ДОДАТОК А

Графічний матеріал кваліфікаційної роботи

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

ФАКУЛЬТЕТ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА УПРАВЛІННЯ
КАФЕДРА КІТС

ІНТЕЛЕКТУАЛЬНА СИСТЕМА НЕЧІТКОГО ПОШУКУ АДРЕС У БАЗІ ДАНИХ

Виконав
Постольний Д.О.

Керівник професор кафедри КІТС
Аксак Н.Г.

Харків, 2025

Мета роботи

Метою роботи є розроблення та експериментальна перевірка інтелектуальної системи нечіткого пошуку адрес, здатної у режимі реального часу:

- нормалізувати користувацьке введення;
- відсіювати нерелевантні кандидати за допомогою швидких лексичних метрик;
- проводити семантичний пошук у векторній БД Weaviate;
- повертати єдиний, найбільш ймовірний результат із точністю ≥ 95 % при латентності ≤ 150 мс.

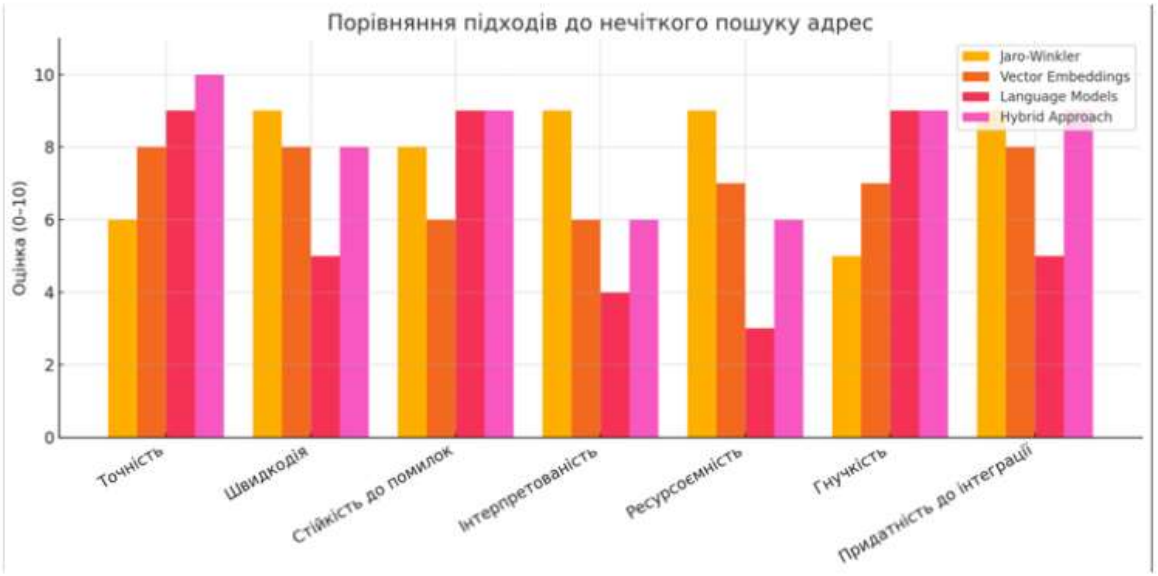
Для досягнення мети розв'язано такі основні завдання:

1) провести огляд існуючих методів адресного зіставлення та визначити їхні обмеження;
2) обґрунтувати вибір гібридного підходу та побудувати загальну архітектуру системи;
3) створити реляційну БД міст/вулиць і підготувати імпорт відкритих даних Укрпошти;
4) реалізувати функції Jaro-Winkler у SQL Server для попереднього фільтру;
5) розгорнути Weaviate у Docker-середовищі та налаштувати мовну модель для створення ембеддингів;
6) спроєктувати REST API, що інкапсулює повний конвеєр пошуку;
7) провести експериментальне оцінювання точності та часу відповіді в реальній вибірці помилкових запитів.

