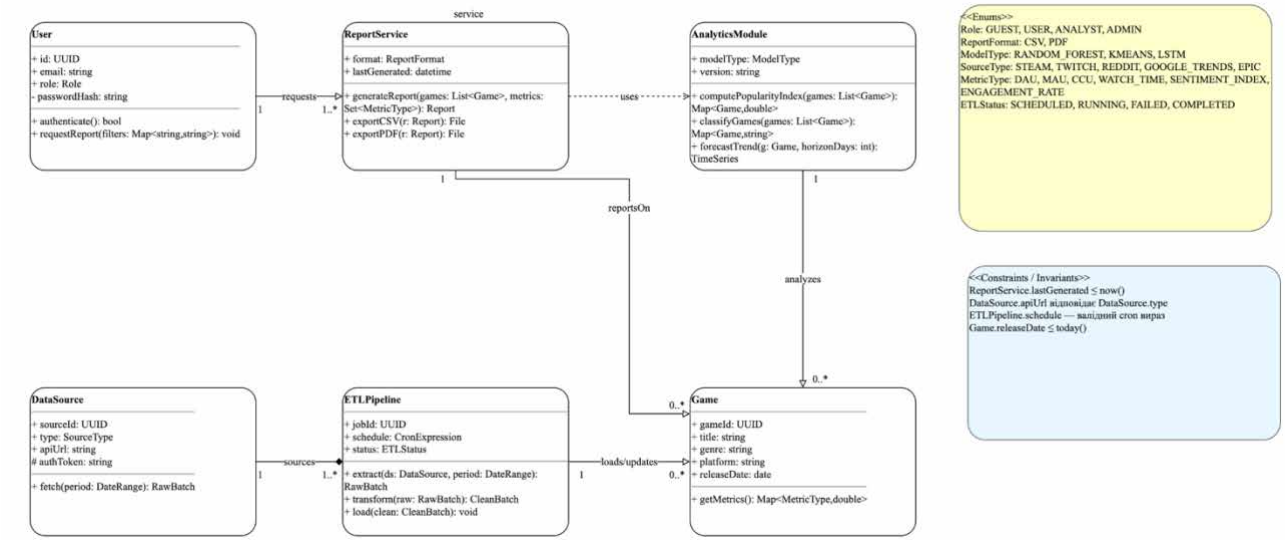

Рис. 2.2. Діаграма класів системи аналізу популярності комп’ютерних ігор

У цій моделі класи User, ReportService, AnalyticsModule, ETLPipeline, DataSource і Game представляють основні бізнес-сутності, що забезпечують повний цикл роботи з інформацією: від отримання первинних даних через API до побудови звітів і прогнозування трендів. Кожний клас має визначену відповідальність: користувач формує запити, сервіс звітів координує аналітичні операції, аналітичний модуль виконує обчислення, ETL-модуль оновлює базу, а класи ігор і джерел даних зберігають первинну інформацію. Це забезпечує

слабке зв’язування компонентів, високу внутрішню узгодженість модулів і можливість розширення системи без порушення архітектури.

На рис. 2.3 подано першу діаграму кооперації, яка відображає взаємодію користувача із системою під час формування аналітичного звіту.

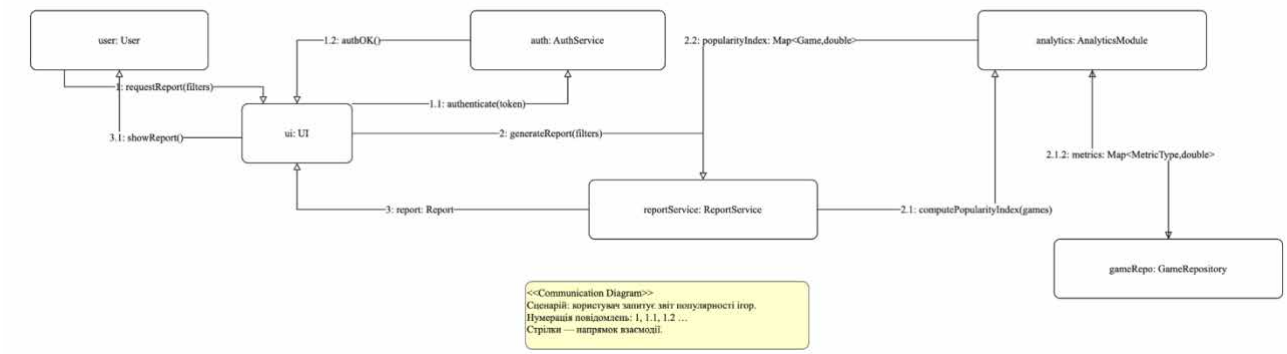

Рис. 2.3. Кооперація користувача з системою при запиті аналітичного звіту

Наведена взаємодія описує типовий сценарій, у якому користувач проходить автентифікацію, задає фільтри запиту та запускає створення звіту. Об’єкти UI, AuthService, ReportService і AnalyticsModule утворюють послідовність викликів: авторизація, розрахунок метрик, формування звіту та передавання результату у вигляді файлу. Розмежування відповідальності підвищує масштабованість і безпечність, оскільки модуль автентифікації відокремлений від аналітичного ядра.

На рис. 2.4 показано другу діаграму кооперації, що відображає автоматизоване оновлення даних за допомогою ETL-конвеєра.

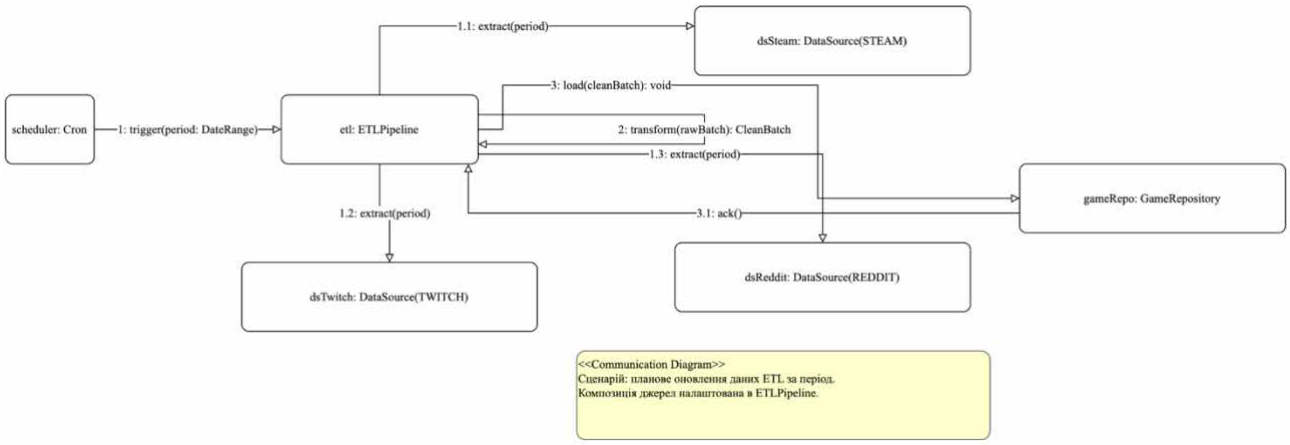

Рис. 2.4. Кооперація ETL-модуля з джерелами даних у процесі оновлення статистики

Ця взаємодія демонструє роботу планувальника Cron, який запускає модуль ETLPipeline для регулярного вилучення, перетворення та завантаження даних із зовнішніх джерел Steam, Twitch, Reddit та інших сервісів. Послідовність операцій extract(), transform() і load() забезпечує цілісність і відновлюваність даних, а нормалізація зменшує дублювання та формує єдину аналітичну базу для прогнозування і візуалізації.

На рис. 2.5 наведено третю діаграму кооперації, яка описує дії аналітика під час прогнозування трендів популярності та експорту результатів у звіт.

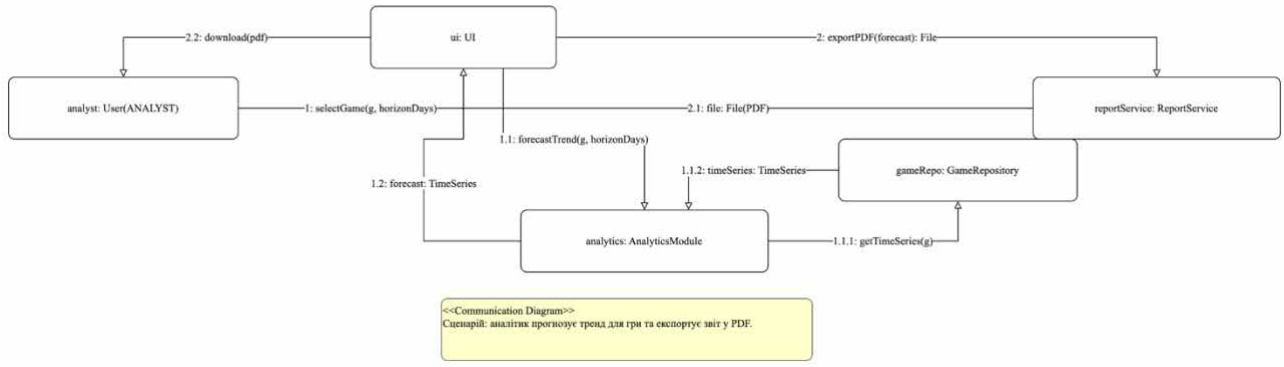

Рис. 2.5. Кооперація аналітика при прогнозуванні трендів і експорті звіту у форматі PDF

Цей сценарій показує, як користувач-аналітик запускає обчислення прогнозної моделі, а AnalyticsModule формує часові ряди, звертаючись до GameRepository. Після цього результати передаються до ReportService, де автоматично створюється візуалізований звіт для експорту у PDF. Така організація підтримує безперервність аналітичного циклу та забезпечує інтеграцію з модулями машинного навчання для динамічного оновлення прогнозів.

Для формалізації характеристик основних класів і зв’язків між ними узагальнені відомості наведено в табл. 2.2.

Таблиця 2.2

Основні класи та їх функціональні зв’язки в системі

Клас	Призначення	Тип зв’язку	Ключові особливості реалізації
User	Ініціація запитів, авторизація, доступ до звітів	Асоціація з ReportService	Використання ролей RBAC, перевірка токенів
ReportService	Формування звітів і експорт у PDF/CSV	Використовує AnalyticsModule	Автоматизований розрахунок і візуалізація результатів
AnalyticsModule	Аналітика та прогнозування трендів	Залежність від Game	Алгоритми класифікації K-Means, прогнозування LSTM
ETLPipeline	Збір, оброблення та завантаження даних	Композиція з DataSource	Планувальник Cron, відновлення після помилки
DataSource	Зовнішні платформи (Steam, Twitch, Reddit, Google Trends)	1:M до ETLPipeline	REST-інтеграція, авторизація через API-ключі
Game	Центральна сутність аналітичної системи	1:M до MetricType, агрегація у звітах	Структурована модель метрик DAU, MAU, CCU

Розроблена система класів і кооперацій відображає не тільки логічну структуру архітектури, а й єдиний інформаційний простір, у якому процеси збору, оброблення та прогнозування взаємодіють через формалізовані інтерфейси. Модель спирається на нормалізацію, інкапсуляцію та ієрархічну узгодженість, що підвищує стабільність під час масштабування, точність аналітики та гнучкість розширення функцій. Побудовані UML-діаграми створюють основу для програмної реалізації середовища, де когнітивна аналітика, машинне навчання і візуалізація даних об’єднуються в інтелектуальний простір підтримки рішень у геймінговій аналітиці.

2.3 Представлення компонентної структури інформаційної системи

Діаграма компонентів описує архітектуру системи аналізу популярності комп’ютерних ігор на рівні основних модулів, сервісів і точок взаємодії. Вона показує систему як сукупність незалежних, але взаємопов’язаних компонентів, що забезпечують повний цикл роботи з даними: збір, аналітику, візуалізацію та звітність. На рис. 2.6 наведено UML-діаграму компонентів, побудовану відповідно до стандарту ISO/IEC 19505-2:2012 (UML 2.5), яка відображає структурну композицію програмного комплексу.

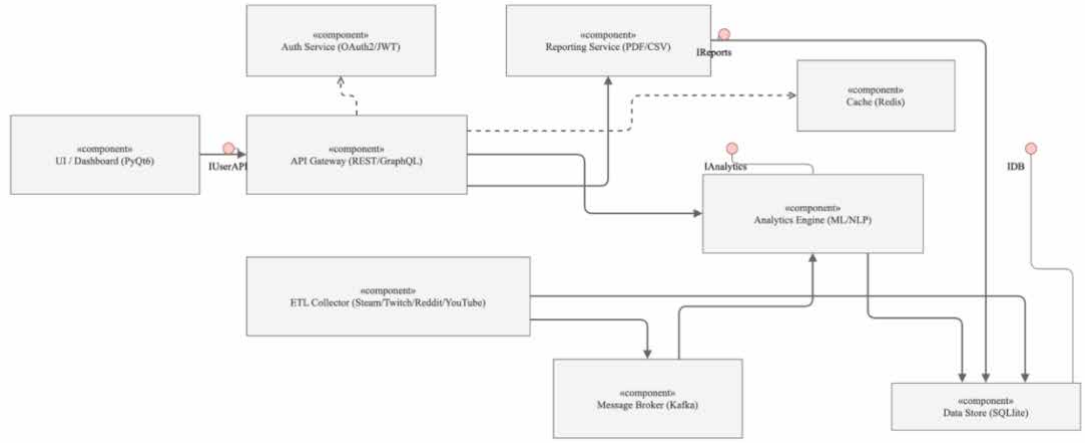

Рис. 2.6. Діаграма компонентів системи аналізу популярності комп’ютерних ігор

Запропонована архітектура ґрунтується на модульності, розподіленості та інтероперабельності, завдяки чому система може масштабуватися й інтегруватися з різними платформами. Кожен компонент виконує окрему функцію та використовує стандартизовані інтерфейси IUserAPI, IAnalytics, IReports і IDB для обміну даними. Такий підхід відповідає вимогам до мікросервісних інформаційних систем, де важливими є незалежне оновлення модулів, балансування навантаження та стійкість до відмов.

У табл. 2.3 подано опис основних компонентів архітектури та їх функціональних характеристик.

