DOI: https://doi.org/10.25140/2411-5363-2026-1(43)-114-127
УДК 004.75

Андрій Іванович Роговенко [1], Костянтин Михайлович Клочко [2], Артем Юрійович Милиця[3]

[1]кандидат технічних наук, доцент кафедри інформаційних та комп'ютерних систем
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
E-mail: arogovenko@stu.cn.ua. **ORCID:** https://orcid.org/0000-0003-4594-5692
ResearcherID: G-3926-2014. **Scopus Author ID:** 57484900000.

[2]студент магістратури, кафедра інформаційних та комп'ютерних систем
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
E-mail: kostya_klochko@ukr.net. **ORCID:** https://orcid.org/0009-0003-8555-8723

[3]старший викладач кафедри інформаційних та комп'ютерних систем
Національний університет «Чернігівська політехніка» (Чернігів, Україна)
E-mail: artemmylytsia@stu.cn.ua. **ORCID:** https://orcid.org/0009-0003-5026-0601

АРХІТЕКТУРА ДЕЦЕНТРАЛІЗОВАНОЇ СИСТЕМИ КАТАЛОГІЗАЦІЇ КНИЖКОВИХ МЕТАДАНИХ ІЗ JSON API ТА ФЕДЕРАТИВНОЮ СИНХРОНІЗАЦІЄЮ

У статті запропоновано архітектуру децентралізованої системи каталогізації книжкових метаданих, орієнтовану на керування метаданими та інтеграцію з іншими інформаційними сервісами через JSON API. Проаналізовано підходи до федерації метаданих, інтероперабельності та соціальної децентралізації формування метаданих у контексті каталогізаційних систем. Розроблену систему реалізовано у формі вебзастосунку з рольовою моделлю доступу (гість, користувач, модератор, адміністратор), що підтримує перегляд і редагування даних про книги, авторів і видавців, а також надає документований JSON API зі специфікацією OpenAPI (Swagger UI).

Архітектурною основою системи є модульний моноліт із клієнт-серверною взаємодією, локальним сховищем метаданих та підсистемою фонової синхронізації з іншими вузлами федерації. Децентралізацію забезпечено через механізм локального кешування даних, отриманих із серверів федерації, та планову/ручну синхронізацію, що дозволяє зменшити кількість віддалених запитів, підвищити доступність даних при тимчасовій недоступності частини вузлів і забезпечити стабільний час відгуку пошукових запитів. Для збереження узгодженості даних застосовано стратегію «джерело істини», глобально унікальні ідентифікатори UUID та множинні бібліографічні ідентифікатори (ISBN-10, ISBN-13, ASIN) для зіставлення записів.

Запропонований підхід поєднує автономність вузлів федерації, практичні вимоги до продуктивності пошуку та простоту інтеграції із зовнішніми системами, що робить його придатним для використання в навчальних, дослідницьких і спільнотних каталогізаційних сервісах. Робота має архітектурно-проєктний характер і формує основу для подальшого кількісного оцінювання продуктивності та відмовостійкості системи у федеративному середовищі.

Ключові слова: *децентралізація; каталогізація; метадані книг; федерація серверів; вебзастосунок; JSON API; синхронізація; Elixir; Ash; Phoenix; PostgreSQL.*

Рис.: 4. Табл.: 1. Бібл.: 16.

Актуальність теми дослідження. Централізовані системи каталогізації книжкових метаданих мають низку практичних обмежень, пов'язаних зі швидкістю оновлення, актуальністю та доступністю даних. Оскільки модерація та наповнення таких каталогів часто зосереджені в межах обмеженого кола учасників, у базах даних накопичуються прогалини, а внесення нових записів і виправлення помилок потребують значного часу. Це знижує оперативність оновлення метаданих і ускладнює їх повторне використання в зовнішніх інформаційних системах.

Децентралізований підхід, заснований на федерації серверів, дає змогу розподілити функції створення, підтримки та поширення метаданих між незалежними вузлами. Такий підхід потенційно підвищує відмовостійкість, зменшує залежність від одного центрального сервера та створює передумови для інтеграції з іншими сервісами через стандартизовані програмні інтерфейси. У контексті книжкової каталогізації особливо важливим є поєднання децентралізації з практичними вимогами до продуктивного пошуку, узгодженості даних і керованості доступу до редагування метаданих.

Постановка проблеми. Проблема полягає у розробленні інформаційної системи каталогізації книг, яка одночасно:

– забезпечує кероване внесення, редагування та валідацію метаданих;

– підтримує інтеграцію з іншими інформаційними системами через програмний інтерфейс;

– мінімізує залежність від єдиного центрального вузла;

– забезпечує продуктивний пошук і доступність даних у разі тимчасової недоступності частини вузлів федерації.

Ключовою технічною вимогою є реалізація механізму синхронізації між серверами федерації, який дає змогу локально кешувати метадані з інших вузлів, виконувати їх зіставлення та використовувати локальне сховище як основу для пошукових сценаріїв.

Вектори атак, унікальні для децентралізованих систем:

– Ін'єкція шкідливих метаданих: зловмисний вузол федерації може надсилати навмисно спотворені або фальсифіковані бібліографічні дані, які потрапляють до локального кешу інших вузлів під час синхронізації.

– Підміна вузла федерації: зловмисник може зареєструвати підроблений сервер федерації та поширювати спам-записи або шкідливий контент через стандартний JSON API.

– UUID-колізії через атаку на генератор: хоча UUID v4 статистично унікальні, компрометований вузол теоретично може генерувати конфліктуючі ідентифікатори для порушення цілісності даних інших учасників.

Аналіз останніх досліджень і публікацій. Проблема управління метаданими у розподілених та федеративних системах розглядається в науковій літературі через призму інтероперабельності, стандартизації та соціальної взаємодії.

Аспект 1: Соціальна децентралізація створення метаданих (Фолксономія vs Таксономія) Важливим напрямком досліджень є аналіз того, як користувачі децентралізовано збагачують метадані книг. У роботі Dani Tellvik (2023) [15] досліджується феномен «фолксономії» (folksonomy) у книговидавництві. Авторка протиставляє її жорстким централізованим таксономіям, таким як коди BISAC (Book Industry Standards and Communication), які контролюються галузевими комітетами. [13, 15] Дослідження показує, що читачі через платформи на кшталт Goodreads створюють «живий цифровий архів» свого читацького досвіду, який є гнучкішим та швидшим у реагуванні на зміни в мові та культурі, ніж централізовані системи. [15] Tellvik зазначає, що фолксономія — це «система класифікації знизу-вгору», яка виникає із соціального тегування, де метадані генеруються не експертами, а споживачами контенту. [15]

Аспект 2: Федерація колекцій та проблеми інтероперабельності Питання об'єднання (федерації) метаданих з різних джерел ґрунтовно розглянуто у роботах Palmer et al. (2007) [13] та Palmer & Knutson (2004) [16],. Дослідники вказують на те, що навіть при використанні таких стандартів, як OAI-PMH та Dublin Core, створення єдиного простору доступу до розподілених колекцій стикається з проблемою різнорідних «культур опису» [13, 16],. Різні інституції та спільноти використовують метадані по-різному, що ускладнює їх автоматичну агрегацію без втрати контексту. Зокрема, Palmer et al. підкреслюють, що агрегація часто призводить до втрати «контекстуальної маси» (contextual mass), необхідної для повноцінного пошуку та розуміння об'єктів. [13]

Аспект 3: Технічні виклики інтеграції різнорідних даних Ordonez et al. (2007) [14] розглядають технічні аспекти управління метаданими у федеративних базах даних. Вони підкреслюють складність інтеграції структурованих даних (бази даних) та напівструктурованих даних (документи, файли), які часто знаходяться поза межами основного сховища. [14] Їхній підхід пропонує використання реляційної бази даних як центрального репозиторію для відстеження зв'язків між об'єктами у розподіленому середовищі, що дозволяє вирішувати проблеми неповноти та неузгодженості даних за допомогою SQL-запитів [14; 15].

