Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)
Кафедра *Системотехніки*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти *другий (магістерський)*
Розробка браузерного розширення для автоматичної кореляції доменів з базами вразливостей та оцінкою ризиків за допомогою LLM
(тема)

Виконав:
здобувач *2-го* року навчання,
групи *ІТПм-24-1*
Роман ТКАЧЕНКО
(власне ім'я, прізвище)

Спеціальність *122 – Комп'ютерні науки*
(код і повна назва спеціальності)
Тип програми *Освітньо-професійна*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Інформаційні технології проектування*
(повна назва освітньої програми)

Керівник *доцент Віктор РЕШЕТНІК*
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри системотехніки _____ *проф. Гребеннік І.В.*
(підпис) (власне ім'я, прізвище)

2025 р.

Я, як студент ХНУРЕ розумію і підтримую політику закладу з академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

16.12.2025  *Роман ТКАЧЕНКО*

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведено 19 грудня 2025 р.

Керівник кваліфікаційної роботи *доц. Віктор РЕШЕТНІК*

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*

Кафедра *Системотехніки*

Рівень вищої освіти *другий (магістерський)*

Спеціальність *122 – Комп'ютерні науки*
(код і повна назва)

Тип програми *Освітньо-професійна*
(освітньо-професійна або освітньо-наукова)

Освітня програма *Інформаційні технології проектування*
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві *Ткаченко Роману Адрійовичу*

(прізвище, ім'я, по батькові)

1. Тема роботи *Розробка браузерного розширення для автоматичної кореляції доменів з базами вразливостей та оцінкою ризиків за допомогою LLM*
затверджена наказом університету від ____ _____20__ р. № _____
2. Термін подання здобувачем роботи до екзаменаційної комісії ____ _____ 20__ р.
3. Вихідні дані до роботи *домен вебресурсу, URL вебсторінки, HTTP(S)-заголовки, специфікація API бази даних National Vulnerability Database (NVD, специфікація API великої мовної моделі (LLM)*
4. Перелік питань, що потрібно опрацювати в роботі *4.1 Вступ. 4.2 Аналіз предметної області. 4.2.1 Поняття уразливостей. 4.2.2 Формалізація поняття кореляції домену з базами вразливостей. 4.2.3 Джерела інформації про відомі уразливості. 4.2.4 Підходи до оцінювання рівня небезпеки уразливостей. 4.2.5 Засоби аналізу безпеки вебресурсів. 4.2.6 Аналіз проблеми та постановка задачі. 4.3 Вимоги до браузерного розширення. 4.4 Проєктування браузерного розширення. 4.4.1 Загальна архітектура. 4.4.2 Функціональна модель. 4.4.3 Функціональні вимоги. 4.4.4 Варіанти використання. 4.4.5 Нефункціональні вимоги. 4.4.6 Структура даних та логічна модель бази даних. 4.4.7 Проектування структури класів. 4.4.8 Послідовність процесу кореляції домену з базами даних вразливостей. 4.4.9 Проєктування алгоритму роботи браузерного розширення. 4.4.10 Проектування алгоритму кореляції доменів з базами даних вразливостей. 4.5 Реалізація браузерного розширення кореляції доменів з базами вразливостей. 4.5.1 Реалізація браузерного розширення. 4.5.2 Інтерфейс користувача браузерного розширення. 4.5.3 Реалізація серверної частини інформаційної системи. 4.5.4 Реалізація бази даних та фізична модель даних.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) *.5.1 Загальна архітектура інформаційної системи кореляції доменів з базами вразливостей. 5.2 Контекстна функціональна модель інформаційної системи кореляції доменів з базами вразливостей. 5.3 Декомпозиція основного процесу аналізу домену в інформаційній системі аналізу безпеки вебресурсів. 5.4 Декомпозиція процесу аналізу вебресурсу. 5.5 Декомпозиція процесу визначення технологій вебресурсу. 5.6 Декомпозиція процесу «Кореляція технологій з базами вразливостей». 5.7 Декомпозиція процесу формування оцінки ризику та рекомендацій. 5.8 Діаграма варіантів використання інформаційної системи кореляції доменів з базами вразливостей. 5.9 Макет інтерфейсу браузерного розширення. 5.10 Логічна модель даних інформаційної системи кореляції доменів з базами вразливостей. 5.11 Діаграма класів серверної частини інформаційної системи. 5.12 Діаграма послідовності процесу кореляції домену з базами даних вразливостей. 5.13 Діаграма діяльності браузерного розширення. 5.14 Діаграма діяльності алгоритму кореляції доменів з базами вразливостей.. 5.15 EER-діаграма бази даних. 5.16 Фрагмент програмного коду файлу background.ts (HTTP запит до серверної частин 5.17 Скріншоти інтерфейсу браузерного розширення. 5.18 Скріншот вебсторінки Vulnweb. 5.19 Графік порівняння еталонних профілів T(d) з результатами сервісу TechnologyDetectionService. 5.20 Порівняння виявлених вразливостей. 5.21 Порівняння еталонних інтегральних оцінок R*(d)

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання для кваліфікаційної роботи	24.11.2025	Виконано
2	Аналіз предметної області	25.11.2025	Виконано
3	Постановка вимог до браузерного розширення	28.11.2025	Виконано
4	Проектування браузерного розширення	03.12.2025	Виконано
5	Реалізація браузерного розширення	06.12.2025	Виконано
6	Оцінка якості браузерного розширення	08.12.2025	Виконано
7	Оформлення пояснювальної записки	10.12.2025	Виконано
8	Подання матеріалів до захисту	15.12.2025	Виконано
9	Захист кваліфікаційної роботи	23.12.2025	Виконано

Дата видачі завдання *24 листопада* 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ *доцент Віктор РЕШЕТНІК*
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Записка пояснювальна: 99 стор., 36 рис., 4 табл., 23 джерел, 2 додатка. Графічна частина дипломної роботи містить 27 плакатів.

БЕЗПЕКА ВЕБ-РЕСУРСІВ, ВРАЗЛИВОСТІ, CVE/CPE, РИЗИК-МОДЕЛЮВАННЯ, НАЦІОНАЛЬНА БАЗА ВРАЗЛИВОСТЕЙ NVD, LLM, БРАУЗЕРНІ РОЗШИРЕННЯ.

Об’єктом дослідження кваліфікаційної роботи є процес автоматизованого аналізу безпеки веб-ресурсів у режимі реального часу.

Предметом дослідження є методи кореляції доменів вебсайтів з публічними базами вразливостей, а також підходи до оцінки ризиків та формування рекомендацій за допомогою великих мовних моделей.

Метою роботи є розробка інформаційної системи, що дозволяє автоматично визначати домен вебсайту, встановлювати їх зв'язок із відомими вразливостями, оцінювати рівень ризику та формувати рекомендації для кінцевого користувача за допомогою LLM, інтегрованої у браузерне розширення.

Методами дослідження є аналіз технічної документації міжнародних стандартів (CVE, CPE, CVSS), методів сигнатурного й евристичного визначення технологій, принципів побудови браузерних розширень, підходів машинного аналізу тексту, моделювання процесів у UML/IDEF0, практична реалізація серверної частини на NestJS, створення локальної бази вразливостей та експериментальна оцінка ефективності роботи системи.

Результатами кваліфікаційної роботи є розроблене браузерне розширення та серверна платформа, здатні автоматично аналізувати домени, визначати технології, виконувати кореляцію з базами вразливостей, обчислювати рівень ризику та генерувати рекомендації за допомогою LLM. Проведено перевірку коректності кореляції, та ефективності оцінки ризику, що підтвердило практичну ефективність створеного рішення.

ABSTRACT

Explanatory note: 99 p., 36 pics., 4 tables, 23 sources, 2 applications. Graphic part of the thesis contains 27 posters.

WEB SECURITY, VULNERABILITY ANALYSIS, CVE/CPE, RISK ASSESSMENT, NVD, LLM, BROWSER EXTENSION.

The object of the study is the process of automated real-time security assessment of web resources.

The subject of the study is the methods of correlating detected web technologies with public vulnerability databases (CVE, CPE, NVD), as well as approaches to risk evaluation and recommendation generation using large language models.

The purpose of the thesis is to develop an information system capable of automatically identifying the technologies used by a website, correlating them with known vulnerabilities, calculating the overall risk level, and generating user-oriented recommendations via an LLM integrated into a browser extension.

The research methods include analysis of international standards (CVE, CPE, CVSS), examination of signature-based and heuristic technology detection approaches, study of browser extension architecture, machine learning–based text modeling, UML/IDEF0 functional modeling, practical implementation of the server logic using NestJS, development of a local vulnerability database, and experimental evaluation of the system's efficiency.

The results of the thesis include a fully implemented browser extension and backend system that automatically analyzes domains, identifies technologies, correlates them with known vulnerabilities, performs risk assessment, and generates recommendations using an LLM. Performance evaluation demonstrated high accuracy of correlation, and high usability of generated recommendations.The system can be applied in cybersecurity platforms

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

NVD – National Vulnerability Database (Національна база вразливостей, США).

CVE – Common Vulnerabilities and Exposures (загальні вразливості та експозиції).

CPE – Common Platform Enumeration (уніфікована система опису ПЗ й обладнання).

CVSS – Common Vulnerability Scoring System (система рейтингування вразливостей).

ORM – Object-Relational Mapping (технологія відображення бази даних в об’єкти).

SSL/TLS – Secure Sockets Layer / Transport Layer Security (криптографічні протоколи захисту даних).

JS-бандл – збірка JavaScript-коду, сформована інструментами типу Webpack.

NestJS – фреймворк Node.js для створення серверних застосунків.

ВСТУП

Розвиток вебтехнологій призвів до того, що значна частина повсякденної діяльності людини пов’язана з використанням вебресурсів від доступу до інформаційних сервісів і спілкування до проведення фінансових операцій. На тлі цього зростає кількість інцидентів, пов’язаних із порушенням конфіденційності, цілісності та доступності даних, які обробляються під час взаємодії користувача з вебресурсами [1, 2]. У таких умовах аналіз відомих уразливостей, що стосуються конкретних вебресурсів, стає важливою складовою оцінювання ризиків для користувача веббраузера.

Відомості про уразливості програмного забезпечення публікуються у відкритих спеціалізованих каталогах і базах даних, які підтримуються державними установами та галузевими організаціями. Перелік Common Vulnerabilities and Exposures (CVE) є публічним реєстром ідентифікаторів уразливостей, який адмініструється корпорацією MITRE за підтримки урядових агентств США та використовується як єдиний механізм ідентифікації уразливостей у різних програмних продуктах [3].

Національна база вразливостей США National Vulnerability Database (NVD) є державним інформаційним ресурсом, що підтримується Національним інститутом стандартів і технологій США (NIST). Вона використовує ідентифікатори CVE та доповнює їх структурованими даними, необхідними для автоматизованого аналізу уразливостей і ризиків [4].

Кількісне оцінювання небезпеки уразливостей у базі NVD здійснюється за методикою Common Vulnerability Scoring System (CVSS), яка розробляється та підтримується галузевим консорціумом FIRST і є відкритим стандартом загального користування. Методика CVSS визначає формалізований набір метрик і шкал для інтерпретації впливу уразливостей на інформаційні системи [5].

Використання каталогів CVE, бази NVD та методики CVSS створює основу для побудови систем аналізу безпеки, однак застосування цих даних до конкретних вебресурсів потребує додаткових етапів кореляції між доменами,

технологіями та записами про уразливості.

Окремим напрямом досліджень є використання великих мовних моделей (Large Language Models, LLM) для підтримки завдань у сфері кібербезпеки. У наукових роботах розглядаються приклади застосування таких моделей для аналізу журналів подій, автоматизованої обробки текстових звітів про уразливості, узагальнення технічної інформації та формування текстових рекомендацій щодо реагування на інциденти [6–8]. Водночас питання інтеграції мовних моделей у інформаційні системи, які працюють у тісному зв’язку з веббраузером і покликані надавати користувачеві вебресурсу інформацію про потенційні загрози під час перегляду сайту, досі висвітлено обмежено.

Одним із підходів до зменшення навантаження на користувача веббраузера та підвищення рівня автоматизації аналізу ризиків є створення інформаційної системи кореляції доменів з базами вразливостей, клієнтська частина якої реалізована у вигляді браузерного розширення. Така система може під час відвідування вебресурсу автоматично одержувати інформацію про домен, визначати використані технології, корелювати їх із записами у базах уразливостей, оцінювати ризики для користувача вебресурсу з урахуванням характеристик цих уразливостей та формувати текстові рекомендації щодо подальшої поведінки під час взаємодії з вебресурсом. Серверна частина інформаційної системи у цьому випадку забезпечує інтеграцію з зовнішніми базами уразливостей, зберігання та актуалізацію даних, а також використання великої мовної моделі для узагальнення результатів аналізу.

Актуальність теми дипломної роботи зумовлена відсутністю зручних програмних засобів, які дозволяли б користувачу веббраузера без спеціальної підготовки оперативно оцінювати ризики взаємодії з конкретним вебресурсом на основі публічних даних про уразливості. Незважаючи на наявність сформованих каталогів уразливостей і методик їх кількісного оцінювання, зокрема CVE, бази NVD та моделі CVSS, ці дані описують уразливості програмних продуктів загалом і не мають прямої прив’язки до доменів, а також не інтегровані у засоби, що працюють безпосередньо під час перегляду вебсторінок. Унаслідок цього між

накопиченими відомостями про уразливості та практичними потребами користувача вебресурсу існує розрив, подолання якого потребує розроблення інформаційної системи, що автоматично корелює домени з базами уразливостей та формує зрозумілу оцінку ризику.

Об’єктом дослідження у дипломній роботі є процес аналізу відомих уразливостей, пов’язаних із вебресурсами, з точки зору їхнього впливу на безпеку користувача під час взаємодії з цими ресурсами у веббраузері.

Предметом дослідження є методи та програмні засоби автоматизованої кореляції доменів з базами уразливостей, а також методи оцінювання рівня ризику для користувача вебресурсу з використанням великої мовної моделі.

Метою роботи є розроблення інформаційної системи кореляції доменів з базами вразливостей, клієнтська частина якої реалізована у вигляді браузерного розширення та забезпечує автоматичну кореляцію даних про домен із записами у базах уразливостей і формування оцінки ризику для користувача вебресурсу із залученням великої мовної моделі.

Для досягнення поставленої мети необхідно послідовно розв’язати низку завдань. Потрібно проаналізувати предметну область, джерела даних про уразливості та наявні програмні засоби аналізу безпеки вебресурсів, сформулювати вимоги до інформаційної системи кореляції доменів з базами вразливостей та її основних модулів з орієнтацією на потреби користувача веббраузера, розробити архітектуру інформаційної системи, включно з клієнтською частиною у вигляді браузерного розширення, серверною частиною та підсистемою взаємодії із зовнішніми сервісами, побудувати функціональну модель і модель потоків даних інформаційної системи, а також логічну модель даних, реалізувати основні модулі інформаційної системи, серед яких модуль визначення технологій вебресурсу, модуль кореляції з базами уразливостей, механізм актуалізації даних і модуль оцінювання ризику для користувача із застосуванням мовної моделі, а також виконати тестування реалізованих компонентів, дослідити роботу інформаційної системи на прикладах реальних вебресурсів та проаналізувати отримані результати.

Методи дослідження включають аналіз і узагальнення наукових публікацій у галузі кібербезпеки та застосування мовних моделей [1–8], застосування методів функціонального моделювання IDEF0, моделювання потоків даних DFD, об’єктно-орієнтованого моделювання UML, а також загальнонаукових методів аналізу, синтезу та порівняння.

Практичне значення роботи полягає у розробленні інформаційної системи кореляції доменів з базами вразливостей, яка призначена для надання користувачеві веббраузера інформації про можливі загрози під час відвідування вебресурсу на основі даних про відомі уразливості та їхній потенційний вплив. Такий підхід може бути використаний як основа для подальшого розвитку засобів інтерактивного аналізу безпеки у веббраузерах та для інтеграції з існуючими підходами до управління уразливостями в організаціях.

Структурно дипломна робота складається зі вступу, п’яти розділів, висновків, списку використаних джерел та додатків.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття уразливостей

Розвиток інформаційних технологій та зростання залежності суспільства від мережевих сервісів призводять до того, що уразливості інформаційних систем безпосередньо впливають на безпеку користувачів вебресурсів. У сучасних оглядах стану кіберзагроз зазначається, що експлуатація уразливостей залишається одним із основних способів отримання початкового доступу до інформаційних систем поряд із фішингом та іншими методами соціальної інженерії. Окремі звіти вказують, що частка інцидентів, пов’язаних із використанням технічних уразливостей, становить суттєву частину зафіксованих випадків вторгнень [1, 2, 9]. У контексті вебресурсів це означає, що помилки реалізації вебзастосунків, неправильно налаштовані служби або застаріле програмне забезпечення можуть створювати умови, за яких дії зловмисника безпосередньо впливають на безпеку користувача, який взаємодіє з вебсайтом через веббраузер.

У нормативних документах з кібербезпеки уразливість визначається як слабке місце в інформаційній системі, її процедурах безпеки, внутрішніх контролях або реалізації, яке може бути використане чи ініційоване джерелом загрози. Таке розуміння подано, зокрема, у Національного інституту стандартів і технологій США (National Institute of Standards and Technology, NIST), де уразливість описується як стан або властивість інформаційної системи чи її компонентів, що дає змогу події або діям порушити встановлену політику безпеки або призвести до інциденту [10, 11]. Окремо виділяються програмні уразливості, які пов’язані з дефектами програмного коду та можуть бути використані для порушення конфіденційності, цілісності чи доступності інформації, що обробляється системою [11].

Поняття уразливості охоплює не лише помилки у програмному коді, але й недоліки конфігурації, прорахунки в проєктуванні архітектури, некоректне використання протоколів, слабкі параметри криптографічних механізмів або

недостатні організаційні заходи. У контексті вебресурсів до уразливостей можуть належати помилки реалізації механізмів автентифікації, некоректне опрацювання вхідних даних на стороні сервера, відсутність належних перевірок під час роботи зі сторонніми скриптами, неправильне налаштування параметрів передачі файлів чи заголовків протоколу, а також використання застарілих версій бібліотек або серверного програмного забезпечення. Усі такі слабкі місця створюють додаткові можливості для реалізації загроз, які проявляються безпосередньо на рівні взаємодії користувача з вебресурсом.

Для повного опису ситуації, пов’язаної з безпекою, одного лише поняття уразливості недостатньо. У міжнародних стандартах і методиках управління інформаційними ризиками, зокрема в рекомендаціях ISO/IEC 27005, розглядається зв’язок між активами, загрозами, уразливостями та наслідками реалізації інцидентів [12, 13]. Загроза в цьому контексті розуміється як потенційна подія або дія, яка може призвести до небажаних наслідків для активів організації або для користувачів, що взаємодіють із інформаційною системою. Для вебресурсу загрозами можуть бути, наприклад, спроби виконання шкідливого коду в контексті веббраузера користувача, несанкціоноване отримання файлів, перехоплення автентифікаційних даних або зміна змісту вебсторінки.

Поняття ризику у сфері кібербезпеки пов’язане з імовірністю реалізації загрози за наявності уразливості та з можливими наслідками такої реалізації. У документі NIST SP 800-30 ризик описується як функція імовірності того, що певна загроза скористається конкретною уразливістю, і величини впливу, який матиме відповідна подія [14]. Аналогічний підхід використовується в стандартах сімейства ISO/IEC 27000, де інформаційний ризик розглядається як комбінація наслідків події, пов’язаної з безпекою, та імовірності її настання [12, 13]. Таким чином, навіть за наявності уразливості фактичний рівень ризику залежить від сукупності факторів, серед яких імовірність експлуатації, доступність цілі, наявні заходи захисту та специфіка сценарію взаємодії користувача з вебресурсом.

Для користувача веббраузера, який відвідує вебресурс, ключовим є не стільки факт існування певної уразливості в інфраструктурі або програмному

забезпеченні, скільки вплив експлуатації цієї уразливості на конфіденційність його облікових даних, доступ до персональної інформації чи можливість нав’язування небажаних дій у межах сеансу взаємодії з вебсайтом. Наприклад, уразливості, що дозволяють виконувати скрипти від імені вебресурсу у веббраузері, можуть призводити до викрадення файлових маркерів автентифікації або маніпулювання вмістом сторінки. Помилки налаштування шифрування або механізмів оброблення сесійних ідентифікаторів потенційно впливають на конфіденційність даних, що передаються між браузером та сервером. Таким чином, аналіз ризиків у сфері кібербезпеки для вебресурсів неминуче включає оцінювання наслідків для користувача веббраузера як учасника взаємодії з відповідною інформаційною системою.

Формалізація понять уразливості та ризику, започаткована в роботах зі стандартизації управління інформаційною безпекою, створює підґрунтя для побудови автоматизованих засобів аналізу, які використовують структуровані дані про уразливості та опис сценаріїв загроз. У поєднанні з відомостями про конфігурацію вебресурсу та особливості технологій, що застосовуються, така формалізація дає змогу оцінювати ризики з точки зору конкретного користувача веббраузера. Це, у свою чергу, створює передумови для розроблення інформаційної системи кореляції доменів з базами вразливостей, здатної надавати користувачеві узагальнену інформацію про можливі загрози під час відвідування вебсайту. Саме на таких підходах ґрунтується подальший аналіз предметної області та постановка задачі, представлені у наступних підрозділах.

1.2 Формалізація поняття кореляції домену з базами вразливостей

У статистиці термін «кореляція» традиційно описує ступінь залежності між випадковими величинами і використовується для кількісної оцінки того, наскільки зміни одного показника пов’язані зі змінами іншого. У класичних визначеннях кореляція розглядається як статистичний зв’язок між числовими рядами, що вимірюється відповідними коефіцієнтами і застосовується для виявлення лінійних

або більш загальних залежностей між змінними.

У сфері кібербезпеки цей термін часто вживається у значенні як встановлення відповідностей між різнорідними наборами даних. Наприклад пов'язування логів подій, конфігураційних параметрів, ідентифікаторів програмних продуктів та записів про відомі уразливості в єдиному інформаційному просторі. Такий підхід близький до сутнісно-зв'язних моделей, де для кожної сутності предметної області задаються відношення з іншими сутностями, що дозволяє описувати структуру об'єктів і залежності між ними.

У цій роботі під кореляцією домену з базами вразливостей розуміється формалізований процес встановлення відповідностей між конкретним вебресурсом і множиною записів про уразливості. Для опису цього процесу вводяться множина доменів D, множина вебтехнологій T та множина уразливостей V.

Процес виявлення технологій домену формалізується у вигляді відображення:

$$f: D \to 2^T, \tag{1.1}$$

де D – множина доменів;

T – множина вебтехнологій;

2^T – множина всіх підмножин множини T.

На основі виявлених технологій встановлюється відповідність між технологіями та уразливостями, яка описується відображенням:

$$g: T \to 2^V, \tag{1.2}$$

де V – множина уразливостей;

T – множина вебтехнологій;

2^V – множина всіх можливих підмножин множини V.

У результаті множина уразливостей, релевантних конкретному домену, визначається як:

$$V(d) = \bigcup_{t \in T(d)} g(t), \tag{1.3}$$

де T(d) – множина технологій, виявлених для домену d;

g(T) – множина уразливостей для з технології t.

Таким чином, кореляція домену з базами вразливостей у межах запропонованої моделі формалізується як послідовне відображення виду d→T(d)→V(d), яке дозволяє однозначно пов’язати вебресурс із відповідними записами про уразливості.

1.3 Джерела інформації про відомі уразливості

Аналіз ризиків для користувача вебресурсу неможливо виконати без доступу до достовірних та актуальних відомостей про відомі уразливості програмного забезпечення, яке бере участь в обробленні запитів і формуванні вмісту вебсторінок. Для систематизації такої інформації створено низку спеціалізованих джерел, що відрізняються моделлю даних, форматом представлення, механізмами актуалізації та призначенням.

Ключовим елементом інфраструктури опису уразливостей є перелік Common Vulnerabilities and Exposures (CVE). Цей перелік, який підтримується організацією MITRE, містить записи, кожен з яких відповідає окремому випадку уразливості або групі тісно пов’язаних вразливих станів у програмному забезпеченні [3]. Для кожного запису у переліку CVE призначається унікальний ідентифікатор, надається короткий текстовий опис та посилання на додаткові джерела інформації. Перелік CVE виконує роль узагальненого індексу, на який посилаються виробники програмного забезпечення, дослідники безпеки, розробники інструментів сканування та інші учасники екосистеми кібербезпеки. Разом з тим у переліку CVE не зберігаються розширені структуровані атрибути, такі як числові оцінки ризику або формалізовані параметри впливу, тому для автоматизованого аналізу цього недостатньо.

Для підтримки автоматизованих процесів управління уразливостями на основі записів переліку CVE створено Національну базу вразливостей National Vulnerability Database (NVD), яку підтримує Національний інститут стандартів і технологій США. У базі NVD для кожного запису, пов’язаного з певним ідентифікатором у переліку CVE, формується розширений структурований опис

[4]. До такого опису інтегруються оцінки за методикою Common Vulnerability Scoring System (CVSS) [5], відомості про програмні платформи та продукти, яких стосується уразливість, додаткові атрибути для фільтрації та класифікації, а також посилання на джерела первинної інформації. Дані у базі NVD публікуються у машиночитаних форматах, що дає змогу отримувати їх за допомогою програмних інтерфейсів, виконувати пошук і фільтрацію за різними критеріями та інтегрувати у інформаційні системи оцінювання ризиків.

Структура окремого запису у базі NVD містить кілька логічних блоків. До них належать ідентифікаційні відомості, до яких відносяться ідентифікатор у переліку CVE, версія запису, дата публікації та останнього оновлення, текстові описи уразливості, оцінки за методикою CVSS для різних версій специфікації, перелік пов’язаних програмних платформ і продуктів, які описуються за схемою Common Platform Enumeration (CPE), а також додаткові посилання на ресурси виробників та дослідників. У записах вразливостей (рис. 1.1) в базі NVD наводяться ідентифікатори у переліку CVE, текстові описи уразливості, оцінки за методикою CVSS та перелік пов’язаних платформ CPE. Такий формат є показовим для розуміння того, які саме дані надалі обробляє інформаційна система аналізу безпеки вебресурсів під час кореляції технологій конкретного вебресурсу з відомими уразливостями.

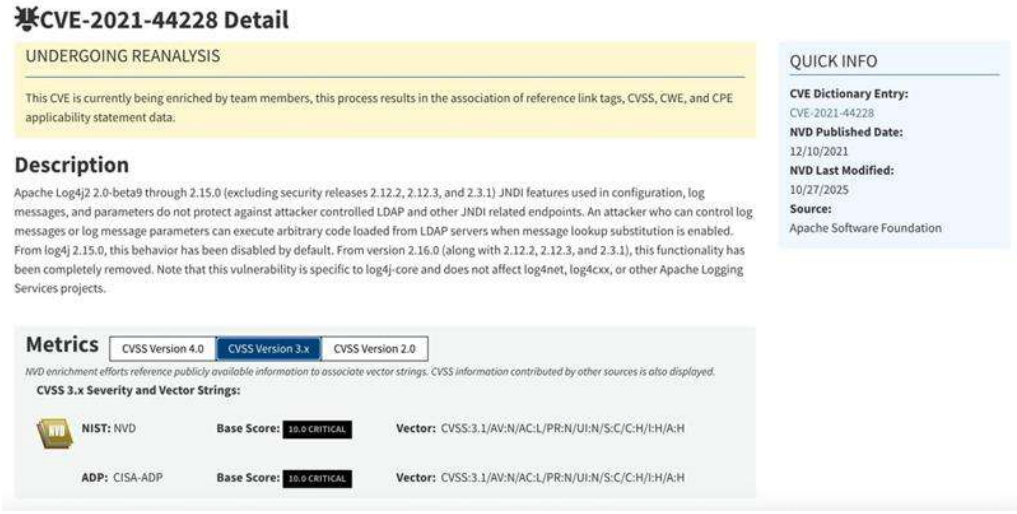

Рисунок 1.1 – Приклад фрагмента с запису про уразливість у базі NVD

Важливим компонентом моделі даних бази NVD є каталоги програмних платформ і продуктів, які описуються за допомогою схеми Common Platform Enumeration. У цих каталогах кожній комбінації виробника, продукту та версії відповідає окремий запис, що дозволяє пов’язувати конкретні версії програмного забезпечення з відповідними уразливостями. Для аналізу вебресурсів це має практичне значення, оскільки на основі визначених технологій можна ідентифікувати пов’язані з ними записи у каталозі CPE та, відповідно, у базі NVD і сформувати перелік відомих уразливостей, які можуть впливати на безпеку користувача під час взаємодії з вебресурсом [4, 5].