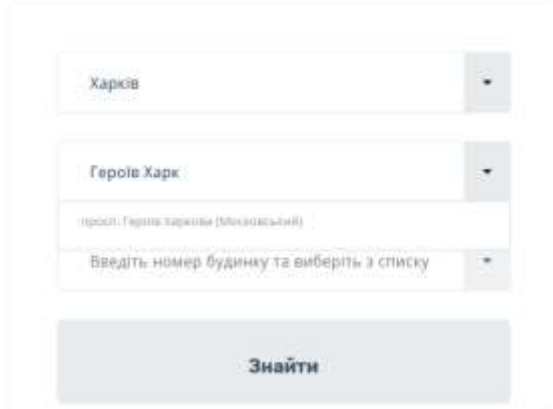
Порівняльний аналіз підходів до реалізації нечіткого пошуку адрес

Критерій / Підхід	Лексичні метрики (Jaro-Winkler)	Векторні представлення (Word2Vec, FastText)	Мовні моделі (BERT, GPT)	Гібридний підхід (JW + Weaviate)
Тип схожості	Посимвольна	Семантична (на рівні слів)	Семантична + контекстна	Комбінована (лексична + семантична)
Облік контексту	Ні	Частково (FastText враховує морфологію)	Так	Так
Точність на реальних запитах	Середня (70–80 %)	Вища (85–90 %)	Висока (90–95 %)	Дуже висока (95+ %)
Швидкодія / латентність	Висока (10–30 мс)	Висока (30–50 мс)	Середня / низька (100–300 мс)	Висока (50–150 мс при оптимізації)
Стійкість до помилок введення	Висока для коротких рядків	Середня	Висока	Висока
Інтерпретованість	Висока	Середня	Низька	Середня
Ресурсоємність	Низька	Середня	Висока	Середня
Гнучкість / масштабованість	Низька	Середня	Висока при кластерному виконанні	Висока (через Weaviate + REST API)
Потреба в навчанні	Відсутня	Необхідне попереднє навчання	Обов'язкове pretraining + fine-tuning	Часткове (embedding-модель + евристики)
Придатність до інтеграції	Висока (може бути вбудований у СУБД)	Висока (у вигляді API або локального модуля)	Обмежена без потужного сервера	Висока (Docker, FastAPI, Weaviate)