Виділення недосліджених частин загальної проблеми. Незважаючи на значний обсяг досліджень у сфері цифрових бібліотек та соціального тегування, низка аспектів залишається недостатньо вивченою в контексті сучасних децентралізованих протоколів (таких як ActivityPub):

1. Архітектурна децентралізація збереження контексту. Більшість досліджень, зокрема Palmer et al., розглядають федерацію як створення централізованого реєстру або репозиторію, який збирає (harvests) дані з розподілених джерел [13, 16]. Недослідженим залишається питання ефективного збереження метаданих у повністю децентралізованих мережах, де відсутній єдиний вузол агрегації, але необхідно зберігати зв'язність даних.

2. Гібридизація фолксономій та стандартів у федеративному середовищі. Tellvik пропонує ідею «коллабулярію» (collabulary) — гібриду таксономії та фолксономії, де експерти співпрацюють зі споживачами [15]. Однак механізми автоматичної синхронізації таких гібридних метаданих між різними вузлами децентралізованої мережі (instance-to-instance) практично не висвітлені в технічній літературі.

3. Вирішення конфліктів даних без центрального арбітра. Ordonez et al.[14] пропонують централізований репозиторій для вирішення проблем неузгодженості даних. [16] Проте у випадку децентралізованих систем (як BookWyrm), де кожен вузол може мати власну версію опису книги, відсутні загальноприйняті алгоритми консенсусу для об'єднання суперечливих метаданих без втрати унікального локального контексту.

Серед поширених систем, які працюють із книжковими даними, можна виділити Goodreads, BookWyrm та Open Library. Goodreads орієнтована на соціальні сценарії та використовує централізовану модель, тоді як BookWyrm підтримує децентралізацію на основі ActivityPub, а Open Library надає каталогізаційний сервіс з API. Для визначення цільових характеристик розробленої системи виконано порівняння за критеріями централізованості, наявності API, типу інтерфейсу, можливостей пошуку та відкритості коду (табл. 1).

Таблиця 1 – Порівняння систем каталогізації книг

Критерій	Goodreads	BookWyrm	Open Library	Розроблена система
Централізованість	+	-	+	-
Децентралізованість	-	+	-	+
API для інтеграції	-	+	+	+
Тип інтерфейсу	-	ActivityPub JSON	JSON/RDF	JSON
Пошук	+	+	+	+
Пошук через API	-	-	+	+
Ліцензія/відкритість коду	пропрієтарна	ACSL	AGPLv3	AGPLv3
Додавання даних через запит	+	+	+	+

Мета дослідження. Розроблення інформаційної системи у формі вебзастосунку для децентралізованої каталогізації книжкових метаданих, яка забезпечує керування метаданими, інтеграцію з іншими сервісами через JSON API та синхронізацію між серверами федерації.

Для досягнення поставленої мети необхідно розв’язати такі завдання: обґрунтувати вибір технологічного стека з урахуванням функціональних вимог до вебінтерфейсу, API, ролей доступу та фонової обробки задач; спроєктувати модульну архітектуру системи та модель даних з підтримкою рольового доступу, федеративної взаємодії та збереження походження записів; реалізувати реактивний адаптивний вебінтерфейс для роботи з каталогами книг, авторів і видавців; розробити документований JSON API для інтеграції з зовнішніми сервісами та вузлами федерації; реалізувати механізм децентралізованої синхронізації з локальним кешуванням даних, фоновим виконанням задач

і засобами адміністрування процесу синхронізації; обґрунтувати вибір стратегії локального кешування як базового підходу до федеративного пошуку порівняно зі сценарієм прямого проксіювання запитів.

Виклад основного матеріалу. Для вирішення поставлених завдань проєктування системи виконано аналіз технологічних рішень, обґрунтовано вибір архітектурного підходу та визначено компонентну структуру системи. Основний акцент зроблено на забезпеченні балансу між функціональними можливостями, простотою підтримки та операційною ефективністю.

Архітектура системи та технологічний стек. Система спроєктована за клієнт-серверною архітектурою та реалізована як модульний моноліт. Такий підхід зберігає переваги модульності (чітке розділення доменних контекстів, інтеграцій, доступу до даних, UI) і водночас зменшує операційну складність порівняно з мікросервісами, що є критичним для невеликих команд розробки та навчальних/дослідницьких інфраструктур. Узагальнену взаємодію компонентів системи наведено на рисунку 1.

На серверній стороні використано мову Elixir, яка працює на віртуальній машині BEAM, і забезпечує ефективну конкурентність, ізоляцію процесів і стійкість до збоїв, а також добре підходить для вебсценаріїв з великою кількістю одночасних підключень [4]. Вебрівень реалізовано з допомогою Phoenix Framework, який надає механізми маршрутизації, контролерів, шаблонізації та роботу з WebSocket, а також типові практики побудови вебзастосунків [2].

Для побудови інтерфейсу обрано архітектуру з мінімалістичним реактивним дизайном та адаптивною верстkою. Phoenix LiveView забезпечує інтерактивність в реальному часі без необхідності писати складний клієнтський JavaScript, а також дозволяє централізовано контролювати видимість даних відповідно до ролей користувачів безпосередньо на сервері [3]. Для оформлення інтерфейсу застосовано TailwindCSS і Flowbite, що дає змогу швидко створювати адаптивні сторінки з уніфікованим мінімалістичним дизайном [9; 10].

Сховище даних реалізовано на PostgreSQL, що забезпечує транзакційність, цілісність і розвинені можливості індексації для пошукових сценаріїв за назвою та ідентифікаторами книг [5].

Для розгортання використано контейнеризацію Docker, що стандартизує середовище виконання і спрощує інсталяцію/оновлення [6]. Для маршрутизації трафіку та термінації TLS застосовано reverse proxy Nginx, який також дозволяє централізовано налаштувати HTTPS та оптимізувати доставку статичних ресурсів [7].

Децентралізація реалізована через JSON API та механізми синхронізації: дані, отримані з інших вузлів федерації, кешуються локально, що зменшує кількість віддалених запитів під час пошуку та підвищує доступність при тимчасовій недоступності частини вузлів. Планові й ручні задачі синхронізації виконуються у фоні через Oban, що забезпечує черги задач, повторні спроби та спостережуваність виконання [8].