Окрім переліку CVE та бази NVD, суттєву роль у практиці кібербезпеки відіграють офіційні бюлетені та оповіщення виробників програмного забезпечення. Виробники вебсерверів, операційних систем, фреймворків для вебзастосунків і бібліотек публікують описи уразливостей, що стосуються їхніх продуктів, а також інструкції щодо оновлення, змін конфігурації або тимчасових заходів реагування. Такі матеріали, як правило, містять посилання на відповідні ідентифікатори у переліку CVE, а іноді включають технічні подробиці, які не завжди повністю відображені у загальнодоступних реєстрах. Ці джерела корисні для поглибленого аналізу конкретних продуктів, однак їх використання в інформаційних системах, орієнтованих на автоматизоване оцінювання ризиків, потребує додаткової нормалізації та узгодження із записами у переліку CVE та базі NVD.

У середовищі відкритого програмного забезпечення сформувалися каталоги уразливостей, орієнтовані на програмні бібліотеки та залежності. Для окремих мов програмування та екосистем збирання програмного забезпечення існують реєстри відомих уразливостей у відкритих компонентах, які використовуються засобами автоматичного аудиту залежностей. Такі реєстри дають змогу оцінювати ризики, пов’язані з використанням певних версій бібліотек під час розроблення вебзастосунків, і можуть доповнювати дані переліку CVE та бази NVD, зокрема на етапі виявлення проблем у нових або спеціалізованих компонентах.

Значний обсяг відомостей про уразливості та способи їх експлуатації

накопичується на спеціалізованих платформах, орієнтованих на дослідників безпеки. Частина таких платформ підтримує бази експлойтів із прив'язкою до ідентифікаторів у переліку CVE, а також містить опис сценаріїв атак і демонстраційні приклади. З погляду побудови інформаційної системи, призначеної для надання користувачеві веббраузера узагальнених відомостей про загрози, ці ресурси мають допоміжний характер, оскільки переважна частина інформації на них орієнтована на фахівців і потребує додаткової інтерпретації.

Для завдань, пов'язаних із оцінюванням ризиків для користувача вебресурсу, важливими є не лише зміст окремих каталогів уразливостей, але й властивості самих джерел. Актуальність даних залежить від частоти оновлення записів у базі NVD, затримки між реєстрацією уразливості у переліку CVE та появою докладного запису, а також своєчасності коригування оцінок за методикою CVSS у разі отримання нової інформації про вектори атак і масштаб наслідків. Повнота відомостей визначається тим, наскільки повно конкретне джерело охоплює технології та платформи, які використовуються на вебресурсах, що підлягають аналізу. Доступність для автоматизованої обробки пов'язана з наявністю формалізованих форматів подання даних і програмних інтерфейсів, а також з параметрами надійності таких інтерфейсів.

Інформаційна система аналізу безпеки вебресурсів, орієнтована на надання користувачеві веббраузера відомостей про можливі загрози, повинна спиратися на такі джерела, які поєднують формалізоване подання уразливостей із можливістю регулярного отримання оновлень у машиночитаному форматі. У цьому контексті база NVD у поєднанні з переліком CVE та каталогами платформ і продуктів, що описуються за схемою CPE, розглядається як базовий елемент інформаційного забезпечення під час проєктування і реалізації механізмів автоматизованої кореляції та оцінювання ризиків для користувача вебресурсу [3–5].

1.4 Підходи до оцінювання рівня небезпеки уразливостей

Оцінювання рівня небезпеки уразливостей є складовою процесів управління

ризиками в інформаційній безпеці, оскільки результати такого оцінювання використовуються для аналізу можливого впливу уразливостей на конфіденційність, цілісність і доступність інформації. Під час оцінювання враховуються властивості самої уразливості, зокрема вектор атаки, складність її експлуатації, потреба в автентифікованому доступі або підвищених привілеях, а також характер потенційного впливу на інформаційні ресурси. Окреме значення має контекст використання вебресурсу, у межах якого користувач веббраузера взаємодіє з серверною частиною та прикладними компонентами системи.

У практиці аналізу уразливостей широкого застосування набула методика Common Vulnerability Scoring System (CVSS), яку розробляє та підтримує організація Forum of Incident Response and Security Teams. Методика CVSS визначає формалізований набір базових метрик, що описують характеристики уразливості, які не залежать від конкретного середовища експлуатації, та дозволяє обчислити числовий показник у діапазоні від 0 до 10. До таких характеристик належать параметри, пов’язані з вектором атаки, складністю експлуатації, необхідними привілеями та впливом на конфіденційність, цілісність і доступність даних [5].

У базі National Vulnerability Database для більшості уразливостей, ідентифікованих у переліку Common Vulnerabilities and Exposures, публікуються значення базових метрик і підсумкові бали за методикою CVSS. Це дозволяє використовувати ці показники без виконання додаткових обчислень і застосовувати їх як уніфіковану числову характеристику потенційного впливу уразливості на інформаційні системи, пов’язані з вебресурсом. Для користувача веббраузера такі оцінки слугують індикатором можливих наслідків експлуатації уразливості, однак самі по собі не описують контекст взаємодії з конкретним сайтом.

Великі мовні моделі можуть застосовуватися для інтерпретації формалізованих оцінок небезпеки уразливостей і представлення результатів у формі текстових пояснень і рекомендацій, зрозумілих для користувача веббраузера [6–8]. У межах інформаційної системи кореляції доменів з базами

вразливостей такі моделі отримують числові оцінки за методикою CVSS разом із відомостями про технології вебресурсу та сценарії взаємодії з ним і на цій основі формують узагальнену оцінку ризику та рекомендації щодо обережної поведінки.

1.5 Засоби аналізу безпеки вебресурсів

Засоби аналізу безпеки вебресурсів охоплюють набір програмних рішень, які застосовуються для виявлення уразливостей, аналізу конфігурації та перевірки параметрів захисту вебзастосунків і пов’язаних з ними інфраструктурних компонентів. До цієї групи належать інструменти динамічного тестування вебзастосунків, онлайн-сервіси аналізу конфігурації та програмні компоненти, що працюють у взаємодії з веббраузером.

Серед інструментів динамічного аналізу вебзастосунків характерним представником є засіб OWASP Zed Attack Proxy (ZAP). У документації інструменту ZAP зазначено, що він працює у режимі проксі-сервера між браузером і вебресурсом, перехоплює HTTP(S)-трафік, виконує пасивний та активний аналіз запитів і відповідей і містить модулі для виявлення поширених уразливостей вебзастосунків [14]. До цього самого класу належить програмний комплекс Burp Suite. У документації програмного комплексу Burp Suite описано компоненти Burp Proxy для перехоплення та модифікації запитів, компонент Burp Repeater для ручного повторення запитів і Burp Scanner як компонент динамічного тестування (Dynamic Application Security Testing, DAST), що виконує автоматизований пошук уразливостей на основі моделювання дій тестувальника [15]. Такі інструменти дають змогу будувати карту вебресурсу, аналізувати механізми оброблення параметрів запитів, досліджувати реалізацію автентифікації й керування сесіями.

До засобів, які доступні без встановлення окремого програмного забезпечення, належать онлайн-сервіси аналізу конфігурації вебресурсів. Прикладом такого підходу є онлайн-сервіс аналізу заголовків безпеки SecurityHeaders.com. Цей сервіс виконує запит до зазначеного домену, реєструє

заголовки HTTP-відповідей і на основі наперед визначених правил формує зведений звіт щодо використання заголовків, пов’язаних із політикою безпеки, зокрема заголовків Content-Security-Policy, HTTP Strict-Transport-Security, X-Frame-Options та інших [16]. Для аналізу параметрів протоколів шифрування Transport Layer Security (TLS) і Secure Sockets Layer (SSL) використовується онлайн-сервіс SSL Server Test у межах проєкту Qualys SSL Labs. У документації сервісу SSL Server Test зазначено, що він перевіряє сертифікати, підтримувані версії протоколів TLS/SSL, набори шифрів і додаткові опції та подає результати у вигляді інтегральної оцінки й деталізованого технічного звіту [17]. В інтерфейсі сервісу SSL Server Test (рис. 1.2) у верхній частині розміщено поле для введення доменного імені, а нижче відображаються підсумкова оцінка та таблиці з параметрами сертифіката, протоколів і наборів шифрів. Така структура інтерфейсу дає змогу послідовно переглядати результати аналізу окремих аспектів настроювання TLS/SSL-з’єднання.

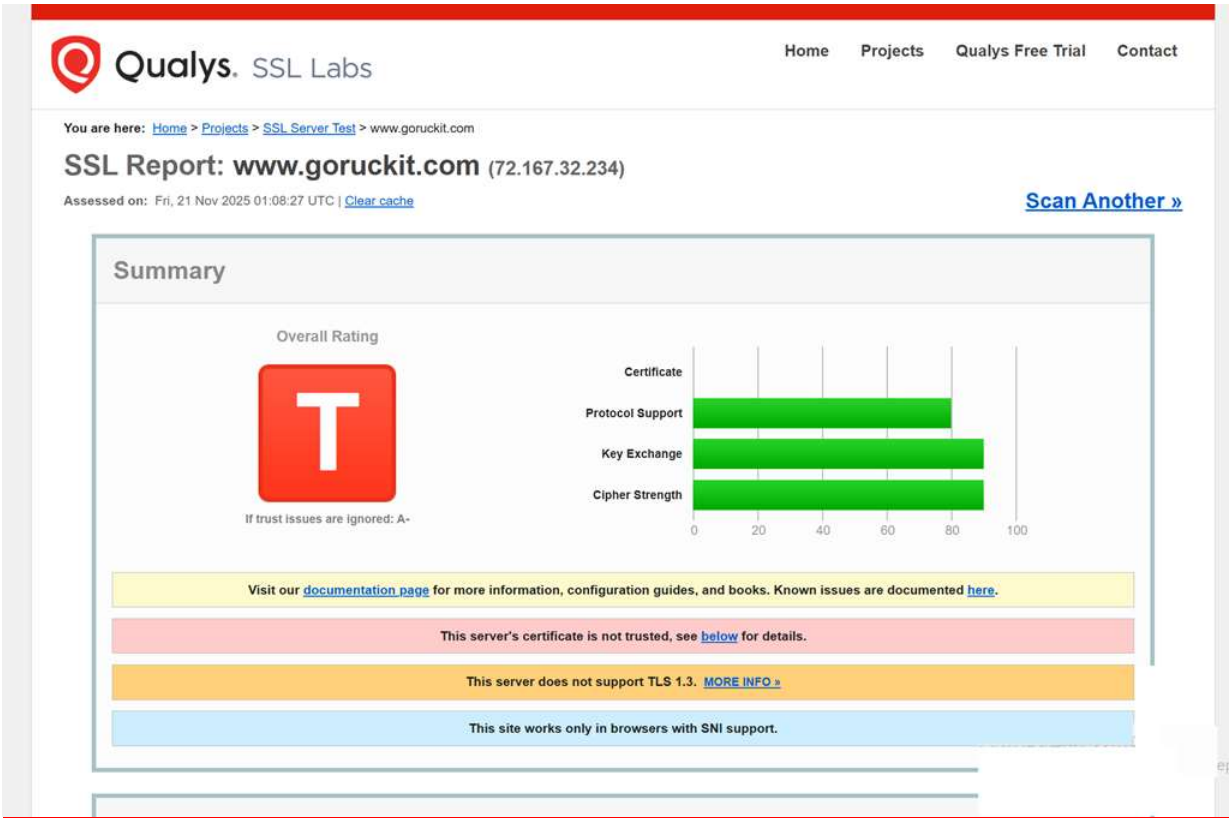

Рисунок 1.2 – Приклад інтерфейсу сервісу SSL Server Test для аналізу конфігурації TLS/SSL вебресурсу

Окрему групу становлять програмні компоненти, що працюють з боку користувача веббраузера у вигляді розширень. До таких компонентів належать, зокрема, розширення браузера NoScript Security Suite, яке за замовчуванням блокує виконання активного вмісту (скриптів, окремих типів мультимедійного контенту та інших потенційно небезпечних елементів) і дозволяє користувачеві формувати перелік сайтів, для яких виконання такого вмісту дозволено. На офіційному сайті розширення NoScript зазначено, що воно реалізує модель «блокування за замовчуванням» і надає додаткові механізми протидії експлойтам у браузері.

Іншим прикладом є розширення фільтрації вмісту uBlock Origin, використовується для блокування рекламних елементів, трекерів і доменів, пов’язаних із розповсюдженням шкідливого програмного забезпечення, на основі наборів фільтрів і списків відомих джерел небажаного трафіку.

Для примусового використання захищеного протоколу шифрування під час підключення до вебресурсів застосовувалися розширення на кшталт HTTPS Everywhere, розробленого компаніями Electronic Frontier Foundation та The Tor Project, а також сучасні аналоги, наприклад розширення Smart HTTPS, які автоматично намагаються встановити з’єднання через HTTPS, якщо сервер підтримує цей режим.

Такі програмні компоненти змінюють спосіб взаємодії користувача з вебресурсами (зокрема через блокування скриптів, небажаного контенту або примусове використання шифрування), проте, як правило, не інтегруються з публічними базами уразливостей і не виконують повної кореляції між використаними технологіями вебресурсу та записами у переліку Common Vulnerabilities and Exposures (CVE) чи базі National Vulnerability Database (NVD).

У джерелах, присвячених засобам аналізу безпеки, також описано інтеграцію інструмента OWASP Zed Attack Proxy (ZAP) у конвеєри безперервної інтеграції та автоматизовані процеси тестування вебзастосунків [14], а для програмного комплексу Burp Suite наведено приклади використання під час аудиту, коли фахівець аналізує відповіді вебзастосунку на модифіковані запити та

досліджує реалізацію критичних функцій [15]. Такі сценарії застосування орієнтовані на етапи розроблення й тестування вебзастосунків і передбачають участь фахівців, які мають доступ до конфігурації та вихідного коду.

Наведені приклади показують, що для аналізу безпеки вебресурсів існує набір інструментів, орієнтованих на різні етапи життєвого циклу вебзастосунку та різні ролі користувачів. Інструменти динамічного тестування та онлайн-сервіси надають детальні технічні звіти, придатні для роботи інженерів і фахівців з безпеки. Розширення браузера впливають на окремі аспекти взаємодії з вебресурсом, зокрема на завантаження скриптів і застосування шифрування. Порівняння засобів представлено в табличці 1.1.

Таблиця 1.1 – Порівняння існуючих засобів для аналізу безпеки вебресурсів

Критерій порівняння	Інструменти динамічного тестування	Онлайн-сервіси аналізу конфігурації	Розширення веббраузера
Рівень інтеграції	Працюють як окремі програмні комплекси або проксі-сервери	Працюють як зовнішні вебсервіси	Інтегруються безпосередньо у веббраузер
Контекст використання	Тестування та аудит вебзастосунків	Разовий аналіз конфігурації домену	Повсякденний перегляд вебресурсів
Необхідність попереднього налаштування	Потребують налаштування та спеціалізованих знань	Не потребують встановлення або складного налаштування	Потребують встановлення розширення та початкової конфігурації
Тип аналізу	Активний і пасивний аналіз HTTP(S)-трафіку	Аналіз відповідей сервера за наперед визначеними правилами	Блокування або модифікація окремих елементів взаємодії
Аналіз використовуваних технологій вебресурсу	Можливий у межах ручного або напівавтоматичного аналізу	Не виконується	Обмежений або відсутній
Інтеграція з переліком CVE та базою NVD	Можлива через додаткові модулі або ручний аналіз	Відсутня	Відсутня
Форма подання результатів	Детальні технічні звіти та журнали	Зведені звіти або інтегральні оцінки	Налаштування правил блокування та індикатори стану
Орієнтація на користувача веббраузера	Не орієнтовані	Частково орієнтовані	Орієнтовані
Автоматичне оновлення результатів під час навігації	Відсутнє	Відсутнє	Частково присутнє
Представлення узагальненої оцінки ризику	Не стандартизоване	Обмежене окремими аспектами	Відсутнє

Наведене порівняння в таблиці 1.1 відображає, що жоден із розглянутих підходів не забезпечує комплексного подання інформації про ризики вебресурсу безпосередньо в контексті його перегляду користувачем веббраузера з урахуванням використаних технологій та актуальних даних з публічних баз

уразливостей. Це зумовлює доцільність розроблення підходу, який поєднує інтеграцію у веббраузер, автоматизований аналіз технологій вебресурсу та кореляцію з записами у переліку CVE і базі NVD.

1.6 Аналіз проблеми та постановка задачі

Аналітичні звіти з кібербезпеки свідчать про зростання кількості інцидентів, пов’язаних з експлуатацією уразливостей у вебінфраструктурі, зокрема у сценаріях, де взаємодія з вебресурсами здійснюється через веббраузер [1–2]. У таких сценаріях користувач веббраузера безпосередньо взаємодіє з вебресурсами, вводить облікові дані, персональну інформацію та виконує фінансові операції, що підвищує значущість своєчасного інформування про потенційні ризики.

Паралельно активно розвивається інфраструктура публічних джерел даних про уразливості, до якої належать перелік Common Vulnerabilities and Exposures (CVE), база National Vulnerability Database (NVD), методика Common Vulnerability Scoring System (CVSS) та інші формалізовані підходи до опису уразливостей [3–5; 10–12]. Ці джерела забезпечують накопичення структурованих відомостей про уразливості програмних продуктів, їх потенційні наслідки та характеристики, що дозволяє використовувати їх у процесах аналізу та управління ризиками інформаційної безпеки.

Наявність детально структурованих даних про уразливості не гарантує їх практичної придатності для користувача веббраузера під час звичайного перегляду вебресурсів. Інформація у публічних базах уразливостей подається у технічній формі та орієнтована переважно на фахівців з кібербезпеки, адміністраторів і розробників вебзастосунків. Без спеціальної підготовки користувач веббраузера не може безпосередньо оцінити, як наявні уразливості пов’язані з конкретним вебресурсом і які ризики вони створюють у контексті його дій.

Розглянуті у попередніх підрозділах засоби аналізу безпеки вебресурсів автоматизують окремі аспекти виявлення уразливостей і аналізу конфігурації,

однак їх застосування, як правило, орієнтоване на професійні сценарії використання. Інструменти динамічного тестування та онлайн-сервіси аналізу конфігурації формують технічні звіти, що не інтегруються безпосередньо у процес повсякденного перегляду вебсторінок і не надають узагальненого пояснення ризиків з урахуванням дій користувача веббраузера.

Таким чином, виникає проблема розриву між наявністю формалізованих відомостей про уразливості у публічних базах даних та можливістю їх використання для інформування користувача веббраузера про потенційні ризики під час взаємодії з конкретними вебресурсами. Цей розрив зумовлює необхідність пошуку підходів, які дозволяють представити наявну інформацію про уразливості у формі, придатній для використання під час звичайного перегляду вебресурсів.

З огляду на викладене у роботі ставиться задача створення засобу, призначеного для інформування користувача веббраузера про потенційні ризики, пов’язані з відомими уразливостями вебресурсів, у процесі їх відвідування. Метою розв’язання цієї задачі є зменшення розриву між наявністю структурованих даних про уразливості у публічних джерелах та можливістю їх практичного використання користувачем веббраузера без спеціальної підготовки у сфері кібербезпеки.

Результатом розв’язання поставленої задачі має бути програмний засіб, який забезпечує подання узагальненої інформації про можливі ризики для користувача веббраузера під час взаємодії з вебресурсами. Під час формулювання задачі передбачається, що такий засіб не виконує активного втручання у роботу вебресурсів і не замінює інструменти тестування безпеки, призначені для використання фахівцями, а обмежується збиранням, узагальненням і поданням відомостей про відомі уразливості у доступній формі.

2 ВИМОГИ ДО БРАУЗЕРНОГО РОЗШИРЕННЯ

Браузерне розширення аналізу безпеки вебресурсів має бути призначене для інформування користувача веббраузера про потенційні ризики під час відвідування вебсайтів у режимі звичайного перегляду. Розширення повинно інтегруватися у робочий процес веббраузера таким чином, щоб користувач отримував узагальнену інформацію про ризики без необхідності відкривати окремі вебсервіси або виконувати додаткові ручні дії. Робота розширення має бути орієнтована на аналіз доступних відомостей про вебресурс і представлення результатів у вигляді індикатора та текстового пояснення, придатного для сприйняття без спеціальної підготовки у сфері кібербезпеки.

Розширення повинно автоматично визначати доменне ім’я поточного вебресурсу під час завантаження вебсторінки, переходу між сторінками та оновлення вкладки. Для кожного визначеного домену розширення має ініціювати отримання результатів аналізу безпеки, забезпечуючи роботу у фоновому режимі та не блокуючи взаємодію користувача з вебсторінкою. Якщо вебресурс відкривається у кількох вкладках, розширення повинно коректно обробляти паралельні події та уникати дублювання запитів у межах короткого проміжку часу, коли це не впливає на актуальність результату.

Розширення повинно забезпечувати збирання мінімально необхідних ознак, які характеризують вебресурс у контексті оцінювання ризиків. До таких ознак мають належати ідентифікатор вебресурсу на рівні доменного імені, схема доступу та інші не персональні параметри, що можуть бути отримані з мережевої взаємодії браузера з вебресурсом і використовуються для формування узагальнених рекомендацій. Отримання або збереження конфіденційних даних користувача, зокрема вмісту форм, паролів, повідомлень, платіжних реквізитів, файлів або історії перегляду, у межах функціональності розширення не передбачається. Розширення не повинно здійснювати модифікацію вмісту вебсторінок, ін’єкцію скриптів у контекст сторінки або втручання у мережеві з’єднання з метою активного тестування безпеки.

Результати аналізу, що отримуються розширенням, повинні включати узагальнену оцінку рівня ризику для поточного вебресурсу та пояснювальний текст, який відображає можливі загрози й їх потенційний вплив у контексті типових дій користувача веббраузера, зокрема автентифікації, введення персональних даних, використання облікового запису та виконання фінансових операцій. Подання інформації має бути компактним, однозначним і таким, що допускає швидке ознайомлення, при цьому текстове пояснення повинно узгоджуватися з виведеним індикатором ризику. Розширення повинно надавати можливість перегляду додаткових технічних відомостей у межах інтерфейсу розширення у разі потреби, але такі відомості не мають бути основним способом взаємодії для користувача веббраузера.

Інтерфейс розширення повинен відображати стан аналізу для поточного вебресурсу, включно зі станами отримання даних, наявності результатів, відсутності даних або помилки. У випадку неможливості отримання результатів розширення має інформувати користувача веббраузера про причину у загальній формі, не розкриваючи внутрішніх технічних деталей, які не впливають на прийняття рішення користувачем. Візуальні елементи розширення не повинні перекривати основний контент сторінки, порушувати доступність елементів інтерфейсу вебсайту або спричиняти небажані зміни верстки. Розширення повинно підтримувати роботу з різними розмірами вікна браузера та коректно відображати інформацію у стандартних умовах масштабування інтерфейсу.

З метою забезпечення актуальності інформації про уразливості розширення має використовувати механізм обмеження строку придатності результатів аналізу для конкретних доменів та ініціювати повторне отримання даних після завершення заданого інтервалу часу. Повинна підтримуватися логіка використання збережених результатів у межах строку придатності з одночасним забезпеченням оновлення даних після його завершення. У разі тимчасової недоступності джерел даних або мережевих збоїв розширення повинно зберігати коректну поведінку та, за можливості, використовувати останні доступні результати з явним позначенням їх актуальності. Розширення має уникати

надмірної кількості запитів до зовнішніх джерел, що можуть призводити до перевищення лімітів або погіршення продуктивності, при цьому механізми кешування та повторних спроб повинні застосовуватися таким чином, щоб не створювати відчутного навантаження на веббраузер і мережеве з’єднання.

Розширення повинно відповідати вимогам безпеки та конфіденційності, що передбачає передавання лише тих даних, які необхідні для ідентифікації вебресурсу та формування результатів аналізу. Усі операції з даними мають виконуватися із урахуванням принципу мінімізації, а також із обмеженням обсягу інформації, що зберігається локально. Розширення повинно забезпечувати відсутність зберігання секретів у відкритому вигляді та коректне використання механізмів безпеки, наданих платформою веббраузера. За наявності налаштувань користувача розширення повинно забезпечувати можливість керування обсягом локально збережених результатів та поведінкою автоматичного оновлення в межах, які не суперечать цільовому призначенню розширення.

Вимоги до сумісності передбачають підтримку сучасних веббраузерів на основі рушія Chromium та використання стандартних механізмів розширень, передбачених платформою. Розширення повинно коректно працювати за стандартних політик безпеки браузера, включно з обмеженнями на доступ до мережевих ресурсів і доступ до контексту сторінки. Розширення повинно бути здатним працювати з вебресурсами, що використовують протокол HTTPS, а у випадку HTTP має забезпечувати коректне відображення результатів з урахуванням того, що відсутність шифрування є окремим фактором ризику.

Розширення повинно забезпечувати відтворюваність результатів з точки зору користувацького інтерфейсу, тобто за однакових умов відвідування вебресурсу та однакової актуальності даних індикатор і пояснювальний текст мають відображатися узгоджено. За необхідності фіксації подій для налагодження або тестування розширення може формувати технічні журнали у межах середовища розробника, однак такі журнали не повинні містити конфіденційних даних користувача та не повинні бути доступні користувачеві веббраузера у звичайному режимі роботи.

3 ПРОЄКТУВАННЯ БРАУЗЕРНОГО РОЗШИРЕННЯ

3.1 Загальна архітектура

Розв’язання задачі кореляції доменів з базами вразливостей, не може бути реалізоване виключно засобами браузерного розширення. Браузерне розширення функціонує в обмеженому середовищі веббраузера та не має можливості виконувати обчислювально складні операції, пов’язані з кореляцією програмних продуктів із записами у публічних базах уразливостей, обробленням значних обсягів даних, регулярним оновленням відомостей з бази National Vulnerability Database, а також інтеграцією з сервісами великих мовних моделей.

Прямий доступ браузерного розширення до публічних баз уразливостей і сервісів мовної моделі обмежується політиками безпеки веббраузера, вимогами до захисту ключів доступу та необхідністю контролю частоти звернень до зовнішніх ресурсів. Реалізація зазначених функцій у клієнтському компоненті призвела б до ускладнення розширення, зниження продуктивності та підвищення ризиків, пов’язаних із безпекою та конфіденційністю.

У зв’язку з цим розроблення браузерного розширення доцільно розглядати як частину більш загальної інформаційної системи, у якій обчислювально складні та ресурсомісткі операції виконуються у серверній частині, а у середовищі веббраузера реалізується лише клієнтський компонент, призначений для ініціювання аналізу та подання результатів користувачеві. Такий підхід забезпечує розподіл функцій між компонентами системи відповідно до їх призначення та обмежень середовища виконання.

Архітектура інформаційної системи кореляції доменів з базами вразливостей побудована за багатокомпонентним принципом і включає клієнтський компонент у середовищі веббраузера, серверний вебзастосунок та зовнішні сервіси, що надають дані про уразливості й забезпечують формування текстових пояснень. Такий підхід дозволяє розділити функції збору вихідних параметрів, аналізу безпеки та оброблення зовнішніх даних між окремими логічними рівнями системи.

Клієнтський компонент реалізовано у вигляді браузерного розширення, яке функціонує в середовищі веббраузера та забезпечує отримання доменного імені активної вебсторінки. Розширення формує запит до серверної частини системи та передає параметри, необхідні для виконання аналізу. Безпосереднє звернення до публічних баз уразливостей і виконання обчислювальних процедур у клієнтському компоненті не передбачене.

