Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук

Допущено до захисту
Завідувач кафедри
комп’ютерних наук
доктор техн. наук, професор

Андрій БОНДАРЧУК

« 01 » червня 2026 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «бакалавр»

Спеціальність 122 Комп’ютерні науки

Освітня програма 122.00.01 Інформатика

Тема: ІНТЕРАКТИВНИЙ ПРОГРАМНИЙ ДОДАТОК АНАЛІЗУ ДОСТОВІРНОСТІ КОНТЕНТУ НОВИННИХ ПОРТАЛІВ

Виконала
студентка групи ІНб-1-22-4.0д
Стаднік Ірина Петрівна

_____

Науковий керівник
Доктор філософії,
Турукало А.В.

_____

Київ – 2026

Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп'ютерних наук

«Затверджую»
Завідувач кафедри комп'ютерних наук
доктор технічних наук, професор
Андрій БОНДАРЧУК
« 31 » жовтня 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА «ІНТЕРАКТИВНИЙ ПРОГРАМНИЙ ДОДАТОК АНАЛІЗУ ДОСТОВІРНОСТІ КОНТЕНТУ НОВИННИХ ПОРТАЛІВ»

Виконавець – студентка групи ІНб-1-22-4.0д,
спеціальності 122 Комп'ютерні науки,
освітньої програми 122.00.01 Інформатика
Стаднік Ірина Петрівна

1. Вихідні дані: новинний контент із відкритих джерел, зокрема RSS-стрічок, Telegram-каналів і вебресурсів; документація .NET, C#, Windows Forms, бібліотек для роботи з RSS, HTTP-запитами, JSON та зовнішніми API; наукові джерела з питань фейкових новин, дезінформації, фактчекінгу, OSINT, NLP, машинного навчання та штучного інтелекту.
2. Основні завдання: проаналізувати предметну область перевірки достовірності новинного контенту; дослідити методи фактчекінгу, OSINT, NLP і машинного навчання; визначити мету, об'єкт, предмет і методи дослідження; сформувати вимоги до програмного додатку; спроєктувати архітектуру настільного застосунку для збору, фільтрації та аналізу новин із RSS і Telegram-джерел; реалізувати модулі завантаження, перегляду, пошуку, фільтрації та попереднього AI-аналізу новин; провести функціональне, нефункціональне, безпекове та регресійне тестування; оформити роботу відповідно до вимог і підготувати матеріали до захисту.
3. Пояснювальна записка: обсяг 35–45 сторінок основного тексту формату А4 з дотриманням вимог стандарту та методичних рекомендацій кафедри. Додатки до основного обсягу не входять.
4. Графічні матеріали: структурні схеми, UML-діаграми, схема архітектури додатку, схема алгоритму перевірки достовірності новин, ілюстрації інтерфейсу, таблиці результатів тестування та комп'ютерна презентація доповіді.
5. Додатки: фрагменти вихідного коду модулів RSS, Telegram, пошуку, фільтрації, AI-аналізу, формування результатів перевірки, а також тест-кейси та сценарії системного тестування.
6. Строк подання роботи на кафедру: «01» червня 2026 р.

Науковий керівник
Доктор філософії
_____Турукало А.В.
31.10.2025 Дата

Виконавець:
_____ Стаднік І.П.
31.10.2025 Дата

Календарний план

№	Етап виконання роботи	Термін виконання	Примітка
1	Затвердження теми кваліфікаційної роботи бакалавра	31.10.25	виконано
2	Аналіз проблеми недостовірної інформації, фейкових новин, дезінформації та формування мети, об’єкта, предмета і завдань дослідження	01.11.25 – 11.01.26	виконано
3	Аналіз предметної області, сучасних підходів до фактчекінгу, OSINT-перевірки та виявлення маніпулятивного новинного контенту	26.01.26 – 15.03.26	виконано
4	Дослідження методів NLP, машинного навчання та штучного інтелекту для попереднього аналізу достовірності новинного контенту	16.03.26 – 31.03.26	виконано
5	Проєктування архітектури інтерактивного програмного додатку, структури основних модулів і алгоритму перевірки новинного контенту	01.04.26 – 15.04.26	виконано
6	Розробка модулів збору, структурування, пошуку та фільтрації новин із RSS-стрічок, Telegram-джерел і відкритих вебресурсів	16.04.26 – 30.04.26	виконано
7	Реалізація AI-модуля попереднього аналізу новин, механізмів оцінювання ризику, обробки посилань і формування результатів перевірки	01.05.26 – 15.05.26	виконано
8	Проведення функціонального, нефункціонального, безпекового та регресійного тестування додатку; підготовка пояснювальної записки, графічних матеріалів і додатків	25.05.26 – 04.06.26	виконано
9	Захист кваліфікаційної роботи	18.06.26	

«Затверджую»

Науковий керівник **Індивідуальний план склав**

PhD, старший викладач, Турукало А.В. Стаднік І.П.

РЕФЕРАТ

Кваліфікаційна робота: 88 с., 16 рис., 13 табл., 44 посилань.

Актуальність роботи зумовлена потребою автоматизованої перевірки новинного контенту в умовах інформаційного перевантаження, поширення фейкових повідомлень, маніпулятивних матеріалів і дезінформаційних кампаній.

Об’єкт дослідження - процес аналізу достовірності новинного контенту в цифровому інформаційному середовищі.

Предмет дослідження - методи, алгоритми та програмні засоби автоматизованої перевірки новин із використанням фактчекінгу, OSINT, NLP і машинного навчання.

Мета роботи - розробити та обґрунтувати інтерактивний програмний додаток для збору, структурування й аналізу новинного контенту з відкритих джерел з метою виявлення потенційно недостовірної або маніпулятивної інформації.

Основні результати: уточнено теоретичні засади аналізу недостовірної інформації; сформовано вимоги до системи; спроєктовано архітектуру додатку; описано модулі RSS, Telegram та AI-перевірки; проведено тестування функціональності, продуктивності, надійності та безпеки.

Практичне значення дослідження полягає у можливості використання програмного додатку як допоміжного інструменту для попереднього аналізу новин, фільтрації підозрілих матеріалів і підготовки інформаційно-аналітичних висновків.

Ключові слова: новинний контент, фейкові новини, дезінформація, фактчекінг, OSINT, NLP, машинне навчання, Telegram, RSS, програмний додаток.

ЗМІСТ

ВСТУП

Сучасний світ переживає безпрецедентний етап розвитку інформаційного простору. Цифрові технології, соціальні мережі, онлайн-медіа та месенджери суттєво змінили спосіб створення, поширення та сприйняття інформації. Якщо ще кілька десятиліть тому новинні потоки формувалися переважно професійними редакціями та проходили багаторівневу перевірку, то сьогодні будь-який користувач мережі може стати джерелом новини. Така демократизація інформаційного простору має безперечні переваги, проте водночас створює серйозні ризики, пов’язані з поширенням недостовірного контенту.

Фейкові новини, маніпулятивні матеріали, дезінформаційні кампанії та пропагандистські наративи стали системним явищем у глобальному медіасередовищі. Недостовірна інформація поширюється швидше, ніж її спростування, а емоційно забарвлений контент часто сприймається аудиторією некритично. Особливо гостро ця проблема проявляється в умовах політичної поляризації, соціальних криз та воєнних конфліктів, коли інформація використовується як інструмент впливу на громадську думку та дестабілізації суспільства [1].

В українських реаліях питання інформаційної безпеки набуло стратегічного значення. Масовані інформаційні атаки, використання бот-мереж, маніпулятивних телеграм-каналів та псевдоновинних ресурсів створюють складний інформаційний ландшафт, у якому відрізнити правду від маніпуляції стає дедалі складніше. У таких умовах критичне мислення та здатність перевіряти джерела інформації перетворюються не лише на індивідуальну навичку, а на необхідну умову суспільної стійкості [2].

Разом із тим обсяг інформації, що генерується щоденно, настільки великий, що ручна перевірка кожного повідомлення є практично неможливою. Аналітик, журналіст або дослідник змушений працювати з сотнями новинних повідомлень із різних платформ - RSS-стрічок, новинних сайтів, соціальних мереж, Telegram-каналів. Такий масив даних потребує автоматизованих

інструментів збору, структурування та попередньої оцінки достовірності контенту.

Сучасні технології штучного інтелекту, обробки природної мови (NLP) та машинного навчання відкривають нові можливості для аналізу інформації [3]. Автоматичні алгоритми здатні визначати емоційне забарвлення тексту, виявляти маніпулятивні мовні конструкції, аналізувати повторювані наративи та зіставляти повідомлення з іншими джерелами. Однак повністю автоматизовані системи не можуть замінити експертний аналіз, а лише доповнюють його. Тому особливої ваги набуває розробка комплексних рішень, що поєднують методи OSINT, фактчекінгу та інтелектуального аналізу даних.

У цьому контексті створення інтерактивного програмного застосунку для збору й аналізу новинного контенту є актуальним та практично значущим завданням. Такий застосунок дозволяє централізувати роботу з різними джерелами інформації, забезпечити уніфікацію структури даних, виконувати первинну аналітичну оцінку та формувати звітні матеріали. Інтеграція AI-модулів дає змогу автоматизувати виявлення потенційно маніпулятивного або пропагандистського контенту, а використання архітектурних підходів, таких як MVVM, забезпечує гнучкість і масштабованість програмного рішення.

Актуальність даної роботи зумовлена необхідністю підвищення ефективності перевірки новин в умовах інформаційного перевантаження. Розробка програмного продукту, здатного поєднати збір даних із відкритих джерел, їх структуризацію та інтелектуальний аналіз, сприятиме підвищенню якості інформаційного середовища та підтримці прийняття обґрунтованих рішень.

Метою роботи є розробка та обґрунтування програмного застосунку для автоматизованого збору інформації з відкритих джерел (RSS-стрічок, Telegram-каналів), її структурування та аналізу з використанням методів OSINT і технологій штучного інтелекту з метою виявлення потенційно недостовірного або маніпулятивного новинного контенту.

Об’єктом дослідження є процес аналізу достовірності новинного контенту в цифровому інформаційному середовищі.

Предметом дослідження є методи, алгоритми та програмні засоби автоматизованої перевірки новин, зокрема:

• методи фактчекінгу та OSINT;

• технології обробки природної мови (NLP);

• алгоритми машинного навчання для виявлення фейкових новин;

• архітектурні рішення для побудови інтерактивного програмного додатку.

Для досягнення поставленої мети в роботі необхідно вирішити такі основні завдання:

1. Проаналізувати теоретичні основи поняття недостовірної інформації, визначити її види та особливості функціонування у сучасному медіапросторі.

2. Дослідити психологічні та соціальні механізми поширення дезінформації, зокрема роль когнітивних упереджень, алгоритмічної персоналізації та групової динаміки.

