МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА

ЖУРОВСЬКИЙ ЯРОСЛАВ ОЛЕГОВИЧ

Допускається до захисту:
В.о. завідувача кафедри
інформаційних технологій
канд. техн. наук, доцент
_____ Оксана ЗЕЛІНСЬКА
«_____» _____ 2025р.

РОЗРОБКА СЕРВЕРНОЇ ЧАСТИНИ САЙТУ ПРО ПОЛЕГЛИХ ГЕРОЇВ ВІННИЦЬКОЇ ОТГ

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (бакалаврська) робота

Керівник:
Павло РИМАР, старший викладач
кафедри інформаційних технологій
_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС /за національною шкалою)
Голова ЕК: _____

Вінниця – 2025

АНОТАЦІЯ

Журовський Я.О. Розробка серверної частини сайту про полеглих героїв Вінницької ОТГ. Спеціальність 122 «Комп'ютерні науки», освітня програма «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця 2025.

У кваліфікаційній (бакалаврській) роботі розроблено серверну частину вебсайту, що реалізує інформаційну систему для збереження, обробки, фільтрації та пошуку даних про загиблих воїнів Вінницької об'єднаної територіальної громади. У якості джерела даних використано відкриті набори з державного порталу opendata.gov.ua, доступ до яких реалізовано через CKAN API. Система підтримує пошук, фільтрацію, визначення героїв поточного дня, отримання координат поховань та ручне оновлення даних.

Ключові слова: серверна частина, загиблі воїни, Вінницька ОТГ, інформаційна система, відкриті дані, CKAN API, ASP.NET Core, JSON, API.

56 сторінок, 6 рис., 41 джерело.

ABSTRACT

Zhurovskyi Ya. Development of the server part of the website about the fallen heroes of Vinnytsia united territorial community. Specialty 122 «Computer Science», educational program «Computer Science». Vasyl Stus Donetsk National University, Vinnytsia 2025.

In the qualification (bachelor's) thesis, the server part of the website was developed, which implements an information system for storing, processing, filtering and searching for data on the fallen soldiers of the Vinnytsia United Territorial Community. As a data source, open sets from the state portal opendata.gov.ua were used, which are accessed via the CKAN API. The system supports search, filtering, identification of heroes of the current day, obtaining coordinates of burials and manual data update.

Keywords: server part, fallen soldiers, Vinnytsia United Territorial Community, information system, open data, CKAN API, ASP.NET Core, JSON, API.

56 pages, 6 figures, 41 sources.

ЗМІСТ

ВСТУП 5
РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД АНАЛОГІВ 13
1.1 Аналіз сучасного стану проблеми 13
1.2 Аналіз існуючих аналогів 14
1.3 Постановка задачі дослідження 16
Висновки до розділу 17
РОЗДІЛ 2. ОГЛЯД ТЕХНОЛОГІЙ ТА РІШЕНЬ 19
2.1. Вибір ASP.NET Core Web API як серверної платформи 19
2.2 Формалізація задачі та вимоги до системи 20
2.3 Моделювання предметної області 21
2.4 Побудова структури даних 22
2.5 Розробка алгоритмів 23
2.6 Модель архітектури системи 24
2.7 Робота з відкритими даними 25
2.8 Побудова моделі фільтрації та агрегації 26
2.9 Система координат та візуалізації 27
2.10 Інтеграція компонентів та загальна схема взаємодії 28
Висновки до розділу 29
РОЗДІЛ 3. РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ .. 31
3.1 Опис середовища розробки 31
3.2 Реалізація структури проєкту 33
3.3 Реалізація API-методів 34
3.4 Візуалізація координат і взаємодія з картою 36
3.5 Взаємодія з CKAN API та обробка JSON 38
3.6 Реалізація алгоритмів 39
3.7 Тестування системи 40
3.8 Аналіз результатів реалізації 46
Висновки до розділу 48

ВИСНОВКИ..... 50
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 54

ВСТУП

Актуальність теми дослідження. Сучасний світ характеризується розвитком інформаційних технологій, що покращує підходи до збереження, обробки та поширення інформації. Поширення цифрових технологій торкнулось усіх сфер людського життя, а саме: науки, освіти, культури, а також історичної пам’яті. Використання сучасних інформаційних систем значною мірою покращує можливості доступу до важливих даних, даючи змогу ефективно зберігати, аналізувати та поширювати інформацію про події, особистості та суспільно значущі явища.

На даний момент, під час важкого стану у нашій країні набуває великого значення можливість збереження пам’яті про історичні події та створення доступних цифрових засобів перегляду цієї інформації. Війна в Україні, що триває вже багато років, є одним із найбільшим викликом суспільства у всьому світі, вона принесла надзвичайно великі людські втрати та спонукала до необхідності створення нових, сучасних, засобів вшанування пам’яті загиблих. Отже, важливим завданням окрім документування подій, є структуризація та формування ефективних сервісів для збереження інформації про тих, хто віддав своє життя за свободу і незалежність держави.

Встановлення меморіалів, написання книг, створення архівних баз у друкованому форматі, несуть у собі великий сенс та матеріалізують пам’ять, але вони мають певні обмеження. Такі як обмеження у масштабності зберігання даних, проблеми у потребі легко оновлювати та редагувати дані, а також додавати їх. У свою чергу, цифрові технології надають можливість створювати інтерактивні інформаційні ресурси, які здатні об’єднати значні об’єми даних, забезпечити їхню актуальність та зручність у використані ресурсів для їх перегляду та обробки.

Особливої уваги заслуговує розробка спеціалізованих онлайн-платформ, які дозволяють користувачам оперативно отримувати потрібну інформацію, застосовувати різноманітні критерії пошуку та проводити аналітичну обробку

даних. Цифрові меморіальні системи дають можливість зберігати біографічні відомості про загиблих воїнів, інформацію про їх військові досягнення, нагороди, місця поховання, а також додаткову інформацію, яка може бути цінною як для дослідників, так і для звичайних громадян.

Окрім того, цифрові платформи відчиняють двері до нових горизонтів інтеграції історичних відомостей у соціальну та освітню сфери. Їх можна застосовувати для проведення дослідницьких проєктів, написання наукових праць, створення статистичних оглядів і навіть для глибшого розуміння суспільних явищ. Отже, інформаційні системи меморіального характеру набувають статусу вагомого чинника формування національної ідентичності, сприяють збереженню історичної пам’яті та виконують функцію своєрідного «мосту» між різними поколіннями.

Ще одним важливим аспектом є зручність та доступність таких ресурсів. Використання веб-технологій дозволяє створювати інтуїтивно зрозумілі інтерфейси, що не потребують спеціальних навичок для роботи з інформаційними базами. Це дає змогу як державним установам, так і громадськості активно користуватися подібними системами, перетворюючи їх на важливий інструмент для вшанування пам’яті, дослідження історії та формування суспільної свідомості.

Враховуючи актуальність проблеми, значний суспільний запит та можливості, які відкриває використання сучасних технологій у сфері історичної пам’яті, досліджувана тема є надзвичайно важливою. Вона охоплює не лише технічні аспекти розроблення інформаційних систем, а й соціокультурний, історичний та етичний виміри, що підкреслює її багатогранність і значущість у сучасному світі.

Метою даного дослідження є різносторонній аналіз сучасних підходів до створення інформаційних систем, націлених на збереження історичної пам’яті, та розробка ефективної концепції цифрового ресурсу, який забезпечить систематизацію, збереження, оновлення та зручне використання даних про загиблих воїнів.

Збереження пам’яті про історичні події та видатних осіб є невід’ємною частиною культурного розвитку суспільства. У сучасних умовах, коли цифрові технології відіграють ключову роль у зберіганні та передачі інформації, особливого значення набуває створення ефективних інформаційних систем, що дають змогу організовувати та підтримувати доступ до даних у зручному та структурованому вигляді. Важливим аспектом є також забезпечення можливості інтерактивної взаємодії з інформаційним ресурсом, що дозволяє не лише переглядати дані, а й здійснювати їх пошук, фільтрацію та аналітичну обробку.

Запропонована концепція цифрового ресурсу повинна враховувати ключові принципи організації даних, їх безпечного зберігання та зручного доступу до них. Основний акцент робиться на розробці такого рішення, яке відповідатиме вимогам надійності, масштабованості, доступності та простоти використання. У цьому контексті дослідження охоплює аналіз існуючих підходів до створення цифрових архівів та меморіальних інформаційних систем, виявлення їхніх переваг та недоліків, визначення найкращих практик у сфері управління даними та збереження історичної пам’яті, розроблення структурної концепції цифрового ресурсу, що забезпечить ефективне збереження та організацію інформації про загиблих воїнів, опрацювання підходів до створення зручного та інтуїтивно зрозумілого інтерфейсу для користувачів, що дозволить швидко знаходити необхідну інформацію та взаємодіяти з даними, врахування питань інформаційної безпеки та захисту персональних даних, які є критично важливими при роботі з подібними системами.

Досягнення поставленої мети дозволить створити ефективний механізм збереження історичної пам’яті, який буде доступним широкому загалу, сприятиме підвищенню рівня суспільної обізнаності та допоможе зберегти інформацію про героїв для майбутніх поколінь.

Таким чином, дослідження має на меті не лише технічне розв’язання проблеми організації даних, а й реалізацію важливої соціальної місії – сприяння формуванню колективної історичної пам’яті та вшануванню пам’яті загиблих воїнів у цифровому просторі.

Задачі дослідження. Для досягнення поставленої мети необхідно здійснити комплексне дослідження, яке охоплює кілька ключових напрямів. Першочерговим завданням є аналіз існуючих цифрових платформ, що використовуються для збереження історичної пам’яті. Це дозволить виявити найкращі практики, оцінити функціональні можливості подібних систем та зрозуміти, які підходи вже застосовуються в цій сфері.

Наступним важливим аспектом дослідження є визначення ключових вимог до інформаційних систем меморіального типу, включаючи принципи організації даних, структурування інформації, вимоги до безпеки та доступності. Зокрема, необхідно врахувати потреби кінцевих користувачів, забезпечивши простоту у взаємодії з системою, швидкий доступ до необхідних даних та можливість ефективного їх оновлення.

Особливу увагу слід приділити дослідженню методів організації та збереження інформації. Це включає аналіз різних моделей баз даних, механізмів індексації та ефективного пошуку інформації. Важливо зрозуміти, які алгоритми та технології забезпечують швидку і точну обробку запитів користувачів, а також які методи зберігання даних забезпечують їхню довготривалу актуальність.

Важливою частиною дослідження є розроблення концепції інформаційного ресурсу, призначеного для систематизації та збереження пам’яті про загиблих воїнів. Це передбачає створення логічної архітектури системи, визначення ключових функціональних можливостей, способів інтеграції з іншими сервісами та потенційного розширення функціоналу в майбутньому.

Окремим завданням є оцінка перспектив впровадження подібних рішень у суспільстві. Необхідно розглянути, як створення подібної інформаційної системи може сприяти збереженню національної пам’яті, популяризації історичних фактів і формуванню цифрового культурного надбання. Аналіз потреб суспільства, потенційних користувачів та організацій, що можуть бути зацікавлені в такому ресурсі, дозволить оцінити рівень його практичної значущості та можливі шляхи подальшого розвитку.

Виконання зазначених завдань дозволить сформувати цілісне бачення сучасних підходів до збереження історичної пам’яті в цифровому просторі, визначити ключові технологічні рішення та запропонувати ефективну концепцію реалізації інформаційного ресурсу, здатного відповідати актуальним викликам та суспільним потребам.

Об’єктом дослідження є цифрові інформаційні системи, які застосовуються для збереження історичних даних, упорядкування архівної інформації та увічнення пам’яті про визначних осіб, зокрема загиблих воїнів. Такі системи відіграють важливу роль у формуванні цифрового культурного простору та забезпеченні доступу до достовірної історичної інформації. З розвитком інформаційних технологій цифрові платформи все більше використовуються для архівування та систематизації даних, що сприяє збереженню національної пам’яті, популяризації історичних фактів та створенню інтерактивних ресурсів для широкого загалу.

Предметом дослідження є методи та технології, що використовуються для створення та функціонування інформаційних систем меморіального характеру. Зокрема, розглядаються способи структуризації, збереження, оновлення та пошуку даних у цифрових системах. Особливу увагу приділено питанням оптимізації процесів обробки інформації, забезпеченню швидкого доступу до неї та можливості інтеграції з іншими інформаційними ресурсами.

Дослідження охоплює сучасні підходи до проєктування цифрових архівів, методи побудови баз даних, а також ефективність різних моделей організації інформації. Аналізуються технічні рішення, що дозволяють забезпечити зручний пошук, фільтрацію та взаємодію користувачів із системою. Важливою складовою дослідження є питання безпеки та надійності збереження даних, оскільки інформаційні системи меморіального типу повинні гарантувати їхню цілісність та довгострокову доступність.