Серверна частина реалізується як окремий вебзастосунок і виконує основні функції аналізу. Вона забезпечує приймання запитів від браузерного розширення, визначення характеристик вебресурсу, кореляцію доменів із записами у публічних базах уразливостей та формування узагальненої оцінки ризику. У межах серверної частини також здійснюється збереження результатів аналізу у внутрішній базі даних, що містить відомості про домени, пов’язані з ними записи про уразливості та часові параметри їх актуальності.

Для отримання даних про відомі уразливості серверний вебзастосунок взаємодіє з публічними джерелами інформації. Основним джерелом є перелік Common Vulnerabilities and Exposures, деталізовані описи та кількісні характеристики уразливостей отримуються з бази National Vulnerability Database та пов’язаних із нею сервісів, у яких застосовуються ідентифікатори Common Platform Enumeration та оцінки за методикою Common Vulnerability Scoring System. Взаємодія з цими джерелами винесена в окремий компонент серверної частини, що забезпечує уніфіковане формування запитів і перетворення отриманих відповідей у внутрішні структури даних.

Окремим елементом архітектури є інтеграція з сервісом мовної моделі великого розміру. Серверна частина формує структурований опис результатів аналізу домену та передає його через програмний інтерфейс до зовнішнього сервісу мовної моделі. У відповідь отримується текстове пояснення, яке узагальнює наявні ризики та доповнює числову оцінку результатами інтерпретації.

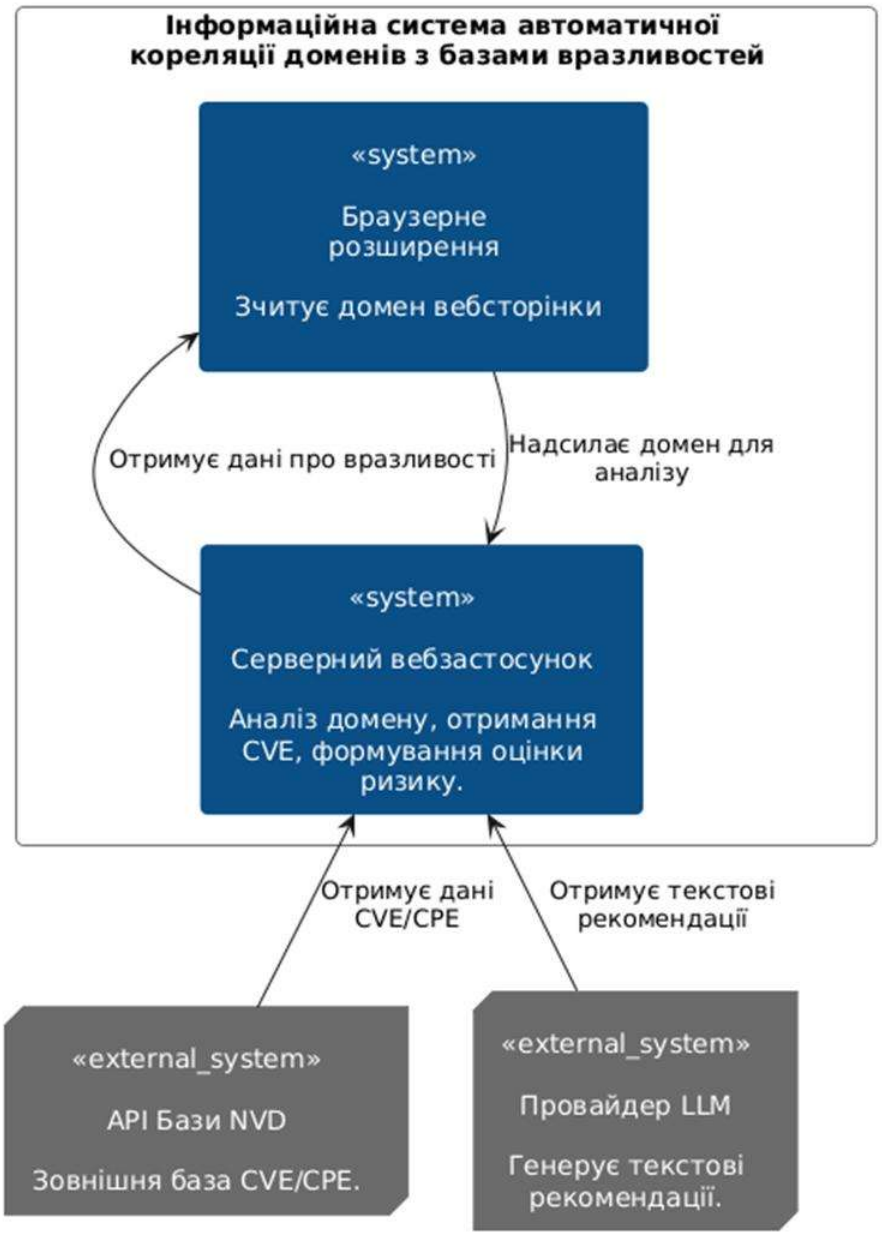

Рисунок 3.1 – Загальна архітектура інформаційної системи кореляції доменів з базами вразливостей

На рисунку 3.1 наведено загальну архітектуру інформаційної системи кореляції доменів з базами вразливостей, де показано взаємодію браузерного розширення, серверного вебзастосунку, зовнішніх баз уразливостей та провайдеру великої мовної моделі, а також основні напрями обміну даними між цими компонентами.

3.2 Функціональна модель

Функціональна модель інформаційної системи кореляції доменів з базами вразливостей описує, як вхідні дані про домен перетворюються на результати, що відображаються у браузерному розширенні. У такому поданні система розглядається не як набір програмних модулів, а як єдиний процес оброблення інформації, який спирається на зовнішні стандарти, публічні бази даних та сервіси мовних моделей великого розміру. Нотація IDEF0 дає змогу формально зафіксувати цей процес через входи, виходи, керувальні впливи та механізми виконання.

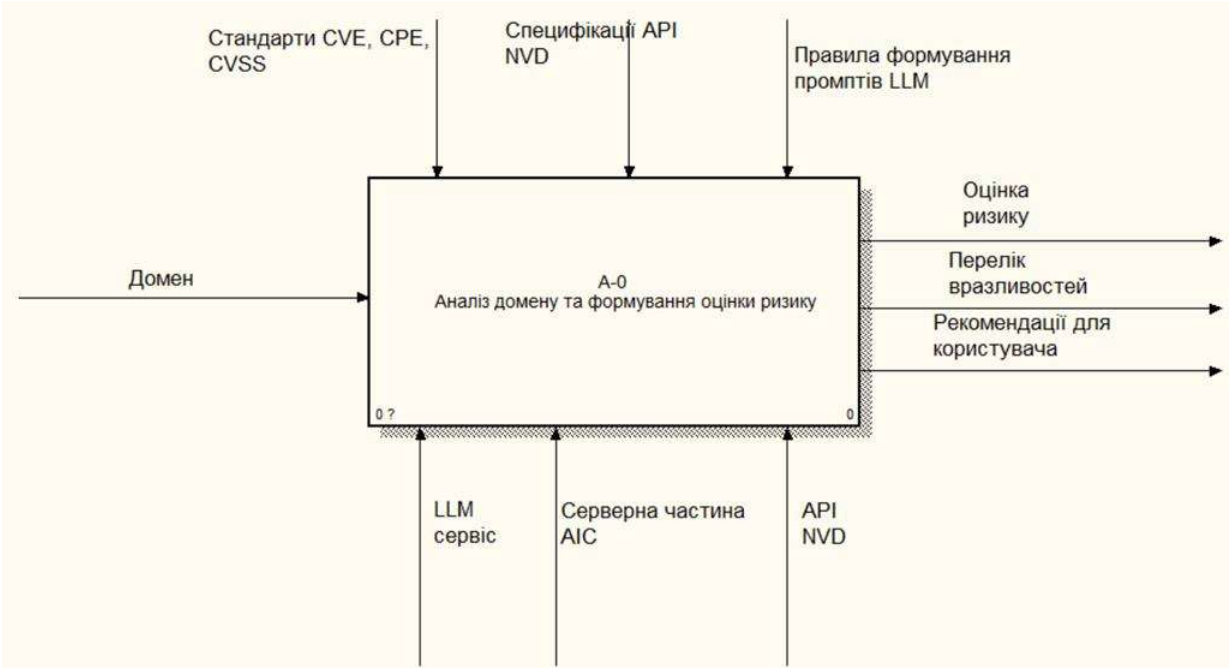

Рисунок 3.2 – Контекстна функціональна модель інформаційної системи кореляції доменів з базами вразливостей

На контекстному рівні (рис.3.2) основний процес позначається як «Аналіз домену та формування оцінки ризику». У цьому процесі поєднуються отримання відомостей про домен від браузерного розширення, пошук пов’язаних записів у базі National Vulnerability Database (NVD) на основі переліку Common Vulnerabilities and Exposures (CVE) та каталогу Common Platform Enumeration (CPE), застосування методики Common Vulnerability Scoring System (CVSS) і

формування текстових пояснень із використанням мовної моделі великого розміру (Large Language Model, LLM). Контекстна діаграма фіксує лише зовнішні зв’язки цього процесу, без розкриття внутрішньої структури модулів.

У моделі ліворуч до блоку A-0 надходить вхідний потік «Домен». Цей потік утілює відомості про вебресурс, що аналізується, зокрема доменне ім’я та допоміжні параметри, які браузерне розширення може передати до серверної частини інформаційної системи. Праворуч із блоку виходять три результати роботи системи. Потік «Оцінка ризику» містить числовий або категоріальний показник, придатний для відображення у вигляді індикатора у браузерному розширенні. Потік «Перелік вразливостей» містить структурований список записів, пов’язаних із доменом, для яких зазначено ідентифікатори з переліку CVE та основні характеристики з бази NVD. Потік «Рекомендації для користувача» містить текстовий опис потенційних загроз для користувача веббраузера та загальні поради щодо обережної поведінки під час роботи з відповідним вебресурсом.

Зверху на блок A-0 діють керувальні впливи. Стандарти переліку CVE, каталогу CPE та методики CVSS визначають структуру опису уразливостей і правила інтерпретації їхніх кількісних показників. Специфікації прикладного програмного інтерфейсу (API) бази NVD задають формат і обмеження запитів до цієї бази та впливають на те, як саме серверна частина інформаційної системи отримує й оновлює дані про уразливості. Правила формування промптів для мовної моделі великого розміру встановлюють структуру й зміст контексту, який передається до LLM-сервісу, а також вимоги до формату відповіді, що використовується для побудови текстових рекомендацій.

Знизу до блока A-0 підведено механізми, за допомогою яких реалізується процес аналізу. Серверна частина інформаційної системи виконує оброблення запитів від браузерного розширення, взаємодію з прикладним програмним інтерфейсом бази NVD, збереження результатів у внутрішній базі даних і розрахунок оцінки ризику. API бази NVD виступає зовнішнім механізмом доступу до записів про уразливості, пов’язаних із певними програмними платформами та

продуктами. LLM-сервіс реалізує можливість побудови текстових пояснень на основі структурованого опису ситуації, який формується серверною частиною інформаційної системи.

Контекстна функціональна модель задає межі інформаційної системи та фіксує зв’язки між вхідними даними, керувальними впливами, механізмами виконання процесу та результатами, які повертаються користувачеві веббраузера. Для детальнішого опису поведінки системи основний процес аналізу домену далі розкладається на окремі етапи, кожен з яких виконує власне перетворення даних у межах загального ланцюга оброблення.

На наступному рівні функціональна модель деталізується через декомпозицію процесу A-0 на окремі етапи, які послідовно перетворюють дані про домен на перелік уразливостей, оцінку ризику та текстові рекомендації. Такий поділ дає змогу пов’язати логічні кроки аналізу з відповідними модулями інформаційної системи та показати, у яких місцях використовується взаємодія з базою National Vulnerability Database та сервісом великою мовною моделлю.

На рисунку 3.3 подано декомпозицію процесу «Аналіз домену та формування оцінки ризику» на рівні A-0-1 – A-0-4.

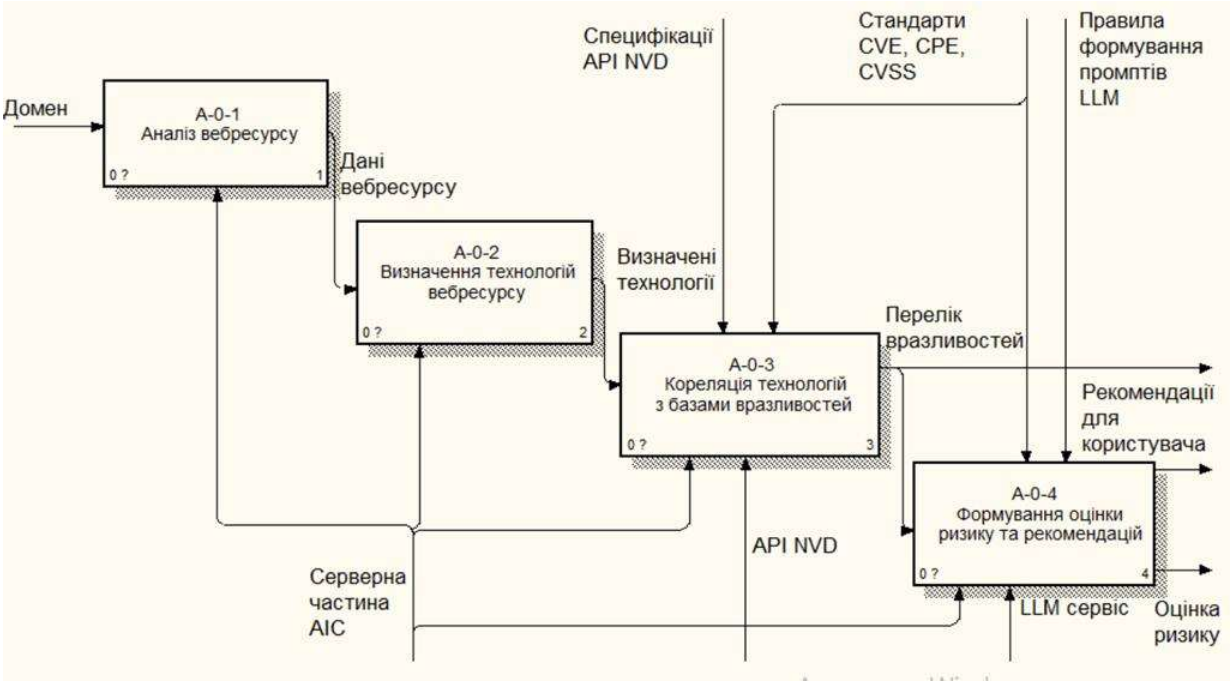

Рисунок 3.3 – Декомпозиція основного процесу аналізу домену

У блоці A-0-1 «Аналіз вебресурсу» система отримує вхідний потік «Домен» від браузерного розширення та перетворює його на внутрішній опис «Дані вебресурсу». До такого опису можуть належати доменне ім’я, схема доступу та інші службові відомості, потрібні для подальшого визначення технологій. Виконання цього етапу спирається на серверну частину інформаційної системи, яка на діаграмі позначена як механізм «Серверна частина ІС».

У блоці A-0-2 «Визначення технологій вебресурсу» використовуються сформовані «Дані вебресурсу». На цьому етапі з доступних ознак формується перелік «Визначені технології», у якому узагальнюються відомості про ймовірні програмні платформи та продукти, що застосовуються на вебресурсі. Цей перелік надалі використовується для звернення до публічних баз уразливостей. Як і на попередньому етапі, механізмом виконання виступає серверна частина інформаційної системи.

У блоці A-0-3 «Кореляція технологій з базами вразливостей» перелік визначених технологій зіставляється з записами каталогу Common Platform Enumeration, після чого на основі одержаних CPE-ідентифікаторів формуються запити до бази National Vulnerability Database. На цей етап впливають специфікації прикладного програмного інтерфейсу бази NVD та стандартизовані підходи до опису уразливостей і метрик, позначені як «Стандарти CVE, CPE, CVSS». Взаємодія з базою NVD здійснюється через механізм «API NVD», а логіка оброблення результатів реалізується серверною частиною інформаційної системи. Результатом цього блока є потік «Перелік вразливостей», який частково передається назовні як один із виходів системи та одночасно виступає вхідними даними для наступного етапу.

У блоці A-0-4 «Формування оцінки ризику та рекомендацій» використовується «Перелік вразливостей», а також керувальні впливи у вигляді стандартів CVE, CPE, CVSS і правил формування промптів для мовної моделі великого розміру. На цьому етапі перелік уразливостей перетворюється на числову або категоріальну «Оцінку ризику» для відповідного домену та структурований опис ситуації, який далі застосовується для побудови текстових

рекомендацій. Взаємодія з мовною моделлю великого розміру реалізується через механізм «LLM сервіс», а серверна частина інформаційної системи відповідає за формування контексту запиту та оброблення відповіді. Виходами цього блока є потік «Оцінка ризику» для відображення індикатора в браузерному розширенні та потік «Рекомендації для користувача», у якому подано опис загроз і поради щодо обережної поведінки під час роботи з вебресурсом.

У межах функціональної моделі доцільно окремо розкрити етап, у якому відбувається перше перетворення вхідних даних про домен на внутрішнє подання, придатне для подальшого визначення технологій вебресурсу. Для цього основний процес аналізу вебресурсу розкладається на низку підпроцесів, які виконуються послідовно на серверній частині інформаційної системи. Такий поділ дає змогу показати, як система реєструє новий домен, перевіряє наявні результати в базі даних і, за потреби, формує оновлені дані для подальшого аналізу.

На рисунку 3.4 наведено декомпозицію процесу A-0-1 «Аналіз вебресурсу», у якій виділено три підпроцеси A-0-1-1, A-0-1-2 та A-0-1-3.

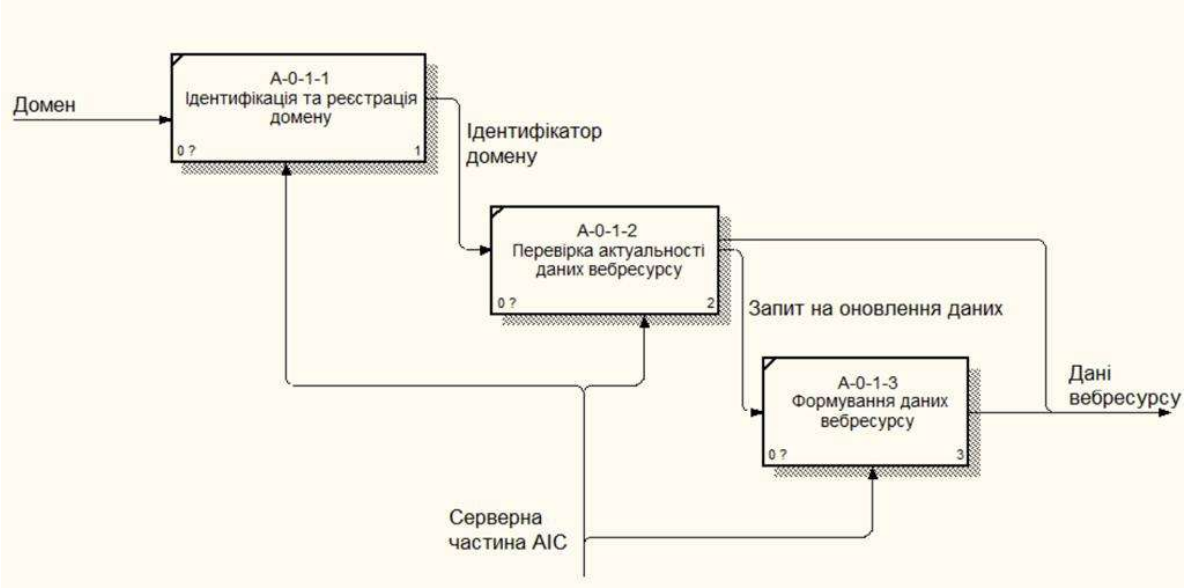

Рисунок 3.4 – Декомпозиція процесу аналізу вебресурсу

У блоці A-0-1-1 «Ідентифікація та реєстрація домену» обробляється вхідний потік «Домен», що надходить від браузерного розширення. Серверна частина інформаційної системи на цьому етапі виконує нормалізацію поданих даних,

зіставляє їх із наявними записами у внутрішній базі та за потреби створює новий запис. Результатом підпроцесу є внутрішній «Ідентифікатор домену», який однозначно пов’язує поточний запит із відповідним записом у базі даних і передається далі за схемою.

У блоці А-0-1-2 «Перевірка актуальності даних вебресурсу» використовується ідентифікатор домену для отримання з внутрішньої бази даних попередніх результатів аналізу. Серверна частина інформаційної системи порівнює часові мітки останнього оновлення з налаштованим інтервалом дії результатів. Якщо збережені дані ще можна використовувати, система не формує запит на новий аналіз, і далі застосовуються вже наявні відомості. Якщо ж термін дії результатів сплив або для домену немає жодних записів, формується службовий потік «Запит на оновлення даних», який ініціює наступний підпроцес.

У блоці А-0-1-3 «Формування даних вебресурсу» опрацьовується запит на оновлення, що надійшов із попереднього етапу. Серверна частина інформаційної системи виконує необхідні дії для одержання актуальних характеристик вебресурсу, зокрема може звертатися до самого вебресурсу, аналізувати відповіді вебсервера та фіксувати ознаки, які надалі використовуються для визначення технологій. Результатом роботи цього підпроцесу є потік «Дані вебресурсу», у якому узагальнюються всі зібрані відомості й який виступає виходом процесу А-0-1 та входом для подальшого етапу визначення технологій.

Усі три підпроцеси виконуються з використанням єдиного механізму, позначеного на діаграмі як «Серверна частина АІС». Така декомпозиція показує, що ще до взаємодії з публічними базами уразливостей система виконує окремий цикл роботи з доменом, у якому розв’язуються задачі реєстрації, перевірки строків дії попередніх результатів і формування актуальних даних вебресурсу. Наступний етап функціональної моделі пов’язаний із використанням цих даних для визначення технологій, які застосовуються на вебресурсі.

Подальше уточнення функціональної моделі стосується етапу, на якому на основі зібраних характеристик вебресурсу формується перелік технологій, що застосовуються на цьому ресурсі. Для цього логіку процесу визначення технологій

поділено на кілька послідовних дій, які виконуються на серверній частині інформаційної системи. Такий поділ дає змогу окремо показати момент виділення ознак, зіставлення їх із довідниковими даними та побудову узгодженого переліку технологій, придатного для подальшої кореляції з базами уразливостей.

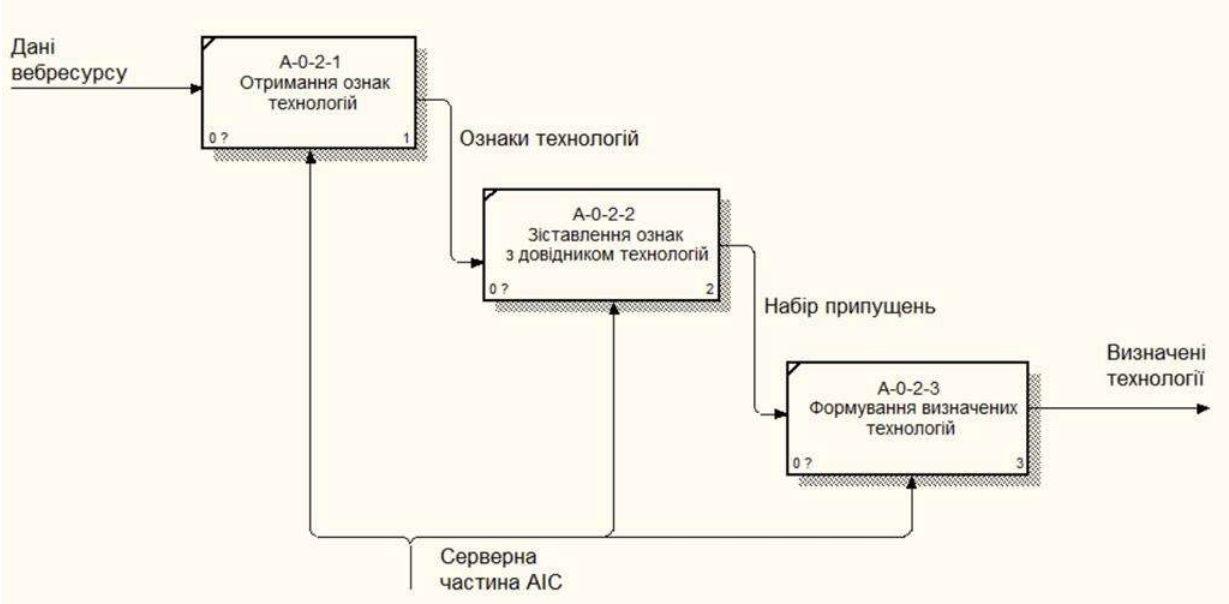

Рисунок 3.5 – Декомпозиція процесу визначення технологій вебресурсу

У блоці A-0-2-1 «Отримання ознак технологій» як вхід використовується потік «Дані вебресурсу», сформований на попередньому етапі аналізу. Серверна частина інформаційної системи витягує з цих даних елементи, які можуть слугувати індикаторами використання певних технологій. До таких елементів належать характерні заголовки протоколу, типові фрагменти HTML-розмітки, назви або шляхи до статичних ресурсів, а також інші технічні ознаки відповіді вебресурсу. Сукупність виділених елементів оформлюється як потік «Ознаки технологій» і передається далі.

У блоці A-0-2-2 «Зіставлення ознак з довідником технологій» цей потік використовується для порівняння з наявним у системі довідником сигнатур. У довіднику для кожної технології або програмного продукту наведено набір ознак, характерних для відповідного вебсерверу, системи керування контентом, фреймворку клієнтської частини або іншого компонента. Серверна частина

інформаційної системи на цьому кроці встановлює, які сигнатури узгоджуються з отриманими ознаками, і формує «Набір припущень» щодо можливих технологій. Для кожного припущення може зберігатися рівень упевненості або інші параметри, що описують надійність зіставлення.

У блоці A-0-2-3 «Формування визначених технологій» набір окремих припущень перетворюється на узгоджений перелік, який надалі використовуватиметься для пошуку уразливостей у публічних базах. На цьому етапі усуваються дублікати, поєднуються близькі за змістом припущення, можуть відкидатися записи з низьким рівнем упевненості. Результатом є потік «Визначені технології», що виступає виходом процесу A-0-2 і надходить до наступного етапу функціональної моделі, присвяченого кореляції технологій з записами у базі National Vulnerability Database та переліку Common Vulnerabilities and Exposures. Усі підпроцеси на діаграмі виконуються за участю механізму «Серверна частина ІС», який забезпечує доступ до довідникових даних, оброблення ознак і збереження результатів для подальшого використання.

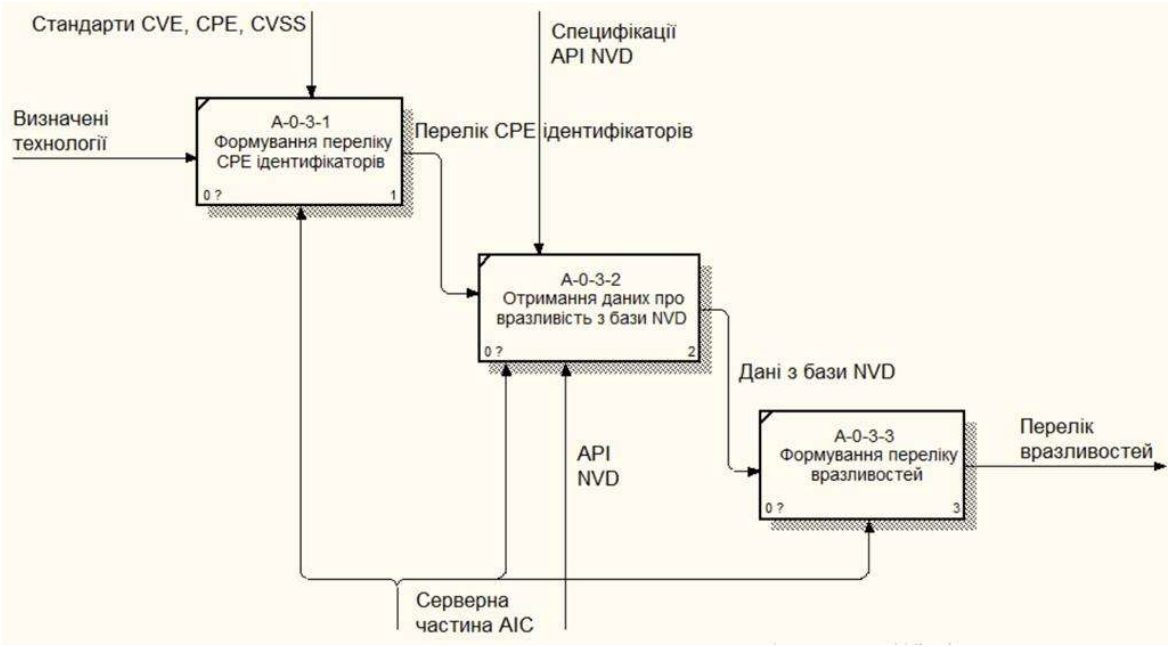

Рисунок 3.6 – Декомпозиція процесу «Кореляція технологій з базами вразливостей»

Наступний фрагмент функціональної моделі описує етап, на якому перелік

визначених технологій перетворюється на перелік уразливостей, отриманий із публічної бази National Vulnerability Database. Для цього основний процес кореляції технологій з базами уразливостей поділено на декілька послідовних підпроцесів, що дозволяє чітко показати, де саме застосовуються стандарти опису уразливостей CVE, CPE, CVSS і яким чином серверна частина інформаційної системи взаємодіє з програмним інтерфейсом бази NVD.