Узагальнена схема взаємодії та архітектура наведена на рис. 1-2.

Користувачі взаємодіють з системою з допомогою вебінтерфейсу, а системи взаємодіють між собою по JSON API для синхронізації даних. Система може складатися з сервера каталогізації (DBI), базиданих (Postgres), які працюють як докер контейнери на віртуальній машині з Debian. Для взаємодії по https треба реверс проксі, на приклад, Nginx з налаштованими SSL сертифікатами.

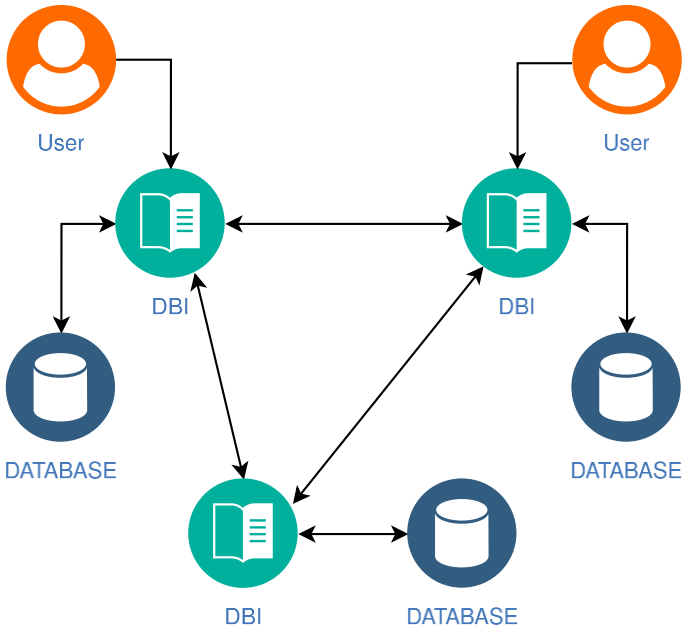

Рис. 1. Узагальнена схема взаємодії компонентів системи каталогізації книжкових метаданих

Джерело: розроблено авторами.

Примітка: клієнтський браузер взаємодіє з вебзастосунком через вебінтерфейс (LiveView) чи JSON API, який працює з PostgreSQL та виконує синхронізацію через JSON API з іншими серверами федерації.

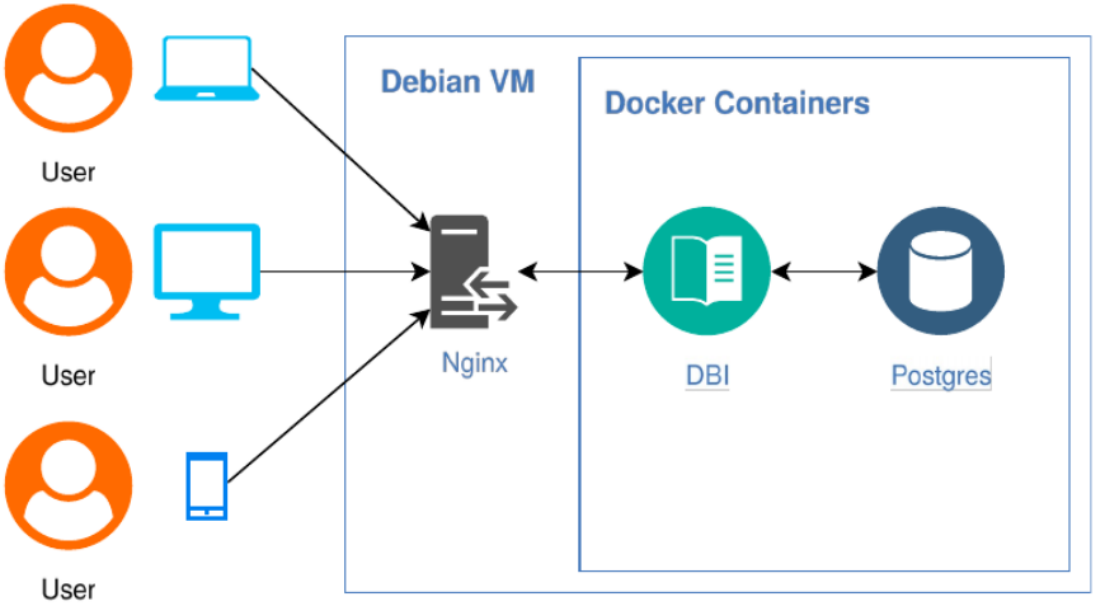

Рис. 2. Узагальнена архітектура системи каталогізації книжкових метаданих

Джерело: розроблено авторами.

Модель даних. Сховище даних реалізовано на PostgreSQL [5] та спроєктовано з урахуванням потреб пошуку, федерації і контролю походження записів. Базовою сутністю є ресурс «книга», для якого зберігаються назва, опис, посилання на обкладинку, дата публікації, мова, формат, кількість сторінок та зв’язки з видавцем і реєстром видань. Для забезпечення цілісності даних передбачено нормалізовані зв’язки та обмеження, а також

індексацію полів, що використовуються у фільтрації та пошуку. Крім того, всі записи використовують UUID як первинні ключі замість послідовних числових ідентифікаторів. Це критично важливо для федерації: UUID генеруються незалежно на кожному вузлі без координації, що усуває конфлікти ідентифікаторів при синхронізації записів з різних серверів. Комбінація UUID для внутрішньої ідентифікації та стандартних бібліографічних ідентифікаторів (ISBN) для зіставлення створює надійну основу для роботи федерації без центрального координатора.

Окремою сутністю виділено ідентифікатори книг (ISBN-10/ISBN-13/ASIN тощо) з атрибутами «тип», «значення» і «порядок сортування». Таке представлення дозволяє виконувати пошук за різними ключами (наприклад, за ISBN-13 та ASIN) без дублювання атрибутів у таблиці книг і без жорсткої прив'язки до конкретного стандарту ідентифікації. При синхронізації ідентифікатори використовуються як додаткові ключі зіставлення та для підвищення точності пошукових сценаріїв у федерації.

Для зв'язку «книга–автор» використано окрему сутність із ролями участі (автор, редактор, перекладач тощо) та порядком відображення. Це дозволяє зберігати не лише факт авторства, а й семантику внеску, що важливо для коректного відображення бібліографічного опису та для інтеграції з зовнішніми інструментами. Структура доменних ресурсів, зв'язків і політик доступу описується декларативно засобами Ash, що забезпечує узгодженість між вебінтерфейсом і API [1].

Запропоновану архітектуру розглянуто як базову модель для подальшої експериментальної валідації в умовах масштабування обсягу метаданих та кількості вузлів федерації.

Реалізація JSON API та інтеграції. JSON API реалізовано як набір endpoint-ів для читання каталогів і окремих елементів (книги, автори, видавці, сервери федерації), а також для отримання ідентифікаторів книги й ролей авторів. API підтримує посторінкову навігацію та фільтрацію (зокрема за назвою), що дозволяє ефективно будувати інтеграції зі сторонніми сервісами та клієнтами. Реалізація базується на Phoenix і Ash, де Ash визначає ресурси/дії/доступ, а Phoenix надає HTTP-шар і маршрутизацію [1; 2].