Порівняльна характеристика методів



ЗВИЧАЙНИЙ МЕТОДИ ПОШУКУ



ПРОБЛЕМА ЗВИЧАЙНОГО ПОШУКУ

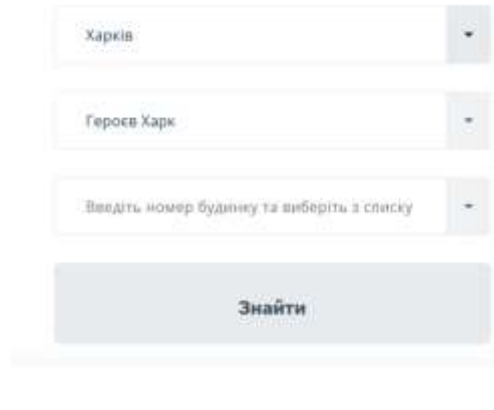


СХЕМА АРХІТЕКТУРИ СИСТЕМИ

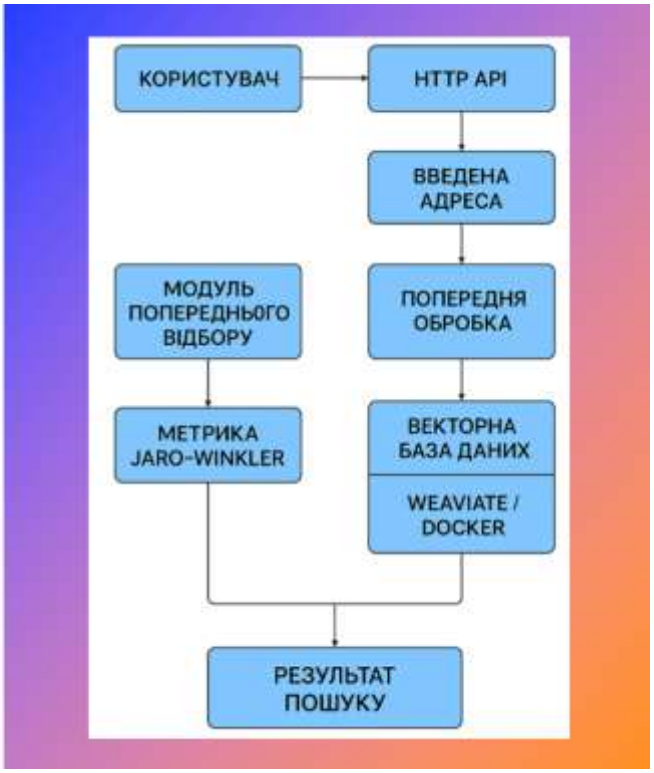
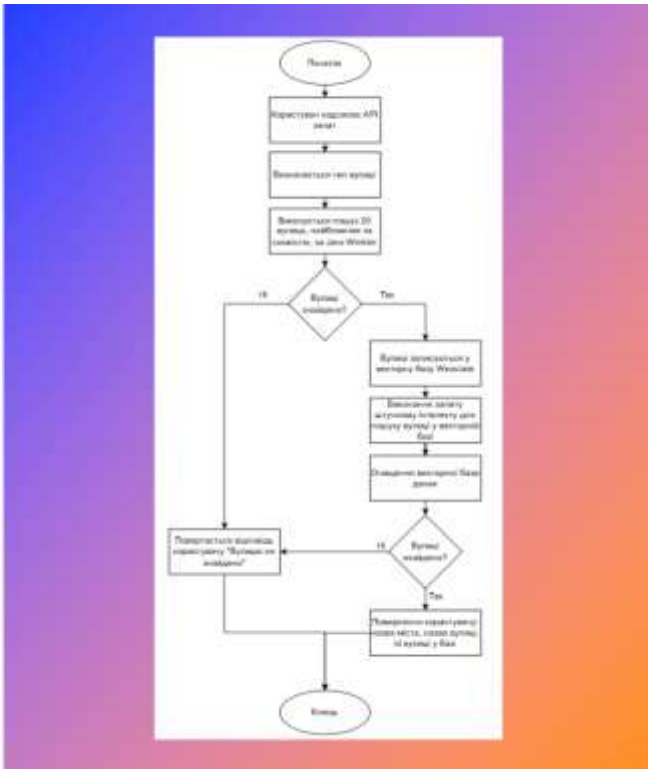


СХЕМА БАЗИ ДАНИХ

СХЕМА ВЕКТОРНОЇ БАЗИ ДАНИХ

```
Streets:
{
  "street": string,
  "street_id": int
}
```

АЛГОРИТМ ПОШУКУ ВУЛИЦІ



ВИКОНАННЯ ІНТЕЛЕКТУАЛЬНОГО ПОШУКУ

Request body

Edit Value | Schema

```
{
  "city": "м. Харків",
  "street": "Героїв Харк"
}
```

Request body

Edit Value | Schema

```
{
  "city": "м. Харків",
  "street": "просп. Академіка Свобод"
}
```

Code | Details

200

Response body

```
{
  "result": {
    "street_id": 17757,
    "street": "просп. Героїв Харкова",
    "city": "м. Харків"
  }
}
```

Code | Details

200

Response body

```
{
  "result": {
    "street_id": 11926,
    "street": "просп. Академіка Свободи",
    "city": "м. Харків"
  }
}
```

ВИСНОВКИ

В ході виконання цієї роботи було:

- Досліджено проблемну область.
- Проаналізовано існуючі підходи до вирішення задачі пошуку вулиць
- Сформовано архітектуру гібридного підходу на основі класичних алгоритмів та нейронних моделей.
- Розроблено серверну частину застосунку на мові Python для обробки запитів користувача та виконання пошуку.