На рисунку 3.6 показано декомпозицію процесу A-0-3 «Кореляція технологій з базами вразливостей» на три підпроцеси A-0-3-1, A-0-3-2 та A-0-3-3.

У підпроцесі A-0-3-1 «Формування переліку CPE ідентифікаторів» використовується потік «Визначені технології», що надійшов з попереднього етапу моделі. Серверна частина інформаційної системи зіставляє назви та версії продуктів із довідниковими записами каталогу CPE і на основі цих зіставлень формує перелік CPE-ідентифікаторів. На виконання підпроцесу впливають стандарти CVE, CPE, CVSS, які визначають узгоджений спосіб опису програмних платформ і уразливостей, а на виході утворюється потік «Перелік CPE ідентифікаторів», що передається далі.

У підпроцесі A-0-3-2 «Отримання даних про вразливість з бази NVD» цей перелік CPE-ідентифікаторів використовується для формування запитів до прикладного програмного інтерфейсу бази NVD. Серверна частина інформаційної системи звертається до API NVD, дотримуючись його специфікацій, і отримує для кожного CPE-ідентифікатора набір записів про уразливості, у яких містяться ідентифікатори з переліку CVE, текстові описи, значення метрик CVSS та інші параметри. Результат оброблення відповідей бази даних оформлюється як потік «Дані з бази NVD», що виступає виходом цього підпроцесу та входом для наступного.

У підпроцесі A-0-3-3 «Формування переліку вразливостей» дані, отримані з бази NVD, приводяться до внутрішнього формату, придатного для подальшого використання в інформаційній системі. Серверна частина усуває дублікати, групує записи за технологіями або компонентами, може застосовувати фільтрацію за датою публікації чи іншими ознаками і формує узгоджений потік «Перелік

вразливостей». Цей потік одночасно є виходом процесу A-0-3 і надходить до наступного етапу функціональної моделі, де на основі переліку уразливостей обчислюється оцінка ризику та готується контекст для взаємодії з сервісом мовної моделі великого розміру.

На рисунку 3.7 подано декомпозицію процесу A-0-4 «Формування оцінки ризику та рекомендацій».

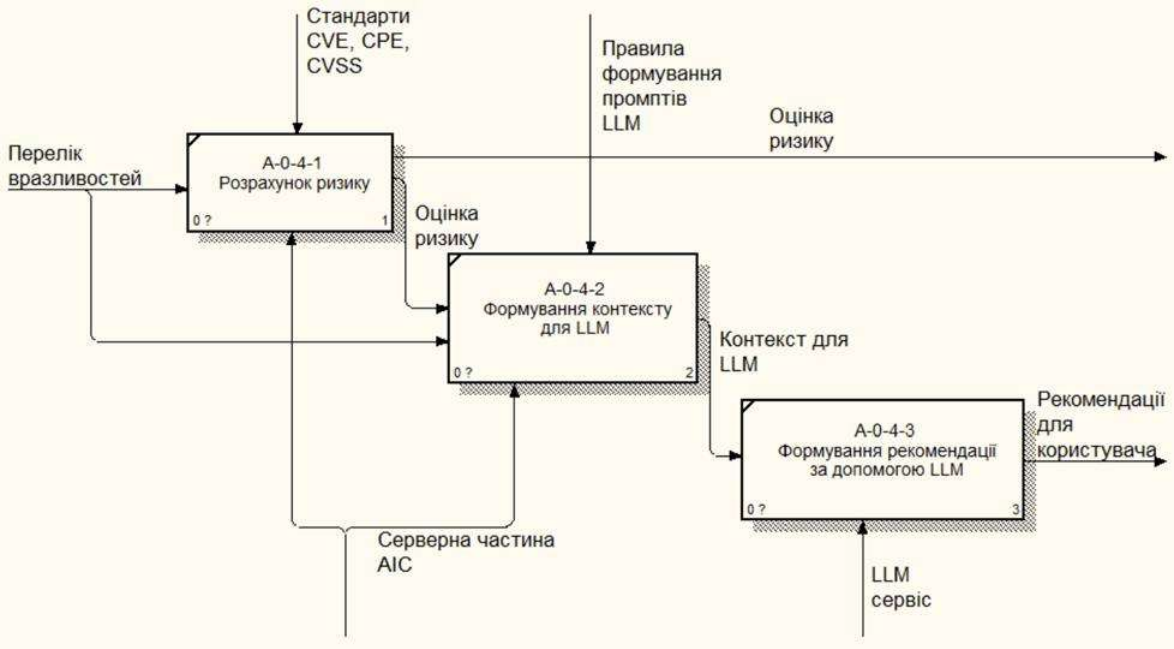

Рисунок 3.7 – Декомпозиція процесу формування оцінки ризику та рекомендацій

У підпроцесі A-0-4-1 «Розрахунок ризику» як вхід використовується потік «Перелік вразливостей», що надійшов із попереднього етапу. Серверна частина інформаційної системи застосовує до цього переліку правила інтерпретації метрик, задані стандартами переліку Common Vulnerabilities and Exposures, каталогу Common Platform Enumeration та методики Common Vulnerability Scoring System. На цій основі формується інтегральна «Оцінка ризику» для поточного домену. Цей показник виступає виходом підпроцесу, частина його виводиться безпосередньо за межі процесу A-0-4 для подальшого відображення у вигляді індикатора, а також використовується як вхідні дані для наступних кроків.

У підпроцесі A-0-4-2 «Формування контексту для LLM» використовується перелік вразливостей та розрахована оцінка ризику. Серверна частина

інформаційної системи формує на їхній основі структурований контекст, придатний для передавання до сервісу мовної моделі великого розміру. У цьому контексті узагальнюються ключові характеристики виявлених вразливостей, виділяються найбільш критичні записи, зазначається оцінка ризику та коротко описується ситуація для відповідного вебресурсу. На підпроцес впливають правила формування промптів для LLM, які визначають формат контексту, допустиму довжину та рівень деталізації. Результатом роботи є потік «Контекст для LLM», що передається далі.

У підпроцесі A-0-4-3 «Формування рекомендацій за допомогою LLM» структурований контекст використовується для звернення до LLM-сервісу. Серверна частина інформаційної системи надсилає контекст, отримує від мовної моделі відповідь у вигляді тексту, у якому описано можливі загрози для користувача веббраузера та наведено загальні поради щодо обережної поведінки під час роботи з вебресурсом. За потреби текст проходить додаткове службове опрацювання, після чого формується вихідний потік «Рекомендації для користувача», що повертається до браузерного розширення. Механізмом виконання цього підпроцесу є LLM-сервіс, а логіка підготовки й оброблення запитів реалізується серверною частиною інформаційної системи.

3.3 Функціональні вимоги

На основі побудованої функціональної моделі в нотації IDEF0 сформульовано сукупність функціональних вимог до інформаційної системи аналізу безпеки вебресурсів. Ці вимоги визначають перелік дій, які система повинна виконувати під час оброблення запиту на аналіз домену, а також склад результатів, що формуються для подальшого відображення в браузерному розширенні.

Інформаційна система повинна приймати від клієнтського компонента дані про домен вебресурсу, включно з доменним іменем та допоміжними параметрами запиту, і ініціювати процес аналізу для кожного такого звернення. Система

повинна забезпечувати ідентифікацію домену у внутрішній базі даних, виконувати перевірку наявності попередніх результатів аналізу та визначати їх актуальність на основі часових міток оновлення. У разі відсутності збережених даних або втрати їх актуальності система повинна формувати оновлені дані вебресурсу для подальшої обробки.

Інформаційна система повинна виконувати визначення технологій, що застосовуються на вебресурсі, на основі зібраних характеристик відповіді вебсервера та інших доступних технічних ознак. Результатом цього етапу має бути узгоджений перелік технологій, придатний для подальшої кореляції з публічними базами уразливостей. Система повинна використовувати довідникові дані та сигнатури технологій для зіставлення ознак і формування підсумкового набору визначених технологій.

Інформаційна система повинна забезпечувати кореляцію визначених технологій з публічними базами уразливостей шляхом використання ідентифікаторів каталогу Common Platform Enumeration. На основі сформованого переліку CPE-ідентифікаторів система повинна виконувати звернення до прикладного програмного інтерфейсу бази National Vulnerability Database та отримувати відомості про уразливості, пов’язані з відповідними технологіями. Отримані дані повинні бути приведені до внутрішнього формату, що містить ідентифікатори з переліку Common Vulnerabilities and Exposures, значення метрик оцінювання та інші необхідні характеристики.

Інформаційна система повинна формувати узгоджений перелік уразливостей, у якому результати з різних запитів до бази NVD групуються, нормалізуються та очищуються від дублікатів. Цей перелік має використовуватися як основа для подальшого оцінювання ризику та побудови текстових пояснень.

Інформаційна система повинна виконувати розрахунок оцінки ризику для домену на основі переліку вразливостей із застосуванням правил інтерпретації метрик, визначених стандартами Common Vulnerabilities and Exposures, Common Platform Enumeration та методикою Common Vulnerability Scoring System. Результатом цього етапу має бути числовий або категоріальний показник ризику,

придатний для подальшого використання в інтерфейсі браузерного розширення.

Інформаційна система повинна забезпечувати формування структурованого контексту для взаємодії з сервісом мовної моделі великого розміру на основі переліку вразливостей і розрахованої оцінки ризику. Сформований контекст повинен відповідати правилам побудови запитів до мовної моделі та містити узагальнений опис ситуації для конкретного вебресурсу.

Інформаційна система повинна забезпечувати взаємодію з сервісом мовної моделі великого розміру з метою отримання текстових рекомендацій. Отриманий від мовної моделі результат повинен оброблятися серверною частиною та формувати вихідний потік рекомендацій, який узагальнює потенційні загрози та містить загальні поради щодо обережної поведінки під час роботи з відповідним вебресурсом.

Інформаційна система повинна повертати до браузерного розширення результати аналізу у вигляді оцінки ризику, переліку вразливостей і текстових рекомендацій, а також забезпечувати збереження цих результатів у внутрішній базі даних для можливості повторного використання протягом визначеного інтервалу часу.

3.4 Варіанти використання

Варіанти використання інформаційної системи кореляції доменів з базами вразливостей описують сценарії взаємодії користувача веббраузера з функціональністю системи. Система орієнтована на фонове виконання аналізу без необхідності явного запуску процесів з боку користувача, тому кількість варіантів використання обмежується одним базовим сценарієм (рис. 3.8).

Основним актором інформаційної системи є користувач веббраузера. Його взаємодія з системою відбувається опосередковано через браузерне розширення, яке функціонує у фоновому режимі та автоматично реагує на зміну активного вебресурсу. Коли користувач переходить на нову вебсторінку або активує вкладку з уже відкритим вебресурсом, браузерне розширення визначає доменне ім’я

поточного вебресурсу та ініціює звернення до серверної частини інформаційної системи.

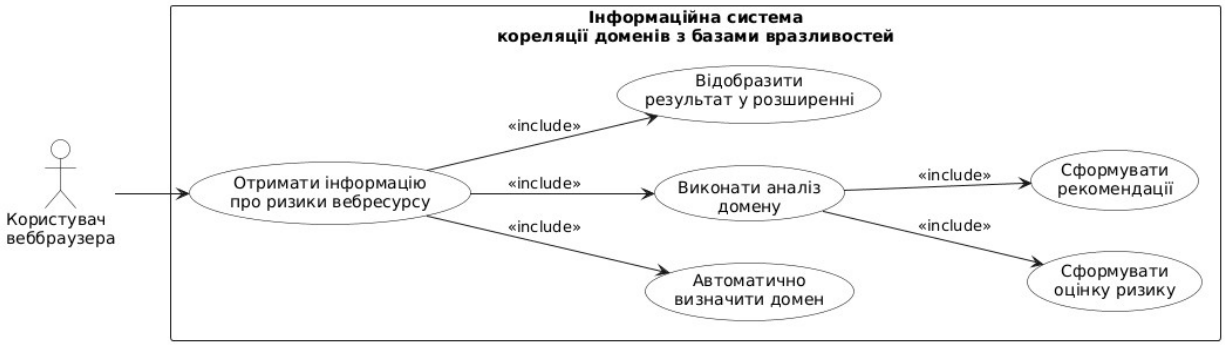

Рисунок 3.8 – Діаграма варіантів використання інформаційної системи кореляції доменів з базами вразливостей

У межах основного варіанту використання система автоматично виконує аналіз домену без додаткових дій з боку користувача веббраузера. Серверна частина послідовно реалізує етапи, визначені функціональною моделлю системи: оброблення даних про домен, визначення технологій вебресурсу, кореляцію встановлених технологій з публічними базами уразливостей, обчислення оцінки ризику та формування текстових рекомендацій із використанням сервісу мовної моделі великого розміру. Усі зазначені дії є внутрішніми процесами інформаційної системи та не розглядаються як окремі варіанти використання, оскільки не ініціюються безпосередньо користувачем.

Результатом виконання основного варіанту використання є повернення до браузерного розширення узагальнених даних про безпеку поточного вебресурсу. Мінімальним способом взаємодії з результатами аналізу є відображення індикатора оцінки ризику у вигляді зміни стану або вигляду значка розширення в інтерфейсі веббраузера. Такий індикатор дозволяє користувачеві швидко оцінити рівень потенційного ризику без необхідності відкриття додаткових елементів інтерфейсу.

Користувач веббраузера може відкрити інтерфейс браузерного розширення, скориставшись його значком. У цьому режимі система надає узагальнену оцінку ризику для поточного домену та текстові рекомендації, сформовані на основі

результатів аналізу. За необхідності може відображатися скорочений перелік виявлених уразливостей із зазначенням ідентифікаторів з переліку Common Vulnerabilities and Exposures та окремих характеристик, отриманих із бази National Vulnerability Database. Представлення результатів орієнтоване на інформування користувача про ризики та не передбачає відображення повних технічних описів уразливостей. Під час проєктування варіантів використання інформаційної системи свідомо обмежено перелік доступних сценаріїв взаємодії. Система не передбачає окремих режимів для явного запуску аналізу, введення довільних доменних імен або пакетного опрацювання переліків вебресурсів. Також не реалізовано функціональність масового експорту детальних технічних даних про уразливості. Такі обмеження зумовлені тим, що інформаційна система орієнтована на аналіз вебресурсу, з яким користувач веббраузера вже взаємодіє, а не на виконання задач професійного аудиту або розвідки уразливостей.

З боку користувача веббраузера взаємодія з інформаційною системою зводиться до одного варіанту використання, який можна узагальнено описати як автоматичне отримання інформації про ризики під час відвідування вебресурсу. Саме цей варіант використання відображено на діаграмі варіантів використання, де внутрішні етапи аналізу представлені як включені дії в межах основного сценарію та не формують окремих гілок поведінки.

3.5 Нефункціональні вимоги

Інформаційна система має забезпечити передавання даних між браузерним розширенням і серверною частиною через захищене з’єднання. Вхідний параметр із доменним іменем підлягає валідації на відповідність допустимому формату до виконання будь-яких звернень до локального сховища та зовнішніх сервісів. Для зменшення ризику деградації працездатності за умов підвищеного навантаження або надсилання великої кількості запитів має бути передбачене обмеження частоти запитів до прикладного програмного інтерфейсу та контроль ресурсів обробки.

Потрібно реалізувати механізм кешування результатів аналізу домену з урахуванням часових міток останньої перевірки та логіки застарівання даних. За наявності актуального запису в локальному сховищі формування відповіді виконується без звернення до зовнішніх джерел. За відсутності запису або після спливу визначеного інтервалу актуальності ініціюється оновлення відомостей із бази NVD з подальшим збереженням результатів та оновленням часових міток.

Під час взаємодії із зовнішніми сервісами враховується можливість часткових або повних відмов, зокрема недоступність прикладного програмного інтерфейсу бази NVD або сервісу великої мовної моделі. У таких випадках, якщо в локальному сховищі наявні попередні результати, допускається повернення останньої доступної версії даних із фіксацією стану аналізу у відповіді. Невдача під час отримання рекомендацій від великої мовної моделі не повинна блокувати повернення структурованої інформації про домен і перелік уразливостей, якщо ці дані отримано з бази NVD або збережено раніше.

Клієнтська частина орієнтується на сучасні веббраузери на базі Chromium і підтримує фоновий режим роботи, реагуючи на зміну активної вкладки або навігацію на нову сторінку. Зчитування доменного імені та ініціювання запиту до серверної частини виконуються автоматично, а відображення результатів здійснюється через інтерфейс розширення без потреби у додаткових налаштуваннях з боку користувача веббраузера.

Інтерфейс браузерного розширення має забезпечувати пріоритетне відображення результату оцінювання ризику у вигляді числового показника, який використовується як основний підсумок аналізу для поточного вебресурсу. Разом із цим показником у межах одного екрану має бути доступною ключова контекстна інформація, зокрема доменне ім’я та час останньої перевірки, що дозволяє співвіднести результат аналізу з конкретним вебресурсом і актуальністю отриманих даних.

Основним змістовним елементом інтерфейсу мають бути рекомендації з підвищення безпеки, сформовані для користувача веббраузера на основі результатів аналізу. Рекомендації мають відображатися одразу після підсумкового

показника ризику та бути представленими у стислому вигляді, придатному для безпосереднього застосування, без необхідності переходу на зовнішні ресурси або виконання додаткових дій для їх відкриття.

Додаткова інформація про виявлені уразливості має залишатися доступною в інтерфейсі як окремий інформаційний блок, що підтримує уточнення підстав сформованого рівня ризику та рекомендацій. Перелік уразливостей має містити ідентифікатор CVE, оцінку за методикою CVSS і короткий опис, що забезпечує можливість швидкого перегляду та подальшого звернення до першоджерел за потреби. На основі наведених вимог до подання результатів аналізу розробленно макет інтерфейсу браузерного розширення (рис. 3.9).

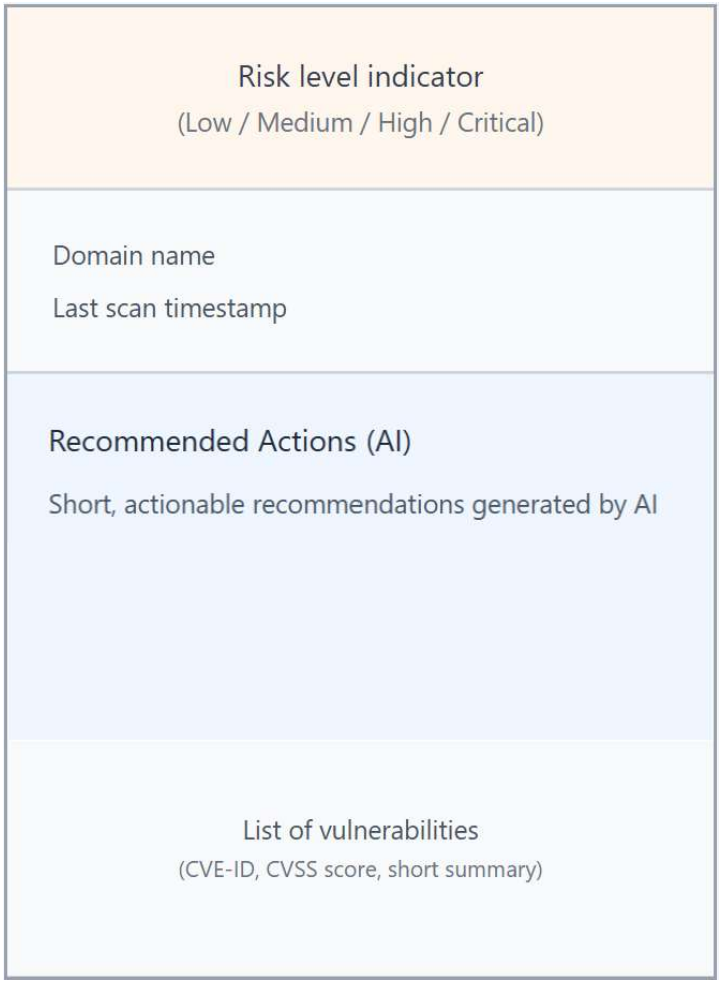

Рисунок 3.9 – Макет інтерфейсу браузерного розширення

3.6 Структура даних та логічна модель бази даних

База даних інформаційної системи призначена для зберігання результатів аналізу доменів, відомостей про використані технології вебресурсів, описів уразливостей та сформованих оцінок ризику й текстових рекомендацій. Структуру цих даних доцільно розглядати на рівні логічної моделі, яка подається у вигляді діаграми зв’язків «сутність–зв’язок» (Entity-Relationship Diagram, ERD) у нотації IDEF1X. Така діаграма відображає сутності, їх атрибути та відношення між ними, що безпосередньо підтримують роботу серверної частини інформаційної системи.

У логічній моделі (рис. 3.10) виділяються сутності, які описують домени, технології, уразливості, звіти аналізу та рекомендації. Сутність domain репрезентує вебресурс, для якого виконується аналіз. Сутність web_technology відображає програмні платформи й компоненти, які можуть бути виявлені на вебресурсі. Зв’язок між цими сутностями реалізується через проміжну таблицю domain_web_technology, що дає змогу фіксувати множину технологій для одного домену та повторно використовувати опис однієї технології для різних доменів.

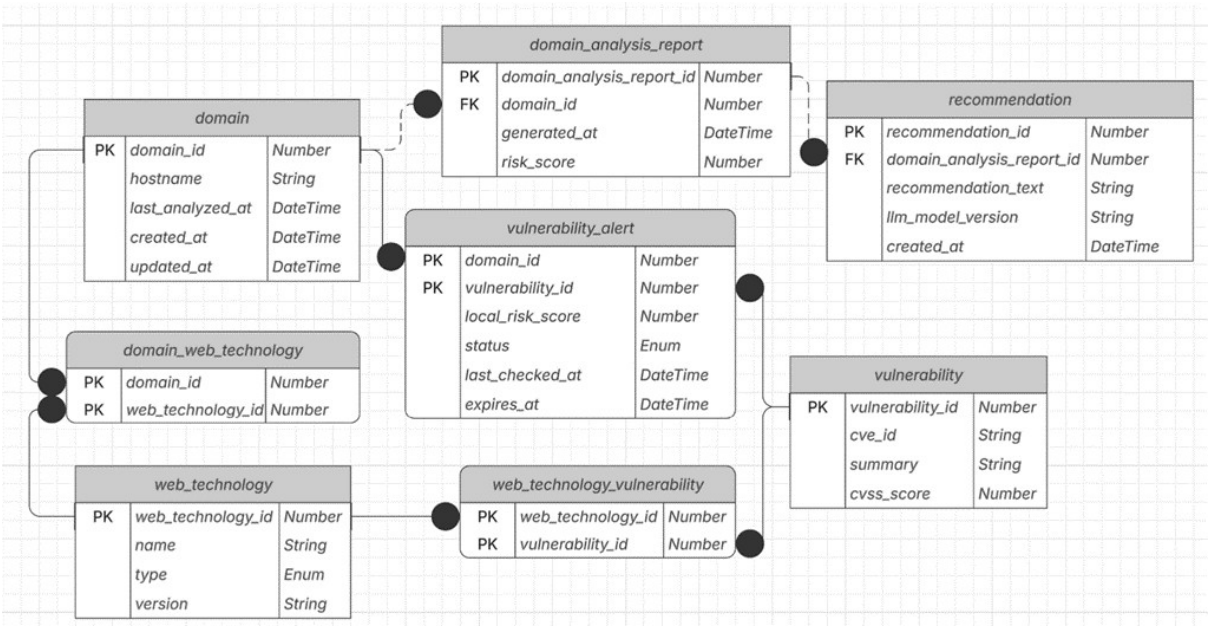

Рисунок 3.10 – Логічна модель даних інформаційної системи кореляції доменів з базами вразливостей

Сутність vulnerability містить узагальнену інформацію про уразливість, включно з

ідентифікатором з переліку Common Vulnerabilities and Exposures та значенням метрики методики Common Vulnerability Scoring System. Через таблицю web_technology_vulnerability фіксується, до яких технологій застосовується певна уразливість, що відповідає зв'язку «багато-до-багатьох» між технологіями та уразливостями.

Для відображення результатів конкретних запусків аналізу використовується сутність `domain_analysis_report`. Вона пов'язана з сутністю «domain» і містить часову мітку формування звіту та числовий показник ризику, який використовується у браузерному розширенні. На основі звіту формується сутність «recommendation», де зберігається текст рекомендацій, згенерований сервісом великої мовної моделі, а також позначення версії моделі, що дозволяє відстежувати умови, за яких було побудовано пояснення. Сутність `vulnerability_alert` описує пов'язані з доменом уразливості з урахуванням локальних характеристик: тут зберігається посилання на домен і уразливість, локальний показник ризику, стан сповіщення, а також пари полів `last_checked_at` та `expires_at`, які визначають строк актуальності оцінки й підтримують механізм обмеження часу дії результатів аналізу.

У показаній моделі таблиця domain містить ідентифікатор домену, символьне поле «`hostname`», відмітку часу останнього аналізу та службові поля створення й оновлення запису. Для кожного домену можуть існувати декілька записів у таблиці `domain_analysis_report`, що відображає можливість повторного аналізу того самого вебресурсу в різні моменти часу. Зв'язок між цими таблицями реалізовано через зовнішній ключ `domain_id` у таблиці звітів. Для кожного звіту може бути створено одну або декілька рекомендацій у таблиці `recommendation`, де додатково фіксуються текст рекомендації, версія використаної мовної моделі та час формування запису.

Блок таблиць, пов'язаних з технологіями, забезпечує опис середовища виконання вебресурсу. Таблиця `web_technology` містить ідентифікатор технології, її назву, тип та версію. Через таблицю `domain_web_technology` серверна частина може зберігати множину записів, які відображають, які саме технології були

виявлені під час аналізу конкретного домену; первинний ключ цієї таблиці складається з пари полів `domain_id` та `web_technology_id`, що унеможливлює дублювання однакових зв'язків. Таблиця `web_technology_vulnerability` пов'язує технології з уразливостями: кожний запис вказує, що відповідна уразливість потенційно стосується певної технології, а отже, у разі використання цієї технології на домені відповідний запис може бути включений до переліку уразливостей для цього домену.

Окремо виділено таблиці, пов'язані з уразливостями та сповіщеннями. Таблиця `vulnerability` містить внутрішній ідентифікатор `vulnerability_id`, символьне поле `cve_id`, у якому зберігається ідентифікатор з переліку Common Vulnerabilities and Exposures, короткий опис уразливості та числове поле «`cvss_score`», яке містить значення метрики методики Common Vulnerability Scoring System. Таблиця `vulnerability_alert` пов'язує уразливості з доменами і містить поля `domain_id` та `vulnerability_id` як складений первинний ключ. У цій таблиці додатково зберігаються локальний показник ризику, поточний стан сповіщення, а також поля `last_checked_at` і `expires_at`, які використовуються для контролю строку актуальності інформації. Така структура дає змогу відокремити глобальний опис уразливості від локальної оцінки її впливу на конкретний домен.

