Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

Бакалаврська кваліфікаційна робота на тему:
«Програмний засіб для розвідки з відкритих джерел на основі ШІ-агентів»

Виконав: студент 4 курсу групи 1БКС–21б
спеціальності 125 Кібербезпека
__________________ Нікіта СУХАРЕВ

Керівник: к. т. н., доцент каф. ЗІ
______________ Леонід КУПЕРШТЕЙН
«16» __червня__ 2025 р.

Рецензент: к. т. н., доцент каф. ПЗ
________________ Олена КОВАЛЕНКО
«16» __червня__ 2025 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
__________ Володимир ЛУЖЕЦЬКИЙ
«16» __червня__ 2025 р.

Вінниця ВНТУ – 2025 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти I (бакалаврський)
Галузь знань – 12 Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітньо-професійна програма – Кібербезпека критичних систем

ЗАТВЕРДЖУЮ
зав. кафедри ЗІ, д. т. н., проф.
__________ Володимир ЛУЖЕЦЬКИЙ
21 березня 2025 року

ЗАВДАННЯ
НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Сухареву Нікіті Олександровичу

1. Тема роботи: «Програмний засіб для розвідки з відкритих джерел на основі ШІ-агентів»
керівник роботи: Куперштейн Леонід Михайлович, к.т.н., доц. каф. ЗІ.
затверджені наказом ректора ВНТУ від 20 березня 2025 року №97.

2. Строк подання студентом роботи 16 червня 2025 року.

3. Вихідні дані до роботи:
- мова програмування – Python.
- тип додатку – CLI-додаток;
- програмна архітектура – CrewAI;

4. Зміст розрахунково-пояснювальної записки: Вступ. Аналіз предметної області. Технічне проектування застосунку. Робоче проектування застосунку. Висновки. Перелік використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: Загальна архітектура мультиагентної системи. Загальна схема алгоритму програмного засобу. Структура проекту розвідки з відкритих джерел. Результати успішної розвідки з відкритих джерел. Результати неуспішної розвідки з відкритих джерел.

6. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Куперштейн Л.М., к.т.н., доц. каф. ЗІ.	24.03.25	10.06.25
2	Куперштейн Л.М., к.т.н., доц. каф. ЗІ.	25.04.25	10.06.25
3	Куперштейн Л.М., к.т.н., доц. каф. ЗІ.	30.05.25	10.06.25

7. Дата видачі завдання 21 березня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	24.03.2025 – 30.03.2025	
2	Аналіз інформаційних джерел за напрямком бакалаврської кваліфікаційної роботи	31.03.2025 – 13.04.2025	
3	Розробка рішень, моделей, алгоритмів	14.04.2025 – 04.05.2025	
4	Практична реалізація, моделювання, експериментування, результати	05.05.2025 – 14.05.2025	
5	Оформлення пояснювальної записки	15.05.2025 – 18.05.2025	
6	Попередній захист БКР	19.05.2025 – 26.05.2025	
7	Виправлення зауважень, підготовка ілюстративного матеріалу	27.05.2025 – 30.05.2025	
8	Перевірка на наявність тестових запозичень	31.05.2025 – 10.06.2025	
9	Представлення БКР до захисту, рецензування	11.06.2025 – 13.06.2025	
10	Захист БКР	17.06.2025 – 20.06.2025	

Студент __________ Нікіта СУХАРЕВ
(підпис)

Керівник роботи __________ Леонід КУПЕРШТЕЙН
(підпис)

АНОТАЦІЯ

Бакалаврська кваліфікаційна робота складається з 106 сторінки формату А4, на яких є 7 рисунків, 6 таблиці, список використаних джерел містить 44 найменувань.

Бакалаврська робота присвячена розробці програмного засобу для розвідки з відкритих джерел на основі ШІ-агентів. Засіб дозволяє автоматизовано здійснювати збір, фільтрацію та аналіз даних у сфері кібербезпеки.

Розроблено засіб для розвідки з відкритих джерел на основі ШІ-агентів. Проведено тестування розробленого засобу. Для реалізації взаємодії між агентами застосовано мультиагентну архітектуру CrewAI. Для генерації висновків та обробки текстової інформації в системі використано велику мовну модель GPT-4o. Проведено тестування розробленого засобу.

Ключові слова: кібербезпека, розвідка з відкритих джерел, велика мовна модель, Python, CrewAI.

ABSTRACT

The Bachelor's qualification thesis consists of 106 A4-format pages, which contain 7 figures, 6 tables, the list of references comprises 44 entries.

The bachelor's thesis is dedicated to the development of a software tool for open-source intelligence based on AI agents. The tool enables automated collection, filtering, and analysis of data in the field of cybersecurity.

A software solution for open-source intelligence based on AI agents has been developed and tested. The interaction between agents is implemented using the multi-agent architecture CrewAI. To generate conclusions and process textual information, the system employs the large language model GPT-4o. The developed tool has been tested to evaluate its functionality.

Keywords: cybersecurity, open-source intelligence, large language model, Python, CrewAI.

ЗМІСТ

ВСТУП

У сучасному цифровому середовищі обсяг відкритої інформації, що генерується щодня, досягає небачених масштабів. Соціальні мережі, публічні реєстри, веб-сайти компаній, новинні ресурси, форуми та багато інших джерел постійно наповнюються даними, які можуть містити важливу інформацію про організації, осіб, події чи технічні інфраструктури. Саме тому Open Source Intelligence (OSINT) – розвідка з відкритих джерел – набула особливої ваги у сфері інформаційної безпеки.

OSINT є одним із ключових етапів аналізу кіберзагроз і передумовою для подальших заходів реагування чи тестування на проникнення. Завдяки можливості отримувати безконтактно детальну інформацію про інфраструктуру організації, співробітників, програмні продукти або уразливості, OSINT використовується як захисниками, так і зловмисниками. Відповідно, важливість ефективного збору, обробки й аналізу такої інформації безперечно зростає.

Однак через велику кількість джерел, різноманітність форматів даних, швидкість їх оновлення й необхідність верифікації отриманої інформації традиційні OSINT-засоби часто виявляються малоефективними або потребують високої кваліфікації з боку аналітиків. Це створює передумови для розробки інноваційних рішень, які дозволяють автоматизувати основні етапи розвідки, підвищити точність і швидкість обробки даних.

Одним із таких рішень є використання мультиагентних систем на основі великих мовних моделей (LLM), які дозволяють розподілити функції між окремими агентами, що взаємодіють між собою. Ці агенти можуть виконувати специфічні завдання – від агрегації даних до виявлення зв’язків, генерації звітів і формулювання висновків. Фреймворк CrewAI, на основі якого реалізується запропонований підхід, надає необхідну інфраструктуру для розробки таких систем.

Таким чином, розробка мультиагентного програмного засобу для розвідки з відкритих джерел із використанням LLM є актуальним напрямом, що поєднує новітні досягнення у сфері штучного інтелекту, кібербезпеки та автоматизованої аналітики.

Об’єктом бакалаврської кваліфікаційної роботи є процеси розвідки з відкритих джерел у сфері кібербезпеки.

Предметом бакалаврської кваліфікаційної роботи є методи і засоби реалізації інструментів для розвідки з відкритих джерел з використанням мультиагентних систем на основі великих мовних моделей.

Метою роботи є підвищення ефективності процесу кіберрозвідки з відкритих джерел шляхом застосування інтелектуальних агентів.

Для досягнення мети потрібно вирішити такі задачі:

– проаналізувати сучасні підходи до OSINT та інструменти, що використовуються в галузі кібербезпеки;
– дослідити можливості застосування ШІ-моделей і мультиагентних систем для автоматизації OSINT;
– розробити архітектуру програмного засобу з використанням CrewAI;
– реалізувати прототип програмного засобу для мультиагентного OSINT;
– провести тестування і оцінку функціональності та ефективності реалізованої системи;
– сформулювати рекомендації щодо використання і розвитку мультиагентних OSINT-систем у сфері кібербезпеки.

Результати проведеної роботи з розробки мультиагентного програмного засобу для розвідки з відкритих джерел на основі великих мовних моделей були представлені на Міжнародній науково-практичній інтернет-конференції студентів, аспірантів та молодих науковців «Молодь у науці: дослідження, проблеми, перспективи (МН-2025)», що проходила у Вінницькому національному технічному університеті (Вінниця, 2025 р.) [1].

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Базові поняття OSINT і його роль у кібербезпеці

Відкрита розвідка або OSINT (Open-Source Intelligence) – це систематизований процес збору, аналізу та використання публічно доступної інформації для цілей розвідки, аналітики або оцінки ризиків [2]. OSINT виник як напрям розвідки ще під час Другої світової війни. Пізніше, у період Холодної війни, він став критичним доповненням до закритих джерел інформації, особливо в хумовах інформаційного протистояння [3].

З часом OSINT набув вагомого значення в контексті національної безпеки, протидії кіберзагрозам, журналістських розслідувань і корпоративної аналітики. В сучасних умовах, коли цифрова інформація є всюдисущою, OSINT перетворився на окрему дисципліну, яка має чітко визначені принципи, методології та правові обмеження [4].

Починаючи з 2000-х років, із бурхливим зростанням обсягів інформації в інтернеті та соціальних мережах, OSINT вийшов за межі урядових і військових структур. Цей процес називають демократизацією розвідки. Тепер не лише спецслужби, а й журналісти, аналітики, правозахисники та звичайні користувачі можуть збирати й аналізувати відкриту інформацію.

До основних джерел OSINT можна віднести:

– телевізійні новини, друковані видання, онлайн-ЗМІ;
– форуми, блоги, соціальні мережі, відкриті бази даних;
– урядові звіти, реєстри, відкриті фінансові документи;
– DNS-записи, WHOIS-дані, IP-інфраструктура;
– фото, відео, супутникові зображення.

У сучасному ландшафті інформаційної безпеки термін OSINT дедалі частіше використовується в контексті комплексного аналізу даних. Деякі суміжні напрямки частково перетинаються з OSINT, якщо базуються на відкритих джерелах.

Наприклад, розвідка, що базується на людських джерелах (Human Intelligence, HUMINT) оперує інформацією, отриманою безпосередньо від людей, і може включати публічні виступи, форуми чи свідчення очевидців, хоча здебільшого залишається закритою розвідкою. Розвідка з соціальних мереж (Social Media Intelligence, SOCMINT), натомість, є складовою OSINT і спеціалізується на відкритих даних із соцмереж, забезпечуючи швидкий моніторинг суспільних настроїв. Технічна розвідка (Technical Intelligence, TECHINT) охоплює технічну документацію та конфігурації, які в разі публічного доступу також можуть бути джерелами OSINT. Аналогічно, геопросторова розвідка (Geospatial Intelligence, GEOINT), що працює з геоданими, супутниковими знімками та мапами, активно використовується в OSINT для просторового аналізу подій [5, 6].

Для кращого розуміння ролі OSINT у контексті інших типів розвідки, наведено порівняльну характеристику суміжних напрямів у таблиці 1.1.

Таблиця 1.1 – Порівняння суміжних напрямів розвідки

Напрям	Джерело даних	Методи збору	Сильні сторони	Обмеження
HUMINT	Люди, інтерв’ю	Спостереження, агентурна діяльність	Глибина розуміння поведінки	Суб’єктивність, етичні питання
SOCMINT	Соціальні мережі	Профілювання, аналіз дописів	Швидкість, масштабованість	Проблеми з достовірністю, приватністю
TECHINT	Технічні джерела	Аналіз техдокументації, зразків	Доступ до тех. характеристик	Потребує експертності
GEOINT	Супутникові знімки, мапи	Геоаналіз, дистанційне зондування	Просторовий контекст	Помилки інтерпретації, вартість

Ефективність розвідки з відкритих джерел значною мірою залежить від методів, які застосовуються для збору інформації. Ці методи поділяються на дві великі категорії – ручні та автоматизовані. Кожна з них має свої переваги, недоліки та сфери застосування. Розуміння відмінностей між ними є ключовим для формування стратегії OSINT в рамках захисту інформаційного середовища, а також при розробці багаторівневих аналітичних систем у складних аналітичних сценаріях, які вимагають багатоступеневого аналізу [7].

Ручні методи збору OSINT є класичним підходом, при якому аналітик безпосередньо здійснює пошук, відбір та перевірку інформації. До таких дій належать класичні техніки пошуку в Google, аналіз профілів у соцмережах, робота з відкритими реєстрами й перевірка метаданих мультимедіа. Ручний підхід дозволяє враховувати контекст, приймати рішення в реальному часі та виявляти приховані залежності. Проте він має обмеження в масштабованості, швидкості та повторюваності.

Автоматизовані методи збору OSINT забезпечують масштабованість, швидкість і автоматичне виявлення патернів у великих обсягах даних. Ці методи базуються на використанні скриптів, парсерів, API, ботів та інструментів із вбудованими алгоритмами машинного навчання. Серед прикладів цих методів можна виділити скрейпінг, роботу з API, наприклад, Shodan, VirusTotal, а також інтеграцію інструментів машинного навчання для класифікації та аналізу текстів. Автоматизовані методи є незамінними при роботі з великим потоком інформації, дозволяють зменшити людський фактор і підвищити точність. Водночас вони вимагають належної технічної реалізації та можуть мати юридичні обмеження, особливо щодо приватності та дотримання стандартів GDPR [8, 9].

Практично OSINT застосовується у багатьох випадках, як приклад, у виявленні терористичних мереж, зокрема ІДІЛ, через активність у соцмережах, пошуку зниклих осіб на основі геолокації, згадок і фотографій, виявленні схем торгівлі дітьми та експлуатації через форуми, даркнет і зливи акаунтів. Ці випадки демонструють важливість обох підходів – ручного і автоматизованого – у складних аналітичних сценаріях [10].

Розвідка з відкритих джерел перетворилась на ключовий елемент у сучасному аналізі інформаційного простору. Завдяки широкій доступності даних та розмаїттю джерел, OSINT відіграє критичну роль у кібербезпеці, журналістиці та розвідувальних операціях. Його використання варіюється від технічного сканування інфраструктури до виявлення загроз у соціальних мережах. Гнучкість і масштабованість роблять OSINT універсальним інструментом у багатьох секторах.

У сфері кібербезпеки OSINT застосовується для виявлення вразливостей, збору інформації про інфраструктуру та моніторингу загроз у відкритих джерелах [11]. Основні напрями його використання включають:

– виявлення фішингових сайтів, витоків паролів, підроблених доменів;
– піддомени, IP-адреси, відкриті порти;
– відстеження активності в даркнеті або форумах з обговоренням хакерських атак;
– виявлення витоків конфігурацій, документів, токенів доступу до API;
– збір індикаторів компрометації (Indicators of Compromise, IoC) із публічних звітів, GitHub, Pastebin.

Ці рішення використовуються аналітиками SOC, пентестерами, та дослідниками загроз (threat hunters) для побудови профілю об’єкта дослідження. Наприклад, компанія Vaadata демонструє сценарії використання OSINT у процесі соціальної інженерії, а Flashpoint – у сфері попередження атак на корпоративні системи.

У розвідувальній діяльності OSINT виконує критичну роль у зборі оперативної інформації про загрози, пов’язані з тероризмом, транснаціональною злочинністю або геополітичними подіями. Він слугує доповненням до традиційних каналів, як-от SIGINT або HUMINT, забезпечуючи швидкий доступ до публічних даних. OSINT дозволяє моніторити поведінку підозрюваних у соцмережах та форумах, відстежувати зв’язки між особами або організаціями через аналіз відкритих реєстрів і зливів, здійснювати геоаналіз конфліктних територій, виявляти кіберзагрози ще до здійснення атаки. Також розвідоргани дедалі частіше розгортають мультиджерельні системи, які поєднують OSINT із SIGINT, IMINT та ін. Це дає змогу оперативно будувати цілісну картину подій у режимі майже реального часу [12].

Попри значні переваги, OSINT має низку критичних обмежень, які потребують врахування під час його використання:

- важко відокремити релевантні дані серед великого потоку;
- поширення дезінформації, фейкових акаунтів і маніпулятивних матеріалів;
- доступ до комерційних сервісів може бути лімітованим або платним;
- OSINT часто стикається з конфліктами конфіденційності, особливо при зборі інформації про приватних осіб;
- порушення політик вебсайтів, обмеження GDPR, локальні закони;
- недостатній рівень підготовки може призвести до помилкових висновків;
- відмінності у підходах до валідації, документування, обміну даними.

Ці виклики накладають обмеження як на аналітиків, так і на автоматизовані системи, зокрема мультиагентні рішення. Наявність продуманих політик етики, тренінгів для користувачів та контроль якості інформації є необхідною умовою безпечного використання OSINT.

Широке впровадження технологій збору відкритих даних спричинило появу численних етичних і правових питань. Хоча інформація, що використовується в OSINT, технічно є публічною, це не означає, що її використання є завжди етичним або законним. Особливо критичними є моменти, коли йдеться про конфіденційність, згоду, дезінформацію чи збір даних про вразливі групи. Розуміння меж допустимого та формування етичних принципів роботи з OSINT є необхідною умовою його відповідального застосування [13].

Попри технічну відкритість даних, використання OSINT часто підпадає під регуляторні обмеження, що залежать від юрисдикції. Основні правові виклики пов’язані з обробкою персональних даних, правами інтелектуальної власності, а також дотриманням принципів законності та пропорційності.

У країнах ЄС OSINT підпорядковується Загальному регламенту про захист даних (General Data Protection Regulation, GDPR). Він вимагає обґрунтування законного інтересу, обмеження обсягу зібраних даних, чіткого визначення цілей і забезпечення права на видалення. Публічна доступність інформації не скасовує вимогу щодо захисту персональних даних.

Законодавство окремих країн може бути ще суворішим. Наприклад, у Німеччині збір даних із соціальних мереж можливий лише при наявності чітко задокументованого інтересу, і він повинен бути обґрунтований перед наглядовими органами.

У США регуляція OSINT є фрагментованою. Основним актом є California Consumer Privacy Act (CCPA), що забезпечує право на доступ, видалення та заборону продажу даних. Окремі штати, як-от Вірджинія або Колорадо, мають додаткові закони, які обмежують автоматизований збір даних без згоди.

Відсутність єдиного глобального регулювання призводить до значних прогалин у правозастосуванні. Практики OSINT повинні дотримуватись не лише національного, але й екстериторіального права, наприклад, при зборі даних про громадян ЄС компанією з США.

Тому юридична відповідність є однією з ключових умов легітимного використання OSINT у розвідці, журналістиці, кібербезпеці та дослідницькій діяльності.

У зв’язку з відсутністю єдиного регулювання, аналітики OSINT несуть значну етичну відповідальність за способи використання інформації. Поява штучного інтелекту, зростання обсягів даних і політична поляризація лише підсилюють ці виклики.

Серед сучасних викликів, пов'язаних із використанням OSINT, варто виділити проблему балансу між відкритістю інформації та збереженням приватності, адже публічність даних часто не гарантує етичного використання. Активне впровадження ШІ пришвидшує аналіз, але водночас ускладнює контроль над упередженнями та можливими зловживаннями. Крім того, урядові й комерційні структури нерідко класифікують важливу для суспільства інформацію, що ускладнює її перевірку. А постійні зміни політик соцмереж щодо доступу до даних створюють додаткові труднощі для дотримання етичних стандартів. Стійке використання OSINT у майбутньому залежить від збалансованого поєднання правової бази, етичної самосвідомості та технічної гнучкості.

Із розвитком цифрових технологій OSINT змінює свої функції – від простого збору даних до складного аналітичного інструмента, що інтегрується з інтелектуальними системами. Сучасні потреби у швидкості, масштабованості та точності обробки інформації викликають потребу в мультиагентних архітектурах, посилених штучним інтелектом.

У майбутньому OSINT може стати критичним елементом стратегічної аналітики в урядових, корпоративних і громадських структурах. Його успішність залежатиме від того, наскільки вдало буде інтегровано AI та збережено баланс між автоматизацією й етикою.

1.2 Порівняльний аналіз інструментів OSINT

У сфері кібербезпеки інструменти OSINT відіграють важливу роль у виявленні загроз, зборі контекстної інформації про об’єкти атаки та підтримці процесів розслідування інцидентів. Вони дозволяють фахівцям отримати ранні сигнали про можливі зловмисні дії, вивчити структуру цифрової присутності організацій та осіб, а також ідентифікувати вразливі компоненти інфраструктури. Залежно від типу даних, з якими вони працюють, такі інструменти можна умовно поділити на п’ять категорій: технічні, соціальні, людинозорієнтовані, геопросторові та універсальні.

Технічні інструменти аналізують IP-адреси, порти, DNS-записи, конфігурації систем, сертифікати та інші параметри, що пов’язані з мережею або хостами. Інструменти соціальної розвідки орієнтовані на аналіз цифрової присутності користувачів у соціальних мережах, форумах, месенджерах і на публічних платформах. Геопросторові рішення призначені для отримання інформації, пов’язаної з місцем розташування, включаючи метадані зображень, трекінг пересування та супутникову аналітику. Людинозорієнтована розвідка у відкритих джерелах охоплює інструменти, що дозволяють виявити й аналізувати персональні дані осіб, а саме, імена, електронні адреси, зв'язки, акаунти. Універсальні інструменти OSINT поєднують функціональність одразу кількох напрямів, автоматизують розвідку з багатьох джерел і дозволяють візуалізувати зв'язки між об'єктами. Вони особливо корисні під час складних досліджень, де необхідно швидко зібрати та корелювати велику кількість розрізненої інформації.

У практиці кібербезпеки класичні OSINT-інструменти часто застосовуються для розвідки мережевої інфраструктури, аналізу цифрових слідів користувачів, а також виявлення потенційно скомпрометованих елементів систем. Втім, попри широке використання, кожен із таких інструментів має як переваги, так і обмеження, що знижує їхню універсальність у динамічному середовищі загроз. Саме тому дедалі частіше виникає потреба у створенні комплексних рішень, здатних гнучко комбінувати різні типи аналітики та оперативно адаптуватися до змін у поведінці супротивника.

У рамках дослідження можливостей OSINT було здійснено порівняльний огляд тридцяти інструментів, що використовуються фахівцями у сфері кібербезпеки. Для зручності аналізу їх було класифіковано за напрямом OSINT, об'єктом збору, типом реалізації та коротким описом призначення. Деталізована таблиця представлена у додатку А, який дозволяє узагальнити підходи та функціональність доступних засобів розвідки з відкритих джерел [14-17].

Серед інструментів SOCMINT варто виділити GHunt, який дозволяє отримати інформацію про Google-акаунти, а саме, зображення, документи, календарі, навіть активність користувача. Інший приклад – CheckUsernames, веб-застосунок для перевірки наявності username на різних соціальних платформах. Social Searcher забезпечує реальний моніторинг публікацій у соцмережах, що корисно для виявлення фейків або інформаційних кампаній. Ці інструменти широко застосовуються в OSINT-процесах, пов'язаних з ідентифікацією особи або аналізом її присутності в мережі.

Серед інструментів TECHINT варто назвати Metagoofil, що здійснює пошук документів і метаданих у відкритих джерелах, дозволяючи отримати інформацію про авторів та структуру файлу. Dnsdumpster збирає публічні DNS-записи та допомагає виявити архітектуру домену. ZoomEye – це пошуковик інтернет-пристроїв і сервісів, схожий на Shodan, але з іншим індексаційним підходом. Всі вони є потужними інструментами на етапах передексплуатаційного аналізу та інфраструктурної розвідки.

До засобів GEOINT входить Exiftool – консольна утиліта для вилучення геоміток і параметрів із медіафайлів. Також варто вказати FOCA, який хоча й фігурує в категорії TECHINT, містить функції для отримання географічних координат із документів. Обидва інструменти активно використовуються в цифровій криміналістиці для локалізації подій або підтвердження місця створення контенту.

HUMINT – напрям OSINT, орієнтований на збір інформації про конкретних осіб. Серед відповідних інструментів можна назвати Pipl, який агрегує профілі з різних платформ і дозволяє ідентифікувати особу за email, ім’ям або номером телефону. PeopleLooker – це сервіс для пошуку даних про осіб у відкритих базах, з можливістю отримання відомостей про адресу, зв’язки чи навіть судимості. Hunter.io забезпечує швидкий пошук корпоративних email-адрес за доменом компанії, що дозволяє ідентифікувати ключових осіб для подальшого аналізу або соціальної інженерії.

Серед універсальних засобів OSINT, які охоплюють одразу кілька типів даних, вирізняються IntelTechniques, що надає набір утиліт і посилань для автоматизованого пошуку по соціальних мережах, базах даних, картографічних і доменних сервісах. Google Dorks використовують можливості пошукового індексу Google для отримання зразків конфігураційних файлів, списків користувачів і публічно доступних документів.

У сучасній практиці кібербезпеки фахівці користуються великим спектром інструментів OSINT, кожен з яких має свою спеціалізацію та область застосування. У таблиці 1.2 представлено десять найбільш поширених інструментів, які активно використовуються у сфері аналізу відкритої інформації, з короткою характеристикою їх призначення, типу реалізації та джерел даних, з якими вони працюють.