3. Розглянути сучасні методи перевірки достовірності новинного контенту, включаючи фактчекінг, OSINT-підходи та аналіз відкритих джерел інформації.

4. Проаналізувати можливості застосування технологій обробки природної мови (NLP) та методів машинного навчання для виявлення маніпулятивних мовних конструкцій і фейкових повідомлень.

5. Сформувати функціональні та нефункціональні вимоги до програмної системи, що забезпечує автоматизований збір та аналіз новин.

6. Спроєктувати архітектуру програмного застосунку та обґрунтувати вибір архітектурного підходу MVVM для його реалізації.

Методи дослідження. У кваліфікаційній роботі використано методи аналізу, порівняння, систематизації, моделювання та тестування. Аналіз застосовано для вивчення недостовірної інформації, фейкових новин, фактчекінгу, OSINT, NLP і машинного навчання. Порівняння дало змогу

зіставити сучасні підходи до перевірки новинного контенту, а систематизація - впорядкувати типи недостовірної інформації та критерії її оцінювання. Моделювання використано під час проєктування структури додатку й алгоритму аналізу новин. Працездатність розробленого рішення перевірено за допомогою функціонального, нефункціонального, безпекового та регресійного тестування.

Практичне значення одержаних результатів. Практичне значення роботи полягає у розробці інтерактивного програмного додатку для попереднього аналізу достовірності новинного контенту. Застосунок забезпечує збір матеріалів із відкритих джерел, зокрема RSS-стрічок і Telegram-каналів, їх структурування та виявлення потенційно недостовірних або маніпулятивних повідомлень. Розроблене рішення може використовуватися журналістами, аналітиками, дослідниками, студентами та викладачами як допоміжний інструмент для швидкого опрацювання новин і підготовки до подальшого фактчекінгу.

Апробація результатів роботи. Результати дослідження апробовано шляхом участі у підготовці та публікації наукової праці, пов’язаної з використанням технологій OSINT для аналізу інформації з відкритих джерел. Матеріали опубліковано у праці: Білоус В. В., Бодненко Д. М., Хохлов О. К., Локазюк О. В., Стаднік І. П. *Open Source Intelligence for War Crime Documentation*. Workshop Cybersecurity Providing in Information and Telecommunication Systems (CPITS 2024). CEUR Workshop Proceedings. 2024. Vol. 3654. P. 368–375. ISSN 1613-0073. [35]

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ АНАЛІЗУ ДОСТОВІРНОСТІ НОВИННОГО КОНТЕНТУ

1.1 Поняття та класифікація недостовірної інформації

Недостовірна інформація - це відомості, що не відповідають дійсності, є неправдивими, неповними або перекрученими, і можуть стосуватися фактів чи подій, яких не було, або ж спотворюють реальні, причому це не просто оціночні судження (як-от "лікар байдужий"), а фактичні твердження про реальний стан речей (як "лікар купив диплом"). Класифікація включає дезінформацію (навмисне спотворення), неправдиві відомості (відверта брехня) та неповні/перекручені дані, а відповідальність варіюється від спростування до кримінальної/адміністративної, особливо у сфері декларування [1, 4].

Поняття недостовірної інформації

- Відсутність відповідності дійсності: Інформація, що відображає події чи факти, які не відбувалися або існують в іншому вигляді, ніж подається.
- Неправдивість: Свідоме викривлення фактів з метою введення в оману.
- Неповнота/Перекручення: Подання інформації, яка є частковою або зміненою, щоб створити хибне враження.
- Факт проти оцінки: Важливо розрізняти фактичні твердження (наприклад, "він має судимість"), які можуть бути недостовірними, від суб'єктивних думок ("він поганий лікар").
- Законодавча база: Конституція України гарантує право спростовувати недостовірну інформацію про себе (ст. 32).

Перш ніж перейти до детального аналізу, необхідно окреслити ключові поняття, що стосуються недостовірної інформації. Нижче наведено основні визначення та їх характеристики:

Місінформація (misinformation): Це інформація, що є хибною, неточною або некоректною, незалежно від наміру її поширення. Місінформація може виникати як наслідок помилок, непорозумінь або ненавмисного викривлення фактів [6].

Дезінформація (disinformation): Це свідомо створена та поширена неправильна інформація зі спеціальною метою маніпулювання аудиторією або

зміни суспільної думки. Дезінформація часто використовується з метою дискредитації опонентів чи впливу на політичні процеси.

Фейкові новини (fake news): Даний термін використовується для опису інформації, яка виглядає як новина, але є навмисно створеною та перевіреною як хибна. Фейкові новини часто містять сенсаційні заголовки, що спрямовані на максимальне залучення уваги та емоційний вплив.

Маніпуляції: Під маніпуляціями розуміють навмисне викривлення або спотворення інформації з метою впливу на погляди, переконання та поведінку людей.

Ця таблиця ілюструє основні відмінності між ключовими поняттями та допомагає окреслити рамки подальшого дослідження (таб.1.1).

Таблиця 1.1

Порівняльний аналіз понять недостовірної інформації

Категорія	**Визначення**	**Намір поширення**	**Основні характеристики**
Місінформація	Хибна, неточна, некоректна інформація	Ненавмисна або помилкова	Виникає через помилки або непорозуміння
Дезінформація	Свідомо створена та поширена неправильна інформація	Намір маніпулювати аудиторією	Створюється з метою впливу на суспільну думку, дискредитації опонентів
Фейкові новини	Інформація, представлену як новини, яка є навмисно хибною	Намір залучити увагу, вплинути на думки	Сенсаційність, перевіреність інформації як неправдива, емоційний вплив
Маніпуляції	Навмисне викривлення або спотворення інформації для впливу на переконання та поведінку	Намір керувати сприйняттям аудиторії	Використання різних маніпулятивних стратегій для досягнення своїх цілей

Класифікація типів недостовірного контенту

Недостовірний контент може набувати різного вигляду в залежності від його призначення, тематичного спрямування та методів поширення. На основі аналізу сучасних досліджень, зокрема згідно з результатами роботи Adams та колег та інших джерел, виокремлено наступні основні типи (рис.1) [6]:

• Політичний контент

Політичний недостовірний контент являє собою різновид інформаційного впливу, спрямованого на зміну політичних установок аудиторії, маніпулювання громадською думкою та втручання у демократичні процеси, зокрема вибори та референдуми. Такий контент може поширюватися у формі неправдивих повідомлень, фальсифікованих фактів або скоординованих інформаційних кампаній, метою яких є дискредитація політичних опонентів, створення суспільної напруги або формування викривленого сприйняття політичної реальності.

• Економічний контент

Економічний недостовірний контент охоплює неправдиву або маніпулятивну інформацію, здатну впливати на фінансові процеси, ділову репутацію, поведінку споживачів та ринкову стабільність. Його поширення може спричиняти панічні настрої, коливання цінових індикаторів, втрату довіри до компаній або спотворення ринкових очікувань. Особливістю такого контенту є його потенційний прямий вплив не лише на інформаційний простір, а й на економічні процеси в реальному середовищі.

• Псевдонауковий контент

Псевдонауковий контент базується на використанні викривлених, неперевірених або навмисно спотворених відомостей, які імітують наукову аргументацію, проте не відповідають принципам наукової достовірності. Його метою часто є формування сумнівів щодо наукового консенсусу, поширення хибних переконань або маніпулювання суспільним сприйняттям складних наукових питань. Особливо небезпечним такий контент є у сферах охорони здоров’я, екології, біотехнологій та інших критично важливих галузях.

• Емоційний контент

Емоційно забарвлений контент є формою інформаційного впливу, побудованою на активації сильних емоційних реакцій аудиторії, зокрема страху, обурення, тривоги або ейфорії. Його особливістю є використання сенсаційності, експресивної риторики та повторюваності повідомлень з метою посилення

віральності та підвищення рівня довіри до змісту незалежно від його достовірності. Такий тип контенту часто використовується як інструмент посилення ефективності дезінформаційних кампаній.

Рисунок 1.1 показує поділ недостовірного контенту на чотири основні категорії за тематичним спрямуванням.

Рисунок 1.1. Класифікація типів недостовірного контенту

Психологічні механізми впливу дезінформації [7]

Психологічний вплив недостовірної інформації є надзвичайно складним і багатовимірним явищем. Дослідження демонструють, що людський мозок обмежений у своїй здатності обробляти інформацію, що створює сприятливі умови для впливу різних когнітивних упереджень.

• Обмежена обробка інформації та ментальні скорочення

Людина не здатна приділяти глибоку увагу кожному новому повідомленню через обмежену когнітивну потужність. Це спричиняє використання множинних ментальних скорочень (heuristics), завдяки яким інформація обробляється поверхнево. Як результат, дезінформація часто сприймається як правдива через знайомість або просте повторення.

• Когнітивний дисонанс та підтверджувальне упередження

Коли людина зустрічається з новою інформацією, що суперечить її попереднім переконанням, виникає когнітивний дисонанс. Щоб зменшити цей дисонанс, індивід може ігнорувати або спотворювати суперечливі дані, що ще більше укріплює хибні переконання. Проблема ускладнюється явищем "worldview backfire effect", коли спростування інформації підсилює первинне переконання.

• Вплив групового членства та соціальної ідентичності

Дослідження показали, що сприйняття інформації сильно залежить від групової приналежності. Інформація, отримана від представників окремої соціальної групи, сприймається як більш достовірна та легкоприймається, що сприяє поширенню дезінформації всередині групи. Цей ефект підсилюється бажанням зберегти соціальну ідентичність і підтримувати згуртованість групи.

• Ефект повторення та ефект ілюзорної істини

Повторення дезінформації має вирішальне значення для її сприйняття. Ефект ілюзорної істини полягає в тому, що повторювана інформація через свою знайомість здається правдивішою. Навіть ефективні спростування можуть не усунути цей ефект, що пояснює явище "continued influence effect", коли хибна інформація продовжує впливати на мислення навіть після спростування.

Діаграма ілюструє послідовність психологічного процесу, у якому дезінформація впроваджується та укріплюється в свідомості отримувача (рис.1.2).

Рисунок 1.2. Психологічний процес впливу дезінформації

Соціальні механізми поширення недостовірної інформації [37 - 40]

Соціальні мережі та інші платформи масштабного обміну інформацією стали головними ареною для передачі та поширення недостовірного контенту. Механізми, що визначають цей процес, є як технологічними, так і соціальними (таб.2) [36].

• Роль соціальних мереж

Сучасні онлайн-сервіси, такі як Facebook, Twitter та інші, забезпечують миттєву комунікацію великими обсягами даних. Система алгоритмічного підбору контенту, заснована на історії поведінки користувачів, може сприяти ехо-камері, де люди отримують підтвердження своїх переконань з обмеженого

кола джерел. Це створює умови для посилення групового ефекту та зростання впливу недостовірних новин.