Публічна частина JSON API реалізує read-only endpoint-и (GET) без автентифікації, що спрощує інтеграцію. Операції модифікації метаданих виконуються через вебінтерфейс із рольовим контролем доступу.

Для пошуку книг за ідентифікатором (ISBN-13/ISBN-10/ASIN) передбачено окремий endpoint, що спрощує інтеграційні сценарії: зовнішня система може передати відомий ідентифікатор і отримати нормалізований запис книги з пов'язаними авторами, видавцем та набором ідентифікаторів. Це підвищує сумісність із бібліографічними інструментами, які оперують різними стандартами ідентифікації.

Специфікація API публікується через Swagger UI (OpenAPI), що формалізує контракт, зменшує ризик розходження реалізації та документації, а також дозволяє тестувати інтеграції на рівні схем/типів і прикладів запитів-відповідей. Документування є важливим також для федерації, оскільки сервери повинні взаємодіяти за єдиним узгодженим протоколом.

Механізм децентралізації та синхронізації. Децентралізація реалізована синхронізацією даних із серверів федерації, що дозволяє отримати дані з допомогою лише GET запитів без необхідності автентифікації. Для цього розроблено модулі клієнтів API (отримання JSON-даних з публічного API із підтримкою пагінації), підсистему перетворення даних у внутрішні структури, механізми зіставлення та злиття з локальною БД, а також контур керування задачами синхронізації. Локальне кешування метаданих дозволяє виконувати пошук переважно по локальній БД, зменшуючи кількість віддалених викликів і підвищуючи стійкість до мережевих збоїв.

Виконання планових задач організовано за допомогою Oban: синхронізація запускається щоденно опівночі або вручну адміністратором, а результати та статуси виконання доступні в адміністративній панелі. Обраний підхід забезпечує керованість фонових процесів (черги, повторні спроби, обмеження конкурентності) та прозорість виконання синхронізацій [8].

Редагування записів, отриманих з інших серверів, обмежено: зміна таких об’єктів через UI не допускається, що запобігає конфліктам і гарантує, що «джерело істини» залишається на вузлі, де запис створено. Натомість локальний вузол може створювати власні записи та публікувати їх через JSON API для інших учасників федерації. Така стратегія узгоджує відповідальність за дані та спрощує трасування походження записів.

Обґрунтування стратегії локального кешування. При проєктуванні механізму федеративного пошуку розглянуто альтернативний підхід — пряме проксіювання запитів до серверів федерації в реальному часі. Однак такий підхід має суттєві недоліки: час відгуку обмежується найповільнішим сервером у федерації, недоступність частини вузлів призводить до неповноти результатів пошуку, а складність агрегації та дедуплікації даних з різних джерел у реальному часі підвищує ймовірність помилок і ускладнює підтримку коду. Крім того, кожен пошуковий запит користувача генерує множинні HTTP-виклики до зовнішніх серверів, що створює надмірне навантаження на інфраструктуру федерації.

Обрана стратегія локального кешування усуває ці обмеження: пошук виконується виключно по локальній БД з використанням індексів, що забезпечує стабільний і швидкий час відгуку; система залишається працездатною при тимчасовій недоступності зовнішніх серверів; зіставлення та дедуплікація записів виконуються під час фонової синхронізації без впливу на користувацькі запити; навантаження на сервери федерації розподіляється у часі завдяки плановим синхронізаціям замість запитів у реальному часі. Компромісом є затримка актуалізації даних до наступної синхронізації, що є прийнятним для каталогізаційних систем, де метадані книг змінюються рідко і не потребують миттєвого оновлення.

Система має механізми захисту інформації та вузлів: білий список серверів для синхронізації, який налаштовується адміністратором, валідація вхідних даних та обмеження редагування даних з інших вузлів.

Хоча публічний API навмисно реалізований без автентифікації для читання, реєстрація серверів федерації в адміністративній панелі формує перший бар'єр: тільки явно зареєстровані вузли беруть участь у синхронізації.

Усі дані, отримані із зовнішніх вузлів, повинні проходити схемну валідацію перед записом у локальну БД. Синхронізація виконується в кілька етапів: отримання JSON запису, трансформування, перевірка наявності, запис даних за потреби до БД. Ash-ресурси з їх декларативними обмеженнями вже закладають основу для цього, однак варто явно зазначити, що трансформатор вхідних даних реалізують перевірки форматів даних (ISBN, UUID, тощо).

Стратегія «джерело істини» як механізм безпеки. Заборона редагування синхронізованих записів виконує подвійну функцію: крім збереження узгодженості даних, вона запобігає поширенню шкідливих змін зі скомпрометованого вузла через ланцюжок синхронізацій.

Запропоновані рішення виявлених проблем. Для вирішення описаних вище проблем у розробленій системі реалізовано комплекс архітектурних та технічних рішень:

Рішення проблеми 1 (децентралізоване збереження контексту): архітектура з локальним кешуванням метаданих на кожному вузлі федерації. Кожен сервер зберігає повну копію синхронізованих даних, що усуває залежність від центрального репозиторію та забезпечує працездатність системи при недоступності частини вузлів. Пошук виконується по локальній базі даних, що гарантує стабільний час відгуку незалежно від стану федерації.

Рішення проблеми 2 (інтеграція гібридних метаданих): відкритий JSON API без вимог автентифікації для читання, що дозволяє вузлам федерації обмінюватися даними через стандартизовані GET-запити. Структура метаданих включає не лише обов'язкові бібліографічні поля (назва, автори, видавець), а й розширювані атрибути (множинні ідентифікатори, ролі авторів, формати видань), що створює основу для інтеграції як експертних, так і користувацьких даних. Документування через OpenAPI забезпечує узгодженість протоколу між вузлами.

Рішення проблеми 3 (конфлікти даних та ідентифікація): стратегія «джерело істини» у поєднанні з глобально унікальними ідентифікаторами (UUID). Кожен запис зберігає інформацію про вузол-джерело та має UUID як первинний ключ, що гарантує унікальність ідентифікаторів навіть при одночасному створенні записів на різних серверах федерації. На відміну від послідовних числових ID, UUID генеруються незалежно на кожному вузлі без необхідності координації чи центрального лічильника, що усуває клас конфліктів ідентифікації при синхронізації.

Локальний вузол не дозволяє редагувати записи, синхронізовані з інших серверів, — зміни можливі лише на вузлі-джерелі, що усуває конфлікти версій. При зіставленні записів під час синхронізації використовуються множинні ідентифікатори (ISBN-13, ISBN-10, ASIN) як ключі, що підвищує точність зіставлення та зменшує ймовірність дублювання. Така модель зберігає автономію вузлів і водночас забезпечує трасованість походження даних.

Вирішення цих проблем формує основу для створення децентралізованої системи каталогізації, яка поєднує переваги федерації (розподіл навантаження, відмовостійкість, автономність вузлів) з практичними вимогами продуктивності, узгодженості даних та простоти інтеграції.

Реалізація системи та інтерфейси. Систему реалізовано згідно з описаною архітектурою (рис 2.). Головна сторінка з каталогом книг реалізована з мінімалістичним адаптивним інтерфейс з можливістю пошуку та фільтрації (рис 3.).