Таблиця 1.2 – Перелік найпоширеніх інструментів OSINT

№	Назва інструменту	Категорія OSINT	Об’єкт збору	Тип реалізації	Призначення
1	Shodan	TECHINT	IP, порти, банери	Web	Виявлення мережевих пристроїв
2	TheHarvester	TECHINT	Email, домени	Console	Збір email-адрес і доменів
3	Maltego	Universal	IP, домени, особи	Desktop	Візуалізація зв’язків
4	SpiderFoot	Universal	IP, домени, email	Desktop/Web	Автоматизована багатоджерельна розвідка
5	Recon-ng	Universal	IP, акаунти, домени	Console	Модульна платформа для розвідки
6	Censys	TECHINT	IP, TLS-сертифікати	Web	Індексація цифрової інфраструктури
7	Sherlock	SOCMINT	Username, соцмережі	Console	Пошук акаунтів у соцмережах
8	TweetDeck	SOCMINT	Пости, хештеги	Web	Моніторинг Twitter у реальному часі
9	Creepy	GEOINT	Геолокація, соцмережі	Desktop	Аналіз координат із публікацій
10	FOCA	TECHINT / GEOINT	Метадані, координати	Desktop	Аналіз документів з метаданими

Shodan [18] – спеціалізований пошуковик для виявлення пристроїв, підключених до Інтернету. Він дозволяє фахівцям з кібербезпеки проводити сканування відкритих портів, визначати конфігурації сервісів, а також виявляти потенційно вразливі вузли в інфраструктурі організацій. Shodan часто застосовується для пошуку пристроїв IoT, незахищених камер, SCADA-систем та серверів з відкритими портами.

TheHarvester [19] – це інструмент командного рядка, який дозволяє збирати електронні адреси, імена субдоменів, DNS-записи та іншу публічну інформацію з відкритих джерел, таких як пошукові системи, сервіси криптоаналізу тощо. Його використовують для первинної розвідки на етапах тестування на проникнення або при аналізі потенційних загроз.

Maltego [20] – застосунок для візуального аналізу та побудови графів взаємозв'язків між об'єктами OSINT. Серед таких об'єктів – домени, IP-адреси, акаунти користувачів, email, організації тощо. Maltego дозволяє швидко встановлювати зв'язки між зібраними даними, що робить його цінним інструментом у цифровій криміналістиці та дослідженнях соціальної інженерії.

SpiderFoot [21] – універсальний інструмент для автоматизованої розвідки, який здатен збирати інформацію з понад 100 джерел. Він охоплює технічні, соціальні та організаційні аспекти об'єктів дослідження, зокрема домени, email-адреси, IP, ASN та інше. SpiderFoot підтримує як інтерактивний веб-інтерфейс, так і CLI-режим, що забезпечує його гнучке застосування в різних сценаріях.

Recon-ng [22] – фреймворк на Python для OSINT-розвідки, орієнтований на модульну обробку даних. Його інтерфейс нагадує Metasploit Framework, що полегшує використання для фахівців з пентестингу. Recon-ng дозволяє автоматизувати збирання інформації про цілі за допомогою підключення різних модулів, API та баз даних.

Censys [23] – онлайн-платформа для дослідження мережевої інфраструктури та аналізу TLS/SSL-сертифікатів. Інструмент дозволяє знаходити відкриті хости, сервіси й цифрові сертифікати на основі індексації Інтернету. Censys використовується для атрибуції кіберзагроз, перевірки відповідності безпеки та моніторингу стану цифрових активів.

Sherlock [24] – легкий CLI-інструмент, що дозволяє шукати профілі користувачів за нікнеймом на сотнях соціальних платформ. Особливо корисний на етапі початкової ідентифікації об'єкта дослідження, Sherlock може швидко знайти цифрові сліди особи, що дає змогу встановити її активність в Інтернеті.

Creepy [25] – інструмент для визначення геолокації користувачів за допомогою аналізу метаданих зображень, постів у соцмережах та інших відкритих джерел. Застосовується для створення профілю поведінки та пересування об'єктів розвідки. Часто використовується у розслідуваннях, що стосуються фізичної безпеки або кіберсталкінгу.

TweetDeck [26] – офіційний інструмент від Twitter, який дозволяє моніторити активність у соцмережі в режимі реального часу. Незважаючи на те, що він не є спеціалізованим засобом OSINT, його використовують для виявлення інформаційних кампаній, оцінки громадського резонансу або моніторингу деструктивного контенту.

FOCA [27] – десктопна утиліта для вилучення метаданих із документів (PDF, DOCX, PPTX тощо), що розміщені у відкритому доступі. Вона дозволяє отримувати інформацію про авторів, шляхи до файлів, програмне забезпечення, версії ОС тощо, що може стати в пригоді при атрибуції організацій або виявленні витоків.

Таким чином, аналіз інструментів OSINT засвідчує наявність широкого спектру спеціалізованих та універсальних рішень, що охоплюють як технічні, так і соціальні аспекти інформаційного середовища. Таке групування дозволяє більш ефективно підходити до добору інструментів під конкретні завдання та сценарії, з урахуванням типу даних і специфіки цільового об’єкта дослідження.

1.3 Потенціал ШІ та мультиагентних систем в OSINT

В останні роки великі мовні моделі (Large Language Models, LLM), такі як GPT-4o, Claude, LLaMA та Mistral, стали визначальним фактором розвитку інтелектуальних систем аналізу відкритих джерел. Вони дозволяють обробляти великі обсяги неструктурованої інформації, формувати зв’язки між об’єктами, виявляти тематики й тренди, а також генерувати узагальнені звіти на основі фрагментованих даних. У сфері кібербезпеки це відкриває нові можливості для проактивної розвідки, виявлення інформаційних атак, цифрової ідентифікації та підтримки інцидент-респонсу [28].

LLM забезпечують семантичний аналіз тексту, здатні виявляти контекстні закономірності, класифікувати інформацію та надавати висновки з урахуванням специфіки запиту. Наприклад, модель може згенерувати зведення з новин про кібератаку, отримати з нього структуру подій, назви компаній і потенційні вектори компрометації. Такі можливості не лише скорочують час аналітичного циклу, але й підвищують його глибину.

Інтеграція LLM у OSINT-системи дає змогу будувати адаптивні аналітичні ланцюги, де модель не лише обробляє текст, але й керує потоками дій – наприклад, запуском перевірки додаткових джерел, уточненням параметрів пошуку або генерацією проміжних запитів. На відміну від класичних алгоритмів обробки даних, LLM демонструють здатність до гнучкої роботи з напівструктурованими або неповними входами, що особливо важливо в умовах обмеженого доступу до інформації.

Разом із цим постає і низка викликів, а саме, висока вартість обчислень, обмежена пояснюваність (black-box nature), ризик генерації упереджених або хибних результатів. Тим не менш, активний розвиток напрямів explainable AI, локального запуску моделей (наприклад, на базі LLaMA), а також інструментів типу retrieval-augmented generation (RAG) дає підґрунтя для безпечного та ефективного застосування LLM у системах OSINT, що працюють у сфері кібербезпеки.

У системах розвідки з відкритих джерел все більшого поширення набуває концепція Multi-Agent Systems (MAS) – архітектур, у яких кілька автономних програмних агентів співпрацюють для досягнення складної аналітичної мети. На відміну від централізованих або одноагентних систем, MAS забезпечують розподілення функцій між спеціалізованими агентами, що підвищує гнучкість, масштабованість і стійкість до відмов [29].

У контексті OSINT, агенти можуть бути налаштовані на виконання окремих ролей, наприклад, один здійснює пошук по API, другий фільтрує результати, третій здійснює тематичне групування, а четвертий генерує звіт. Подібна модульність дозволяє адаптувати систему до різних сценаріїв кіберрозвідки, зокрема моніторингу соціальних мереж, перевірки доменів, виявлення витоків даних або формування попереджень щодо загроз.

Архітектура MAS зазвичай включає три основні компоненти, а саме, індивідуальні агенти з власною логікою поведінки, систему комунікації між агентами (наприклад, через повідомлення або шину подій) та механізм координації (централізований або децентралізований). У межах OSINT MAS можуть працювати як у статичному режимі (обробка за сценарієм), так і в реактивному режимі (адаптація до зміни вхідних даних).

Однією з основних переваг MAS є можливість паралельного виконання задач та мінімізація залежності від єдиної точки відмови. Це критично важливо у сфері кібербезпеки, де обсяги даних швидко зростають, а швидкість реагування може бути вирішальною. Крім того, MAS дозволяють комбінувати агенти з різним рівнем інтелектуальності – від простих фільтрів до LLM-керованих аналітиків.

Порівняння архітектур одноагентного та мультиагентного типу в системах OSINT дозволяє виявити ключові відмінності в ефективності, масштабованості й гнучкості роботи. У моделі з одним агентом уся аналітична робота виконується єдиною моделлю – від отримання запиту до генерації відповіді. Такий підхід простий у реалізації та зручний для завдань з обмеженою складністю.

Проте одноагентна архітектура має суттєві обмеження, наприклад, складність в обробці розподілених задач, відсутність спеціалізації, вузька гнучкість у адаптації до нових умов. У багатьох випадках вона не забезпечує належного рівня прозорості та масштабування.

Мультиагентні системи, навпаки, дозволяють розподіляти функції між кількома агентами з чіткими ролями. Це сприяє побудові логічно зрозумілих потоків, підвищує відмовостійкість, полегшує діагностику й дозволяє здійснювати паралельну обробку даних. Завдяки чіткому розмежуванню завдань такі системи легше масштабуються, інтегруються з новими джерелами та адаптуються до змін у поведінці загроз.

У таблиці 1.3 подано порівняння одноагентного та мультиагентного підходів у контексті архітектур OSINT-систем, що дозволяє оцінити їх ефективність, масштабованість і стійкість до відмов у реальних сценаріях.

Таблиця 1.3 – Порівняння одноагентної і мультиагентної архітуктур

Критерій	Одноагентна архітектура	Мультиагентна архітектура
Простота реалізації	Висока, один LLM виконує всі задачі	Складніша, потребує координації кількох агентів
Гнучкість	Обмежена, важко адаптувати під нові задачі	Висока, можна змінювати ролі та сценаріїв
Масштабованість	Низька, один агент має обмеження	Висока, можливо додати більше агентів
Продуктивність	Обмежена паралельність	Паралельна робота кількох агентів
Відповідальність	Централізована, один агент робить усе	Розподілена, кожен агент має свою функцію
Відмова компонентів	Вразлива до збоїв	Більш стійка, агенти можуть заміщувати один одного
Складність налагодження	Низька	Вища, потребує контролю сценаріїв

У процесі побудови OSINT-систем на основі штучного інтелекту важливим є вибір відповідного фреймворка, що визначає спосіб реалізації агентної логіки, масштабованості та взаємодії між компонентами. Сучасні фреймворки поділяються на одноагентні та мультиагентні – залежно від кількості задіяних LLM-агентів і ступеня розподілення функцій. Серед поширених фреймворків можна виділити CAMEL, MetaGPT, CrewAI, LangChain Agents і AutoGen.

LangChain Agents – це фреймворк, що дозволяє створювати агентів на основі LLM із доступом до зовнішніх інструментів, API, баз даних та логіки управління. Його підхід базується на концепції агентів, які можуть отримувати завдання, планувати дії, запускати інструменти та генерувати звіти. LangChain Agents ідеально підходить для побудови одноагентних сценаріїв розвідки, інтегрованих у бізнес-логіку або прості OSINT-процеси [30].

CAMEL (Communicative Agents for Mind Exploration of Large Language Models) – це архітектурний підхід, заснований на концепції рольової взаємодії між агентами, який реалізується на базі однієї LLM. CAMEL імітує багатоступеневу міжагентну комунікацію, створюючи структуру «роль + завдання» для покращення логіки генерації відповідей. Хоча він не є повноцінним фреймворком з інструментальною підтримкою, CAMEL широко застосовується як стратегія для розв'язання складних задач у сфері OSINT через покращення аналітичного мислення та симуляцію співпраці [31].

CrewAI – мультиагентна архітектура, яка дозволяє створювати команди спеціалізованих агентів із визначеними ролями, які спільно виконують розвідку. Сценарії взаємодії між агентами будуються у формі послідовностей дій, які узгоджуються відповідно до результатів кожного етапу. CrewAI підтримує розподілення завдань, управління через природну мову та інтеграцію з інструментами, необхідними для OSINT [32].

AutoGen – мультиагентна платформа від Microsoft, орієнтована на реалізацію сценаріїв співпраці між агентами та користувачем (human-in-the-loop). Агенти можуть вести діалоги, делегувати функції одне одному, перевіряти результати й доповнювати логіку розвідки. AutoGen добре підходить для складних дослідницьких задач, де потрібна поступова перевірка даних або генерація гіпотез із залученням користувача [33].

MetaGPT – мультиагентний фреймворк, який імітує командну роботу в стилі організації стартапу, кожен агент виконує професійну роль (наприклад, продакт-менеджер, інженер, аналітик). У контексті OSINT MetaGPT може бути використаний для комплексної обробки запитів, від збору до звітності, з дотриманням структурованого сценарію. Його перевага – чітка рольова координація і можливість масштабування під проєктні задачі [34].

З огляду на аналіз розглянутих фреймворків, вибір на користь CrewAI у межах цієї бакалаврської кваліфікаційної роботи є найбільш обґрунтованим. Його мультиагентна архітектура, зручна логіка побудови сценаріїв взаємодії та підтримка чіткої рольової взаємодії між агентами дозволяють ефективно реалізувати складні OSINT-сценарії без необхідності розробки низькорівневого управління. Додатковими перевагами є інтеграція з LLM, можливість масштабування й простота у впровадженні завдяки готовим шаблонам і гнучким конфігураціям. Саме ці характеристики роблять CrewAI найкращим кандидатом для реалізації інтелектуальної системи розвідки у сфері кібербезпеки.

У фреймворку CrewAI ключову роль відіграє система взаємодії між агентами, яка базується на принципі поділу обов’язків за ролями. Кожен агент виконує чітко визначену функцію у межах аналітичного процесу, що забезпечує структурованість, модульність і гнучкість системи. Наприклад, агенти можуть бути визначені як “Data Collector”, “OSINT Analyst”, “Validation Agent”, “Report Generator” тощо – залежно від задачі, яку виконує система.

Однією з особливостей CrewAI є можливість формування передісторії (backstory) агента, який описує його фахову спеціалізацію, стиль взаємодії, обсяг компетенцій та очікувану поведінку. Це дозволяє створювати більш керовані, послідовні та контекстно-залежні реакції агентів під час виконання сценаріїв.

Логіка взаємодії реалізується через сценарії взаємодії, які задають послідовність виконання дій та передачі інформації між агентами. Наприклад, потік може починатися з агента-збирача, який шукає дані з відкритих джерел, передає їх аналітику для обробки, а потім – редактору для підготовки звіту. У разі виявлення неузгодженостей або неповної інформації, агент може ініціювати зворотний запит або передати завдання іншому учаснику. Такий підхід наближує логіку роботи системи до реальних аналітичних процесів, де рішення приймаються на основі взаємодії між фахівцями.

Фреймворк підтримує як синхронну, так і асинхронну взаємодію між агентами, що дозволяє створювати як прості сценарії лінійного виконання, так і складні паралельні структури з умовним переходом. Наприклад, залежно від оцінки ризику, агент-аналітик може передати дані або для формування звіту, або повторно агенту-збирачу для уточнення джерел. Такий механізм дозволяє ефективно управляти процесом OSINT, підвищуючи якість, релевантність і швидкість реакції системи на запити користувача.

У системах, побудованих за мультиагентною архітектурою, однією з ключових проблем є ефективна координація дій між агентами. Для цього у фреймворку CrewAI використовується підхід, що базується на спеціалізованих предметно-орієнтованих мовах (Domain-Specific Languages, DSL). На відміну від універсальних мов програмування, DSL дозволяє описувати взаємодію, послідовність дій та логіку обробки інформації у зрозумілій, компактній і структурованій формі, орієнтованій на конкретну предметну область – у даному випадку OSINT [35].

DSL у CrewAI використовується для опису так званих сценаріїв взаємодії між агентами. Наприклад, за допомогою кількох інструкцій можна описати, що після отримання даних агент-збирач передає їх аналітику, а той – звітуючому агенту. Інтеграція таких сценаріїв із великими мовними моделями дозволяє використовувати природну мову для конфігурації поведінки агентів, що значно спрощує процес налаштування системи навіть для користувачів без досвіду програмування.

У CrewAI можливе використання як внутрішньої DSL (вбудованої у Python-код), так і зовнішніх представлень сценаріїв – наприклад, у вигляді YAML-файлів. Це робить фреймворк придатним до інтеграції в більш складні аналітичні платформи. Крім того, використання DSL підвищує прозорість та відтворюваність аналітичних процесів, адже кожен сценарій чітко формалізується та може бути повторно використаний або адаптований до нових задач.

Таким чином, DSL у CrewAI виконує роль інтерфейсу між користувачем і мультиагентною системою, дозволяючи створювати складні сценарії взаємодії без глибоких знань у сфері програмування. Це відкриває шлях до впровадження OSINT-рішень у ширше коло застосувань – від оперативного реагування на кіберінциденти до стратегічної розвідки в аналітичних центрах.

1.4 Формалізація вимог і постановка задачі

Розроблюваний програмний засіб орієнтований на фахівців з кібербезпеки, які залучені до процесів інформаційного моніторингу, аналізу загроз, виявлення інцидентів та проведення OSINT-досліджень. Це можуть бути співробітники аналітичних підрозділів, спеціалісти операційних центрів безпеки (SOC), слідчі з кіберзлочинності або експерти, які займаються аудитом інформаційної безпеки. Усі ці фахівці потребують інструментів, які дозволяють ефективно та швидко збирати, обробляти та інтерпретувати великі обсяги відкритої інформації з різноманітних джерел для прийняття рішень, підготовки звітів або документування доказів.

Основна функціональна мета системи – забезпечити автоматизований процес збору OSINT-даних, їх попередньої обробки, нормалізації, фільтрації, виявлення взаємозв'язків між об'єктами та формування узагальнених висновків. У межах системи передбачається можливість дослідження об'єктів різної природи, а саме, осіб, організацій, доменних імен, IP-адрес, телефонних номерів, електронних поштових скриньок, акаунтів у соціальних мережах, криптовалютних гаманців тощо. Інформація про ці об'єкти може бути розпорошена серед різних відкритих джерел, тому система повинна мати можливість працювати з пошуковими системами, спеціалізованими базами даних, публічними реєстрами (зокрема WHOIS), API сторонніх OSINT-сервісів, агрегаторами витоків, соціальними платформами та іншими відкритими вебресурсами.

Важливою вимогою є підтримка мультиагентної архітектури на основі CrewAI, яка дозволяє розділити функціональність системи між спеціалізованими агентами, кожен з яких відповідає за конкретну задачу, наприклад, пошук даних, тематичну класифікацію, перевірку на дублікатність, фільтрацію результатів, аналіз ризиків або формування звітів. Усі агенти функціонують на основі великих мовних моделей, що належать до категорії генеративного штучного інтелекту, завдяки чому здатні адаптивно інтерпретувати запити користувача та працювати з відкритими текстовими даними у природній мові. Така структура забезпечує масштабованість і гнучкість, дає змогу підключати нових агентів без переробки всієї системи. Крім того, взаємодія агентів між собою повинна бути реалізована у вигляді узгодженої послідовності дій, що дозволяє досягти високого ступеня автоматизації та адаптивності до нових типів задач.

Очікується, що система буде інтегруватися з зовнішніми інтерфейсами через API, з дотриманням обмежень на кількість запитів, підтримуючи як безкоштовні ключі для початкового використання, так і можливість масштабування при переході на платні тарифні плани. Інтерфейс користувача має бути простим і інтуїтивним, з можливістю налаштовувати параметри запитів, обирати джерела, зберігати результати пошуку та експортувати звіти. Також важливо враховувати параметри продуктивності, наприклад швидкість обробки запитів, точність побудованих зв’язків, надійність збереження результатів, а також захист персональних налаштувань користувача та історії запитів.

Сформульовані вимоги окреслюють загальну задачу, що полягає у створенні ефективного, адаптивного та функціонально повного програмного засобу для автоматизованої розвідки з відкритих джерел, здатного покращити ефективність роботи фахівців у сфері кібербезпеки, зменшити час на обробку інформації та підвищити точність виявлення потенційно небезпечних об’єктів або індикаторів компрометації.

2 ТЕХНІЧНЕ ПРОЕКТУВАННЯ СИСТЕМИ

2.1 Загальна архітектура системи та взаємодія компонентів

Розроблювана система кіберрозвідки з відкритих джерел ґрунтується на мультиагентному підході, що передбачає поділ функціональності між кількома незалежними, але координованими інтелектуальними агентами. Така модель дозволяє досягти високої гнучкості, масштабованості та точності під час обробки великого обсягу розрізненої інформації. Окремі агенти спеціалізуються на роботі з конкретними типами об’єктів (домен, організація, особа), а також на зборі та узагальненні даних, що дозволяє адаптувати систему під різні аналітичні сценарії.

Архітектура побудована як послідовний ланцюжок, у якому агенти активуються один за одним у фіксованому порядку. Кожен з них виконує окрему роль у загальному процесі обробки даних, від локального аналізу й пошуку в Інтернеті до поглибленого дослідження об’єктів і формування підсумкового звіту. Такий підхід дозволяє уникнути перевантаження центрального процесу й рівномірно розподіляє навантаження між компонентами. Завдяки чітко визначеним межам відповідальності між агентами система зберігає модульність, її легко розширювати новими можливостями без порушення загальної логіки роботи.

Усі агенти працюють на основі налаштованих конфігурацій, що описують їхню роль, ціль, контекстну передісторію та очікувані результати. У процесі виконання проміжні дані кешуються, щоб уникнути повторних звернень до зовнішніх API, зменшити навантаження на мережу та прискорити отримання результату. Ініціація системи здійснюється через командний інтерфейс, а підсумковий результат формується у вигляді структурованого звіту, зручного для перегляду та подальшого аналізу.

Загальна архітектура системи програмного засобу для розвідки з відкритих джерел зображена на рисунку 2.1.

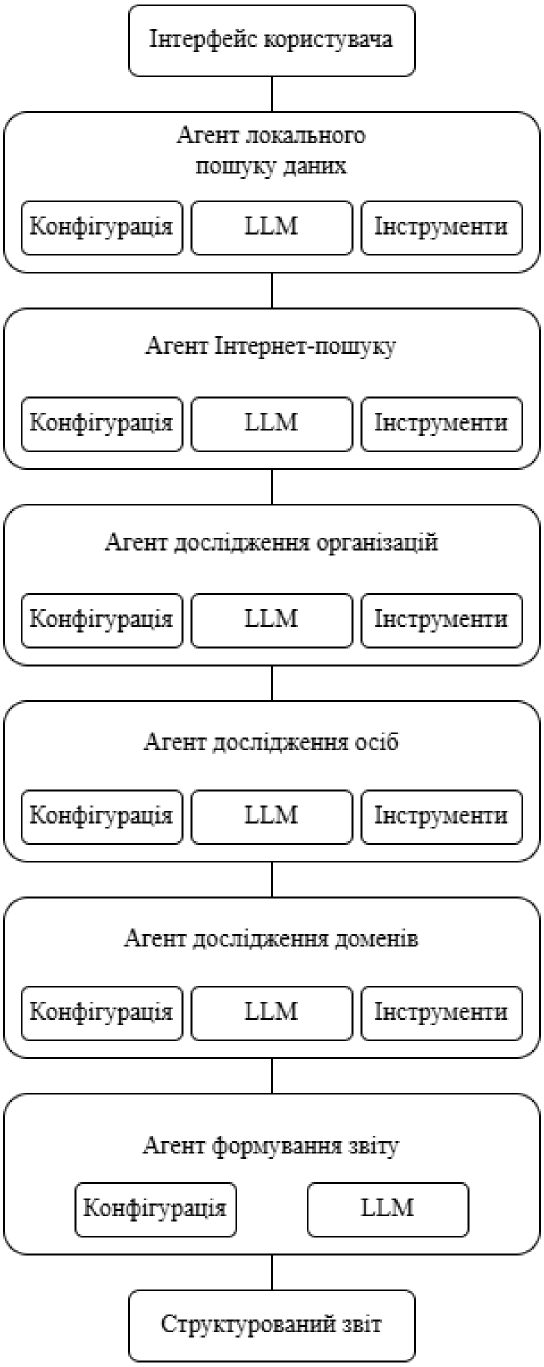


Рисунок 2.1 – Загальна архітектура системи

Архітектура мультиагентної системи включає шість агентів, кожен із яких має власну конфігурацію з описом ролі, цілі, передісторії та очікуваного результату. Для кожного агента задаються окремі інструменти та велика мовна модель, яка забезпечує генерацію відповідей. Агент локального пошуку обробляє внутрішні дані, агент Інтернет-пошуку знаходить релевантну інформацію в мережі, агент дослідження осіб вивчає персональні характеристики, агент дослідження організацій знаходить структуру, активність та зв'язки компаній, агент дослідження доменів виконує технічну аналітику, а агент формування звіту оформлює структурований звіт на основі всієї зібраної інформації.