За допомогою таблиці доменів, технологій, уразливостей та сповіщень серверна частина може відновити повний контекст для розрахунку ризику й формування рекомендацій, а механізм часових атрибутів у сутності `vulnerability_alert` забезпечує можливість реалізувати стратегію регулярної актуалізації даних з бази National Vulnerability Database без дублювання інформації.

3.7 Проектування структури класів

Для проєктування серверної частини інформаційної системи важливо зафіксувати структуру класів і межі їхньої відповідальності, а також визначити інтерфейси взаємодії між ними. Для цього використовується UML (Unified

Modeling Language) діаграма класів, яка відображає структуру сервісного шару та вхідної точки прикладного програмного інтерфейсу без прив’язки до порядку викликів і часових характеристик виконання.

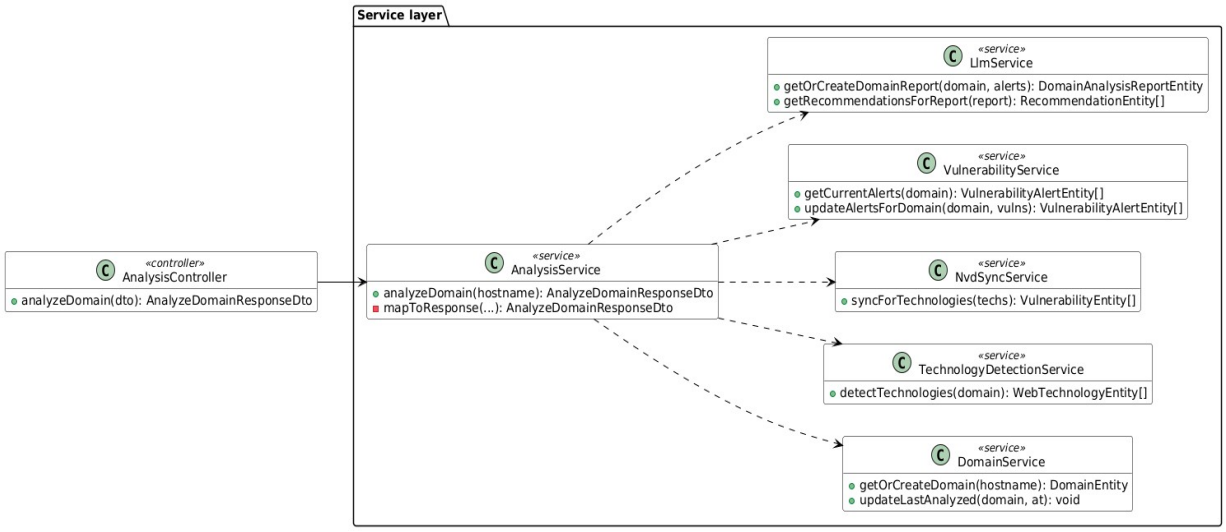

Рисунок 3.11 – Діаграма классів серверної частини інформаційної системи

На рисунку 3.11 подано проєктовану структуру класів серверної частини, у якій виділено вхідний шар оброблення запитів та сервісний шар. Такий поділ узгоджується з підходом, де контролери відповідають за приймання запитів і формування відповідей, а сервісні класи реалізують прикладну логіку.

Вхідною точкою прикладного програмного інтерфейсу виступає клас `AnalysisController`, який надає один публічний метод `analyzeDomain(dto): AnalyzeDomainResponseDto`. На рівні проєктування цей клас розглядається як елемент маршрутизації та узгодження формату обміну даними з браузерним розширенням, тоді як виконання бізнес логіки винесено в сервісний шар.

Координацію основного сценарію аналізу доменного імені зосереджено в класі `AnalysisService`, який має публічний метод `analyzeDomain(hostname)`. Для формування вихідної структури передбачено допоміжний метод `mapToResponse`, що забезпечує перетворення доменних сутностей у відповідь, придатну для відображення у браузерному розширені.

Операції, пов’язані з доменом як об’єктом предметної області, виділено в окремий сервіс `DomainService`. Його інтерфейс складається з методів

getOrCreateDomain(hostname) та updateLastAnalyzed(domain). Таке рішення концентрує створення та пошук домену, а також оновлення часових атрибутів у межах одного компонента, що зменшує кількість прямих залежностей інших сервісів від деталей доступу до даних.

Визначення технологій вебресурсу передбачено виконувати у TechnologyDetectionService, який надає метод detectTechnologies(domain). На діаграмі цей сервіс використовується AnalysisService як джерело вхідних даних для подальшої кореляції з відомостями про уразливості, тоді як логіка побудови результату визначення технологій залишається інкапсульованою в межах спеціалізованого сервісу.

Інтеграцію з базою NVD у частині актуалізації відомостей про уразливості для виявлених технологій представлено сервісом NvdSyncService з методом syncForTechnologies(techs). На рівні проєктування цей інтерфейс фіксує межу відповідальності компонента: отримання узгодженого набору сутностей уразливостей для визначеного переліку технологій, без деталізації внутрішнього протоколу звернення до бази NVD.

Доменну прив’язку уразливостей та керування станом попереджень для конкретного домену відокремлено в VulnerabilityService, який містить методи getCurrentAlerts(domain) та updateAlertsForDomain(domain, vulns). Така структура розділяє поняття «глобальна уразливість, пов’язана з продуктом/технологією» і «попередження для домену», що використовується під час відображення результатів аналізу та побудови звіту.

Генерацію та пов’язування рекомендацій із результатами аналізу домену віднесено до LlmService. На діаграмі цей компонент надає метод getOrCreateDomainReport(domain, alerts), який визначає інтерфейс формування або оновлення сформованого звіту, а також метод getRecommendationsForReport(report), що повертає набір рекомендацій, пов’язаних зі сформованим звітом. На рівні UML-діаграми класів фіксуються саме інтерфейси взаємодії інших компонентів із цим сервісом, тоді як внутрішній механізм отримання тексту рекомендацій залишається поза межами структурної

моделі.

Спроєктована структура серверної частини подана як композиція контролера прикладного програмного інтерфейсу та набору сервісів із розмежованими зонами відповідальності. Клас `AnalysisService` виступає координаційним елементом, що узгоджує результати визначення технологій, кореляції уразливостей із базою NVD, актуалізації попереджень для домену та формування звіту з рекомендаціями, не змішуючи ці задачі в одному компоненті.

3.8 Послідовність процесу кореляції домену з базами даних вразливостей

Послідовність процесу кореляції домену з базами даних уразливостей (рис.3.12) проєктується як керований сценарій оброблення запиту на аналіз безпеки вебресурсу, у межах якого забезпечується узгоджена взаємодія між клієнтським компонентом у середовищі веббраузера, серверною частиною інформаційної системи та зовнішніми джерелами відомостей про уразливості. Процес ініціюється надходженням запиту від браузерного розширення, у якому передається доменне ім’я вебресурсу як базовий ідентифікатор для подальших етапів аналізу.

Після приймання запиту серверна частина ініціює отримання або створення запису про домен у локальному сховищі інформаційної системи. Такий підхід забезпечує уніфіковане представлення вебресурсу в межах системи та дозволяє виконувати всі наступні операції щодо узгодженого об’єкта домену незалежно від того, чи аналізувався цей вебресурс раніше. Після цього виконується перевірка наявності попередньо сформованих попереджень про уразливості, пов’язаних із цим доменом, а також перевірка їх актуальності з урахуванням встановленого строку чинності.

У разі, якщо для домену наявні попередження, строк дії яких не завершився, система переходить до скороченого сценарію оброблення. У межах цього сценарію не виконується повторне визначення технологій вебресурсу та повторне отримання відомостей про уразливості з бази даних NVD, оскільки відповідні дані

вже присутні у локальному сховищі та відповідають критеріям актуальності. Подальша обробка зводиться до формування агрегованого звіту аналізу та ініціювання отримання рекомендацій на основі збереженого набору попереджень.

Якщо ж актуальні попередження для домену відсутні або збережені дані втратили актуальність, система переходить до повного циклу кореляції. На першому етапі цього циклу виконується визначення технологій вебресурсу для заданого домену, результатом чого є перелік технологій, що описує виявлений технологічний стек вебресурсу та слугує основою для подальшого аналізу.

На основі сформованого переліку технологій здійснюється кореляція з відомостями про уразливості, що зберігаються у базі даних NVD. Для кожної виявленої технології виконується пошук відповідних ідентифікаторів уразливостей з урахуванням типу технології, її призначення та, за наявності, версійних характеристик. Отримані записи проходять етап узгодження, у межах якого усуваються дублікати, відсікаються нерелевантні або застарілі записи та формується єдиний узгоджений перелік уразливостей, релевантний конкретному домену.

Після узгодження відомостей про уразливості сформований перелік трансформується у доменно-орієнтовані попередження, що зберігаються у локальному сховищі інформаційної системи. Для кожного попередження фіксуються параметри актуальності, зокрема час останнього оновлення та граничний строк чинності, а також числові характеристики, необхідні для подальшого оцінювання рівня ризику. кореляції за умови збереження актуальності даних.

Після формування або оновлення набору попереджень виконується побудова агрегованого звіту аналізу домену. У межах цього етапу узагальнюються відомості про домен, пов’язані з ним уразливості та числові показники, що використовуються для оцінювання рівня ризику. Паралельно ініціюється отримання текстових рекомендацій, які формуються на основі змісту агрегованого звіту та призначені для подальшого відображення у клієнтському інтерфейсі браузерного розширення.

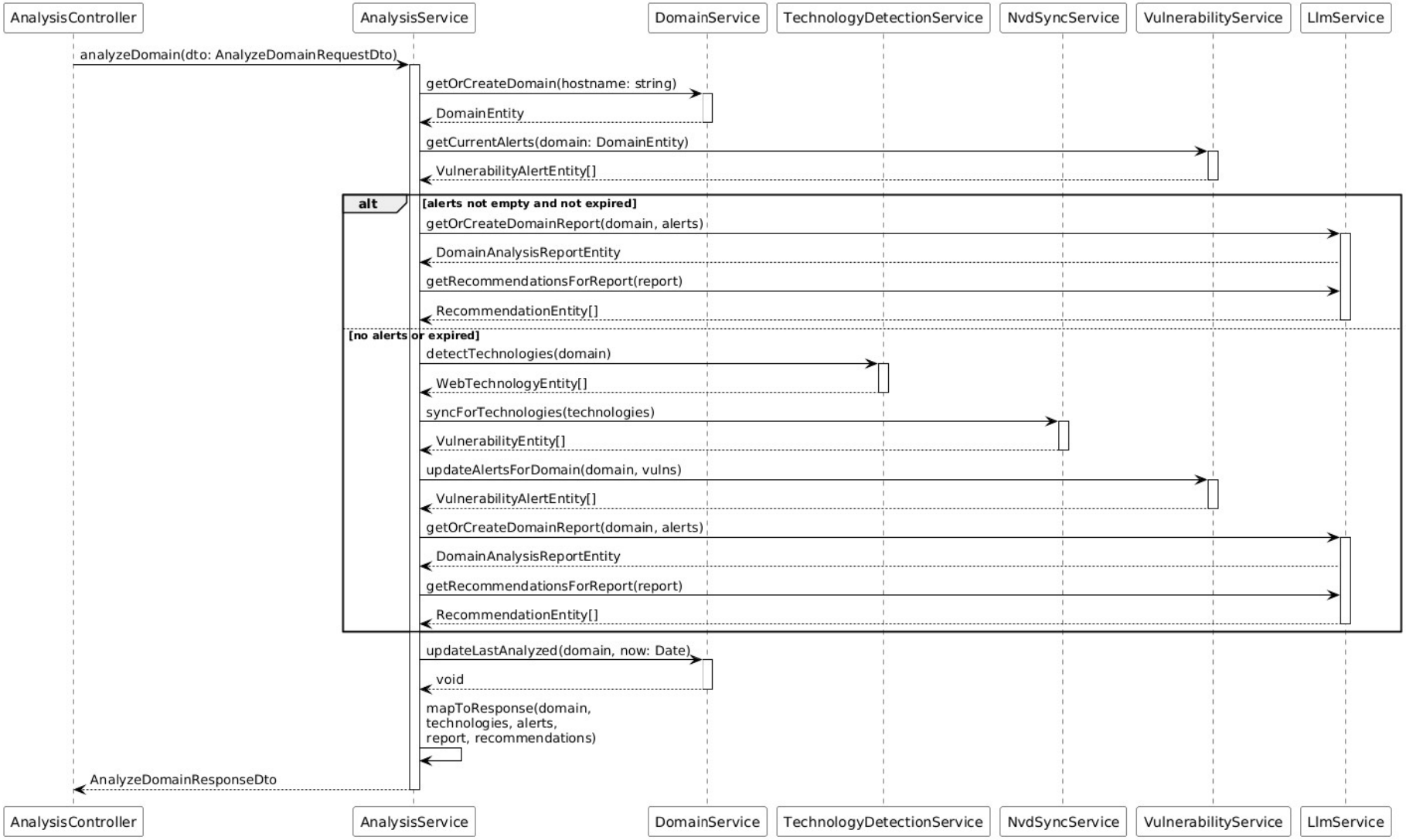

Рисунок 3.12 – Діаграма послідовності процесу кореляції домену з базами даних вразливостей

Завершальним етапом процесу є формування результату аналізу у стандартизованому форматі, який включає доменне ім’я, час останньої перевірки, узагальнену оцінку рівня ризику, перелік уразливостей і текст рекомендацій у разі його наявності. Після цього у локальному записі про домен фіксується час виконаного аналізу, що забезпечує коректну роботу механізму контролю актуальності даних та повторного використання результатів під час наступних звернень до інформаційної системи.

3.9 Проєктування алгоритму роботи браузерного розширення

Для формалізації процесу роботи браузерного розширення використовується UML (Unified Modeling Language) діаграма діяльності (рис. 3.13), яка фіксує послідовність операцій і точки прийняття рішень у двох частинах розширення.

Архітектурно передбачається розділення логіки на два модулі: фоновий скрипт і скрипт спливаючого вікна. Фоновий скрипт працює у подієвому режимі та реагує на зміну стану вкладок, зокрема на активацію вкладки та завершення завантаження сторінки. Після настання події виконується отримання адреси поточної вкладки та перевірка її придатності для аналізу. У проєкті визначаються критерії, за якими адреси службових сторінок браузера та локальних ресурсів не підлягають аналізу, щоб уникати зайвих звернень до серверної частини й формування звітів для об’єктів, що не є вебресурсами в межах предметної області.

Якщо адреса відповідає критеріям придатності, виконується виділення доменного імені з URL та підготовка запиту до серверного прикладного програмного інтерфейсу інформаційної системи. У відповідь очікується структурований результат аналізу, який містить оцінку рівня ризику, перелік уразливостей і сформовані рекомендації. Для відокремлення моменту отримання даних від моменту їх відображення передбачається збереження останнього отриманого результату в локальному сховищі розширення. Якщо під час запиту

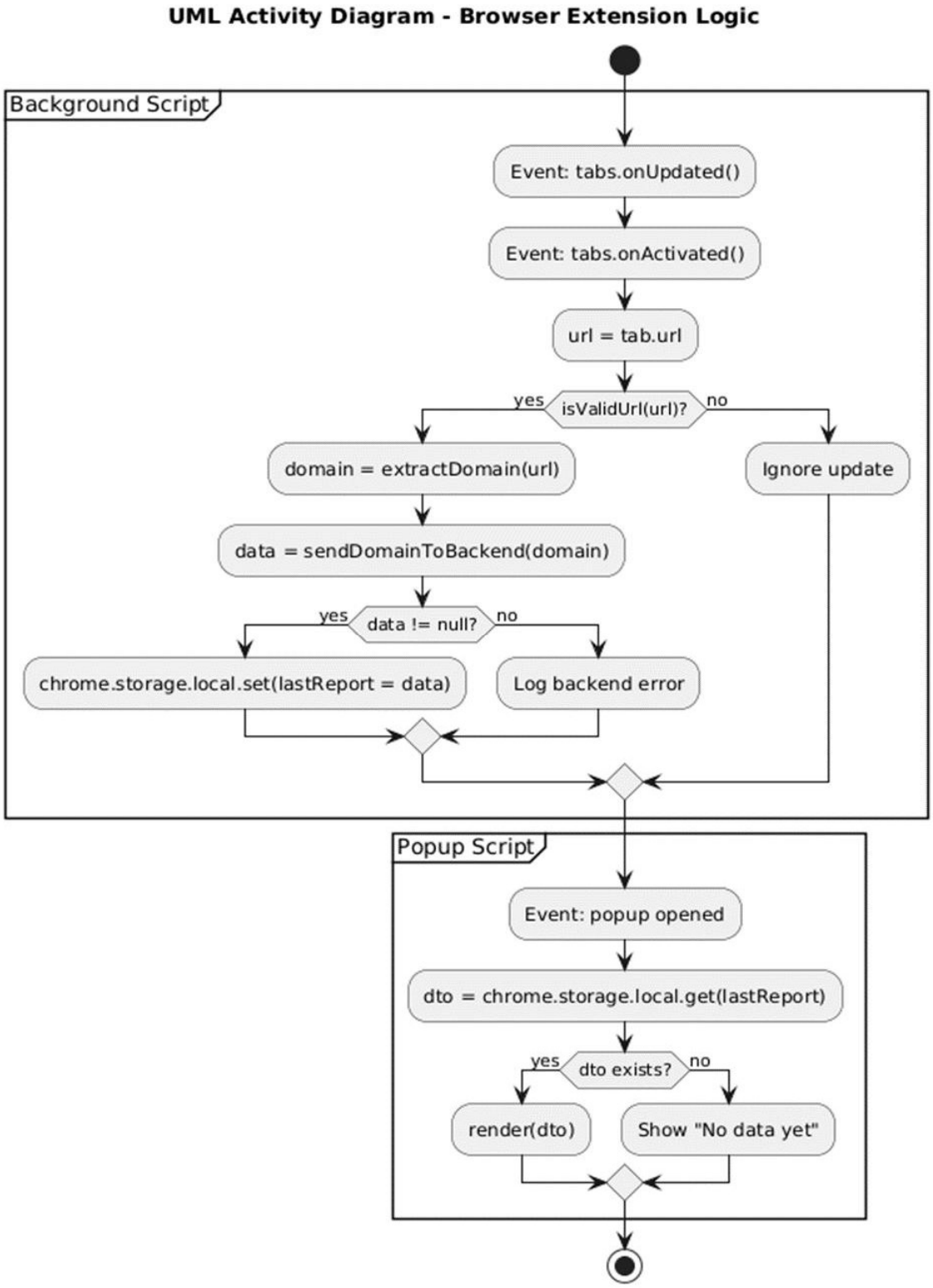

Рисунок 3.13 – Діаграма діяльності браузерного розширення

до серверної частини виникає помилка або відповідь не містить коректних даних, оновлення локального сховища не виконується, щоб не заміщати попередній

результат некоректним станом.

Скрипт спливаючого вікна активується лише під час відкриття інтерфейсу розширення користувачем веббраузера. У межах проєктування цей модуль не ініціює мережевих запитів, а використовує дані, що були підготовлені фоновим скриптом. Після ініціалізації скрипт читає зі сховища останній доступний результат аналізу й передає його в модуль відображення, який формує вміст інтерфейсу: підсумковий показник ризику, рекомендації та додаткові відомості про виявлені уразливості. Якщо у сховищі відсутні дані, інтерфейс відображає службове повідомлення про відсутність результатів, що відповідає сценарію першого запуску або ситуації, коли аналіз для поточного вебресурсу ще не завершено.

Запроєктована послідовність дій дозволяє розмежувати подієве отримання результатів аналізу та їх візуальне подання. Фоновий скрипт виконує автоматичне оновлення даних під час навігації користувача веббраузера, тоді як спливаюче вікно забезпечує перегляд вже підготовленої інформації без додаткових затримок, пов’язаних із мережевими запитами.

3.10 Проектування алгоритму кореляції доменів з базами даних вразливостей

Алгоритм кореляції доменів з базами даних вразливостей з двох послідовних етапів. Спочатку серверна частина визначає технологічний профіль вебресурсу, а потім на основі виявлених технологій виконує пошук відповідних записів про уразливості у локальному сховищі, яке синхронізується з базою вразливостей. Обидва етапи реалізовано у вигляді окремих сервісів, що інкапсулюють відповідну бізнес-логіку та працюють поверх сутностей бази даних і клієнтів зовнішніх прикладних програмних інтерфейсів.

На рисунку 3.14 наведено UML-діаграму алгоритму кореляції доменів з базами даних вразливостей.

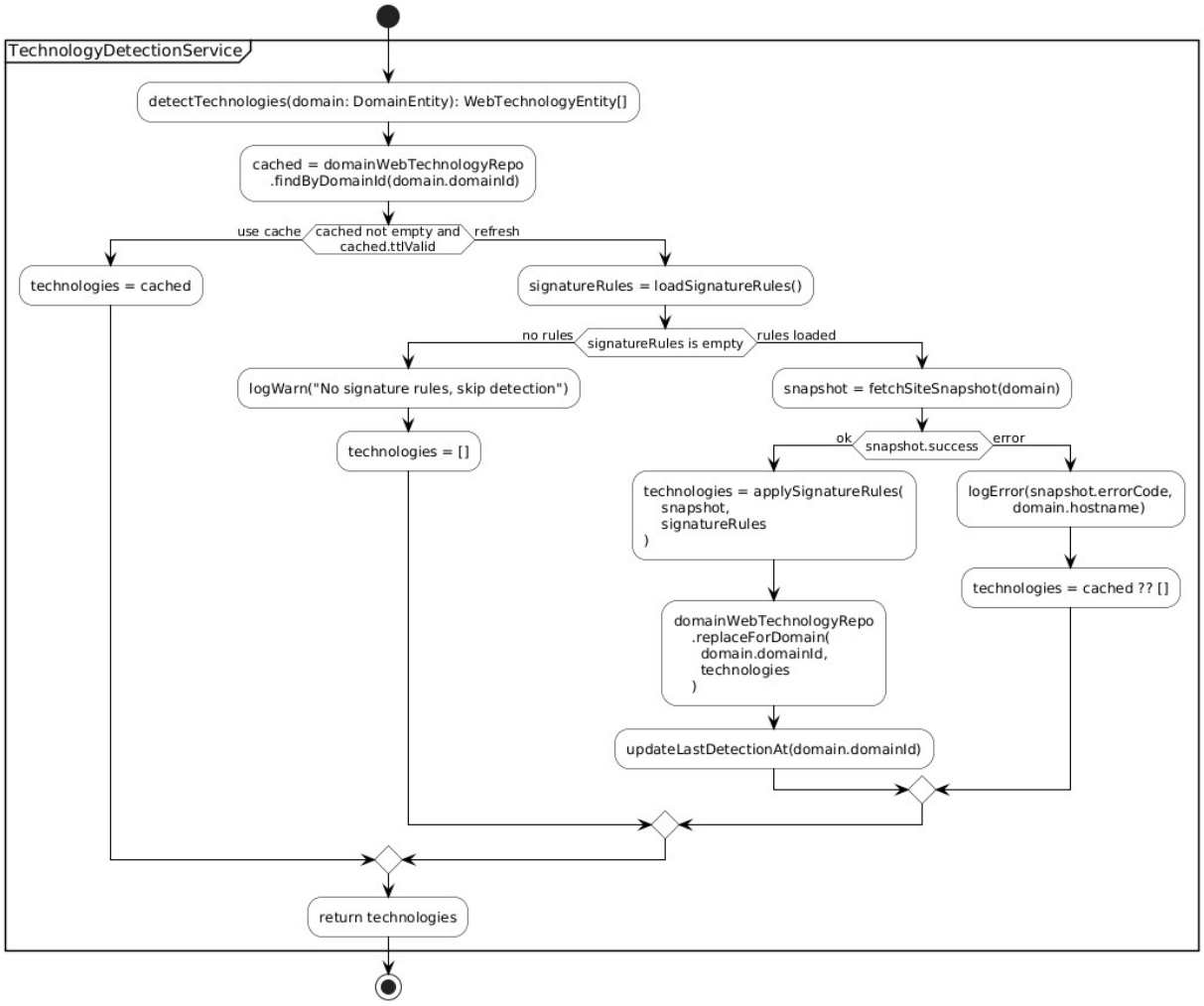

Рисунок 3.14 – Діаграма діяльності алгоритму кореляції доменів з базами вразливостей

Публічний метод `detectTechnologies(domain)` сервісу `TechnologyDetectionService` приймає об’єкт домену й на початку намагається використати вже наявні результати аналізу. Для цього за допомогою виконується пошук пов’язаних записів у таблиці зв’язку між доменами та вебтехнологіями. Якщо кеш не порожній і часовий атрибут ttl для цих записів вказує на те, що результати ще не вважаються застарілими, метод просто повертає отриманий список `WebTechnologyEntity` без додаткових мережевих запитів. Така гілка алгоритму відповідає ситуації, коли кореляція домену з базою вразливостей надалі може виконуватися на основі вже відомого технологічного профілю.

У випадку, коли кеш відсутній або строк його чинності минув, серверна частина переходить до оновлення даних про технології. На діаграмі

діяльності(рис. 3.14) це відповідає гілці `refresh`. Спочатку викликається допоміжний метод `loadSignatureRules()`, який завантажує файл сигнатур з локальної файлової системи або з кешу в оперативній пам'яті. У цьому файлі зберігається структурований набір правил, що описують відповідність фрагментів HTTP-заголовків, HTML-вмісту та інших ознак конкретним технологіям. Якщо з будь-якої причини файл сигнатур недоступний або після розбору виявляється порожнім, сервіс фіксує попередження через `logWarn` і повертає порожній список технологій; у такій ситуації подальша кореляція домену з базою вразливостей неможлива й інформаційна система відображає, що для даного вебресурсу не вдалося визначити технологічний стек.

За наявності коректного набору правил сервіс ініціює отримання миттєвого знімка вебресурсу. Для цього викликається метод `fetchSiteSnapshot(domain)`, який формує HTTP-запит до домену, зберігає отримані заголовки та HTML-код і повертає їх у вигляді об'єкта snapshot. У разі успішної відповіді викликається функція `applySignatureRules(snapshot, signatureRules)`, що послідовно застосовує правила до зібраних даних. Для кожного правила перевіряються регулярні вирази або шаблони для HTTP-заголовків, тіло відповіді сканується на наявність характерних маркерів, а у разі збігу створюється екземпляр `WebTechnologyEntity` з назвою технології, типом та, за можливості, версією. Сформований таким чином список технологій записується до бази даних: репозиторій `domainWebTechnologyRepo` виконує операцію replaceForDomain(domain.domainId, technologies), видаляючи застарілі записи й додаючи нові. Після цього оновлюється часовий атрибут у базі даних за допомогою виклику `updateLastDetectionAt(domain.domainId)`, який фіксує момент останнього успішного визначення технологій для цього домену. Сам метод `detectTechnologies` повертає список `WebTechnologyEntity`, що надалі використовується наступними компонентами алгоритму кореляції.

Другий етап алгоритму полягає у пошуку уразливостей, релевантних виявленим технологіям. На рівні реалізації цю частину кореляції виконують сервіси `NvdSyncService` та `VulnerabilityService`. Перший працює з локальним

представленням бази вразливостей і зовнішнім прикладним програмним інтерфейсом, а другий відповідає за зв’язування уразливостей із доменами. Після отримання списку технологій координуючий сервіс аналізу викликає метод `syncForTechnologies(techs)` сервісу `NvdSyncService`. Усередині цього методу технології зіставляються з ідентифікаторами програмних продуктів, які використовуються у базі вразливостей, формуються запити до локального сховища даних, що містить синхронізовані записи з національної бази вразливостей, і перевіряється, коли востаннє виконувалося оновлення даних для конкретних технологій. Якщо часові атрибути вказують на те, що локальні записи ще актуальні, метод повертає існуючі об’єкти `VulnerabilityEntity`. Якщо ж інформація застаріла або відсутня, сервіс надсилає запити до прикладного програмного інтерфейсу бази вразливостей, отримує перелік записів CVE разом із їхніми оцінками за методикою CVSS, оновлює таблиці vulnerability та `web_technology_vulnerability` і тільки після цього повертає оновлений список `VulnerabilityEntity` для переданого набору технологій.