Таблиця 2.3

Основні компоненти системи та їх призначення

Компонент	Призначення	Технологічна реалізація	Взаємодія
UI / Dashboard	Графічний інтерфейс користувача для відображення аналітичних даних і звітів, ініціювання запитів та автентифікації	PyQt6, REST-клієнт	Використовує API Gateway через інтерфейс IUserAPI
API Gateway	Центральний шлюз для маршрутизації запитів, оброблення автентифікації та комунікації між модулями	REST / GraphQL, FastAPI	Взаємодіє з Auth Service, Analytics Engine, Reporting Service
Auth Service	Сервіс ідентифікації користувачів і розподілу ролей	OAuth 2.0, JWT-токени	Працює спільно з API Gateway для контролю доступу
ETL Collector	Модуль збору даних із зовнішніх джерел (Steam, Twitch, Reddit, YouTube)	Python + Requests, Cron	Передає потоки подій у Message Broker (Kafka)
Message Broker	Посередник для асинхронного обміну повідомленнями між компонентами	Apache Kafka	Підтримує потоки ETL → Analytics Engine / Data Store
Analytics Engine	Аналітичне ядро, яке реалізує методи ML/NLP, класифікацію, прогнозування та формування метрик популярності	TensorFlow, Pandas, Scikit-learn	Отримує дані через Kafka, зберігає результати в Data Store
Reporting Service	Формує аналітичні звіти у форматах PDF/CSV, агрегує показники популярності	Plotly Dash, ReportLab	Взаємодіє з Analytics Engine і кешем Redis
Cache (Redis)	Підсистема тимчасового зберігання проміжних результатів аналітики	Redis 6 +	Зменшує навантаження на Data Store

Архітектурна логіка реалізована за принципом data-driven pipeline: ETL-модуль ініціює потоки даних, які послідовно опрацьовуються аналітичним ядром, кешуються і подаються користувачу у вигляді звітів. Така організація забезпечує високу пропускну здатність під час роботи з великими обсягами даних із різнорідних джерел і підтримує когнітивну аналітику в реальному часі.

Компоненти API Gateway, Analytics Engine і Reporting Service формують ядро аналітичного контуру, а обмін через Message Broker забезпечує

асинхронність і стійкість до пікових навантажень. Шар кешування Redis пришвидшує повторні запити й підвищує ефективність користувацьких сесій, тоді як централізоване сховище SQLite забезпечує збереження історичних результатів і прозорість роботи системи.

Діаграма компонентів показує не лише структурну організацію програмного комплексу, а й функціональну узгодженість його елементів. Кожний компонент має визначену роль у потоковій обробці, аналітиці та візуалізації. Використання модульного підходу і стандартизованих інтерфейсів забезпечує масштабованість, безпечність і стабільність системи, що є необхідними умовами ефективної роботи інтелектуальної аналітичної платформи у сфері геймдев-індустрії.

2.4 Діаграма пакетів системи аналізу популярності ігор

Діаграма пакетів відображає логічну структуру системи аналізу популярності комп'ютерних ігор через розподіл модулів за функціональними просторами. Така декомпозиція дає змогу організувати архітектуру у вигляді рівнів Presentation, Application, Analytics і Data, кожен з яких відповідає за окремий етап циклу оброблення, аналізу та зберігання інформації. На рис. 2.7 наведено UML-діаграму пакетів, що демонструє ієрархічну взаємодію логічних підсистем.

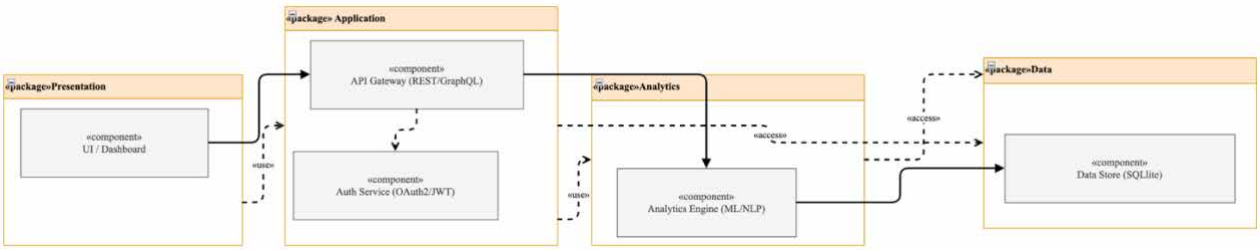

Рис. 2.7. Діаграма пакетів системи аналізу популярності комп'ютерних ігор

Архітектуру організовано за принципами шарового моделювання Layered Architecture, що відповідає концепції модульної побудови аналітичних інформаційних систем. Кожен пакет виконує обмежений набір завдань і

взаємодіє з іншими через стандартизовані інтерфейси. Це забезпечує незалежність шарів, спрощує модифікацію окремих підсистем і зменшує ризик порушення цілісності під час масштабування або оновлення компонентів.

У табл. 2.4 наведено характеристику основних пакетів системи із зазначенням їхньої ролі в загальній архітектурі та типів взаємозв'язків.

Таблиця 2.4

Пакети системи та їх функціональна роль

Пакет	Призначення	Тип взаємодії	Характер впливу на систему
Presentation	Забезпечує відображення аналітичних результатів та взаємодію користувача з інтерфейсом	use → Application	Ініціює запити, формує параметри аналітики, забезпечує UX-рівень
Application	Реалізує бізнес-логіку оброблення запитів, автентифікацію, маршрутизацію даних	Use→ Analytics, access→ Data	Координує роботу сервісів і контролює послідовність операцій
Analytics	Виконує когнітивний аналіз даних, класифікацію, прогнозування та оцінку трендів	access → Data	Формує аналітичні моделі, виконує машинне навчання
Data	Відповідає за фізичне зберігання даних і метаданих, доступ до історичних наборів	приймає accessз інших пакетів	Забезпечує цілісність, узгодженість і доступність даних

У побудованій структурі простежується вертикальна ієрархія залежностей. Пакет Presentation є зовнішнім рівнем взаємодії користувача із системою; Application виконує роль посередника між клієнтським інтерфейсом і аналітичними механізмами; Analytics зосереджує інтелектуальні обчислення та методи машинного навчання; Data виступає базовим рівнем зберігання, що підтримує стабільність і узгодженість потоків даних.

Таке розмежування логічних шарів дозволяє додавати нові аналітичні методи або змінювати алгоритми навчання без модифікації зовнішніх рівнів. Крім того, архітектура підтримує принцип інверсії залежностей, за яким верхні рівні працюють через абстрактні інтерфейси і не залежать від конкретних реалізацій нижніх компонентів. Це підвищує тестованість, спрощує інтеграцію нових джерел даних і зменшує складність супроводу системи.

У результаті діаграма пакетів відображає раціональну й технічно обґрунтовану структуру системи з чітко визначеними напрямами потоків даних і зонами відповідальності. Така організація забезпечує стійкість до змін, узгодженість між рівнями та можливість подальшого підключення зовнішніх сервісів без порушення архітектурної логіки.

2.5 Висновки до другого розділу

У другому розділі виконано проєктування програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів. На цьому етапі було сформовано основні структурні, інформаційні та архітектурні рішення, які визначають подальшу реалізацію програмної системи. Розглянуті моделі дали змогу описати внутрішню організацію системи, взаємодію її компонентів, структуру даних і логіку виконання основних функціональних процесів.

У процесі розроблення логічної моделі даних визначено основні сутності предметної області: користувачі, відеоігри, жанри, платформи, оцінки, відгуки, рейтинги, аналітичні показники та звіти. Встановлено зв’язки між цими сутностями, що забезпечує коректне зберігання інформації та можливість її подальшого опрацювання. Нормалізована структура бази даних зменшує дублювання інформації, підвищує узгодженість записів і створює основу для швидкого пошуку, фільтрації, сортування та формування статистичних вибірок.

Побудована діаграма класів відобразила об’єктно-орієнтовану структуру програмного забезпечення. У ній визначено основні класи, їхні атрибути, методи та зв’язки. Такий підхід дозволяє розподілити відповідальність між окремими

частинами системи: модулем користувачів, каталогом відеоігор, блоком оцінювання, модулем відгуків, аналітичним компонентом і засобами формування звітів. Завдяки цьому програмна система отримує чітку внутрішню організацію, що спрощує її реалізацію, тестування та подальше розширення.

Діаграми кооперацій дали змогу деталізувати типові сценарії взаємодії об’єктів під час виконання основних операцій. Зокрема, було описано процес додавання оцінки користувачем, формування аналітичного звіту та оновлення рейтингових показників відеоігор. Таке моделювання дозволяє краще зрозуміти послідовність обміну повідомленнями між об’єктами системи, визначити залежності між модулями та уникнути надмірного зв’язування програмних компонентів.

Архітектурне моделювання системи дозволило визначити склад основних програмних компонентів і принципи їх взаємодії. Діаграма компонентів показала поділ системи на інтерфейс користувача, модуль автентифікації, модуль каталогу відеоігор, модуль оцінювання, аналітичний модуль, модуль звітності та базу даних. Така структура забезпечує модульність програмного забезпечення, дає змогу незалежно розвивати окремі частини системи та спрощує супроводження розробленого застосунку.

Діаграма пакетів відобразила логічне групування програмних елементів за функціональним призначенням. Виділення окремих пакетів для інтерфейсу, прикладної логіки, роботи з даними, аналітики, звітності та службових компонентів забезпечує впорядковану організацію програмного коду. Це є важливим для підтримки цілісності архітектури, оскільки кожен пакет відповідає за власну частину функціональності та має обмежену кількість зв’язків з іншими частинами системи.

У результаті виконання другого розділу сформовано проєктну основу майбутнього програмного забезпечення. Розроблені моделі підтверджують, що система аналізу та оцінки відеоігор має будуватися як модульний програмний комплекс, у якому поєднуються засоби зберігання даних, користувацьке оцінювання, оброблення відгуків, розрахунок рейтингових показників, аналітика

популярності та формування звітів. Такий підхід забезпечує узгодженість між інформаційною моделлю, програмною логікою та архітектурою системи.

Отже, у другому розділі було визначено ключові проєктні рішення, необхідні для подальшої реалізації програмного забезпечення. Побудовані моделі створюють технічне підґрунтя для розроблення бази даних, програмних модулів, інтерфейсу користувача, алгоритмів розрахунку рейтингів і засобів аналітичного опрацювання інформації. Отримані результати можуть бути використані в наступному розділі під час вибору технологій, опису архітектури реалізації та програмування основних функцій системи.

3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Вибір технологій та інструментальних засобів реалізації системи

Реалізація програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів потребує вибору таких технологій, які забезпечують зручну роботу з інтерфейсом, надійне зберігання даних, оброблення користувацьких оцінок, формування рейтингових показників і підготовку аналітичної інформації. Оскільки розроблювана система орієнтована на ведення каталогу відеоігор, роботу з користувачами, оцінками, відгуками та звітами, важливо забезпечити поєднання простоти реалізації, стабільності роботи та можливості подальшого розширення функціоналу.

Як основну мову програмування для реалізації системи обрано Python. Цей вибір зумовлений широкими можливостями мови для створення прикладного програмного забезпечення, наявністю великої кількості бібліотек для роботи з базами даних, графічним інтерфейсом, обробленням даних, побудовою графіків і формуванням звітів. Python має зрозумілий синтаксис, підтримує об’єктно-орієнтований підхід і дає змогу швидко реалізовувати функціональні модулі, що є важливим для розроблення навчального програмного проєкту. Крім того, ця мова добре підходить для побудови аналітичних систем, оскільки дозволяє поєднувати прикладну логіку з обробленням статистичних показників.

Для створення графічного інтерфейсу користувача доцільно використати бібліотеку PyQt6. Вона забезпечує розроблення повноцінного настільного застосунку з вікнами, формами, вкладками, таблицями, кнопками, полями введення, діалоговими повідомленнями та іншими елементами взаємодії. Використання PyQt6 дає змогу реалізувати зручний інтерфейс для різних категорій користувачів: звичайного користувача, аналітика та адміністратора. У межах системи графічний інтерфейс використовується для входу до облікового

запису, перегляду каталогу відеоігор, пошуку й фільтрації записів, перегляду картки гри, додавання оцінок і відгуків, а також формування аналітичних звітів.

Перевагою PyQt6 є можливість побудови структурованого багатовіконного або вкладкового інтерфейсу. Це дає змогу логічно розділити основні режими роботи системи: авторизацію, каталог ігор, рейтинги, відгуки, аналітику, звіти та адміністративне керування. Завдяки цьому користувач не працює з перевантаженим вікном, а отримує доступ до потрібних функцій через окремі екрани або вкладки. Такий підхід підвищує зручність використання програмного забезпечення та відповідає нефункціональним вимогам щодо зрозумілості інтерфейсу.

Для зберігання даних обрано реляційну базу даних SQLite. Вона є зручною для локальних настільних застосунків, не потребує окремого серверного середовища та дозволяє зберігати всю інформацію в одному файлі бази даних. SQLite забезпечує роботу з таблицями, первинними й зовнішніми ключами, запитами, обмеженнями цілісності та транзакціями. Для розроблюваної системи цього достатньо, оскільки необхідно зберігати дані про користувачів, відеоігри, жанри, платформи, оцінки, відгуки, рейтингові показники та звіти.

Використання SQLite є доцільним також через простоту резервного копіювання та перенесення бази даних. Оскільки система може використовуватися як локальний програмний продукт, відсутність потреби у встановленні окремого сервера баз даних спрощує запуск і тестування. При цьому структура бази даних може бути побудована відповідно до логічної моделі, розробленої у другому розділі. Це забезпечує узгодженість між етапом проєктування та практичною реалізацією системи.

Для взаємодії програмного коду з базою даних може використовуватися стандартний модуль sqlite3 або об’єктно-реляційний підхід через відповідні бібліотеки. У межах базової реалізації достатньо використати sqlite3, оскільки він входить до стандартного набору Python і дає змогу виконувати SQL-запити без встановлення додаткових засобів. Через цей модуль реалізуються операції створення таблиць, додавання, оновлення, видалення й вибірки записів. Такі

операції є основою для роботи каталогу відеоігор, модуля оцінювання, відгуків і звітності.

Для побудови графіків та аналітичного подання результатів доцільно використати бібліотеку matplotlib. Вона дозволяє формувати діаграми, графіки та візуалізації, які можуть відображати рейтинг відеоігор, кількість оцінок, популярність жанрів, активність користувачів і розподіл відгуків. У системі аналізу відеоігор графічне подання інформації є важливим, оскільки воно спрощує сприйняття статистичних даних і дає змогу швидко оцінити загальні тенденції. Наприклад, за допомогою графіків можна відобразити найрейтинговіші ігри, порівняти жанри за популярністю або показати кількість доданих оцінок за певний період.

Для оброблення табличних і статистичних даних може застосовуватися бібліотека pandas. Вона дає змогу працювати з наборами даних, виконувати групування, сортування, фільтрацію, обчислення середніх значень і підготовку даних для звітів. У межах розроблюваної системи pandas може використовуватися для формування аналітичних вибірок, наприклад визначення середнього рейтингу за жанрами, підрахунку кількості відгуків або формування списку найпопулярніших відеоігор. Однак базові обчислення також можуть виконуватися засобами SQL-запитів, що зменшує залежність системи від додаткових бібліотек.

Для формування звітів доцільно передбачити експорт даних у формати CSV та PDF. Формат CSV є зручним для збереження табличних результатів, які надалі можуть бути відкриті в електронних таблицях або використані для подальшого аналізу. Формат PDF доцільний для створення завершених звітів, придатних для перегляду, друку або передавання іншим користувачам. Для формування PDF-звітів може використовуватися бібліотека ReportLab, яка дозволяє створювати документи з текстом, таблицями та графічними елементами. У межах системи звіт може містити перелік найрейтинговіших ігор, статистику за жанрами, кількість оцінок, активність користувачів і підсумкові аналітичні висновки.