Таким чином, у межах цього дослідження розглядається широкий спектр технологій і методів, що використовуються при створенні цифрових меморіальних платформ, із метою визначення оптимальних підходів до їх

реалізації та підвищення ефективності збереження історичної інформації в цифровому середовищі.

Теоретичне значення даного дослідження полягає у ґрунтовному аналізі та узагальненні знань щодо сучасних підходів до розроблення інформаційних систем, призначених для збереження історичної пам’яті. Робота сприяє розвитку наукових уявлень про методи організації, зберігання та обробки даних у цифрових архівах, а також розглядає новітні технології, що можуть бути використані для створення ефективних платформ меморіального типу. Окрім того, дослідження зосереджується на вивченні особливостей реалізації пошукових механізмів, алгоритмів оптимізації роботи з великими масивами інформації та можливостей інтеграції таких систем із іншими цифровими ресурсами. Аналіз наукової літератури та практичних реалізацій у цій сфері дає змогу узагальнити та систематизувати підходи до вирішення завдань цифрового увічнення пам’яті.

Практичне значення даної роботи полягає у можливості застосування отриманих результатів для розроблення ефективних інформаційних систем, що дають змогу систематизувати, зберігати та надавати доступ до інформації про загиблих воїнів. Запропоновані підходи можуть бути використані при створенні нових або вдосконаленні існуючих цифрових меморіальних платформ, забезпечуючи високу швидкість обробки запитів, зручний пошук за різними критеріями, а також можливість оновлення та верифікації даних.

Окремої уваги заслуговує соціальна значущість практичної реалізації дослідження. В умовах зростаючої потреби у збереженні національної пам’яті цифрові платформи стають важливим інструментом комеморації, що сприяє не лише збереженню історичних фактів, а й формуванню суспільної свідомості щодо внеску воїнів у захист держави. Запропоновані в дослідженні методи можуть бути адаптовані для інших подібних проєктів, які пов’язані з увічненням пам’яті, історичним архівуванням та створенням інтерактивних ресурсів для громадськості.

Таким чином, дослідження має як наукове, так і практичне значення, оскільки сприяє розширенню знань у сфері цифрового архівування та водночас пропонує конкретні шляхи реалізації інформаційних систем, що забезпечують ефективне збереження та поширення історичної пам’яті.

Апробація результатів дослідження. Основні результати дослідження опубліковано у науковій фаховій статті (фаховий журнал категорії Б):

Римар П.В., Журовський Я.О., Коновал М.С. Розробка веб-сайту про полеглих героїв Вінницької ОТГ. *Вісник Хмельницького національного університету. Технічні науки*. 2025.

Результати роботи обговорювалися на VI Всеукраїнській науково-практичній конференції здобувачів вищої освіти та молодих вчених «Прикладні інформаційні технології 2025» з публікацією тез доповідей:

Журовський Я.О., Римар П.В. Розробка серверної частини сайту про полеглих героїв на основі відкритих даних. *Прикладні інформаційні технології 2025*: Матеріали всеукр. науково-практ. конф. здобувачів вищ. освіти та молодих вчен., м. Вінниця, 22 трав. 2025р. 2025.

Отримані теоретичні та практичні результати знайшли своє практичне застосування в рамках співпраці з Вінницькою міською радою, де вони були використані для розробки інформаційного ресурсу, що дає змогу систематизувати дані про загиблих воїнів та забезпечити їх зручний доступ для громадян. Реалізація цього проєкту дозволила підвищити ефективність роботи з інформацією, забезпечити інтеграцію даних та розширити можливості пошуку та фільтрації інформації про загиблих героїв. Видана довідка Департаментом інформаційних технологій Вінницької міської ради про впровадження результатів роботи.

Таким чином, апробація результатів дослідження підтвердила його актуальність та практичну значущість, що дозволяє рекомендувати отримані висновки та розроблені підходи для подальшого застосування у різних сферах, пов’язаних зі збереженням та обробкою історичної інформації.

Структура роботи. Кваліфікаційна (бакалаврська) робота складається зі вступу, трьох розділів, висновків, списку використаних джерел в кількості 41 найменування та 6 рисунків.

У вступі обґрунтовано актуальність теми, сформульовано мету та завдання дослідження, визначено об’єкт і предмет дослідження, а також окреслено теоретичну та практичну значущість роботи.

Перший розділ присвячений аналізу сучасних підходів до збереження історичної пам’яті. У ньому розглянуто особливості цифрових меморіальних систем, здійснено порівняльний аналіз існуючих рішень та виявлено їхні переваги та недоліки. Додатково вивчено досвід застосування інформаційних технологій у сфері збереження пам’яті про загиблих воїнів, що дало змогу визначити ключові проблеми, які вирішуються у даній роботі.

Другий розділ містить опис концепції інформаційної системи для увічнення пам’яті загиблих воїнів. У ньому розглянуто основні вимоги до такої системи, визначено її архітектуру, функціональні можливості та методи організації даних. Представлено підходи до створення бекенду та бази даних, що забезпечують ефективну роботу з інформацією та швидкий доступ до неї.

Третій розділ присвячений аналізу ефективності запропонованих рішень та перспектив їхнього розвитку. У ньому розглянуто можливості інтеграції інформаційної системи з іншими цифровими платформами, досліджено шляхи вдосконалення її функціоналу та забезпечення довгострокової підтримки. Особливу увагу приділено оцінці продуктивності системи, тестуванню її ключових функцій та аналізу відгуків користувачів.

У висновках підбито підсумки дослідження, узагальнено отримані результати та сформульовано рекомендації щодо подальшого вдосконалення системи.

Список використаних джерел містить наукові та технічні праці, офіційні документи та інші інформаційні ресурси, які були використані при написанні роботи.

РОЗДІЛ 1
ПОСТАНОВКА ЗАДАЧІ ТА ОГЛЯД АНАЛОГІВ

1.1 Аналіз сучасного стану проблеми

Сучасний розвиток цифрових технологій значно змінив способи збереження, обробки та поширення інформації, відкриваючи нові можливості для суспільства у вшануванні пам’яті видатних особистостей та історичних подій. Особливої актуальності цей напрям набуває в умовах збройних конфліктів, коли збереження пам’яті про загиблих воїнів стає важливим моральним, соціальним та історичним завданням. Традиційні методи увічнення пам’яті, такі як друковані книги, меморіальні комплекси, музейні експозиції та встановлення пам’ятників, мають обмежену ефективність у сучасному інформаційному суспільстві. Вони не забезпечують швидкого доступу до необхідних даних, обмежені в можливостях оновлення інформації та часто потребують значних матеріальних ресурсів для створення та підтримки. У зв’язку з цим актуальною стає розробка цифрових інформаційних систем, які дають змогу зберігати, упорядковувати та надавати доступ до даних у режимі реального часу.

Цифрові технології значно розширюють можливості меморіальних платформ, оскільки вони забезпечують миттєве оновлення інформації, зручний доступ до архівних матеріалів, а також можливість інтеграції з іншими сервісами та базами даних. Наприклад, використання геолокаційних технологій дозволяє створювати інтерактивні мапи місць поховань, що дає змогу користувачам швидко знаходити необхідну інформацію. Крім того, сучасні інформаційні системи можуть використовувати штучний інтелект для аналізу великих обсягів даних, автоматичної категоризації записів, виявлення зв’язків між подіями та персоналіями.

На сьогодні існує кілька підходів до створення подібних цифрових платформ. Одні з них орієнтовані на збір та систематизацію біографічних даних загиблих воїнів, інші інтегруються з офіційними державними архівами та містять розширені відомості про бойові дії, місця загибелі та нагородження. Окремі

платформи передбачають можливість взаємодії користувачів із системою, дозволяючи додавати власні спогади, фотографії та документи, що робить процес збереження історичної пам’яті більш персоналізованим.

Ще одним важливим аспектом є безпека та достовірність даних у таких системах. Оскільки інформація про загиблих воїнів є чутливою, необхідно забезпечити надійні механізми її верифікації, а також захист від несанкціонованого доступу та маніпуляцій. Державні установи, громадські організації та приватні ініціативи використовують різні методи для підтвердження даних: офіційні архіви, військові документи, свідчення рідних та очевидців.

Отже, аналіз сучасного стану проблеми показує, що інформаційні технології відіграють ключову роль у збереженні пам’яті про загиблих воїнів. Вони дають змогу не лише упорядкувати та зберігати великі масиви даних, а й інтегрувати їх у єдину цифрову екосистему, забезпечуючи зручний доступ для широкого кола користувачів. Розробка спеціалізованих онлайн-платформ є важливим напрямом розвитку меморіальних проєктів, що дозволяє поєднати традиційні методи увічнення пам’яті із сучасними цифровими можливостями.

1.2 Аналіз існуючих аналогів

В Україні існує кілька інформаційних систем, які спрямовані на збереження пам’яті про загиблих воїнів та надання суспільству доступу до перевірених даних про них. Такі платформи розробляються як на державному, так і на громадському рівнях, проте кожна з них має свої особливості, функціональні можливості та певні недоліки, які впливають на ефективність використання.

Однією з найвідоміших українських платформ є «Національна Книга Пам’яті», яка містить інформацію про військовослужбовців, що загинули під час бойових дій. Ця система розроблена за підтримки державних органів і є офіційним джерелом даних. Основна перевага цієї платформи полягає в її авторитетності, адже інформація береться з перевірених джерел і постійно оновлюється. Водночас недоліком є обмежений функціонал пошуку та фільтрації

даних. Користувачі можуть знайти інформацію лише за базовими параметрами, такими як прізвище чи дата загибелі, проте більш детальні фільтри, наприклад, за місцем поховання або військовим підрозділом, відсутні. Крім того, інтерфейс платформи є досить статичним і не передбачає активної взаємодії з користувачами, що ускладнює процес внесення змін або доповнень.

Іншим прикладом є регіональні електронні книги пам'яті, які створюються на рівні обласних адміністрацій або громадських організацій. Такі ресурси часто мають розширену інформацію про загиблих воїнів конкретного регіону, включаючи фото, біографічні дані та місце поховання. Вони надають можливість місцевим жителям доповнювати інформацію та долучатися до увічнення пам'яті. Однак такі системи мають і певні недоліки. Однією з головних проблем є відсутність централізованої бази даних, що ускладнює пошук інформації про загиблих з інших регіонів. Крім того, через обмежене фінансування та відсутність єдиних стандартів деякі з цих платформ мають застарілий інтерфейс або працюють із перебоями.

Додатково можна відзначити ініціативи громадських організацій та волонтерів, які створюють онлайн-меморіали загиблих воїнів. Такі платформи часто мають інтерактивні можливості, зокрема, картографічний пошук місць поховання, можливість залишати коментарі та завантажувати фотографії. Однак їхньою слабкою стороною є недостатній рівень верифікації інформації, що може призводити до появи неточностей у даних. Окрім цього, багато з таких ініціатив мають локальний характер і не інтегровані з офіційними державними базами.

Проведений аналіз показує, що всі існуючі платформи мають як переваги, так і суттєві недоліки, які обмежують їх ефективність. Найбільш поширеними проблемами є відсутність єдиної централізованої бази даних, обмежені можливості фільтрації та пошуку інформації, складність оновлення даних і відсутність інтеграції з офіційними державними ресурсами. У зв'язку з цим існує потреба у створенні нової інформаційної системи, яка враховуватиме недоліки попередніх рішень, забезпечуватиме зручний пошук, автоматизоване оновлення інформації та високий рівень достовірності даних.

1.3 Постановка задачі дослідження

Аналіз існуючих інформаційних систем для збереження пам’яті про загиблих воїнів в Україні показав, що наявні рішення мають певні обмеження, зокрема, проблеми з централізованим доступом до інформації, недостатню інтеграцію з офіційними джерелами, а також обмежені можливості пошуку та фільтрації даних. У зв’язку з цим виникає потреба у створенні нової інформаційної системи, яка буде відповідати сучасним вимогам до цифрових меморіальних платформ.

Першочергово потрібно визначити загальні вимоги до інформаційної системи, що буде використовуватись для збереження пам’яті про загиблих воїнів. Це включає аналіз даних, які мають бути представлені на платформі, розробку вимог до пошукових механізмів, інтеграції з офіційними джерелами, підтримки актуальності інформації та забезпечення зручного доступу для користувачів. Важливо враховувати досвід існуючих рішень, проаналізувавши їхні переваги та недоліки, щоб уникнути повторення помилок та реалізувати найбільш ефективні підходи.

Наступним кроком є розроблення концепції цифрової платформи, яка дозволить не лише зберігати дані, а й надавати можливість їхнього ефективного пошуку, фільтрації та взаємодії з користувачами. Це включає розробку моделі бази даних, визначення структури інформації, що зберігатиметься у системі, а також вибір методів її обробки. Особливу увагу потрібно приділити питанням верифікації даних та їхнього оновлення, оскільки достовірність інформації є одним із найважливіших аспектів подібних систем.