Сервіс `VulnerabilityService` завершує кореляцію на рівні установлення зв’язку між конкретним доменом та знайденими уразливостями. Його метод `updateAlertsForDomain(domain, vulns)` отримує домен та перелік уразливостей, пов’язаних з його технологіями, завантажує чинні записи з таблиці, де зберігаються попередні результати аналізу, і для кожної пари домен–уразливість або створює новий запис, або оновлює наявний. Під час такого оновлення зберігається ідентифікатор уразливості, її числова оцінка за CVSS та службові часові атрибути, які визначають момент останньої перевірки та строк актуальності локального результату. У сукупності виклики `detectTechnologies`, `syncForTechnologies` та `updateAlertsForDomain` реалізують практичний алгоритм кореляції домену з базами вразливостей: від технологічного профілю вебресурсу через локально синхронізовані дані бази вразливостей до структурованого набору записів про уразливості, які можуть бути використані для подальшого розрахунку рівня ризику та формування рекомендацій за допомогою сервісу великої мовної моделі.

4 РЕАЛІЗАЦІЯ БРАУЗЕРНОГО РОЗШИРЕННЯ КОРЕЛЯЦІЇ ДОМЕНІВ З БАЗАМИ ВРАЗЛИВОСТЕЙ

4.1 Реалізація браузерного розширення

Клієнтську частину інформаційної системи реалізовано у вигляді браузерного розширення для веббраузерів на базі платформи Chromium, оскільки таке середовище надає прикладні програмні інтерфейси для роботи з вкладками та локальним збереженням стану, що є потрібним для автоматичного зчитування адреси активної вкладки й відображення підготовленого результату аналізу.

Браузерне розширення у контексті платформи Chromium є окремим прикладним компонентом, який виконується у межах середовища веббраузера та взаємодіє з ним через стандартизовані прикладні програмні інтерфейси. На відміну від вебзастосунків, що працюють у контексті конкретної вебсторінки, розширення має власний життєвий цикл та не прив’язане безпосередньо до вмісту сторінки. Воно підключається до браузера на рівні платформи розширень і отримує доступ до подій навігації, стану вкладок та локального сховища незалежно від того, який вебресурс відкрито користувачем.

Архітектурно браузерне розширення складається з окремих ізольованих компонентів, що виконують різні ролі. Фоновий модуль відповідає за обробку подій браузера та взаємодію з серверною частиною інформаційної системи, тоді як інтерфейс користувача реалізується у вигляді спливаючого вікна, доступного через панель інструментів браузера. Такий підхід дозволяє інтегрувати логіку аналізу безпеки у робочий процес веббраузера без модифікації вебсторінок і без втручання у мережеві з’єднання, обмежуючись використанням можливостей, які надає сама платформа Chromium.

Файл `manifest.json` (рис. 4.1) визначає ключові параметри розширення: підключення фонового модуля через поле `background.service_worker`, надання доступу до `tabs` і `storage`, а також прив’язку інтерфейсу спливаючого вікна до `action.default_popup`. Таке налаштування забезпечує подієву модель виконання фонового коду та доступ до API вкладок і сховища,

```
{
  "manifest_version": 3,
  "action": { "default_popup": "popup.html" },
  "permissions": ["tabs", "storage"],
  "background": {
    "service_worker": "background.js",
    "type": "module"
  }
}
```

Рисунок 4.1 – Програмний код файлу manifest.json

Логіку фонового модуля реалізовано як оброблення подій вкладок. Для цього використовуються події зміни активної вкладки та оновлення стану вкладки (зокрема завершення завантаження сторінки), після чого виконується отримання URL і запуск процедури аналізу. Ключові обробники подій наведено на рис. 4.2.

```
chrome.tabs.onActivated.addListener(async ({ tabId }) => {
  const tab = await chrome.tabs.get(tabId);
  if (tab?.url) await analyzeIfPossible(tab.url);
});
chrome.tabs.onUpdated.addListener(async (_tabId, changeInfo, tab) => {
  if (changeInfo.status === "complete" && tab?.url) {
    await analyzeIfPossible(tab.url);
  }
});
```

Рисунок 4.2 – Фрагмент програмного коду файлу `background.ts` (обробники подій)

Перед надсиланням запиту до серверної частини виконується перевірка придатності URL для аналізу та виділення доменного імені. Перевірка (рис.4.3) обмежує аналіз ресурсами з протоколами http/https і відсікає локальні адреси, що не розглядаються у межах предметної області.

```
function isAnalyzableUrl(url: string): boolean {
  const u = new URL(url);
  if (u.protocol !== "http:" && u.protocol !== "https:") return false;
  if (u.hostname === "localhost" || u.hostname === "127.0.0.1") return false;
  return true;
}

function extractHostname(url: string): string {
  return new URL(url).hostname;
}
```

Рисунок 4.3 – Фрагмент програмного коду файлу `background.ts` (валідація URL та виділення домену)

Обмін із серверною частиною реалізовано асинхронним HTTP-запитом, у якому передається DTO запиту з доменним іменем, а у відповіді отримується DTO результату аналізу. Структури DTO описано в TypeScript, що фіксує формат обміну на рівні коду та узгоджує поля відповіді, які використовуються під час відображення в інтерфейсі розширення. Відповідний фрагмент типів наведено на рис. 4.4.

```
export interface AnalyzeDomainRequestDto { hostname: string; }

export interface AnalyzeDomainResponseDto {
  hostname: string;
  lastCheckedAt: string;
  riskScore: number;
  recommendations: string[];
  vulnerabilities: { cveId: string; cvssScore: number | null; summary: string }[];
}g
```

Рисунок 4.4 – Програмний коду файлу `dtos.ts`.

Для виконання запиту до серверного прикладного програмного інтерфейсу використано `fetch` із передаванням DTO у форматі JSON та перевіркою коду відповіді. У запиті задається метод `POST`, заголовок `Content-Type: application/json` і тіло запиту, сформоване шляхом серіалізації об’єкта DTO. У разі отримання статус-коду, що не належить до діапазону успішних відповідей, ініціюється помилка, щоб результат аналізу не зберігався у локальному сховищі як коректний. Фрагмент реалізації запиту наведено на рис. 4.5.

```
async function fetchAnalysis(dto: AnalyzeDomainRequestDto) {
  const res = await fetch(`${API_BASE_URL}/api/analysis/analyze-domain`, {
    method: "POST",
    headers: { "Content-Type": "application/json" },
    body: JSON.stringify(dto),
  });
  if (!res.ok) throw new Error(String(res.status));
  return (await res.json()) as AnalyzeDomainResponseDto;
}
```

Рисунок 4.5 – Фрагмент програмного коду файлу `background.ts`
(HTTP запит до серверної частини)

Отриманий результат аналізу зберігається у локальному сховищі розширення через `chrome.storage.local`. Це узгоджується з моделлю Manifest V3,

де фоновий service worker працює подієво, а стан, потрібний для інтерфейсу, доцільно фіксувати у персистентному сховищі. Фрагмент збереження результату наведено на рис. 4.6.

```
await chrome.storage.local.set({
  lastHostname: hostname,
  lastReport: report
});
```

Рисунок 4.6 – Фрагмент програмного коду файлу background.ts (збереження результату у chrome.storage.local)

Інтерфейс спливаючого вікна (popup) відкривається через налаштування default_popup у manifest.json і активується лише у момент взаємодії користувача веббраузера з піктограмою розширення. Під час ініціалізації popup виконується читання останнього збереженого звіту зі сховища та запуск відображення. Приклад ініціалізації наведено на рис. 4.7.

```
document.addEventListener("DOMContentLoaded", async () => {
  const { lastReport } = await chrome.storage.local.get(["lastReport"]);
  if (lastReport) render(lastReport);
  else showEmptyState();
});
```

Рисунок 4.7 – Фрагмент програмного коду файлу popup.ts (читання зі сховища та запуск відображення)

Відображення результатів у спливаючому вікні організовано через функції рендерингу, при цьому пріоритет надається числовому показнику ризику та рекомендаціям, а відомості про уразливості відображаються як додатковий інформаційний блок. Фрагмент рендерингу ключових полів наведено на рис. 4.8.

```
function render(report: AnalyzeDomainResponseDto) {
  setText("riskScore", String(report.riskScore));
  setText("hostname", report.hostname);

  renderRecommendations(report.recommendations);
  renderVulnerabilities(report.vulnerabilities);
}
```

Рисунок 4.8 – Фрагмент програмного коду файлу popup.ts (рендеринг ключових полів)

4.2 Інтерфейс користувача браузерного розширення

Інтерфейс браузерного розширення (рис. 4.9) побудовано у вигляді спливаючого вікна, яке відкривається після натискання на значок розширення у панелі веббраузера. Вікно відображається поверх активної сторінки, що дає змогу користувачеві веббраузера оцінити ситуацію щодо безпеки, не перериваючи поточну роботу з сайтом. Структура інтерфейсу орієнтована на послідовне подання інформації від загального підсумку до детальних технічних даних, щоб користувач веббраузера спочатку бачив узагальнену оцінку ризику, а вже потім, знайомився з переліком виявлених уразливостей.

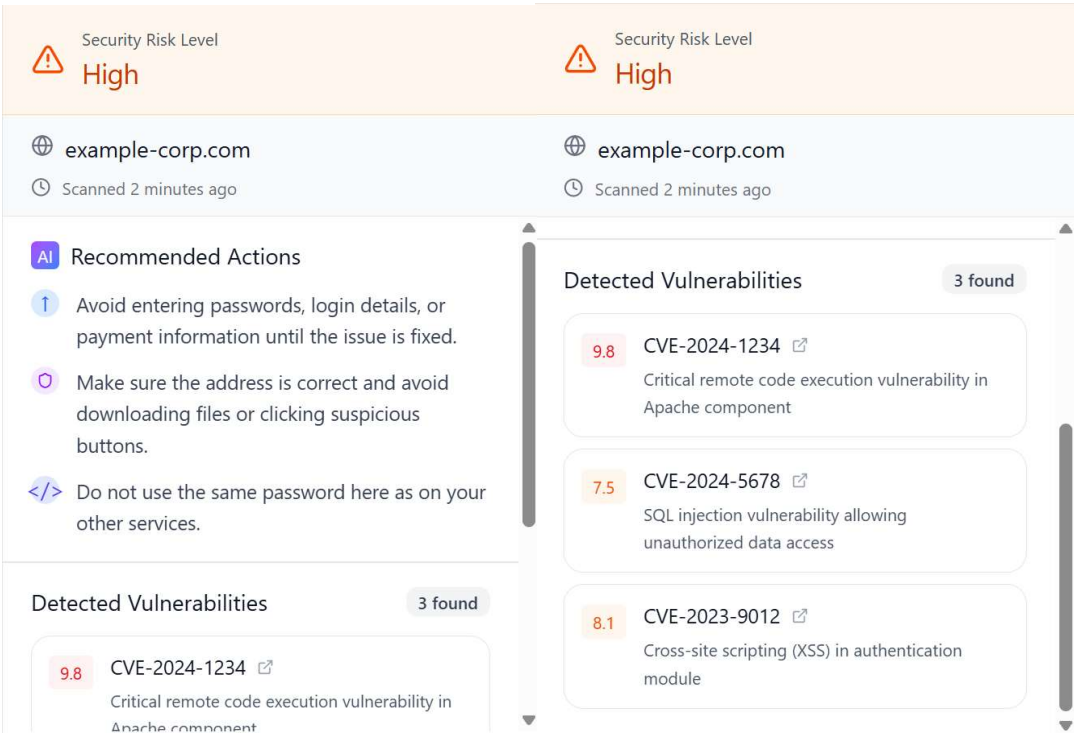

Рисунок 4.9 – Скріншоти інтерфейсу браузерного розширення

У верхній частині вікна розміщується заголовок з текстом, який пояснює, що саме відображається, та індикатор рівня ризику для поточного домену. Оцінка оформлена у вигляді словесної категорії (наприклад, High) із візуальним виділенням кольором та піктограмою попередження. Поруч розташовується доменне ім’я вебресурсу, яке було проаналізовано, а нижче вказується час

формування поточного звіту. Така комбінація дає змогу одразу співвіднести оцінку ризику з конкретним вебресурсом і зрозуміти, наскільки свіжими є показані результати.

Під блоком з оцінкою ризику розміщується область із рекомендаціями для користувача веббраузера. У цій області в узагальненій формі подано кілька текстових порад, які були сформовані сервісом великої мовної моделі на основі результатів аналізу домену та виявлених уразливостей. Кожна рекомендація відображається окремим абзацом із невеликою піктограмою, яка вказує на тип поради, наприклад обмеження введення конфіденційних даних, перевірка коректності адреси вебресурсу або застереження щодо повторного використання паролів. Текст рекомендацій сформульовано без використання спеціалізованих термінів, що дозволяє користувачеві веббраузера застосовувати їх на практиці без додаткового тлумачення технічних деталей.

Нижня частина вікна призначена для відображення переліку виявлених уразливостей. Над списком розташовано заголовок, який позначає, що показано саме результати автоматичного аналізу, а також компактна позначка із числом, що вказує кількість виявлених записів. Кожна уразливість відображається у вигляді окремого блоку, який містить значення числового показника з методики Common Vulnerability Scoring System, ідентифікатор з переліку Common Vulnerabilities and Exposures та короткий текстовий опис. Значення показника CVSS відображається у невеликому виділеному полі, розташованому ліворуч, що дозволяє швидко оцінити відносну небезпечність конкретної уразливості. Поруч із ідентифікатором CVE розміщується піктограма посилання, натискання на яку відкриває сторінку з детальним описом відповідного запису у публічному джерелі.

Список уразливостей відображається у прокручуваній області, тому у разі великої кількості записів користувач веббраузера має змогу поступово переглядати їх без зміни структури верхньої частини вікна, де містяться оцінка ризику та рекомендації. Така організація дозволяє сприймати інформацію у декількох рівнях деталізації: спочатку користувач бачить лише загальний рівень ризику, далі – текстові поради, а вже потім, за потреби, може ознайомитися з

окремими уразливостями, які призвели до такої оцінки. Спливаюче вікно не передбачає редагування даних або будь-яких дій, що впливають на серверну частину інформаційної системи; його роль полягає у відображенні результатів аналізу в формі, придатній для користувача веббраузера, який не працює безпосередньо з переліком уразливостей або базою National Vulnerability Database.

4.3 Реалізація серверної частини інформаційної системи

Архітектура серверної частини інформаційної системи побудована у вигляді вебзастосунку з прикладним програмним інтерфейсом, реалізованим за принципами RESTful API. Такий підхід передбачає чітке розділення відповідальності між клієнтською та серверною частинами, використання стандартизованих HTTP-методів для обміну даними та передавання результатів аналізу у структурованому форматі. Серверна логіка зосереджена в межах одного прикладного застосунку, що реалізує функції координації процесу аналізу домену та взаємодії із зовнішніми джерелами інформації, без поділу на окремі мікросервіси, що спрощує керування станом і узгодження результатів.

Реалізація серверної частини виконана з використанням платформи Node.js та фреймворку NestJS, який забезпечує модульну структуру застосунку, підтримку інверсії керування та чітку організацію прикладних компонентів. Для доступу до даних використовується об’єктно-реляційне відображення, що забезпечує абстракцію роботи з базою даних і спрощує реалізацію логіки збереження та оновлення доменів, уразливостей і пов’язаних попереджень. Обмін даними між клієнтською та серверною частинами здійснюється з використанням об’єктів передавання даних (Data Transfer Object, DTO), що дозволяє формалізувати контракт прикладного програмного інтерфейсу та відокремити внутрішню модель даних від зовнішнього представлення.

Структурно серверна частина поділяється на кілька логічних рівнів. Рівень контролерів відповідає за приймання HTTP-запитів від браузерного розширення та ініціювання відповідних сценаріїв оброблення. Рівень прикладної логіки

реалізує координацію процесів аналізу домену, перевірку актуальності даних, взаємодію з сервісами визначення технологій, синхронізації уразливостей і формування рекомендацій. Рівень доступу до даних забезпечує роботу з постійним сховищем, зокрема збереження доменів, уразливостей і попереджень, а також перевірку строків їх актуальності. Такий поділ спрощує супровід коду, локалізує зміни та забезпечує узгодженість реалізації основного сценарію аналізу.

Основний сценарій оброблення запиту (рис. 4.10) реалізовано в `AnalysisService`, де виконується отримання об’єкта домену, вибірка пов’язаних попереджень із таблиці `vulnerability_alert` та перевірка їх актуальності за полем `expires_at`. Якщо для домену наявні попередження, строк дії яких не завершився, результат формується на їх основі без повторного визначення технологій та без повторної синхронізації уразливостей.

```
  async analyzeDomain(hostname: string): Promise<AnalyzeDomainResponseDto> {
    const domain = await this.domainService.getOrCreateDomain(hostname);

    const now = new Date();
    const cachedAlerts = await
this.vulnerabilityService.getAlertsByDomain(domain.domainId);

    const validAlerts = cachedAlerts.filter((a) => a.expiresAt > now);
    const alerts = validAlerts.length > 0 ? validAlerts : await
this.refreshAlerts(domain, now);

    const report = await this.llmService.createDomainReport(domain, alerts, now);
    const recommendations = await
this.llmService.getRecommendations(report.domainAnalysisReportId);

    await this.domainService.updateLastAnalyzed(domain.domainId, now);

    return this.mapToResponse(domain, report, recommendations, alerts);
  }

  private async refreshAlerts(domain: DomainEntity, now: Date):
Promise<VulnerabilityAlertEntity[]> {
    const technologies = await
this.technologyDetectionService.detectTechnologies(domain);
    const vulnerabilities = await this.nvdSyncService.syncForTechnologies(technologies);

    return this.vulnerabilityService.updateAlerts(domain, vulnerabilities, now);
  }
```

Рисунок 4.10 – Фрагмент програмного коду `AnalysisService` (координація аналізу домену)

Визначення технологій реалізовано в `TechnologyDetectionService`, який

формує перелік записів web_technology, пов’язаний із доменом через таблицю domain_web_technology. У реалізації спочатку виконується спроба використати вже збережені зв’язки домену з технологіями, що дозволяє повторно використовувати результати попередніх визначень без виконання додаткових мережевих запитів. Якщо зв’язків немає, виконується отримання вмісту сторінки та/або заголовків відповіді, застосування набору сигнатур і перетворення результату на сутності технологій із подальшим збереженням зв’язку домену з отриманим переліком. У межах цього кроку враховується можливість неповного завантаження сторінки або відмінностей у заголовках відповіді, тому правила детекції застосовуються до кількох джерел ознак (HTML та headers). Після збереження отриманих технологій у локальному сховищі вони можуть використовуватися як основа для подальшої кореляції з уразливостями через таблицю web_technology_vulnerability. Ключовий фрагмент реалізації наведено на рис. 4.11.

```
  async detectTechnologies(domain: DomainEntity): Promise<WebTechnologyEntity[]> {
    const hydratedDomain = await this.domainRepo.findOne({
      where: { domainId: domain.domainId },
      relations: { technologies: true },
    });

    if (!hydratedDomain) return [];

    if (hydratedDomain.technologies?.length) {
      return hydratedDomain.technologies;
    }

    const { data, headers } = await firstValueFrom(
      this.http.get(`https://${hydratedDomain.hostname}`, { validateStatus: () => true
}),
    );

    const detected = this.applySignatureRules(String(data ?? ""), headers ?? {});
    const persisted = await this.upsertTechnologies(detected);

    hydratedDomain.technologies = persisted;
    await this.domainRepo.save(hydratedDomain);

    return persisted;
  }
```

Рисунок 4.11 – Фрагмент програмного коду TechnologyDetectionService (визначення та збереження технологій)

Формування агрегованого результату аналізу та робота з рекомендаціями

реалізовані в LlmService (рис. 4.12) через таблиці domain_analysis_report і recommendation. Під час створення звіту фіксується generated_at і обчислений risk_score, після чого забезпечується збереження та повторне отримання рекомендацій, пов'язаних із конкретним звітом. Для рекомендацій зберігається текст у полі recommendation_text, а також версія моделі в полі llm_model_version, що дозволяє розмежовувати результати, отримані за різних конфігурацій генерації. У межах сервісу рекомендації формуються на основі контексту, який включає дані домену та актуальні попередження vulnerability_alert, що забезпечує прив'язку порад до конкретної ситуації вебресурсу.

```
  async createDomainReport(
    domain: DomainEntity,
    alerts: VulnerabilityAlertEntity[],
    generatedAt: Date,
  ): Promise<DomainAnalysisReportEntity> {
    const riskScore = this.computeRiskScore(alerts);

    const report = this.reportRepo.create({
      domainId: domain.domainId,
      generatedAt,
      riskScore,
    });

    return this.reportRepo.save(report);
  }

  async getRecommendations(domainAnalysisReportId: number):
Promise<RecommendationEntity[]> {
    return this.recRepo.find({ where: { domainAnalysisReportId } });
  }

  async generateAndSaveRecommendations(
    domainAnalysisReportId: number,
    prompt: string,
    llmModelVersion: string,
  ): Promise<RecommendationEntity[]> {
    const texts = await this.llmClient.generateRecommendations(prompt);
    const entities = texts.map((text) =>
      this.recRepo.create({
        domainAnalysisReportId,
        recommendationText: text,
        llmModelVersion,
        createdAt: new Date(),
      }),
    );
    return this.recRepo.save(entities);
  }
```

Рисунок 4.12 – Фрагмент програмного коду LlmService
(формування звіту та збереження рекомендацій)

4.4 Реалізація бази даних та фізична модель даних

У серверній частині інформаційної системи кореляції доменів з базами вразливостей дані зберігаються в реляційній базі даних під керуванням системи PostgreSQL. Така система керування базами даних сумісна з використаною бібліотекою об’єктно-реляційного відображення TypeORM у межах фреймворку NestJS і підтримує транзакційну модель з властивостями ACID, зовнішні ключі, обмеження цілісності та індекси. Це дає змогу безпосередньо у схему бази даних закладати ті зв’язки між сутностями, які використовуються серверною частиною під час аналізу доменів, технологій та уразливостей.

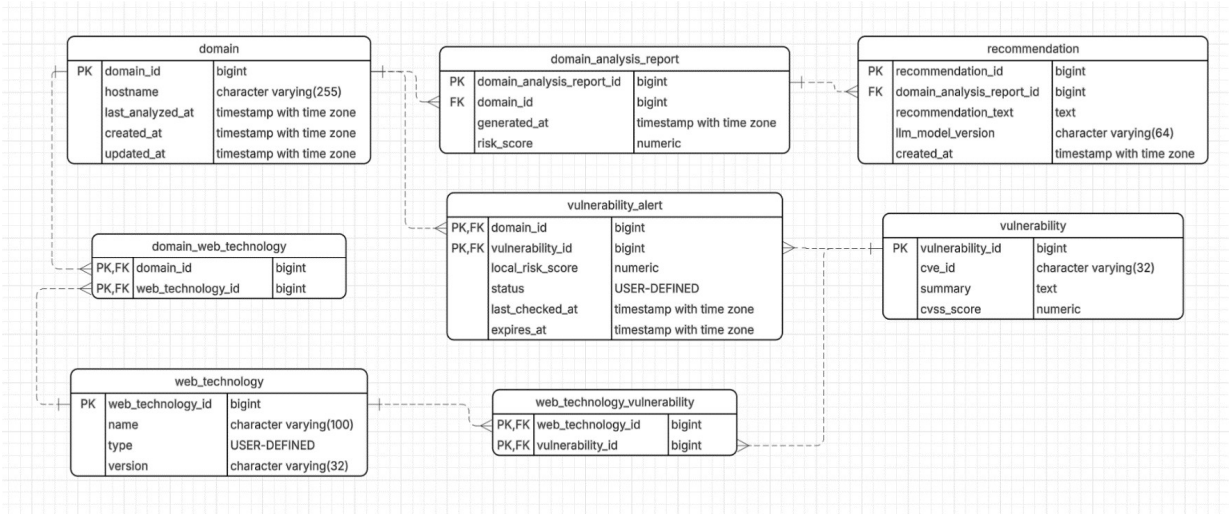

Рисунок 4.5 – EER-діаграма бази даних

Фізична структура бази даних подана у вигляді розширеної сутнісно-зв’язної EER-діаграми(рисунок 4.5). Діаграма відображає фактичну схему, що використовується серверною частиною інформаційної системи. Центральним елементом є таблиця domain, у якій зберігаються записи про доменні імена вебресурсів. Кожен запис містить внутрішній цілочисельний ідентифікатор, доменне ім’я у вигляді унікального рядкового поля, а також часові атрибути, пов’язані з моментом створення, оновлення та останнього аналізу домену. Для поля з доменним ім’ям використано індекс, що забезпечує швидкий пошук за назвою.

Технологічний профіль вебресурсу зберігається в таблиці `web_technology`. У цій таблиці містяться назви технологій, типи технологій та, за наявності, версії. Багатозначний зв'язок між доменами та технологіями реалізовано через таблицю `domain_web_technology`, яка містить пари ідентифікаторів `domain_id` і `web_technology_id`, а також службові часові поля, пов'язані з моментом виявлення технологій та строком чинності результатів. Для полів з ідентифікаторами домену й технології створено індекси, оскільки ці поля активно використовуються в запитах під час побудови списку технологій для певного домену та під час пошуку доменів, що використовують конкретну технологію.

Відомості про уразливості, отримані з національної бази вразливостей, зберігаються в таблиці `vulnerability`. Для кожної уразливості використовується внутрішній числовий ідентифікатор, а глобальний ідентифікатор з переліку CVE зберігається в окремому унікальному полі, за яким визначено індекс. У таблиці також зберігається числовий показник за методикою CVSS, текстові описи та додаткові атрибути, необхідні для аналізу. Зв'язок між технологіями та уразливостями реалізовано таблицею `web_technology_vulnerability`, у якій фіксуються пари `web_technology_id` та `vulnerability_id`. Така структура дає змогу прив'язувати одну уразливість до кількох технологій і, навпаки, будувати перелік уразливостей для конкретної технології на основі синхронізованих даних з бази NVD.

Підсумковий стан аналізу для кожного домену описується в таблиці `domain_analysis_report`. У цій таблиці зберігається посилання на домен, глобальна числова оцінка ризику, рівень ризику у категорійній шкалі, часові мітки створення та оновлення звіту, а також технічні метадані, пов'язані з викликом сервісу великої мовної моделі. Таблиця `recommendation` містить текстові рекомендації, згенеровані для певного звіту: кожен запис пов'язано із `domain_analysis_report` за допомогою зовнішнього ключа, зберігаються код рекомендації, коротка назва, текстовий опис, пріоритет і, за потреби, перелік ідентифікаторів CVE, до яких відноситься порада.

Результати кореляції між доменами та уразливостями з боку локального

сховища відображаються в таблиці `vulnerability_alert`. У цій таблиці кожен запис пов’язує конкретний домен із конкретною уразливістю через зовнішні ключі на domain та `vulnerability`, містить локальний числовий показник ризику, а також часові поля `last_checked_at` і `expires_at`. Значення цих полів використовуються серверною частиною для визначення того, коли результати попереднього аналізу можуть бути використані повторно, а коли виникає потреба в оновленні інформації про уразливості. Для поля `domain_id` та поля `expires_at` створено індекси, оскільки саме за ними найчастіше виконується фільтрація під час вибору актуальних записів для конкретного домену.

Схема бази даних організована таким чином, щоб атрибути, пов’язані з доменами, технологіями, уразливостями, звітами та рекомендаціями, зберігалися в окремих таблицях, а багатозначні зв’язки реалізовувалися через проміжні таблиці типу `domain_web_technology` та `web_technology_vulnerability`. Це дозволяє уникати дублювання інформації та підтримувати цілісність даних при оновленні записів. EER-діаграма фізичної моделі бази даних відображає фактичну структуру схеми, яку використовує серверна частина інформаційної системи кореляції доменів з базами вразливостей, і узгоджується зі структурами класів-сутностей TypeORM, застосованих під час реалізації.