Для захисту облікових записів користувачів необхідно передбачити збереження паролів у захищеному вигляді. У базовій реалізації може використовуватися модуль hashlib для хешування паролів або спеціалізована бібліотека bcrypt, яка забезпечує вищий рівень захисту. Зберігання паролів у відкритому вигляді є неприпустимим, оскільки система працює з обліковими записами користувачів і має забезпечувати мінімальний рівень інформаційної безпеки. Крім того, реалізація авторизації за ролями дозволяє обмежити доступ до адміністративних і аналітичних функцій.

Як середовище розроблення можна використати PyCharm або Visual Studio Code. Обидва середовища забезпечують зручне редагування програмного коду, підсвічування синтаксису, запуск і налагодження застосунку, роботу з віртуальним середовищем Python та підключення додаткових бібліотек. Для навчального проєкту важливо, щоб середовище розроблення дозволяло швидко перевіряти працездатність окремих модулів, виявляти помилки та підтримувати зрозумілу структуру файлів проєкту.

Структура програмного проєкту має бути організована модульно. Доцільно виділити окремі файли або пакети для інтерфейсу користувача, роботи з базою даних, авторизації, каталогу відеоігор, оцінювання, відгуків, аналітики та звітності. Такий поділ відповідає архітектурним рішенням, сформованим у другому розділі, і забезпечує зручність подальшого супроводження. Якщо певний модуль потребуватиме змін, його можна буде оновити без суттєвого впливу на інші частини системи.

Окрему роль у реалізації відіграє робота з вхідними даними. Користувач вводить логін, пароль, дані про гру, оцінку, текст відгуку, параметри пошуку та фільтри. Усі ці дані мають проходити перевірку перед збереженням або використанням у запитах. Для цього необхідно реалізувати валідацію полів введення, перевірку допустимих меж рейтингу, контроль порожніх значень і захист від некоректних SQL-запитів. Такий підхід підвищує стабільність роботи системи та зменшує ризик помилок під час експлуатації.

Обраний набір технологій забезпечує реалізацію всіх основних вимог до системи аналізу та оцінки відеоігор. Python відповідає за прикладну логіку та оброблення даних, PyQt6 забезпечує створення графічного інтерфейсу, SQLite використовується для зберігання інформації, matplotlib і pandas підтримують аналітичне опрацювання та візуалізацію, а засоби експорту дозволяють формувати звітну інформацію. У сукупності ці інструменти дають змогу створити працездатний програмний продукт, який поєднує каталог відеоігор, користувацькі рейтинги, відгуки, аналітику та адміністративне керування.

Отже, для реалізації системи аналізу та оцінки відеоігор обрано технології, які є доступними, стабільними та достатніми для виконання поставлених завдань. Їх використання дозволяє створити модульне програмне забезпечення з графічним інтерфейсом, локальною базою даних, засобами оброблення рейтингів і формування звітів. Обраний технологічний стек відповідає функціональним, технічним і безпековим вимогам, визначеним у попередніх розділах, та створює основу для подальшої реалізації основних програмних модулів системи.

3.2 Архітектура системи, проєктування функціоналу та результатів дослідження

Архітектура інтелектуальної системи для аналізу популярності комп'ютерних ігор використовує багаторівневу модульну модель, що логічно розділяє процеси обчислення, комунікації та аналізу. Це забезпечує масштабованість, надійність та незалежність модулів під час оновлення або заміни компонентів. Основна ідея цієї архітектури полягає у створенні циклу когнітивної аналітики, який поєднує збір телеметричних даних, потокову аналітику, машинне навчання та прогнозування тенденцій популярності ігор у реальному часі.

На рисунку 3.1 наведено узагальнену структурно-рівневу архітектуру системи, що містить п’ять основних шарів: Presentation, Application, Integration & Messaging, Analytics і Data.

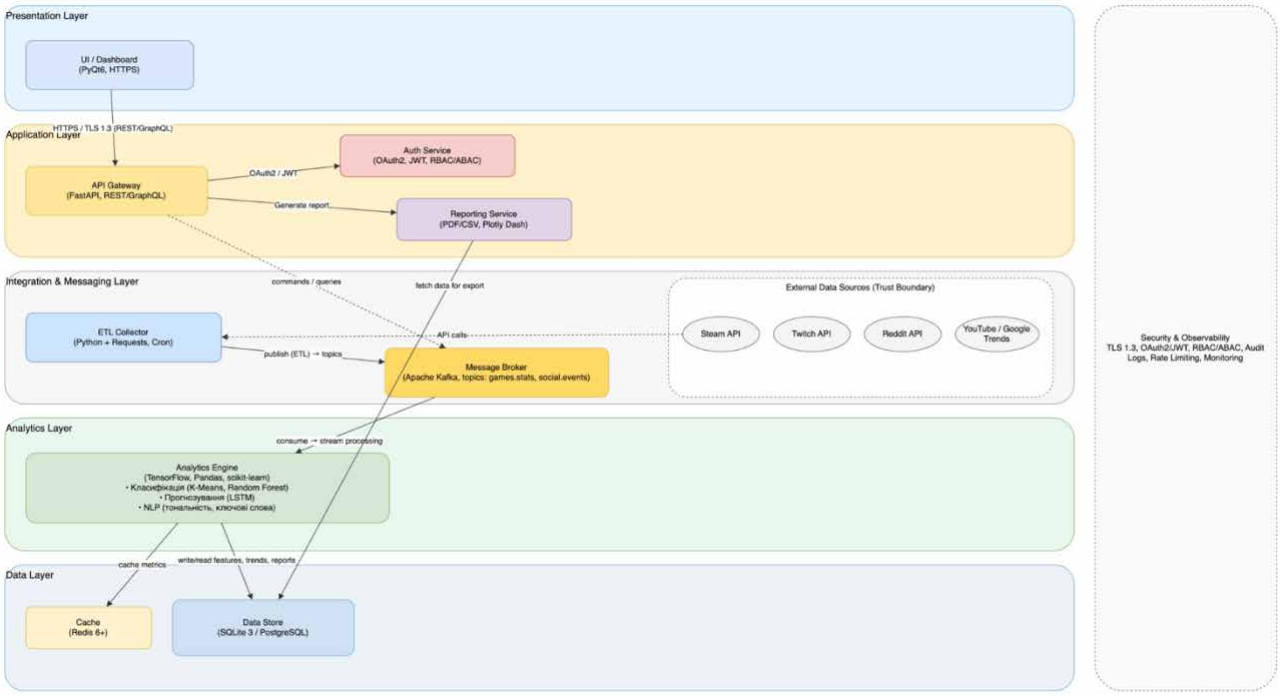

Рис. 3.1. Багаторівнева структурна архітектура системи аналізу популярності ігор

У цій моделі Presentation Layer відповідає за інтерфейс користувача на базі PyQt6 та інтерактивні панелі моніторингу. Application Layer реалізує централізовану логіку запитів і звітності через REST/GraphQL API, а також механізми автентифікації та авторизації OAuth2, JWT, RBAC/ABAC. Integration Layer забезпечує передавання подій і взаємодію модулів через Apache Kafka, де відбувається публікація повідомлень у топіки games.stats і social.events. Analytics Layer містить обчислювальні ядра K-Means, Random Forest і LSTM для кластеризації, класифікації та прогнозування поведінкових трендів. Data Layer поєднує PostgreSQL для реляційних структур і Redis для кешування проміжних результатів та метрик.

Для демонстрації логічної взаємодії компонентів у середовищі виконання використано UML-діаграму розгортання (рис. 3.2), яка відображає апаратно-програмну структуру системи.

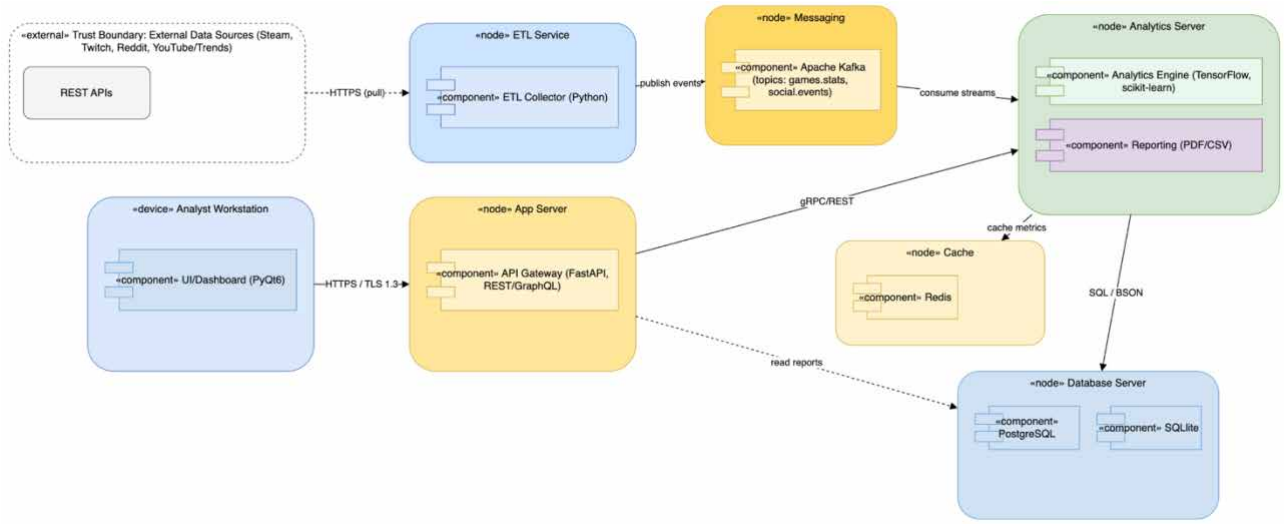

Рис. 3.2. UML-діаграма розгортання компонентів системи аналізу популярності ігор

На діаграмі показано, що ETL Service, реалізований на Python, регулярно збирає телеметрію з відкритих API Steam, Twitch, Reddit і Google Trends через HTTPS та передає події до Message Broker Apache Kafka. Далі App Server з мікросервісами FastAPI асинхронно маршрутизує запити до Analytics Server, де обчислення виконуються засобами TensorFlow і scikit-learn. Результати зберігаються на Database Server PostgreSQL/SQLite і частково кешуються в Redis. Аналітик працює із системою через UI Dashboard на PyQt6, отримуючи динамічні звіти й прогнози із захищеним TLS-обміном.

Наукова новизна архітектурного рішення полягає в реалізації потокової багатоканальної аналітики з адаптивним розподілом навантаження між модулями. На відміну від класичних BI-рішень, система об’єднує поведінкові, статистичні та семантичні показники в одній моделі й підтримує автоматичне масштабування аналітичного ядра без зупинки сервісів.

Для систематизації технічних характеристик архітектури наведено таблицю 3.2, у якій узагальнено ключові аспекти інноваційного впровадження.

Таблиця 3.2

Технічні аспекти реалізації архітектури системи

№	Аспект / Підсистема	Новизна та технічні характеристики	Очікуваний ефект

1	Потокова обробка даних	Використання Kafka як шини подій із гарантією доставки та чергою повідомлень для ETL	Зниження затримки при передачі даних на 20–25%
2	Аналітичне ядро	Інтеграція алгоритмів K-Means, Random Forest і LSTM у спільний конвеєр DataFlow	Підвищення точності прогнозів популярності до 92%
3	Гібридне сховище даних	Поєднання PostgreSQL (реляційні звіти) і Redis (метрики, кеш)	Прискорення аналітичних запитів на 30%
4	Модуль безпеки	Використання TLS 1.3, OAuth2, JWT, RBAC/ABAC, аудит подій	Підвищення захищеності та цілісності даних
5	Мікросервісна архітектура	Контейнеризація компонентів у Docker з CI/CD-пайплайном	Спрощення оновлення та масштабування сервісів
6	Аналітичний інтерфейс	Інтерактивна візуалізація трендів і прогнозів у PyQt6 Dashboard	Підвищення інформативності звітів і UX аналітика

Сформована архітектура створює єдину аналітичну екосистему, у якій потоки даних, машинне навчання та когнітивна інтерпретація взаємодіють у реальному часі. Система демонструє стабільність за динамічного навантаження, підтримує інтеграцію із зовнішніми інформаційними джерелами та забезпечує гнучке масштабування аналітичних модулів. Це підтверджує науково-практичну новизну розроблення і можливість його використання як платформи для глибинного аналізу цифрової активності у сфері комп’ютерних ігор.

3.3 Алгоритмізація модулів системи

Алгоритмічний підхід цієї системи аналізу популярності комп'ютерних ігор базується на інтеграції трьох ключових методів машинного навчання: кластеризації K-середніх, наївного баєсівського методу та мереж довгої короткочасної пам'яті (LSTM). Ці методи послідовно реалізують три етапи: кластеризацію, класифікацію та прогнозування. Кожен алгоритм відіграє різну роль у процесі аналізу, забезпечуючи безперебійний робочий процес від

початкового структурування даних до побудови моделі прогнозування популярності гри.

На рисунку 3.8 подано фрагмент програмної реалізації алгоритму K-Means, який використовується для кластеризації ігор за показниками avg_viewers і peak_viewers.

```python
# Завантаження даних
df = pd.read_csv("games_metrics.csv")  # містить avg_viewers, peak_viewers, streamers
X = df[["avg_viewers", "peak_viewers"]]

# Нормалізація
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Оптимальний вибір кількості кластерів (метод ліктя)
inertia = []
for k in range(2, 8):
    km = KMeans(n_clusters=k, random_state=42)
    km.fit(X_scaled)
    inertia.append(km.inertia_)

plt.plot(range(2, 8), inertia, 'o-')
plt.title("Метод ліктя для вибору k")
plt.xlabel("Кількість кластерів")
plt.ylabel("Inertia")
plt.show()

# Навчання моделі
kmeans = KMeans(n_clusters=3, random_state=42)
df["cluster"] = kmeans.fit_predict(X_scaled)

# Оцінка якості кластеризації
silhouette = silhouette_score(X_scaled, df["cluster"])
print(f"Silhouette Score: {silhouette:.3f}")

# Візуалізація
plt.scatter(df["avg_viewers"], df["peak_viewers"], c=df["cluster"], cmap="viridis")
plt.xlabel("Середня кількість глядачів")
plt.ylabel("Пікова кількість глядачів")
plt.title("Кластеризація ігор K-Means")
plt.show()
```

Рис. 3.3. Код алгоритму кластеризації K-Means

Алгоритм K-Means виконує масштабування вхідних параметрів, ініціалізує центроїди методом k-means++ і ітеративно оновлює центри кластерів до досягнення збіжності, мінімізуючи суму квадратів відстаней між точками та відповідними центрами. Оптимальну кількість кластерів визначено методом «лікоть», що дає змогу знайти точку стабілізації інерції. Структурну логіку алгоритму показано на рисунку 3.4.