Алгоритм роботи системи побудований як послідовна взаємодія агентів, де кожен наступний працює на основі результатів попереднього. Починається все з локального збору, потім підключається Інтернет-пошук, після чого дані збагачуються аналітикою про організації та осіб. Завершальним є технічний аналіз домену, після чого дані передаються для фінального формування звіту. Такий підхід забезпечує цілісність дослідження та зручність представлення результатів (рис. 2.2).

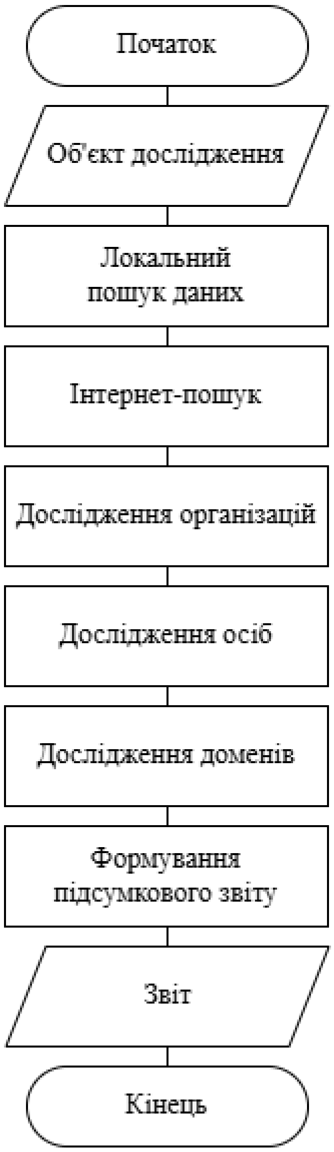


Рисунок 2.2 – Загальний алгоритм роботи програмного засобу

Після отримання запиту від користувача (домен, назва організації або ПІБ особи) процес контролера CrewAI ініціює чітко визначену послідовність роботи агентів. Спочатку виконується агент локального пошуку даних, який відбирає всю наявну внутрішню інформацію та готує первинний контекст для подальшого аналізу. На наступному етапі агент Інтернет-пошуку розширює цей контекст, виконуючи веб-пошук і збираючи публічні згадки та зовнішні джерела. Така почергова обробка дозволяє визначити ключові дані із внутрішніх баз і доповнити їх актуальними зовнішніми даними, перш ніж перейти до глибшої аналітики.

Після збору ключових даних процес переходить до агента дослідження організацій і агента дослідження осіб, які відповідно вивчають корпоративний та персональний аспекти об’єкта. Цей підхід дає змогу спочатку побудувати загальне уявлення про сутність (організацію чи особу), а потім уточнити характеристики на рівні конкретних осіб. Лише після цього запускається агент дослідження доменів, який виконує технічний OSINT на основі вже зібраної контекстної інформації.

Такий лінійний, крок за кроком, алгоритм забезпечує логічну цілісність, дозволяє кожному учаснику процесу оперувати релевантними даними та легко інтегрувати нові агенти або інструменти без порушення загального порядку виконання.

Проміжні дані можуть коригувати поведінку наступних агентів, що підвищує глибину та точність OSINT-дослідження. На завершальному етапі агент формування звіту узагальнює всі фрагменти інформації, усуває дублікати й формує структурований аналітичний висновок. Завдяки чітко налаштованим інструкціям і жорстко визначеній послідовності процес залишається прозорим, цілісним і легко розширюваним без зміни базового робочого ланцюжка.

Операції кожного агента в системі забезпечуються за допомогою великої мовної моделі GPT-4o, яка виступає центральним механізмом генерації відповідей, обробки текстових даних і формування релевантного контенту на основі контексту розвідки. Ця модель поєднує високу якість розуміння природної мови зі швидкою обробкою запитів, що особливо важливо в умовах багатоступеневого аналізу, характерного для мультиагентних систем розвідки з відкритих джерел.

Однією з ключових переваг GPT-4o є її здатність працювати з великими обсягами неструктурованих даних, витягуючи з них релевантні фрагменти та формулюючи узгоджені висновки. Модель ефективно підтримує довготривалі ланцюжки інструкцій, зберігаючи контекст і логіку всередині кожного агента, навіть за умов складних сценаріїв взаємодії. Завдяки зниженій латентності порівняно з попередніми версіями, GPT-4o забезпечує швидке виконання кожного етапу, що прямо впливає на загальну продуктивність системи.

Сумісність GPT-4o із сучасними інструментами CrewAI дозволяє використовувати її без потреби в додаткових адаптаціях, що спрощує інтеграцію та підтримку системи. Така модель виявляється особливо доречною для інтелектуального керування задачами, що вимагають як аналітичного мислення, так і генерації структурованого тексту, зокрема під час формування підсумкового звіту.

2.2 Структура агентів мультиагентної системи

У межах розробленої мультиагентної системи для проведення OSINT-досліджень ключову роль відіграють спеціалізовані агенти, кожен з яких відповідає за окрему задачу в межах єдиного процесу розвідки. Такий підхід дозволяє побудувати розширювану архітектуру, де логіка збору, обробки й аналізу даних не зосереджена в одному централізованому модулі, а децентралізована між окремими автономними сутностями – агентами. Це забезпечує як більшу масштабованість системи, так і її адаптивність до нових типів об’єктів і завдань.

Вибір саме агентного підходу, на відміну від класичних процедурних методів, зумовлений потребою в автоматизованому аналізі відкритих джерел, де характер і структура інформації можуть бути довільними, а обсяг – надзвичайно великим. Завдяки інтеграції великих мовних моделей, кожен агент здатний інтерпретувати запити природною мовою, виконувати логічні висновки та формувати структуровані відповіді на основі неструктурованих вхідних даних.

Таким чином, система може функціонувати не просто як сукупність інструментів, а як інтелектуальна машина для розвідки з відкритих джерел.

Кожен агент у системі описується за кількома ключовими складовими, які визначають його роль, логіку функціонування, контекст застосування та очікувані результати. Першим елементом є роль агента, що формулюється як коротке завдання в стилі “ти – фахівець у певній галузі, виконуєш конкретну функцію”, наприклад, “ти OSINT-аналітик, який досліджує технічні характеристики доменів”. Цей компонент служить як інструкція для моделі, яка керує поведінкою агента під час виконання завдань.

Другим елементом є ціль агента, яка розширює контекст його діяльності та вказує, які саме результати він має досягнути. Зазвичай, вона містить більш деталізоване формулювання, наприклад, “агент повинен зібрати всі технічні дані про домен, включно з DNS-записами, IP-інфраструктурою, сертифікатами тощо”. Таке уточнення дозволяє направити логіку LLM-моделі в межах очікуваної предметної області.

Третім компонентом виступає передісторія агента, яка задає мотиваційний або професійний передісторія агента. У контексті роботи LLM вона використовується як додатковий вектор для формування стилю, глибини аналізу та прийомів роботи агента. Наприклад, можна вказати, що агент є експертом із цифрової інфраструктури, який має досвід роботи з інструментами для розвідки з відкритих джерел.

Окремо визначається велика мовна модель, яка використовується для обробки завдань – у даній роботі всі агенти побудовані на основі моделі gpt-4o, яка забезпечує високу якість генерації відповідей за відносно низьких обчислювальних витрат.

Завершальним елементом структури агента є задача, яка формалізує його конкретну дію в рамках сценарію виконання розвідки. Завдання можуть містити як інструкцію до дій, так і опис очікуваного результату, наприклад, “здійснити аналіз домену і сформувати структурований звіт, що містить WHOIS-дані, DNS-записи, IP-репутацію тощо”. Завдяки такому рівню деталізації можна досягнути узгодженого функціонування всіх агентів в єдиному процесі.

У таблиці 2.1 для кожного з шести агентів вказано його роль, мету, передісторію, завдання та очікуваний вихід.

Таблиця 2.1 – Характеристика агентів мультиагентної системи

Назва агенту	Роль	Мета	Передісторія	Задача	Вихід
Агент локального пошуку даних	Локальний дослідник даних	Зібрати всі релевантні записи з локальних CSV-джерел	Експерт із роботи з приватними базами даних	Пошук і нормалізація даних за ПІБ, ЄДРПОУ, email, телефоном	Текстовий блок із записами (назва/ПІБ, адреса, контакти)
Агент Інтернет-пошуку	Спеціаліст із веб-пошуку	Зібрати актуальні згадки з публічних інтернет-джерел	Досвідчений OSINT-розвідник веб-контенту	Інтернет-пошук з Google Dorks, Serper, фільтрація результатів	Резюме з ключовими URL і короткими описами
Агент дослідження організацій	Аналітик організацій	Оцінити структуру, реєстраційні та фінансові дані компанії	Аналітик корпоративної інформації	Збір реєстраційних відомостей, складання власницьких структур	Список реєстраційних даних, структури власності, згадки
Агент дослідження осіб	Аналітик персон	Сформувати цифровий портрет фізичної особи	Спеціаліст із розвідки персон через OSINT	Пошук соцпрофілів, перевірка email, санкційних списків	Звіт із соцпрофілями, email-адресами та санкційними перевірками
Агент дослідження доменів	Технічний OSINT-аналітик	Провести технічний аудит домену	Експерт із TECHINT	Збір технічних даних домену	Технічний звіт з аналізу домену
Агент формування звіту	Старший профільний аналітик	Об’єднати всі результати в єдиний OSINT-звіт	Аналітик із глибокою перевіркою й узагальненням	Агрегація даних, формування звіту	Повний структурований OSINT-звіт

Агент локального пошуку даних виступає в ролі агента з привілейованим доступом до внутрішніх CSV-баз та інших приватних сховищ інформації. Він спеціалізується на швидкому пошуку по структурованих таблицях, вміє фільтрувати дані за різними ключами, наприклад, ПІБ, код ЄДРПОУ, електронна пошта, телефон, та забезпечує попередню нормалізацію результатів для подальшої обробки. Завдяки глибоким знанням формату локальних джерел і досвіду роботи з великими обсягами даних агент гарантує високу точність і цілісність первинного контексту розвідки.

Задача цього агента полягає в пошуку всієї доступної інформації про заданий об'єкт у внутрішніх базах. Він видає єдиний текстовий блок, що містить перелік знайдених записів у стандартизованому вигляді. Отримані дані далі використовуються як вхідні параметри для інших агентів системи, дозволяючи їм глибше аналізувати та перевіряти підтверджені факти.

Агент Інтернет-пошуку володіє навичками складання влучних OSINT-запитів і парсингу публічних ресурсів через API та спеціальні інструменти. Він уміє формулювати та оптимізувати пошукові стратегії для знаходження релевантних новин, статей, соціальних згадок та офіційних публікацій у відкритих джерелах, гарантуючи високу релевантність і вичерпність результатів.

Задача цього агента полягає у проведенні всебічного інтернет-пошуку за темою розвідки. Він повертає стисле резюме знайдених матеріалів із ключовими посиланнями та короткими описами, яке включає заголовки, уривки тексту та URL. Отриманий звіт стає основою для подальшої роботи профільних агентів, даючи їм можливість фокусуватися на глибокому аналізі й інтеграції зовнішніх даних у фінальний OSINT-звіт.

Агент дослідження домену виконує функцію технічного OSINT-аналітика, зосередженого на дослідженні цифрової інфраструктури, пов'язаної з заданим доменом. Його роль полягає у збиранні повного спектру технічної інформації, зокрема DNS-записів, SSL-сертифікатів, IP-інфраструктури, PTR-записів, підмереж, піддоменів та інших характеристик. Основна мета агента – сформувати повний технічний профіль об'єкта дослідження, який дозволяє оцінити особливості конфігурації інфраструктури, можливі вектори ризику та загальний рівень прозорості цифрових слідів. У передісторії агент постає як фахівець з цифрової безпеки, що має значний досвід у роботі з відкритими технічними джерелами та глибоке розуміння мережевих структур. У своїй роботі агент використовує велику мовну модель gpt-4o, що забезпечує здатність логічно структурувати інформацію, розпізнавати приховані зв'язки та формулювати висновки природною мовою [36].

Задача агента передбачає послідовне виконання кількох етапів дослідження. Зокрема, здійснюється запит до реєстру WHOIS, отримуються всі наявні DNS-записи (A, MX, NS, TXT, SOA та інші), аналізується конфігурація SSL-сертифіката, виконується перевірка PTR-записів, а також ідентифікується належність IP-адреси до певної підмережі. Додатково досліджуються геолокація, організаційна приналежність IP, а також потенційна наявність фаєрволу чи специфічної інфраструктури. Очікуваним результатом є структурований аналітичний звіт, який відображає повну технічну картину домену та може слугувати основою для подальшого аналізу цифрової присутності об’єкта [37].

Агент дослідження організацій відповідає за збір, обробку та представлення відкритих даних про юридичних осіб, що можуть мати значення з точки зору кіберрозвідки. У межах своєї ролі він аналізує ключові аспекти, а саме, реєстраційні відомості (назва, код ЄДРПОУ, юридична адреса, дата створення, організаційно-правова форма), дані про керівництво та засновників, зв’язки з іншими компаніями, судову історію, податкову репутацію та публічну активність. Особливу увагу приділено фінансовим взаємозв’язкам (участь у тендерах, державних закупівлях), наявності ліцензій, дозволів і згадкам в офіційних реєстрах. Агент також перевіряє наявність домену або інших цифрових активів, що належать організації, аналізує їхню цифрову присутність та репутацію.

Його основна задача полягає у формуванні комплексного OSINT-звіту щодо об'єкта розвідки – організації, зазначеної користувачем. У межах цієї задачі агент виконує автоматичний пошук по державних реєстрах, агрегаторах відкритих даних та новинних джерелах, виявляє прямі та опосередковані зв’язки організації з іншими суб’єктами, а також перевіряє наявність у санкційних списках або базах витоків даних. Отримана інформація дозволяє сформувати загальне уявлення про структуру, діяльність, репутаційні ризики та потенційні загрози, пов’язані з діяльністю організації. Агент використовує велику мовну модель GPT-4o для генерації відповідей, узагальнень і побудови логічних висновків.

Агент дослідження осіб відповідає за формування повного цифрового портрету фізичної особи на основі відкритих джерел. У межах своєї ролі він аналізує як структуровані, так і неструктуровані дані для виявлення ключової персональної інформації – повного імені, місця проживання, освіти, місця роботи, а також онлайн-активності. Цей агент здатен виявляти соціальні профілі (LinkedIn, Facebook, GitHub тощо), знайдені email-адреси, псевдоніми, номерні записи, інформацію з баз витоків або згадок у медіа. Крім того, він може виявляти можливі зв’язки з організаціями або іншими особами, що є цінним для OSINT-аналізу у контексті кіберзагроз.

Задача агента полягає у виявленні, систематизації та стислому викладі всієї релевантної інформації про особу, яку зазначив користувач. Його ціллю є побудова аналітичного звіту, що включає як основні ідентифікаційні відомості, так і результати пошуку по соціальних мережах, публічних реєстрах та витоках даних. Результати представлені у вигляді узагальненого тексту з посиланнями на джерела або вказівками на знайдені профілі, що забезпечує можливість подальшого розслідування. Агент використовує велику мовну модель GPT-4o для інтерпретації запиту та гнучкого реагування на контекст, включно з неоднозначно сформульованими запитами типу "знайди все про Івана Іванова з Києва".

Агент формування звіту виконує роль аналітика, який об’єднує результати, отримані всіма іншими агентами, у цілісний і зрозумілий для користувача звіт. Його головне завдання – зіставити зібрані дані, усунути дублікати, підкреслити найважливіші аспекти, виявити закономірності або зв’язки між об'єктами дослідження (наприклад, спільні контактні дані, прив’язка до однієї організації чи IP-інфраструктури). Цей агент функціонує як завершальний етап розвідки, забезпечуючи логічно структуровану відповідь, що може бути використана як для подальшого аналізу, так і для звітності.

У межах поставленої задачі агент формування звіту здійснює контекстний аналіз усіх зібраних результатів, перетворюючи їх у формалізований OSINT-звіт. Він здатен порівнювати, узагальнювати та формулювати висновки природною мовою, дотримуючись об’єктивності та логічної послідовності. У результаті користувач отримує узгоджене текстове представлення результатів розвідки по одному або декількох об’єктах. Для генерації такого звіту агент використовує LLM

GPT-4o, що дозволяє адаптивно обробляти як структуровані, так і фрагментарні дані, отримані від інших агентів.

2.3 Інструменти для дослідження об'єктів

Для забезпечення ефективного функціонування мультиагентної системи розвідки з відкритих джерел кожен агент використовує набір інструментів, які відповідають його спеціалізації та завданням. Ці інструменти виконують роль каналів доступу до відкритих джерел інформації, дозволяючи отримувати, обробляти та структурувати дані, релевантні до об’єкта дослідження. У зв’язці з агентом інструменти відіграють роль «сенсорів», які забезпечують вхідні дані для наступного етапу аналітичного процесу.

Підбір інструментів здійснювався з урахуванням вимог відкритості, безкоштовного доступу, можливості автоматизації та релевантності отриманих результатів. Особливу увагу було приділено забезпеченню високої точності й достовірності даних, що мають вирішальне значення в процесі аналізу цифрового сліду, репутації, технічної інфраструктури об'єкта тощо. Крім того, важливою вимогою стала стабільність роботи обраних API та наявність зрозумілої документації для їх інтеграції. Усі інструменти реалізовані у вигляді окремих модулів, що взаємодіють із відповідними агентами.

Завдяки модульній структурі інструменти можна швидко оновлювати або замінювати без втручання в основну логіку агента. Це дозволяє адаптувати систему до змін у зовнішніх API, появи нових джерел або потреби в розширенні функціональності. Такий підхід забезпечує гнучкість архітектури та полегшує масштабування системи при зростанні обсягів даних або кількості об'єктів для дослідження.

У таблиці 2.1 представлено перелік використаних інструментів, їх призначення та об’єкт дослідження.

Таблиця 2.2 – Розподіл інструментів за об’єктами дослідження

Засоби дослідження	Призначення	Об’єкт дослідження
WHOIS Lookup	Отримання реєстраційних даних домену або IP.	Домен
DNS Lookup	Перегляд DNS-записів домену.	Домен
SSL Certificate	Аналіз SSL-сертифіката сайту.	Домен
IPInfo API	Отримання даних про IP, геолокація, ASN, провайдер.	Домен
Nmap	Сканування портів, служб, ОС, фаєрволів.	Домен, організація
PTR Lookup	Зворотна перевірка IP на доменне ім’я.	Домен
IP Subnetwork Tool	Визначення мережі, підмережі та її власника.	Домен
Subdomain Finder	Виявлення піддоменів організації.	Домен, організація
DataGovUA Lookup Tool	Пошук державних даних про організацію.	Організація
Hunter.io	Пошук email-адрес, пов’язаних з особою.	Особа
ServAPI	Перевірка наявності акаунтів за email або username.	Особа
OpenDataBot	Отримання публічних даних про особу (ФОП, майно, суди).	Особа, домен, організація
Google Dorks	Специфічний пошук згадок про об'єкт у відкритих джерелах.	Особа, домен, організація,
SerperDev	Швидкий пошук іглибоке сканування результатів через Serper API	Особа, домен, організація,
ScrapeWebsiteTool	Отримання структурованих даних зі сторінок веб-сайтів	Особа, домен, організація,
WebsiteSearchTool	Пошук контенту всередині заданих сайтів із фільтрацією за ключами	Особа, домен, організація,
CSVSearchTool	Пошук інформації за ключами в локальних CSV-базах	Особа, домен, організація,

Використання кожного з інструментів тісно пов’язане зі сферою компетенції відповідного агента, його функціональним призначенням та поставленими завданнями. Їх поєднання дозволяє формувати комплексне уявлення про об’єкт розвідки з різних ракурсів.

Агент локального пошуку даних спеціалізується на роботі з внутрішніми таблицями та базами CSV, що містять інформацію про фізичних осіб і організації. Його головна мета – побудувати повний контекст дослідження, формуючи первинний набір даних із всіх доступних локальних джерел. Для цього агент застосовує інструмент CSVSearchTool, що дозволяє виконувати багаторівневий пошук за різними полями (ПІБ, код ЄДРПОУ, email, телефон) і фільтрацію за власними критеріями [38].

Для забезпечення роботи агента локального пошуку можуть використовуватись загальнодоступні набори відкритих даних, які публікуються на спеціалізованих платформах, таких як Єдиний державний вебпортал відкритих даних. Ці ресурси містять структуровану інформацію по різних напрямках: реєстри фізичних та юридичних осіб, звітність, фінансування, інфраструктура тощо. Дані з таких джерел можуть бути завантажені у форматі CSV, збережені локально в межах програмного засобу та використані для попереднього аналізу або перехресної перевірки. Це дозволяє масштабувати систему, розширюючи обсяг внутрішньої аналітики — наприклад, шляхом підключення великих таблиць із десятками тисяч записів, які можуть бути розміщені як локально, так і на спеціалізованих сховищах. Такий підхід відкриває перспективи для подальшого розвитку програмного засобу, зокрема в напрямку глибшої інтеграції з масивами урядових та галузевих даних [39].

Агент Інтернет-пошуку спеціалізується на зборі публічних даних із відкритих веб-джерел для доповнення локального контексту. Його головна мета – отримати актуальні згадки, новини та інші зовнішні відомості про об’єкт дослідження за допомогою SerperDevTool, WebsiteSearchTool і ScrapeWebsiteTool [40-42].

Спочатку агент формує цілеспрямовані пошукові запити, аналізує результати SerperDevTool для виявлення релевантних URL і метаданих, а потім фільтрує контент за ключовими словами. За необхідності здійснює збір і структурування даних із конкретних веб-сторінок. В результаті відбираються найважливіші фрагменти тексту, заголовки та посилання, які агент об’єднує в стисле резюме з ключовими URL і короткими описами.

Агент дослідження домену спеціалізується на вивченні технічної інфраструктури, що пов’язана з доменом, зазначеним у запиті. Його головна мета – це сформувати технічно насичений портрет цифрової присутності об'єкта, з урахуванням усіх можливих відкритих джерел. Для досягнення цієї мети агент застосовує низку інструментів, що дозволяють провести багаторівневу перевірку мережевої конфігурації. Зокрема, DNS-запити дають змогу виявити всі активні записи домену, а саме, A, AAAA, MX, NS, TXT, SOA, SRV. Це дозволяє оцінити типи активних сервісів, поштові політики безпеки, резервні вузли та інфраструктурні вказівки. SSL-сертифікат аналізується на предмет відповідності, терміну дії, алгоритмів і альтернативних імен, що допомагає виявити додаткові піддоменні ресурси. Через WHOIS-запити та IP-аналітику агент отримує інформацію про реєстранта, організацію-власника, країну, DNSSEC, PTR-записи, CIDR-діапазон IP, хостинг-провайдера та контактні дані.

Окрім базового збору, агент здійснює глибше сканування інфраструктури за допомогою Nmap, що дозволяє визначити відкриті порти, працюючі служби, тип операційної системи, наявність фаєрволу або проксі. Спеціалізовані інструменти для пошуку піддоменів дозволяють ідентифікувати пов’язані ресурси, які часто залишаються прихованими, але можуть бути ключовими для розуміння архітектури компанії. У сукупності, цей набір інструментів дає змогу агенту побудувати повну картину цифрової інфраструктури домену, виявити потенційні вектори атаки або цифрові сліди, що заслуговують на подальше вивчення [43].

Агент дослідження організацій відповідає за глибоке профілювання компаній, установ або інших юридичних осіб, що є об’єктами OSINT-аналізу. Основна мета цього агента – зібрати з відкритих джерел максимально повний і структурований портрет організації, що включає як базові реєстраційні дані, так і розширену інформацію про цифрову, кадрову та фінансову присутність. На відміну від технічного аналізу доменів, цей агент працює переважно з публічними базами даних, соціальними мережами, аналітичними платформами й веб-ресурсами.

Перший рівень дослідження охоплює офіційні реєстри, що дозволяють встановити юридичну особу, дату реєстрації, керівництво, засновників, ліцензії та держконтракти. Такі дані є критично важливими для верифікації компанії, виявлення пов’язаних структур і оцінки рівня прозорості.

На наступному етапі агент переходить до цифрової присутності організації. За допомогою інструментів типу Google Dorks або сервісів мережевої розвідки він може виявити згадки про компанію в документах, презентаціях, оголошеннях або звітах. Використовуючи аналітику соціальних мереж, агент ідентифікує працівників компанії, аналізує їхню активність, що дозволяє оцінити структуру підрозділів або потенційні вектори соціального інжинірингу. У випадку, коли компанія має значну цифрову інфраструктуру, агент також може залучати інструменти типу DNSdumpster чи сканери піддоменів і портів, що дозволяє поєднати профіль організації з технічними характеристиками. У результаті діяльності агента формується всебічний профіль організації, який включає як структуру та цифрову видимість, так і потенційні ризики для кібербезпеки .