5 ОЦІНКА ЯКОСТІ ТА ЕФЕКТИВНОСТІ БРАУЗЕРНОГО РОЗШИРЕННЯ КОРЕЛЯЦІЇ ДОМЕНІВ З БАЗАМИ ВРАЗЛИВОСТЕЙ

5.1 Мета та методика оцінювання

Практична частина роботи спрямована на дві ключові перевірки. По-перше, оцінюється, наскільки інформаційна система аналізу безпеки вебресурсів коректно виконує кореляцію доменів з базами вразливостей, тобто чи виявляє вона ті технології та уразливості, які очікувано пов’язані з конкретним вебресурсом. По-друге, оцінюється формування інтегральної оцінки ризику: перевіряється, чи узгоджується глобальний показник ризику, сформований за допомогою сервісу великої мовної моделі, з наявними числовими оцінками за методикою CVSS для виявлених уразливостей.

Оцінювання виконується у контрольованому середовищі, де серверна частина розгорнута на основі фреймворку NestJS з використанням бібліотеки TypeORM та реляційної бази даних PostgreSQL. У цій базі зберігаються дані про домени, технології, уразливості та результати аналізу. Інформація про уразливості включає ідентифікатори CVE та бали CVSS, які використовуються системою як числовий орієнтир суворості. Взаємодія з сервісом великої мовної моделі здійснюється через локально розгорнутий інстанс ChatGPT, до якого серверна частина надсилає структурований промпт з переліком уразливостей для конкретного домену й отримує у відповідь JSON з глобальною оцінкою ризику та рекомендаціями.

Перша група перевірок присвячена коректності кореляції доменів з базами вразливостей. Для цього формується вибірка доменів із різними технологічними профілями. Для кожного домену окремо визначаються еталонні множини технологій і уразливостей за допомогою наявних інструментів визначення технологій та ручного аналізу записів CVE, пов’язаних з відповідними програмними продуктами. Після цього для тих самих доменів виконується аналіз за допомогою реалізованої інформаційної системи. Порівнюються множини технологій і уразливостей, отримані системою, з еталонними множинами;

фіксуються збіги, випадки пропущених технологій чи уразливостей, а також ситуації, у яких система визначає додаткові відповідності, не включені до ручного опису. Такий підхід дозволяє охарактеризувати якість реалізованого алгоритму кореляції «домен – технології – уразливості» з точки зору відтворення очікуваних відповідностей.

Друга група перевірок стосується оцінювання інтегральної оцінки ризику, що повертається сервісом великої мовної моделі. Для кожного домену на основі локально збережених уразливостей будується базовий показник ризику, який ґрунтується на числових балах CVSS: використовується сукупність балів для всіх уразливостей, пов’язаних з доменом, та їх якісна градація за рівнями небезпеки. Цей показник розглядається як орієнтир для інтерпретації значення `globalRiskScore` та рівня `riskLevel`, які модель повертає у JSON-відповіді. Порівнюються тенденції: чи отримує домен з кількома уразливостями з високими балами CVSS підвищену глобальну оцінку ризику, чи відповідають рівні `riskLevel` очікуваній шкалі на основі CVSS, чи не виникають суперечливі випадки, коли сукупність уразливостей з високими значеннями CVSS супроводжується низькою інтегральною оцінкою. Додатково аналізується зміст згенерованих рекомендацій, зокрема їх прив’язка до ідентифікаторів CVE та відповідність контексту користувача веббраузера.

Результати обох груп перевірок фіксуються у вигляді таблиць із порівнянням еталонних і одержаних множин технологій та уразливостей, а також зіставленням базових CVSS-орієнтирів із глобальними оцінками ризику, сформованими сервісом ChatGPT. Така методика дає змогу оцінити і коректність реалізованого алгоритму кореляції, і узгодженість інтегральної оцінки ризику з числовими показниками, на яких вона базується.

5.2 Перевірка коректності кореляції доменів з базою вразливостей

Для оцінювання коректності реалізованого алгоритму кореляції «домен – технології – уразливості» використовується демонстраційний майданчик

Vulnweb, який містить набір навмисно вразливих вебзастосунків. Ці вебресурси призначені для тестування інструментів аналізу безпеки та мають відкриту документацію щодо використовуваних серверних технологій. На основі цього майданчика формується тестова вибірка доменів, для яких можна задати еталонні множини технологій і надалі порівнювати їх з результатами, що повертає інформаційна система аналізу безпеки вебресурсів.

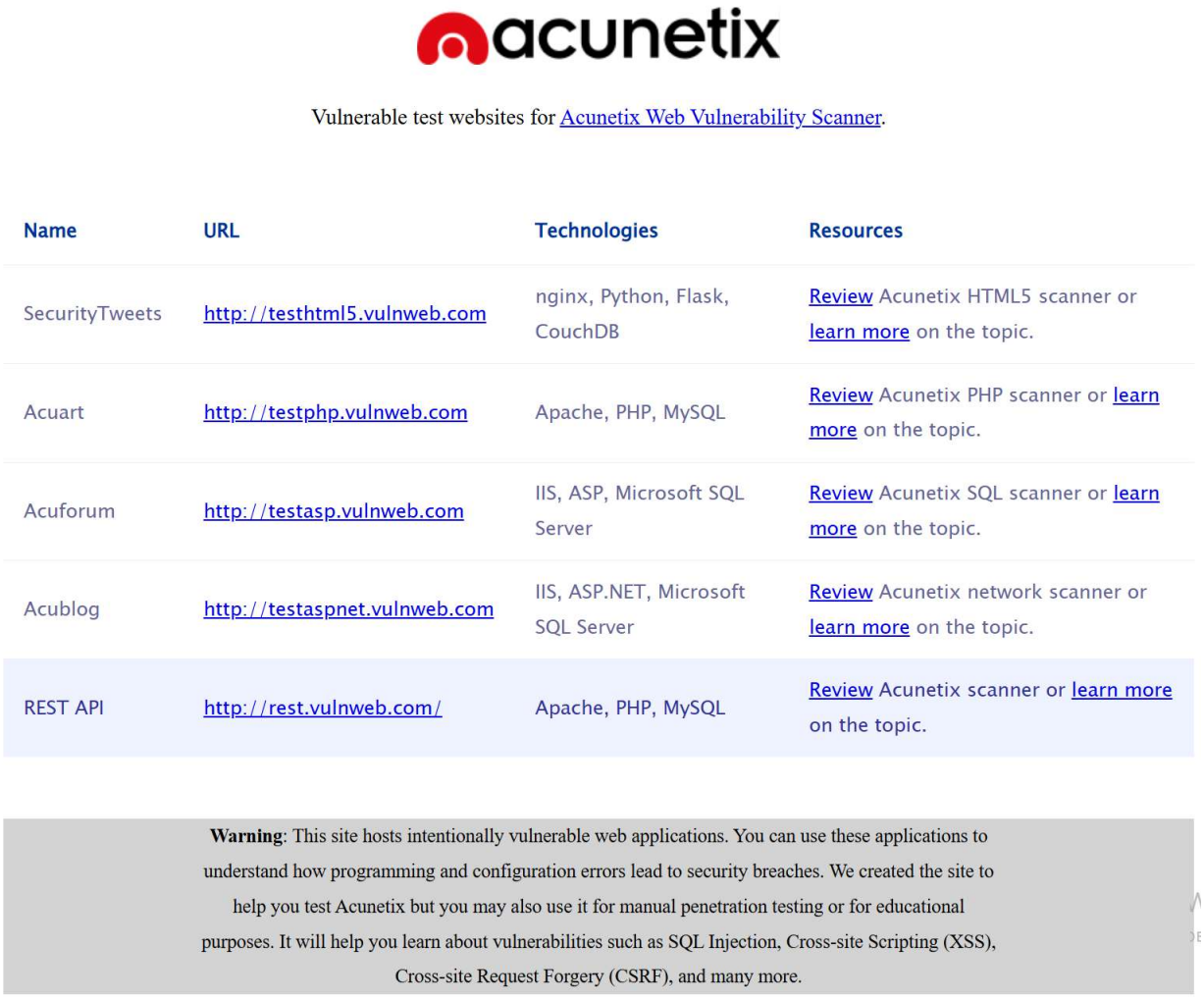

Рисунок 5.1 – Скріншот вебсторінки Vulnweb

Сторінки Vulnweb (рис. 5.1) містить перелік демонстраційних застосунків. Для кожного з них наведено назву, URL та короткий опис, який вказує на призначення конкретного тестового стенду. Кожен застосунок розгорнуто на окремому піддомені, тому в термінах розробленої інформаційної системи кожен

URL розглядається як окремий домен, що підлягає аналізу. Один вебресурс Vulnweb надає набір демонстраційних застосунків, доступних за різними піддоменами, що дозволяє сформувати вибірку доменів для аналізу.

У таблиці 5.1 зведено характеристики обраних демонстраційних застосунків Vulnweb. Для кожного запису подано назву стенду, його URL та основні серверні технології, задіяні під час обробки HTTP-запитів. Наведені характеристики використовуються як еталонні множини технологій для відповідних доменів і застосовуються під час формування еталонних наборів T∗(d)T^*(d)T∗(d) для порівняння з результатами сервісу визначення технологій.

Таблиця 5.1 – Таблиця демонстраційних сайтів і їх технологій

Назва	Домен	Технологія
SecurityTweets	http://testhtml5.vulnweb.com	nginx, Python, Flask, CouchDB
Acuart	http://testphp.vulnweb.com	Apache, PHP, MySQL
Acuforum	http://testasp.vulnweb.com	IIS, ASP, Microsoft SQL Server
Acublog	http://testaspnet.vulnweb.com	IIS, ASP.NET, Microsoft SQL Server
REST API	http://rest.vulnweb.com	Apache, PHP, MySQL

Для подальшого аналізі кожен із наведених у таблиці доменів обробляється браузерним розширенням. Для кожного домену формується множина технологій T(d), яку повертає сервіс TechnologyDetectionService, після чого ця множина порівнюється з еталонним набором T*(d), побудованим на основі даних Vulnweb. Такий підхід дозволяє оцінити, наскільки реалізований алгоритм виявлення технологій відтворює очікувані технологічні профілі навмисно вразливих вебзастосунків і чи можна використовувати отримані множини T(d) як надійну основу для подальшої кореляції з базою вразливостей.

Під час формування еталонних множин технологій для доменів використовуються зовнішні інструменти визначення технологічного стеку вебресурсів. Для цього залучено три незалежні сервіси класу technology profiler, які аналізують HTML-код сторінок, HTTP-заголовки та допоміжні ресурси й на основі вбудованих сигнатур формують перелік виявлених технологій.

Сервіс Wappalyzer використовується як перше джерело еталонних даних. Цей інструмент надає браузерне розширення та вебінтерфейс, які дають змогу для

довільного домену отримати перелік виявлених систем керування вмістом, вебфреймворків, вебсерверів, бібліотек JavaScript, систем аналітики, платіжних сервісів та інших компонентів технологічного стеку. Результати Wappalyzer подаються у вигляді структурованого списку технологій із розподілом за категоріями, що дозволяє виділити саме серверні платформи й прикладні фреймворки, які є релевантними для побудови множин T*(d) у контексті аналізу вразливостей.

Другим інструментом є сервіс BuiltWith, який надає вебінтерфейс і розширення для браузера та орієнтований на побудову технологічних профілів вебсайтів. Після введення URL сервіс повертає перелік виявлених технологій із групуванням за категоріями, такими як вебсервер, система керування вмістом, платформа електронної комерції, аналітика, рекламні та маркетингові інструменти. Дані BuiltWith використовуються як незалежне джерело для перевірки результатів Wappalyzer. Якщо певна технологія для досліджуваного домену одночасно присутня в обох сервісах, така технологія включається до еталонного набору для цього домену.

Третім джерелом еталонної інформації є сервіс Ful.io, який також реалізує функції технологічного профайлера. Ful.io надає вебінтерфейс і окреме браузерне розширення для Technology Lookup, яке здійснює автоматичне сканування вебресурсу та формує перелік виявлених технологій, включно з платформами керування вмістом, фреймворками, службами аналітики та іншими компонентами. Результати Ful.io враховуються під час остаточного формування еталонної множини технологій для домену: якщо технологія підтверджується більшістю з використаних інструментів, вона розглядається як така, що входить до T*(d) для відповідного домену.

Використання трьох окремих сервісів визначення технологій дає змогу зменшити залежність від особливостей реалізації конкретного профайлера та сформувати еталонні технологічні профілі доменів на основі перехресної перевірки результатів. Надалі ці профілі порівнюються з множинами T(d), які повертає реалізований сервіс TechnologyDetectionService, що дозволяє кількісно

оцінити коректність роботи алгоритму визначення технологій у розробленій інформаційній системі аналізу безпеки вебресурсів.

Для ілюстрації практичної процедури формування еталонного технологічного профілю розглядається домен http://testphp.vulnweb.com з таблиці 5.1. Для цього домену послідовно виконано аналіз за допомогою сервісів Wappalyzer, BuiltWith та Ful.io, після чого результати кожного інструменту зіставляються між собою. На основі таких зіставлень виділяється узгоджена множина серверних технологій, яка надалі використовується як еталонний профіль T*(d) для цього вебресурсу.

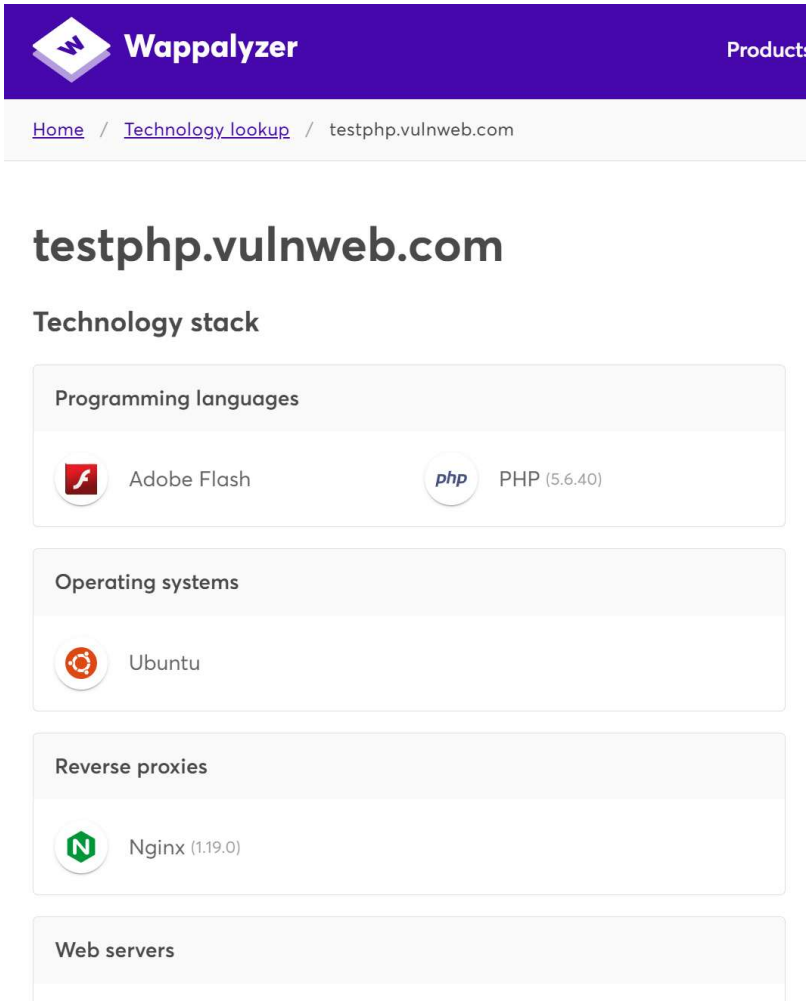

Рисунок 5.2 – Скріншот результатів аналізу домену за допомогою засоба Wappalyzer

На рисунку 5.2 наведено результат аналізу домену Acuart засобом

Wappalyzer. На скріншоті відображено перелік технологій, виявлених під час завантаження головної сторінки вебресурсу. Серед них окремо фіксуються вебсервер Nginx, мова виконання сценаріїв на боці сервера PHP та система керування базами даних MySQL.

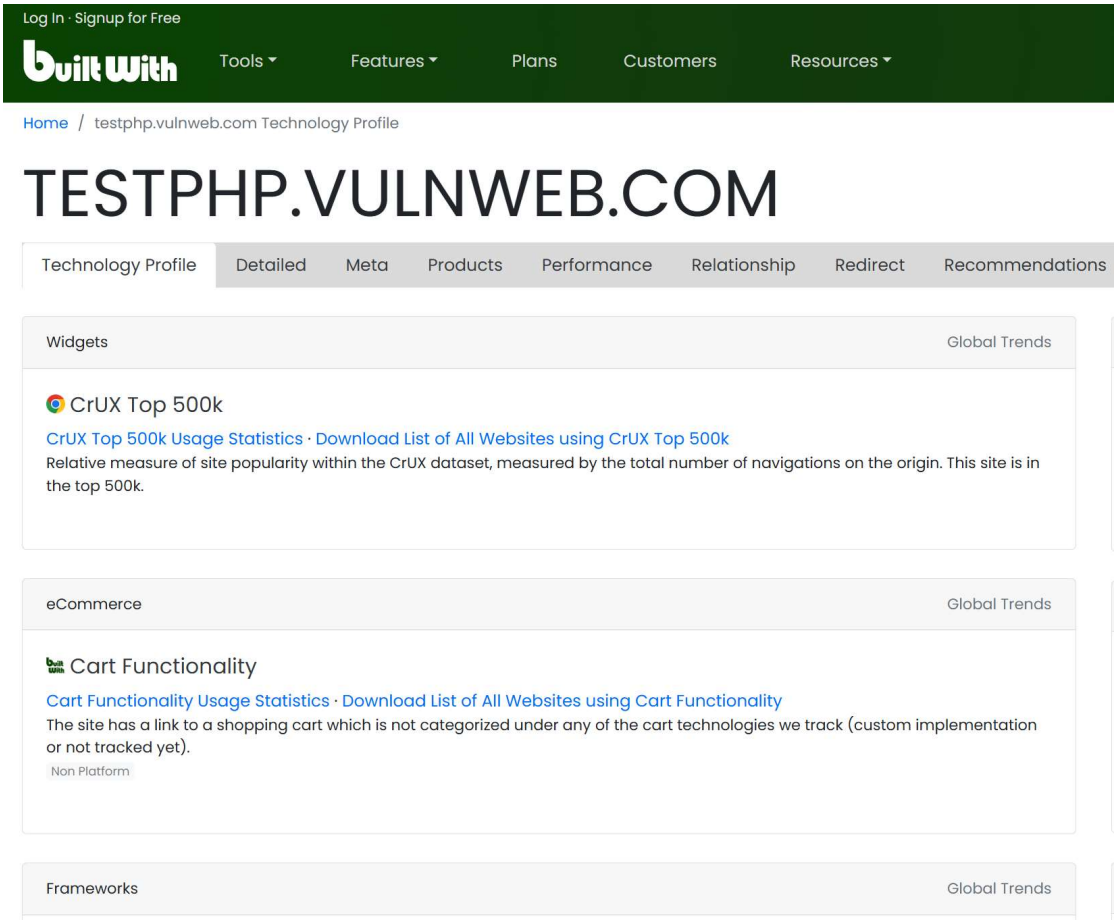

Рисунок 5.3 – Скріншот аналізу домен за допомогою сервіса BuiltWith

Аналогічний аналіз домену Acuart виконано за допомогою сервісу BuiltWith. На рисунку 5.3 наведено фрагмент звітної сторінки BuiltWith для URL http://testphp.vulnweb.com. Для досліджуваного домену в цих категоріях так само фіксуються Apache, PHP та MySQL як ключові серверні компоненти. Це підтверджує результати, отримані від Wappalyzer, і свідчить про те, що обидва інструменти узгоджено ідентифікують технологічний стек цього стенду.

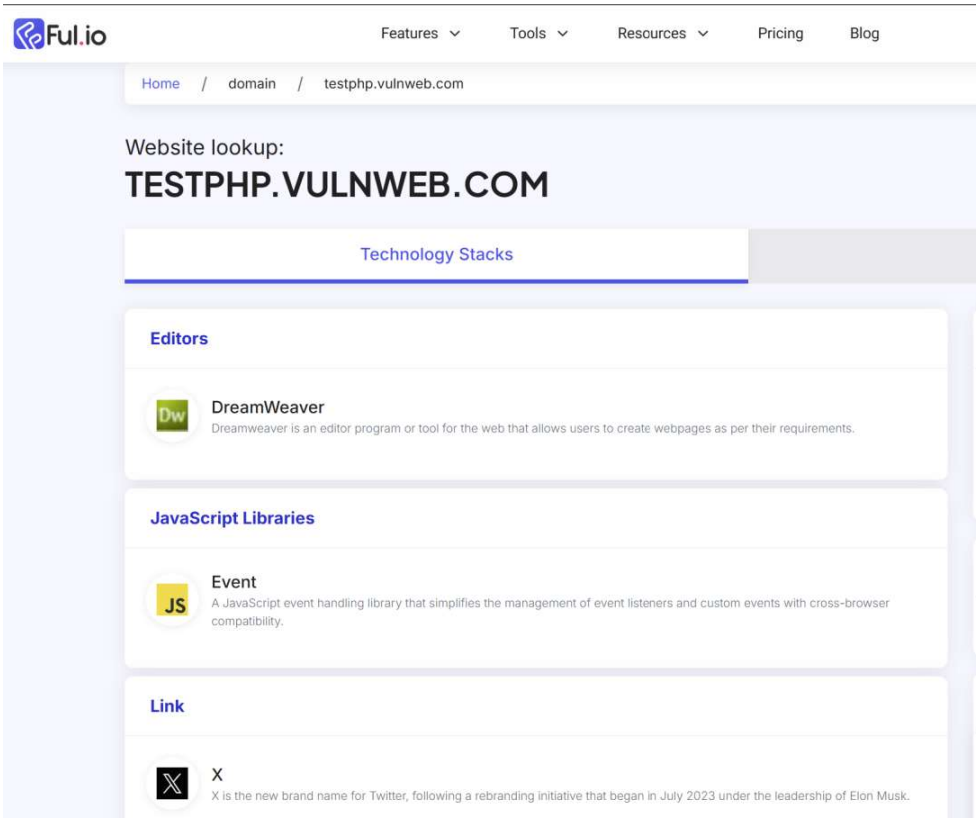

Рисунок 5.4 – Скріншот аналізу домен за допомогою засоба Ful.io

Третій крок передбачає аналіз того самого домену засобом Ful.io. На рисунку 5.4 наведено інтерфейс результатів сканування домену Acuart у сервісі Ful.io з переліком виявлених технологій. У цьому переліку знову присутні Apache, PHP і MySQL як основні серверні компоненти, які використовуються для обробки HTTP-запитів. Як і у випадку інших інструментів, Ful.io може додатково виявляти допоміжні служби або клієнтські бібліотеки, проте у даному дослідженні для побудови еталонного профілю враховуються лише серверні технології, що мають прямий зв’язок із записами у базі вразливостей.

Після аналізу домену Acuart за допомогою трьох інструментів аналогічна процедура послідовно виконувалася для решти демонстраційних застосунків сторінки Vulnweb. Для кожного з п’яти доменів фіксувалися серверні технології, зазначені на сторінці Vulnweb, та результати, отримані від Wappalyzer, BuiltWith і Ful.io. На основі цих даних для кожного домену окремо визначався узгоджений технологічний профіль, до якого включалися лише ті компоненти, які одночасно підтверджувалися більшістю використаних інструментів і відповідали опису на

майданчику Vulnweb.

Узагальнені результати такого порівняння наведено в таблиці 5.2. У таблиці для кожного домену подано вихідний опис серверних технологій за даними Vulnweb, результати трьох зовнішніх профайлерів та остаточно обраний еталонний технологічний профіль, який позначається як T*(d) і надалі використовується як орієнтир при оцінюванні роботи реалізованого сервісу визначення технологій.

Таблиця 5.2 – Порівняння результатів визначення технологій для доменів сторінки Vulnweb

Доменне ім'я	Vulnweb	Wappalyzer	BuiltWith	Ful.io	Еталонний профіль T*(d)
testhtml5.vulnweb	nginx, Python, Flask, CouchDB	nginx, Flask, CouchDB jQuery	nginx, Python, Flask, jQuery, Bootstrap	nginx, Python, Flask, CouchDB, jQuery	nginx, Python, Flask, jQuery CouchDB
testphp.vulnweb	Apache, PHP, MySQL	nginx, PHP, MySQL, jQuery, Bootstrap	Apache, PHP, jQuery, Bootstrap	Apache, PHP, MySQL, jQuery	Apache, PHP, MySQL, jQuery, Bootstrap
testasp.vulnweb	IIS, ASP, Microsoft SQL Server	IIS, ASP, jQuery	IIS, ASP, Microsoft SQL Server, jQuery	IIS, Microsoft SQL Server	IIS, ASP, Microsoft SQL Server, jQuery
testaspnet.vulnweb	IIS, ASP.NET, Microsoft SQL Server	IIS, ASP.NET, jQuery	IIS, ASP.NET, Microsoft SQL Server, jQuery, Bootstrap	ASP.NET, Microsoft SQL Server, jQuery	IIS, ASP.NET, Microsoft SQL Server, jQuery
rest.vulnweb	Apache, PHP, MySQL	Apache, PHP, MySQL	Apache, PHP, MySQL, jQuery	Apache, PHP, MySQL, Laravel, Axios	Apache, PHP, MySQL

У таблиці 5.2 наведено зіставлення технологій, зазначених у описах демонстраційних доменів з вебзастосунку Vulnweb, з результатами їх визначення зовнішніми інструментами. Для окремих доменів зафіксовано розбіжності між результатами різних інструментів, зокрема в частині серверних компонентів та веббібліотек, які відображаються у звітах. У деяких звітах відсутні окремі технології, згадані в описі доменів (наприклад, Microsoft SQL Server або MySQL),

водночас у результатах можуть з’являтися додаткові визначення (наприклад, Laravel, jQuery або Bootstrap), які не наведені на сторінці Vulnweb.

Еталонний профіль T*(d) для кожного домену формується як узгоджений перелік технологій, до якого включаються лише ті компоненти, що визначаються принаймні двома з трьох інструментів і не суперечать задекларованому опису. Надалі саме ці еталонні профілі використовуються як орієнтир для порівняння з множинами технологій T(d), які повертає реалізований сервіс TechnologyDetectionService.

Після формування еталонних технологічних профілів T*(d) для кожного вибраного домену Vulnweb було виконано аналіз цих доменів за допомогою розробленого браузерного розширення та серверного прикладного програмного інтерфейсу. Отримані результати визначення технологій було зіставлено з еталонними профілями T*(d) для подальшої оцінки відтворення зазначених технологій.

Браузерне розширення надсилало запит аналізу домену до серверної частини, де сервіс TechnologyDetectionService завантажував HTML-вміст і HTTP-заголовки, застосовував локальний файл сигнатур на основі та формував множину технологій T(d), яка зберігалася у таблицях web_technology і domain_web_technology. Для кожного домену ця множина порівнювалася з еталонною T*(d), що дало змогу кількісно оцінити, наскільки реалізований алгоритм визначення технологій відтворює очікуваний технологічний профіль.

Узагальнені результати порівняння наведено в таблиці 5.3. Для кожного URL подано еталонну множину |T*(d)|, кількість технологій |T(d)|, виявлених сервісом TechnologyDetectionService, кількість збігів |T∩| між T*(d) та T(d), а також відносне покриття еталону, обчислене як частка збігів відносно кількості еталонних технологій. Також описані виявлені технології |T(d)| , які були отримані за допомогою аналізу домена інформаційною системою для кореляції доменів з базами вразливостей.