Ще одним важливим завданням є вибір архітектури системи, що забезпечуватиме швидкий доступ до інформації, масштабованість, гнучкість у розширенні функціональності та інтеграцію з зовнішніми джерелами даних. Потрібно обґрунтувати використання конкретних технологій для створення бекенду, зокрема, вибір мови програмування, типу бази даних, серверного середовища та інших технічних аспектів, які впливатимуть на продуктивність системи. Також необхідно реалізувати розробку бекенду системи, включаючи

розгортання серверної частини, налаштування бази даних, створення API для роботи з інформацією, а також тестування основних функціональних можливостей. На цьому етапі важливо оцінити продуктивність системи, її зручність у використанні та можливість подальшого розширення.

Остаточним етапом дослідження є оцінка ефективності запропонованого рішення, що включає аналіз отриманих результатів, визначення сильних та слабких сторін реалізованої системи, а також рекомендації щодо її вдосконалення та можливих напрямів подальшого розвитку.

Таким чином, дослідження спрямоване на розробку інформаційної системи, що забезпечить зручний та ефективний доступ до даних про загиблих воїнів, надаватиме можливість швидкого пошуку інформації, підтримуватиме її актуальність та інтеграцію з офіційними джерелами. Отримані результати можуть бути використані для подальшого вдосконалення цифрових меморіальних платформ та їхнього впровадження у практичну діяльність державних установ та громадських організацій.

Висновки до розділу

У першому розділі було детально проаналізовано сучасні підходи до збереження історичної пам’яті за допомогою цифрових технологій. Розглянуто особливості використання інформаційних платформ, що забезпечують збереження даних про загиблих воїнів, та визначено ключові проблеми, які виникають при створенні та експлуатації таких систем. Зокрема, було встановлено, що більшість існуючих платформ мають обмежений функціонал, недостатню інтеграцію з офіційними джерелами, незручний пошук даних та відсутність автоматизованого оновлення інформації.

Окрім цього, проведено порівняльний аналіз функціональних можливостей та недоліків аналогічних систем, що використовуються в Україні. Це дало змогу визначити напрямки вдосконалення інформаційної системи, яка розробляється в межах дослідження. Виявлено, що одним із головних аспектів ефективності подібних рішень є використання сучасних технологій обробки та зберігання

даних, що забезпечують високу швидкість доступу до інформації, зручність її оновлення та інтеграцію з іншими цифровими сервісами.

Наступні розділи дослідження будуть присвячені деталізації архітектури системи, вибору відповідних технологічних рішень та їхньому практичному впровадженню. Також буде проведено тестування створеного програмного забезпечення, що дасть змогу оцінити його ефективність та визначити можливості для подальшого вдосконалення.

РОЗДІЛ 2
ОГЛЯД ТЕХНОЛОГІЙ ТА РІШЕНЬ

2.1. Вибір ASP.NET Core Web API як серверної платформи

У процесі розробки інформаційної системи постало питання вибору технології для реалізації серверної логіки. Після порівняння таких варіантів, як Node.js, Django, Spring Boot та ASP.NET Core, було обрано саме ASP.NET Core Web API [1]. Цей фреймворк забезпечує високу продуктивність, підтримує кросплатформеність, масштабованість, має вбудовану підтримку маршрутизації, контролерів, обробки запитів, а також потужний механізм Dependency Injection.

Важливу роль відіграла й зручна інтеграція з Swagger, що дозволяє легко тестувати API через браузер [6]. Завдяки знайомству з C# та середовищем Visual Studio, розробка велася швидко та впевнено [12]. Крім того, ASP.NET Core має хорошу підтримку логування, обробки помилок, middleware та легко адаптується до багаторівневої архітектури [35].

Для зберігання даних було обрано формат JSON – універсальний, читабельний і зручний як для машини, так і для людини. Такий вибір продиктований побажаннями замовника – Вінницької міської ради, яка надає відкриті дані саме у цьому форматі. У проєкті використовувалася бібліотека Newtonsoft.Json, яка дозволяє ефективно працювати з вкладеними структурами.

Як джерело даних було інтегровано CKAN API, який надає доступ до ресурсів із відкритими даними через RESTful інтерфейс [10] [32]. Саме через CKAN здійснюється запит до інформації про загиблих воїнів та місця їх поховань, що гарантує актуальність і структурованість отриманої інформації [4].

Під час реалізації також активно використовувалися допоміжні інструменти. Visual Studio виступала основним середовищем розробки, забезпечуючи зручну навігацію, автодоповнення та підтримку .NET 8. Swagger (OpenAPI) слугував як інтерфейс для тестування та документування HTTP-запитів. Для реалізації автоматичного оновлення даних при зміні JSON-файлів використовувався компонент FileSystemWatcher, що дозволив реалізувати

реактивну логіку без потреби перезапуску сервера [7]. А для фільтрації, пошуку та трансформації колекцій у C# була активно задіяна технологія LINQ, яка значно спростила обробку масивів даних [2].

Завдяки обраному технологічному стеку вдалося створити стабільну, гнучку й ефективну серверну частину, яка легко адаптується під нові вимоги та може бути масштабована в майбутньому.

2.2 Формалізація задачі та вимоги до системи

У межах розробки системи основним завданням було створити інструмент, який дозволяє не лише зберігати інформацію про загиблих воїнів, а й ефективно з нею працювати – фільтрувати, шукати, оновлювати, відображати у зручному форматі. Кожен запис у системі містить повний опис особи – від імені та звання до координат поховання, і ці дані мають бути доступні для гнучкої обробки.

Система повинна давати можливість користувачу знайти, наприклад, усіх, хто загинув у конкретному році, мав певне звання або нагороди, а також підтримувати пошук за частковим іменем. Особливу увагу приділено функції визначення «героїв дня» - захисників, які загинули цього календарного дня, що дає змогу щодня вшановувати конкретних осіб.

Вся взаємодія реалізована через RESTful API на базі ASP.NET Core, що дозволяє здійснювати HTTP-запити до контролерів, отримуючи дані у форматі JSON. Реалізовано функціонал для фільтрації, пошуку, визначення героїв дня, вивантаження координат поховань, а також – механізм автоматичного оновлення, який реагує на зміну JSON-файлів без необхідності ручного втручання [37].

Окрім функціональних можливостей, важливою складовою стали нефункціональні вимоги – зокрема, стабільність роботи, надійність при обробці помилок і можливість масштабування. Для цього система побудована на багаторівневій архітектурі, яка розділяє обробку запитів, логіку та доступ до даних. Реалізовано механізми логування, обробки винятків, автоматичного тестування через Swagger.

Таким чином, задача, яка стояла перед системою, була сформульована як створення доступного, стабільного та функціонально насиченого інструменту, що дозволяє працювати з даними про загиблих воїнів не лише з технічної, а й із меморіальної точки зору – гідно зберігати пам’ять і робити її доступною для громади.

2.3 Моделювання предметної області

Предметна область інформаційної системи охоплює збереження, обробку та подання структурованої інформації про загиблих воїнів, пов’язану не лише з їхніми біографічними даними, а й з місцем поховання, координатами, нагородами та інституційною інформацією. Для коректної обробки цієї інформації було створено модельну структуру, яка дозволяє ефективно поєднувати дані з кількох джерел.

Основним елементом системи є модель Soldier, яка представляє воїна й охоплює ключову інформацію: ім’я, звання, дати народження та смерті, належність до підрозділу, нагороди, зображення та унікальний QR-номер. Ці дані надходять із відкритого ресурсу opendata.gov.ua у форматі JSON, після чого десеріалізуються у відповідні C#-класи [5] [11].

Паралельно використовується модель RipSoldier, яка містить додаткову інформацію про поховання – прізвище, ім’я, по батькові, місце поховання, назву кладовища, координати, ID цвинтаря, а також інформацію від муніципальних служб. Ці записи надходять з іншого ресурсу CKAN API, але мають спільний ідентифікатор – qrNumber, який використовується як ключ для поєднання даних.

Завдяки цьому поля з різних джерел об’єднуються: один запис дозволяє дізнатися не тільки, хто і коли загинув, а й де похований, які точні координати могили, і чи був цей воїн відзначений державою.

Третьою суттєвою одиницею в моделі є Cemetery – об’єкт, що виділяється з інформації про місце поховання і дозволяє фільтрувати дані за кладовищами. Це особливо важливо для реалізації функцій географічної фільтрації або візуалізації на мапі.

Такий підхід дозволив побудувати логічно зв’язану структуру даних, у якій кожна модель доповнює іншу, забезпечуючи повноцінний і достовірний опис кожного героя. Це створює ґрунт не лише для відображення інформації, а й для її подальшої аналітики, інтеграції та вшанування пам’яті в цифровому форматі [24].

2.4 Побудова структури даних

Створення чіткої структури даних стало основою для всієї логіки функціонування системи. Саме на моделі спираються ключові операції – фільтрація, пошук, визначення героїв дня, оновлення інформації тощо. Для обміну даними між клієнтом і сервером були використані DTO (Data Transfer Object) – прості об’єкти без логіки, які містять лише потрібні поля, що полегшує передачу й обробку інформації.

Основною моделлю виступає Soldier, яка описує дані про воїна – ім’я, звання, дати, нагороди, підрозділ, зображення, унікальний QR-номер та інші супровідні поля. Вона містить як основну біографічну інформацію, так і посилання на зовнішні ресурси, зокрема сторінку пам’яті.

Модель RipSoldier доповнює Soldier і надає деталі щодо місця поховання: окремі поля ПІБ, координати, назву кладовища, причину смерті, інформацію про орган, що займався похованням. Важливо, що ці два типи даних не дублюються, а логічно пов’язуються через поле qrNumber.

Також було виділено окрему модель Cemetery, яка дозволяє зберігати ідентифікатор та назву кладовища, що стало необхідним для реалізації фільтрів та групування записів за місцем поховання.

Серед додаткових службових моделей – PaginatedResult для реалізації посторінкової навігації, FilterOption для зручного побудови списків фільтрів, SoldierSearchResult для компактного відображення результатів пошуку, а також SoldierCoordinates, яка містить ідентифікатор воїна і пару координат для візуалізації на мапі.

На основі цих моделей було побудовано UML-діаграму класів, яка демонструє зв’язки між основними сутностями. Незважаючи на те, що моделі Soldier і RipSoldier не пов’язані напряму, їх об’єднує загальне поле – QR-номер. Інші об’єкти використовуються як допоміжні компоненти у пошуку, фільтрації, візуалізації, що робить структуру даних гнучкою та масштабованою.

Типи даних у моделях обирались відповідно до очікуваної структури вхідних JSON-файлів. Використання nullable-типів для необов’язкових полів дозволило обробляти неповні записи без помилок під час десеріалізації.

В результаті створено оптимальну, модульну структуру моделей, яка повністю відображає предметну область, відповідає технічним вимогам та закладає основу для подальшого розширення функціональності системи.

2.5 Розробка алгоритмів

Щоб система виконувала не лише роль сховища даних, а й забезпечувала гнучку обробку запитів, у межах проєкту було реалізовано кілька важливих алгоритмів. Вони охоплюють задачі пошуку, фільтрації, автоматичного оновлення та визначення героїв дня [14].

HeroOfTheDaySelector відповідає за пошук захисників, які загинули саме в цей день, незалежно від року. Алгоритм ігнорує рік у даті смерті та порівнює лише день і місяць. Це дозволяє системі щодня виводити список для вшанування пам’яті, створюючи емоційно значущий компонент.

WeightedNameSearch – алгоритм пошуку за частковим ім’ям, який присвоює бали залежно від відповідності окремих частин імені (прізвища, імені, по батькові). Результати сортуються за ступеню відповідності, що особливо важливо при роботі з великими масивами даних, де користувач вводить неповну або неточну інформацію.

ReactiveDataUpdateWatcher – механізм автоматичного оновлення даних у системі. Завдяки використанню FileSystemWatcher, система слідкує за змінами у вхідних JSON-файлах і, за необхідності, перезавантажує їх без зупинки сервера.

Це реалізує реактивну поведінку, підвищуючи зручність користування та актуальність даних.

MultiAttributeFilter – багатокритеріальна фільтрація за званням, підрозділом, нагородами, роком смерті та статтю. Фільтри застосовуються динамічно, лише якщо передані користувачем. Для текстових полів із кількома значеннями (наприклад, нагороди) реалізовано попередню нормалізацію: перетворення у нижній регістр, видалення пробілів, розбиття на масиви.

Усі ці алгоритми написані з урахуванням гнучкості, продуктивності та зручності розширення. Вони суттєво підвищують цінність системи, роблячи її не просто інформаційною базою, а повноцінним інструментом для пошуку, аналізу та вшанування пам’яті.

2.6 Модель архітектури системи

Для реалізації програмної системи було обрано тришарову архітектуру, яка забезпечує чіткий розподіл відповідальностей і спрощує супровід, тестування та масштабування. Архітектура побудована на трьох логічних рівнях: контролери, сервіси та репозиторії [8] [15] [29].

Контролери є точкою входу для HTTP-запитів. Вони не містять бізнес-логіки, а лише приймають параметри, викликають відповідні методи сервісів і повертають результати. Наприклад, SoldiersController обробляє запити на фільтрацію, пошук, перегляд героїв дня, а RipSoldierController – роботу з похованнями.