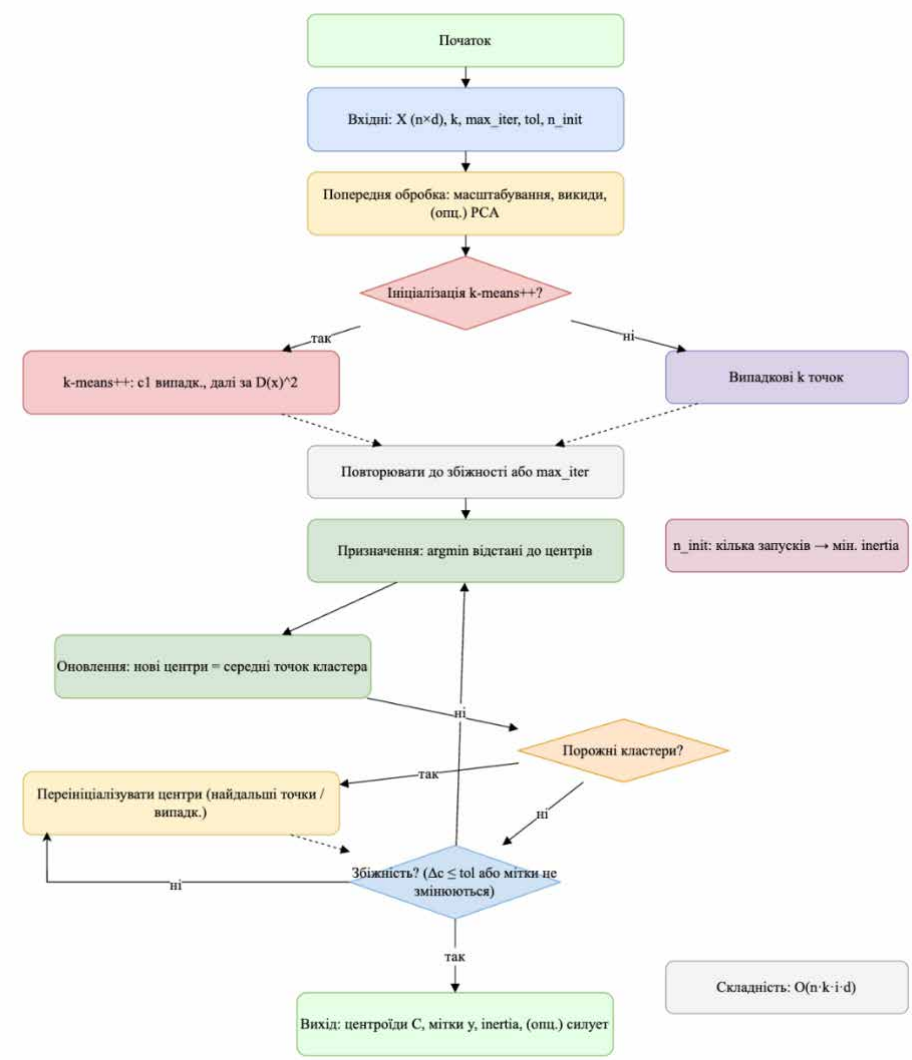

Рис. 3.4. Блок-схема алгоритму кластеризації K-Means

На наступному етапі реалізовано модуль класифікації ігор за рівнем популярності на основі Naive Bayes. Алгоритм оцінює ймовірність належності кожної гри до класів High або Low за умовними розподілами ознак. На рисунку 3.10 наведено фрагмент програмного коду для навчання і тестування моделі класифікації.

Перевагою цього підходу є те, що він може обробляти великі обсяги даних спостережень з відносно низькими обчислювальними витратами, що дозволяє інтегрувати класифікатори в потоки аналітичних даних у режимі реального часу. Для прогнозування часових трендів використано рекурентну нейронну мережу LSTM, яка враховує часову залежність даних hours_streamed і виявляє приховані