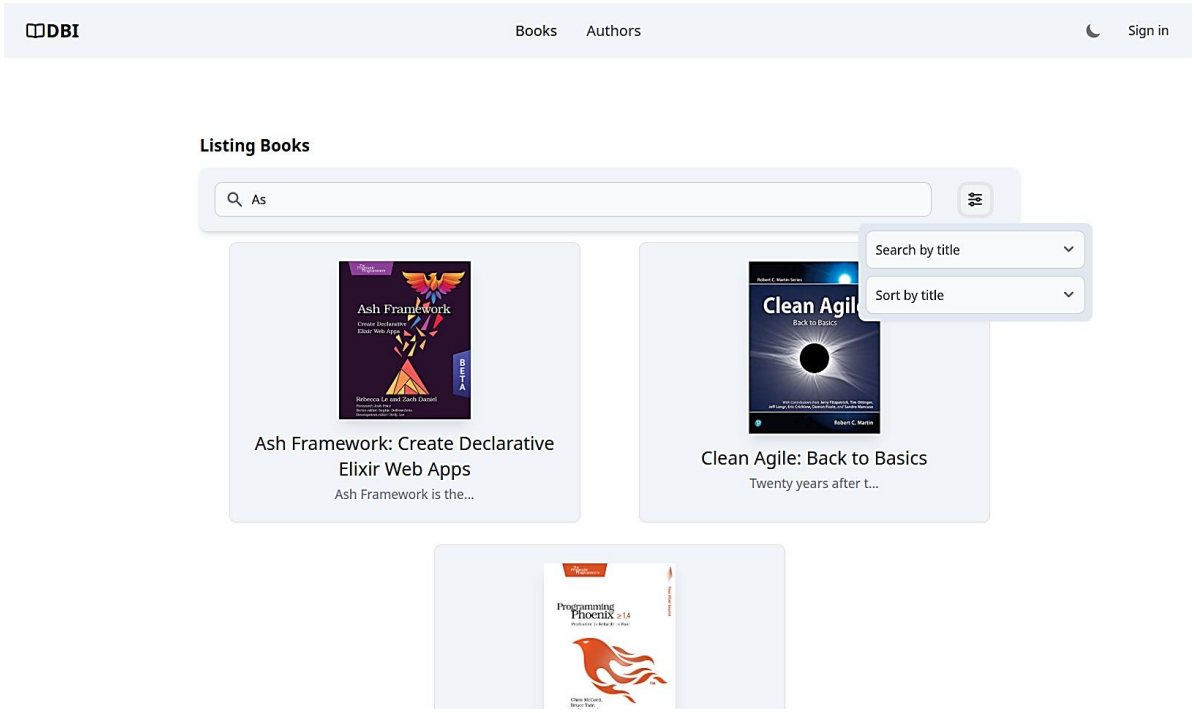

Рис. 3. Головна сторінка каталогу книг

Джерело: розроблено авторами.

Детально сторінка книги (рис. 4, 5) відображає повну структуру метаданих: назву, авторів з ролями участі, множинні ідентифікатори (ISBN-13, ISBN-10, ASIN), видавця та додаткові атрибути.

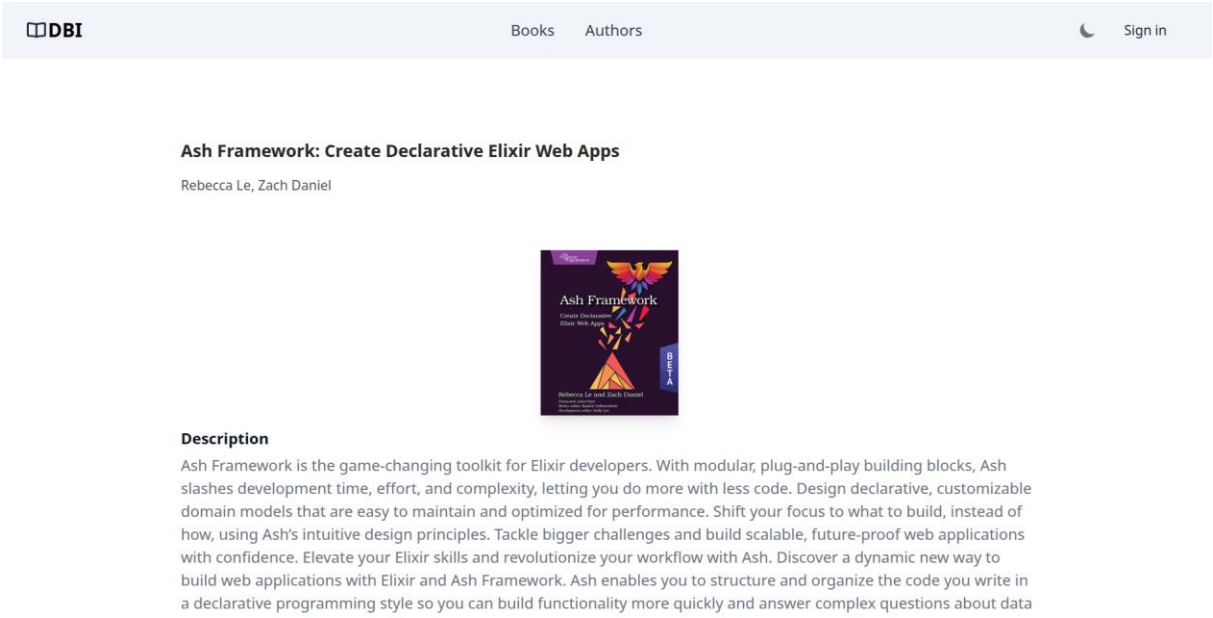

Рис. 4. Сторінка книги з метаданими та ідентифікаторами

Джерело: розроблено авторами.