• Групова динаміка та соціальний доказ

Роль групової динаміки у поширенні інформації дуже вагома. Коли велика кількість користувачів ділиться певним повідомленням, це створює соціальний доказ достовірності даних. Люди, спостерігаючи за високою активністю (лайки, репости, коментарі), схильні приймати інформацію за правдиву, навіть якщо вона є хибною. Таким чином, соціальна мережа стає каталізатором швидкого розповсюдження дезінформації.

• Технологічні маніпуляції та бот-мережі [8]

Сучасні інформаційні системи дозволяють не тільки спонтанне поширення повідомлень, а й штучне їх посилення за рахунок бот-мереж та автоматизованих систем. Ці механізми можуть штучно підвищувати рівень залученості певних повідомлень, створюючи видимість їх популярності навіть за умов низького реального інтересу. Також використання технологій цифрового підроблення, наприклад, deepfake, ускладнює визначення достовірності контенту, що показано в таблиці 1.2

Таблиця 1.2

Соціальні механізми та їх вплив на поширення інформації

Механізм	**Опис**	**Вплив на аудиторію**
Алгоритмічна персоналізація	Підбір контенту відповідно до вподобань користувача	Збільшує ехо-ефект, посилює підтвердження власних переконань
Соціальний доказ	Велика кількість взаємодій (лайків, репостів) підсилює сприйняття правдивості контенту	Сприяє формуванню думки “як всі, так і я”
Бот-мережі та штучне посилення	Використання автоматизованих облікових записів для розповсюдження повідомлень	Створює ілюзію високої популярності інформації
Цифрові маніпуляції	Використання deepfake та інших технологій цифрової підробки	Знижує довіру до традиційних джерел, ускладнює перевірку фактів

1.2 Методи перевірки достовірності новин

У сучасному інформаційному просторі достовірність новин набуває вирішального значення, оскільки інформація більше не виконує виключно повідомлювальну функцію, а перетворюється на інструмент впливу на громадську думку, політичні процеси й соціальну поведінку. Розвиток цифрових технологій, соціальних мереж та онлайн-платформ призвів до того, що поширення новин відбувається практично миттєво, а значна частина контенту створюється без професійної редакторської перевірки. За цих умов фейкові новини, маніпулятивні повідомлення та дезінформаційні наративи стали системним явищем, здатним суттєво впливати на суспільні настрої та процеси прийняття рішень.

Особливо актуальною проблема перевірки достовірності новин стає в умовах гібридних конфліктів, політичної поляризації та глобальних криз. Дезінформація використовується як інструмент інформаційної війни, спрямованої на дестабілізацію суспільства, підрив довіри до офіційних інституцій та формування хибних уявлень про реальність. У таких умовах потреба у науково обґрунтованих методах верифікації інформації є необхідною умовою інформаційної безпеки суспільства.

Перевірка достовірності новин є багаторівневим процесом, який поєднує журналістські, аналітичні й технологічні підходи. Серед основних методів особливе місце посідає фактчекінг, який полягає в систематичній перевірці фактів, тверджень і даних, що містяться в інформаційних повідомленнях. Фактчекінг спрямований на встановлення відповідності інформації реальним подіям, офіційним документам, статистичним даним та іншим перевіреним джерелам (рис.1.3).

У науковій і журналістській практиці фактчекінг розглядається як інструмент протидії дезінформації, який забезпечує підвищення якості інформаційного простору. На відміну від звичайної перевірки новин у редакціях, фактчекінг характеризується підвищеною методологічною прозорістю, чітким описом етапів перевірки та публічним обґрунтуванням висновків. Саме це

робить фактчекінг важливим елементом формування довіри між медіа та аудиторією.

Рисунок 1.3. Протидія дезінформації

Ручний фактчекінг є традиційною і найбільш надійною формою верифікації інформації, оскільки він передбачає активну залученість людини-аналітика. На цьому етапі ключову роль відіграють навички критичного мислення, уміння працювати з джерелами та здатність оцінювати контекст поширення інформації. Процес ручного фактчекінгу починається з виявлення перевірюваних тверджень у тексті новини. Не всі елементи повідомлення є фактами, тому важливо відокремлювати оціночні судження, емоційні формулювання та припущення від конкретних даних, які можуть бути перевірені.

Наступним етапом є пошук першоджерел інформації. Першоджерелом може бути офіційна заява, статистичний звіт, нормативний документ, пряма відеофіксація події або публікація на офіційному вебресурсі. Саме на встановленні першоджерела найчастіше ґрунтується успіх фактчекінгу, оскільки вторинні публікації можуть містити спотворення, неточності або навмисні маніпуляції.

Важливою складовою ручного фактчекінгу є перехресна перевірка інформації шляхом зіставлення даних з кількома незалежними джерелами. Якщо факт підтверджується різними джерелами, що не пов’язані між собою, рівень

його достовірності значно зростає. Особливо цінними вважаються дані міжнародних організацій, державних статистичних служб, наукових інституцій та авторитетних медіа.

Окрему увагу ручний фактчекінг приділяє аналізу контексту. Маніпулятивні повідомлення часто використовують реальні факти, але формують хибне уявлення завдяки вириванню їх із ширшого контексту. Наприклад, економічні показники можуть подаватися без урахування інфляції, зміни курсу національної валюти або сезонних коливань, що призводить до викривленого сприйняття ситуації.

Результатом ручного фактчекінгу є формулювання чіткого висновку про характер інформації. У практиці фактчекінгових організацій часто використовуються категорії на кшталт "правда", "часткова правда", "маніпуляція", "фейк". Такий підхід дозволяє не обмежуватися дихотомією "правда - неправда", а показувати ступінь спотворення інформації.

Поряд з ручним фактчекінгом у сучасному інформаційному просторі широко застосовуються автоматизовані підходи до перевірки інформації. Вони базуються на використанні алгоритмів машинного навчання та обробки природної мови, які здатні аналізувати великі масиви текстових даних, виявляти повторювані твердження та зіставляти їх з наявними записами в базах даних.

Автоматизований фактчекінг має низку переваг, серед яких варто відзначити високу швидкість обробки інформації та можливість роботи в режимі реального часу. Це особливо актуально для соціальних мереж, де дезінформація може поширюватися надзвичайно швидко. Водночас автоматизовані системи мають суттєві обмеження, оскільки вони часто не здатні коректно розпізнавати іронію, сарказм, приховані підтексти та складні контекстуальні зв’язки [7].

У зв’язку з цим у сучасній практиці перевірки достовірності новин переважає комбінований підхід, який поєднує переваги ручного й автоматизованого фактчекінгу. Автоматизовані системи виконують функцію попереднього фільтру та ідентифікації потенційно сумнівного контенту, після чого людина-аналітик здійснює глибший контекстуальний аналіз.

Важливим доповненням до фактчекінгу є використання OSINT - розвідки на відкритих джерелах. OSINT передбачає збір, аналіз та інтерпретацію інформації з публічно доступних ресурсів, таких як вебсайти, соціальні мережі, форуми, супутникові знімки та відкриті реєстри. У контексті верифікації новин OSINT дозволяє не обмежуватися аналізом тексту, а розглядати ширший інформаційний ландшафт.

Одним з найбільш поширених OSINT-методів є геолокаційний аналіз, який полягає у визначенні місця події на основі візуальних ознак, відображених на зображеннях або відео. Для цього використовуються супутникові карти, панорами вулиць, архітектурні деталі та природні орієнтири. Геолокація особливо ефективна при перевірці відеоматеріалів з зон конфліктів та надзвичайних подій [9].

Іншим важливим OSINT-інструментом є хронологічний аналіз, який дозволяє встановити час створення контенту або зображеної події. Аналіз тіней, погодних умов, метаданих файлів та публікацій у соціальних мережах дає змогу виявляти випадки, коли старі відео або фотографії використовуються для маніпуляції громадською думкою шляхом представлення їх як нових.

Важливою складовою OSINT-аналізу є дослідження соціальних мереж як основного середовища поширення сучасних новин. Соціальні платформи створюють сприятливі умови для швидкої циркуляції інформації, однак водночас значно ускладнюють процес її перевірки. Новини у соціальних мережах часто поширюються без посилань на першоджерела, у вигляді коротких повідомлень або емоційно забарвлених візуальних матеріалів. Саме тому аналіз соціально-мережевої активності є важливим інструментом перевірки достовірності інформації.

OSINT-аналіз соціальних мереж передбачає вивчення акаунтів, які першими опублікували або активно поширюють новину. До уваги береться дата створення профілю, регулярність публікацій, тематична спрямованість контенту, а також наявність ознак автоматизованої або координованої поведінки. Акаунти, створені нещодавно та орієнтовані виключно на поширення резонансного або

політично забарвленого контенту, часто є індикатором інформаційних кампаній або маніпуляцій [21, 22].

Окрему роль відіграє аналіз мережі поширення новини. Дослідження того, які ресурси, сторінки або групи одночасно публікують ідентичний або дуже схожий контент, дозволяє виявляти скоординовані інформаційні операції. Такий підхід широко використовується у журналістських розслідуваннях та дослідженнях інформаційного впливу, особливо в контексті політичних процесів і воєнних конфліктів.

Ще одним важливим напрямом OSINT є аналіз цифрових слідів джерел інформації. Він включає перевірку доменних імен сайтів, на яких публікуються новини, даних про їх реєстрацію, історії змін контенту та технічних параметрів. Використання відкритих сервісів для перевірки доменів дозволяє встановити дату створення сайту, країну реєстрації та інші ознаки, що можуть свідчити про надійність або сумнівність ресурсу. Сайти, створені нещодавно та орієнтовані на масове поширення сенсаційних новин, часто використовуються як платформи для дезінформації.

Не менш важливим є аналіз візуального контенту - фотографій і відео, які супроводжують новинні повідомлення. Візуальні матеріали мають потужний емоційний вплив і часто використовуються для підсилення маніпулятивного ефекту. Одним із базових методів перевірки зображень є зворотний пошук, який дозволяє встановити, чи використовувалося дане зображення раніше та в якому контексті. Нерідко виявляється, що фотографії, подані як ілюстрація актуальної події, насправді були зроблені кілька років тому або в іншій країні.

Аналіз метаданих зображень і відео також є важливим елементом верифікації. Метадані можуть містити інформацію про дату створення файлу, тип пристрою, а іноді й геолокацію. Хоча метадані легко піддаються зміні або видаленню, у поєднанні з іншими методами вони дозволяють отримати додаткові підтвердження або спростування інформації.

Окрему увагу слід приділяти аналізу відеоконтенту, оскільки сучасні технології дозволяють створювати високоякісні маніпуляції, включаючи так

звані deepfake-відео. Перевірка відео включає аналіз візуальних артефактів, невідповідностей у звуці та зображенні, а також порівняння з іншими джерелами відеоматеріалів з тієї ж події.