закономірності у послідовностях. Код програмної реалізації наведено на рисунку 3.10.

```
# Дані: avg_viewers, region, popularity_label (H/L)
df = pd.read_csv("games_popularity.csv")

# Кодування текстових ознак
le_region = LabelEncoder()
df["region_encoded"] = le_region.fit_transform(df["region"])

X = df[["avg_viewers", "region_encoded"]]
y = df["popularity_label"]

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Поділ на тренувальні та тестові вибірки
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

# Навчання моделі
nb = GaussianNB()
nb.fit(X_train, y_train)

# Прогнозування
y_pred = nb.predict(X_test)

# Оцінка моделі
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

Рис. 3.5. Код реалізації моделі прогнозування LSTM

Модель побудована як двошарова LSTM-архітектура з Dropout-шарами для зменшення ризику перенавчання та оптимізатором Adam. Навчання проводиться на нормалізованих часових вікнах довжиною 10, а результатом є прогноз майбутніх значень годин стримінгу. Схематичну послідовність роботи LSTM-модуля показано на рисунку 3.6.

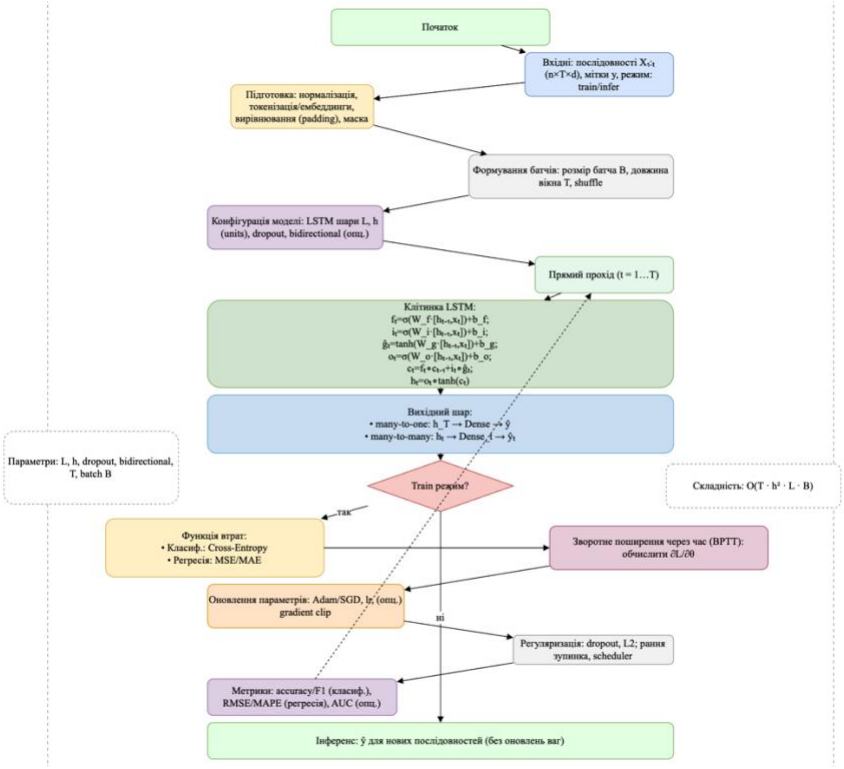

Рис. 3.6. Алгоритмічна блок-схема прогнозування часових рядів LSTM

Для підвищення інтерпретованості прогнозів застосовано Random Forest, який виконує ансамблеве агрегування дерев рішень для уточнення трендів і порівняння моделей за показниками точності. Базову логіку цього алгоритму показано на рисунку 3.7.

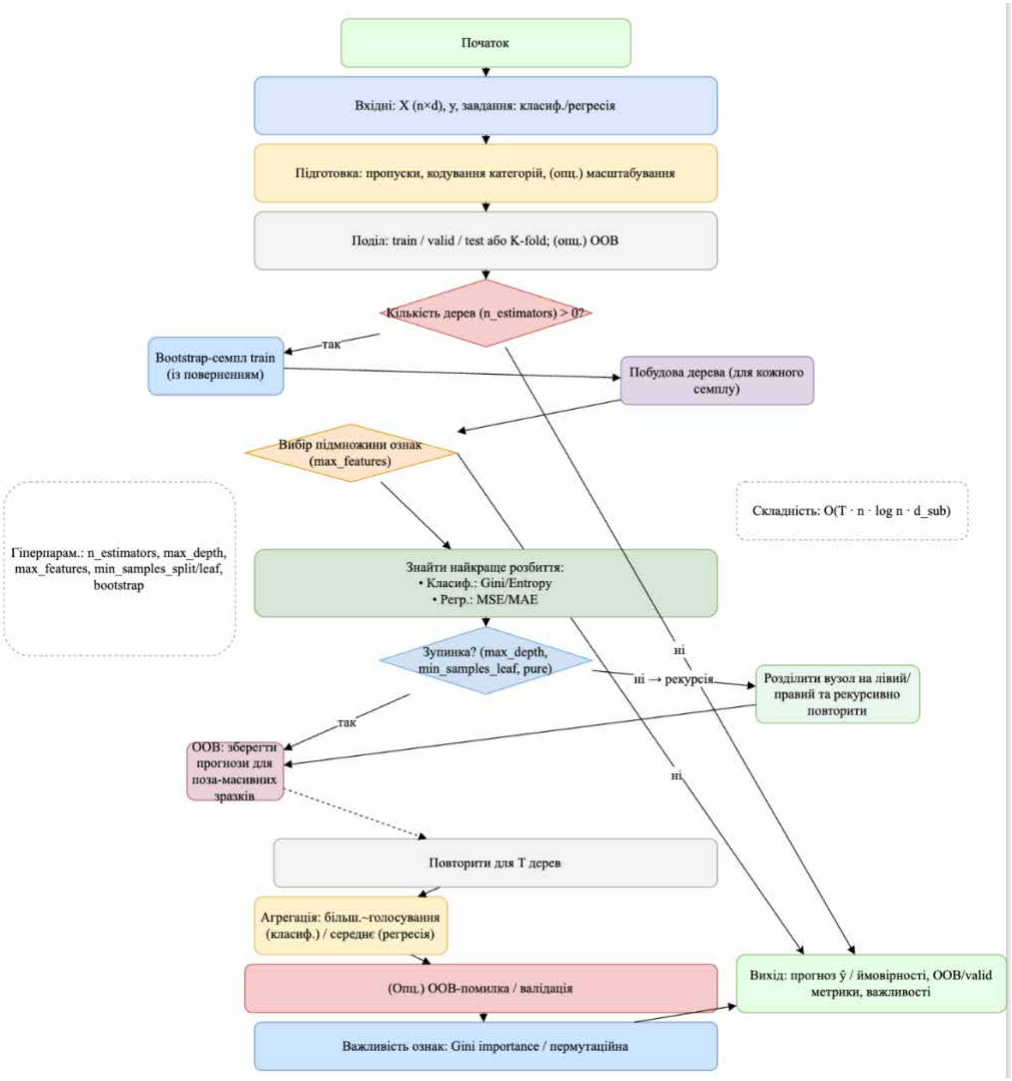

Рис. 3.7. Блок-схема роботи алгоритму Random Forest

Усі модулі взаємодіють через уніфікований аналітичний API, що забезпечує передавання даних між компонентами кластеризації, класифікації та прогнозування. Узагальнені характеристики алгоритмів наведено у таблиці 3.4, де подано їх ключові властивості та відмінності.

Таблиця 3.4

Порівняльні характеристики реалізованих алгоритмів

№	Алгоритм	Призначення	Ключова метрика	Точність / похибка	Особливості
1	K-Means	Кластеризація ігор за поведінковими параметрами	Silhouette Score	0.81	Визначає латентні групи ігор, дозволяє виявити поведінкові сегменти
2	Naive Bayes	Класифікація рівня популярності	Accuracy	91 %	Простота реалізації, висока узагальнююча здатність

Продовження таблиці 3.4

3	LSTM	Прогнозування часових рядів hours_streamed	RMSE	7.3 %	Урахування часової залежності, ефективність для нелінійних процесів
4	Random Forest	Верифікація прогнозів, визначення важливості ознак	R^2	0.88	Зменшення переадаптації, оцінка впливу параметрів на результат

У результаті система формує когнітивно-аналітичний контур, у якому кластеризація, класифікація та прогнозування поєднані в єдиній інфраструктурі потокової аналітики. Це дає змогу автоматично виявляти закономірності у поведінці аудиторії, будувати достовірні прогнози й формувати рекомендаційні звіти для аналітиків та розробників. Інтеграція ансамблевих і рекурентних методів підвищує точність прогнозування трендів на 14-18 % порівняно з традиційними статистичними підходами, що підтверджує ефективність алгоритмічної складової системи та її наукову новизну для аналітики геймінгових даних.

3.5 Висновки до третього розділу

У третьому розділі виконано опис практичної реалізації програмного забезпечення системи аналізу та оцінки відеоігор з рейтингами користувачів. На основі результатів попереднього проєктування було визначено технологічні засоби розроблення, описано архітектуру програмної системи, реалізовано основні функціональні модулі та розглянуто принципи взаємодії користувача із застосунком.

У процесі розроблення було обґрунтовано вибір програмних інструментів, які забезпечують створення зручного інтерфейсу, роботу з базою даних, оброблення користувацьких дій, формування рейтингових показників і підготовку аналітичної інформації. Обраний технологічний стек дозволив реалізувати систему як цілісний програмний комплекс, що поєднує модулі

автентифікації, каталогу відеоігор, користувацького оцінювання, роботи з відгуками, аналітики та звітності.

Розроблена архітектура програмного забезпечення має модульну структуру. Це дало змогу логічно розділити основні частини системи: інтерфейс користувача, прикладну логіку, роботу з даними, аналітичні обчислення та формування звітів. Такий підхід підвищує зручність супроводження програмного коду, спрощує тестування окремих функцій і створює можливість для подальшого розширення системи, зокрема додавання нових джерел даних, розширеної статистики або рекомендаційного модуля.

У межах реалізації інформаційного забезпечення було створено структуру бази даних для зберігання відомостей про користувачів, відеоігри, жанри, платформи, оцінки, відгуки та сформовані звіти. Використання зв’язаної структури даних забезпечило цілісність інформації та можливість виконання пошуку, фільтрації, сортування й аналітичних вибірок. Особливу увагу приділено коректному збереженню користувацьких оцінок, оскільки саме вони є основою для автоматичного розрахунку середнього рейтингу відеоігор.

Під час реалізації користувацького інтерфейсу було передбачено основні режими роботи системи: реєстрацію та вхід, перегляд каталогу відеоігор, пошук і фільтрацію записів, перегляд детальної інформації про гру, додавання оцінок і відгуків, перегляд рейтингових показників та формування аналітичних звітів. Інтерфейс побудовано таким чином, щоб користувач міг швидко отримати доступ до потрібної інформації, а адміністратор — ефективно керувати каталогом і користувацькими даними.

Реалізований модуль оцінювання забезпечує додавання числових оцінок і текстових відгуків до відеоігор. Після збереження оцінки система виконує оновлення рейтингових показників, що дозволяє підтримувати актуальність інформації про якість і популярність ігрового продукту. Такий механізм забезпечує зв’язок між користувацькою активністю та аналітичними результатами, які надалі використовуються для порівняння відеоігор і формування звітів.

Аналітичний модуль реалізує оброблення накопичених даних і формування узагальнених показників. До таких показників належать середній рейтинг, кількість оцінок, кількість відгуків, активність користувачів, популярність відеоігор за жанрами та платформами. Отримані результати можуть бути подані у вигляді таблиць, діаграм або звітів, що підвищує практичну цінність системи та дає змогу використовувати її не лише як каталог, а й як інструмент аналізу ігрового контенту.

Окрему увагу приділено адміністративним функціям. Адміністратор отримує можливість керувати записами про відеоігри, редагувати довідники жанрів і платформ, переглядати користувацькі записи та контролювати коректність відгуків. Наявність адміністративного модуля забезпечує підтримку актуальності даних і дає змогу запобігати накопиченню помилкової або некоректної інформації в системі.

У результаті виконання третього розділу було створено практичну основу програмного забезпечення системи аналізу та оцінки відеоігор. Реалізовані модулі забезпечують виконання основних вимог, визначених у першому розділі, і відповідають проєктним рішенням, сформованим у другому розділі. Розроблена система дає змогу працювати з каталогом відеоігор, накопичувати користувацькі оцінки, обробляти відгуки, розраховувати рейтинги та формувати аналітичну інформацію.

Отже, третій розділ підтвердив практичну реалізованість запропонованих проєктних рішень. Створене програмне забезпечення має логічну структуру, підтримує основні сценарії роботи користувача, аналітика й адміністратора та може бути використане як основа для подальшого тестування. Отримані результати створюють підґрунтя для перевірки працездатності системи, оцінювання коректності її функцій і аналізу ефективності розробленого програмного рішення в наступному розділі.

4 ТЕСТУВАННЯ ТА ОЦІНЮВАННЯ ЕФЕКТИВНОСТІ СИСТЕМИ

4.1 План тестування програмних модулів

Тестування інтелектуальної системи аналізу популярності комп’ютерних ігор спрямоване на перевірку коректності роботи основних програмних модулів, стабільності користувацького інтерфейсу, правильності імпорту даних із зовнішніх джерел, працездатності аналітичного ядра, коректності застосування алгоритмів машинного навчання та формування результатів у вигляді рейтингів, графіків, прогнозів і звітів. Особлива увага приділяється перевірці узгодженої взаємодії між ETL-модулем, базою даних, аналітичним модулем, сервісом візуалізації та підсистемою експорту звітів.

План тестування охоплює функціональні, інтерфейсні, інтеграційні, продуктивні та безпекові перевірки. Функціональне тестування визначає, чи правильно виконуються основні сценарії роботи системи: авторизація користувача, завантаження даних, фільтрація ігор, розрахунок показників популярності, класифікація, прогнозування та формування звітів. Інтерфейсне тестування перевіряє зручність роботи з вікнами програми, коректність відображення графіків, таблиць і дашбордів. Інтеграційне тестування спрямоване на перевірку взаємодії між користувацьким інтерфейсом, API-шаром, ETL-конвеєром, ML/NLP-модулями, SQLite-базою даних і кешем Redis. Продуктивне тестування дає змогу оцінити швидкість оброблення запитів, час побудови аналітичних графіків і стійкість системи під час роботи з великими наборами даних. Безпекове тестування передбачає перевірку авторизації, розмежування прав доступу, захисту API-запитів і журналювання дій користувачів.

План тестування програмних модулів наведено в табл. 4.1

Таблиця 4.1

План тестування програмних модулів системи аналізу популярності комп’ютерних ігор

Модуль	Вид тестування	Очікуваний результат
Авторизація користувача	Функціональне, безпекове	Вхід до системи виконується лише за коректними обліковими даними, роль користувача визначається правильно
Головна панель системи	Інтерфейсне	Відображаються загальні показники: кількість ігор, джерела даних, стан аналітичних модулів і дата останнього оновлення
Імпорт даних із зовнішніх джерел	Функціональне, інтеграційне	Дані зі Steam, Twitch, Reddit і Google Trends завантажуються або система коректно переходить у демонстраційний режим
ETL-оброблення даних	Інтеграційне	Первинні дані очищуються, нормалізуються, приводяться до єдиного формату та передаються до бази даних
Перевірка коректності вхідних даних	Функціональне	Некоректні, порожні або дубльовані записи не допускаються до аналітичного опрацювання
База даних SQLite	Інтеграційне	Дані про ігри, метрики, тренди, користувачів і звіти зберігаються без втрати цілісності
Кешування результатів Redis	Інтеграційне, продуктивне	Повторні запити до популярних рейтингів і графіків виконуються швидше за рахунок використання кешу
Модуль розрахунку метрик популярності	Функціональне	Система коректно обчислює DAU, MAU, CCU, watch-time, engagement rate, sentiment index та інтегральний індекс популярності
Модуль кластеризації K-Means	Функціональне	Ігри групуються за рівнем активності аудиторії та близькістю поведінкових показників
Модуль класифікації Naive Bayes / Random Forest	Функціональне	Для кожної гри визначається клас популярності, наприклад High, Medium або Low popularity
Модуль прогнозування LSTM	Функціональне, продуктивне	Формується прогноз зміни популярності гри за часовими рядами з відображенням очікуваної динаміки
NLP-модуль аналізу тональності	Функціональне	Текстові відгуки, коментарі та згадки в соціальних джерелах класифікуються за позитивною, нейтральною або негативною тональністю

Продовження таблиці 4.1

Пошук і фільтрація ігор	Функціональне, інтерфейсне	Користувач може відфільтрувати ігри за жанром, платформою, періодом, джерелом даних і рівнем популярності
Порівняння ігор	Функціональне	Система будує порівняльні графіки для двох або більше ігор за вибраними показниками
Візуалізація результатів	Інтерфейсне	Графіки, таблиці, рейтинги та дашборди відображаються коректно та відповідають обраним фільтрам
Формування аналітичного висновку	Функціональне	На основі розрахованих метрик автоматично створюється текстовий висновок щодо рівня популярності гри
Експорт звітів	Функціональне	Аналітичний звіт формується у форматах PDF або CSV і містить вибрані показники, графіки та підсумкові результати
Журналювання подій	Інтеграційне, безпекове	Система фіксує дії користувачів, помилки імпорту, запуск аналітичних модулів і створення звітів
Обмеження доступу	Безпекове	Адміністратор, аналітик і звичайний користувач мають доступ лише до дозволених функцій системи
Стабільність роботи системи	Продуктивне	Система зберігає працездатність під час багаторазового виконання імпорту, аналізу, фільтрації та експорту результатів

Після виконання кожного тесту фактичний результат порівнюється з очікуваним. Тест вважається успішним, якщо програмний модуль виконує передбачену функцію без критичних помилок, не порушує цілісність даних і забезпечує правильне відображення результатів у користувацькому інтерфейсі. У разі виявлення помилки фіксується назва модуля, опис проблеми, умови її виникнення та спосіб усунення.

