МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет “Львівська політехніка”

Інститут комп’ютерних наук та інформаційних технологій

Кафедра інформаційних систем та мереж



**Звіт**

до лабораторної роботи № 6

**«Побудова системи федералізації даних»**

з дисципліни

**«Технології інтеграції інформаційних ресурсів»**

Виконав: ст. гр. ІТІС-11

Стельмах В. Д.

Прийняв:

Щербак С.С.

Львів – 2025

**Лабораторна робота № 6**

**Тема:** Побудова системи федералізації даних

Мета роботи: Ознайомитися з основами системи федералізації даних із використанням технологій Global As View (GAV) та Local As View (LAV), продемонструвавши динамічне відображення даних з віддалених джерел.

**Порядок виконання роботи:**

1. Налаштувати середовище розробки Python та встановити необхідні бібліотеки (FastAPI, Uvicorn, Pydantic).
2. Розробити вебсервіс на Python з використанням FastAPI, який демонструватиме роботу системи федералізації даних для теми "Каталог товарів магазину електроніки".
3. Визначити два окремі джерела даних (файли JSON): products\_source.json (основна інформація про товари) та inventory\_source.json (дані про запаси та постачальників).
4. Визначити глобальну схему (Pydantic модель FederatedProduct) для консолідованого представлення товару.
5. Реалізувати логіку посередника (медіатора), який за запитом користувача до єдиного ендпоінту:

* Видобуває дані з products\_source.json.
* Видобуває дані з inventory\_source.json.
* Динамічно об'єднує (федерує) отримані дані відповідно до глобальної схеми.

1. Створити ендпоінт API (наприклад, /federated/products/{product\_id}), який повертає федеративне представлення товару.
2. Продемонструвати роботу вебсервісу, викликаючи ендпоінти та аналізуючи результати об'єднання даних з різних джерел "на льоту".
3. Задокументувати структуру вебсервісу, приклади його роботи (скріншоти), та надати посилання на репозиторій

**Короткі теоретичні відомості**

Федералізація даних – це підхід до інтеграції даних, що дозволяє уніфікований доступ до інформації, яка зберігається в різних, часто гетерогенних та географічно розподілених джерелах, без необхідності їх фізичного переміщення або копіювання в єдине централізоване сховище. Система федералізації даних діє як посередник (медіатор), який приймає запити користувача до віртуальної глобальної схеми, декомпозиціє їх на підзапити до відповідних локальних джерел, отримує результати та об'єднує їх "на льоту", повертаючи користувачеві консолідовану відповідь

Ключовими підходами до моделювання відображень між глобальною схемою та схемами джерел є Global As View (GAV) та Local As View (LAV).

* Global As View (GAV): Глобальна схема визначається як набір представлень (views) над даними локальних джерел. Тобто, кожен елемент глобальної схеми описується через запит до даних джерел. При обробці запиту користувача система розгортає його відповідно до цих визначень.
* Local As View (LAV): Навпаки, кожне локальне джерело описується як представлення над даними глобальної схеми. Переформулювання запиту в LAV є складнішим, оскільки системі потрібно знайти комбінацію локальних представлень, що відповідає глобальному запиту.

FastAPI – це сучасний вебфреймворк для створення API на Python, який добре підходить для реалізації посередника в системі федералізації завдяки своїй швидкості, підтримці асинхронності та автоматичній валідації даних за допомогою Pydantic.

**Результати виконання роботи**

**Демонстрація роботи системи федералізації даних**

Робота системи федералізації даних демонструється шляхом виклику ендпоінту API, який повертає об'єднані дані. Для тестування використовувалася автоматично генерована документація Swagger UI.

Загальний вигляд документації API (Swagger UI) та доступні ендпоінти:

Представлено головну сторінку документації API в Swagger UI, де видно основний ендпоінт /federated/products/{product\_id} для отримання федеративних даних, а також допоміжний ендпоінт (наприклад, /source/products) для перегляду даних з одного з джерел.

Зображення, що містить текст, знімок екрана, число, Шрифт

Вміст, створений ШІ, може бути неправильним.

Рис. 1. Загальний вигляд документації API системи федералізації даних в Swagger

Запит та успішна відповідь для товару, присутнього в обох джерелах:

Демонструється виклик ендпоінту /federated/products/{product\_id} для товару, інформація про який є в обох файлах-джерелах (наприклад, product\_id: 101). Скріншот показує JSON-відповідь, де об'єднано дані з products\_source.json (назва, категорія, ціна) та inventory\_source.json (кількість на складі, постачальник, місцезнаходження).

Зображення, що містить текст, знімок екрана, програмне забезпечення, Веб-сторінка

Вміст, створений ШІ, може бути неправильним.

Рис. 2. Федеративна відповідь для товару, дані про який є в обох джерелах (ID 101).

Запит та відповідь для товару, присутнього лише в одному джерелі (основному каталозі):

Показано виклик ендпоінту /federated/products/{product\_id} для товару, інформація про який є тільки в products\_source.json, але відсутня в inventory\_source.json (наприклад, product\_id: 104 з наданих прикладів даних). JSON-відповідь містить дані з основного каталогу, а поля, що мали б прийти з інвентарного джерела, мають значення null.

Зображення, що містить текст, знімок екрана, програмне забезпечення, Веб-сторінка

Вміст, створений ШІ, може бути неправильним.

Рис. 3. Федеративна відповідь для товару, дані про який відсутні в інвентарному джерелі (ID 104).

Обробка помилки (товар не знайдено в основному джерелі):

Демонстрація реакції системи на запит інформації про товар за ID, якого немає в основному файлі-джерелі products\_source.json. Сервер повертає відповідь зі статусом 404 Not Found.

Зображення, що містить текст, програмне забезпечення, Шрифт, число

Вміст, створений ШІ, може бути неправильним.

Рис. 4. Реакція системи на запит даних про неіснуючий товар.

**Аналіз роботи системи федералізації**

Розроблений вебсервіс успішно демонструє принципи федералізації даних на прикладі підходу, близького до Global As View (GAV). Запит до єдиного ендпоінту /federated/products/{product\_id} ініціює динамічне звернення до двох окремих файлів-джерел. Сервіс-посередник об'єднує отриману інформацію "на льоту" відповідно до визначеної глобальної схеми FederatedProduct та повертає користувачеві консолідований результат. Продемонстровано коректну обробку випадків, коли дані про товар присутні в обох джерелах, лише в одному, або взагалі відсутні в основному джерелі. Це показує гнучкість підходу федералізації, де дані залишаються у своїх оригінальних сховищах.

**Посилання на вихідний код програми та файли даних**

Повний вихідний код розробленого сервісу федералізації даних на Python з використанням FastAPI, а також приклади файлів-джерел (products\_source.json, inventory\_source.json) розміщено у репозиторії GitHub за наступним посиланням:

<https://github.com/VitaliiStelmakh/TIIR/blob/main/lab_6>

**Висновки**

Під час виконання лабораторної роботи я ознайомився з концепцією федералізації даних та реалізував простий вебсервіс-посередник на FastAPI, що демонструє цей підхід. Було створено систему, яка динамічно об'єднує дані з двох окремих джерел за запитом користувача, не вимагаючи попереднього копіювання або переміщення даних. Ця робота дозволила зрозуміти переваги федеративного підходу до інтеграції даних, зокрема можливість отримання єдиного представлення розподіленої інформації, та освоїти базові принципи реалізації таких систем.