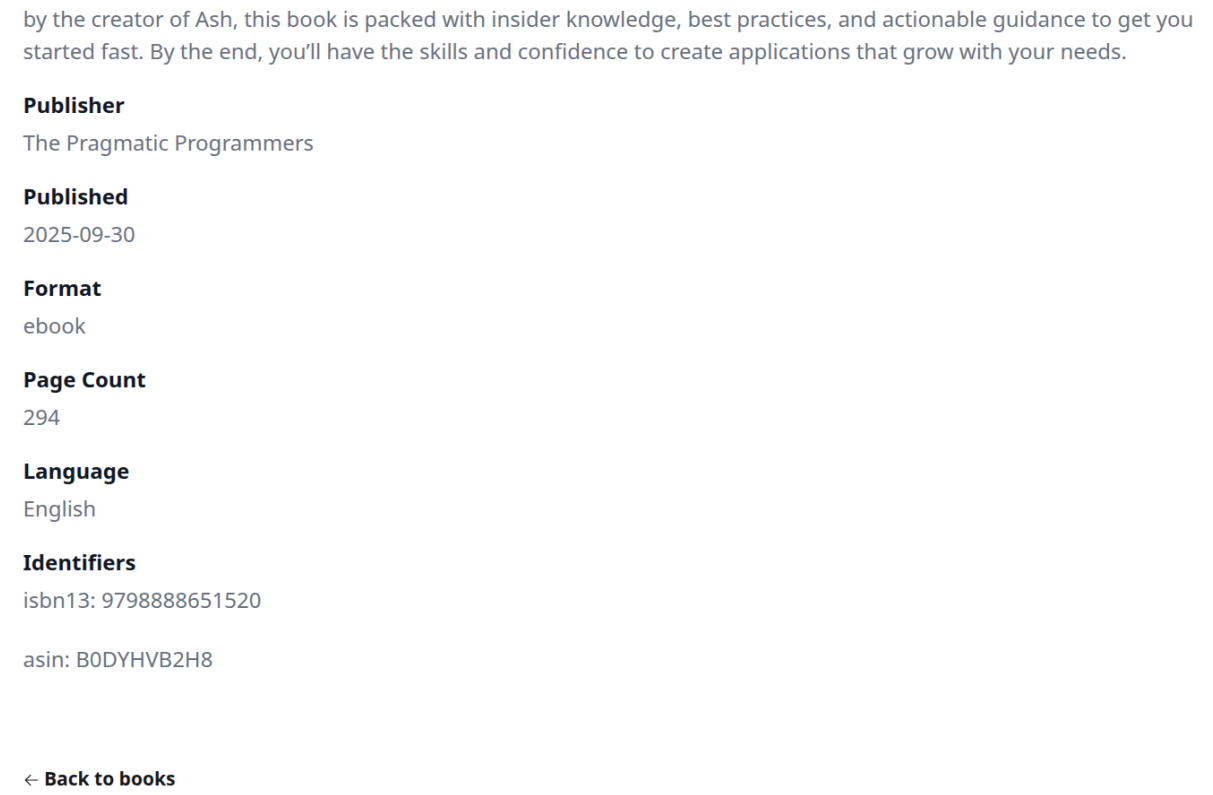

Рис. 5. Сторінка книги з метаданими та ідентифікаторами (продовження)

Джерело: розроблено авторами.

Для забезпечення прозорості інтеграцій API документовано через Swagger UI (рис. 6), що дозволяє розробникам сторонніх систем ознайомитися зі специфікацією endpoint-ів та тестувати запити безпосередньо у браузері.

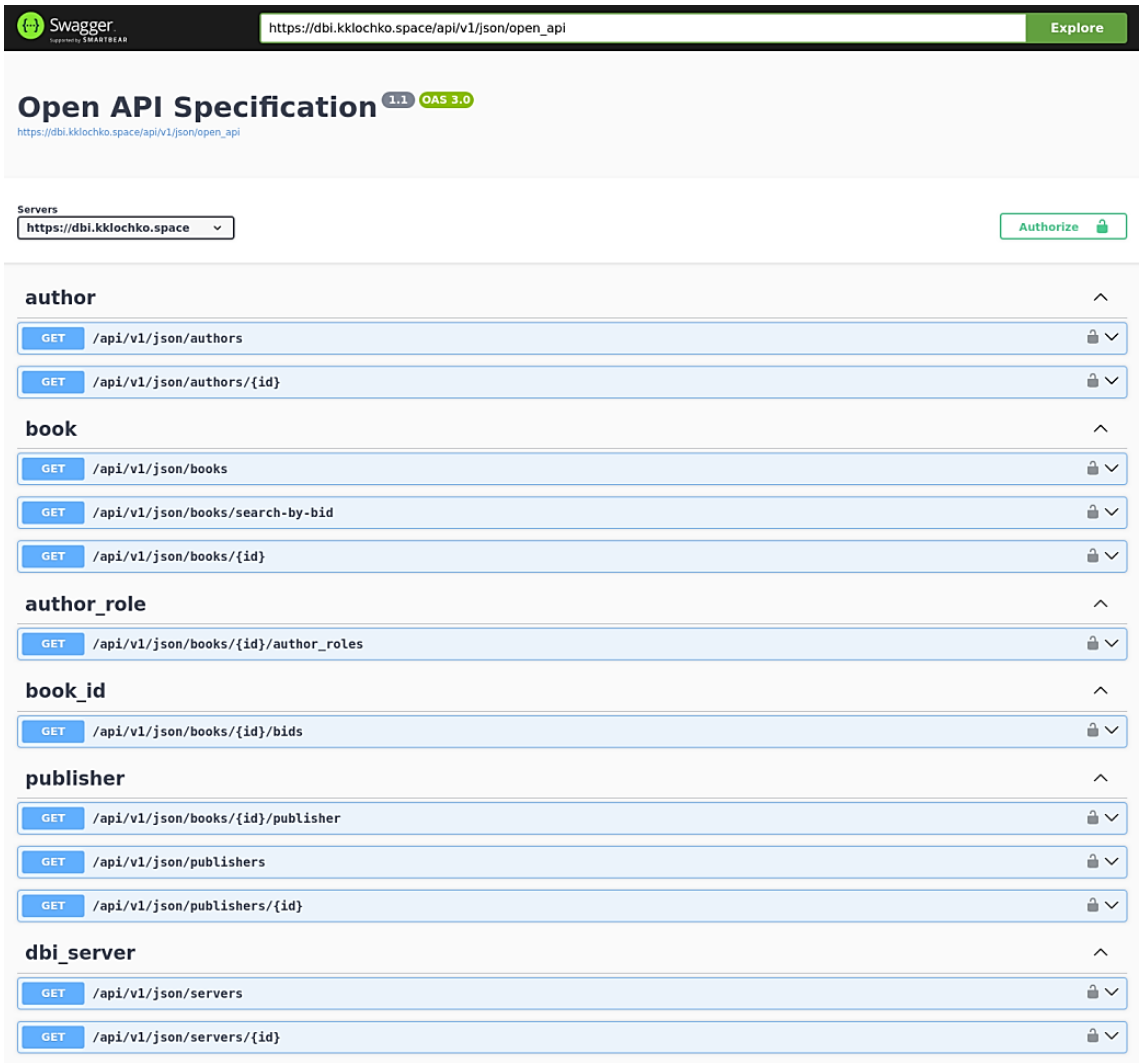

Рис. 6. Документація JSON API через Swagger UI

Джерело: розроблено авторами.

Обмеження дослідження та напрями подальшого розвитку

Поточне дослідження має архітектурно-проєктний характер і зосереджене на обґрунтуванні архітектури, реалізації прототипу системи та описі механізмів децентралізованої синхронізації й інтеграції через JSON API. На цьому етапі не проводилося повномасштабне експериментальне оцінювання продуктивності, відмовостійкості та якості зіставлення метаданих у федеративному середовищі, що є обмеженням роботи.

Разом із тим реалізований прототип і запропонована архітектура створюють основу для подальшого кількісного дослідження. Подальший розвиток роботи доцільно здійснювати за такими напрямами:

1. Експериментальне оцінювання продуктивності пошуку та синхронізації. Передбачається вимірювання часу відповіді пошукових запитів, затримки, тривалості циклу синхронізації, а також оцінка впливу кількості вузлів федерації та обсягу даних на продуктивність системи.