Сервіси реалізують бізнес-логіку: обробку фільтрів, обрахунок відповідності пошуку, визначення героїв дня, тощо. Вони координують роботу між контролерами та репозиторіями, обробляючи дані у зручному вигляді [31].

Репозиторії працюють із джерелами даних – JSON-файлами. Вони завантажують, десеріалізують і оновлюють файли, реагуючи на їх зміни через FileSystemWatcher. Репозиторії надають зручні методи доступу до даних, що дозволяє сервісам працювати ефективно.

Окремим компонентом є HeroesApiClient, який відповідає за завантаження відкритих даних із CKAN API. Він формує запити, додає ключі авторизації та зберігає результати у форматі JSON для подальшого використання системою.

Уся система побудована за принципом ізольованих залежностей: контролери взаємодіють тільки з сервісами, сервіси – з репозиторіями, а ті – з файлами або зовнішніми API. Це дозволяє легко замінити одне джерело даних іншим (наприклад, перейти на базу даних) без масштабної перебудови структури [9].

Обрана архітектура показала себе гнучкою, стабільною та придатною для майбутнього розширення. Кожен компонент може бути протестований окремо, що значно спрощує розробку й підвищує надійність системи в цілому.

2.7 Робота з відкритими даними

Однією з ключових переваг розробленої системи є використання офіційних відкритих державних даних з національного порталу opendata.gov.ua [30]. Це забезпечує не лише достовірність джерела, а й підсилює публічну, суспільно важливу місію системи – гідно вшанувати пам’ять загиблих захисників.

Для інтеграції з відкритими даними використовується CKAN API – універсальна платформа, що надає REST-доступ до публічних ресурсів [26]. У проєкті задіяно два ключові ресурси: дані про загиблих воїнів і дані про місця поховань. Зчитування даних реалізовано через окремий компонент HeroesApiClient, який формує запити з вказаними resource_id, додає ключ авторизації у заголовок (за потреби) й обробляє відповіді.

Отримані дані надходять у структурованому форматі JSON, де основний масив інформації міститься в полі records. Далі ці записи зберігаються локально у файлах (result_heroes.json, result_ripheroes.json) і десеріалізуються у C#-об’єкти за допомогою бібліотеки Newtonsoft.Json [3] [28]. Це дозволяє працювати з ними як зі звичайними моделями .NET, не змінюючи структуру полів – повна відповідність формату на стороні джерела спрощує підтримку та адаптацію до змін [25].

Для підвищення продуктивності система працює з локально збереженими файлами, не звертаючись до зовнішнього API при кожному запиті. Запуск оновлення ініціюються вручну (через /api/data/update), або автоматично – за допомогою компонента FileSystemWatcher, який реагує на зміни у JSON-файлах і запускає оновлення репозиторіїв у реальному часі. Такий підхід дозволяє завжди тримати дані актуальними, не потребуючи ручного перезапуску чи втручання адміністратора [34].

Таким чином, система об’єднує зовнішню надійність CKAN, зручність локального кешування та гнучкість автоматичного оновлення. Це робить її стабільною, реактивною та готовою до роботи з великими масивами відкритих даних у реальних умовах.

2.8 Побудова моделі фільтрації та агрегації

Фільтрація – це одна з основних функцій інформаційної системи, яка дозволяє зручно працювати з великим обсягом даних. У нашому випадку її роль ще важливіша, адже користувачі часто шукають конкретну групу воїнів: за званням, роком загибелі, підрозділом чи іншими критеріями. Саме тому особливу увагу було приділено розробці моделі фільтрації, яка є простою у використанні, але водночас гнучкою та точною.

Робота над фільтрами почалась із вивчення формату вхідних даних. Було помічено, що значення деяких полів (наприклад, звання або корпус) мають неоднорідне оформлення – різний регістр, зайві пробіли, іноді й розбіжності в написанні. Щоб уникнути дублювання значень, перед фільтрацією ці поля нормалізуються: текст обрізається від пробілів, переводиться у нижній регістр і перетворюється на єдиний формат із великої літери.

Після цього система виконує агрегацію даних – підраховує, скільки записів має кожне з унікальних значень. Таким чином формується не просто перелік фільтрів, а ще й додається кількість елементів для кожного, що дозволяє користувачам швидко зорієнтуватися, скільки воїнів, наприклад, мають звання «Старший солдат» чи загинули у 2023 році.

Фільтри формуються по кількох полях: rank – звання, corps – підрозділ, awardsId – нагороди (які перед тим розбиваються на окремі значення), deathDate – з якого виділяється лише рік. Кожне значення перетворюється на об’єкт спеціальної моделі FilterOption, яка складається з трьох частин: фактичного значення (value), текстового заголовка (title) та кількості записів (count), які цьому значенню відповідають.

Ця модель використовується універсально – незалежно від типу фільтра, що дозволяє створити єдиний механізм для генерації фільтрів у контролерах API. Завдяки цьому розробникам не потрібно створювати окремі методи для кожного типу фільтра, а на клієнтському боці дані легко інтегруються в інтерфейс.

У підсумку, фільтрація в цій системі не лише дозволяє зручно звужувати пошук, а й виступає ключовим інструментом аналітики. Її реалізація базується на попередній нормалізації, точній агрегації та універсальній моделі представлення, що гарантує зручність і точність під час пошуку серед великого масиву даних [33].

2.9 Система координат та візуалізації

Однією з важливих особливостей системи стала можливість візуалізувати місця поховання загиблих захисників на мапі. Це не лише покращує сприйняття даних, а й дозволяє користувачам швидше знаходити конкретні місця для вшанування пам’яті.

Координати зберігаються у полі coordinates моделі RipSoldier, що надходить із відкритих даних CKAN API. Значення зберігаються у вигляді рядка, наприклад, «49.2403305, 28.4352071», і під час обробки перетворюються у модель SoldierCoordinates, яка містить SoldierId, широту (X) та довготу (Y). Це дозволяє точно визначити точку на мапі й прив’язати до неї додаткову інформацію про воїна.

Для зручності реалізовано окрему API-точку, яка повертає або всі координати, або лише для конкретного воїна, залежно від параметра soldierId. Це

створює гнучкість – можна як показати загальну мапу поховань, так і деталізовано відобразити дані про одну людину [21].

Надалі така система координат відкриває можливості для розширення функціоналу – створення зон поховань, візуалізації статистики або підключення картографічних сервісів. Уже зараз вона підвищує емоційний і практичний ефект від користування системою.

2.10 Інтеграція компонентів та загальна схема взаємодії

Після реалізації окремих функціональних модулів – контролерів, сервісів, репозиторіїв, моделей і алгоритмів – наступним кроком стало їхнє об’єднання в цілісну систему. Основне завдання цього етапу – забезпечити логічну взаємодію всіх компонентів для коректної обробки запитів і формування відповідей.

У структурі проєкту використовується тришарова архітектура. Контролери відповідають за приймання HTTP-запитів, сервіси – за обробку даних та виконання бізнес-логіки, а репозиторії – за доступ до джерел інформації (файлів або зовнішніх API). Кожен шар має свою чітку зону відповідальності, і саме така структура дозволяє системі залишатися стабільною та масштабованою [18].

Коли користувач надсилає запит – наприклад, на фільтрацію даних чи пошук за іменем – він потрапляє до відповідного контролера, який передає параметри у сервіс. Далі сервіс звертається до репозиторію, отримує всі необхідні дані, обробляє їх згідно з логікою (застосовує фільтри, сортує, ранжує, агрегує тощо) і передає результат назад у контролер. Контролер, своєю чергою, формує відповідь у форматі JSON і надсилає її клієнту [20].

Наприклад, при запиті /api/soldiers?rank=старший солдат&deathYear=2022, контролер SoldiersController викликає метод сервісу GetFilteredSoldiers, який завантажує список воїнів з JSON-файлу через SoldierRepository, застосовує фільтри, формує результат у вигляді PaginatedResult і передає його назад.

Весь цей процес реалізовано так, щоб бути якнайшвидшим, реактивним і надійним [39]. Зокрема, за актуальність даних відповідає компонент FileSystemWatcher, який автоматично реагує на зміну JSON-файлів, а за

взаємодію з відкритими державними джерелами – спеціалізований API-клієнт HeroesApiClient.

Компоненти не лише взаємодіють логічно, але й ізольовано – контролери не знають, звідки беруться дані, сервіси не залежать від формату запитів, а репозиторії можуть змінити джерело (наприклад, перейти з JSON на БД) без порушення всієї архітектури.

Завдяки такій побудові, система працює як єдиний механізм: зрозумілий, легко підтримуваний та готовий до розширення. Вона вже зараз забезпечує швидку обробку запитів і може адаптуватися до нових задач, технологій чи джерел даних без глобального переписування [16].

Висновки до розділу

У другому розділі було закладено фундамент для практичної реалізації інформаційної системи, яка обробляє, зберігає та візуалізує дані про загиблих захисників Вінницької територіальної громади. Проведено вибір ключових технологій для розробки На основі аналізу предметної області визначено основні об'єкти системи – Soldier, RipSoldier, Cemetery, SoldierCoordinates, кожен із яких відповідає окремим джерелам інформації.

В розділі розроблено й описано алгоритми, які забезпечують основну функціональність API: пошук із ваговим ранжуванням, фільтрація за кількома параметрами, автоматичне оновлення при зміні даних, а також визначення героїв дня за поточною датою. Всі ці рішення адаптовані до обробки великих масивів даних і забезпечують гнучку роботу системи.

Окремо було сформовано модель фільтрації, що дозволяє агрегувати значення з різних полів і надавати користувачу список доступних фільтрів. Також реалізовано логіку роботи з координатами поховань, що закладає основу для геовізуалізації у подальших версіях проєкту.

Архітектура побудована за тришаровою моделлю, що включає контролери, сервіси та репозиторії. Такий підхід дозволив чітко розмежувати обов’язки

компонентів системи, забезпечити зручну підтримку та масштабованість у майбутньому.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

3.1 Опис середовища розробки

Visual Studio 2022 – це основне інтегроване середовище розробки, яке використовувалося протягом усього циклу реалізації системи [38]. Саме воно дозволяло зручно працювати з проєктом, структурувати файли, автоматично керувати залежностями, використовувати автодоповнення коду, запускати проєкт локально, налагоджувати логіку, підключати зовнішні бібліотеки через NuGet та тестувати API завдяки вбудованій підтримці запуску Swagger UI. Висока інтеграція з ASP.NET Core Web API дала змогу швидко вбудовувати нові функції та забезпечити продуктивність на кожному етапі роботи.

.NET 8 (ASP.NET Core Web API) – обрана серверна технологія, що забезпечила основу для реалізації API. Вона є сучасною, стабільною, кросплатформною платформою з підтримкою REST, маршрутизації, Dependency Injection, middleware-логіки та модульності. Завдяки мінімалістичному підходу у .NET 8, усі частини API легко зчитуються, масштабуються, розширюються, а також повністю підтримують концепцію тришарової архітектури. Усі контролери, сервіси й репозиторії взаємодіють між собою в межах стандартної структури ASP.NET Core [13].

JSON – формат, обраний для зберігання й обміну даними. Він був не лише зручним з точки зору реалізації, але й рекомендований замовником як основний спосіб взаємодії з відкритими даними. Усі отримані дані з CKAN API були представлені саме у цьому форматі. За допомогою бібліотеки Newtonsoft.Json вдалося ефективно десеріалізувати об’єкти, зберігати вкладені структури, працювати з null-значеннями та точно відповідати схемі вхідних даних. JSON-файли result_heroes.json і result_ripheroes.json слугують локальними копіями отриманих даних для обробки в системі.

CKAN API – основне джерело даних, з яким взаємодіє система. Це інтерфейс платформи відкритих даних opendata.gov.ua, через який отримуються

офіційні ресурси про загиблих воїнів та їхні поховання. Усі запити виконуються через спеціально створений компонент HeroesApiClient, який формує запити, додає ключі авторизації, отримує відповіді та зберігає їх локально. Завдяки використанню CKAN API, система базується на достовірній, державній інформації, що гарантує актуальність і верифікованість джерел.

FileSystemWatcher – важливий компонент, який дає змогу автоматично відстежувати зміни JSON-файлів на диску. Якщо файли оновлено (наприклад, у результаті нового запиту до CKAN), система автоматично перезавантажує відповідні репозиторії без перезапуску програми. Це дозволяє реалізувати реактивну поведінку, при якій система динамічно реагує на зміни в даних.

LINQ – вбудована мова запитів у C#, що стала ключовим інструментом для реалізації фільтрації, пошуку, сортування, агрегації та об’єднання даних. Саме через LINQ реалізовано більшість логіки у сервісах, таких як SoldierService та RipSoldierService. Його виразність і зручність дозволили писати компактний, читабельний та гнучкий код, що відповідає складним умовам обробки запитів.