Агент дослідження осіб відповідає за формування повного цифрового портрету фізичної особи на основі відкритих джерел. Він використовує Google Dorker для пошуку згадок про ім’я чи прізвище, Hunter.io для виявлення та валідації професійних email-адрес, а також OpenSanctions для перевірки наявності в санкційних і PEP-списках. У ході роботи агент аналізує варіанти імені, місце роботи та проживання, соціальні профілі й публічні згадки, після чого повертає звіт, що містить перелік знайдених акаунтів, email-адрес із оцінкою довіри та результати санкційної перевірки, забезпечуючи структуровану й готову до подальшої інтеграції інформацію для фінального OSINT-звіту [44].

Агент формування звіту виконує роль фінального аналітика, який узагальнює та гармонізує результати від усіх попередніх агентів. Він не проводить нових пошукових операцій і не викликає API, а зосереджується на аналізі й синтезі вже зібраних даних, приводить їх до єдиного формату, усуває дублікати, групує інформацію за тематичними розділами й виокремлює ключові факти. Далі агент застосовує логічні правила й контекстні зв’язки між результатами доменного, організаційного та персонального досліджень, щоб виокремити найважливіші висновки та потенційні ризики.

У підсумковому звіті дані подані у структурованому вигляді з чітко визначеними розділами, технічний аналіз домену, корпоративний профіль організації, цифровий портрет особи, результати веб-пошуку й локальні дані. Завдяки такій організації користувач бачить повну картину дослідження без зайвих деталей – лише ту інформацію, яка безпосередньо стосується поставлених завдань.

Такий підхід гарантує, що кінцевий документ містить узгоджені, достовірні та комплексні висновки без інформаційного шуму чи надбудов, дозволяючи ефективно використовувати звіт для прийняття рішень, підготовки доповідей або подальшого глибинного аналізу.

Застосування інструментів OSINT у рамках мультиагентної архітектури дозволяє досягти високої точності, масштабованості та швидкості під час збору відкритих даних. Кожен агент використовує набір інструментів, підібраний відповідно до його спеціалізації, що забезпечує глибину та повноту дослідження у своїй сфері відповідальності, чи то доменна структура, юридична особа або цифрова активність особи. Така модульність та чіткий розподіл функцій між агентами дозволяє підвищити якість кінцевого звіту, уникнути дублювання інформації та зосередити аналітичні ресурси там, де це справді потрібно. Завдяки цьому система забезпечує ефективну кіберрозвідку на основі відкритих джерел без потреби у складній ручній роботі чи дорогих платних рішеннях.

3 РОБОЧЕ ПРОЕКТУВАННЯ ЗАСОБУ

3.1 Програмна реалізація засобу розвідки з відкритих джерел

Згідно з офіційною документацією CrewAI, розміщеною на сайті проекту, за допомогою CLI-інтерфейсу є можливість автоматично створити базовий каркас мультиагентної системи. Такий каркас включає всі необхідні компоненти, точки входу, конфігураційні файли, заготовки для інструментів і схему зберігання секретів. Це значно прискорює початковий етап розробки, оскільки дозволяє відразу перейти до реалізації логіки агентів та інтеграції OSINT-інструментів, не витрачаючи час на ручне налаштування структури проєкту [45].

```
python --version
irm https://astral.sh/uv/install.ps1 | iex
uv tool install crewai
crewai create crew my_osint_project
pip install -r requirements.txt
```

Спочатку перевіряється і підтверджується відповідність інтерпретатора Python версійному діапазону 3.10-3.13. Наступним кроком через PowerShell встановлюється утиліту UV, яка використовується для керування CLI-інструментами. За її допомогою інсталюється сам інтерфейс CrewAI, що надає можливість виконувати високорівневі команди для створення та керування «crew». Виклик crewai create crew my_osint_project формує типовий скелет проєкту, файли main.py та crew.py, директорію tools/, шаблони agents.yaml і tasks.yaml, а також створюється файл .env для безпечного зберігання API-ключів. Завершальний крок – встановлення всіх Python-залежностей і налаштування середовища для подальшої розробки та тестування мультиагентної OSINT-системи.

Після успішного створення базового каркасу система готова до подальшого налаштування та розширення. Розробник має змогу редагувати заздалегідь підготовлені шаблони файлів, такі як crew.py, main.py, agents.yaml, tasks.yaml, а також створювати власні інструменти в директорії tools/. Це дозволяє реалізувати індивідуальну логіку роботи агентів, відповідно до тематики дослідження. Таким чином, вихідна структура проекту виступає лише стартовою точкою, яку можна гнучко адаптувати до різних сценаріїв розвідки з відкритих джерел, зокрема за об’єктами типу організація, домен або фізична особа. У процесі розробки формуються власний набір агентів і задач, підключаються необхідні зовнішні бібліотеки, додаються інтерфейси для взаємодії з API та будується повноцінна мультиагентна система, здатна виконувати складні операції збору й аналізу інформації з відкритих джерел (рис. 3.1).

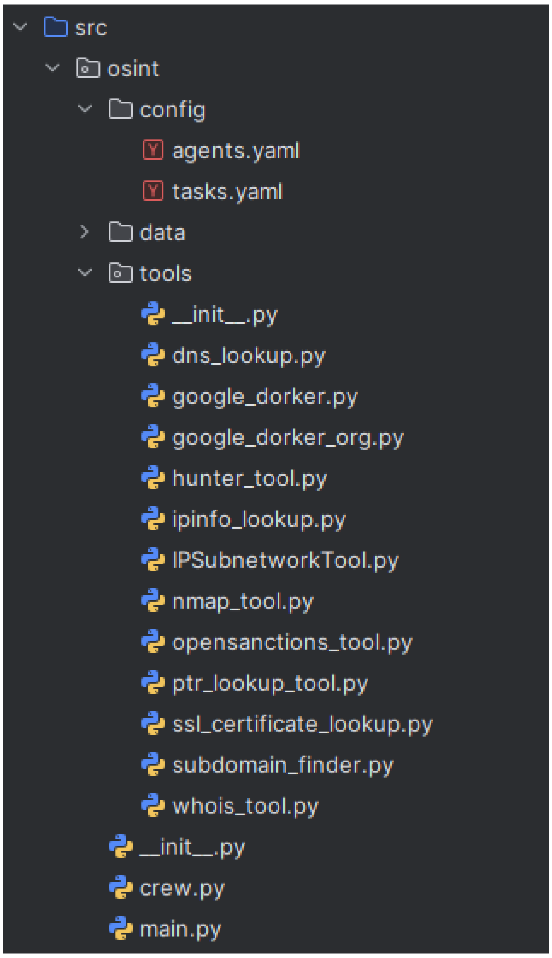


Рисунок 3.1 – Структура проекту розвідки з відкритих джерел

Директорія src/osint містить логічно впорядковану структуру компонентів, необхідних для функціонування мультиагентної системи. Вона включає в себе окремі файли та підкаталоги, що відповідають за конфігурацію, логіку роботи агентів, задачі, а також засоби збору даних.

Файл crew.py є визначальним елементом усієї архітектури, саме в ньому описано логіку формування агентської команди, порядок виконання задач та перелік агентів. У цьому файлі визначається структура взаємодії між компонентами системи, включаючи активовані агенти та їх пов’язані завдання.

Файл main.py виконує роль точки запуску програми. Він містить сценарії активації агентської взаємодії, обробки параметрів запуску та виклику відповідних режимів роботи, таких як тестування, тренування або повний запуск мультиагентної системи.

Папка config містить конфігураційні файли agents.yaml і tasks.yaml, у яких текстово описуються характеристики агентів (роль, мета, поведінка) та задач (умови виконання, очікуваний результат, зв’язок із конкретним агентом). Завдяки використанню YAML-формату, конфігурація системи може змінюватися без необхідності втручання в програмний код.

Папка tools містить реалізації інструментів, які агенти можуть використовувати для збору й обробки інформації. Кожен Python-файл у цій директорії реалізує окремий інструмент, наприклад, взаємодію з WHOIS-сервісами, сканування портів через Nmap або зчитування DNS-записів. Інструменти мають уніфікований формат, що дозволяє легко додавати нові методи розвідки.

Окрему функцію виконує директорія data. У ній зберігаються всі локальні набори даних, які можуть бути використані агентами під час дослідження. Наприклад, CSV-файли з попередньо зібраною інформацією або специфічні реєстри, до яких агент може звертатися без необхідності зовнішніх запитів.

Ця структура забезпечує чіткий поділ обов’язків між модулями, що є важливим для підтримки, масштабування та адаптації системи до різних сценаріїв OSINT-дослідження.

Файл main.py містить набір функцій для взаємодії з мультиагентною системою в різних режимах. Зокрема, реалізовано окремі методи для запуску повного процесу розвідки, навчання агентів, повторного прогону з певного етапу та тестування системи з оцінкою результатів. Кожна з функцій дозволяє виконати відповідний сценарій через просту CLI-взаємодію.

Метод запуску розвідки реалізований таким чином, що користувач може передати об'єкт дослідження безпосередньо через командний рядок. Це дозволяє запускати систему з будь-яким новим запитом без потреби змінювати код вручну. Спочатку програма перевіряє, чи передано хоча б один аргумент (наприклад, run). Якщо аргумент відсутній, виводиться підказка про правильне використання команди.

```
 if len(sys.argv) < 2:
    print("Використання:   python   main.py   [run|train|test|replay]
[аргументи...]")
    sys.exit(1)

command = sys.argv[1]

if command == "run":
    if len(sys.argv) >= 3:
        topic = sys.argv[2]
    else:
        topic = "Вінницький національний технічний університет"
    inputs = {
        "topic_of_research": topic,
        "current_year": str(datetime.now().year)
    }
    Osint().crew().kickoff(inputs=inputs)
```

Якщо в команді run додатково вказано об'єкт дослідження (наприклад, ПІБ, назву організації чи домен), він зчитується як аргумент і зберігається у змінній topic. Якщо об'єкт не передано, використовується значення за замовчуванням. У підсумку створюється словник inputs, який передається всій системі для запуску розвідки, а параметр topic_of_research відіграє роль ключового фокуса дослідження.

Файли agents.yaml та tasks.yaml, які утворюють окремий конфігураційний рівень управління поведінкою мультиагентної системи. У agents.yaml задається «портрет» кожного програмного агента – його роль у дослідженні, кінцева мета, стисло сформульована професійна передісторія та, за потреби, додаткові параметри. Файл tasks.yaml описує сценарії роботи, формулювання конкретних завдань, формат очікуваного виходу й прив’язку кожної задачі до відповідного агента.

Під час запуску CrewAI послідовно зчитує ці YAML-файли, перетворює кожен текстовий блок на об’єкти Python і автоматично підставляє їх у методи, позначені анотаціями @agent та @task. Відокремлення конфігураційного рівня від коду дає змогу змінювати ролі або порядок дій, редагуючи лише YAML-файли, без модифікації програмних модулів. Такий підхід спрощує супровід, дозволяє оперативно перевіряти різні сценарії OSINT-розвідки і водночас виконує документаційну функцію, адже вся логіка системи зберігається у зрозумілому форматі.

```
internet_researcher:
  role: >
    Спеціаліст з Пошуку в Інтернеті.
  goal: >
    Знайти в інтернеті публічну інформацію, новини та згадки про задану особу чи організацію.
  backstory: >
    Ти – експерт з OSINT. Вмієш складати влучні пошукові запити для знаходження релевантних публічних даних, новинних статей, соціальних профілів та офіційних записів онлайн.
```

Role формулює коротку професійну ідентичність агента і слугує «заголовком» його компетенції. У секції goal чітко описується кінцевий результат діяльності – агрегування публічних відомостей про об’єкт розвідки з урахуванням різних типів контенту (новини, соцмережі, офіційні сайти). Розділ backstory задає стислий контекст експертності. Агент позиціонується як фахівець OSINT, що володіє методами тонкого пошукового запиту й здатен оперативно відфільтровувати релевантні джерела. Саме ці три параметри під час ініціалізації автоматично передаються у конструкцію Agent у crew.py, формуючи промпт для мовної моделі й визначаючи стиль, глибину та тональність генерованих відповідей.

Файл tasks.yaml – це документ, у якому текстово задається сценарій роботи всієї мультиагентної системи. У ньому послідовно перелічено задачі, які повинні виконати агенти, від первинного пошуку в локальних даних і мережі до фінального узагальнення результатів.

Такий формат дає змогу змінювати порядок кроків або додавати нові етапи, редагуючи лише YAML-конфігурацію, без втручання у Python-код. Під час запуску CrewAI зчитує цей файл, перетворює кожен текстовий блок на об'єкт Task і далі передає їх у роботу відповідним агентам згідно з обраним режимом процесу.

```
internet_search_task:
  description: >
    Провести ґрунтовний пошук в Інтернеті за {topic_of_research}.
    Знайти публічні записи, новинні статті, офіційні сайти, огляди та згадки в соціальних мережах.
    Підсумувати результати та надати ключові посилання
  expected_output: >
    Коротке резюме знайденої в Інтернеті інформації з важливими URL.
  agent: internet_researcher
```

Поле description формулює завдання природною мовою, включаючи змінну {topic_of_research}, що підставляється під час запуску. Розділ expected_output встановлює вимогу до результату, агент має видати стислий огляд із добіркою найважливіших посилань. Параметр agent напряму пов'язує цю задачу з виконавцем internet_researcher, визначеним раніше у agents.yaml. Саме цей агент отримує промпт, звертається до Serper API, агрегує знайдені джерела й повертає узагальнений підсумок у потрібному форматі.

Решта агентів та пов'язаних із ними завдань структуровано аналогічно, для кожного у agents.yaml визначено роль, мету й контекст, а в tasks.yaml – опис операції, очікуваний вихід і ідентифікатор виконавця. Повний перелік цих конфігураційних записів, разом із форматами результатів, наведено в додатку В.

У файлі crew.py реалізовано базову структуру мультиагентної системи, яка визначає конфігурацію та взаємозв'язок усіх агентів і задач у межах програмного засобу. Цей файл виконує роль центрального координаційного модуля, що інкапсулює основні компоненти системи: перелік агентів, набір задач і логіку їхньої послідовної активації. Завдяки декоратору @CrewBase, визначена структура автоматично реєструється у фреймворку CrewAI, що забезпечує зручне управління всіма етапами роботи, від ініціалізації до обробки результатів.

```
@CrewBase
class Osint():
    """Osint crew"""

    agents: List[BaseAgent]
    tasks: List[Task]
```

Спеціальна анотація @CrewBase, запропонована фреймворком CrewAI, слугує сигналом компілятору, що всередині цього класу буде декларовано склад агентів і послідовність їхніх завдань. Завдяки такій позначці фреймворк під час ініціалізації автоматично зчитує решту методів класу, формує реєстр агентів і зв'язує їх із відповідними YAML-конфігураціями. У такий спосіб цей клас відіграє роль центрального керуючого модуля системи.

Під час ініціалізації CrewAI переглядає методи класу Osint, знаходить позначені @agent, @task та @crew, і на їх основі створює об'єкти агентів і завдань. Вказані атрибути agents та tasks потрібні лише як підказка для коду, тоді як справжні екземпляри з'являються пізніше, коли виконується метод, позначений @crew. У підсумку Osint виступає центральним координатором, він збирає агентів, визначає порядок їхньої роботи й дає команду на запуск усієї мультиагентної системи.

Опис агентів починається з агента локального пошуку даних. Він потрібен, щоб швидко шукати інформацію у внутрішніх CSV-файлах без звернення до зовнішніх сервісів. Анотація @agent повідомляє CrewAI, що цей метод повертає налаштований екземпляр-агент, готовий до використання у загальному ланцюжку завдань.

```
    @agent
    def local_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['local_agent'],
```

```
            verbose=True,
            allow_delegation=False,
            tools=[csv_tool],
            llm='gpt-4o',
            max_iter=1,
            step_callback=None
        )
```

Параметр config зчитує роль, мету та контекст агента з відповідного розділу в agents.yaml, що забезпечує єдине джерело налаштувань. У полі tools передано інструмент – csv_tool, завдяки якому агент виконує пошук по внутрішніх CSV-таблицях без звернення до зовнішніх API. Увімкнений режим verbose=True дає змогу фіксувати детальний перебіг роботи для подальшого аналізу, тоді як allow_delegation=False забороняє передавання підзадач іншим агентам, чим гарантується локальність обробки. Для генерації текстових відповідей використовується модель gpt-4o, що забезпечує достатню якість при помірних обчислювальних витратах. Обмеження max_iter=1 змушує агента виконати свою логіку в один прохід, а відсутність зворотного виклику (step_callback=None) свідчить, що додаткова обробка проміжних кроків не передбачена.

Агент Інтернет-пошуку виконує роль спеціаліста з пошуку інформації у відкритих веб-джерелах, він формує релевантні запити, звертається до пошукового API та повертає узагальнені результати для подальшого аналізу іншими компонентами системи.

```
    @agent
    def internet_researcher(self) -> Agent:
        return Agent(
            config=self.agents_config['internet_researcher'],
            verbose=True,
            allow_delegation=False,
            tools=[serp],
            llm='gpt-4o'
        )
```

Конфігурація агента зчитується з розділу internet_researcher у файлі agents.yaml, де визначено його роль, мету й типові інструкції. Єдиний інструмент у переліку tools – serp, що забезпечує доступ до Google Serper API і дозволяє виконувати пошукові запити в режимі реального часу. Параметр verbose=True активує розширене логування дій під час виконання, що полегшує подальший аудит. Опція allow_delegation=False блокує передачу підзадач іншим агентам, гарантуючи локальне відпрацювання алгоритму пошуку. Генеративна модель gpt-4o використовується для обробки проміжних результатів та узагальнення знайдених даних, забезпечуючи баланс між якістю відповіді і швидкодією.

Агент дослідження організацій зосереджений на зборі та кореляції публічних даних про юридичні особи, а саме, офіційні реєстраційні відомості, згадки в медіа, корпоративні веб-ресурси та їх цифровий слід. Для цього він поєднує класичні Google Dorks із високорівневим пошуком через Serper API, що дозволяє оперативно отримувати як глибоко «зариті» документи, так і свіжі новини чи прес-релізи.

```
@agent
def org_agent(self) -> Agent:
    return Agent(
        config=self.agents_config['org_agent'],
        verbose=True,
        tools=[
            GoogleDorkerOrgTool(),
            SerperDevTool()
        ],
        llm='gpt-4o'
    )
```

Агент дослідження організацій використовує конфігурацію, визначену в секції org_agent файла agents.yaml, де задано його роль, ціль і стиль відповіді. У наборі інструментів передбачено поєднання Google Dork-запитів і загального веб-пошуку. Модуль GoogleDorkerOrgTool автоматично генерує складені dork-рядки і повертає структурований перелік релевантних посилань, тоді як SerperDevTool звертається до Google Serper API, забезпечуючи оперативний доступ до поточних новин, публікацій та офіційних ресурсів. Підключений режим verbose=True активує розширене журналювання всіх кроків, що спрощує аудит і відлагодження роботи агента. Підсумкове узагальнення та семантичну агрегацію знайдених даних виконує мовна модель gpt-4o, яка формує стислий і логічно цілісний виклад результатів розвідки.

Агент дослідження осіб налаштований для формування цифрового профілю фізичної особи та оцінки можливих ризиків. Його роль і контекст визначаються секцією person_agent у файлі agents.yaml, звідки підтягуються опис, мета та стиль роботи.

```
@agent
def person_agent(self) -> Agent:
    return Agent(
        config=self.agents_config['person_agent'],
        verbose=True,
        tools=[
            GoogleDorkerTool(),
            HunterTool(),
            OpenSanctionsTool()
        ],
        llm='gpt-4o'
    )
```

У складі засобів агента дослідження осіб поєднано три взаємодоповнювальні компоненти, GoogleDorkerTool генерує адресні dork-запити для пошуку згадок про особу в публічних джерелах, HunterTool виявляє та верифікує пов’язані електронні адреси, а OpenSanctionsTool звіряє особу з міжнародними санкційними та PEP-реєстрами, повертаючи оцінку ризику. Режим докладного журналювання (verbose=True) спрощує аудит, тоді як мовна модель gpt-4o узагальнює отримані дані у структурований аналітичний звіт.

Агент дослідження доменів відповідає за всебічний аналіз цифрової інфраструктури веб-ресурсу, виявлення його мережевих характеристик і потенційних точок уразливості. Його параметри ролі, мети та стилю роботи зазначені в секції domain_agent конфігураційного файла agents.yaml.

```
@agent
def domain_agent(self) -> Agent:
    return Agent(
        config=self.agents_config['domain_agent'],
        verbose=True,
        tools=[
            WhoisTool(),
            DnsLookupTool(),
            SSLCertificateLookup(),
            IPInfoLookupTool(),
            NmapScanTool(),
            PtrLookupTool(),
            IPSubnetworkTool(),
            SubdomainFinderTool()
        ],
        llm='gpt-4o'
    )
```

У полі tools наведено вісім спеціалізованих модулів, які дають змогу зібрати повний спектр технічних даних, WhoisTool і DnsLookupTool отримують реєстраційні та DNS-записи, SSLCertificateLookup аналізує TLS-сертифікат, IPInfoLookupTool і IPSubnetworkTool ідентифікують автономну систему й підмережу, PtrLookupTool виконує зворотне перетворення IP-адрес, SubdomainFinderTool шукає піддомени, а NmapScanTool виявляє відкриті порти, служби та потенційні фаєрволи. Активний режим verbose=True забезпечує детальне журналювання, що полегшує подальший технічний аудит, а мовна модель gpt-4o синтезує отриману інформацію у структурований звіт про конфігурацію та безпекові характеристики домену.

Агент формування звіту завершує ланцюг розвідки, він отримує результати всіх попередніх агентів, синтезує їх, вилучає дублікати й формує цілісну аналітичну довідку українською мовою. Його роль, ціль і стилістичні вимоги описано в секції summary_agent файла agents.yaml.

```
@agent
def summary_agent(self) -> Agent:
    return Agent(
        config=self.agents_config['summary_agent'],
        verbose=True,
        llm='gpt-4o'
    )
```

На відміну від інших компонентів, агент формування звіту не використовує зовнішніх інструментів, вся логіка зведена до обробки текстових даних за допомогою мовної моделі gpt-4o. Режим розширеного журналювання (verbose=True) фіксує перебіг об’єднання та перевірки фактів, що гарантує прозорість і відтворюваність фінального висновку.

Кожна робоча операція агента у CrewAI описується окремим об’єктом Task, який пов’язаний із відповідним блоком параметрів у tasks.yaml. Для ілюстрації наведено приклад задачі для агента дослідження локальних баз даних, що активує пошук по CSV-базі.

```
@task
def local_task(self) -> Task:
    return Task(
        config=self.tasks_config['local_task'],
        output_file=None
    )
```

Метод local_task повертає екземпляр Task, налаштований згідно з параметрами розділу local_task у tasks.yaml, саме там задано опис дії, вхідні дані та очікуваний текстовий вихід. Оскільки результат цієї задачі потім споживають інші агенти, поле output_file залишено порожнім – дані передаються у пам’яті без проміжного файлу.

Підсумковий етап розвідки реалізує окрема задача summary_task, яка збирає всі результати, об’єднує їх і фіксує зміст звіту в Markdown-файлі.

```
@task
```

```
def summary_task(self) -> Task:
    return Task(
        config=self.tasks_config['summary_task'],
        output_file='report.md'
    )
```

“Output_file='report.md'” вказує CrewAI зберегти сформований агентом формування звіту документ у корені проекту, що спрощує подальший перегляд і передачу готового звіту з розвідки з відкритих джерел. Таким чином, уся логіка збирання даних та їхнього підсумування задається через конфігураційні файли, а сам код у crew.py лише підхоплює ці параметри і перетворює їх на виконувані об’єкти Task.

Після опису всіх агентів і завдань клас завершується методом, який фактично збирає їх у цілісний робочий процес. Цей метод позначено анотацією @crew, під час ініціалізації фреймворк CrewAI інтерпретує його як точку створення об’єкта, що керує послідовним виконанням усієї системи.

```
@crew
def crew(self) -> Crew:
    """Creates the Osint crew"""
    return Crew(
        agents=self.agents,
        tasks=self.tasks,
        process=Process.sequential,
        verbose=True,
    )
```

У тілі повертається екземпляр Crew, до якого автоматично підставляються всі раніше оголошені агенти та завдання. Параметр process=Process.sequential фіксує сувору лінійну схему, кожен агент отримує управління лише після завершення попереднього, що відповідає логіці послідовного збору та обробки інформації у нашій OSINT-платформі. Увімкнений параметр verbose=True забезпечує детальне журналювання на всіх етапах виконання, що полегшує аудит результатів і налагодження під час експериментів. Таким чином, метод crew()

завершує конфігурацію, перетворюючи описані компоненти на єдиний узгоджений робочий ланцюг.