Важливим аспектом перевірки достовірності новин є аналіз посилань, що містяться в публікаціях. Посилання можуть вести на першоджерела, офіційні документи або, навпаки, на сумнівні ресурси, створені для імітації надійних джерел. Перевірка URL-адрес, структури сайту та наявності ознак фішингу дозволяє знизити ризик довіри до неправдивої інформації. Особливої уваги потребують посилання на ресурси, які візуально копіюють дизайн відомих медіа або державних установ (рис. 1.4).

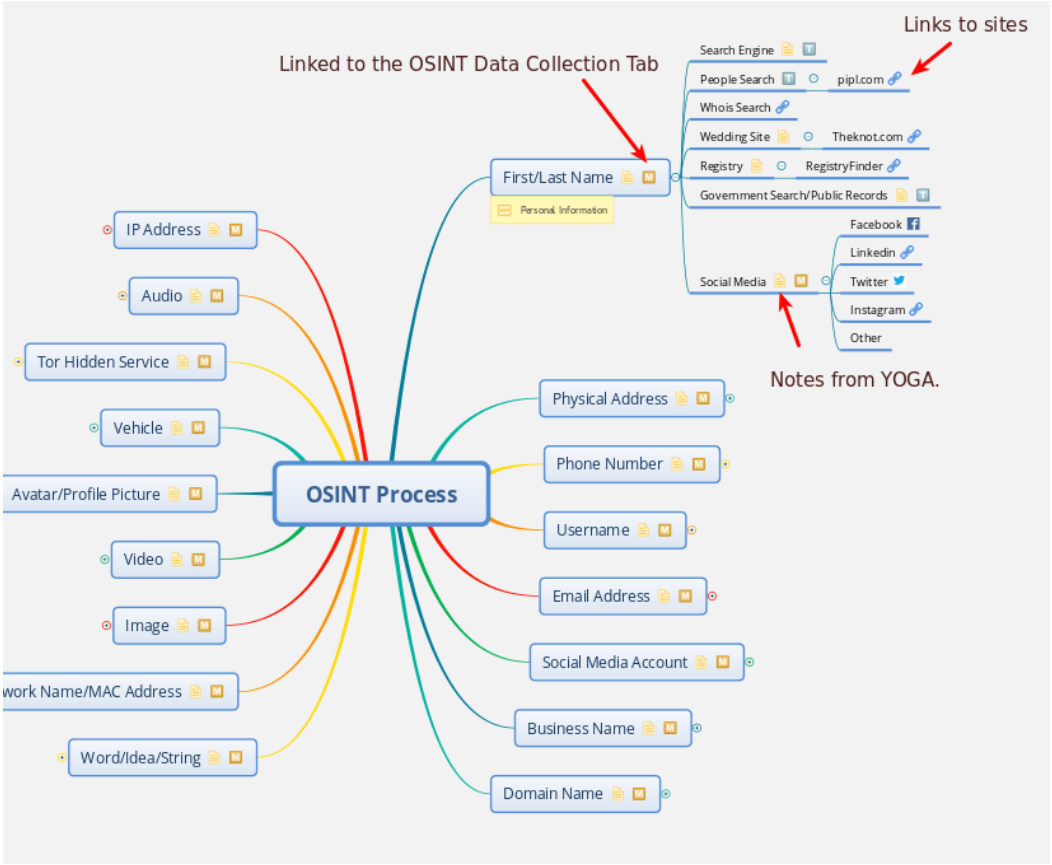

Рисунок 1.4. Процес перевірки

Не менш значущим є аналіз дати та часу публікації новини. Старі матеріали часто повторно поширюються як актуальні, що створює хибне уявлення про події. Така практика особливо поширена під час кризових ситуацій, коли аудиторія схильна до емоційного реагування. Перевірка дати першої публікації,

а також порівняння з іншими повідомленнями на цю тему дозволяє виявляти подібні маніпуляції.

У процесі перевірки достовірності новин важливу роль відіграє також аналіз мови та стилістики повідомлення. Маніпулятивні матеріали часто характеризуються надмірною емоційністю, використанням оціночних суджень, узагальнень та категоричних тверджень без належних доказів. Лінгвістичний аналіз дозволяє виявляти такі ознаки та оцінювати ймовірність маніпулятивного характеру контенту.

Комплексне застосування фактчекінгу, OSINT та аналізу джерел інформації дозволяє значно підвищити ефективність перевірки новин (рис.1.5). Жоден з методів не є універсальним, проте їх поєднання створює багаторівневу систему захисту від дезінформації. Такий підхід є особливо важливим в умовах сучасних інформаційних викликів, коли швидкість поширення новин часто перевищує швидкість їх осмислення.

Рисунок 1.5. Методи перевірки

Узагальнюючи, можна зазначити, що перевірка достовірності новин є не лише технічною або журналістською процедурою, а й елементом формування критичного мислення суспільства. Розвиток навичок аналізу інформації, уміння

працювати з відкритими джерелами та розуміння механізмів маніпуляції є необхідними складовими інформаційної грамотності в сучасному світі.

1.3 Сучасні технології аналізу контенту

Стрімке зростання обсягів цифрової інформації, що циркулює у сучасному медіапросторі, зумовило необхідність переходу від виключно ручних методів перевірки достовірності новин до комплексних технологічних рішень. За даними міжнародних досліджень, щоденно у глобальних соціальних мережах публікуються мільйони новинних повідомлень, значна частина яких не проходить жодної редакторської перевірки. У таких умовах традиційні підходи фактчекінгу не здатні забезпечити достатню швидкість реагування на поширення дезінформації.

Сучасні технології аналізу контенту базуються на використанні методів штучного інтелекту, зокрема обробки природної мови (Natural Language Processing, NLP) [23], машинного навчання та аналізу великих даних. Їх головною метою є автоматизоване виявлення потенційно недостовірних матеріалів, маніпулятивних наративів та інформаційних кампаній. Важливо підкреслити, що такі технології не замінюють експертний аналіз, а виконують функцію попереднього відбору та аналітичної підтримки.

Застосування автоматизованих технологій аналізу контенту є особливо актуальним у контексті гібридних інформаційних війн, політичної пропаганди та масового використання соціальних мереж як каналу впливу на громадську думку. Саме тому провідні міжнародні організації, технологічні компанії та фактчекінгові платформи інвестують значні ресурси у розробку інструментів автоматизованої перевірки інформації.

Обробка природної мови є одним із ключових напрямів сучасного штучного інтелекту, що забезпечує можливість аналізу текстової інформації на рівні, наближеному до людського сприйняття. У контексті перевірки достовірності новин NLP використовується для аналізу лексичних, синтаксичних та семантичних характеристик тексту [11].

Основні завдання NLP у сфері аналізу новин включають:

- автоматичне розпізнавання тематики повідомлення;
- визначення тональності та емоційного забарвлення тексту;
- виявлення маніпулятивних мовних конструкцій;
- розпізнавання іменованих сутностей (імен людей, організацій, географічних об’єктів);
- аналіз логічної структури тексту.

Застосування NLP дозволяє автоматично аналізувати тисячі текстів за короткий проміжок часу та виявляти матеріали, які мають підвищений ризик недостовірності.

Одним з найбільш поширених і водночас ефективних методів обробки природної мови у перевірці новин є аналіз тональності (sentiment analysis). Цей підхід ґрунтується на автоматичному визначенні емоційного забарвлення тексту та його впливу на аудиторію. У контексті виявлення дезінформації аналіз тональності має особливе значення, оскільки фейкові та маніпулятивні матеріали, як правило, апелюють не до раціонального осмислення фактів, а до емоційної реакції читача [8].

Тональність тексту зазвичай класифікується як позитивна, негативна або нейтральна, проте сучасні NLP-моделі здатні здійснювати значно глибший аналіз, виокремлюючи конкретні емоції: страх, гнів, тривогу, обурення, ейфорію, сенсаційність тощо. Саме ці емоції найчастіше використовуються у дезінформаційних кампаніях, оскільки вони знижують здатність людини до критичного мислення та підвищують ймовірність некритичного поширення інформації.

Наукові дослідження у сфері медіапсихології та обробки природної мови підтверджують зв’язок між емоційною насиченістю тексту та його достовірністю. Зокрема, роботи Google Research, MIT Media Lab та Університету Індіани показують, що неправдиві або маніпулятивні новини статистично частіше містять емоційно насичену лексику, ніж перевірені інформаційні повідомлення. Такі матеріали зазвичай використовують драматичні заголовки, перебільшення, емоційні маркери та категоричні формулювання.

У практичному застосуванні NLP аналіз тональності використовується як один з індикаторів ризику. Наприклад, новинні повідомлення з високим рівнем негативної або сенсаційної тональності автоматично позначаються системами як такі, що потребують додаткової перевірки. Важливо підкреслити, що сама по собі емоційність тексту не є прямим доказом його недостовірності, проте у поєднанні з іншими ознаками вона значно підвищує ймовірність маніпулятивного характеру матеріалу.

Сучасні NLP-моделі аналізу тональності використовують різні підходи. Класичні методи ґрунтуються на словниках емоційної лексики, де кожному слову надається певна емоційна вага. Більш складні підходи застосовують нейронні мережі та трансформерні моделі, які здатні враховувати контекст використання слів. Це особливо важливо, оскільки одне й те саме слово може мати різне емоційне значення залежно від контексту.

Використання трансформерних моделей, таких як BERT або RoBERTa [9], дозволяє аналізувати емоційну структуру тексту на рівні речень і абзаців, а не окремих слів. Це значно підвищує точність аналізу, особливо у випадках прихованої маніпуляції, коли емоційний вплив досягається не прямими оцінками, а через композицію тексту, риторичні прийоми та логічні акценти.

Окрему увагу в аналізі тональності приділяють заголовкам новин. Саме заголовок часто є ключовим елементом емоційного впливу, оскільки більшість користувачів соціальних мереж сприймають інформацію поверхнево, не читаючи повний текст. Дослідження показують, що фейкові новини значно частіше використовують емоційно забарвлені або провокаційні заголовки, які викликають негайну реакцію.

Окрім аналізу тональності, важливим напрямом застосування NLP у перевірці новин є виявлення характерних мовних патернів, притаманних дезінформації. Маніпулятивні тексти мають низку спільних лінгвістичних ознак, які можуть бути формалізовані та автоматично розпізнані.

Одним з найпоширеніших патернів є використання узагальнень без посилання на конкретні джерела. Формулювання на кшталт "усі знають",

"експерти стверджують", "вчені довели" створюють ілюзію авторитетності, не надаючи жодних перевірюваних доказів. NLP-моделі здатні виявляти такі конструкції шляхом аналізу синтаксичних шаблонів та відсутності конкретних іменованих сутностей або посилань.