Swagger (OpenAPI) – невід’ємна частина процесу тестування та документації. Після налаштування в Startup класі він автоматично генерує інтерфейс для взаємодії з усіма маршрутами API. Swagger став основним інструментом ручного тестування під час розробки – через нього перевіряли фільтрацію, пошук, визначення героїв дня, роботу з координатами, оновлення файлів, а також обробку помилок і повідомлень.

Загалом, середовище розробки було сформоване з урахуванням актуальних вимог до стабільності, швидкості, адаптивності та підтримки відкритих стандартів. Вибрані технології чудово поєднуються між собою, утворюючи єдину, узгоджену інфраструктуру, що дозволила ефективно реалізувати весь функціонал системи та забезпечити її готовність до масштабування й подальшої розробки.

3.2 Реалізація структури проєкту

Після завершення етапу моделювання та вибору архітектурного підходу розпочалася безпосередня реалізація проєкту. Структура програмного рішення була організована відповідно до принципів чистої архітектури, де кожен компонент має чітко визначену відповідальність. Це забезпечує зручність навігації по проєкту, легкість у підтримці коду, а також дозволяє ефективно розділити логіку між модулями.

Розробка велась у межах одного основного проєкту з назвою HeroesApi, побудованого на основі ASP.NET Core Web API [17]. У кореневій структурі були створені стандартні директорії, які відображають функціональний поділ системи: контролери, сервіси, репозиторії, моделі, конфігураційні файли та директорія для тимчасових JSON-даних.

Контролери зосереджені у папці Controllers. Вони відповідають за обробку зовнішніх HTTP-запитів та маршрутизацію до відповідних сервісів. Усі вони реалізовані як API-контролери з відповідними атрибутами маршрутизації [22]. Основні з них: SoldiersController, який опрацьовує запити фільтрації, пошуку, деталізації та визначення героїв дня; RipSoldierController, який працює з даними про поховання, кладовища і координати; DataController, що викликає оновлення JSON-файлів; та HeroesController, який використовується для тестових викликів до CKAN API.

У папці Services зосереджено всю бізнес-логіку. Тут реалізуються алгоритми обробки даних, зокрема логіка фільтрації, пошуку, підрахунку значень фільтрів, ранжування результатів пошуку, визначення героїв дня. SoldierService та RipSoldierService є основними класами, які обробляють вхідні дані, викликають методи репозиторіїв і формують результат у вигляді DTO-структур. Усі сервіси впроваджуються в систему через механізм Dependency Injection, що дозволяє зручно тестувати, замінювати або розширювати логіку.

Репозиторії знаходяться у папці Repositories та відповідають за прямий доступ до джерел даних. Оскільки система працює з JSON-файлами, репозиторії зчитують їх, десеріалізують в об’єкти C# та забезпечують зручні методи

отримання даних. SoldierRepository працює з файлом result_heroes.json, а RipSoldierRepository – з result_ripheroes.json. Обидва репозиторії підтримують автоматичне оновлення даних завдяки інтеграції з FileSystemWatcher – компонентом, що реагує на зміну файлів у реальному часі та викликає перезавантаження відповідних колекцій.

Усі моделі знаходяться у папці Models. Це набір DTO-об'єктів, що представляють структуру вхідних та вихідних даних: Soldier, RipSoldier, Cemetery, SoldierCoordinates, FilterOption, PaginatedResult<T>, SoldierSearchResult. Кожна модель точно відповідає структурі JSON-файлів з урахуванням назв полів, які описані за допомогою атрибутів [JsonProperty("…")]. Це дозволяє уникати помилок десеріалізації та гарантує, що структура об'єктів у програмі відповідає фактичним даним, які надаються порталом відкритих даних.

У папці publish зберігаються ті самі JSON-файли, які надходять від CKAN API. Вони виступають кешованими копіями, що використовуються системою під час виконання запитів. Ця папка є тимчасовою і не входить до системи контролю версій, оскільки її вміст постійно змінюється.

Файл Program.cs відповідає за конфігурацію програми: налаштовуються маршрути, додаються всі сервіси та клієнти, підключається Swagger для документування API, а також реєструється HeroesApiClient, що виконує запити до зовнішнього CKAN API. Тут також відбувається запуск вебсервера.

Завдяки такій структурі система залишається логічно впорядкованою, легко читається, масштабовано зростає та готова до командної роботи. Кожен розробник може швидко знайти потрібну частину проєкту, зрозуміти її роль і при необхідності – внести зміни без ризику порушення загальної логіки або архітектури системи.

3.3 Реалізація API-методів

У межах реалізації інформаційної системи було створено повноцінний набір RESTful API-методів, що забезпечують усю необхідну функціональність для роботи з даними. Кожен метод виконує конкретну задачу – фільтрацію,

пошук, оновлення, агрегацію або отримання координат – і працює у форматі JSON, що робить їх універсальними та легко інтегрованими.

Основна частина логіки зосереджена в контролері SoldiersController, який обробляє запити, пов’язані з даними про загиблих захисників. Метод для отримання списку солдатів підтримує параметри для пагінації, пошуку, фільтрації за званням, корпусом, нагородами, роком загибелі та статтю. Це дозволяє зручно й ефективно формувати результати під конкретні потреби користувача. Крім цього, є окремий метод для отримання доступних фільтрів, які генеруються динамічно на основі наявних даних, що дозволяє відображати лише ті варіанти, які реально присутні у системі.

Особливу функціональність реалізовано у вигляді методу визначення «героїв дня», який повертає тих, хто загинув саме цього календарного дня, незалежно від року. Це виконується через просте порівняння дня та місяця дати смерті з поточною датою. Також у SoldiersController реалізовано детальний перегляд одного захисника за його унікальним ідентифікатором. Система об’єднує дані з двох джерел – воєнної та меморіальної – через поле qrNumber. Пошук за частковим ім’ям реалізовано з урахуванням вагового ранжування: результат впорядковується так, щоб найбільш релевантні відповіді з’являлися першими.

Дані про місця поховання обробляє контролер RipSoldierController. Він дозволяє отримувати список захисників із другого джерела – похоронного реєстру – з можливістю пошуку та фільтрації за ідентифікатором кладовища. Окремий метод відповідає за надання GPS-координат для подальшого відображення на мапі. Його перевага в тому, що він може повертати як усі координати, так і координати конкретного воїна. Це відкриває можливості для інтерактивної візуалізації. Додатково реалізовано два методи для роботи з кладовищами: отримання списку всіх унікальних назв та отримання назви за ID.

Ключовий технічний компонент, що забезпечує оновлення даних, зосереджений у DataController. Його метод дозволяє вручну оновити локальні JSON-файли, завантаживши нові дані з CKAN API. Після оновлення ці файли

десеріалізуються в пам’ять, що забезпечує актуальність усіх наступних запитів. Така можливість важлива для адміністратора системи, який може ініціювати оновлення без потреби перезапуску програми чи змін у коді [27].

Допоміжну функцію виконує контролер HeroesController, який дозволяє вручну перевірити доступ до CKAN API. Тут реалізовано два тестові маршрути – один для перевірки завантаження даних про загиблих, інший – про поховання. Вони не використовуються у продакшн-середовищі, але допомагають переконатися в наявності зв’язку з відкритими державними джерелами.

Всі API-методи побудовані за принципами REST. Вони мають чітко описані маршрути, підтримують параметри запитів, повертають дані у форматі JSON і легко тестуються через Swagger UI. Завдяки цьому розробка та тестування API були швидкими та прозорими, а структура методів – передбачуваною й зручною для розширення. У підсумку, реалізовані API забезпечують повний цикл обробки даних: від зчитування й оновлення до складної фільтрації, пошуку, аналізу й виведення інформації у зручному форматі. Система готова до реального використання та подальшої інтеграції з зовнішніми клієнтами.

3.4 Візуалізація координат і взаємодія з картою

Однією з важливих складових функціональності розробленої інформаційної системи є можливість візуалізувати дані про місця поховання захисників на географічній мапі. Це не лише розширює межі сприйняття даних, а й виконує практичну функцію – дозволяє швидко знайти поховання конкретної особи або переглянути розташування кладовищ, де поховані загиблі воїни. Реалізація цього компоненту стала можливою завдяки правильній організації обробки координат та створенню відповідної структури.

Інформація про координати отримується із зовнішнього JSON-файлу result_ripheroes.json, де кожен об’єкт містить поле "coordinates" у вигляді рядка, що зберігає широту та довготу, розділені комою. Наприклад: "49.203491, 28.440406". Оскільки дані імпортуються з відкритого API у такому вигляді, було

вирішено не змінювати формат на етапі збереження, а вже під час запиту до API розбивати координати на окремі числові значення, зручні для візуалізації.

Для обробки координат у системі створено окрему DTO-модель SoldierCoordinates. Вона містить три поля: унікальний ідентифікатор солдата (SoldierId), широту (X) та довготу (Y). Ця структура є максимально простою, але повністю достатньою для інтеграції з будь-яким клієнтським рішенням на кшталт Leaflet або Google Maps. Модель дозволяє передавати до клієнта лише ту інформацію, яка безпосередньо необхідна для побудови точок на мапі, не перевантажуючи систему зайвими даними.

API-система передбачає окрему кінцеву точку, яка надає координати у форматі SoldierCoordinates. Метод доступний за адресою /api/rip-soldiers/soldiers/coordinates і підтримує як запит усіх координат одразу, так і координат лише одного солдата за його ID. Такий підхід дозволяє реалізувати два основні сценарії використання: відображення загального розподілу поховань або виведення окремого місця на персональній сторінці.

Усі координати формуються в репозиторії RipSoldierRepository, де виконується перевірка наявності координат у записі, а також розділення рядка на широту та довготу. Таким чином, навіть за великого обсягу даних підготовка результатів відбувається швидко та ефективно. Завдяки простоті структури, ці координати можуть бути безпосередньо інтегровані у клієнтські картографічні компоненти без додаткової трансформації.

З практичного боку, така реалізація дозволяє користувачу швидко знайти місце поховання потрібного захисника, побачити відображення всіх точок на території міста чи регіону, а в майбутньому – розширити функціонал до кластеризації, фільтрації поховань за категоріями або побудови маршрутів.

Загалом, система координат і візуалізації дозволяє поєднати дані з реальним географічним простором, підвищити ефективність використання платформи та покращити користувацький досвід. Такий функціонал є важливою складовою цифрового вшанування пам’яті загиблих, а також практичним інструментом навігації для всіх, хто хоче вшанувати героїв особисто.

3.5 Взаємодія з CKAN API та обробка JSON

Інформаційна система, розроблена в межах цього проєкту, працює з офіційними відкритими даними, розміщеними на державному порталі opendata.gov.ua. Для цього було реалізовано повноцінну взаємодію з публічним програмним інтерфейсом CKAN – API, який дозволяє отримувати актуальні дані у форматі JSON у режимі реального часу. Саме через цей механізм система отримує списки загиблих воїнів та інформацію про їхнє поховання.

Основним елементом, що забезпечує взаємодію з цим API, є клас HeroesApiClient. Його мета – відправити запит на зовнішній ресурс, отримати відповідь, зберегти отриманий JSON у локальному вигляді, а також повернути його для подальшої обробки. Клієнт працює асинхронно, що дозволяє не блокувати роботу основного потоку, забезпечуючи при цьому стабільність і високу швидкість завантаження.

Щоб отримати дані, в систему передається URL-адреса ресурсу CKAN (включно з resource_id, що є унікальним ідентифікатором потрібного набору), а також умовне ім’я файлу, до якого потрібно зберегти результат. Наприклад, для основного ресурсу використовується ID 41aea193-a212-499a-9b94-90f341b958e4, який містить перелік загиблих військовослужбовців. Запит будується у вигляді відкритого посилання, після чого надсилається через вбудований HttpClient.

Ключовим елементом у відповіді CKAN є масив records, який містить перелік об’єктів у форматі JSON. Саме ці об’єкти десеріалізуються у DTO-моделі Soldier або RipSoldier за допомогою бібліотеки Newtonsoft.Json. Для забезпечення коректної відповідності полів між JSON і моделями використовується атрибут [JsonProperty("...")].

Одразу після отримання даних, клієнт зберігає JSON у вигляді локального файлу. Це дозволяє уникнути повторних звернень до API та створює можливість для роботи з кешованими даними. Усі файли зберігаються у папці publish, з назвами result_heroes.json та result_ripheroes.json відповідно. Форматування відповіді виконується у вигляді читабельного дерева (Formatting.Indented), що робить ці файли зручними як для перегляду, так і для дебагу.

Окрему увагу варто приділити компоненту автоматичного оновлення. Завдяки використанню FileSystemWatcher, система стежить за будь-якими змінами у JSON-файлах. Якщо файл перезаписується (наприклад, через виклик ручного оновлення /api/data/update), відповідний репозиторій автоматично перезавантажує дані, без необхідності перезапуску сервісу. Це забезпечує реактивність та мінімізує затримки між оновленням на стороні API і відображенням актуальної інформації у користувача.