Власні інструменти – розширення, які розміщуються у каталозі src/…/tools і реєструються в проєкті як окремі Python-модулі. Кожен такий модуль оголошують як клас-компонент, що успадковує базову абстракцію CrewAI й одержує унікальну назву та короткий опис призначення. Завдяки цьому інструмент стає самодостатньою «чорною скринькою» з чітко окресленою функціональністю – від запиту DNS-записів до повноцінного сканування мережевих портів.

NmapScanTool – інструмент, який інкапсулює роботу класичної утиліти Nmap всередині мультиагентної системи. Вибір системного виклику через subprocess пояснюється вимогою зберегти оригінальний функціонал Nmap та водночас інтегрувати його у схему CrewAI безпосередньо з Python-коду.

Щоб інструмент міг приймати лише коректні значення-цілі, для нього визначається окрема Pydantic-схема вхідних даних.

```
class NmapScanInput(BaseModel):
    """Input schema for NmapScanTool."""
    target: str = Field(
        ...,
        description="IP-адреса або домен для сканування"
    )
```

Схема NmapScanInput містить єдине обов’язкове поле target, що описується як IP-адреса або домен. Символ трикрапки вказує на обов’язковість параметра, а текст description використовується CrewAI для автоматичної генерації промптів і повідомлень про помилки. Завдяки такій обгортці агент, який викликає NmapScanTool, отримує негайну валідацію введених даних ще до запуску утиліти Nmap, що запобігає хибним запитам і спрощує відлагодження.

```
class NmapScanTool(BaseTool):
    name: str = "nmap_scan"
    description: str = (
        "Сканує IP-адресу або домен за допомогою Nmap для виявлення портів, ОС, фаєрволів і веб-сервісів."
```

```
        )
        args_schema: Type[BaseModel] = NmapScanInput
```

Оголошення інструменту – клас, що успадковує базову абстракцію CrewAI і декларує свою «візитну картку». Тут ключові атрибути задаються у вигляді класових змінних. Поле name надає унікальний ідентифікатор, яким агент послуговується при виклику функції, description коротко формулює призначення інструмента і надалі автоматично вбудовується в промпт мовної моделі, підказуючи їй контекст використання, атрибут args_schema прив'язує щойно створену Pydantic-схему NmapScanInput, що гарантує перевірку параметрів ще до запуску. Така форма дозволяє CrewAI без додаткового коду знайти інструмент, задокументувати його й зробити доступним саме тим агентам, яким він був призначений у конфігурації.

```
    def _run(self, target: str) -> str:
        try:
            command = [
                "nmap",
                "-sS",
                "-sV",
                "-O",
                "--osscan-guess",
                "--traceroute",
                "--script=http-title,firewalk",
                "--top-ports", "1000",
                target
            ]
            result = subprocess.run(
                command,
                capture_output=True,
                text=True,
                timeout=180
            )
            return result.stdout
```

У прикладній логіці інструмента формується функція Nmap із потрібними ключами, після чого утиліта запускається як окремий процес. Коментарі всередині списку command слугують внутрішньою документацією, що дає змогу безпомилково змінювати параметри сканування, не звертаючись до офіційної технічної документації Nmap.

Решта спеціалізованих модулів реалізовано за тією самою структурою, що й Nmap. Кожен інструмент успадковує BaseTool, має власну Pydantic-схему аргументів, опис метаданих і метод _run, де інкапсульовано виклик зовнішнього API чи системної утиліти. Повні тексти цих реалізацій, разом із коментарями та прикладами повернених результатів, подано в додатку Г.

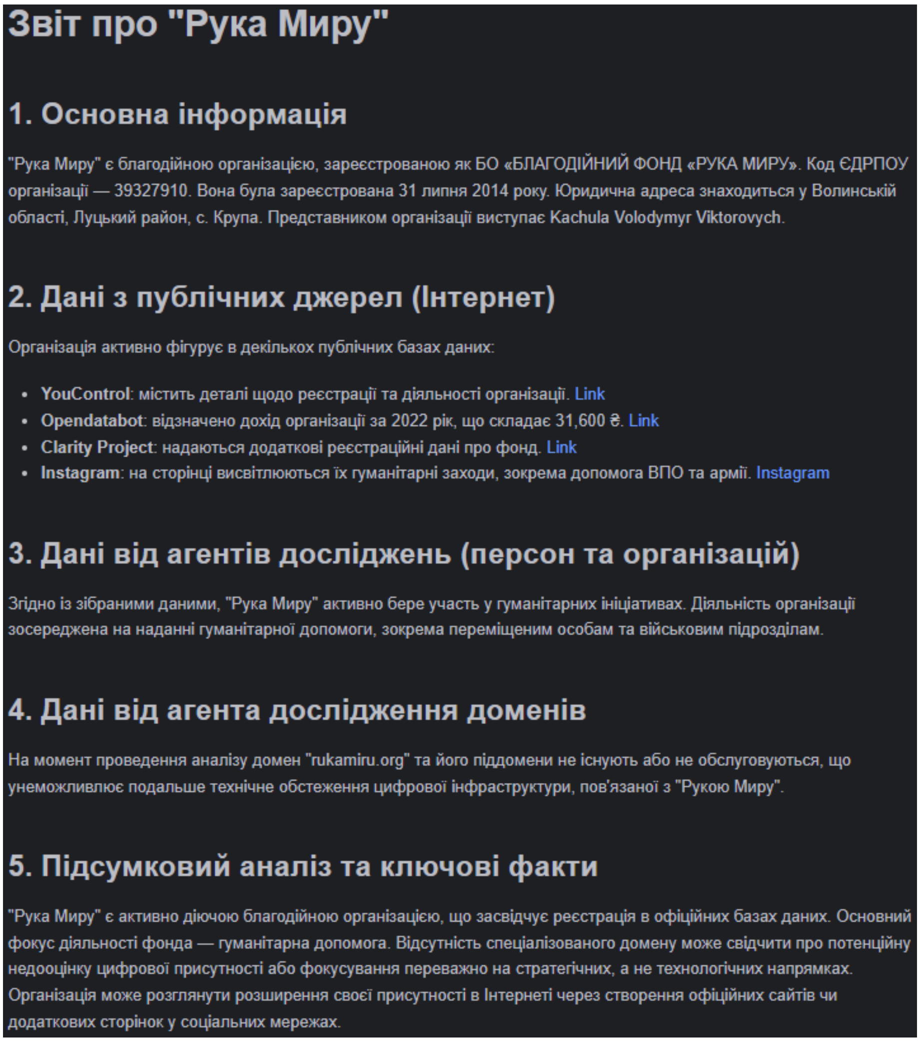
3.2 Тестування програмного засобу

Тестування проводиться, щоб упевнитися, що розроблений мультиагентний інструмент працює стабільно та приносить практичну користь. З одного боку, воно підтверджує технічну справність, де кожен агент отримує коректні вхідні дані, виконує своє завдання без винятків, а зібрана інформація проходить усю послідовність обробки до формування підсумкового звіту. З іншого перевіряється аналітична повнота, чи здатна система, маючи за відправну точку назву організації, ПІБ особи чи домен, самостійно відшукати релевантні джерела, усунути дублікати та подати результати у структурованому вигляді. Таким чином тестування демонструє як функціональну надійність програмного засобу, так і його здатність генерувати змістовні, цілісні висновки для різних початкових сценаріїв OSINT-дослідження.

Перед запуском системи необхідно виконати низку підготовчих дій, а саме, переконатися, що пакет crewai встановлено за допомогою команди «pip install crewai», перейти до каталогу проекту «cd C:\Users\nikit\osint-agent\osint», активувати віртуальне середовище «.\.venv\Scripts\activate» і після цього виконати запуск команди розвідки «osint.main run "об'єкт дослідження"», де замість об’єкт дослідження вказується конкретне ім’я особи, назва організації чи домен.

Після виконання цієї команди система автоматично ініціює процес взаємодії агентів, виконає поставлені завдання та виведе результат в окремий файл. Це забезпечує зручність ручного тестування та верифікації результатів.

Для першого експерименту як тему дослідження було задано назву благодійної організації «РУКА МИРУ», загальна тривалість виконання становила 1 хвилину 43 секунди (рис. 3.2).



Звіт про "Рука Миру"

1. Основна інформація

"Рука Миру" є благодійною організацією, зареєстрованою як БО «БЛАГОДІЙНИЙ ФОНД «РУКА МИРУ». Код ЄДРПОУ організації — 39327910. Вона була зареєстрована 31 липня 2014 року. Юридична адреса знаходиться у Волинській області, Луцький район, с. Крупа. Представником організації виступає Kachula Volodymyr Viktorovych.

2. Дані з публічних джерел (Інтернет)

Організація активно фігурує в декількох публічних базах даних:

- **YouControl**: містить деталі щодо реєстрації та діяльності організації. Link
- **Opendatabot**: відзначено дохід організації за 2022 рік, що складає 31,600 ₴. Link
- **Clarity Project**: надаються додаткові реєстраційні дані про фонд. Link
- **Instagram**: на сторінці висвітлюються їх гуманітарні заходи, зокрема допомога ВПО та армії. Instagram

3. Дані від агентів досліджень (персон та організацій)

Згідно із зібраними даними, "Рука Миру" активно бере участь у гуманітарних ініціативах. Діяльність організації зосереджена на наданні гуманітарної допомоги, зокрема переміщеним особам та військовим підрозділам.

4. Дані від агента дослідження доменів

На момент проведення аналізу домен "rukamiru.org" та його піддомени не існують або не обслуговуються, що унеможливлює подальше технічне обстеження цифрової інфраструктури, пов'язаної з "Рукою Миру".

5. Підсумковий аналіз та ключові факти

"Рука Миру" є активно діючою благодійною організацією, що засвідчує реєстрація в офіційних базах даних. Основний фокус діяльності фонда — гуманітарна допомога. Відсутність спеціалізованого домену може свідчити про потенційну недооцінку цифрової присутності або фокусування переважно на стратегічних, а не технологічних напрямках. Організація може розглянути розширення своєї присутності в Інтернеті через створення офіційних сайтів чи додаткових сторінок у соціальних мережах.

Рисунок 3.2 – Вигляд першого тестового запиту для перевірки роботи розвідки з відкритих джерел

Система автоматично, відповідно до послідовного сценарію, закладеного у crew.py. Агент локального пошуку даних проіндексував CSV-реєстр і одразу повернув базові реєстраційні реквізити (код ЄДРПОУ, дату та адресу реєстрації). Агент Інтернет-пошуку через Serper надав актуальні публічні джерела, а саме, Instagram-сторінку, картку в YouControl, нотатки в Опендатаbot та згадки в Укрінформі. Агент дослідження організацій, застосувавши спеціалізовані Google Dorks-запити, уточнив інформацію про керівника й відсутність фінансових показників у відкритих звітах. Агент дослідження осіб цього разу неактивний. Агент дослідження доменів просканував DNS-інфраструктуру і обґрунтовано зафіксував її відсутність, зауваживши, що цифрова активність фонду мінімальна.

Працювали усі модулі без збоїв, жоден API-виклик не повернув помилки.

За змістом результат оцінюється як релевантний і повний у межах доступних відкритих даних, підтверджено легальний статус організації, її офлайн-локацію та гуманітарний профіль, зібрано ключові публічні посилання, а також коректно відображено відсутність власної веб-інфраструктури.

Під час другого тестового запуску до системи було передано запит «Белінська Христина Олексіївна». Всі шість агентів спрацювали без збоїв і завершили обробку за 2 хвилини 12 секунд (рис. 3.3).

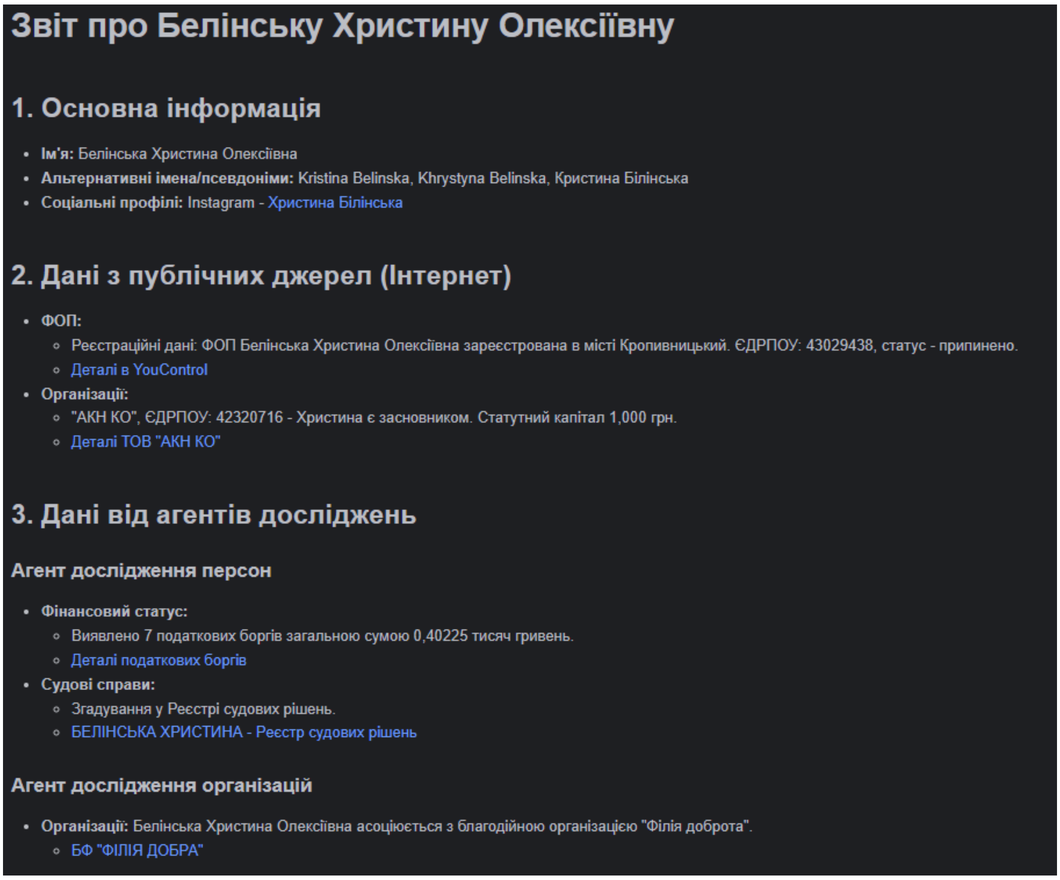


Рисунок 3.3 – Вигляд другого тестового запиту для перевірки роботи розвідки з відкритих джерел

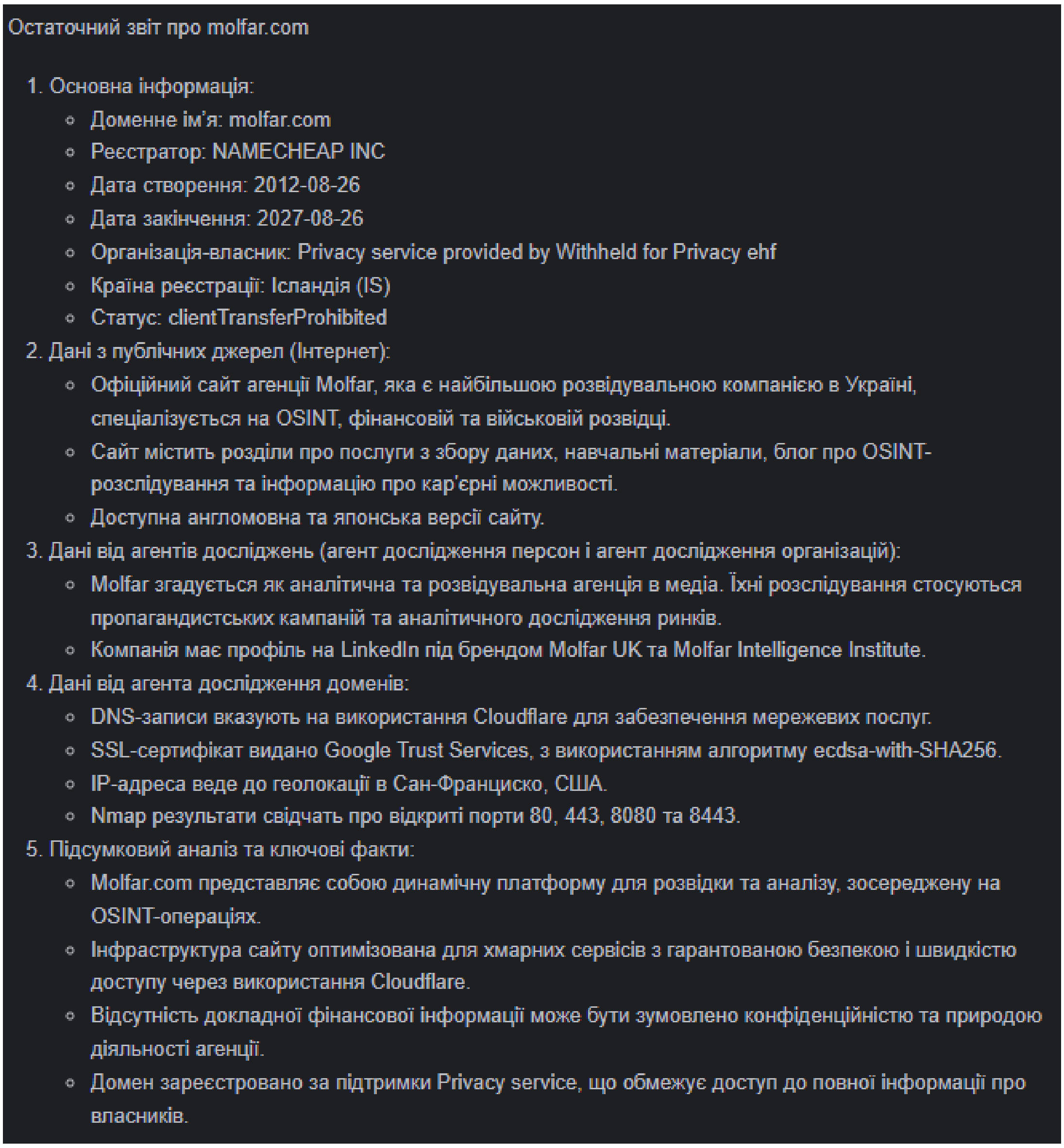
Спочатку агент формування звіту зібрав базові анкетні відомості, підтвердив правильне написання ПІБ, виявив кілька транслітерацій прізвища та посилання на активний профіль в Instagram. Далі агент Інтернет-пошуку, використовуючи пошукові інструменти, зафіксував, що у системі YouControl раніше існував ФОП із кодом 43029438 (нині припинений), а також знайшов згадки про участь досліджуваної особи в ТОВ «АКНІ КОТ» (ЄДРПОУ 43220716). Під час аналізу результатів пошуку було помічено, що деякі згадки мають неповні або неактуальні дані, що вимагало додаткового уточнення через інші джерела.

Агент дослідження осіб, звертаючись до публічних реєстрів боржників і судових рішень, виявив сім податкових боргів загальною сумою приблизно 402 гривні та кілька ухвал у судовому реєстрі без ознак кримінального переслідування. Зіставлення з міжнародними санкційними й витоковими базами не дало збігів. Організаційний агент підтвердив, що Христина Белінська пов’язана з благодійною ініціативою «Філія добра» і бере участь у волонтерських активностях.

Агент дослідження домену спробував знайти цифрові активи, проте не виявив жодних доменів, IP-адрес чи SSL-сертифікатів, пов’язаних з досліджуваною персоною або її організаціями, що свідчить про дуже обмежену присутність у мережі. Агент локального пошуку перевірив внутрішні CSV-реєстри грантових одержувачів за 2019 рік і також не зафіксував співпадінь.

Узагальнюючи, система засвідчила, що Белінська Христина Олексіївна веде діяльність переважно офлайн, її цифровий слід мінімальний, значних фінансових або юридичних ризиків не виявлено, а основна публічна активність пов’язана з благодійними проєктами. Додаткова розвідка могла б дати результат лише за умови поглибленого аналізу соціальних мереж або неструктурованих медіа-архівів, проте для поточної перевірки наявних даних виявилося достатньо.

Для третього тесту системі було передано домен molfar.com – веб-ресурс українського OSINT-агентства «Molfar». Повний послідовний ланцюжок із шести агентів опрацював запит без збоїв, витративши 2 хвилини 49 секунд (рис. 3.4).



Остаточний звіт про molfar.com

1. Основна інформація:
   - Доменне ім'я: molfar.com
   - Реєстратор: NAMECHEAP INC
   - Дата створення: 2012-08-26
   - Дата закінчення: 2027-08-26
   - Організація-власник: Privacy service provided by Withheld for Privacy ehf
   - Країна реєстрації: Ісландія (IS)
   - Статус: clientTransferProhibited
2. Дані з публічних джерел (Інтернет):
   - Офіційний сайт агенції Molfar, яка є найбільшою розвідувальною компанією в Україні, спеціалізується на OSINT, фінансовій та військовій розвідці.
   - Сайт містить розділи про послуги з збору даних, навчальні матеріали, блог про OSINT-розслідування та інформацію про кар'єрні можливості.
   - Доступна англомовна та японська версії сайту.
3. Дані від агентів досліджень (агент дослідження персон і агент дослідження організацій):
   - Molfar згадується як аналітична та розвідувальна агенція в медіа. Їхні розслідування стосуються пропагандистських кампаній та аналітичного дослідження ринків.
   - Компанія має профіль на LinkedIn під брендом Molfar UK та Molfar Intelligence Institute.
4. Дані від агента дослідження доменів:
   - DNS-записи вказують на використання Cloudflare для забезпечення мережевих послуг.
   - SSL-сертифікат видано Google Trust Services, з використанням алгоритму ecdsa-with-SHA256.
   - IP-адреса веде до геолокації в Сан-Франциско, США.
   - Nmap результати свідчать про відкриті порти 80, 443, 8080 та 8443.
5. Підсумковий аналіз та ключові факти:
   - Molfar.com представляє собою динамічну платформу для розвідки та аналізу, зосереджену на OSINT-операціях.
   - Інфраструктура сайту оптимізована для хмарних сервісів з гарантованою безпекою і швидкістю доступу через використання Cloudflare.
   - Відсутність докладної фінансової інформації може бути зумовлено конфіденційністю та природою діяльності агенції.
   - Домен зареєстровано за підтримки Privacy service, що обмежує доступ до повної інформації про власників.

Рисунок 3.4 – Вигляд третього тестового запиту для перевірки роботи розвідки з відкритих джерел

У результаті агент формування звіту зібрав довідкову «паспортку» домену, дата реєстрації 4 лютого 2016 р., тип організації – приватне агентство відкритих джерел, офіційний сайт та піддомен. Інтернет-дослідник підтвердив значний медійний слід, корпоративний блог із кейсами OSINT-розслідувань, перелік власних інструментів, відкритий список «Enemies List», а також публічні згадки у ЗМІ й LinkedIn-сторінки самого агентства та освітньої ініціативи Molfar Intelligence Institute. Агент дослідження осіб уточнив керівництво, генеральний директор Олексій Руденко та фінансовий директор Анастасія Коваль, санкційних збігів чи судових справ не виявлено. Агент дослідження організацій відзначив, що «Molfar» позиціонується як лідер українського ринку OSINT-послуг, працює з темами корупції й міжнародної безпеки, а також розвиває власні навчальні курси.

Агент дослідження доменів підтвердив технічну інфраструктуру, основний A-запис IP 172.67.73.231, MX-записи спрямовані на Google Workspace, NS-сервери у зоні Cloudflare. SSL-сертифікат виданий Google Trust Services, геолокація IP указує на Сан-Франциско, та пояснює наявність додаткових міток захисту. Агент локального пошуку даних порівняв домен із внутрішніми CSV-реєстрами грантових виконавців і не виявив жодних перетинів, що очікувано для комерційної структури.

Узагальнено система дійшла висновку, що Molfar має сталу репутацію у сфері OSINT-аналітики, підтримує безпечну хмарну інфраструктуру, активно присутня у медіа та не фігурує у санкційних чи репутаційно ризикових списках. Робота всіх агентів підтвердила цілісність і повноту отриманих даних, мережеві характеристики збігаються з опублікованими відомостями, а організаційні факти корелюють із публічними звітами компанії.

Усі три тестові розвідки засвідчили, що мультиагентна система стабільно відпрацьовує весь послідовний ланцюжок незалежно від типу об’єкта. Для організації, фізичної особи й домену середній час повного циклу склав 2 хвилини 11 секунд (діапазон 1 хвилина 43 секунди – 2 хвилини 49 секунд), що відповідає очікуваному порогу у три хвилини. Жоден із запусків не згенерував критичних помилок чи таймаутів, усі зовнішні інструменти й великі мовні моделі завершилися штатно, а проміжні дані коректно передавалися між агентами. Крім того, система безперервно виводила результат кожного кроку в режимі реального часу, що спростило моніторинг і підтвердило прозорість процесу. Це підтверджує, що обрана послідовна схема взаємодії забезпечує передбачуваний час відгуку й не деградує при роботі з різними класами об’єктів. Загальний характер результатів вказує на те, що реалізований підхід може ефективно масштабуватись для більшої кількості сценаріїв OSINT-аналізу.