2. Дослідження відмовостійкості в умовах часткової недоступності вузлів федерації. Планується оцінити працездатність системи при недоступності частини серверів, частку успішних задач синхронізації, поведінку механізмів повторних спроб та вплив збоїв мережі на повноту локального кешу.

3. Розширення моделі даних і підтримки бібліографічних форматів. Перспективним є додавання механізмів імпорту/експорту у поширених бібліографічних форматах, розширення опису видань, а також підтримка додаткових типів ідентифікаторів для підвищення інтероперабельності.

4. Оцінювання практичної придатності системи в реальних сценаріях використання. Окремим напрямом є апробація системи в навчальних, дослідницьких або спільнотних середовищах з аналізом зручності використання, типових сценаріїв каталогізації та інтеграції з іншими сервісами.

5. Дослідження моделі довіри та безпеки федерації. Відкритим питанням залишається формалізація моделі довіри між вузлами: визначення критеріїв прийнятності вузла для участі у федерації, механізмів відкликання довіри та ізоляції скомпрометованого сервера без порушення роботи решти мережі. Перспективним є також дослідження застосування підписів метаданих (наприклад, на основі криптографічних ключів вузла) для верифікації автентичності та цілісності синхронізованих записів.

Отже, подальші дослідження мають бути спрямовані на кількісну валідацію запропонованих архітектурних рішень, поглиблення механізмів інтеграції та підвищення якості обробки метаданих у децентралізованому середовищі.

Наукова новизна одержаних результатів полягає в тому, що запропоновано архітектуру децентралізованої системи каталогізації книжкових метаданих у формі модульного моноліту, яка поєднує реактивний вебінтерфейс, документований JSON API та механізми федеративної синхронізації. А також удосконалено підхід до федеративної синхронізації метаданих за рахунок поєднання локального кешування, стратегії «джерело істини» та фонової планової/ручної синхронізації, що зменшує залежність від доступності зовнішніх вузлів під час пошуку;

Практичне значення одержаних результатів полягає у створенні працездатної інформаційної системи для децентралізованої каталогізації книжкових метаданих, реалізованої у формі вебзастосунку з рольовим доступом, публічним JSON API та підсистемою фонової синхронізації між серверами федерації. Запропоновані архітектурні рішення можуть бути використані у навчальних і університетських бібліотечних сервісах, у локальних книжкових спільнотах та тематичних каталогах, як основа для інтеграції з бібліографічними інструментами й сторонніми інформаційними системами, а також у навчальному процесі під час вивчення дисциплін з вебтехнологій, проєктування інформаційних систем та розподілених систем.

Висновки. У роботі запропоновано архітектуру децентралізованої системи каталогізації книжкових метаданих у формі вебзастосунку, що поєднує реактивний вебінтерфейс, документований JSON API та механізми федеративної синхронізації з локальним кешуванням даних. Запропонований підхід орієнтований на забезпечення балансу між автономністю вузлів федерації, продуктивністю пошуку та простотою інтеграції з іншими інформаційними системами.

Реалізацію системи виконано на основі технологічного стека Elixir/Phoenix/Ash/PostgreSQL із використанням Phoenix LiveView для реактивного інтерфейсу, PostgreSQL для транзакційного зберігання та індексованого пошуку, а також Oban для організації фонових задач синхронізації. Застосування модульного моноліту дозволило зберегти чітке розмежування доменних контекстів і підсистем інтеграції при нижчій операційній складності порівняно з мікросервісним підходом.

У межах запропонованої архітектури реалізовано рольову модель доступу (гість, користувач, модератор, адміністратор), яка забезпечує керованість змін метаданих та узгоджене застосування політик доступу у вебінтерфейсі й API. Для інтеграції з іншими сервісами реалізовано публічний read-only JSON API із документацією через OpenAPI (Swagger UI), що спрощує підключення зовнішніх клієнтів і вузлів федерації.

Ключовим архітектурним рішенням є використання локального кешування метаданих замість прямого проксіювання пошукових запитів до серверів федерації. Такий підхід зменшує залежність користувацьких сценаріїв від доступності зовнішніх вузлів, забезпечує стабільніший час відгуку пошуку та переносить витрати на інтеграцію і дедуплікацію у фоновий контур синхронізації. Для підвищення узгодженості даних у федеративному середовищі використано стратегію «джерело істини», глобально унікальні ідентифікатори UUID та множинні бібліографічні ідентифікатори для зіставлення записів.

На поточному етапі дослідження зосереджене на архітектурному обґрунтуванні та реалізації прототипу системи. Повномасштабне кількісне експериментальне оцінювання продуктивності, відмовостійкості та точності зіставлення метаданих визначено як наступний етап роботи. Така валідація дозволить формалізовано оцінити ефективність локального кешування у федеративному середовищі, порівняти запропонований підхід зі сценаріями прямого проксіювання та уточнити параметри подальшої оптимізації синхронізації.

Подальший розвиток системи доцільно спрямувати на експериментальну перевірку запропонованих рішень, розширення підтримки бібліографічних форматів, а також поглиблення інтеграцій із зовнішніми каталогами та сервісами.

Безпекова модель системи базується на явній реєстрації вузлів федерації, схемній валідації синхронізованих метаданих та обмеженні на редагування зовнішніх записів. Крім того, систему захисту можна покращити на базі моделі довіри між вузлами та реалізації механізмів захисту від шкідливих учасників федерації.

Заява про використання генеративного ШІ та технологій на основі ШІ в процесі написання тексту статті

Під час написання цього матеріалу автор(и) використовували Claude (Anthropic) для перевірки граматичної правильності українського та англійського текстів, а також для поліпшення структури окремих речень. Також було використано NotebookLM (Google) для аналізу джерел дослідження. Після використання цих інструментів автор(и) переглянув(ли) та відредагував(ли) зміст за потреби і взяв(ли) на себе повну відповідальність за зміст публікації.

Список використаних джерел

1. Ash Framework. (n.d.). Ash HQ. https://ash-hq.org/.
2. Phoenix Framework. (n.d.). Phoenix Framework. https://www.phoenixframework.org/.
3. Tate, B., & DeBenedetto, S. (2025). *Programming Phoenix LiveView: Interactive Elixir web programming without writing any JavaScript*. The Pragmatic Bookshelf.
4. The Elixir programming language. (n.d.). *The Elixir programming language*. https://elixir-lang.org/.
5. PostgreSQL. (n.d.). *PostgreSQL*. https://www.postgresql.org/.
6. Docker: Accelerated container application development. (n.d.). Docker. https://www.docker.com/.
7. Nginx. (n.d.). nginx. https://nginx.org/en/.
8. Robust job processing. (n.d.). Oban Pro. https://oban.pro/.
9. Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. (n.d.). Tailwind CSS. https://tailwindcss.com/.
10. Flowbite. (n.d.). Flowbite. https://flowbite.com/.

11.GitHub - bookwyrm-social/bookwyrm: Social reading and reviewing, decentralized with ActivityPub. (n.d.). GitHub. https://github.com/bookwyrm-social/bookwyrm.

12.GitHub - internetarchive/openlibrary: One webpage for every book ever published! (n.d.). GitHub. https://github.com/internetarchive/openlibrary.