У підсумку, реалізація взаємодії з CKAN API надала системі можливість отримувати офіційні, достовірні й структуровані дані з відкритого державного ресурсу. Водночас збереження у форматі JSON, як вимагалося замовником, забезпечило гнучкість і прозорість з боку системи. Усе це разом формує надійну основу для подальшої обробки, фільтрації, візуалізації та аналітики.

3.6 Реалізація алгоритмів

Функціональність інформаційної системи базується не лише на збереженні та виведенні даних, а й на низці спеціалізованих алгоритмів, які дозволяють виконувати складні операції з фільтрації, пошуку, агрегації та оновлення. У цьому підрозділі розглянуто, як саме було реалізовано алгоритми, попередньо описані в теоретичній частині.

Одним із ключових функціональних компонентів системи є можливість визначення героїв дня. Для цього реалізовано алгоритм, що дозволяє порівнювати дати загибелі солдатів із поточною датою, ігноруючи при цьому рік. Такий підхід дає змогу показати саме тих захисників, які загинули цього дня в різні роки, створюючи щоденне вшанування. У реалізації для кожного запису формується дата у вигляді лише дня та місяця, і вона зіставляється з аналогічним представленням поточної дати. Результатом є колекція записів, яку API повертає через окремий маршрут.

Для покращення користувацького пошуку за частковим ім’ям було реалізовано алгоритм вагового ранжування. Його суть полягає в тому, щоб визначити, наскільки запит користувача відповідає кожному запису в базі. Для

цього прізвище, ім’я та по батькові кожного запису порівнюються з пошуковим запитом, а кожна відповідність оцінюється з певною вагою: найбільшу вагу має збіг у прізвищі на початку слова, меншу – в імені або по батькові. Завдяки цьому система не просто фільтрує результати, а видає їх у впорядкованому вигляді, підвищуючи зручність і точність пошуку.

Ще однією важливою частиною стало впровадження механізму реактивного оновлення даних. Система повинна миттєво реагувати на зміну вмісту JSON-файлів, не чекаючи на перезапуск або ручне втручання. Для цього було використано компонент FileSystemWatcher, який відслідковує зміни у файлах, що зберігаються локально, і при їхньому оновленні автоматично викликає методи перезавантаження даних у відповідному репозиторії. Це дозволяє тримати систему постійно синхронізованою з відкритими даними, що оновлюються вручну або через виклики до CKAN API.

Також у системі реалізовано алгоритм агрегації фільтрів, який дозволяє зібрати унікальні значення для різних параметрів – таких як звання, підрозділ, нагороди, рік смерті або стать. Цей алгоритм працює динамічно: під час кожного запиту до /filters система сканує всі наявні записи, нормалізує значення (прибирає зайві пробіли, зводить до нижнього регістру, капіталізує першу літеру), об’єднує їх і повертає у вигляді набору списків. Таким чином, фільтри завжди є актуальними, навіть якщо структура даних або самі значення змінюються.

Усі зазначені алгоритми працюють без суттєвих затримок навіть за обробки великої кількості записів, не потребують сторонніх бібліотек та легко масштабуються. Їхня реалізація базується на вбудованих можливостях мови C# та фреймворку .NET, що дозволяє досягти хорошого балансу між продуктивністю та простотою підтримки коду.

3.7 Тестування системи

Після реалізації основної логіки та налаштування API наступним етапом стало проведення повноцінного тестування системи. Оскільки мова йде про серверну частину, основну увагу було приділено перевірці роботи всіх HTTP-

методів, обробці помилок, правильності виводу даних та стійкості системи до нештатних ситуацій. Для цього використовувалося ручне тестування через вбудований інтерфейс Swagger UI, який дозволяє викликати всі кінцеві точки прямо з браузера, задаючи вхідні параметри та одразу бачити результат.

Під час тестування перевірялися всі основні запити. Зокрема, успішно пройдено перевірку для маршруту /api/soldiers яку показано на рис. 3.1, де було протестовано роботу фільтрації за різними параметрами, включно з пошуком за частковим ім’ям, фільтрацією за званням, нагородами, роком загибелі та статтю. Кожен запит повертав дані у форматі JSON, відповідно до описаної моделі, без втрат полів або порушень структури [36]. Також було перевірено пагінацію – результати змінювались відповідно до заданого розміру сторінки.

Рисунок 3.1 – Тестування запиту /api/soldiers

Особливу увагу приділено перевірці маршруту /api/soldiers/heroesOfTheDay яке показано на рис. 3.2, де реалізовано алгоритм визначення героїв за календарною датою. У відповідь повертались лише ті

записи, дати загибелі яких збігалися з поточним днем та місяцем, що підтвердило коректність реалізації логіки без урахування року.

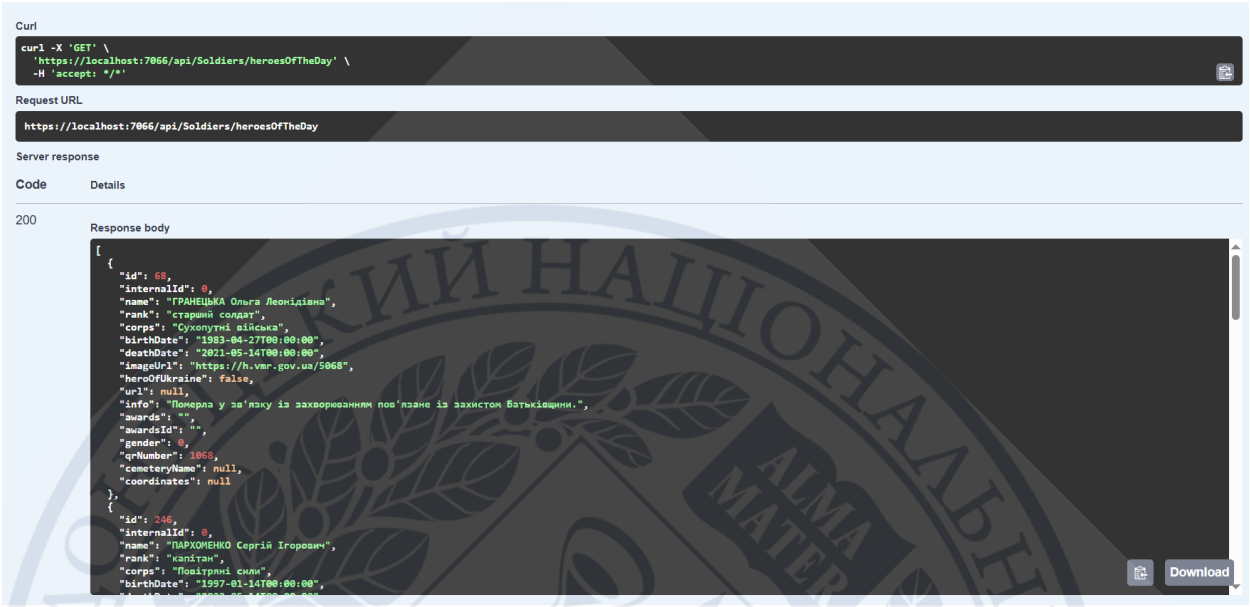

Рисунок 3.2 – Тестування запиту /api/soldiers/heroesOfTheDay

Також тестувались маршрути для отримання фільтрів та пошуку: /api/soldiers/filters і /api/soldiers/search які показані на рис. 3.3 та 3.4. Перший повертав актуальні списки значень, які можна використовувати у фронтенді для фільтрації, другий – демонстрував впорядковані результати з частковим збігом пошукового запиту.

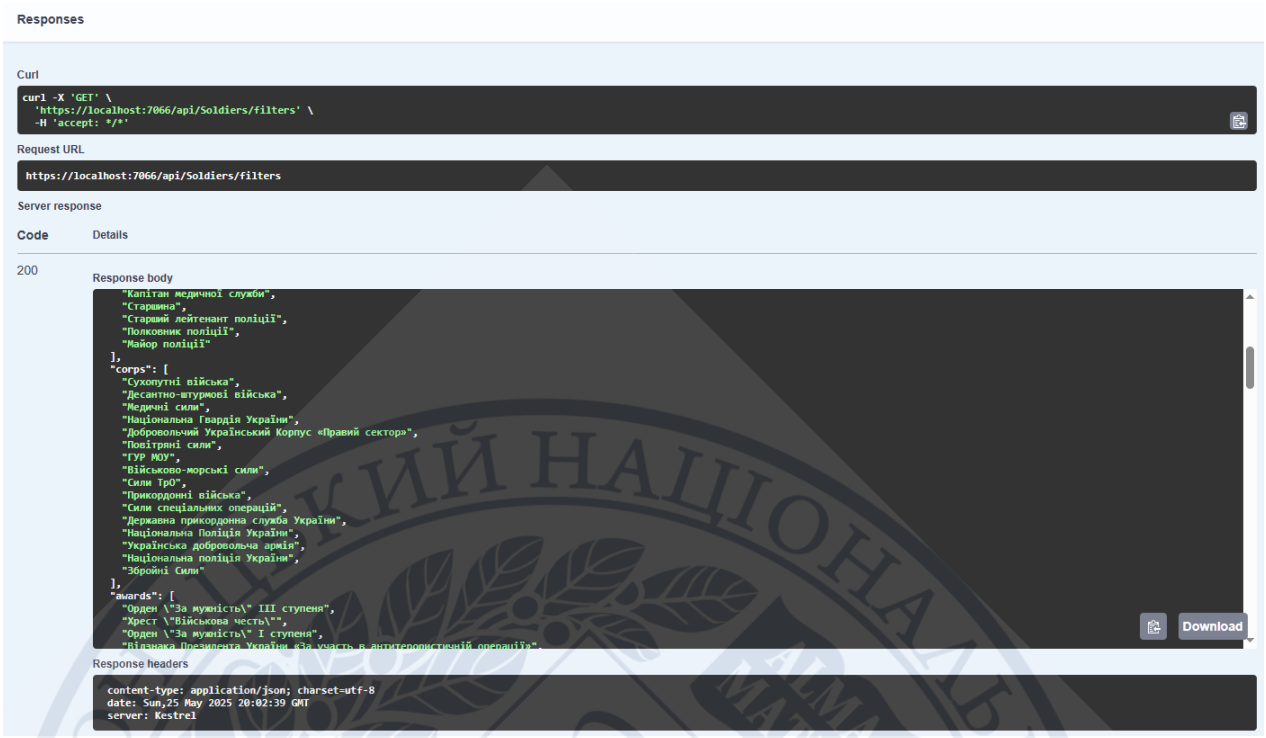

Рисунок 3.3 – Тестування запиту /api/soldiers/filters

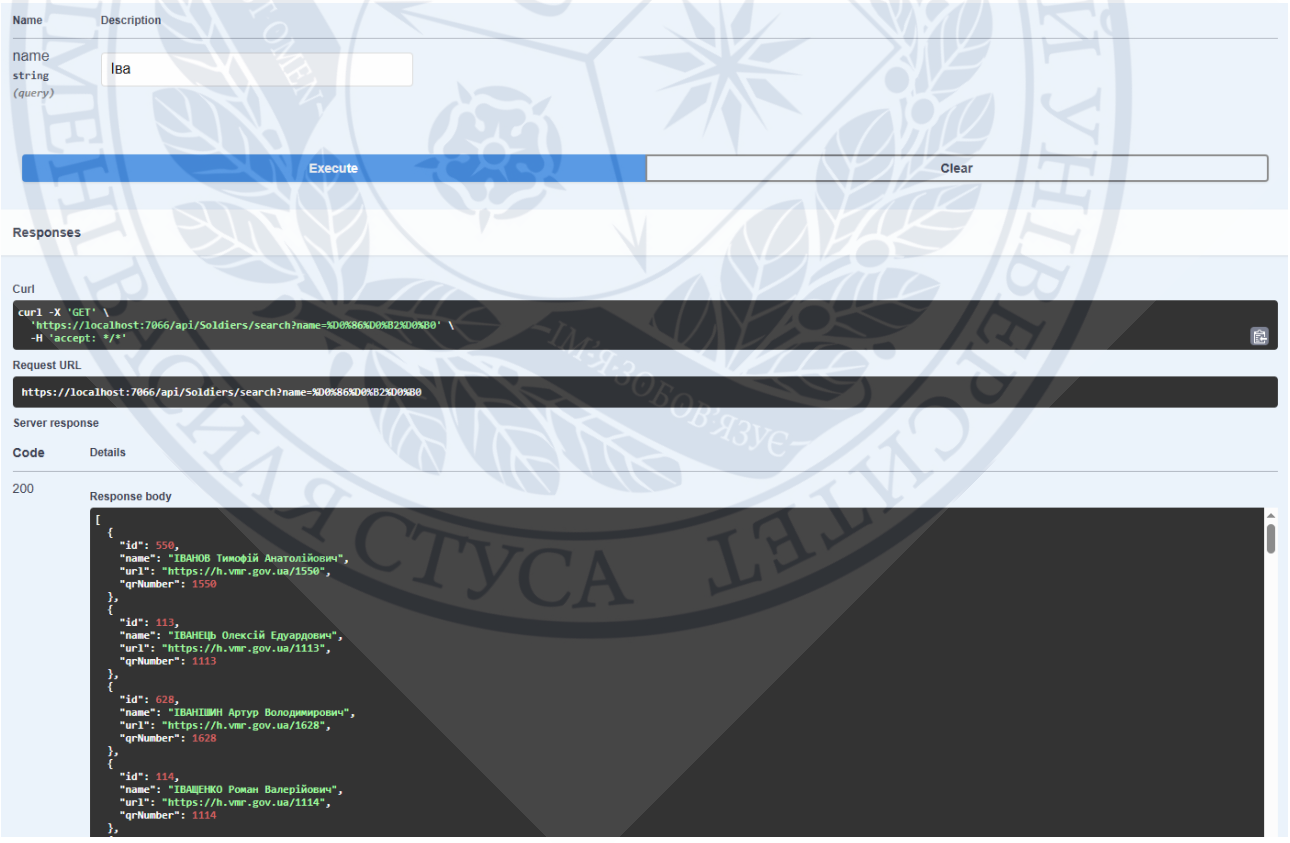

Рисунок 3.4 – Тестування запиту /api/soldiers/search

Було проведено тестування методів, пов’язаних із даними поховань – зокрема, запити до /api/rip-soldiers принцип роботи такий як у /api/soldiers, координат /api/rip-soldiers/soldiers/coordinates зображено на рис. 3.5, а також до переліку кладовищ /api/rip-soldiers/cemeteries зображено на рис. 3.6. Усі запити відповідали очікуваним результатам і дозволяли вивести інформацію про поховання або використовувати координати для виведення на мапу.