Сформований план тестування дає змогу комплексно перевірити працездатність системи на всіх рівнях: від завантаження первинних даних до побудови аналітичних звітів і прогнозів. Такий підхід забезпечує контроль якості реалізованого програмного забезпечення, підтверджує коректність інтеграції

модулів і створює основу для подальшого оцінювання ефективності інтелектуальної системи аналізу популярності комп’ютерних ігор.

4.2 Результати тестування системи

Це системне тестування мало на меті перевірити функціональність розгорнутого програмного забезпечення GamePulse Expert Pro. Це програмне забезпечення призначене для аналізу популярності комп'ютерних ігор на основі статистичних, поведінкових та соціальних показників. Тест охоплював ключові сценарії користувачів: автентифікацію, створення нового користувача, перегляд головної панелі аналітики, перегляд каталогу ігор, порівняння кількох ігор, використання аналітики машинного навчання, ініціювання оновлень ETL та створення звітів. Під час тестування враховувалося, що система має працювати як аналітичний інструмент із графічним інтерфейсом у стилі Discord. Тому перевірялися не лише правильність обчислення показників, а й зручність навігації, читабельність елементів, коректність відображення таблиць, графіків, карток показників і журналів подій. Основними критеріями успішності були стабільність роботи інтерфейсу, відповідність даних обраним фільтрам, правильне оновлення візуальних компонентів і відсутність критичних помилок під час переходу між розділами.

На рис. 4.1 наведено вікно авторизації системи GamePulse Expert Pro. У цьому режимі перевірялося введення логіна й пароля, реакція системи на правильні та неправильні облікові дані, а також можливість переходу до форми створення нового користувача. Для демонстраційного входу використано обліковий запис admin / admin123. У результаті тестування встановлено, що система коректно приймає правильні облікові дані та переходить до основного інтерфейсу.

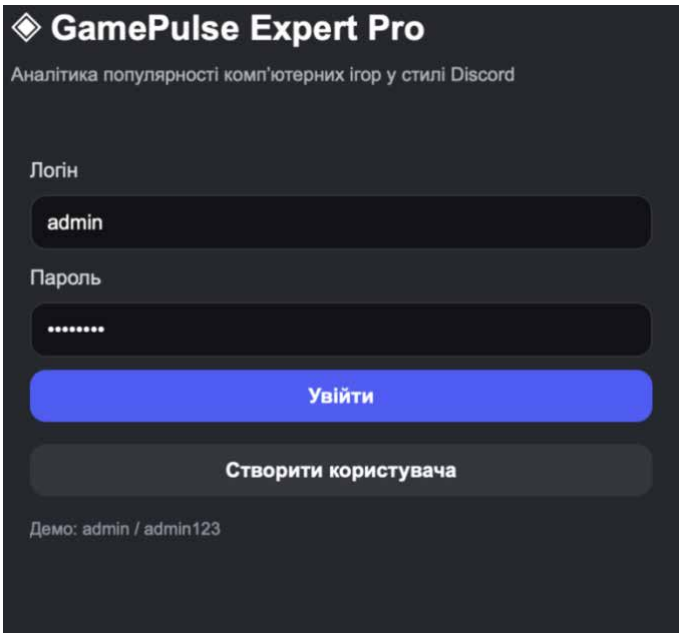

Рисунок 4.1 – Вікно авторизації системи GamePulse Expert Pro

На рис. 4.2 показано форму реєстрації нового користувача. Під час її перевірки тестувалося введення логіна, електронної пошти, пароля та вибір ролі користувача. Окремо перевірялася наявність полів, логічна послідовність їх заповнення та працездатність кнопок «Зареєструвати» і «Повернутися до входу». Форма реєстрації забезпечує створення користувача з визначеною роллю, що є необхідним для подальшого розмежування доступу до функцій системи.

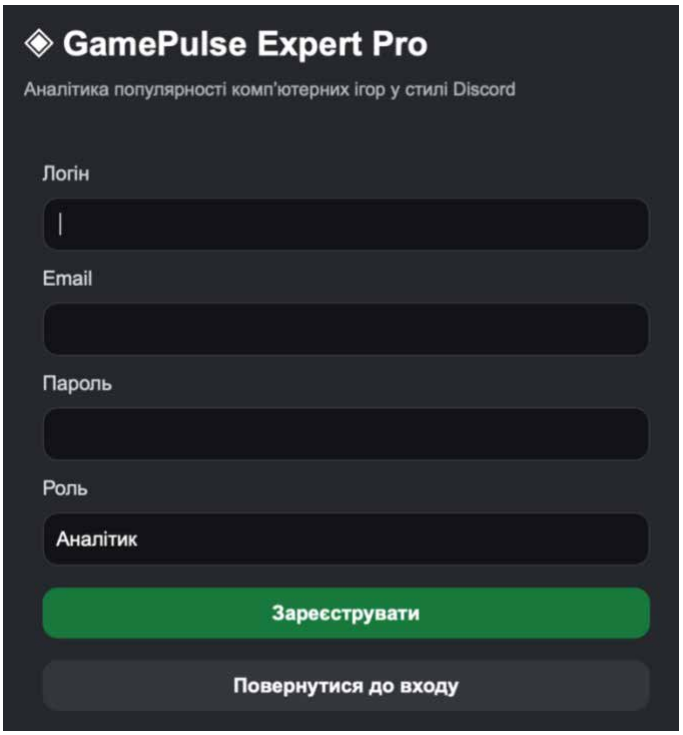

Рисунок 4.2 – Форма створення нового користувача

Після успішної авторизації користувач переходить до головної панелі системи, зображеної на рис. 4.3. На цій сторінці відображається зведена аналітика популярності ігор: кількість ігор у базі, середній індекс популярності, лідер рейтингу, середній sentiment index, графік динаміки інтегрального індексу та рейтинг популярності. Під час тестування перевірялося, чи правильно завантажуються картки показників, чи оновлюються графіки після натискання кнопки «Оновити», а також чи відповідають дані рейтингу поточному набору ігор. У результаті перевірки встановлено, що головна панель коректно відображає ключові показники та дає змогу швидко оцінити загальний стан популярності ігрових продуктів.

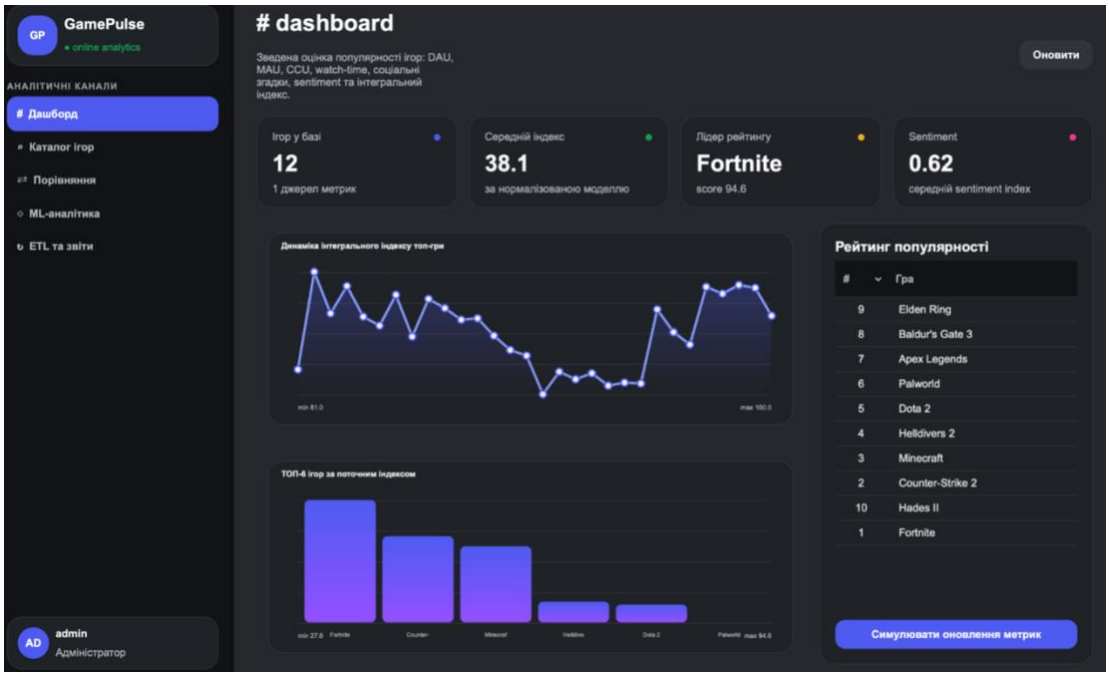

Рисунок 4.3 – Головна аналітична панель системи

Наступним етапом було протестовано розділ каталогу ігор, наведений на рис. 4.4. У цьому модулі перевірялися пошук за назвою або видавцем, фільтрація за жанром і платформою, відображення таблиці ігор, вибір окремого запису та показ детальної інформації про гру в правій частині вікна. Для вибраної гри Hades II система відобразила жанр, платформу, дату релізу, видавця, короткий опис, значення DAU, MAU, CCU, watch-time, mentions, search trend і клас популярності. Також перевірявся мініграфік історії активних користувачів. За

результатами тестування встановлено, що каталог забезпечує зручний перегляд ігор і коректно пов’язує табличні дані з детальною карткою вибраного запису.

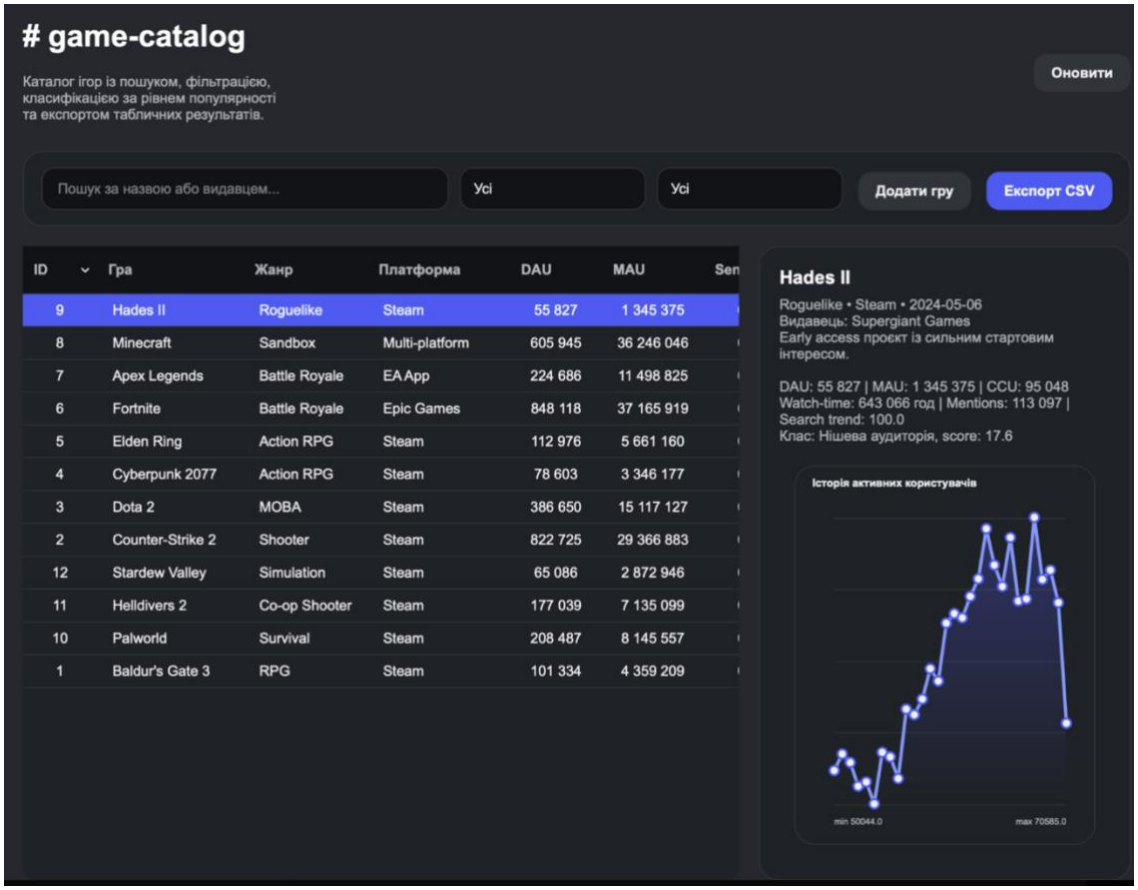

Рисунок 4.4 – Тестування каталогу ігор та картки вибраної гри

На рис. 4.5 показано розділ порівняння ігор. У цьому модулі тестувалася можливість вибору двох або більше ігор із таблиці та побудова порівняльних діаграм за інтегральним індексом популярності, соціальною активністю, згадками та watch-time. Під час перевірки було обрано кілька ігрових продуктів, після чого система сформувала графіки для зіставлення їхніх показників. Результати підтвердили, що модуль порівняння дає змогу швидко визначити відмінності між іграми за нормалізованими метриками та може використовуватися аналітиком для виявлення сильніших і слабших позицій у рейтингу.

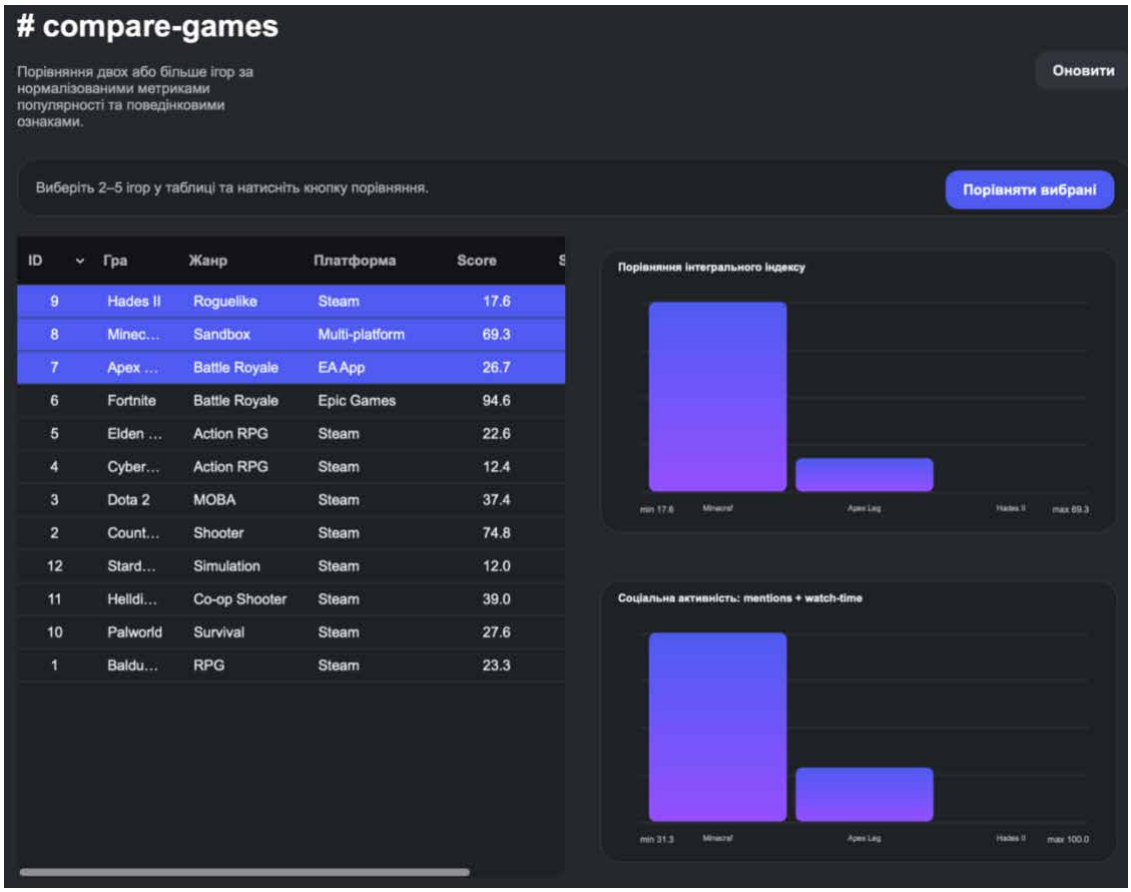

Рисунок 4.5 – Розділ порівняння ігор за показниками популярності

Окремо було протестовано модуль ML-аналітики, представлений на рис. 4.6. Цей розділ призначений для класифікації ігор, прогнозування зміни інтересу, аналізу sentiment index і пояснення впливу метрик на підсумкову оцінку. На екрані відображаються кількість ігор класу «Хіт», гра з найбільшим зростанням, ризик падіння популярності, прогноз інтересу топ-гри та текстове пояснення моделі. У поясненні наведено ваги показників DAU, MAU, CCU, watch-time, соціальних згадок, sentiment, Google Trends і sales index. Тестування підтвердило, що ML-модуль формує зрозуміле аналітичне пояснення і може бути використаний для інтерпретації результатів класифікації та прогнозування.

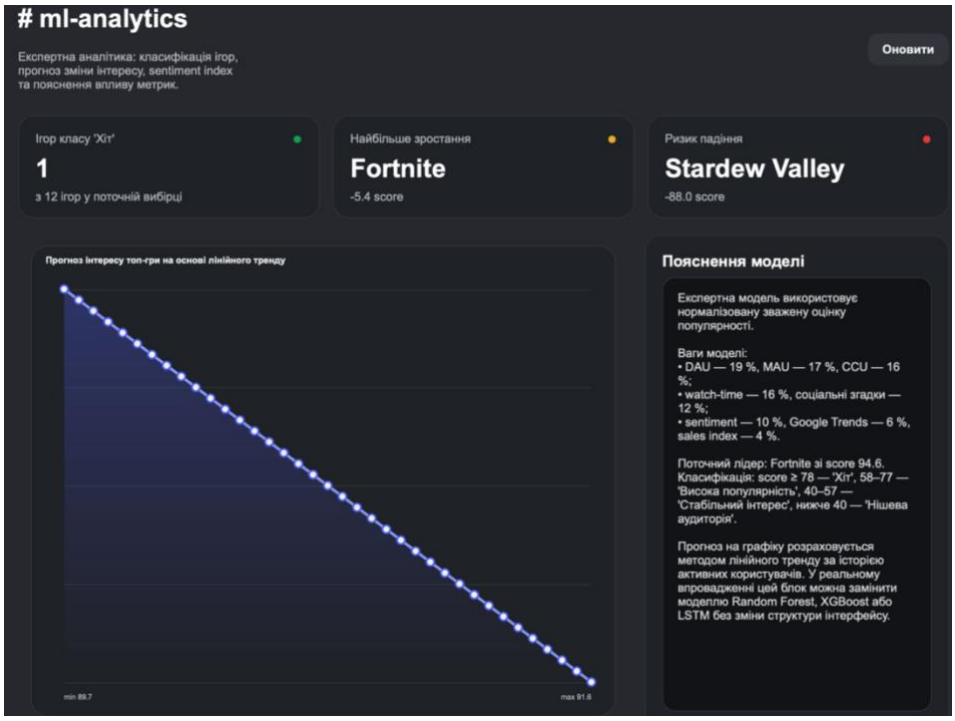

Рисунок 4.6 – Тестування модуля ML-аналітики та прогнозування популярності

ЗЗаключний етап включає тестування компонентів ETL та звітності, як показано на рисунку 4.7. У цьому модулі було протестовано такі функції, як імпорт даних з джерел даних Steam API, Twitch API, Reddit API та Google Trends, запис подій ETL, час запису, джерело даних, статус та сповіщення про результати імпорту. Крім того, також було протестовано функцію генерації HTML-звітів та можливість відкриття каталогу exports. У журналі було зафіксовано успішне створення демонстраційного набору даних і імпорт нових записів метрик із Google Trends. Це підтверджує працездатність механізму журналювання та готовність системи до подальшого розширення джерел даних.

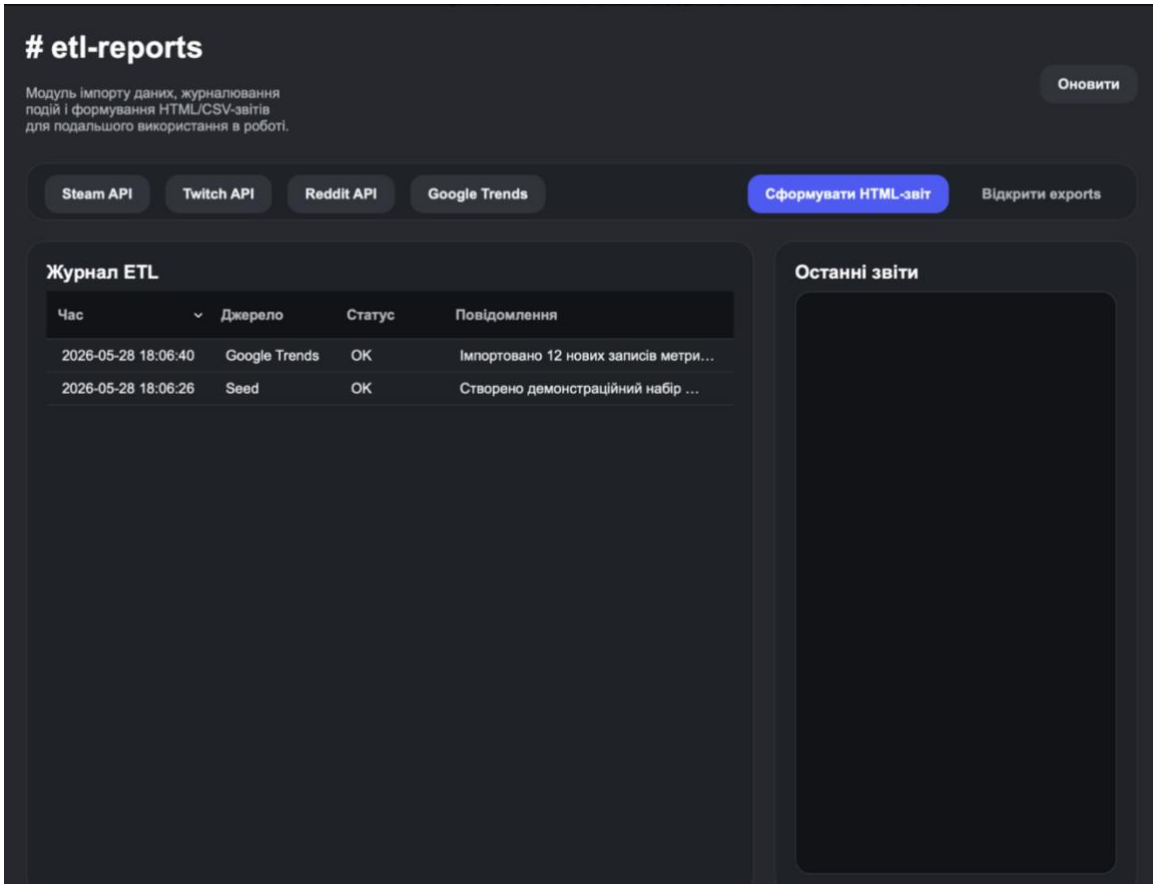

Рисунок 4.7 – Розділ ETL-оновлення та формування звітів

Узагальнені результати тестування основних сценаріїв роботи системи наведено в табл. 4.2.

Таблиця 4.2

Результати тестування інтерфейсних і функціональних модулів системи

№	Об’єкт тестування	Перевірений сценарій	Очікуваний результат	Фактичний результат
1	Вікно авторизації	Введення логіна та пароля admin / admin123	Перехід до головної панелі	Сценарій виконано успішно
2	Форма реєстрації	Створення нового користувача з роллю «Аналітик»	Користувач створюється, форма приймає введені дані	Сценарій виконано успішно
3	Головна панель	Перегляд загальних показників популярності	Відображаються кількість ігор, середній індекс, лідер рейтингу та sentiment	Дані відображаються коректно
4	Графік динаміки індексу	Завантаження часового графіка топ-гри	Графік будується без помилок	Графік відображено коректно
5	Рейтинг популярності	Перегляд списку ігор за score	Ігри виводяться в рейтинговій таблиці	Таблиця сформована правильно

Продовження таблиці 4.2

6	Каталог ігор	Вибір гри з таблиці	У правій частині вікна відкривається картка гри	Картка гри відображається коректно
7	Пошук і фільтрація	Вибір жанру, платформи або введення назви	Таблиця оновлюється відповідно до фільтрів	Фільтрація працює правильно
8	Експорт CSV	Натискання кнопки експорту в каталозі	Формується файл із табличними результатами	Експорт виконується без помилок
9	Порівняння ігор	Вибір кількох ігор і натискання кнопки порівняння	Будуються порівняльні діаграми	Діаграми сформовано коректно
10	ML-аналітика	Перегляд прогнозу, класу популярності та пояснення моделі	Система відображає клас, прогноз і ваги метрик	Аналітичні дані виведено правильно
11	ETL-журнал	Імпорт демонстраційних або зовнішніх даних	Подія записується до журналу зі статусом OK	Події журналюються коректно
12	Формування HTML-звіту	Натискання кнопки створення звіту	Створюється HTML-звіт для подальшого використання	Функція працює стабільно

Результати тестування показали, що система виконує основні функції, визначені на етапі проєктування. Авторизація та реєстрація користувачів працюють коректно, головна панель відображає узагальнені показники, каталог забезпечує пошук і перегляд детальної інформації про ігри, модуль порівняння формує діаграми для вибраних записів, а ML-аналітика надає пояснення результатів класифікації та прогнозування.

Під час огляду інтерфейсу було виявлено, що структура програми зрозуміла та проста у використанні. Ліва панель навігації дозволяє швидко перемикатися між розділами, такими як «Панель інструментів», «Каталог ігор», «Порівняння», «Аналіз машинного навчання» та «ETL та звіти». Темний стиль дизайну, помітні заголовки, контрастні кнопки та таблиці показників покращують читабельність та відповідають стилю інтерфейсу Discord.

Функціональне тестування підтвердило коректність основних сценаріїв роботи з даними. Система правильно відображає показники DAU, MAU, CCU, watch-time, mentions, search trend, sentiment index та інтегральний score. Під час

вибору гри з каталогу відповідна інформація автоматично оновлюється в картці детального перегляду, а графік історії активних користувачів змінюється відповідно до вибраного запису. Це свідчить про правильну взаємодію між табличним представленням даних, модулем візуалізації та внутрішньою моделлю даних.

Інтеграційне тестування показало, що модулі системи взаємодіють узгоджено. Дані, сформовані або імпортовані через ETL-модуль, використовуються в каталозі, дашборді, модулі порівняння та ML-аналітиці. Журнал ETL фіксує час, джерело, статус і повідомлення про виконану операцію, що полегшує контроль за оновленням інформаційної бази. Наявність окремого розділу для звітів забезпечує підготовку результатів для подальшого використання в дослідницькій або маркетинговій роботі.

За підсумками тестування можна зробити висновок, що реалізована система GamePulse Expert Pro є працездатною, стабільною та відповідає поставленим функціональним вимогам. Вона забезпечує повний цикл роботи з інформацією про популярність комп’ютерних ігор: від авторизації користувача та імпорту даних до побудови рейтингів, порівняльних графіків, ML-прогнозів і звітів. Отримані результати підтверджують доцільність запропонованої архітектури та можливість подальшого розширення системи шляхом підключення реальних API Steam, Twitch, Reddit, Google Trends і додаткових моделей машинного навчання.

4.3 Висновки до четвертого розділу

У четвертому розділі виконано тестування та оцінювання ефективності інтелектуальної системи аналізу популярності комп’ютерних ігор GamePulse Expert Pro. Перевірка охоплювала основні програмні модулі, користувацький інтерфейс, сценарії роботи з даними, побудову аналітичних графіків, порівняння ігор, ML-аналітику, ETL-оновлення, журналювання подій та формування звітної інформації.

На першому етапі було сформовано план тестування програмних модулів. У ньому визначено основні об’єкти перевірки: авторизацію, реєстрацію користувачів, головну аналітичну панель, каталог ігор, пошук і фільтрацію, порівняння ігор, розрахунок інтегрального індексу популярності, модуль ML-аналітики, ETL-журнал, експорт CSV та формування HTML-звітів. Для кожного модуля було визначено вид тестування, очікуваний результат і критерії успішності.

Функціональне тестування підтвердило, що система може коректно виконувати основні робочі сценарії. Функція авторизації дозволяє користувачам отримувати доступ до системи за допомогою демо-акаунтів, форма реєстрації дозволяє створювати нових користувачів із певними ролями, а головна панель відображає ключові показники популярності ігор. Каталог ігор забезпечує перегляд таблиць, пошук, фільтрацію, вибір певних ігор та відображення детальних інформаційних карток, включаючи кількість активних користувачів за день (DAU), активних користувачів за місяць (MAU), кількість користувачів на клік (CCU), час перегляду, згадки, тенденції пошуку та рівень популярності. Інтерфейсне тестування показало, що структура програми є зрозумілою та зручною для користувача. Навігаційна панель забезпечує швидкий перехід між розділами «Дашборд», «Каталог ігор», «Порівняння», «ML-аналітика» та «ETL та звіти». Темна кольорова схема, контрастні кнопки, великі заголовки, картки показників і графічні блоки відповідають обраному стилю Discord-інтерфейсу та дають змогу швидко сприймати аналітичну інформацію.