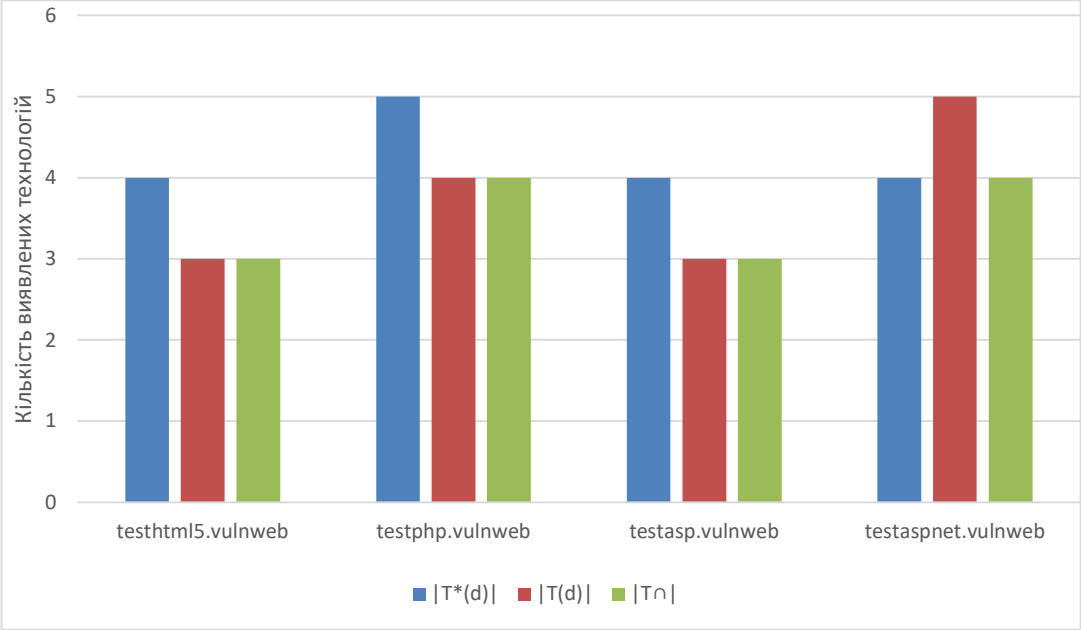

Рисунок 5.1 – Графік порівняння еталонних профілів T*(d) з результатами сервісу TechnologyDetectionService

Результати порівняння (рис. 5.1) доводять, що для всіх доменів сервіс TechnologyDetectionService відтворює основні серверні та клієнтські технології, включені до еталонних профілів T*(d). Для доменів testaspnet.vulnweb та rest.vulnweb.com/ покриття еталонних технологій становить 100 %, при цьому множини T(d) додатково містять фронтенд-компоненти на кшталт jQuery, Bootstrap та фреймворку Laravel. Для решти доменів покриття еталонних технологій становить 75–80 %: множини T(d) містять усі базові серверні компоненти, від яких залежить побудова кореляції з базою уразливостей, але можуть не включати окремі технології, що складніше ідентифікуються на основі сигнатур. У сукупності це свідчить про те, що отримані множини T(d) є достатньо наближеними до еталонних T*(d) для використання як вхідні дані під час пошуку уразливостей та подальшої оцінки ризику.

Для побудови еталонних множин уразливостей V*(d) для тестових доменів спочатку використовується еталонний технологічний профіль T*(d), сформований на основі майданчика Vulnweb та результатів трьох зовнішніх профайлерів. Для кожної технології з T*(d) виконується пошук записів у переліку відомих уразливостей Common Vulnerabilities and Exposures (CVE) в базі National

Vulnerability Database, а також у пов’язаних оглядах вендорів і спеціалізованих звітах з безпеки. Пошук обмежується сучасними версіями програмного забезпечення й часовим інтервалом останніх років експлуатації компонентів (орієнтовно з 2023 року), щоб уникати застарілих проблем, які вже давно втратили практичну актуальність.

Для кожної технології з T*(d) (наприклад, Apache HTTP Server, PHP, MySQL, nginx, Apache CouchDB, фреймворки ASP.NET і Flask, система керування базами даних Microsoft SQL Server, а також клієнтські бібліотеки на кшталт jQuery) відбирається характерні уразливостей з каталогу CVE. Перевага надається записам, для яких у базі NVD або у вендорних бюлетенях опубліковано повні вектори CVSS версії 3.1 і чітко описаний вплив: віддалене виконання коду, розкриття конфіденційних даних, відмова в обслуговуванні чи збережені XSS-атаки. Таким чином формується невелика, але репрезентативна підмножина уразливостей, яка використовується саме як еталон для експериментів, а не як повний перелік усіх проблем безпеки для відповідної платформи.

Для серверних компонентів вебстека (Apache HTTP Server, nginx, PHP, MySQL, Apache CouchDB, Microsoft SQL Server, ASP.NET Core) відбираються уразливості, що стосуються обробки HTTP-запитів, виконання серверного коду та доступу до даних. Наприклад, для Apache HTTP Server враховуються CVE-записи серії 2024-3847x, пов’язані з модулем mod_rewrite та обробкою UNC-шляхів, які можуть призводити до SSRF-сценаріїв і віддаленого виконання коду на вебсервері. Для PHP та PHP-CGI відображаються сучасні уразливості на кшталт CVE-2024-4577, де некоректна обробка параметрів у CGI-режимі відкриває можливість виконання довільного коду через HTTP-запити. Для Flask-екосистеми розглядається, зокрема, CVE-2024-25128 у Flask-AppBuilder, що пов’язаний з помилками автентифікації через OpenID та може надавати несанкціонований доступ до адміністративних інтерфейсів вебзастосунку.

Для систем керування базами даних MySQL Server та Microsoft SQL Server використовуються уразливості, які безпосередньо впливають на доступність сервісу або цілісність даних. Прикладами є CVE-2024-20971 для MySQL Server, де

спеціально сформовані запити можуть призводити до відмови в обслуговуванні екземпляра сервера, та CVE-2023-21713 для Microsoft SQL Server, що описує сценарій віддаленого виконання коду при обробці певних вхідних даних. Додатково враховується CVE-2023-36728 як приклад уразливості, що призводить до виходу за межі буфера й падіння процесу.

У таблиці 5.3 наведено приклади уразливостей, відібраних для основних технологій, що входять до еталонних технологічних профілів T*(d) демонстраційних доменів Vulnweb. Для кожної технології подано один або кілька CVE-ідентифікаторів, рік публікації, базову оцінку за методикою Common Vulnerability Scoring System (CVSS) версії 3.1, класифікацію рівня небезпеки та короткий опис впливу. Добір прикладів виконано таким чином, щоб у таблиці були представлені різні типи наслідків уразливостей, зокрема порушення конфіденційності, цілісності та доступності. Наведені записи використовуються як опорні дані під час зіставлення результатів кореляції уразливостей із доменами та під час інтерпретації значень інтегральної оцінки ризику.

Таблиця 5.3 – Узагальнені приклади вразливостей

Технологія	Ідентифікатор CVE	Рік	CVSS v3.1	Рівень небезпеки (CVSS)
Apache HTTP Server	CVE-2024-38475	2024	9.1	Critical
Apache HTTP Server	CVE-2024-38472	2024	7.5	High
PHP	CVE-2024-4577	2024	9.8	Critical
MySQL Server	CVE-2024-20971	2024	4.9	Medium
nginx	CVE-2024-24989	2024	7.5	High
Apache CouchDB	CVE-2023-45725	2023	4.8	Medium
Flask / Flask-AppBuilder	CVE-2024-25128	2024	9.1	Critical
Microsoft SQL Server	CVE-2023-21713	2023	8.8	High
Microsoft SQL Server	CVE-2023-36728	2023	5.5	Medium
ASP.NET Core	CVE-2023-35391	2023	7.5	High

На основі таблиці 5.3 для кожної технології з еталонного профілю T*(d) визначається підмножина відповідних CVE-записів, які утворюють еталонну множину уразливостей V*(d). Якщо профіль певного домену містить, наприклад, Apache HTTP Server, PHP та MySQL, до V*(d) для цього домену включаються уразливості CVE, наведені для цих трьох компонентів. Важливо, що сама наявність технології у T*(d) не означає автоматичної вразливості домену: кожна уразливість застосовується лише за умови, що версія програмного забезпечення

потрапляє у вказаний в CVE діапазон. У рамках експериментів такий підхід дає змогу побудувати прозорий і відтворюваний еталонний набір V*(d), який надалі використовується для порівняння з множинами V(d), що формує реалізована інформаційна система аналізу безпеки вебресурсів.

Після формування еталонних множин уразливостей V*(d) для кожного з тестових доменів було виконано запуск реалізованої інформаційної системи кореляції доменів з базами вразливостей. Браузерне розширення послідовно надсилало запити аналізу для кожного URL майданчика Vulnweb, серверна частина проводила визначення технологій, виконувала синхронізацію з базою NVD та формувала множини уразливостей V(d), пов'язаних з відповідним доменом. Для кожного домену було зафіксовано кількість еталонних уразливостей |V*(d)|, кількість уразливостей |V(d)|, які система зберігає у локальній базі, а також кількість збігів |V∩| між еталонним переліком і множиною V(d). На основі цих величин обчислено відносне покриття еталонних уразливостей для кожного домену у відсотках.

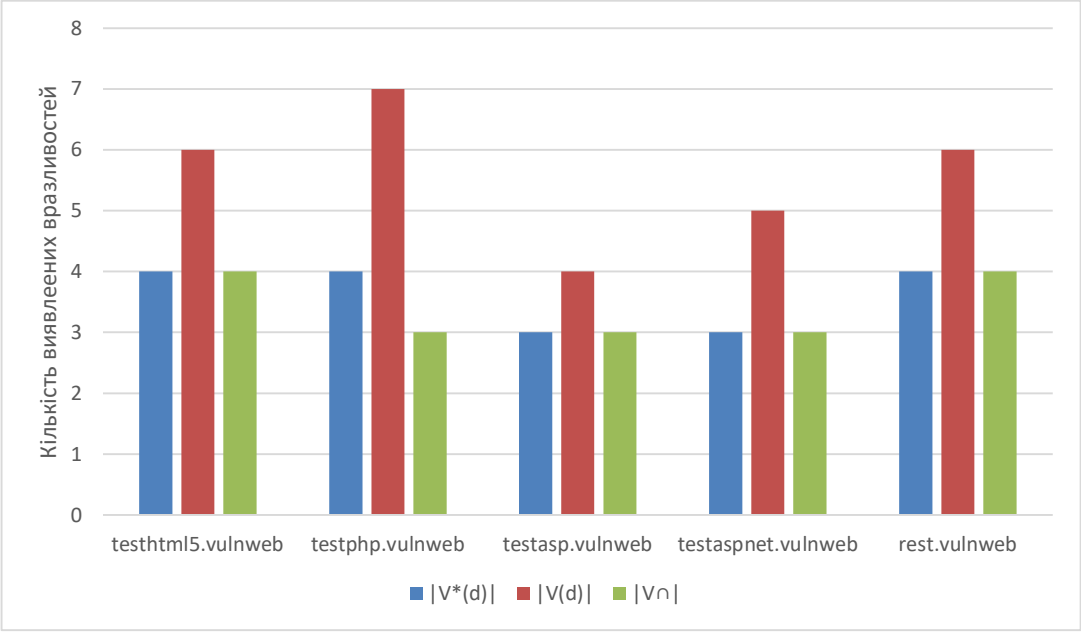

Рисунок 5.2 – Порівняння виявлених вразливостей

Результати порівняння (рис. 5.2) доводять, що для чотирьох з п'яти доменів множина V(d), яку формує інформаційна система, повністю містить усі еталонні

уразливості V*(d); при цьому загальна кількість елементів у V(d) є більшою, ніж у V*(d), оскільки система додатково враховує інші релевантні записи CVE для тих самих технологій. Для домену testphp.vulnweb покриття еталонних уразливостей становить 75 %, тобто три з чотирьох вручну відібраних CVE були відтворені автоматично, а одна еталонна уразливість не потрапила до V(d) через відмінності у виборі версій або часових меж синхронізації. Водночас для цього домену система зафіксувала сім уразливостей замість чотирьох, тобто перелік V(d) містить додаткові записи CVE для компонентів Apache, PHP або пов’язаних бібліотек, які не були включені до еталонного переліку, але є актуальними за даними бази NVD.

За підсумками порівняння технологічних профілів T*(d) і T(d) та множин уразливостей V*(d) і V(d) можна зробити висновок, що реалізований браузерний застосунок для тестових доменів Vulnweb коректно відтворює як структуру технологічного стеку, так і кореляцію технологій до записів у базі вразливостей. Множини T(d), сформовані сервісом TechnologyDetectionService, відрізняються від еталонних T*(d) лише окремими другорядними компонентами, а всі основні серверні технології, що визначають клас уразливостей, збігаються. Множини V(d), отримані в результаті роботи сервісу NvdSyncService, містять більшість або всі еталонні уразливості V*(d) і доповнюються додатковими CVE для тих самих продуктів. Така узгодженість дає підстави використовувати результати кореляції «домен – технології – уразливості» як надійну основу для подальшого етапу, пов’язаного з побудовою інтегральної оцінки ризику та формуванням рекомендацій за допомогою сервісу великої мовної моделі.

5.3 Оцінювання інтегральної оцінки ризику за результатами роботи LLM

Після перевірки коректності кореляції доменів з базою вразливостей виконується зіставлення оцінки ризику, сформованої сервісом великої мовної моделі, з числовими показниками CVSS для уразливостей, пов’язаних з кожним доменом. Для цього для множин уразливостей V(d), що були отримані під час

попередніх експериментів для п’яти доменів майданчика Vulnweb, обчислюється еталонний інтегральний показник ризику R*(d), а потім цей показник порівнюється з глобальною оцінкою ризику globalRiskScore, яку повертає локально розгорнутий інстанс моделі типу ChatGPT.

Еталонний показник R*(d) визначається на основі базових оцінок CVSS для уразливостей із множини V(d) та їх належності до серверних або клієнтських технологій. Для кожного домену враховуються, з одного боку, максимальні й середні значення CVSS для уразливостей, пов’язаних із серверними компонентами на кшталт Apache HTTP Server, nginx, PHP, MySQL, Microsoft SQL Server або ASP.NET, а з іншого боку, оцінки для уразливостей фронтенд-бібліотек і плагінів, зокрема компонентів jQuery-екосистеми. Отримані значення нормуються на інтервал від 0 до 10, унаслідок чого R*(d) можна інтерпретувати як еталонну інтегральну оцінку ризику, що базується виключно на числових характеристиках CVSS з урахуванням типу технологій.

Сервіс LlmService формує вхідні дані для великої мовної моделі у вигляді JSON-структури, до якої входить список уразливостей для домену з множини V(d) та відповідна службова інформація. Для кожного запису передаються ідентифікатор CVE, числовий показник cvssScore, текстове значення severity, назва технології, до якої належить уразливість, а також стисле текстове резюме опису цієї уразливості. Додатково в структурі містяться агреговані характеристики, які відображають розподіл уразливостей за рівнями небезпеки та за типом технологій (серверні або клієнтські), що дозволяє моделі враховувати відмінність між загрозами, пов’язаними з компрометацією серверної частини, та ризиками, які впливають переважно на оточення користувача веббраузера. Локально розгорнутий інстанс ChatGPT отримує цей JSON разом із текстовою інструкцією, у якій зазначено, що необхідно оцінити ризик саме для користувача, який відвідує домен, і повернути лише структуровану відповідь з полями globalRiskScore, riskLevel та масивом рекомендацій.

Для кожного з п’яти тестових доменів Vulnweb інформаційна система запускається в повному режимі. Після побудови множини V(d) й обчислення R*(d)

сервіс LlmService надсилає відповідний запит до локального екземпляра моделі, зберігає отриману відповідь і витягує із JSON значення globalRiskScore та riskLevel. У результаті для кожного домену фіксуються три параметри: еталонна інтегральна оцінка ризику R*(d), числова оцінка globalRiskScore, сформована LLM, та категорія рівня ризику riskLevel, яку модель призначає на основі власної інтерпретації переданих даних.

На графіку (рис. 5.3) подано зведені результати цієї перевірки. Для кожного домену зазначено максимальний показник CVSS для серверних уразливостей, кількість серверних уразливостей із рівнями High або Critical, кількість клієнтських уразливостей із цими рівнями, еталонну інтегральну оцінку R*(d), числову оцінку globalRiskScore, що повертає LLM.

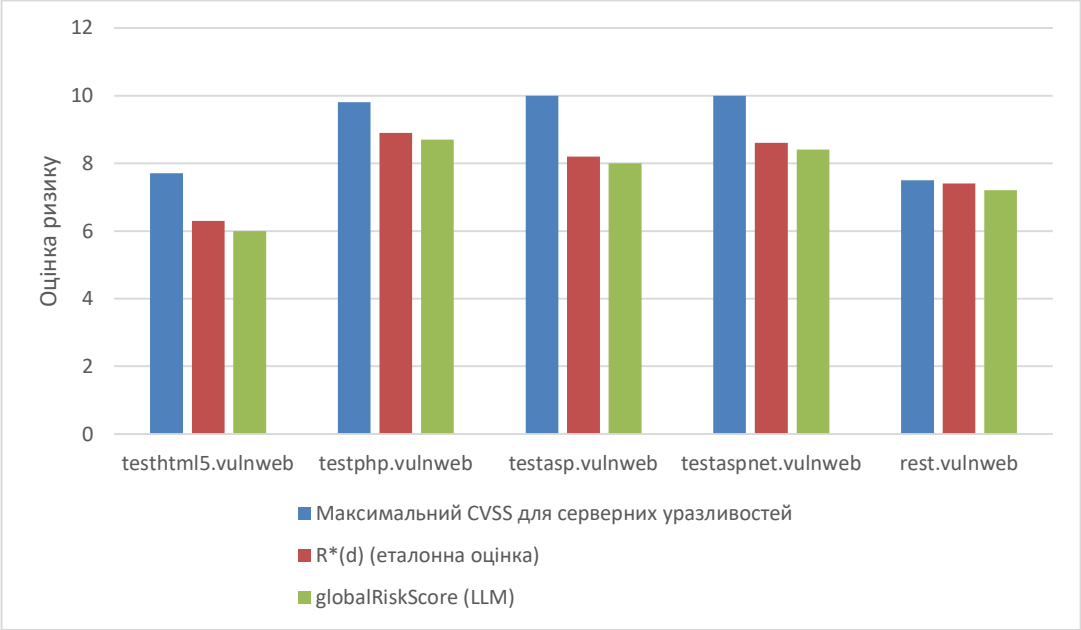

Рисунок 5.3 – Порівняння еталонних інтегральних оцінок R*(d) з globalRiskScore, сформованими LLM

Результати порівняння (рис. 5.3) доводять, що для всіх доменів числові значення globalRiskScore, сформовані локальним екземпляром ChatGPT, наближені до еталонних оцінок R*(d), отриманих на основі числових показників CVSS із врахуванням ваг серверних і клієнтських уразливостей. Для домену testhtml5.vulnweb, який має одну серверну уразливість рівня High та одну

клієнтську уразливість із таким самим рівнем небезпеки, еталонна оцінка ризику R*(d) дорівнює 6,3, тоді як модель повертає globalRiskScore 6,0 і класифікує домен як середньоризиковий (MEDIUM). Для домену http://testphp.vulnweb.com, де множина V(d) містить кілька уразливостей рівнів High та Critical для серверних компонентів стеку Apache–PHP–MySQL, еталонна оцінка становить 8,9, а LLM повертає значення 8,7 з рівнем ризику HIGH. Аналогічна картина спостерігається для доменів на базі IIS та ASP/ASP.NET, де наявність критичних серверних уразливостей призводить до високих значень як R*(d), так і globalRiskScore.

Розбіжності між R*(d) та globalRiskScore для кожного домену не перевищують кількох десятих бала, при цьому порядок доменів за зростанням ризику збігається для обох показників. Домен із найбільшою кількістю критичних серверних уразливостей отримує найвищу оцінку як за еталонною моделлю, побудованою на CVSS, так і за результатами роботи LLM; домени зі меншою кількістю уразливостей високого рівня та більшою часткою клієнтських уразливостей мають нижчі значення інтегральної оцінки і не переводяться у категорію CRITICAL. Це свідчить про те, що локально розгорнутий інстанс ChatGPT у рамках заданого промпта адекватно інтерпретує кількість і тяжкість уразливостей, відрізняє серверні та клієнтські компоненти та формує інтегральну оцінку ризику, яка узгоджується з числовими характеристиками CVSS.

У межах зіставлення інтегральної оцінки ризику з числовими показниками CVSS встановлено відповідність загальної тенденції: домени, для яких у збережених даних присутні уразливості з вищими значеннями CVSS, отримують підвищену інтегральну оцінку ризику у відповіді мовної моделі. Разом із результатами перевірки коректності отримання відомостей про уразливості для вибраних доменів це дає підстави застосовувати розроблене браузерне розширення як засіб орієнтовної оцінки ризику під час відвідування демонстраційних вебресурсів із навмисно вразливою конфігурацією, зокрема стендів Vulnweb.

ВИСНОВКИ

У кваліфікаційній роботі було розроблено браузерне розширення для кореляції домменів з базами вразливостей та оцінки ризику за допомогою LLM. Головною метою дослідження було створення ефективного інструмента, здатного в режимі реального часу інформувати користувача про потенційні загрози під час перегляду веб-ресурсів. На основі проведеного аналізу, моделювання та програмної реалізації можна стверджувати, що поставлена мета була досягнута повністю.

У рамках кваліфікаційної роботи спроєктовано та реалізовано браузерне розширення, яке взаємодіє із серверним прикладним програмним інтерфейсом на базі NestJS та локальним сховищем відомостей про уразливості, що синхронізується з базою NVD. Реалізовано модуль визначення технологій вебресурсу, який формує перелік програмних продуктів для подальшої кореляції з записами CVE. Оцінювання ризику виконується на основі числових показників CVSS та параметрів конфігурації вебресурсу, отриманих під час аналізу. Для формування рекомендацій використано мовну модель, яка перетворює технічні відомості про уразливості у текстові поради для користувача веббраузера.

Практична цінність розробленого браузерного розширення полягає в тому, що воно дає змогу користувачу веббраузера отримувати відомості про уразливості, пов’язані з доменом поточного вебресурсу, та узагальнену оцінку ризику у зручному вигляді без виконання додаткових дій. Розширення автоматично зчитує домен активної вкладки, надсилає його на серверний прикладний програмний інтерфейс і відображає результат аналізу у спливаючому вікні. За результатами оцінювання коректності встановлено, що запропонований підхід забезпечує відтворювану кореляцію доменів з відомостями з бази NVD та узгоджене формування рекомендацій за допомогою мовної моделі, що підвищує придатність результатів для практичного використання.

У кваліфікаційній роботі було визначено ряд обмежень браузерного розширення, пов’язаних із неможливістю повної ідентифікації прихованих

технологій вебзастосунків, залежністю від зовнішніх джерел даних та статистичною природою рекомендацій, сформованих мовною моделлю. Однак ці обмеження не зменшують загальної ефективності рішення та відкривають перспективи подальшого вдосконалення системи.

У подальшому розвиток системи може бути спрямований на інтеграцію механізмів виявлення аномалій у реальному часі, підтримку додаткових джерел вразливостей, впровадження модулів поведінкового аналізу доменів, а також створення розширених рекомендацій для власників веб-ресурсів. Також перспективним є застосування гібридних моделей машинного навчання для підвищення точності визначення технологій і формування CPE.

Виконана кваліфікаційна робота підтверджує доцільність і ефективність поєднання класичних методів аналізу вразливостей із сучасними інтелектуальними технологіями, створюючи інструмент, здатний підвищити рівень безпеки повсякденної роботи користувача у веб-просторі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. European Union Agency for Cybersecurity. ENISA Threat Landscape 2023 [Електронний ресурс]. – Режим доступу: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2023. – Дата звернення: 21.11.2025.

2. Verizon. 2023 Data Breach Investigations Report (DBIR) [Електронний ресурс]. – Режим доступу: https://www.verizon.com/business/resources/reports/2023-data-breach-investigations-report-dbir.pdf. – Дата звернення: 21.11.2025.

3. The MITRE Corporation. CVE — Common Vulnerabilities and Exposures: Overview [Електронний ресурс]. – Режим доступу: https://www.cve.org/about/overview. – Дата звернення: 21.11.2025.

4. National Institute of Standards and Technology. National Vulnerability Database (NVD): Program description [Електронний ресурс]. – Режим доступу: https://www.nist.gov/programs-projects/national-vulnerability-database-nvd. – Дата звернення: 21.11.2025.

5. Forum of Incident Response and Security Teams. Common Vulnerability Scoring System v3.1: Specification Document [Електронний ресурс]. – Режим доступу: https://www.first.org/cvss/v3-1/specification-document. – Дата звернення: 21.11.2025.

6. Coelho da Silva G. de J., Westphall C. B. A Survey of Large Language Models in Cybersecurity. arXiv:2402.16968, 2024 [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2402.16968. – Дата звернення: 21.11.2025.

7. Xu H., Wang S., Li N. та ін. Large Language Models for Cyber Security: A Systematic Literature Review. arXiv:2405.04760, 2024 [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2405.04760. – Дата звернення: 21.11.2025.

8. Jaffal N. O., Alkhanafseh M., Mohaisen D. Large Language Models in Cybersecurity: A Survey of Applications, Vulnerabilities, and Defense Techniques // AI. – 2025. – Vol. 6, No. 9. – Art. 216. – DOI: 10.3390/ai6090216 [Електронний ресурс]. – Режим доступу: https://www.mdpi.com/2673-2688/6/9/216. – Дата звернення:

21.11.2025.

9. Paulsen C. (ed.). NISTIR 7298 Rev. 3. Glossary of Key Information Security Terms. – Gaithersburg, MD : National Institute of Standards and Technology, 2019 [Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.7298r3.pdf. – Дата звернення: 21.11.2025.

10. National Institute of Standards and Technology. SP 800-30 Rev. 1. Guide for Conducting Risk Assessments. – Gaithersburg, MD : National Institute of Standards and Technology, 2012 [Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf. – Дата звернення: 21.11.2025.

11. International Organization for Standardization; International Electrotechnical Commission. ISO/IEC 27005:2022 — Information security, cybersecurity and privacy protection — Guidance on managing information security risks [Електронний ресурс]. – Режим доступу: https://www.iso.org/standard/80585.html. – Дата звернення: 21.11.2025.

12. Forum of Incident Response and Security Teams. Exploit Prediction Scoring System (EPSS) [Електронний ресурс]. – Режим доступу: https://www.first.org/epss/. – Дата звернення: 21.11.2025.

13. Mell P., Spring J. Likely Exploited Vulnerabilities: A Proposed Metric for Vulnerability Exploitation Probability. NIST Cybersecurity White Paper (CSWP) 41. – Gaithersburg, MD : National Institute of Standards and Technology, 2025. – DOI: 10.6028/NIST.CSWP.41 [Електронний ресурс]. – Режим доступу: https://nvlpubs.nist.gov/nistpubs/cswp/nist.cswp.41.pdf. – Дата звернення: 21.11.2025.

14. The OWASP Foundation. OWASP Zed Attack Proxy (ZAP) — Official website [Електронний ресурс]. – Режим доступу: https://www.zaproxy.org/. – Дата звернення: 21.11.2025.

15. PortSwigger Ltd. Burp Suite — Web vulnerability scanner and documentation [Електронний ресурс]. – Режим доступу: https://portswigger.net/burp/. – Дата звернення: 21.11.2025.

16. SecurityHeaders.com. Analyse your HTTP response headers [Електронний ресурс]. – Режим доступу: https://securityheaders.com/. – Дата звернення: 21.11.2025.

17. Qualys SSL Labs. SSL Server Test [Електронний ресурс]. – Режим доступу: https://www.ssllabs.com/ssltest/. – Дата звернення: 21.11.2025.

18. Google. Chrome Extensions — Manifest V3 overview [Електронний ресурс]. – Режим доступу: https://developer.chrome.com/docs/extensions/mv3/intro/. – Дата звернення: 21.11.2025.

19. Google. chrome.tabs API (Chrome Extensions) [Електронний ресурс]. – Режим доступу: https://developer.chrome.com/docs/extensions/reference/api/tabs. – Дата звернення: 21.11.2025.

20. Google. chrome.storage API (Chrome Extensions) [Електронний ресурс]. – Режим доступу: https://developer.chrome.com/docs/extensions/reference/api/storage. – Дата звернення: 21.11.2025.

21. NestJS. Official documentation [Електронний ресурс]. – Режим доступу: https://docs.nestjs.com/. – Дата звернення: 21.11.2025.

22. TypeORM. Documentation [Електронний ресурс]. – Режим доступу: https://typeorm.io/. – Дата звернення: 21.11.2025.

23. PostgreSQL Global Development Group. PostgreSQL Documentation [Електронний ресурс]. – Режим доступу: https://www.postgresql.org/docs/. – Дата звернення: 21.11.2025.