Іншою характерною ознакою є категоричні твердження без аргументації. Маніпулятивні тексти часто подають складні або спірні питання у вигляді однозначних і безапеляційних висновків. Це знижує схильність аудиторії до критичного аналізу та сприяє прийняттю нав’язаного наративу. Автоматичні системи аналізують частоту використання категоричних модальних конструкцій та абсолютних формулювань.

Надмірне використання оціночних прикметників також є типовою рисою дезінформаційних матеріалів. Замість нейтрального опису фактів застосовуються слова з яскраво вираженим емоційним забарвленням, які формують у читача певне ставлення до події або суб’єкта новини. NLP-алгоритми здатні кількісно оцінювати співвідношення нейтральної та оціночної лексики у тексті.

Окрему групу маніпулятивних патернів становить використання псевдонаукової лексики. Такі тексти часто імітують науковий стиль, використовуючи складні терміни без чіткого визначення або у некоректному контексті. Це створює ілюзію наукової обґрунтованості та підвищує довіру аудиторії. NLP-моделі можуть виявляти такі випадки шляхом аналізу невідповідності між термінологією та контекстом її використання.

Ще однією поширеною формою маніпуляції є логічні підміни та хибні причинно-наслідкові зв’язки. У таких текстах події, які збігаються у часі, подаються як такі, що перебувають у причинному зв’язку. Автоматичне виявлення подібних помилок є складним завданням, однак сучасні NLP-системи здатні фіксувати нетипові логічні переходи та суперечності у структурі тексту.

Автоматичне виявлення маніпулятивних мовних патернів не має на меті остаточне визначення достовірності новини. Його основна функція полягає у класифікації текстів за рівнем потенційної маніпулятивності та формуванні

пріоритетів для подальшої експертної перевірки. Поєднання аналізу тональності та виявлення мовних патернів створює ефективний інструмент первинного фільтрування інформаційного контенту.

Таким чином, використання NLP у аналізі тональності та мовних патернів дозволяє систематизувати процес виявлення маніпулятивних матеріалів і значно підвищити ефективність сучасних систем перевірки достовірності новин. У поєднанні з машинним навчанням та експертним аналізом ці методи формують основу комплексного підходу до протидії дезінформації в цифровому середовищі.

1.4 Методи машинного навчання у виявленні фейкових новин

Машинне навчання (Machine Learning) використовується для побудови моделей, які навчаються на великих наборах даних і здатні автоматично класифікувати новини за рівнем достовірності. Для цього застосовуються спеціально підготовлені датасети, що містять приклади перевірених та фейкових матеріалів.

Процес навчання ML-моделі зазвичай включає:

- Збір і маркування навчальних даних.
- Виділення текстових і контекстуальних ознак.
- Навчання моделі на основі обраного алгоритму.
- Оцінку точності та корекцію параметрів.

Основні типи моделей машинного навчаннянаведені в таблиці 1.3

Таблиця 1.3

Моделі машинного навчання у виявленні фейкових новин

Тип моделі	Характеристика	Переваги	Обмеження
Логістична регресія	Базовий класифікатор	Швидкість, простота	Низька точність для складних текстів
SVM	Аналіз складних ознак	Висока точність	Високі обчислювальні витрати
Random Forest	Ансамблевий метод	Стійкість до шуму	Обмежена інтерпретованість

Нейронні мережі	Глибокий аналіз	Контекстуальність	Потреба у великих даних
Transformer-моделі	Контекстне розуміння	Найвища точність	Складність реалізації

Сучасні системи дедалі частіше використовують transformer-архітектури (BERT, RoBERTa), оскільки вони здатні враховувати контекст слів у реченні, а не лише їхню частотність.

Попри значний потенціал методів машинного навчання у виявленні та аналізі недостовірного контенту, їх застосування супроводжується низкою суттєвих обмежень, які необхідно враховувати під час розроблення та впровадження відповідних систем. Насамперед, ефективність моделей машинного навчання безпосередньо залежить від якості, повноти та репрезентативності навчальних даних. Якщо набір даних містить помилки, упередження, недостатнє тематичне покриття або не відображає реальні сценарії поширення дезінформації, модель може формувати неточні закономірності та демонструвати хибні результати. Іншими словами, принцип garbage in - garbage out у цій сфері працює безжально: слабкі дані породжують слабкі рішення.

Суттєвим викликом є також проблема збалансованості навчальних вибірок. На практиці достовірний контент часто значно переважає недостовірний, що створює дисбаланс класів і може призводити до зниження точності класифікації, особливо щодо виявлення рідкісних або нових форм дезінформації. Окрім цього, процес маркування даних часто потребує участі експертів, а сама розмітка може бути суб’єктивною, особливо в ситуаціях, коли межа між маніпуляцією, пропагандою, сатирою та свідомою дезінформацією є нечіткою.

Другим важливим обмеженням є недостатня інтерпретованість багатьох сучасних моделей, особливо тих, що базуються на глибокому навчанні. Значна частина таких алгоритмів функціонує за принципом так званої "чорної скриньки", коли система генерує результат, але логіка прийняття рішення залишається непрозорою для користувача або дослідника. Це створює труднощі при поясненні причин класифікації певного повідомлення як фейкового чи достовірного, а також ускладнює аудит, верифікацію та виявлення помилок у

роботі моделі. Для систем, що застосовуються у сферах медіамоніторингу, кібербезпеки або інформаційної протидії, проблема пояснюваності є не лише технічним, а й етичним та правовим питанням.

Окремою проблемою є зниження ефективності моделей у разі зміни мовного, культурного або соціального контексту. Модель, натренована на англомовних джерелах або на матеріалах певного регіону, може демонструвати значно нижчу точність при аналізі контенту іншою мовою, в іншому інформаційному середовищі або в умовах специфічних локальних наративів. Це пов’язано з тим, що дезінформація часто використовує культурні коди, політичні контексти, мовні особливості, гумор, алюзії та специфічні форми маніпуляцій, які складно формалізувати універсально. Те, що є очевидним маркером маніпуляції в одному середовищі, може бути нейтральним повідомленням в іншому.

Ще одним суттєвим обмеженням є проблема адаптації моделей до динамічної природи дезінформації. Інформаційні атаки постійно еволюціонують: змінюються риторичні прийоми, формати подачі, канали поширення та тактики обходу автоматизованих систем виявлення. Унаслідок цього модель, яка демонструвала високу точність на момент навчання, з часом може втрачати ефективність через явище концептуального дрейфу (concept drift), коли закономірності у вхідних даних змінюються швидше, ніж система встигає адаптуватися.

Крім того, машинні моделі є вразливими до навмисного протидіючого впливу. Йдеться про так звані adversarial-підходи, коли зловмисники спеціально модифікують тексти, змінюють формулювання, додають шум, стилістичні варіації або використовують приховані маніпулятивні конструкції, щоб обійти алгоритми детекції. Це створює постійне змагання між системами виявлення та механізмами обходу, у якому статичні моделі часто програють без регулярного оновлення.

Розвиток цифрового інформаційного середовища та стрімке зростання обсягів контенту, що поширюється через соціальні мережі, новинні платформи

та онлайн-медіа, зумовили потребу у створенні спеціалізованих систем автоматизованого аналізу контенту. Особливої актуальності такі системи набули в умовах поширення дезінформації, маніпулятивних повідомлень, фейкових новин та інших форм інформаційного впливу, виявлення яких традиційними ручними методами стає недостатньо ефективним. У відповідь на ці виклики були розроблені різноманітні програмні рішення, що використовують методи обробки природної мови, машинного навчання, аналізу мережевих зв'язків, семантичного моделювання та інші підходи для виявлення, класифікації та оцінювання інформаційного контенту. Аналіз існуючих систем у цій сфері дозволяє оцінити сучасний стан технологічних рішень, виявити їх функціональні можливості, переваги, обмеження та визначити підходи, які можуть бути використані при проєктуванні власного програмного рішення (таб.4) [11].

Snopes є однією з найстаріших фактчекінгових платформ у світі. Вона поєднує ручний експертний аналіз із використанням автоматизованих інструментів пошуку повторюваних наративів. NLP використовується для тематичної класифікації матеріалів та виявлення схожих тверджень.

StopFake - український фактчекінговий проєкт, створений у відповідь на масове поширення дезінформації. Основний акцент робиться на OSINT, аналізі зображень та експертній перевірці, однак також застосовуються автоматизовані інструменти пошуку й аналізу контенту.

DeepTrust спеціалізується на виявленні deepfake-контенту. Система використовує нейронні мережі для аналізу відео та аудіо, зокрема виявлення аномалій у міміці, синхронізації звуку та зображення.

Google Fact Check Tools включає Fact Check Explorer і стандарт ClaimReview. Інструмент дозволяє автоматично знаходити перевірені факти та використовує NLP для аналізу заяв і новин.

Порівняльний аналіз представлених систем подано у таблиці 1.4

Таблиця 1.4

Порівняння систем аналізу контенту

Система	NLP	Машинне навчання	OSINT	Людський фактор
Snopes	+	+	–	+
StopFake	–	частково	+	+
DeepTrust	+	+	–	–
Google FactCheck	+	+	–	+

Висновки до розділу 1

У першому розділі було розкрито теоретичні засади аналізу достовірності новинного контенту. Передусім уточнено зміст понять, з якими працює дослідження: недостовірна інформація, місінформація, дезінформація, фейкові новини, маніпуляція та пропагандистський контент. Це розмежування має практичне значення, оскільки різні типи інформаційного викривлення мають різну природу. Помилкова інформація може поширюватися ненавмисно, тоді як дезінформація створюється для свідомого впливу на аудиторію. Саме тому майбутній програмний додаток повинен не просто шукати неправдиві твердження, а враховувати намір, контекст, джерело та спосіб подання повідомлення.

Окремо було розглянуто класифікацію недостовірного контенту за тематичним спрямуванням. У роботі показано, що фейкові та маніпулятивні матеріали можуть бути політичними, економічними, псевдонауковими або емоційно забарвленими. Політичні повідомлення часто використовуються для впливу на громадську думку, економічні - для створення панічних настроїв або репутаційних втрат, псевдонаукові - для імітації доказовості, а емоційні - для швидкого поширення через страх, тривогу чи обурення. Така класифікація дала змогу зрозуміти, що аналіз достовірності новин має охоплювати не тільки фактологічну частину, а й мовні, психологічні та соціальні ознаки повідомлення.