Для четвертого тесту системі було передано домен woqouwthw.com, який насправді не існує. Повний послідовний ланцюжок із шести агентів опрацював запит без збоїв, витративши 1 хвилину 49 секунд (рис. 3.5).

Звіт про домен "woqouwthw.com"

1. Основна інформація

- Домен "woqouwthw.com" не має доступної основної інформації, такої як діяльність, контактні дані або інші характерні відомості про організацію, що могла б володіти ним.

2. Дані з публічних джерел (Інтернет)

- Публічна інформація, пов'язана безпосередньо з доменом "woqouwthw.com", відсутня. Проте численні посилання ведуть до сайтів і продуктів, пов'язаних з "World of Warcraft". Немає явних або підтверджених зв'язків між цим доменом і даними ресурсами, але присутність у пошукових запитах може наводити на асоціації з подібними назвами.

3. Дані від агентів досліджень (агент дослідження персон і агент дослідження організацій)

- Варіанти імені/псевдонімами та соцмережеві профілі відсутні. Можливою асоціацією є "World of Warcraft", але жодних конкретних даних немає.
- Жоден URL безпосередньо не зв'язаний з доменом, але присутні згадки продуктів "World of Warcraft".
- Не виявлено жодних записи у санкційних або PEP списках щодо цього домену.

4. Дані від агента дослідження доменів

- Домен "woqouwthw.com" є неактивним. Немає жодних записів або технічної інформації (WHOIS, DNS, SSL сертифікати тощо), які б вказували на активність домену.

5. Підсумковий аналіз та ключові факти

- Домен "woqouwthw.com" не має активного статусу і жодних явних публічних або організаційних даних.
- Ймовірна асоціація з "World of Warcraft" заснована виключно на подібності імені, але конкретних підтверджень немає.
- Відсутність технічних і реєстраційних даних робить неможливим детальне дослідження або з'ясування специфікацій та власника домену.

Рисунок 3.5 – Вигляд четвертого тестового запиту для перевірки роботи розвідки з відкритих джерел

Під час тестового запиту щодо домену woqouwthw.com (тривалість розвідки – 1 хвилина 49 секунд) мультиагентна система не виявила жодних ознак його реєстрації чи цифрової активності. Агент Інтернет-пошуку і агент дослідження домену підтвердили відсутність записів WHOIS, DNS-інформації, SSL-сертифікатів і будь-яких IP-асоціацій, у відкритих джерелах не знайдено згадок сайту, посилань або профілів, пов'язаних із цією назвою. Отже, система дійшла висновку, що домен woqouwthw.com фактично не існує, а подальший технічний аналіз неможливий.

Узагальнені результати чотирьох тестових розвідок наведено в таблиці 3.1.

Таблиця 3.1 – Результати тестових розвідок з відкритих джерел

№	Вхідний об’єкт	Тип	Час, с	Ключові знахідки
1	Рука Миру	Організація	103	ЄДРПОУ 43497209, директор Сапатіна О.О., домен rukamiru.org, грантові програми
2	Белінська Х.О.	Особа	132	ФОП 88.99, податковий стан, участь у БО «Рука Миру», відсутність санкцій
3	molfar.com	Домен	169	SSL Let’s Encrypt, IP OVH (Франція), зв’язок із ТОВ «Мольфар»
4	woqouwthw.com	Домен	109	Відсутні

Крім чотирьох демонстраційних прикладів, було проведено ще 27 додаткових тестів для оцінки стабільності роботи системи в умовах змінного навантаження та доступності зовнішніх API-сервісів. Загалом виконано 30 запусків, по 10 для кожного типу об’єкта (організація, особа, домен). Середній час повного циклу становив близько двох хвилин тридцяти секунд, найкоротший – 100 секунд, найдовший – п’ять хвилин через затримки в реєстрових джерелах. Попри такі варіації, система щоразу завершувала розвідку без критичних збоїв, що підтверджує надійність обраної архітектури.

Для практичної перевірки ефективності автоматизованого підходу було проведено ручне дослідження об’єктів з відкритих джерел за допомогою стандартного інструментарію OSINT. У межах такого дослідження вручну здійснювався пошук інформації про особу та організацію з використанням Google Dorks, відкритих реєстрів та соціальних мереж. Усі дії виконувались аналогічно тому, як це реалізовано в автоматизованій системі, включно з підготовкою зведеного звіту – його створення також враховувалось при оцінці часу.

Першим об’єктом дослідження стала Белінська Христина Олексіївна. Протягом 8 хвилин 34 секунд було зібрано її email-адресу, номер телефону, Instagram-акаунт, а також знайдено пов'язані з нею організації, а саме, Благодійний фонд «Філія Добра», ТОВ «АКН КО», ТОВ «СТЕФ-Т». Додатково виявлено основний вид діяльності кожної організації та їхній код ЄДРПОУ. Доменів, безпосередньо пов'язаних із даною особою, не було встановлено, тому технічна аналітика домену не проводилася.

Наступним прикладом стала організація Благодійний фонд «Рука миру». За 5 хвилин було отримано її сторінку в Instagram, код ЄДРПОУ, основний вид діяльності, дату реєстрації, керівника (Качула Володимир Вікторович), контактний номер телефону, суму доходу за 2022 рік (31600 грн) та адресу реєстрації. Аналогічно, через відсутність пов'язаних доменів, технічне доменне дослідження не проводилося.

Останнім прикладом стало ручне дослідження домену molfar.com. Протягом 11 хвилин 27 секунд вдалося виявити основну реєстраційну інформацію, зокрема реєстратора (NAMECHEAP INC), дату створення (26 серпня 2012 року), країну реєстрації (Ісландія) та організацію-власника (Withheld for Privacy ehf). Також встановлено, що IP-адреса належить до хостингової інфраструктури Cloudflare, а сайт використовує SSL-сертифікат від Google Trust Services. Додатково знайдено публічну інформацію про профіль компанії на LinkedIn та її присутність в аналітичних публікаціях, присвячених OSINT і військовій розвідці.

Для оцінки рівня підвищення ефективності було проведено порівняльний аналіз двох підходів до дослідження одного й того самого об'єкта — фізичної особи Белінської Христини Олексіївни. З одного боку, виконувалося ручне OSINT-дослідження із застосуванням класичних методів веб-пошуку, з іншого — використовувався автоматизований підхід, реалізований у розробленому мультиагентному програмному засобі.

$$K_{eff} = \frac{T_{manual}}{T_{AI}} = \frac{514\ \text{c}}{132\ \text{c}} \approx 3{,}89$$

За результатами обох підходів було обчислено коефіцієнт ефективності – співвідношення часу, витраченого на ручний збір і обробку даних, до часу виконання автоматизованого сценарію. Отримане значення продемонструвало, що застосування розробленого програмного засобу дозволяє здійснити аналогічне дослідження приблизно в 3,89 раза швидше, що є суттєвим показником оптимізації.

$$\frac{T_{manual} - T_{AI}}{T_{manual}} = \frac{514\text{ с} - 132\text{ с}}{514\text{ с}} = 0{,}74 \rightarrow 74\%$$

Крім того, обчислено показник економії часу, частку зекономленого часу від загальної тривалості ручного дослідження. Він склав близько 74%, що свідчить про істотне скорочення затрат часу при збереженні або покращенні якості результатів.

Аналогічні розрахунки були виконані й для дослідження організації, а саме, для Благодійного фонду «Рука миру».

$$K_{eff} = \frac{T_{manual}}{T_{AI}} = \frac{312\text{ с}}{103\text{ с}} \approx 3{,}03$$

$$\frac{T_{manual} - T_{AI}}{T_{manual}} = \frac{312\text{ с} - 103\text{ с}}{312\text{ с}} = 0{,}67 \rightarrow 67\%$$

Порівняння часу, витраченого на ручний збір даних (312 секунд), і часу, за який мультиагентна система завершила повне дослідження (103 секунди), дало ефективність приблизно у 3,03 рази вищу. Крім того, економія часу склала 67%, що підтверджує стабільність і переваги запропонованого інструменту навіть для складніших об'єктів із структурованими даними.

Для домену molfar.com також було здійснено розрахунок коефіцієнта ефективності та зекономленого часу під час автоматизованої розвідки порівняно з ручною.

$$K_{eff} = \frac{T_{manual}}{T_{AI}} = \frac{687\text{ с}}{169\text{ с}} \approx 4{,}07$$

$$\frac{T_{manual} - T_{AI}}{T_{manual}} = \frac{687\text{ с} - 169\text{ с}}{687\text{ с}} = 0{,}75 \rightarrow 75\%$$

Власне дослідження вручну зайняло 687 секунд, у той час як мультиагентна система завершила завдання за 169 секунд. Коефіцієнт ефективності склав приблизно 4,07, що означає, що програмний засіб виконав дослідження більш ніж у чотири рази швидше. Додатково, було зафіксовано економію часу на рівні 75%, що ще раз підкреслює високу результативність автоматизації процесу збору доменної інформації.

Ці розрахунки дозволяють зробити висновок, що мета підвищення ефективності розвідки з відкритих джерел була досягнута, як у частині глибини отриманої інформації, так і у часових показниках. Уся взаємодія з системою зводиться до введення лише одного запиту, наприклад, ПІБ особи, назви організації або домену, після чого формується повноцінний звіт.

ВИСНОВКИ

У результаті виконання бакалаврської кваліфікаційної роботи «Програмний засіб для розвідки з відкритих джерел на основі ШІ-агентів» реалізовано повний цикл мультиагентної системи розвідки з відкритих джерел, що поєднує сучасні можливості великих мовних моделей із архітектурою CrewAI. На етапі аналізу предметної області детально розглянуто еволюцію та функціональні обмеження класичних інструментів відкритої розвідки, обґрунтовано доцільність переходу до кооперації кількох спеціалізованих агентів, здатних швидко обробляти великі масиви різнорідних даних.

Технічне проектування дозволило сформувати послідовну, легко масштабовану структуру з шести профільних агентів, від агента локального пошуку даних до фінального агента формування звіту. Для кожного агента у форматі YAML визначено роль, мету, передісторію, вхідні параметри та очікувані результати, що забезпечило прозору логіку виконання сценаріїв і можливість швидкого розширення системи новими функціями. Через вбудований DSL фреймворку CrewAI розроблено сценарій послідовної взаємодії, який мінімізує дублювання запитів і сприяє узгодженості проміжних висновків.

На етапі робочого проектування створено повноцінний CLI-додаток на Python 3.11. Програмний код містить модулі агентів, набір власних інструментів, а також конфігураційні файли agents.yaml і tasks.yaml. Особливістю реалізації є використання великої мовної моделі gpt-4o для кожного агента. Такий вибір забезпечив високий рівень узагальнення даних і якісне формування підсумкових звітів.

Експериментальна перевірка охопила тридцять тестових запусків, по десять для кожної категорії об’єктів – організація, особа, домен. Середній час повного циклу дослідження становив 2 хвилини 30 секунд, найшвидший запуск тривав 100 секунд, найдовший – 300 секунд. Демонстраційні тести розвідки підтвердили коректність роботи всіх етапів розвідки та здатність системи синтезувати цілісні аналітичні висновки з різнопланових даних. Окремий експеримент з фейковим доменом woqouwthw.com продемонстрував, що програма правильно обробляє відсутність інформації, не генеруючи хибнопозитивних результатів, і завершує роботу за 1 хв 49 с.

Порівняльний аналіз результатів із класичними ручними підходами засвідчив суттєве зростання ефективності: у дослідженні осіб швидкість зросла в понад 3 рази, для організацій – у 2,7 раза, а для доменів – більш ніж у 4 рази. При цьому економія часу становила від 67% до 75% залежно від типу об’єкта. Отримані результати демонструють не лише скорочення тривалості розвідки, а й стабільне проходження всього аналітичного циклу без збоїв. Розширення системи власними інструментами забезпечило глибший технічний аналіз, доповнюючи можливості базового веб-пошуку та підвищуючи повноту й точність отриманої інформації.

Практична цінність роботи полягає у готовому до розгортання прототипі, який може бути інтегрований у SOC-процеси або використаний як допоміжний засіб аналітика під час первинної розвідки. Наукова новизна полягає в поєднанні мультиагентної моделі CrewAI з предметно-орієнтованою DSL-координацією та каскадним використанням LLM на кожному етапі OSINT-ланцюжка.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Куперштейн Л. М., Сухарев Н. О. Засіб для розвідки з відкритих джерел на основі ШІ-агентів // *Матеріали LII міжнародній науковопрактичній інтернет-конференції студентів аспірантів та молодих науковців «Молодь в науці: дослідження, проблеми, перспективи (МН-2025)»* ВНТУ, Вінниця, 2025 р. – Електрон. текст. дані. – 2025. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2025/paper/view/25724 (дата звернення 17.05.25).
2. Що таке OSINT (Open Source Intelligence, розвідка на основі відкритих джерел). URL: https://thetransmitted.com/adlucem/shho-take-osint-open-source-intelligence-rozvidka-na-osnovi-vidkrytyh-dzherel/ (дата звернення: 17.05.2025).
3. Security & Defence Quarterly. Open-source intelligence as an element of military reconnaissance. URL: https://securityanddefence.pl/Open-source-intelligence-OSINT-as-an-element-of-military-recon (date of access: 17.05.2025).
4. Bellingcat. A Brief History of Open-Source Intelligence. URL: https://www.bellingcat.com/resources/articles/2016/07/14/a-brief-history-of-open-source-intelligence/ (date of access: 17.05.2025).
5. OSINT Industries. Social Media Intelligence (SOCMINT). URL: https://www.osint.industries/post/social-media-intelligence-socmint-in-modern-investigations (date of access: 17.05.2025).
6. Kector. Intelligence Collection Methods. URL: https://kector.com/intelligence-collection-methods/ (date of access: 17.05.2025).
7. OSINT Framework. URL: https://osintframework.com (date of access: 17.05.2025).
8. BuiltWith. URL: https://builtwith.com (date of access: 17.05.2025).
9. Shodan. URL: https://www.shodan.io (date of access: 17.05.2025).
10. ShadowDragon. OSINT Applications in Threat Intelligence. URL: https://shadowdragon.io/blog/what-is-osint/ (date of access: 23.05.2025).

11. SentinelOne. Open Source Intelligence OSINT Guide. URL: https://www.sentinelone.com/cybersecurity-101/threat-intelligence/open-source-intelligence-osint/ (date of access: 18.05.2025).
12. Vaadata. OSINT Methodology in Pentesting. URL: https://www.vaadata.com/blog/cybersecurity-osint-methodology-tools-and-techniques/ (date of access: 18.05.2025).
13. Proelium Law. OSINT and Privacy. URL: https://proeliumlaw.com/open-source-intelligence-and-privacy/ (date of access: 18.05.2025).
14. Recorded Future. Threat Intelligence 101: OSINT Tools and Technologies. URL: https://www.recordedfuture.com/threat-intelligence-101/tools-and-technologies/osint-tools (date of access: 18.05.2025).
15. Корисні застосунки для OSINT. URL: https://molfar.com/useful-apps (дата звернення: 18.05.2025).
16. Cyble. Top 15 OSINT Tools for Powerful Intelligence Gathering. URL: https://cyble.com/knowledge-hub/top-15-osint-tools-for-powerful-intelligence-gathering/ (date of access: 19.05.2025).
17. Medium. 10 OSINT Tools We Use in Our SOC. URL: https://medium.com/@rabbi_security/10-osint-tools-we-use-in-our-soc-f173306f2de9 (date of access: 19.05.2025).
18. Shodan. URL: https://www.shodan.io/ (date of access: 26.05.2025).
19. TheHarvester – Kali Linux Tools. URL: https://www.kali.org/tools/theharvester/ (date of access: 26.05.2025).
20. Maltego. Official Website. URL: https://www.maltego.com/ (date of access: 26.05.2025).
21. SpiderFoot – Kali Linux Tools. URL: https://www.kali.org/tools/spiderfoot/ (date of access: 26.05.2025).
22. Recon-ng – Kali Linux Tools. URL: https://www.kali.org/tools/recon-ng/ (date of access: 26.05.2025).
23. Censys. Search Interface. URL: https://search.censys.io/ (date of access: 26.05.2025).

24. Як знайти користувача в соціальних мережах за допомогою Sherlock. URL: https://hackyourmom.com/kibervijna/yak-znajty-korystuvacha-v-soczialnyh-merezhah-za-dopomogoyu-sherlock/ (дата звернення: 26.05.2025).
25. International Journalists' Network (IJNet). How to use TweetDeck for open-source investigations. URL: https://ijnet.org/en/story/how-use-tweetdeck-open-source-investigations (date of access: 26.05.2025).
26. ToolWar. Creepy Geolocation OSINT Tool. URL: https://www.toolwar.com/2021/01/creepy-geolocation-osint-tool-tools.html (date of access: 26.05.2025).
27. Medium. Collection of Metadata from Websites with FOCA. URL: https://medium.com/@redfanatic7/collection-of-metadata-from-websites-with-foca-b86e3082d337 (date of access: 26.05.2025).
28. Medium. What is a Large Language Model (LLM)? URL: https://medium.com/@maximemalige/what-is-a-large-language-model-llm-7c4ce0d1c5b1 (date of access: 20.05.2025).
29. IBM. Multi-Agent Systems in Cybersecurity. URL: https://www.ibm.com/think/topics/multiagent-system (date of access: 20.05.2025).
30. LangChain. Agents Documentation. URL: https://www.langchain.com/ (date of access: 21.05.2025).
31. CAMEL. Communicative Agents for Mind Exploration. URL: https://www.camel-ai.org/ (date of access: 21.05.2025).
32. DeepLearning.AI. Multi-AI Agent Systems with CrewAI. URL: https://www.deeplearning.ai/short-courses/multi-ai-agent-systems-with-crewai/ (date of access: 21.05.2025).
33. Microsoft AutoGen. Getting Started Guide. URL: https://microsoft.github.io/autogen/0.2/docs/Getting-Started/ (date of access: 21.05.2025).
34. MetaGPT. GitHub repository. URL: https://www.ibm.com/think/topics/metagpt (date of access: 21.05.2025).

35. JetBrains. Towards DSLs for LLM-based Agent Orchestration. URL: https://www.jetbrains.com/mps/concepts/domain-specific-languages/ (date of access: 22.05.2025).
36. Що таке SSL-сертифікат. URL: https://hostpro.ua/blog/ua/what-is-ssl-certificate/ (дата звернення: 28.05.2025).
37. Сервіс WHOIS для перевірки домену. URL: https://www.ukraine.com.ua/domains/whois-service/ (дата звернення: 28.05.2025).
38. CSV RAG Search – CrewAI Docs. URL: https://docs.crewai.com/tools/file-document/csvsearchtool (date of access: 28.05.2025).
39. Портал відкритих даних України. URL: https://data.gov.ua/dataset (дата звернення: 02.06.2025).
40. Serper – The World's Fastest & Cheapest Google Search API. URL: https://serper.dev/ (date of access: 02.06.2025).
41. Website RAG Search – CrewAI Docs. URL: https://docs.crewai.com/tools/search-research/websitesearchtool (date of access: 04.06.2025).
42. Scrape Website – CrewAI Docs. URL: https://docs.crewai.com/tools/web-scraping/scrapewebsitetool (date of access: 10.06.2025).
43. Nmap – Network Mapper. URL: https://nmap.org/ (date of access: 10.06.2025).
44. OpenSanctions – An open-source international database of sanctions data, politically exposed persons, and entities of interest. URL: https://www.opensanctions.org/ (date of access: 10.06.2025).
45. Installation – CrewAI Documentation. URL: https://docs.crewai.com/installation (date of access: 12.06.2025).

ДОДАТКИ

Додаток А

Перелік та характеристика OSINT-інструментів

Таблиця А.1 – Порівняльна характеристика OSINT-інструментів

№	Назва інструменту	Категорія OSINT	Об’єкт збору	Тип реалізації	Призначення
1	Sherlock	SOCMINT	Username, соцмережі	Console	Пошук акаунтів у соцмережах
2	Shodan	TECHINT	IP-адреси, порти, банери	Web	Виявлення мережевих пристроїв
3	Maltego	Universal	IP-адреси, домени, особи	Desktop	Візуалізація зв’язків
4	SpiderFoot	Universal	IP-адреси, домени, email	Desktop /Web	Автоматизована багатоджерельна розвідка
5	GHunt	SOCMINT	Google акаунти	Console	Аналіз Google-профілю
6	PeopleLooker	HUMINT	Імена, адреси, судимості	Web	Пошук інформації про особу
7	Censys	TECHINT	IP-адреси, TLS-сертифікати	Web	Індексація цифрової інфраструктури
8	Recon-ng	Universal	IP-адреси, акаунти, домени	Console	Модульна платформа для розвідки
9	Creepy	GEOINT	Геолокація, соцмережі	Desktop	Аналіз координат із публікацій
10	Exiftool	GEOINT	Метадані зображень	Console	Отримання геоданих та параметрів фото
11	theHarvester	TECHINT	Email, домени	Console	Збір email-адрес і доменів
12	Metagoofil	TECHINT	Документи, метадані	Console	Пошук метаданих у файлах
13	FOCA	TECHINT / GEOINT	Метадані, координати	Desktop	Аналіз документів з метаданими
14	Pipl	HUMINT	Імена, email, телефони	Web	Ідентифікація осіб за контактами
15	WHOIS	TECHINT	Доменні дані	Web/API	Реєстраційна інформація про домени

Продовження таблиці А.1

16	TweetDeck	SOCMINT	Пости, хештеги	Web	Моніторинг Twitter у реальному часі
17	Hunter.io	HUMINT	Email-адреси	Web/API	Пошук корпоративних email за доменом
18	Dnsdumpster	TECHINT	DNS записи	Web	Збір публічної інформації про домени
19	ZoomEye	TECHINT	IP-адреси, сервіси	Web	Індексація відкритих сервісів
20	IntelTechniques	Universal	Різні джерела	Web	Набір OSINT-інструментів для пошуку
21	HaveIBeenPwned	TECHINT	Email, паролі	Web/API	Перевірка витоку облікових даних
22	Google Dorks	Universal	Пошукові індекси	Browser	Розширений пошук Google
23	BuiltWith	TECHINT	Вебтехнології	Web	Інформація про структуру сайтів
24	Wayback Machine	Universal	Архівні дані	Web	Доступ до історичних версій сайтів
25	Dataplor	TECHINT	Бізнес-дані	Web	Аналітика компаній за локацією
26	CheckUsernames	SOCMINT	Username	Web	Пошук username на різних платформах
27	Spyse	TECHINT	IP-адреси, сертифікати	Web	Інфраструктурна розвідка
28	DarkSearch	TECHINT	Індекси даркнету	Web	Пошук інформації в даркнеті
29	ReconBot	Universal	IP-адреси, email, соцмережі	Console	Сценарії розвідки по шаблону
30	Social Searcher	SOCMINT	Пости, акаунти	Web	Пошук по соцмережах у реальному часі

Додаток Б

78

Додаток Б

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Програмний засіб для розвідки з відкритих джерел на основі ШІ-агентів

Автор роботи: Сухарев Нікіта Олександрович

Тип роботи: бакалаврська кваліфікаційна робота

Підрозділ кафедра захисту інформації ФІТКІ , група 1 БКС-21 б

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism **1,17 %**

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

√ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

Завідувач кафедри ЗІ д. т. н., проф. ________ Володимир ЛУЖЕЦЬКИЙ

Гарант освітньої програми «Кібербезпека критичних систем» к. т. н., доц. ________ Юрій БАРИШЕВ

Особа, відповідальна за перевірку ________ Валентина КАПЛУН

З висновком експертної комісії ознайомлений(-на)

Керівник ________ Мусерметова М.М.

Здобувач ________ Сухарев Н.О.