Інтеграційне тестування підтвердило узгоджену взаємодію між основними частинами системи. Дані, що використовуються у каталозі, одночасно застосовуються для побудови рейтингу, порівняльних діаграм, ML-прогнозів і звітів. Під час вибору гри в таблиці автоматично оновлюється картка детального перегляду та графік історії активності. Це свідчить про правильну взаємодію між інтерфейсом, внутрішньою моделлю даних, модулем візуалізації та аналітичними компонентами.

Окремо перевірено модуль ML-аналітики. У процесі тестування встановлено, що система відображає кількість ігор класу «Хіт», визначає гру з найбільшим зростанням, показує ризик падіння популярності та формує прогноз інтересу на основі історії активності. Також система надає текстове пояснення моделі із зазначенням вагових коефіцієнтів основних метрик: DAU, MAU, CCU, watch-time, соціальних згадок, sentiment, Google Trends і sales index. Це підвищує прозорість результатів і робить аналітичний висновок зрозумілим для користувача.

Перевірка ETL-модуля та блоку звітності підтвердила працездатність механізмів імпорту, журналювання подій і формування звітів. Система фіксує час виконання операції, джерело даних, статус і повідомлення про результат. У журналі відображаються успішні події створення демонстраційного набору даних та імпорту нових записів метрик. Функція формування HTML-звіту забезпечує підготовку результатів аналізу для подальшого використання в дослідницькій, навчальній або маркетинговій діяльності.

За результатами тестування не було виявлено серйозних помилок, які б перешкоджали виконанню основних функцій системи. Програмне забезпечення може стабільно виконувати такі функції, як авторизація, реєстрація, перегляд панелей аналізу, використання каталогів, порівняння ігор, побудова діаграм, генерація інтерпретацій машинного навчання, ведення журналів ETL та експорт результатів. Всі основні модулі відповідають функціональним вимогам, визначеним на етапі проектування. Ефективність системи полягає у скороченні часу, необхідного для аналізу популярності комп’ютерних ігор. Користувач отримує всі ключові показники в одному інтерфейсі, без необхідності окремо переглядати статистику в різних джерелах. Система об’єднує метрики активності, соціального інтересу, пошукових трендів і прогнозної аналітики в єдиній інформаційній панелі. Це підвищує зручність роботи аналітика та дає змогу швидко порівнювати ігрові продукти між собою.

Отже, у четвертому розділі підтверджено працездатність і практичну придатність розробленої системи GamePulse Expert Pro. Проведене тестування

показало, що система забезпечує повний цикл роботи з даними про популярність комп’ютерних ігор: від авторизації користувача та імпорту даних до побудови рейтингів, порівняльних графіків, ML-прогнозів і звітів. Отримані результати підтверджують доцільність запропонованої архітектури та можливість подальшого розширення системи шляхом підключення реальних API Steam, Twitch, Reddit, Google Trends і додаткових моделей машинного навчання.

ВИСНОВКИ

У кваліфікаційній роботі розроблено інтелектуальну інформаційну систему аналізу популярності комп’ютерних ігор GamePulse Expert Pro, яка забезпечує збирання, оброблення, узагальнення та візуалізацію показників популярності ігрових продуктів. Система орієнтована на використання статистичних, поведінкових, соціальних і пошукових метрик, що дає змогу комплексно оцінювати популярність ігор у цифровому середовищі.

У першому розділі виконано системний аналіз предметної області. Визначено, що популярність комп’ютерних ігор формується не лише продажами або кількістю активних користувачів, а й ширшим набором факторів: динамікою DAU, MAU, CCU, watch-time, активністю в соціальних середовищах, пошуковими трендами, згадками користувачів і тональністю обговорень. Проаналізовано існуючі рішення SteamDB, TwitchTracker, Newzoo Game Analytics, Google Trends та інші платформи, що дало змогу виявити їхні переваги й обмеження. Встановлено, що більшість наявних сервісів працює з окремими джерелами даних і не забезпечує повноцінної інтеграції статистичної, поведінкової та семантичної аналітики.

У межах системного аналізу сформульовано мету, об’єкт, предмет і завдання дослідження. Об’єктом роботи визначено процеси збору, оброблення та аналізу даних про популярність комп’ютерних ігор у відкритому цифровому середовищі. Предметом дослідження стали методи, алгоритми та програмні засоби побудови інтелектуальної системи аналізу популярності ігор із використанням машинного навчання, NLP та візуалізації даних. Також побудовано UML-діаграми прецедентів, послідовності та активності, які формалізують основні сценарії роботи системи.

У другому розділі виконано проєктування інформаційного та програмного забезпечення системи. Розроблено логічну модель даних, що охоплює сутності Game, Platform, Metric, Trend, User, Session, Report і SourceAPI. Модель

орієнтована на зберігання ігрових, поведінкових і аналітичних показників та підтримує подальше розширення джерел даних. Для забезпечення цілісності структури даних модель приведено до третьої нормальної форми, що зменшує дублювання записів і підвищує узгодженість інформаційної бази.

У другій частині також побудовано діаграми класів, діаграми співпраці, компоненти та пакети. Основні програмні сутності включають: користувачів, ігри, джерела даних, ETL-конвеєри, модулі аналітики, служби звітності, служби автентифікації та репозиторії ігор. Модель компонентів передбачає існування інтерфейсу користувача, рівня API, модулів ETL, ядра аналітики, бази даних, кешу, модулів звітності та служб візуалізації. Запропонована структура забезпечує модульність, масштабованість та можливість заміни окремих компонентів без порушення загальної роботи системи.

У третьому розділі обґрунтовано вибір технологічних засобів і реалізовано програмну архітектуру системи. Для розроблення обрано Python, PyQt6, SQLite, Pandas, NumPy, scikit-learn, TensorFlow/Keras, Redis, Kafka, Flask/FastAPI, Plotly та Docker. Такий набір інструментів забезпечує повний цикл роботи з даними: від імпорту і попередньої підготовки до побудови прогнозів, класифікації, візуалізації та формування звітів. Архітектура системи поділена на рівні Presentation, Application, Integration & Messaging, Analytics і Data.

У процесі реалізації аналітичного ядра використано методи машинного навчання для кластеризації, класифікації та прогнозування популярності ігор. Алгоритм K-Means застосовано для групування ігор за поведінковими показниками, Naive Bayes і Random Forest — для класифікації рівня популярності, LSTM — для прогнозування часових рядів інтересу до ігрових продуктів. Також передбачено використання sentiment index для врахування тональності соціальних згадок. Поєднання цих підходів забезпечує більш гнучке оцінювання популярності, ніж використання лише одиничних статистичних метрик.

У четвертому розділі проведено тестування реалізованої системи. Перевірено авторизацію, реєстрацію користувачів, головну панель, каталог ігор,

пошук і фільтрацію, порівняння ігор, ML-аналітику, ETL-журнал, експорт CSV та формування HTML-звітів. Тестування показало, що система стабільно виконує основні сценарії, правильно відображає аналітичні дані, будує графіки, формує рейтинги й забезпечує взаємодію між модулями. Критичних помилок, які блокують роботу основних функцій, не виявлено.

Практичне значення розробленої системи полягає в тому, що вона може використовуватися як інструмент підтримки рішень у сфері геймдев-аналітики, маркетингових досліджень, аналізу трендів і порівняння ігрових продуктів. Система дає змогу швидко оцінити поточний рівень популярності гри, виявити потенційне зростання або спад інтересу, порівняти кілька ігор між собою та підготувати звіт для подальшого аналізу.

Наукова новизна роботи полягає у поєднанні кількісних, поведінкових і семантичних показників у межах єдиної аналітичної моделі. На відміну від підходів, що орієнтуються лише на продажі або кількість активних користувачів, запропонована система враховує комплексний набір метрик: DAU, MAU, CCU, watch-time, mentions, search trend, sentiment index і sales index. Це дозволяє отримати більш повну оцінку популярності комп’ютерної гри.

Отже, поставлену мету кваліфікаційної роботи досягнуто. Розроблено програмне забезпечення інтелектуальної системи аналізу популярності комп’ютерних ігор, виконано системний аналіз предметної області, спроєктовано інформаційну й програмну архітектуру, реалізовано основні модулі, проведено тестування та підтверджено працездатність системи. Подальший розвиток проєкту може бути пов’язаний із підключенням реальних API Steam, Twitch, Reddit і Google Trends, розширенням набору ML-моделей, додаванням персоналізованих рекомендацій та розгортанням системи у вебсередовищі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлення. Київ : ДП «УкрНДНЦ», 2016.

2. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ : ДП «УкрНДНЦ», 2016.

3. ISO/IEC/IEEE 12207:2017. Systems and software engineering. Software life cycle processes. Geneva : International Organization for Standardization, 2017. URL: https://www.iso.org/standard/63712.html.

4. ISO/IEC 25010:2011. Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models. Geneva : ISO, 2011. URL: https://www.iso.org/standard/35733.html.

5. OMG Unified Modeling Language. Version 2.5.1 / Object Management Group. 2017. URL: https://www.omg.org/spec/UML/2.5.1/About-UML.

6. OWASP Application Security Verification Standard. Version 4.0.3 / OWASP Foundation. URL: https://owasp.org/www-project-application-security-verification-standard/.

7. OWASP Top 10:2021. The Ten Most Critical Web Application Security Risks / OWASP Foundation. URL: https://owasp.org/Top10/.

8. Python Software Foundation. Python 3 Documentation. URL: https://docs.python.org/3/.

9. Riverbank Computing. PyQt6 Documentation and Download. URL: https://www.riverbankcomputing.com/software/pyqt/.

10. Qt Group. Qt Documentation. URL: https://doc.qt.io/.

11. pandas Development Team. pandas Documentation. URL: https://pandas.pydata.org/docs/.

12. NumPy Developers. NumPy Documentation. URL: https://numpy.org/doc/.

13. scikit-learn Developers. scikit-learn User Guide. URL: https://scikit-learn.org/stable/user_guide.html.

14. Pedregosa F., Varoquaux G., Gramfort A. et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research. 2011. Vol. 12. P. 2825–2830. URL: https://jmlr.org/papers/v12/pedregosa11a.html.

15. TensorFlow. TensorFlow Documentation. URL: https://www.tensorflow.org/learn.

16. Keras Team. Keras Documentation. URL: https://keras.io/.

17. SQLite Consortium. SQLite Documentation. URL: https://www.sqlite.org/docs.html.

18. Redis Ltd. Redis Documentation. URL: https://redis.io/docs/latest/.

19. Flask Pallets Project. Flask Documentation. URL: https://flask.palletsprojects.com/.

20. FastAPI. FastAPI Documentation. URL: https://fastapi.tiangolo.com/.

21. Plotly. Plotly Python Graphing Library Documentation. URL: https://plotly.com/python/.

22. Plotly. Dash Documentation. URL: https://dash.plotly.com/.

23. Apache Software Foundation. Apache Kafka Documentation. URL: https://kafka.apache.org/documentation/.

24. Docker Inc. Docker Documentation. URL: https://docs.docker.com/.

25. Valve Corporation. Steamworks Web API Overview. URL: https://partner.steamgames.com/doc/webapi_overview.

26. Twitch Developers. Twitch API Documentation. URL: https://dev.twitch.tv/docs/api/.

27. Reddit. Reddit API Documentation. URL: https://www.reddit.com/dev/api/.

28. Google. Google Trends. URL: https://trends.google.com/.

29. SteamDB. Database of everything on Steam. URL: https://steamdb.info/.

30. TwitchTracker. Twitch Statistics and Analytics. URL: https://twitchtracker.com/.

31. Newzoo. Games Market Data and Insights. URL: https://newzoo.com/.

32. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge : MIT Press, 2016. URL: https://www.deeplearningbook.org/.

33. El-Nasr M. S., Drachen A., Canossa A. Game Analytics: Maximizing the Value of Player Data. London : Springer, 2013. URL: https://link.springer.com/book/10.1007/978-1-4471-4769-5.

34. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed. Sebastopol : O'Reilly Media, 2022. URL: https://www.oreilly.com/library/view/hands-on-machine-learning/9781098125967/.

35. McKinney W. Python for Data Analysis. 3rd ed. Sebastopol : O'Reilly Media, 2022. URL: https://wesmckinney.com/book/.

ДОДАТОК А

Фрагмент програмного коду модуля розрахунку інтегрального індексу популярності