Важливим результатом розділу стало пояснення механізмів, через які дезінформація впливає на людину. Було встановлено, що користувачі часто сприймають інформацію не через глибоку перевірку, а через знайомість, емоційність, повторюваність і відповідність власним переконанням. Ефект ілюзорної істини, підтверджувальне упередження, групова належність і

соціальний доказ роблять аудиторію вразливою до маніпуляцій. У цифровому середовищі ці чинники посилюються алгоритмічною персоналізацією, ехо-камерами та бот-мережами. Це доводить, що проблема недостовірних новин є не лише технологічною, а й поведінковою.

У межах аналізу методів перевірки було розглянуто ручний фактчекінг, OSINT-підходи та автоматизовану попередню перевірку. Ручний фактчекінг залишається найбільш надійним у складних випадках, коли потрібно встановити першоджерело, перевірити дату, контекст, цитати, зображення або відео. Водночас він потребує значного часу і не може повністю покривати великі потоки повідомлень із RSS, сайтів, соціальних мереж і Telegram. Саме тому доречним є поєднання людської експертизи з інструментами автоматизованого збору, сортування та первинної оцінки матеріалів.

Технології NLP і машинного навчання розглянуто як основу для прискорення аналізу контенту. Вони дають змогу визначати тональність тексту, знаходити емоційні маркери, виявляти узагальнення без джерел, категоричні твердження та інші мовні ознаки маніпуляції. Водночас у роботі підкреслено, що такі методи мають обмеження: вони залежать від якості навчальних даних, гірше працюють із локальним контекстом, сарказмом, прихованими смислами та новими формами дезінформації. Отже, автоматизований аналіз не повинен замінювати аналітика, а має допомагати йому швидше знаходити ризикові повідомлення.

Порівняння існуючих систем аналізу контенту показало, що найкращі результати дають комбіновані рішення, у яких технічні засоби доповнюють експертну перевірку. Це стало підґрунтям для подальшого проєктування власного додатку. Перший розділ підтвердив актуальність теми та визначив вимоги до майбутнього інструменту: він має працювати з різними джерелами, структурувати дані, підтримувати пошук і фільтрацію, виділяти підозрілі повідомлення та залишати можливість для людського висновку. Саме ці положення стали основою для проєктування системи у другому розділі.

РОЗДІЛ 2 ПРОЄКТУВАННЯ ІНТЕРАКТИВНОГО ПРОГРАМНОГО ДОДАТКУ

2.1 Постановка задачі та вимоги до системи

Після проведеного аналізу предметної області, дослідження сучасних підходів до виявлення недостовірного контенту, а також огляду існуючих програмних систем аналізу інформації постає необхідність переходу від теоретичного розгляду проблеми до її формалізації у вигляді конкретного прикладного завдання. Такий перехід є логічним етапом проєктування, оскільки саме на цьому рівні визначається, які проблеми має вирішувати розроблювана система, у яких умовах вона функціонуватиме, які процеси автоматизуватиме та яким вимогам повинна відповідати для досягнення поставленої мети.

Етап постановки задачі є одним із базових у структурі розроблення програмного забезпечення, оскільки саме він формує концептуальну основу майбутнього рішення. На цьому етапі здійснюється формалізація цілей системи, визначається її функціональне призначення, окреслюються межі застосування, встановлюються типи вхідних даних, параметри їх обробки, структура очікуваних результатів, а також вимоги до механізмів прийняття рішень. Фактично йдеться про побудову логічної моделі системи ще до початку її технічної реалізації.

Окремого значення на цьому етапі набуває визначення вимог до системи, оскільки саме вимоги виступають критеріями, відповідно до яких оцінюватиметься коректність архітектурних рішень, вибір алгоритмів, ефективність реалізації та якість кінцевого програмного продукту. При цьому мова йде не лише про функціональні вимоги, пов’язані з безпосереднім виконанням основних завдань аналізу контенту, а і про нефункціональні характеристики, зокрема продуктивність, точність роботи моделей, швидкодію, масштабованість, надійність, адаптивність до змін інформаційного середовища, безпеку обробки даних та зручність взаємодії користувача із системою.

Важливість цього етапу також зумовлена тим, що системи аналізу недостовірного контенту функціонують у складному та динамічному

середовищі, де інформаційні потоки є великими за обсягом, неоднорідними за структурою та змінними за змістом. У таких умовах нечітко сформульована задача або недостатньо обґрунтовані вимоги можуть призвести до створення системи, яка не забезпечуватиме належного рівня точності, не витримуватиме реальних навантажень або не зможе адаптуватися до нових типів інформаційних загроз. Саме тому постановка задачі в даному контексті є не формальною частиною проєктування, а фундаментом, від якого залежить ефективність усієї подальшої реалізації.

Крім того, чітке формулювання задачі та системних вимог створює підґрунтя для аргументованого вибору технологічного стеку, методів машинного навчання, інструментів обробки природної мови, архітектурного підходу та засобів інтеграції компонентів системи. Без такого обґрунтування розроблення програмного рішення ризикує перетворитися на набір технічних рішень без цілісної логіки побудови. Натомість правильно сформульована постановка задачі дозволяє забезпечити узгодженість між вимогами предметної області, функціональністю системи та технічними засобами її реалізації.

2.1.1 Актуальність задачі

В умовах сучасного інформаційного середовища, особливо під час воєнних дій та інформаційного протистояння, значно зросла кількість дезінформації, маніпулятивних матеріалів та пропагандистських повідомлень. Основними каналами поширення такої інформації є новинні веб-ресурси, соціальні мережі та месенджери, зокрема Telegram. [26]

На практиці аналітик або дослідник змушений працювати з великими обсягами неструктурованих даних, які надходять з різних джерел у режимі реального часу. Ручна перевірка кожного повідомлення є трудомісткою та неефективною, що обумовлює необхідність створення програмного продукту, здатного автоматизувати процес збору, обробки та попередньої оцінки достовірності інформації.

У зв’язку з цим виникає задача розробки програмного застосунку, який поєднує методи OSINT (Open Source Intelligence), аналіз медіаконтенту та інструменти штучного інтелекту для перевірки достовірності новин і повідомлень [6, 12].

2.1.2 Мета та завдання розробки

Метою роботи є розробка програмного застосунку для автоматизованого збору інформації з відкритих джерел (RSS-стрічок, Telegram-каналів), її структурування та аналізу з метою виявлення потенційно недостовірного або пропагандистського контенту.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- реалізувати модуль збору новин з RSS-джерел;
- реалізувати модуль отримання повідомлень з Telegram-каналів та груп;
- забезпечити збереження отриманих даних у локальній базі даних;
- реалізувати єдину стрічку подій для різних джерел;
- розробити алгоритм первинної перевірки достовірності контенту;
- інтегрувати механізми AI-аналізу тексту;
- забезпечити зручний користувацький інтерфейс для аналітика.

2.1.3 Функціональні вимоги

До основних функціональних вимог системи належать [13]:

Збір даних

- автоматичне завантаження новин з RSS-стрічок;
- отримання повідомлень з Telegram-каналів та груп, на які підписаний користувач;
- підтримка оновлення даних у реальному часі або з заданим інтервалом.

Обробка даних

- нормалізація структури повідомлень;

- уніфікація полів (дата, джерело, текст, посилання, категорії);
- фільтрація за датою, джерелом та ключовими словами.
- Аналітика
- попередня оцінка достовірності повідомлення;
- виявлення ознак пропаганди;
- формування аналітичного висновку.

Інтерфейс

- відображення даних у вигляді таблиці;
- можливість перегляду повного тексту повідомлення;
- експорт результатів у формат HTML.

2.1.4 Нефункціональні вимоги

До нефункціональних вимог відносяться:

- надійність - система не повинна завершувати роботу при помилках окремих джерел;
- продуктивність - обробка новин має здійснюватися без значних затримок;
- масштабованість - можливість підключення нових джерел;
- зручність використання - інтерфейс має бути інтуїтивно зрозумілим;
- безпека - конфіденційні дані (API-ключі) зберігаються локально.

2.2 Архітектура додатку

Програмний застосунок реалізовано з використанням архітектурного підходу MVVM (Model-View-ViewModel), який є стандартним для WPF-додатків. Такий підхід дозволяє чітко розділити логіку обробки даних, користувацький інтерфейс та модель даних [14].

Архітектура додатку складається з таких основних компонентів:

Model - описує структуру даних (NewsItem, TGItem тощо);

View - відповідає за відображення даних користувачу;

ViewModel - містить бізнес-логіку та забезпечує зв’язок між Model і View;

Service-модулі - виконують роботу з зовнішніми джерелами;

AI-модуль - здійснює аналіз текстового контенту.

Узагальнена структура проєкту має вигляд:

```
Vidoglyad
│
├── Views
│   ├── NewsBrowserView.xaml
│   ├── TGBrowserView.xaml
│   ├── AICheckView.xaml
│
├── ViewModels
│   ├── NewsBrowserViewModel.cs
│   ├── TGBrowserViewModel.cs
│   ├── AICheckViewModel.cs
│
├── Services
│   ├── RssService.cs
│   ├── TelegramModule.cs
│   ├── AIAnalyzerService.cs
│
├── Models
│   ├── NewsItem.cs
│   ├── TGItem.cs
│   ├── AICheckResult.cs
│
├── settings
│   ├── rss.json
│   ├── telegram.json
│
└── reports
```

├── rss

├── telegram

├── ai

RSS-модуль відповідає за:

- читання списку RSS-адрес з файлу rss.json;
- завантаження XML-даних;
- парсинг стрічки з використанням SyndicationFeed;
- перетворення отриманих даних у внутрішню модель NewsItem.

Кожен елемент RSS-стрічки нормалізується до уніфікованої структури, що дозволяє надалі обробляти його незалежно від джерела.

Модуль Telegram використовує API клієнта для отримання повідомлень з каналів і груп. Конфігураційні дані зберігаються у файлі telegram.json, що дозволяє уникнути жорсткого кодування ключів у програмі.

Отримані повідомлення перетворюються у модель TGItem, яка містить:

- дату і час;
- назву каналу;
- текст повідомлення;
- посилання на джерело;
- тип джерела (канал або група).

Для зручності аналітика реалізовано механізм об’єднання даних з різних джерел у єдину стрічку. Це дозволяє:

бачити хронологію подій незалежно від джерела;

порівнювати повідомлення з різних платформ;

швидко виявляти інформаційні вкиди.

2.3 Алгоритм перевірки достовірності

Перевірка достовірності повідомлень здійснюється у декілька етапів (рис. 2.1). По-перше, проводиться первинний аналіз тексту, під час якого оцінюється загальна структура повідомлення, стиль викладу, наявність емоційного забарвлення та можливих логічних невідповідностей. По-друге, відбувається виявлення ключових маркерів - характерних слів, фраз, типових помилок або шаблонів, які часто зустрічаються в недостовірній інформації. По-третє, виконується порівняння з іншими джерелами для підтвердження або спростування фактів, наведених у повідомленні. По-четверте, застосовується AI-аналіз контексту, який дозволяє враховувати семантичні зв’язки, приховані смисли та загальний фон інформації.