Додаток В

Опис конфігураційних файлів

Файл agents.yaml:


```yaml
local_agent:
    role: >
      Локальний Дослідник Даних.
    goal: >
      Знайти інформацію про фізичних осіб та організації у внутрішній CSV-базі
даних.
    backstory: >
      Ти — ретельний дослідник із ексклюзивним доступом до приватної бази даних
українських контактів.
      Твоя сила — швидко та точно знаходити записи за різними ідентифікаторами,
таким як ПІБ, код ЄДРПОУ,
      електронна пошта або телефон.

internet_researcher:
    role: >
      Спеціаліст з Пошуку в Інтернеті.
    goal: >
      Знайти в інтернеті публічну інформацію, новини та згадки про задану особу чи
організацію.
    backstory: >
      Ти — експерт з OSINT. Вмієш складати влучні пошукові запити для знаходження
релевантних публічних даних,
      новинних статей, соціальних профілів та офіційних записів онлайн.

org_agent:
  role: >
    Агент дослідження організацій, який шукає публічні згадки та ресурси
{topic_of_research}
    через Google Dorks.
  goal: >
    Зібрати ключові публікації, новини, соціальні профілі, прес-релізи
    і базові реєстраційні дані про {topic_of_research}.
  backstory: >
    Ти — досвідчений OSINT-аналітик, який раніше успішно відстежував публічні
згадки
    великих корпорацій у медіа.

person_agent:
  role: >
    Агент дослідження людей, який шукає дані про {topic_of_research}
  goal: >
    Зібрати публічні дані про особу: варіанти імені, соціальні профілі, Google
Dorks, перевірити наявність у санкційних/PEP-списках через OpenSanctions
    та результати пошуку у локальних даних.
  backstory: >
    Ти спеціалізуєшся на OSINT-розвідці персон: використовуєш Google Dorks,
Hunter.io, OpenSanctions для перевірки на публічні дані, санкції/PEP

domain_agent:
  role: >
    Агент дослідження доменів, що шукає технічну інформацію про
{topic_of_research}
  goal: >
    Збирати повну технічну інформацію про {topic_of_research}, включаючи всі типи
DNS-записів, IP-аналіз, SSL-сертифікати, підмережі, PTR-записи, політики поштової
безпеки, піддомени та ознаки внутрішньої
```

```
    інфраструктури.
  backstory: >
    Ти — експерт із технічного OSINT, що досліджує цифрову інфраструктуру за
допомогою WHOIS, DNS-запитів, SSL, IP WHOIS, PTR-lookup, аналізу підмереж і Nmap.
У твоєму арсеналі повний збір DNS-записів із аналізом SPF/DKIM/DMARC, виявлення
адміністраторів через SOA/SRV, а також сканування портів, служб і ОС.

summary_agent:
  role: >
    Агент формування звіту
  goal: >
    Скласти вичерпний та структурований звіт про об’єкт, аналізуючи дані з усіх
джерел
  backstory: >
    Ти — досвідчений аналітик із гострим оком на деталі. Береш сирі дані від
різних дослідників,
    синтезуєш їх, видаляєш дублікати, перевіряєш факти та створюєш чіткий,
структурований
    та практичний фінальний звіт українською мовою.
```

Файл tasks.yaml:

```
local_task:
  description: >
    Знайти будь-яку інформацію про {topic_of_research} у локальній базі даних.
    Шукати збіги за ПІБ, кодом ЄДРПОУ, email або телефоном.
    Повідомити всі знайдені записи в тому вигляді, в якому вони є в базі
  expected_output: >
    Блок тексту з усіма записами, знайденими за запитом у локальній базі
(Назва/ПІБ, керівник, адреса, ЄДРПОУ/Контакти)
  agent: local_agent

internet_search_task:
  description: >
    Провести ґрунтовний пошук в Інтернеті за {topic_of_research}. Знайти публічні
записи, новинні статті, офіційні сайти, огляди та згадки в соціальних мережах.
Підсумувати результати та надати ключові посилання
  expected_output: >
    Коротке резюме знайденої в Інтернеті інформації з важливими URL.
  agent: internet_researcher

org_lookup_task:
  description: >
    Провести OSINT-дослідження організації за назвою {topic_of_research},
    зібрати короткі описи, реєстраційні дані, згадки ЄДРПОУ, структуру власності
та керівний склад, основні фінансові показники (дохід, прибуток, активи), публічні
згадки в медіа та Інтернеті
  expected_output: >
    Список об’єктів з полями:
      - Реєстраційні дані
      - Структура власності та керівний склад
      - Основні фінансові показники:
          - Дохід
          - Прибуток
          - Активи
      - Публічні згадки в медіа та Інтернеті
  agent: org_agent

person_lookup_task:
  description: >
    Провести OSINT-дослідження особи за значенням {topic_of_research}, включно з
```

```
варіантами імені/псевдонімами, соціальними профілями, списком URL з Google Dorks,
перевіркою санкцій/PEP через OpenSanctions (score, topics).
  expected_output: >
    Звіт із:
      1. Варіантами імені/псевдонімами
      2. Соцмережевими профілями
      3. Списком URL з Google Dorks
      4. Результатами перевірки санкцій/PEP (score, topics)
  agent: person_agent

domain_lookup_task:
  description: >
    Провести поглиблене OSINT-дослідження домену {topic_of_research}, зібравши всі
технічні характеристики:
    WHOIS-дані, DNS-записи (A, AAAA, MX, NS, TXT, SOA, SRV), SSL-сертифікат, PTR-
запис, геолокацію,nінфраструктуру, підмережу (CIDR), IP-організацію, відкриті
порти, служби, ознаки фаєрволу, трасування, тип інфраструктури (фаєрвол / хмара /
сервер), та операційну систему.
  expected_output: >
    Структурований OSINT-звіт, що містить:
      1. Загальна інформація про домен (назва, реєстрант, дата реєстрації,
організація, країна).
      2. DNS-записи: A, AAAA, MX, NS, CNAME, TXT, SOA, SRV із TTL.
      3. SSL-сертифікат: емітент, термін дії, альтернативні імена, TLS-версія,
алгоритм, довжина ключа.
      4. IP-інформація: IP, геолокація, ASN, організація, інфраструктура, PTR-
запис.
      5. Аналіз підмережі IP-адреси: CIDR, назва мережі, організація-власник,
країна, контактний email.
      6. Репутація IP, результати з AbuseIPDB, Scylla.
      7. Nmap: відкриті порти, служби, тип ОС, трасування, сигнали фаєрволу.
  agent: domain_agent

summary_task:
  description: >
    Проаналізувати інформацію, надану Локальним Дослідником Даних, Спеціалістом з
Пошуку в Інтернеті, Агентом дослідження людей, Агентом дослідження організацій,
Агентом дослідження доменів.
    Об'єднати дані, видалити дублікати та структурувати в остаточний, вичерпний
профіль. Якщо тема дослідження є доменом, у звіті вказуй її саме у вигляді домену,
а не назви сайту.
    Фінальний звіт має бути українською та містити чіткі розділи.
  expected_output: >
    Докладний та структурований звіт про {topic_of_research} українською мовою.
      Звіт має включати:
      1. Основна інформація.
      3. Дані з публічних джерел (Інтернет).
      4. Дані від агентів досліджень (агент дослідження персон і агент дослідження
організацій)
      5. Дані від агента дослідження доменів
      6. Підсумковий аналіз та ключові факти.

  agent: summary_agent
```

Додаток Г

Код програмного засобу

Файл main.py:

```
import sys
import warnings

from datetime import datetime

from dotenv import load_dotenv
load_dotenv()

from osint.crew import Osint

warnings.filterwarnings("ignore", category=SyntaxWarning, module="pysbd")


if __name__ == "__main__":
    if len(sys.argv) < 2:
        print("Використання: python main.py [run|train|test|replay]
[аргументи...]")
        sys.exit(1)

    command = sys.argv[1]

    if command == "run":
        if len(sys.argv) >= 3:
            topic = sys.argv[2]
        else:
            topic = "Лужецький Володимир Андрійович"
        inputs = {
            "topic_of_research": topic,
            "current_year": str(datetime.now().year)
        }
        Osint().crew().kickoff(inputs=inputs)

    elif command == "train":
        if len(sys.argv) != 5:
            print("Використання: python main.py train [n_iterations] [filename]
[topic_of_research]")
            sys.exit(1)
        inputs = {
            "topic_of_research": sys.argv[4],
            "current_year": str(datetime.now().year)
        }
        Osint().crew().train(n_iterations=int(sys.argv[2]), filename=sys.argv[3],
inputs=inputs)

    elif command == "test":
        if len(sys.argv) != 5:
            print("Використання: python main.py test [n_iterations] [model]
[topic_of_research]")
            sys.exit(1)
        inputs = {
            "topic_of_research": sys.argv[4],
            "current_year": str(datetime.now().year)
```

```
        }
        Osint().crew().test(n_iterations=int(sys.argv[2]), eval_llm=sys.argv[3],
inputs=inputs)

    elif command == "replay":
        if len(sys.argv) != 3:
            print("Використання: python main.py replay [task_id]")
            sys.exit(1)
        Osint().crew().replay(task_id=sys.argv[2])

    else:
        print(f"Невідома команда: {command}")
        sys.exit(1)
```

Файл crew.py:

```
from crewai import Agent, Crew, Process, Task
from crewai.project import CrewBase, agent, crew, task
from crewai.agents.agent_builder.base_agent import BaseAgent
from typing import List

import os
import openai
import sys

from crewai_tools import SerperDevTool
from crewai_tools import ScrapeWebsiteTool
from crewai_tools import WebsiteSearchTool
from crewai_tools import CSVSearchTool

from osint.tools.dns_lookup import DnsLookupTool
from osint.tools.ipinfo_lookup import IPInfoLookupTool
from osint.tools.ptr_lookup_tool import PtrLookupTool
from osint.tools.ssl_certificate_lookup import SSLCertificateLookup
from osint.tools.whois_tool import WhoisTool
from osint.tools.nmap_tool import NmapScanTool
from osint.tools.IPSubnetworkTool import IPSubnetworkTool
from osint.tools.subdomain_finder import SubdomainFinderTool
from osint.tools.google_dorker import GoogleDorkerTool
from osint.tools.hunter_tool import HunterTool
from osint.tools.opensanctions_tool import OpenSanctionsTool
from osint.tools.google_dorker_org import GoogleDorkerOrgTool


serp = SerperDevTool()
csv_tool = CSVSearchTool('C:/Users/nikit/osint-agent/osint/src/osint/data/2019-10-
04_gumanitarka_ier.csv')

@CrewBase
class Osint():
    """Osint crew"""

    agents: List[BaseAgent]
    tasks: List[Task]

    @agent
    def local_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['local_agent'],  # type: ignore[index]
            verbose=True,
            allow_delegation=False,
```

```
            tools=[csv_tool],
            llm='gpt-4o',
            max_iter=1,
            step_callback=None
        )

    @agent
    def internet_researcher(self) -> Agent:
        return Agent(
            config=self.agents_config['internet_researcher'],  # type:
ignore[index]
            verbose=True,
            allow_delegation=False,
            tools=[serp],
            llm='gpt-4o'
        )

    @agent
    def org_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['org_agent'],  # type: ignore[index]
            verbose=True,
            tools=[
                GoogleDorkerOrgTool(),
                SerperDevTool()
            ],
            llm='gpt-4o'
        )

    @agent
    def person_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['person_agent'], # type: ignore[index]
            verbose=True,
            tools=[
                GoogleDorkerTool(),
                HunterTool(),
                OpenSanctionsTool()
            ],
            llm='gpt-4o'
        )

    @agent
    def domain_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['domain_agent'], # type: ignore[index]
            verbose=True,
            tools=[WhoisTool(),
                   DnsLookupTool(),
                   SSLCertificateLookup(),
                   IPInfoLookupTool(),
                   NmapScanTool(),
                   PtrLookupTool(),
                   IPSubnetworkTool(),
                   SubdomainFinderTool()
                   ],
            llm='gpt-4o'
        )

    @agent
    def summary_agent(self) -> Agent:
        return Agent(
            config=self.agents_config['summary_agent'], # type: ignore[index]
```

```
            verbose=True,
            llm='gpt-4o'
        )

    @task
    def local_task(self) -> Task:
        return Task(
            config=self.tasks_config['local_task'],  # type: ignore[index]
            output_file=None  # або за потреби
        )

    @task
    def internet_search_task(self) -> Task:
        return Task(
            config=self.tasks_config['internet_search_task'],  # type:
ignore[index]
            output_file=None  # або за потреби
        )

    @task
    def org_lookup_task(self) -> Task:
        return Task(
            config=self.tasks_config['org_lookup_task'],  # type: ignore[index]
            output_file=None  # або за потреби
        )

    @task
    def person_lookup_task(self) -> Task:
        return Task(
            config=self.tasks_config['person_lookup_task'],  # type: ignore[index]
            output_file=None  # або за потреби
        )

    @task
    def domain_lookup_task(self) -> Task:
        return Task(
            config=self.tasks_config['domain_lookup_task'], # type: ignore[index]
            output_file=None  # або за потреби
        )

    @task
    def summary_task(self) -> Task:
        return Task(
            config=self.tasks_config['summary_task'], # type: ignore[index]
            output_file='report.md'
        )

    @crew
    def crew(self) -> Crew:
        """Creates the Osint crew"""
        return Crew(
            agents=self.agents, # Automatically created by the @agent decorator
            tasks=self.tasks, # Automatically created by the @task decorator
            process=Process.sequential,
            verbose=True,
        )
```

Файл dns_lookup.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
```

```
import dns.resolver

class DnsLookupInput(BaseModel):
    """Input schema for DnsLookupTool."""
    domain: str = Field(..., description="Доменне ім’я для отримання повних DNS-
записів")

class DnsLookupTool(BaseTool):
    name: str = "dns_lookup"
    description: str = (
        "Отримує DNS-записи типів A, AAAA, MX, NS, CNAME, TXT, SOA та SRV для
заданого домену. "
        "Показує TTL кожного запису, якщо можливо. "
        "Корисно для виявлення інфраструктури, політик безпеки пошти,
адміністраторів зони та служб."
    )
    args_schema: Type[BaseModel] = DnsLookupInput

    def _run(self, domain: str) -> str:
        record_types = ["A", "AAAA", "MX", "NS", "CNAME", "TXT", "SOA", "SRV"]
        results = []

        for rtype in record_types:
            try:
                answers = dns.resolver.resolve(domain, rtype)
                ttl = answers.rrset.ttl if hasattr(answers, 'rrset') else "N/A"
                records = [str(r.to_text()) for r in answers]
                for record in records:
                    results.append(f"{rtype} (TTL: {ttl}): {record}")
            except dns.resolver.NoAnswer:
                results.append(f"{rtype}: Відповідь відсутня")
            except dns.resolver.NXDOMAIN:
                results.append(f"{rtype}: Домен не існує")
            except dns.exception.DNSException as e:
                results.append(f"{rtype}: Помилка - {str(e)}")

        return "\n".join(results)
```

Файл google_dorker.py:

```
import os
import requests
from typing import List, Dict, Type, ClassVar
from pydantic import BaseModel, Field
from crewai.tools import BaseTool
from dotenv import load_dotenv

# Завантажуємо змінні середовища з .env
load_dotenv()

class GoogleDorkerInput(BaseModel):
    dork: str = Field(..., description="Dork-рядок для пошуку")
    max_results: int = Field(
        10,
        ge=1,
        le=10,
        description="Макс. кількість результатів (1-10)"
    )

class GoogleDorkerTool(BaseTool):
    # Ось тут обов'язково вказуємо типи
    name: str = "google_dorker"
```

```
    description: str = "Шукає URL через Google Custom Search JSON API"
    args_schema: Type[GoogleDorkerInput] = GoogleDorkerInput

    # Це константа, не поле Pydantic — позначаємо ClassVar
    BASE_URL: ClassVar[str] = "https://www.googleapis.com/customsearch/v1"

    def _run(self, dork: str, max_results: int = 10) -> List[Dict[str, str]]:
        api_key = os.getenv("GOOGLE_API_KEY")
        cx      = os.getenv("GOOGLE_CX")
        if not api_key or not cx:
            raise RuntimeError("Не знайдено GOOGLE_API_KEY або GOOGLE_CX у .env")

        params = {
            "key": api_key,
            "cx": cx,
            "q": dork,
            "num": max_results
        }
        response = requests.get(self.BASE_URL, params=params, timeout=10)
        response.raise_for_status()
        items = response.json().get("items", [])
        return [
            {
                "title": item.get("title"),
                "snippet": item.get("snippet"),
                "url": item.get("link")
            }
            for item in items
        ]
```

Файл google_dorker_org.py:

```
import os
import requests
from typing import List, Dict, Type, ClassVar
from pydantic import BaseModel, Field
from crewai.tools import BaseTool
from dotenv import load_dotenv

load_dotenv()

class GoogleDorkerOrgInput(BaseModel):
    org: str = Field(..., description="Назва організації для пошуку")
    max_results: int = Field(
        10, ge=1, le=10,
        description="Макс. кількість результатів (1-10)"
    )

class GoogleDorkerOrgTool(BaseTool):
    name: str = "google_dorker_org"
    description: str = "Шукає URL та згадки про організацію через Google Dorks"
    args_schema: Type[GoogleDorkerOrgInput] = GoogleDorkerOrgInput

    BASE_URL: ClassVar[str] = "https://www.googleapis.com/customsearch/v1"

    def _run(self, org: str, max_results: int = 10) -> List[Dict[str, str]]:
        api_key = os.getenv("GOOGLE_API_KEY")
        cx      = os.getenv("GOOGLE_CX")
        if not api_key or not cx:
            raise RuntimeError("Не знайдено GOOGLE_API_KEY або GOOGLE_CX у .env")
        # автоматично формуємо dork-рядок з назвою організації
        dork_query = f'"{org}" site:*/press OR site:*/news OR site:linkedin.com OR
```

```
site:facebook.com'
        params = {
            "key": api_key,
            "cx": cx,
            "q": dork_query,
            "num": max_results
        }
        resp = requests.get(self.BASE_URL, params=params, timeout=10)
        resp.raise_for_status()
        items = resp.json().get("items", [])
        return [
            {"title": it.get("title"),
             "snippet": it.get("snippet"),
             "url": it.get("link")}
            for it in items
        ]
```

Файл hunter_tool.py:

```
import os
from typing import Type
from dotenv import load_dotenv
import requests
from pydantic import BaseModel, Field
from crewai.tools import BaseTool

# Підгружаємо .env при імпорті модуля
load_dotenv()

class HunterInput(BaseModel):
    domain: str | None = Field(None, description="Домен для пошуку email-ів")
    first_name: str | None = Field(None, description="Ім'я для побудови
комбінацій")
    last_name: str | None = Field(None, description="Прізвище для побудови
комбінацій")

class HunterTool(BaseTool):
    name: str = "hunter"
    description: str = "Збирає та валідує email-адреси через API Hunter.io"
    args_schema: Type[HunterInput] = HunterInput

    def _run(self, domain=None, first_name=None, last_name=None):
        api_key = os.getenv("HUNTER_API_KEY")
        if not api_key:
            raise RuntimeError("Не вказано HUNTER_API_KEY у .env")
        headers = {"Accept": "application/json"}
        results = []

        if domain:
            resp = requests.get(
                "https://api.hunter.io/v2/domain-search",
                params={"domain": domain, "api_key": api_key},
                headers=headers,
                timeout=10
            )
            data = resp.json().get("data", {})
            for e in data.get("emails", []):
                results.append({
                    "email": e["value"],
                    "type": e["type"],
                    "confidence": e["confidence"],
                    "sources": [s["domain"] for s in e.get("sources", [])]
```

```
                })
        elif first_name and last_name:
            resp = requests.get(
                "https://api.hunter.io/v2/email-finder",
                params={
                    "domain": "",
                    "first_name": first_name,
                    "last_name": last_name,
                    "api_key": api_key
                },
                headers=headers,
                timeout=10
            )
            e = resp.json().get("data", {})
            if e.get("email"):
                results.append({
                    "email": e["email"],
                    "confidence": e["confidence"],
                    "pattern": e["pattern"]
                })
        return results
```

Файл ipinfo_lookup.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
import requests
import socket
import os

class IPInfoToolInput(BaseModel):
    domain: str = Field(..., description="Доменне ім’я або IP для аналізу.")

class IPInfoLookupTool(BaseTool):
    name: str = "ipinfo_lookup"
    description: str = (
        "Отримує базову інфраструктурну інформацію про IP-адресу з ipinfo.io: ASN,
провайдер, геолокація, організація."
    )
    args_schema: Type[BaseModel] = IPInfoToolInput

    def _run(self, domain: str) -> str:
        try:
            # Перетворюємо домен на IP
            ip = socket.gethostbyname(domain)
            token = os.getenv("IPINFO_TOKEN")  # Не обов'язково, але можна додати
у .env
            url = f"https://ipinfo.io/{ip}/json"
            if token:
                url += f"?token={token}"
            response = requests.get(url)
            if response.status_code == 200:
                return response.text
            else:
                return f"Error: Status code {response.status_code}, response:
{response.text}"
        except Exception as e:
            return f"Exception during ipinfo lookup: {e}"
```

Файл IPSubnetworkTool.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
from ipwhois import IPWhois


class IPSubnetworkInput(BaseModel):
    """Input schema for IPSubnetworkTool."""
    ip_address: str = Field(..., description="IP-адреса для аналізу підмережі")


class IPSubnetworkTool(BaseTool):
    name: str = "ip_subnetwork_lookup"
    description: str = (
        "Аналізує IP-адресу та визначає підмережу, CIDR, маску, організацію, яка
володіє блоком, і контактну інформацію."
    )
    args_schema: Type[BaseModel] = IPSubnetworkInput

    def _run(self, ip_address: str) -> str:
        try:
            obj = IPWhois(ip_address)
            res = obj.lookup_rdap()

            network = res.get("network", {})
            cidr = network.get("cidr", "Невідомо")
            name = network.get("name", "Невідомо")
            country = network.get("country", "Невідомо")
            org = res.get("asn_description", "Невідомо")
            email = next((c.get("email") for c in res.get("objects", {}).values()
if isinstance(c, dict) and c.get("email")), "Невідомо")

            return (
                f"IP-адреса: {ip_address}\n"
                f"CIDR: {cidr}\n"
                f"Назва мережі: {name}\n"
                f"Країна: {country}\n"
                f"Організація: {org}\n"
                f"Контактний email: {email}"
            )

        except Exception as e:
            return f"Помилка при аналізі IP-підмережі: {str(e)}"
```

Файл nmap_tool.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
import subprocess


class NmapScanInput(BaseModel):
    """Input schema for NmapScanTool."""
    target: str = Field(..., description="IP-адреса або домен для сканування")


class NmapScanTool(BaseTool):
    name: str = "nmap_scan"
```

```
    description: str = (
        "Сканує IP-адресу або домен за допомогою Nmap для виявлення портів, ОС,
фаєрволів і веб-сервісів."
    )
    args_schema: Type[BaseModel] = NmapScanInput

    def _run(self, target: str) -> str:
        try:
            command = [
                "nmap",
                "-sS",            # TCP SYN scan
                "-sV",            # Версії сервісів
                "-O",             # Визначення ОС
                "--osscan-guess", # Навіть якщо не впевнено
                "--traceroute",   # Маршрут до цілі
                "--script=http-title,firewalk",  # HTTP-заголовки та фаєрвол
                "--top-ports", "1000",  # Найпоширеніші порти
                target
            ]

            result = subprocess.run(
                command,
                capture_output=True,
                text=True,
                timeout=180
            )
            return result.stdout

        except subprocess.TimeoutExpired:
            return "Сканування перевищило час виконання (timeout)."
        except Exception as e:
            return f"Помилка під час виконання сканування: {e}"
```

Файл opensanctions_tool.py:

```
import os
import requests
from typing import Any, Dict, List, Type

from pydantic import BaseModel, Field
from crewai.tools import BaseTool


class OpenSanctionsQuery(BaseModel):
    schema: str = Field(..., description="Тип сутності для пошуку (Person, Company
тощо)")
    properties: Dict[str, List[str]] = Field(
        ..., description="Словник властивостей для пошуку (name, birthDate тощо)"
    )


class OpenSanctionsTool(BaseTool):
    name: str = "opensanctions_match"
    description: str = (
        "Шукає збіги в базі OpenSanctions через matching API "
        "(default, sanctions, peps) за заданими властивостями."
    )
    args_schema: Type[OpenSanctionsQuery] = OpenSanctionsQuery

    def _run(self, schema: str, properties: Dict[str, List[str]]) ->
List[Dict[str, Any]]:
        api_key = os.getenv("OPENSANCTIONS_API_KEY")
```

```
        if not api_key:
            raise RuntimeError("Не вказано OPENSANCTIONS_API_KEY у середовищі")

        # Доступні датасети: default, sanctions, peps
        dataset = "default"
        url = f"https://api.opensanctions.org/match/{dataset}"

        session = requests.Session()
        session.headers["Authorization"] = f"ApiKey {api_key}"

        # Формуємо запит у вигляді batch (тут лише один запит з ID "q1")
        payload = {
            "queries": {
                "q1": {
                    "schema": schema,
                    "properties": properties
                }
            }
        }
        params = {"algorithm": "best"}  # можна змінити на "name-based",
"sanctions" тощо

        resp = session.post(url, json=payload, params=params, timeout=10)
        resp.raise_for_status()

        # Відповідь містить responses.q1.results
        response_batch = resp.json().get("responses", {}).get("q1", {})
        return response_batch.get("results", [])
```

Файл ptr_lookup_tool.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
import socket

class PtrLookupInput(BaseModel):
    """Input schema for PtrLookupTool."""
    ip_address: str = Field(..., description="IP-адреса для виконання зворотного
DNS lookup")

class PtrLookupTool(BaseTool):
    name: str = "ptr_lookup"
    description: str = (
        "Виконує зворотний DNS lookup (PTR-запит) для IP-адреси, щоб виявити
пов'язане доменне ім’я."
    )
    args_schema: Type[BaseModel] = PtrLookupInput

    def _run(self, ip_address: str) -> str:
        try:
            hostname, _, _ = socket.gethostbyaddr(ip_address)
            return f"PTR-запис для IP {ip_address}: {hostname}"
        except Exception as e:
            return f"Не вдалося знайти PTR-запис для IP {ip_address}: {str(e)}"
```

Файл ssl_certificate_lookup.py:

```
from crewai.tools import BaseTool
from typing import Type
```

```
from pydantic import BaseModel, Field
import ssl
import socket
from datetime import datetime
from cryptography import x509
from cryptography.hazmat.backends import default_backend


class SSLCertificateLookupInput(BaseModel):
    """Input schema for SSLCertificateLookup."""
    domain: str = Field(..., description="Domain to fetch the SSL/TLS certificate
from.")


class SSLCertificateLookup(BaseTool):
    name: str = "ssl_certificate_lookup"
    description: str = (
        "Отримує SSL/TLS-сертифікат для домену, включно з емітентом, терміном дії,
альтернативними іменами, "
        "версією TLS, криптографічним алгоритмом та довжиною ключа."
    )
    args_schema: Type[BaseModel] = SSLCertificateLookupInput

    def _run(self, domain: str) -> str:
        try:
            ctx = ssl.create_default_context()
            with ctx.wrap_socket(socket.socket(), server_hostname=domain) as s:
                s.settimeout(5.0)
                s.connect((domain, 443))
                cert_bin = s.getpeercert(binary_form=True)
                tls_version = s.version()

            cert = x509.load_der_x509_certificate(cert_bin, default_backend())

            subject = cert.subject.rfc4514_string()
            issuer = cert.issuer.rfc4514_string()
            valid_from = cert.not_valid_before_utc.strftime("%Y-%m-%d %H:%M:%S")
            valid_to = cert.not_valid_after_utc.strftime("%Y-%m-%d %H:%M:%S")
            sig_algo = cert.signature_algorithm_oid._name
            key_size = cert.public_key().key_size

            try:
                san_extension =
cert.extensions.get_extension_for_class(x509.SubjectAlternativeName)
                san = ",
".join(san_extension.value.get_values_for_type(x509.DNSName))
            except x509.ExtensionNotFound:
                san = "Not available"

            return (
                f"SSL Certificate for: {domain}\n"
                f"Issuer: {issuer}\n"
                f"Subject: {subject}\n"
                f"Valid From: {valid_from}\n"
                f"Valid To: {valid_to}\n"
                f"Subject Alternative Names (SAN): {san}\n"
                f"TLS Version: {tls_version}\n"
                f"Signature Algorithm: {sig_algo}\n"
                f"Key Size: {key_size} bits"
            )

        except Exception as e:
            return f"Error retrieving SSL certificate for {domain}: {str(e)}"
```

Файл subdomain_finder.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
import requests
import json


class SubdomainFinderInput(BaseModel):
    """Input schema for SubdomainFinderTool."""
    domain: str = Field(..., description="Основний домен для пошуку піддоменів")


class SubdomainFinderTool(BaseTool):
    name: str = "subdomain_finder"
    description: str = (
        "Шукає піддомени домену через аналіз публічних сертифікатів (CRT.sh)."
    )
    args_schema: Type[BaseModel] = SubdomainFinderInput

    def _run(self, domain: str) -> str:
        try:
            url = f"https://crt.sh/?q=%25.{domain}&output=json"
            response = requests.get(url, timeout=10)
            if response.status_code != 200:
                return f"Помилка при запиті до crt.sh: код {response.status_code}"

            data = response.json()
            subdomains = set()
            for entry in data:
                name_value = entry.get("name_value", "")
                for sub in name_value.split("\n"):
                    if sub.endswith(domain):
                        subdomains.add(sub.strip())

            if not subdomains:
                return f"Піддомени для {domain} не знайдені."

            return f"Знайдені піддомени для {domain}:\n" +
"\n".join(sorted(subdomains))

        except Exception as e:
            return f"Помилка при пошуку піддоменів: {str(e)}"
```

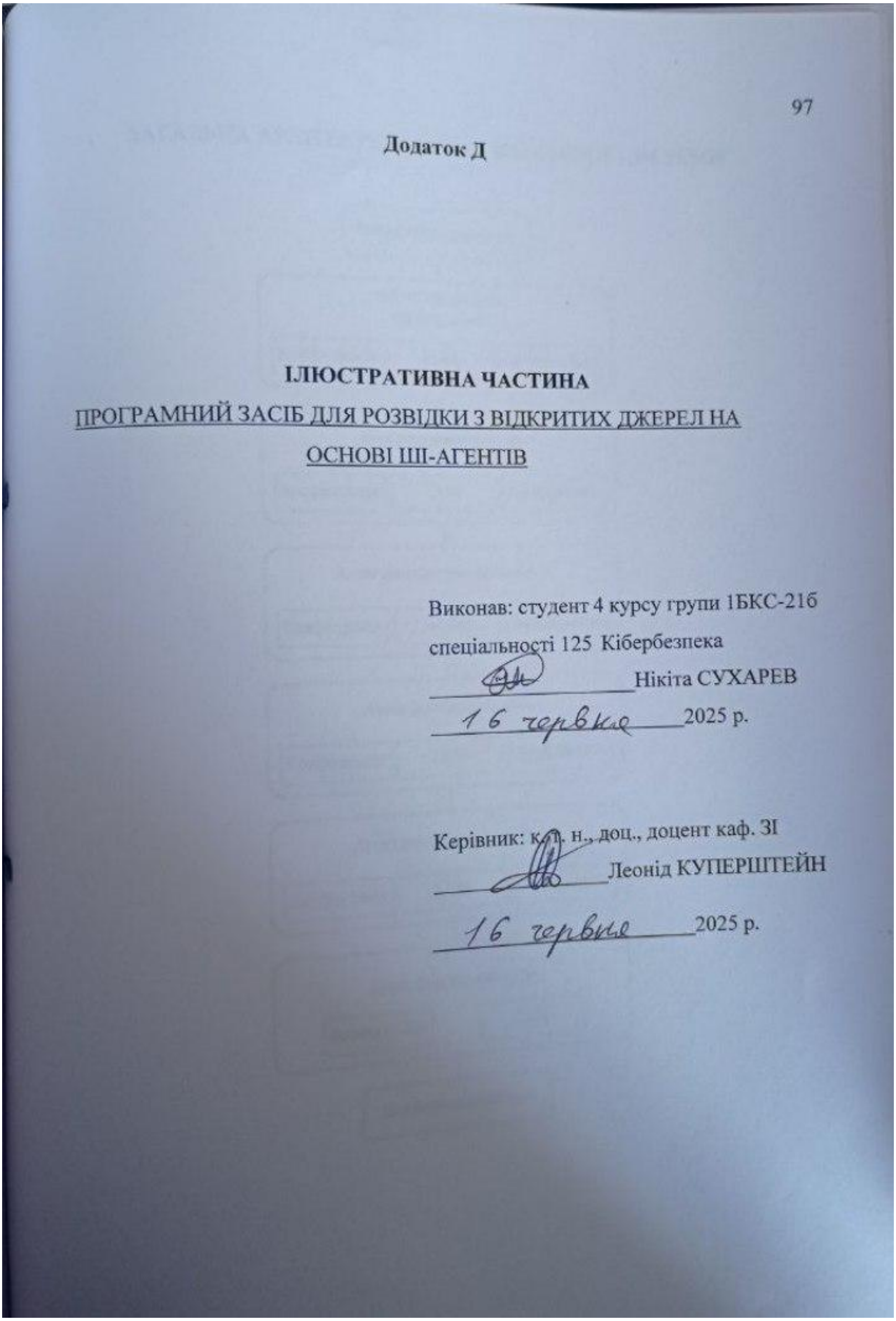
Файл whois_tool.py:

```
from crewai.tools import BaseTool
from typing import Type
from pydantic import BaseModel, Field
import whois
import socket
import subprocess
import re

class WhoisToolInput(BaseModel):
    """Input schema for WhoisTool."""
    domain: str = Field(..., description="Доменне ім’я для отримання WHOIS
інформації")

class WhoisTool(BaseTool):
```

```
    name: str = "whois_lookup"
    description: str = (
        "Отримує WHOIS інформацію про домен і IP-адресу: організація-власник,
дати, підмережа, ASN, "
        "хостинг-провайдер та інші реєстраційні деталі."
    )
    args_schema: Type[BaseModel] = WhoisToolInput

    def _run(self, domain: str) -> str:
        try:
            # WHOIS по домену
            w = whois.whois(domain)
            dnssec_raw = getattr(w, 'dnssec', None)
            if dnssec_raw == "unsigned":
                dnssec_status = "Вимкнено"
            elif dnssec_raw in ["signed", "signedDelegation", True]:
                dnssec_status = "Увімкнено"
            else:
                dnssec_status = "Не визначено"

            # Об’єднання: Організація-власник
            org_fields = filter(None, [getattr(w, 'org', None), getattr(w, 'name',
None)])
            org_owner = " / ".join(org_fields) if org_fields else "Не вказано"

            domain_info = [
                f"Доменне ім’я: {w.domain_name}",
                f"Реєстратор: {w.registrar}",
                f"Дата створення: {w.creation_date}",
                f"Дата закінчення: {w.expiration_date}",
                f"Статус: {w.status}",
                f"Службовий email: {w.emails}",
                f"DNSSEC: {dnssec_status}",
                f"Організація-власник: {org_owner}",
                f"Країна: {getattr(w, 'country', 'Невизначено')}",
                f"Адреса: {getattr(w, 'address', 'Невизначено')}"
            ]

            # WHOIS по IP
            try:
                ip = socket.gethostbyname(domain)
                ip_whois_raw = subprocess.run(
                    ["whois", ip], capture_output=True, text=True, timeout=30
                ).stdout

                ip_info = [
                    f"IP-адреса: {ip}",
                    self._extract_field(ip_whois_raw,
r"(OrgName|org|organisation):\s*(.*)", "Організація-IP"),
                    self._extract_field(ip_whois_raw, r"(CIDR|inetnum):\s*(.*)",
"Підмережа"),
                    self._extract_field(ip_whois_raw,
r"(netname|NetName):\s*(.*)", "NetName"),
                    self._extract_field(ip_whois_raw,
r"(descr|Description):\s*(.*)", "Опис"),
                    self._extract_field(ip_whois_raw, r"(country):\s*(.*)",
"Країна-IP"),
                    self._extract_field(ip_whois_raw, r"(aut-num|origin):\s*(.*)",
"ASN")
                ]

            except Exception as ip_e:
                ip_info = [f"WHOIS по IP: не вдалося отримати — {str(ip_e)}"]
```

```
            return "[WHOIS по домену]\n" + "\n".join(domain_info) + "\n\n[WHOIS по IP]\n" + "\n".join(ip_info)

        except Exception as e:
            return f"Помилка при виконанні WHOIS-запиту: {str(e)}"

    def _extract_field(self, text: str, pattern: str, label: str) -> str:
        match = re.search(pattern, text, re.IGNORECASE)
        if match:
            return f"{label}: {match.group(2).strip()}"
        return f"{label}: Не знайдено"
```

Додаток Д


97

Додаток Д

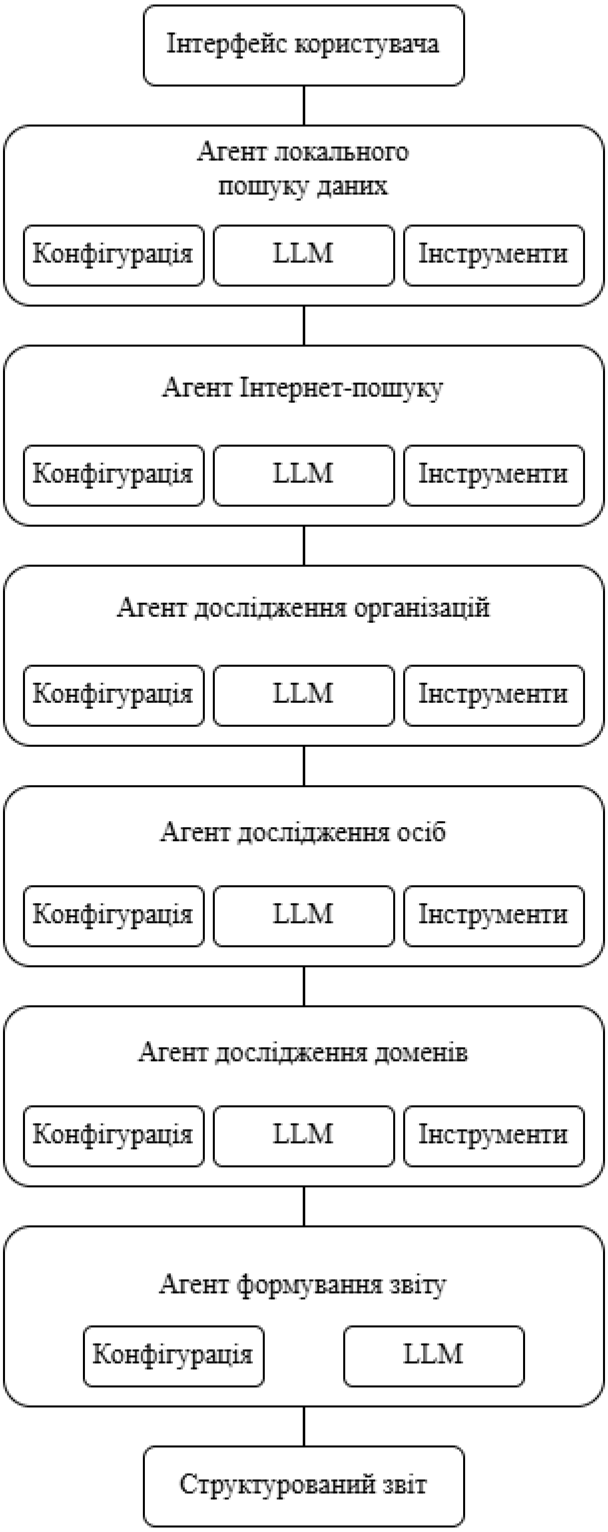
ІЛЮСТРАТИВНА ЧАСТИНА

ПРОГРАМНИЙ ЗАСІБ ДЛЯ РОЗВІДКИ З ВІДКРИТИХ ДЖЕРЕЛ НА ОСНОВІ ШІ-АГЕНТІВ

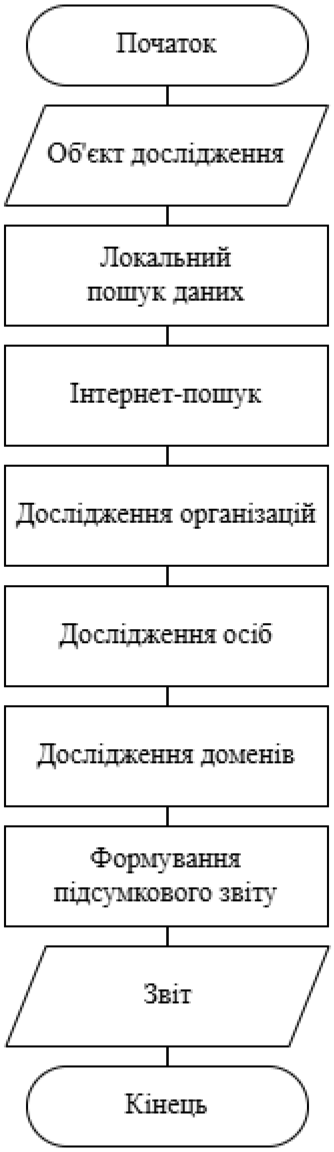
Виконав: студент 4 курсу групи 1БКС-21б
спеціальності 125 Кібербезпека
_______________Нікіта СУХАРЕВ
16 червня 2025 р.

Керівник: к.т. н., доц., доцент каф. ЗІ
_______________Леонід КУПЕРШТЕЙН
16 червня 2025 р.

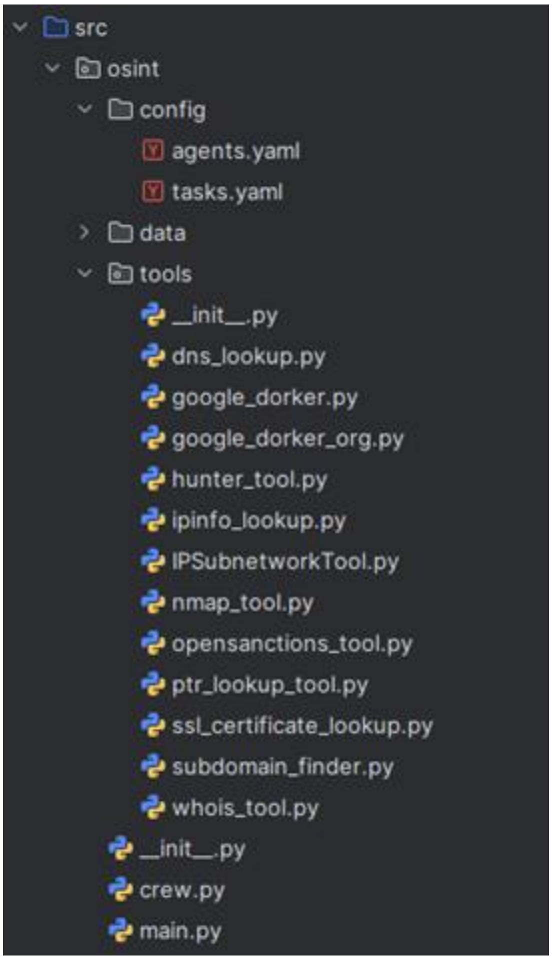
ЗАГАЛЬНА АРХІТЕКТУРА МУЛЬТИАГЕНТНОЇ СИСТЕМИ



ЗАГАЛЬНА СХЕМА АЛГОРИТМУ ПРОГРАМНОГО ЗАСОБУ



СТРУКТУРА ПРОЕКТУ РОЗВІДКИ З ВІДКРИТИХ ДЖЕРЕЛ



ЗВІТ З РОЗВІДКИ ПРО ОРГАНІЗАЦІЮ

Звіт про Благодійний Фонд «РУКА МИРУ»

1. Основна інформація:
   - Назва: Благодійний Фонд «РУКА МИРУ»
   - Код ЄДРПОУ: 39327910
   - Дата реєстрації: 31.07.2014
   - Юридична адреса: 45243, Волинська обл., Ківерцівський р-н, с. Прилуцьке, вул. Жовтнева
2. Дані з публічних джерел (Інтернет):
   - Instagram профіль: Благодійний Фонд «Рука Миру» на Instagram
   - YouControl: Деталі організації на YouControl
   - Опендатабот: Інформація про юридичну особу БО «БФ «РУКА МИРУ»
   - Публікація на Укрінформ: Завдання волонтерських фондів - Укрінформ
3. Дані від агентів досліджень:
   - Організація та керівник: Керівником є Качула Володимир Вікторович
   - Відомості про фінансові показники: Дохід, прибуток та активи не зазначені.
   - Соціальні згадки та публікації пов'язані з наданням гуманітарної допомоги, співпрацею з іншими благодійними організаціями.
4. Дані від агента дослідження доменів:
   - Доменні записи та SSL сертифікати відсутні, що свідчить про відсутність точки доступу чи активності в мережі Інтернет.
   - Немає даних про IP адреси чи субмережі, що пов'язано з малою активністю в цифровому просторі.
5. Підсумковий аналіз та ключові факти:
   - Благодійний фонд «РУКА МИРУ» здійснює свою діяльність переважно в офлайні, без значного впливу в Інтернеті.
   - Основними напрямками діяльності є надання гуманітарної допомоги, підтримка соціально незахищених груп населення.
   - Організація співпрацює з іншими волонтерськими та благодійними організаціями з метою підвищення ефективності своєї діяльності.
   - Фонд не має конфліктів із санкційними списками та не фігурує у звітах про потенційні ризики.

ЗВІТ З РОЗВІДКИ ПРО ОСОБУ

Звіт про Белінську Христину Олексіївну

1. Основна інформація

- **Ім'я:** Белінська Христина Олексіївна
- **Альтернативні імена/псевдоніми:** Kristina Belinska, Khrystyna Belinska, Кристина Білінська
- **Соціальні профілі:** Instagram - Христина Білінська

2. Дані з публічних джерел (Інтернет)

- **ФОП:**
  - Реєстраційні дані: ФОП Белінська Христина Олексіївна зареєстрована в місті Кропивницький. ЄДРПОУ: 43029438, статус - припинено.
  - Деталі в YouControl
- **Організації:**
  - "АКН КО", ЄДРПОУ: 42320716 - Христина є засновником. Статутний капітал 1,000 грн.
  - Деталі ТОВ "АКН КО"

3. Дані від агентів досліджень

Агент дослідження персон

- **Фінансовий статус:**
  - Виявлено 7 податкових боргів загальною сумою 0,40225 тисяч гривень.
  - Деталі податкових боргів
- **Судові справи:**
  - Згадування у Реєстрі судових рішень.
  - БЕЛІНСЬКА ХРИСТИНА - Реєстр судових рішень

Агент дослідження організацій

- **Організації:** Белінська Христина Олексіївна асоціюється з благодійною організацією "Філія доброта".
  - БФ "ФІЛІЯ ДОБРА"

ЗВІТ З РОЗВІДКИ ПРО ДОМЕН

Звіт про Molfar

1. Основна інформація

- **Назва**: Molfar
- **Вебсайт**: https://molfar.com
- **Тип**: Агентство відкритих джерел (OSINT)
- **Домен**: molfar.com
- **Дата реєстрації**: 2016-02-04

2. Дані з публічних джерел (Інтернет)

- **Офіційний вебсайт Molfar**: Найбільше агентство розвідки в Україні, яке спеціалізується на роботі з відкритими джерелами, фінансовим та військовим аналізом.
- **Інструменти та ресурси OSINT**: Перелік інструментів для аналізу відкритих джерел доступний на Molfar Official Website.
- **Блог Мольфар**: Містить всі OSINT-розслідування та статті, Molfar Blog.
- **Перелік ворогів і зрадників України**: Список осіб, які становлять загрозу національній безпеці — Enemies List.

3. Дані від агентів досліджень

- **Структура власності та керівний склад**:
  - **Генеральний директор**: Олексій Руденко
  - **Фінансовий директор**: Анастасія Коваль
- **Публічні згадки в медіа**:
  - Molfar UK | LinkedIn — Агентство підтримки аналітики.
  - Molfar Intelligence Institute | LinkedIn — Навчальні програми в галузі OSINT.

4. Дані від агента дослідження доменів

- **DNS-записи**:
  - IP: 172.67.73.231
  - MX: aspmx.l.google.com та інші
  - NS: olga.ns.cloudflare.com, howard.ns.cloudflare.com
- **SSL-сертифікат**:
  - Емітент: Google Trust Services
- **IP-інформація**:
  - Розташування: San Francisco, California, US
  - Інфраструктура: Cloudflare

5. Підсумковий аналіз та ключові факти

- Molfar є провідним агентством в Україні, яке спеціалізується на аналізі відкритих джерел інформації, маючи значний вплив в галузі розслідувань корупції та міжнародної безпеки.
- Платформа активно розвиває навчальні програми з аналітики OSINT та HUMINT, зокрема впроваджує курси, які фокусуються на реальних практичних кейсах.
- Компанія забезпечує безпечну цифрову інфраструктуру, зокрема, через використання послуг Cloudflare для керування мережею та безпекою даних.

РЕЗУЛЬТАТИ НЕУСПІШНОЇ РОЗВІДКИ З ВІДКРИТИХ ДЖЕРЕЛ

Звіт про домен "woqouwthw.com"

1. Основна інформація

- Домен "woqouwthw.com" не має доступної основної інформації, такої як діяльність, контактні дані або інші характерні відомості про організацію, що могла б володіти ним.

2. Дані з публічних джерел (Інтернет)

- Публічна інформація, пов'язана безпосередньо з доменом "woqouwthw.com", відсутня. Проте численні посилання ведуть до сайтів і продуктів, пов'язаних з "World of Warcraft". Немає явних або підтверджених зв'язків між цим доменом і даними ресурсами, але присутність у пошукових запитах може наводити на асоціації з подібними назвами.

3. Дані від агентів досліджень (агент дослідження персон і агент дослідження організацій)

- Варіанти імені/псевдонімами та соцмережеві профілі відсутні. Можливою асоціацією є "World of Warcraft", але жодних конкретних даних немає.
- Жоден URL безпосередньо не зв'язаний з доменом, але присутні згадки продуктів "World of Warcraft".
- Не виявлено жодних записи у санкційних або PEP списках щодо цього домену.

4. Дані від агента дослідження доменів

- Домен "woqouwthw.com" є неактивним. Немає жодних записів або технічної інформації (WHOIS, DNS, SSL сертифікати тощо), які б вказували на активність домену.

5. Підсумковий аналіз та ключові факти

- Домен "woqouwthw.com" не має активного статусу і жодних явних публічних або організаційних даних.
- Ймовірна асоціація з "World of Warcraft" заснована виключно на подібності імені, але конкретних підтверджень немає.
- Відсутність технічних і реєстраційних даних робить неможливим детальне дослідження або з'ясування специфікацій та власника домену.