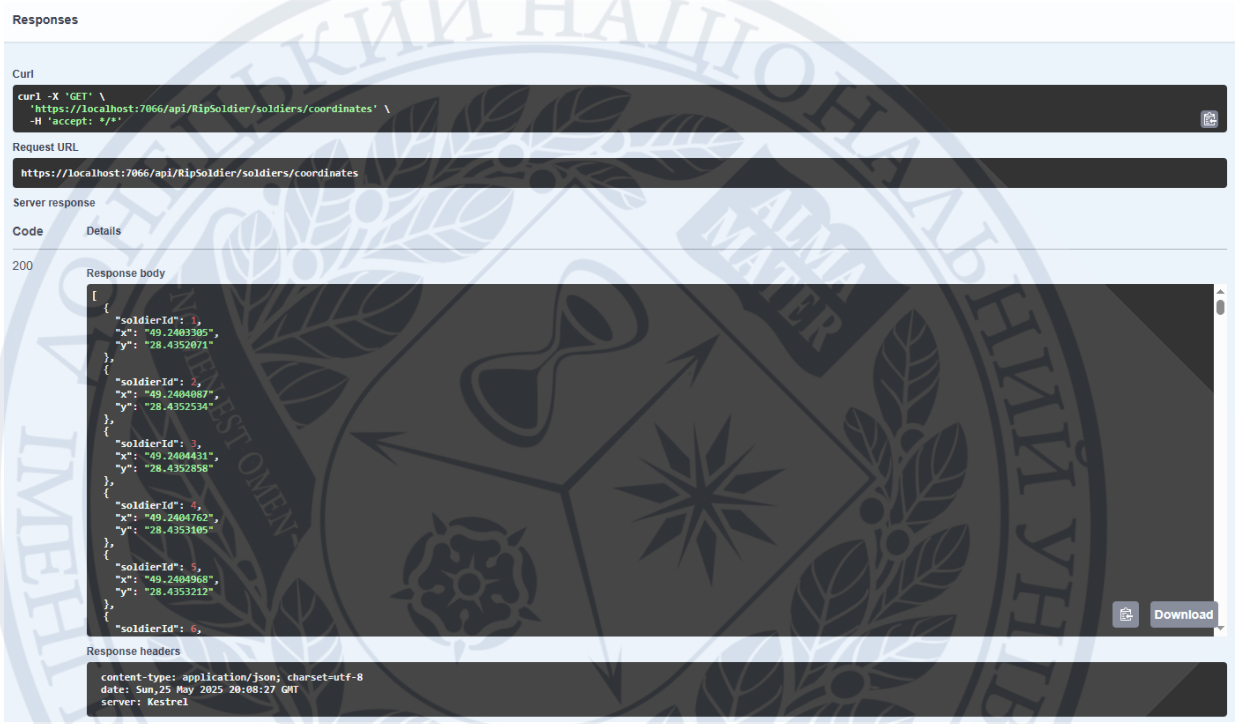

Рисунок 3.5 – Тестування запиту /api/rip-soldiers/soldiers/coordinates

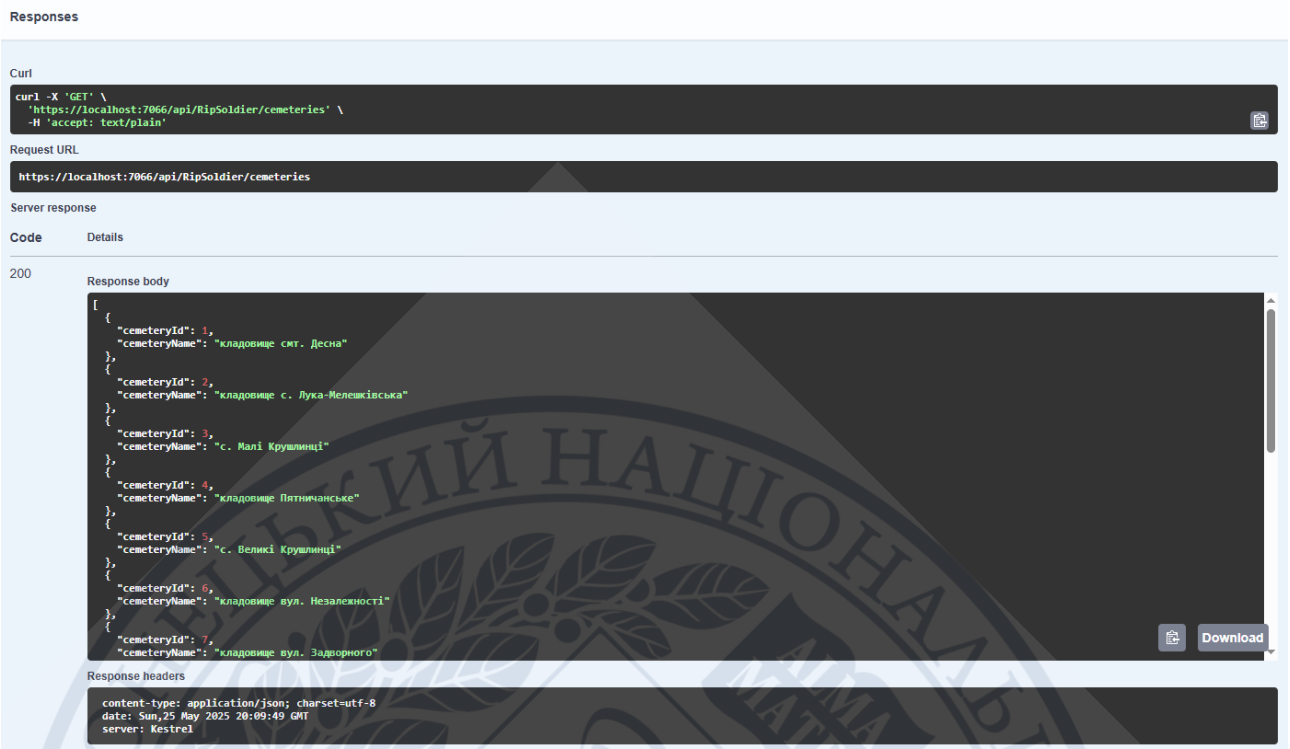

Рисунок 3.6 – Тестування запиту /api/rip-soldiers/cemeteries

Валідація даних відбувається переважно на рівні моделей та логіки фільтрації. Система стійка до відсутніх параметрів, неправильного формату дат або порожніх значень – такі запити не викликають збоїв, а просто повертають порожній результат або повідомлення про помилку. Це свідчить про достатній рівень обробки вхідних даних.

Також було здійснено початкову оцінку продуктивності. Запити до локального API, навіть із фільтрацією по 1000+ записах, виконувалися за частки секунди, що є хорошим показником для проєкту, що працює без СУБД. Особливо ефективною виявилася обробка координат – навіть повний список передавався за кілька мілісекунд завдяки простоті моделі SoldierCoordinates.

У результаті тестування система показала стабільну та передбачувану поведінку. Всі реалізовані методи функціонують відповідно до специфікації, правильно обробляють параметри, коректно реагують на помилки та видають результат у структурованому вигляді. Ручне тестування через Swagger дозволило покроково перевірити всю логіку роботи сервера без потреби в додаткових інструментах.

3.8 Аналіз результатів реалізації

Результати розробки системи дозволяють зробити висновок, що поставлену мету було досягнуто повною мірою. В рамках проєкту вдалося створити працездатний серверний застосунок, який забезпечує стабільну, зручну та масштабовану роботу з чутливою інформацією – даними про загиблих захисників. Ключова ідея полягала не просто у створенні базового API, а в розробці гнучкої основи, яка дозволяє якісно обробляти, структурувати та оновлювати інформацію з відкритих державних ресурсів.

Важливо зазначити, що реалізована система демонструє високий ступінь відповідності задачам, які були поставлені на етапі планування. Реалізовано всі заплановані API-методи: фільтрація за кількома критеріями, пошук за частковим іменем, динамічна побудова фільтрів, визначення героїв дня, доступ до координат поховань, ручне оновлення з CKAN API та перегляд інформації про кладовища. Кожен метод протестовано, а результати роботи відповідають очікуванням – це підтверджується як під час ручного тестування, так і за відгуками з боку користувачів у ході демонстрації.

Окремо варто відзначити, що обрана архітектура на базі ASP.NET Core Web API з використанням трьох логічних шарів (контролери, сервіси, репозиторії) продемонструвала свою ефективність. Таке розділення забезпечує легкість підтримки, можливість окремого тестування компонентів та подальше масштабування – наприклад, у вигляді додавання фронтенду, інтеграції з іншими API або переходу до роботи з базою даних у майбутньому. Реалізація через DI (впровадження залежностей) дозволила забезпечити слабке зв’язування між модулями та дотримання принципів SOLID [23].

З технічної точки зору, система показала високу продуктивність навіть без використання повноцінної СУБД. Операції фільтрації, агрегації та пошуку виконуються виключно в оперативній пам’яті, працюючи з десеріалізованими JSON-даними. При тестуванні на реальних наборах, що містять тисячі записів, відповіді від API надходили за частки секунди, що свідчить про ефективність алгоритмів, використання LINQ і грамотну побудову моделей.

Особливу увагу в реалізації приділено гнучкості у роботі з даними. Структура фільтрів не є жорстко зашитою – навпаки, вона генерується динамічно на основі актуальних значень у файлі, що дозволяє системі адаптуватися до будь-яких змін у джерелі даних без потреби переписування коду. Так само гнучко реалізовано виведення координат, яке легко масштабувати до будь-якої картографічної бібліотеки. Це дозволяє швидко змінювати підхід до візуалізації залежно від цілей (інтерактивна карта, heatmap, маршрути тощо).

У ході роботи вдалося забезпечити стійкість до помилок та виняткових ситуацій. Система правильно реагує на некоректні запити, відсутні параметри або помилки у джерелі. Навіть якщо JSON-файли будуть пошкоджені або недоступні, обробка не спричинить збоїв – це реалізовано через винятки, логування та безпечне завантаження даних. Крім того, реалізовано механізм реактивного оновлення: при зміні локального JSON-файлу відповідний репозиторій одразу перезавантажує інформацію завдяки FileSystemWatcher, не потребуючи втручання адміністратора.

Слід відзначити й те, що створена система є готовою до масштабування. У майбутньому вона може бути інтегрована з фронтендом на базі React або Angular, отримати мобільну версію, розширення функцій (статистика, фільтри за регіонами, додаткові поля), а також бути перенесеною на повноцінну СУБД для кращого контролю над великими обсягами даних. Завдяки вже наявному API таке масштабування може здійснюватися поетапно, без критичних змін у базовій логіці.

Важливо підкреслити і соціальну значущість системи. Робота з даними про загиблих – це не лише технічне завдання, а й відповідальність. Розроблений інструмент надає зручний і швидкий доступ до інформації про воїнів, сприяє збереженню пам’яті, робить її доступною для громадян, родичів і представників громадськості. Цей проєкт доводить, що сучасні технології можуть бути використані для гідного вшанування героїв та ефективного обслуговування потреб громади.

Таким чином, проєкт реалізовано успішно, і він відповідає як функціональним, так і нефункціональним вимогам. Обрана архітектура, технології, алгоритми і структура API забезпечили основу для створення надійної, розширюваної і значущої інформаційної системи, готової до подальшого розвитку та інтеграції в інші цифрові сервіси.