Такий багатоетапний підхід дає змогу ефективно поєднувати класичні методи OSINT (Open Source Intelligence) із сучасними інструментами машинного навчання, що значно підвищує точність і швидкість верифікації інформації [8, 10, 36].

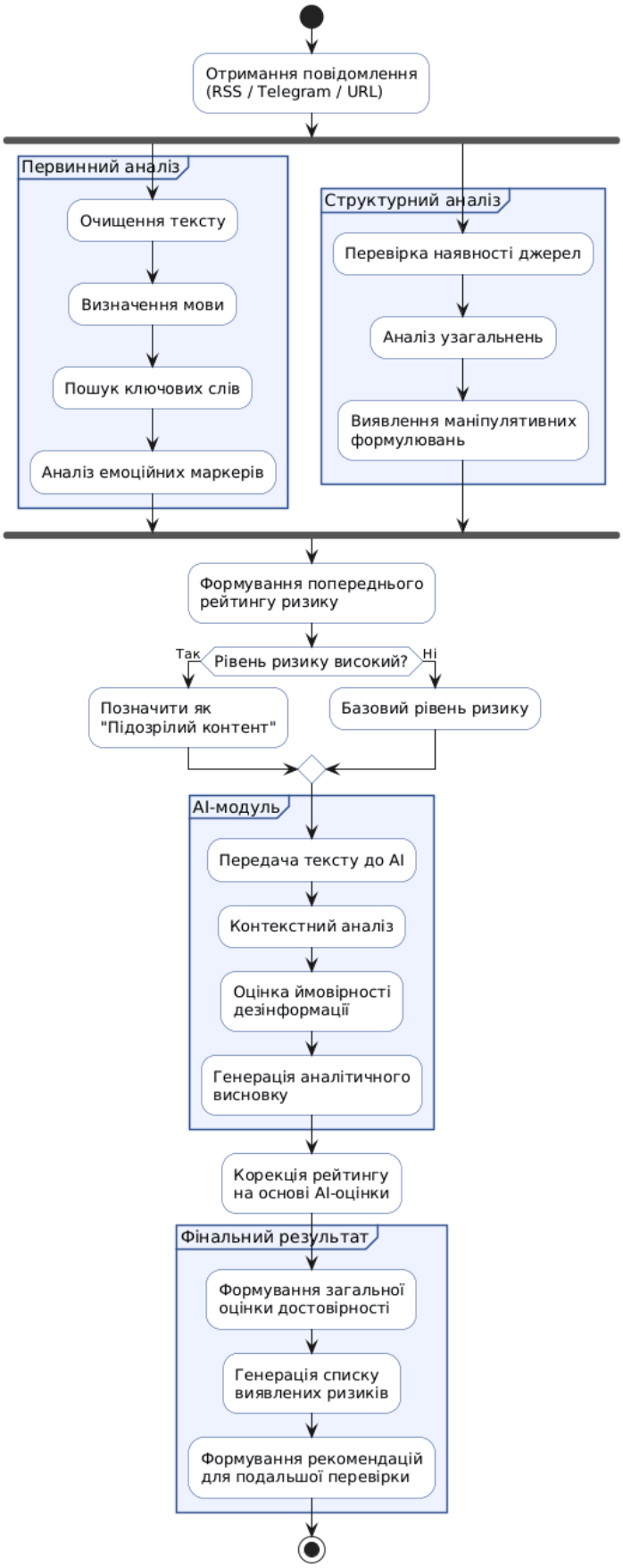

Рисунок 2.1. Алгоритм роботи програми

Висновки до розділу 2

У другому розділі було виконано перехід від теоретичного аналізу проблеми до проєктування конкретного програмного рішення. На цьому етапі визначено, які саме задачі має виконувати інтерактивний додаток для аналізу достовірності новинного контенту, які джерела він повинен обробляти та яким вимогам має відповідати. Основна ідея системи полягає у тому, щоб дати користувачу єдине робоче середовище для збору повідомлень із RSS-стрічок і Telegram, їх нормалізації, фільтрації, пошуку, первинної перевірки та підготовки результатів до подальшого аналізу.

Постановка задачі показала, що майбутній застосунок має працювати з неоднорідними даними. Новини з RSS, повідомлення з Telegram, вебпосилання, текстові фрагменти та медіафайли мають різну структуру, формат і рівень повноти. Саме тому важливою стала вимога уніфікувати дані: зберігати дату, джерело, текст, посилання, тип повідомлення та інші службові ознаки в єдиному форматі. Без такої нормалізації неможливо було б коректно виконувати пошук, фільтрацію, зіставлення джерел і подальший AI-аналіз.

У розділі було сформовано функціональні вимоги до системи. До них належать завантаження RSS-новин, отримання повідомлень із Telegram-каналів і груп, відображення даних у таблиці, фільтрація за датою, джерелом і ключовими словами, запуск AI-перевірки, аналіз URL і медіаданих, експорт результатів та робота з налаштуваннями доступу. Такі вимоги відповідають реальному сценарію користувача-аналітика: спочатку зібрати інформацію, потім швидко звузити масив даних, перевірити підозрілий матеріал і зафіксувати результат для звіту або подальшого дослідження.

Нефункціональні вимоги визначили якість роботи системи. Для такого додатку важливі не тільки наявність потрібних кнопок і таблиць, а й стійкість до помилок зовнішніх джерел, прийнятна швидкодія, можливість підключення нових каналів, зручність інтерфейсу та безпечне зберігання конфігураційних даних. RSS-джерело може бути тимчасово недоступним, Telegram може

повертати помилки доступу або обмеження частоти запитів, AI-сервіс може відповідати із затримкою. Тому система повинна коректно обробляти такі ситуації, не завершувати роботу аварійно та повідомляти користувача про стан виконання операцій.

Архітектуру додатку спроєктовано з опорою на підхід MVVM, що є доцільним для настільного застосунку з розвиненим інтерфейсом і значною кількістю даних. Поділ на Model, View і ViewModel дозволяє відокремити структуру даних від візуальної частини, а логіку роботи модулів - від елементів інтерфейсу. Завдяки цьому RSS-модуль, Telegram-модуль, AI-перевірку, налаштування і звітність можна розвивати окремо. Такий підхід зменшує ризик того, що зміна одного екрана порушить роботу інших частин програми.

У структурі проєкту виділено моделі даних, представлення, ViewModel-компоненти та сервісні модулі. RSS-сервіс відповідає за завантаження й парсинг новинних стрічок, Telegram-модуль - за отримання повідомлень із каналів і груп, AI-модуль - за первинну оцінку тексту, URL або медіаданих. Така організація є практичною, оскільки кожен компонент має власну відповідальність. Якщо в майбутньому потрібно буде додати нове джерело, змінити спосіб аналізу або розширити експорт, це можна буде зробити без повного переписування всієї системи.

Окремим результатом розділу стало формування алгоритму перевірки достовірності. Він передбачає первинний аналіз тексту, пошук маркерів ризику, зіставлення з іншими джерелами та AI-оцінку контексту. Такий алгоритм не проголошує автоматичну перевірку остаточною, а працює як фільтр для виявлення матеріалів, які потребують уваги людини. У підсумку другий розділ сформував технічний фундамент роботи: визначено задачі, вимоги, архітектуру, структуру модулів і логіку аналізу, що створило основу для реалізації програмного продукту в третьому розділі.

РОЗДІЛ 3 РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОДУКТУ

3.1 Структура та функціонал інтерфейсу

Головне вікно додатку має чітко продуману структуру і поділене на чотири основні функціональні області. У верхній частині розташовано меню навігації, яке забезпечує швидкий доступ до головних розділів системи. Нижче розміщені вкладки основних модулів, що дозволяють користувачеві моментально перемикатися між ключовими функціональними блоками (каталог автомобілів, клієнти, договори, фінанси, сервісне обслуговування тощо). Центральну частину вікна займає основна таблиця даних, яка відображає поточний набір записів з можливістю сортування, фільтрації та пошуку. Праворуч розташоване опціональне бічне меню, яке динамічно з’являється залежно від контексту і містить додаткові інструменти та дії, пов’язані з активним модулем.

Такий модульний поділ інтерфейсу дозволяє працювати з великими обсягами інформації, зберігаючи при цьому повний контекст поточної операції та забезпечуючи високу зручність використання навіть під час інтенсивної роботи кількох користувачів одночасно.

Інтерфейс програми побудовано модульно: кожна велика функція (збір RSS, Telegram-парсер, AI-перевірка) оформлена як окремий екран (UserControl) або окреме вікно (Window). Навігація здійснюється через меню (MenuItem), яке викликає команди ViewModel. Такий підхід дозволяє розвивати додаток поступово: модулі можна додавати незалежно, не ламаючи основний інтерфейс (рис.7).

Практичний принцип, якого я трималась: "однакова структура для всіх стрічок". Тобто: зліва - таблиця (DataGrid), справа - панель опцій. Це виглядає простіше, ніж "все в одному вікні", але для OSINT-аналізу працює краще: очі звикають, інструменти повторюються, і користувач швидше орієнтується.

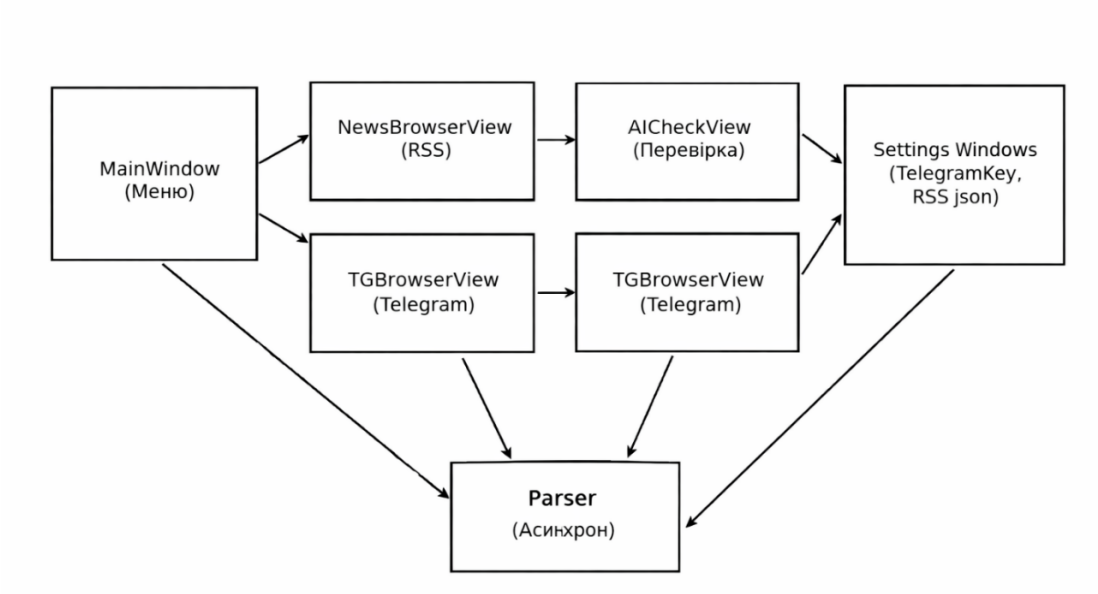

Рисунок 3.1. Схема навігації між основними модулями

На схемі показано логіку: головне меню відкриває модулі (RSS, Telegram, AI, мапа), а також вікна налаштувань (наприклад, TelegramKeyWindow). Це важливо, бо налаштування не повинні бути "всередині" робочої стрічки - вони змінюються рідко і краще тримати їх окремо [27, 28].