13.Palmer, C. L., Zavalina, O. L., & Mustafoff, M. (2007). Trends in metadata practices: A longitudinal study of collection federation. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries* (pp. 386–395). https://doi.org/10.1145/1255175.1255251.

14.Ordonez, C., Chen, Z., & García-García, J. (2007). Metadata management for federated databases. In *Proceedings of the 2007 ACM Workshop on CyberInfrastructure: Information Management in eScience* (pp. 47–54). https://doi.org/10.1145/1317353.1317361.

15.Tellvik, D. (2023). *Metadata decentralized: An examination of folksonomy in book publishing* [Master's thesis, Portland State University]. PDXScholar. https://pdxscholar.library.pdx.edu/eng_bookpubpaper/77/.

16.Palmer, C. L., & Knutson, E. M. (2004). Metadata practices and implications for federated collections. In *Proceedings of the 67th ASIS&T Annual Meeting* (Vol. 41, pp. 456–462). https://doi.org/10.1002/meet.1450410153.

References

1. Ash Framework. (n.d.). Ash HQ. https://ash-hq.org/.

2. Phoenix Framework. (n.d.). Phoenix Framework. https://www.phoenixframework.org/.

3. Tate, B., & DeBenedetto, S. (2025). *Programming Phoenix LiveView: Interactive Elixir web programming without writing any JavaScript*. The Pragmatic Bookshelf.

4. The Elixir programming language. (n.d.). *The Elixir programming language*. https://elixir-lang.org/.

5. PostgreSQL. (n.d.). *PostgreSQL*. https://www.postgresql.org/.

6. Docker: Accelerated container application development. (n.d.). Docker. https://www.docker.com/.

7. Nginx. (n.d.). nginx. https://nginx.org/en/.

8. Robust job processing. (n.d.). Oban Pro. https://oban.pro/.

9. Tailwind CSS - Rapidly build modern websites without ever leaving your HTML. (n.d.). Tailwind CSS. https://tailwindcss.com/.

10.Flowbite. (n.d.). Flowbite. https://flowbite.com/.

11.GitHub - bookwyrm-social/bookwyrm: Social reading and reviewing, decentralized with ActivityPub. (n.d.). GitHub. https://github.com/bookwyrm-social/bookwyrm.

12.GitHub - internetarchive/openlibrary: One webpage for every book ever published! (n.d.). GitHub. https://github.com/internetarchive/openlibrary.

13.Palmer, C. L., Zavalina, O. L., & Mustafoff, M. (2007). Trends in metadata practices: A longitudinal study of collection federation. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries* (pp. 386–395). https://doi.org/10.1145/1255175.1255251.

14.Ordonez, C., Chen, Z., & García-García, J. (2007). Metadata management for federated databases. In *Proceedings of the 2007 ACM Workshop on CyberInfrastructure: Information Management in eScience* (pp. 47–54). https://doi.org/10.1145/1317353.1317361.

15.Tellvik, D. (2023). *Metadata decentralized: An examination of folksonomy in book publishing* [Master's thesis, Portland State University]. PDXScholar. https://pdxscholar.library.pdx.edu/eng_bookpubpaper/77/.

16.Palmer, C. L., & Knutson, E. M. (2004). Metadata practices and implications for federated collections. In *Proceedings of the 67th ASIS&T Annual Meeting* (Vol. 41, pp. 456–462). https://doi.org/10.1002/meet.1450410153.

Дата першого надходження статті до видання: 18.12.2025
Дата прийняття статті до друку після рецензування: 06.01.2026

TECHNICAL SCIENCES AND TECHNOLOGIES

UDC 004.75

Andrii Rohovenko [1], Kostiantyn Klochko [2], Artem Mylytsia[3]

[1] PhD in Technical Sciences, Associate Professor, Department of Information and Computer Systems
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
E-mail: arogovenko@stu.cn.ua. **ORCID:** https://orcid.org/0000-0003-4594-5692
ResearcherID: G-3926-2014. **Scopus Author ID:** 57484900000.

[2] Master's student, Department of Information and Computer Systems
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
E-mail: kostya_klochko@ukr.net. **ORCID:** https://orcid.org/0009-0003-8555-8723.

[3] Senior Lecturer, Department of Information and Computer Systems
Chernihiv Polytechnic National University (Chernihiv, Ukraine)
E-mail: artemmylytsia@stu.cn.ua. **ORCID:** https://orcid.org/0009-0003-5026-0601.

ARCHITECTURE OF A DECENTRALIZED BOOK METADATA CATALOGING SYSTEM WITH JSON API AND FEDERATED SYNCHRONIZATION

This paper proposes an architecture of a decentralized book metadata cataloging system designed for metadata management and integration with external information services through a JSON API. The relevance of the study is determined by the limitations of centralized cataloging platforms, where metadata moderation and updates are often concentrated within a small group of contributors, resulting in delays, incomplete records, and reduced interoperability with third-party systems.

The study analyzes approaches to metadata federation, interoperability, and the social decentralization of metadata creation in cataloging systems. Particular attention is paid to the challenges of combining heterogeneous metadata sources, preserving contextual information during federation, and supporting flexible metadata enrichment practices. Based on this analysis, the paper identifies practical requirements for a decentralized cataloging solution.

The developed system is implemented as a web application with role-based access control (guest, user, moderator, and administrator). It supports viewing and editing data about books, authors, and publishers, and provides a documented JSON API using an OpenAPI (Swagger UI) specification. The architectural foundation of the system is a modular monolith with client-server interaction, a local metadata store, and a background synchronization subsystem for communication with other federation nodes. This architectural choice preserves modular separation of concerns while reducing operational complexity compared to microservice-based deployments, which is especially important for small development teams and educational or research infrastructures.

Decentralization is achieved through local caching of data retrieved from federation servers and scheduled/manual synchronization. This strategy reduces the number of remote requests during user search operations, improves data availability when some federation nodes are temporarily unavailable, and provides more stable response times for search queries. To maintain data consistency in a federated environment, the system applies a "source of truth" strategy, globally unique UUID identifiers, and multiple bibliographic identifiers (ISBN-10, ISBN-13, ASIN) for record matching and deduplication. In addition, synchronized records obtained from external nodes are treated as non-editable on the local node, which reduces the risk of version conflicts and preserves source provenance.

The proposed approach combines federation node autonomy, practical search performance requirements, and ease of integration with external systems, making it suitable for educational, research, and community-based cataloging services. The study has an architectural and design-oriented character and provides a basis for future quantitative evaluation of system performance, synchronization efficiency, fault tolerance, and metadata matching accuracy in a federated environment.

Keywords: *decentralisation; book cataloging; metadata; server federation; web application; JSON API; synchronisation; Elixir; Ash; Phoenix; PostgreSQL.*

Fig.: 4. Table: 1. References: 16.

Роговенко А. І., Клочко К. М., Милиця А. Ю. Архітектура децентралізованої системи каталогізації книжкових метаданих із JSON API та федеративною синхронізацією. *Технічні науки та технології*, 2026. № 1(43). С. 114-127. DOI: https://doi.org/10.25140/2411-5363-2026-1(43)-114-127.