Висновки до розділу

Проведена реалізація програмної частини інформаційної системи дозволила повністю підтвердити працездатність запропонованих моделей, алгоритмів та архітектурних рішень, розроблених у попередньому розділі. Побудована система функціонує стабільно, демонструє передбачувану поведінку під навантаженням і повністю відповідає як вимогам, що були поставлені на етапі формалізації задачі, так і технічному завданню.

Результати розробки підтверджують коректність вибору трирівневої архітектури, де контролери відповідають за маршрути, сервіси – за бізнес-логіку, а репозиторії – за доступ до джерел даних. Таке розділення дозволило легко організувати код, досягти прозорості у його підтримці та забезпечити зручність подальшого розширення. Всі реалізовані класи та методи мають чітке призначення, що дозволило уникнути дублювання коду і підвищити рівень повторного використання функціоналу.

Застосовані моделі (Soldier, RipSoldier, Cemetery, SoldierCoordinates) виявилися достатніми та гнучкими для обробки реальних даних з відкритих державних ресурсів. Десеріалізація JSON, структуризація полів і передача даних через DTO відбуваються без помилок, зберігаючи повноту й логічну цілісність інформації.

Функціональність, реалізована в межах API, охоплює широкий спектр можливостей – від простого виведення даних до фільтрації, пошуку, візуалізації координат, визначення героїв дня та автоматичного оновлення даних із зовнішнього джерела. Це робить систему повністю готовою до практичного

використання як у складі публічного порталу, так і в рамках закритого рішення для муніципального чи громадського користування.

Таким чином, завершення цього етапу розробки свідчить про повну реалізацію функціонального ядра системи та підтверджує правильність обраного підходу до її проєктування. Результати можуть бути використані як у навчальному, так і в прикладному контексті, а також слугувати основою для майбутнього розвитку проєкту.

ВИСНОВКИ

Досягнення мети та вирішення поставлених задач. Система, створена в рамках цієї роботи, охоплює повний спектр функціональних можливостей, необхідних для комплексної роботи з даними про загиблих воїнів. Одним із ключових елементів є реалізація фільтрації. Користувач має можливість гнучко відбирати записи за кількома критеріями одночасно: можна задати, наприклад, конкретне військове звання, підрозділ (корпус), перелік отриманих нагород, рік загибелі або ж обрати фільтрацію за статтю. Таке поєднання дозволяє значно звузити результати і швидко знайти потрібну інформацію навіть у великих обсягах даних.

Ще однією важливою складовою є реалізований пошук за частковим збігом імені. Система аналізує введений текст, розбиває його на складові та шукає всі можливі варіанти відповідності серед ПІБ захисників. При цьому впроваджено алгоритм вагового ранжування, завдяки чому результати виводяться не просто у випадковому порядку, а з урахуванням того, наскільки сильно ім’я збігається з пошуковим запитом – наприклад, пріоритет надається тим записам, де прізвище починається з відповідної літери.

Окрему увагу в системі приділено функції визначення так званих «героїв дня» – йдеться про воїнів, які загинули цього самого дня, але в різні роки. Для цього реалізовано механізм порівняння дати без урахування року, що дозволяє щодня оновлювати відповідний розділ пам’яті, акцентуючи увагу на важливості конкретної дати в контексті історії громади.

Ще однією технічно цікавою реалізацією є отримання географічних координат поховань. Система вміє працювати з координатами у форматі широта-довгота, які зберігаються в джерелах даних. Це створює передумови для подальшої інтеграції з картографічними сервісами – наприклад, для відображення місць поховань на інтерактивній мапі, що зробить інформацію ще більш наочною й доступною.

У контексті актуальності та динамічності даних особливо важливою є реалізована можливість оновлення інформації. По-перше, система підтримує ручне оновлення – за запитом адміністратора можна отримати найсвіжіші дані з державного порталу open data. По-друге, додатково реалізовано автоматичну реакцію на зміну локальних JSON-файлів: у разі їх оновлення, видалення чи створення, система миттєво перезавантажує дані без потреби в рестарті або повторному запуску сервісу.

Ще однією перевагою системи є механізм побудови списків фільтрів – тобто користувач не вводить значення вручну, а отримує готові варіанти для вибору на основі реальних, присутніх у даних значень. Це знижує ймовірність помилок і робить процес взаємодії з API зручнішим.

Насамкінець варто згадати й про опрацювання поховань: система дозволяє отримувати інформацію про місця поховання, включно з назвою кладовища, його ідентифікатором і координатами. Це забезпечує глибший зв’язок між персональними даними загиблих і просторовим розташуванням місця їхнього останнього спочинку.

Сукупність цих можливостей свідчить про те, що розроблена система не просто виконує базову функцію зберігання даних, а є потужним інструментом для їх аналітики, фільтрації, актуалізації та вшанування пам’яті.

Оцінка ефективності технічних рішень. Під час реалізації проєкту важливу роль відігравав правильний вибір технологій, на основі яких будується вся система. Вибір не був випадковим – він здійснювався з урахуванням поставлених задач, обмежень, потреб замовника, а також перспектив подальшого розвитку рішення. Кожне технічне рішення, яке було прийняте, мало своє обґрунтування й підтвердило свою доцільність у процесі розробки та тестування.

Основною технологією, обраною для реалізації серверної частини, став фреймворк ASP.NET Core Web API. Вона чудово підходить для створення RESTful API, які є основою майже кожної сучасної клієнт-серверної архітектури. Забезпечує високу швидкодію та низьке споживання ресурсів, що особливо важливо при роботі з великими наборами даних. Крім того, платформа зручна

для організації коду за принципами чистої архітектури – з розділенням відповідальності між контролерами, сервісами та репозиторіями.

Щодо формату зберігання даних, то з самого початку було прийнято рішення використовувати формат JSON. Цей вибір повністю відповідав побажанню замовника, а також ідеї прозорого, відкритого зберігання інформації. Також було реалізовано інтеграцію з CKAN API – відкритим програмним інтерфейсом державної платформи даних. Саме через нього система отримує свіжі записи про загиблих захисників та поховання.

У результаті використання саме цих трьох ключових технологій – ASP.NET Core, JSON, CKAN API – забезпечило стабільну, швидку й легко масштабовану основу для розробленої системи. Технології були не лише доречні в контексті проєкту, а й продемонстрували свою ефективність у реальній роботі: як під час запуску, так і під час оновлення даних, фільтрації, пошуку та обробки запитів.

Готовність до практичного використання та значення системи. Розроблена система пройшла усі етапи повного циклу створення – від постановки задачі та проєктування архітектури до безпосередньої реалізації та тестування. За своїм функціональним наповненням, технічною структурою та якістю виконання вона вже сьогодні може використовуватися у реальному середовищі. Всі ключові компоненти системи стабільно працюють, API відповідає на запити коректно, виняткові ситуації обробляються без збоїв, а інтеграція з відкритими джерелами даних функціонує згідно з очікуваннями.

Особливо важливо, що реалізація підтримує не лише основні запити (отримання списків, фільтрацію, пошук), а й включає можливість динамічного оновлення даних без зупинки сервісу – як вручну, так і автоматично. Це критично важливо у контексті роботи з реальними джерелами, які можуть змінюватися у часі. Таким чином, система показала свою життєздатність і адаптивність, що є ключовими факторами для впровадження у публічне або організаційне середовище.

З практичної точки зору, система вже використовується як окремий сайт, але також може використовуватись громадськими організаціями, меморіальними

ініціативами або просто як частина офіційного сайту ОТГ. Вона здатна автоматично підвантажувати та обробляти дані, надавати користувачам доступ до інформації у зручній формі, генерувати аналітику або інтегруватися з картографічними сервісами.

Однак практична значущість цієї розробки не обмежується лише технічним застосуванням. Її важливість полягає також у людському, етичному та соціальному аспектах. У часи, коли країна переживає важкі випробування, збереження пам'яті про тих, хто віддав своє життя, - це не просто питання цифровізації. Це – моральний обов'язок громади. І цифрові інструменти, реалізовані у межах цієї роботи, покликані не замінити живу пам'ять, а підсилити її, зробити доступною кожному, хто хоче знати, пам'ятати й вшанувати.

Таким чином, розроблена система – це не лише результат технічної роботи, а і приклад того, як інформаційні технології можуть служити високим суспільним цілям. Вона показала, що сучасні фреймворки, відкриті дані та структуроване мислення здатні дати реальний інструмент для збереження історії, пам'яті та гідності.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Microsoft. ASP.NET Core Documentation. URL: https://learn.microsoft.com/en-us/aspnet/core/

2. Microsoft. LINQ Documentation. URL: https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/

3. Newtonsoft.Json Documentation. URL: https://www.newtonsoft.com/json/help/html/Introduction.htm

4. CKAN API Documentation. URL: https://docs.ckan.org/en/latest/api/

5. Open Data Portal of Ukraine. URL: https://opendata.gov.ua/

6. Swagger OpenAPI Specification. URL: https://swagger.io/specification/

7. FileSystemWatcher Class Documentation. Microsoft Docs. URL: https://learn.microsoft.com/en-us/dotnet/api/system.io.filesystemwatcher

8. Freeman A., Sanderson A. Pro ASP.NET Core MVC 2. Apress, 2017. – 1017 p.

9. Martin R. Clean Architecture. Prentice Hall, 2017. – 432 p.

10. Rouse M. RESTful API Design. TechTarget. URL: https://www.techtarget.com/searchapparchitecture/definition/RESTful-API

11. Open Data Handbook. Open Knowledge Foundation. URL: https://opendatahandbook.org/

12. Vasyliev A. Programming in C# for Beginners. Kyiv: Profibooks, 2019. – 400 p.

13. Lok E. ASP.NET Core in Action. Manning Publications, 2021. – 600 p.

14. Katryk V. L., Hrabovskyi P. I. Software Engineering: Methodical Principles. Cherkasy: ChDTU, 2019. – 146 c.

15. Греков М. М. Архітектура програмного забезпечення. – К.: НАУ, 2020. – 168 с.

16. ISO/IEC 27001:2013 – Information security management systems.

17. Ritchy C. Implementing Repository and Unit of Work Patterns in ASP.NET Core. CodeMaze, 2021.

18. Mehlhorn D. Software Architecture for Developers. Leanpub, 2020.

19. Jurgen Appelo. Management 3.0: Leading Agile Developers. Addison-Wesley, 2011.

20. Robbins J., Atwood J. Debugging .NET Applications. Microsoft Press, 2018.

21. Sommers T. Web API Design: Crafting Interfaces that Developers Love. Apigee, 2014.

22. Pluralsight. ASP.NET Core Web API Fundamentals. URL: https://www.pluralsight.com

23. Microsoft Docs. Dependency Injection in .NET. https://learn.microsoft.com/en-us/dotnet/core/extensions/dependency-injection

24. Basham B. C#, The Complete Reference. McGraw-Hill, 2020.

25. Stahl R., Nagel C. Beginning Visual C# and .NET. Wrox, 2021.

26. GitHub. CKAN Open Source Data Management System. https://github.com/ckan/ckan

27. GitHub. Swashbuckle.AspNetCore – Swagger for .NET Core. https://github.com/domaindrivendev/Swashbuckle.AspNetCore

28. Google. JSON Guide. https://developers.google.com/maps/documentation/utilities/polylinealgorithm

29. Habr. Побудова багаторівневої архітектури в ASP.NET Core. URL: https://habr.com/ru/articles/

30. Ukrainian Open Data Portal Dataset Catalog. https://data.gov.ua/dataset

31. Бублик С. В. Архітектура інформаційних систем. Львів: ЛНУ, 2019. – 210 с.

32. Назаренко Т. О. Основи розробки REST API сервісів. Дніпро: ДНУ, 2021. – 190 с.

33. Гнатюк С. Програмна інженерія: підхід на основі моделей. К.: КНЕУ, 2022. – 300 с.

34. Simões A., Silva G. Data Collection from Public Portals. Springer, 2020.

35. Microsoft Learn. Introduction to Middleware. https://learn.microsoft.com/en-us/aspnet/core/fundamentals/middleware/

36. C# Corner. JSON Handling in ASP.NET Core Web API. https://www.c-sharpcorner.com

37. DotNetCurry. Advanced ASP.NET Core Features. https://www.dotnetcurry.com

38. Visual Studio Magazine. Modern Web Apps with ASP.NET Core. https://visualstudiomagazine.com

39. Baeldung. API Security Best Practices. https://www.baeldung.com/security-api-tips

40. Римар П.В., Журовський Я.О., Коновал М.С. Розробка веб-сайту про полеглих героїв Вінницької ОТГ. *Вісник Хмельницького національного університету. Технічні науки*. 2025.

41. Журовський Я.О., Римар П.В. Розробка серверної частини сайту про полеглих героїв на основі відкритих даних. *Прикладні інформаційні технології 2025*: Матеріали всеукр. науково-практ. конф. здобувачів вищ. освіти та молодих вчен., м. Вінниця, 22 трав. 2025р. 2025.