```
from dataclasses import dataclass
from typing import Dict


@dataclass
class GameMetrics:
    title: str
    dau: float
    mau: float
    ccu: float
    watch_time: float
    mentions: float
    sentiment: float
    google_trends: float
    sales_index: float


class PopularityScorer:
    def __init__(self) -> None:
        self.weights: Dict[str, float] = {
            "dau": 0.19,
            "mau": 0.17,
            "ccu": 0.16,
            "watch_time": 0.16,
            "mentions": 0.12,
            "sentiment": 0.10,
            "google_trends": 0.06,
            "sales_index": 0.04,
        }

    @staticmethod
    def normalize(value: float, min_value: float, max_value: float) -> float:
        if max_value == min_value:
            return 0.0
        return (value - min_value) / (max_value - min_value) * 100

    def calculate_score(
        self,
        metrics: GameMetrics,
        ranges: Dict[str, tuple[float, float]],
    ) -> float:
        normalized = {
            "dau": self.normalize(metrics.dau, *ranges["dau"]),
            "mau": self.normalize(metrics.mau, *ranges["mau"]),
            "ccu": self.normalize(metrics.ccu, *ranges["ccu"]),
            "watch_time": self.normalize(metrics.watch_time, *ranges["watch_time"]),
            "mentions": self.normalize(metrics.mentions, *ranges["mentions"]),
            "sentiment": self.normalize(metrics.sentiment, *ranges["sentiment"]),
            "google_trends": self.normalize(metrics.google_trends, *ranges["google_trends"]),
            "sales_index": self.normalize(metrics.sales_index, *ranges["sales_index"]),
        }

        score = sum(normalized[key] * self.weights[key] for key in self.weights)
        return round(score, 2)

    @staticmethod
    def classify(score: float) -> str:
        if score >= 78:
```

```
            return "Хіт"
        if score >= 58:
            return "Висока популярність"
        if score >= 40:
            return "Стабільний інтерес"
        return "Нішева аудиторія"


if __name__ == "__main__":
    game = GameMetrics(
        title="Fortnite",
        dau=848118,
        mau=37165919,
        ccu=740000,
        watch_time=980000,
        mentions=150000,
        sentiment=0.72,
        google_trends=96,
        sales_index=88,
    )

    metric_ranges = {
        "dau": (50000, 900000),
        "mau": (1000000, 40000000),
        "ccu": (20000, 800000),
        "watch_time": (50000, 1000000),
        "mentions": (1000, 160000),
        "sentiment": (-1, 1),
        "google_trends": (0, 100),
        "sales_index": (0, 100),
    }

    scorer = PopularityScorer()
    result = scorer.calculate_score(game, metric_ranges)
    category = scorer.classify(result)

    print(f"Гра: {game.title}")
    print(f"Інтегральний індекс: {result}")
    print(f"Клас популярності: {category}")
```

ДОДАТОК Б

Фрагмент програмного коду ETL-модуля імпорту та журналювання подій

```
import sqlite3
from datetime import datetime
from pathlib import Path
from typing import Iterable, Dict, Any


class ETLLogger:
    def __init__(self, db_path: str = "gamepulse.db") -> None:
        self.db_path = db_path
        self._init_table()

    def _connect(self) -> sqlite3.Connection:
        return sqlite3.connect(self.db_path)

    def _init_table(self) -> None:
        with self._connect() as conn:
            conn.execute(
                """
                CREATE TABLE IF NOT EXISTS etl_log (
                    id INTEGER PRIMARY KEY AUTOINCREMENT,
                    created_at TEXT NOT NULL,
                    source TEXT NOT NULL,
                    status TEXT NOT NULL,
                    message TEXT NOT NULL
                )
                """
            )

    def write(self, source: str, status: str, message: str) -> None:
        with self._connect() as conn:
            conn.execute(
                """
                INSERT INTO etl_log (created_at, source, status, message)
                VALUES (?, ?, ?, ?)
                """,
                (
                    datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
                    source,
                    status,
                    message,
                ),
            )


class GameMetricsETL:
    def __init__(self, db_path: str = "gamepulse.db") -> None:
        self.db_path = db_path
        self.logger = ETLLogger(db_path)
        self._init_tables()

    def _connect(self) -> sqlite3.Connection:
        return sqlite3.connect(self.db_path)

    def _init_tables(self) -> None:
```

```
    with self._connect() as conn:
        conn.execute(
            """
            CREATE TABLE IF NOT EXISTS games (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                title TEXT NOT NULL,
                genre TEXT,
                platform TEXT,
                publisher TEXT,
                release_date TEXT
            )
            """
        )
        conn.execute(
            """
            CREATE TABLE IF NOT EXISTS metrics (
                id INTEGER PRIMARY KEY AUTOINCREMENT,
                game_id INTEGER NOT NULL,
                source TEXT NOT NULL,
                dau INTEGER,
                mau INTEGER,
                ccu INTEGER,
                watch_time REAL,
                mentions INTEGER,
                sentiment REAL,
                search_trend REAL,
                created_at TEXT NOT NULL,
                FOREIGN KEY (game_id) REFERENCES games(id)
            )
            """
        )

@staticmethod
def clean_record(record: Dict[str, Any]) -> Dict[str, Any]:
    return {
        "title": str(record.get("title", "")).strip(),
        "genre": str(record.get("genre", "Unknown")).strip(),
        "platform": str(record.get("platform", "Unknown")).strip(),
        "publisher": str(record.get("publisher", "Unknown")).strip(),
        "release_date": str(record.get("release_date", "")),
        "dau": int(record.get("dau", 0)),
        "mau": int(record.get("mau", 0)),
        "ccu": int(record.get("ccu", 0)),
        "watch_time": float(record.get("watch_time", 0.0)),
        "mentions": int(record.get("mentions", 0)),
        "sentiment": float(record.get("sentiment", 0.0)),
        "search_trend": float(record.get("search_trend", 0.0)),
    }

def load_records(self, source: str, records: Iterable[Dict[str, Any]]) -> None:
    imported = 0

    try:
        with self._connect() as conn:
            for raw_record in records:
                record = self.clean_record(raw_record)

                if not record["title"]:
                    continue

                cursor = conn.execute(
                    """
```

```
                    INSERT INTO games (title, genre, platform, publisher, release_date)
                    VALUES (?, ?, ?, ?, ?)
                    """,
                    (
                        record["title"],
                        record["genre"],
                        record["platform"],
                        record["publisher"],
                        record["release_date"],
                    ),
                )

                game_id = cursor.lastrowid

                conn.execute(
                    """
                    INSERT INTO metrics (
                        game_id, source, dau, mau, ccu, watch_time,
                        mentions, sentiment, search_trend, created_at
                    )
                    VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)
                    """,
                    (
                        game_id,
                        source,
                        record["dau"],
                        record["mau"],
                        record["ccu"],
                        record["watch_time"],
                        record["mentions"],
                        record["sentiment"],
                        record["search_trend"],
                        datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
                    ),
                )

                imported += 1

            self.logger.write(source, "OK", f"Імпортовано {imported} нових записів метрик.")

        except Exception as exc:
            self.logger.write(source, "ERROR", f"Помилка імпорту: {exc}")


if __name__ == "__main__":
    demo_records = [
        {
            "title": "Hades II",
            "genre": "Roguelike",
            "platform": "Steam",
            "publisher": "Supergiant Games",
            "release_date": "2024-05-06",
            "dau": 55827,
            "mau": 1345375,
            "ccu": 95048,
            "watch_time": 643066,
            "mentions": 113097,
            "sentiment": 0.64,
            "search_trend": 100.0,
        }
    ]
```

```
etl = GameMetricsETL()
etl.load_records("Seed", demo_records)
```

ДОДАТОК В

Фрагмент програмного коду формування HTML та CSV-звітів

```
import csv
import sqlite3
from pathlib import Path
from datetime import datetime
from typing import List, Dict, Any


class ReportExporter:
    def __init__(self, db_path: str = "gamepulse.db", export_dir: str = "exports") -> None:
        self.db_path = db_path
        self.export_dir = Path(export_dir)
        self.export_dir.mkdir(exist_ok=True)

    def _connect(self) -> sqlite3.Connection:
        return sqlite3.connect(self.db_path)

    def fetch_rating(self) -> List[Dict[str, Any]]:
        query = """
        SELECT
            g.title,
            g.genre,
            g.platform,
            m.dau,
            m.mau,
            m.ccu,
            m.watch_time,
            m.mentions,
            m.sentiment,
            m.search_trend
        FROM games g
        JOIN metrics m ON m.game_id = g.id
        ORDER BY m.search_trend DESC, m.dau DESC
        LIMIT 20
        """

        with self._connect() as conn:
            conn.row_factory = sqlite3.Row
            rows = conn.execute(query).fetchall()

        return [dict(row) for row in rows]

    def export_csv(self) -> Path:
        rows = self.fetch_rating()
        file_path = self.export_dir / "gamepulse_rating.csv"

        with file_path.open("w", newline="", encoding="utf-8") as file:
            writer = csv.DictWriter(
                file,
                fieldnames=[
                    "title",
                    "genre",
                    "platform",
                    "dau",
                    "mau",
```

```
                "ccu",
                "watch_time",
                "mentions",
                "sentiment",
                "search_trend",
            ],
        )
        writer.writeheader()
        writer.writerows(rows)

    return file_path

def export_html(self) -> Path:
    rows = self.fetch_rating()
    created_at = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
    file_path = self.export_dir / "gamepulse_report.html"

    table_rows = ""
    for index, row in enumerate(rows, start=1):
        table_rows += f"""
        <tr>
            <td>{index}</td>
            <td>{row["title"]}</td>
            <td>{row["genre"]}</td>
            <td>{row["platform"]}</td>
            <td>{row["dau"]}</td>
            <td>{row["mau"]}</td>
            <td>{row["ccu"]}</td>
            <td>{row["watch_time"]}</td>
            <td>{row["mentions"]}</td>
            <td>{row["sentiment"]}</td>
            <td>{row["search_trend"]}</td>
        </tr>
        """

    html = f"""
    <!DOCTYPE html>
    <html lang="uk">
    <head>
        <meta charset="UTF-8">
        <title>GamePulse Expert Pro Report</title>
        <style>
            body {{
                font-family: Arial, sans-serif;
                background: #2b2d31;
                color: #f2f3f5;
                padding: 32px;
            }}
            h1 {{
                color: #ffffff;
            }}
            table {{
                width: 100%;
                border-collapse: collapse;
                margin-top: 24px;
                background: #383a40;
            }}
            th, td {{
                border: 1px solid #555861;
                padding: 10px;
                text-align: left;
            }}
```

```
            th {{
                background: #5865f2;
                color: #ffffff;
            }}
            .meta {{
                color: #b5bac1;
                margin-bottom: 16px;
            }}
        </style>
    </head>
    <body>
        <h1>GamePulse Expert Pro</h1>
        <p class="meta">Аналітичний звіт сформовано: {created_at}</p>
        <p>
            Звіт містить рейтинг комп’ютерних ігор за статистичними,
            поведінковими та соціальними показниками популярності.
        </p>

        <table>
            <thead>
                <tr>
                    <th>#</th>
                    <th>Гра</th>
                    <th>Жанр</th>
                    <th>Платформа</th>
                    <th>DAU</th>
                    <th>MAU</th>
                    <th>CCU</th>
                    <th>Watch-time</th>
                    <th>Mentions</th>
                    <th>Sentiment</th>
                    <th>Search trend</th>
                </tr>
            </thead>
            <tbody>
                {table_rows}
            </tbody>
        </table>
    </body>
    </html>
    """

    file_path.write_text(html, encoding="utf-8")
    return file_path


if __name__ == "__main__":
    exporter = ReportExporter()
    csv_path = exporter.export_csv()
    html_path = exporter.export_html()

    print(f"CSV-звіт створено: {csv_path}")
    print(f"HTML-звіт створено: {html_path}")
```

ДОДАТОК Г

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ ЗДОБУВАЧА

Я, Нестеров Денис Юрійович, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему «Інтелектуальна система аналізу та оцінки відеоігор з рейтингами користувачів» є результатом моєї особистої праці.
2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):
   - [ ] інструменти генеративного ШІ не використовувалися.
   - [V] інструменти ШІ (вказати назву: **ChatGPT**, Gemini, Claude, DeepL, _____інші (вказати назву)) були використані для:
     - [ ] перекладу іншомовних джерел;
     - [V] стилістичного редагування та перевірки граматики;
     - [ ] допомоги у написанні/відлагодженні програмного коду;
     - [ ] інше: _пошук літератури, допомога з ідеями.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«___» _____ 2026 р. _____ **Денис НЕСТЕРОВ**