• MVVM та зв’язування даних (Bindings) [35]

Модулі реалізовані у стилі MVVM: XAML описує лише вигляд і прив’язки, а логіка завантаження, фільтрації і експорту знаходиться у ViewModel. Так я уникаю зайвого code-behind і отримую більш керовану структуру коду.

У проєкті для більшості кнопок використано ICommand (RelayCommand). Наприклад, кнопки "Оновити", "Фільтр", "Експорт" прив’язані до команд у ViewModel. А DataGrid прив’язаний до ObservableCollection, що забезпечує автоматичне оновлення списку в UI при зміні даних.

Окремий нюанс: DataContext я задавала прямо в XAML у UserControl.DataContext. Це знижує ризик ситуації, коли модуль відкрився, але нічого не відображається через відсутній контекст. У дипломі це можна пояснити як "захист від помилок ініціалізації".

• Модуль RSS-стрічки новин (NewsBrowserView)

RSS-модуль - один з базових у Vidoglyad (рис.3.2). Його завдання: завантажувати перелік RSS-URL із settings/rss.json, тягнути новини, показувати

їх у таблиці і давати швидку фільтрацію. Я зробила його як "еталонний екран", щоб потім за тим самим принципом зробити Telegram-екран.

Лістинг коду 3.1 - Реалізація сервісу завантаження новин із RSS-стрічки

```
using System;
using System.Collections.Generic;
using CodeHollow.FeedReader;
public class RssService
{
public List<RssItem> LoadFeed(string url)
{
var result = new List<RssItem>();
var feed = FeedReader.ReadAsync(url).Result;
foreach (var item in feed.Items)
{
result.Add(new RssItem
{
Title = item.Title,
Link = item.Link,
PublishDate = item.PublishingDate ?? DateTime.Now
});
}
return result;
}
}
public class RssItem
{
public string Title { get; set; }
public string Link { get; set; }
public DateTime PublishDate { get; set; }
}
```

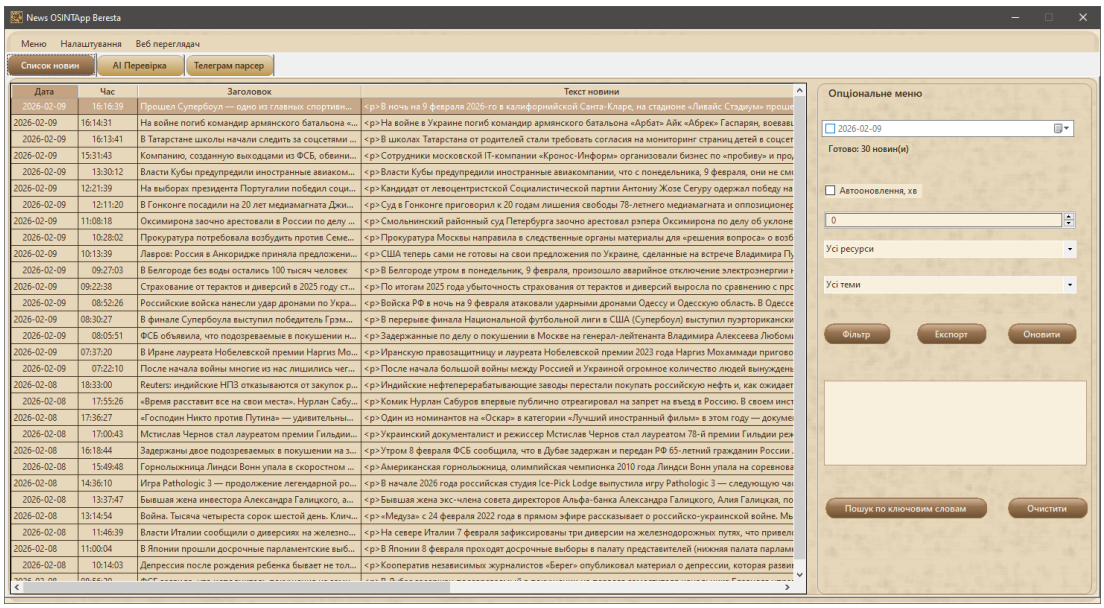

Рисунок 3.2. Компонування елементів NewsBrowserView

Для практичної роботи додано дві функції: tooltip (підказка при наведенні на рядок) і копіювання посилання по подвійній дії мишею. Ці дрібні речі реально економлять час під час перевірки інформації.

Лістинг коду 3.2 - Налаштування взаємодії з таблицею новин у WPF-інтерфейсі

```
<DataGrid.InputBindings>
    <MouseBinding MouseAction="LeftDoubleClick"
                  Command="{Binding CopyLinkCommand}"
                  CommandParameter="{Binding SelectedItem,
RelativeSource={RelativeSource AncestorType=DataGrid}}"/>
</DataGrid.InputBindings>
<!-- ROW HOVER TOOLTIP -->
<DataGrid.RowStyle>
    <Style TargetType="DataGridRow">
        <Setter Property="ToolTip">
            <Setter.Value>
                <Border Background="#FFF7E6"
                        BorderBrush="#8B6B3E"
                        BorderThickness="1"
                        Padding="8"
                        MaxWidth="600">
                    <StackPanel>
```

```
                        <TextBlock Text="{Binding Title}"
                                   FontWeight="Bold"
                                   TextWrapping="Wrap"
                                   Margin="0,0,0,6"/>
                        <TextBlock Text="{Binding Summary}"
                                   TextWrapping="Wrap"/>
                    </StackPanel>
                </Border>
            </Setter.Value>
        </Setter>
    </Style>
</DataGrid.RowStyle>
```

• Панель автооновлення та статус "оновлюється / завершено"

Автооновлення реалізовано через DispatcherTimer. Користувач вмикає CheckBox, задає інтервал у хвилинах, і програма періодично викликає ReloadAsync. Щоб було зрозуміло, що саме відбувається, біля поля інтервалу я додала статусний TextBlock (RefreshStatusText) та прапорець стану (IsRefreshing).

Це рішення закриває типову проблему інтерфейсів з мережевими запитами: користувач бачить, що програма працює, а не "зависла".

Лістинг коду 3.3 - Налаштування взаємодії з таблицею новин у WPF-інтерфейсі

```
     <!-- AUTO REFRESH + STATUS -->
<StackPanel Orientation="Vertical" Margin="0,0,0,12">
    <StackPanel Orientation="Horizontal">
        <CheckBox Content="Автооновлення, хв"
                  IsChecked="{Binding IsAutoRefreshEnabled}"
                  Margin="0,0,10,0"/>
        <TextBox Width="60"
                 VerticalContentAlignment="Center"
                 Text="{Binding RefreshMinutesText}"
                 IsEnabled="{Binding IsAutoRefreshEnabled}"/>
    </StackPanel>
```

```
    <TextBlock Text="{Binding RefreshStatusText}"
               FontSize="11"
               Margin="22,4,0,0"
               Foreground="{Binding IsRefreshing,
                           Converter={StaticResource
BoolToStatusBrushConverter}}"/>
</StackPanel>
```

• Модуль Telegram: вікно ключів та екран стрічки [15, 35]

Telegram-модуль у моєму проєкті складається з двох UI-частин: вікна для введення доступів (TelegramKeyWindow) і екрану перегляду повідомлень (TGBrowserView) (рис.3.3). Розділення потрібне, бо доступи - це налаштування, а стрічка - робочий інструмент (рис.3.4).

Лістинг коду 3.4 - Реалізація блоку автооновлення новинної стрічки

```
using WTelegram;
using TL;
public class TelegramService
{
private Client _client;
public async void Init(string apiId, string apiHash)
{
_client = new Client(config =>
{
if (config == "api_id") return apiId;
if (config == "api_hash") return apiHash;
return null;
});
await _client.LoginUserIfNeeded();
}
public async Task<List<string>> LoadMessages(string channel)
{
var messages = new List<string>();
var peer = await _client.Contacts_ResolveUsername(channel);
var history = await _client.Messages_GetHistory(peer.peer,
limit: 50);
```

```
foreach (var msg in history.Messages)
{
if (msg is Message m && !string.IsNullOrEmpty(m.message))
messages.Add(m.message);
}
return messages;
}
}
```

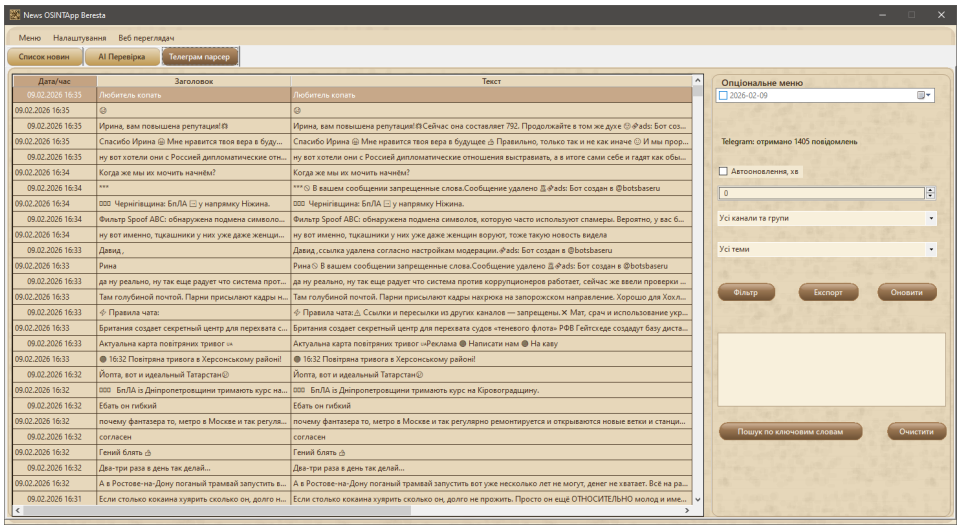

Рисунок 3.3. Телеграм модуль

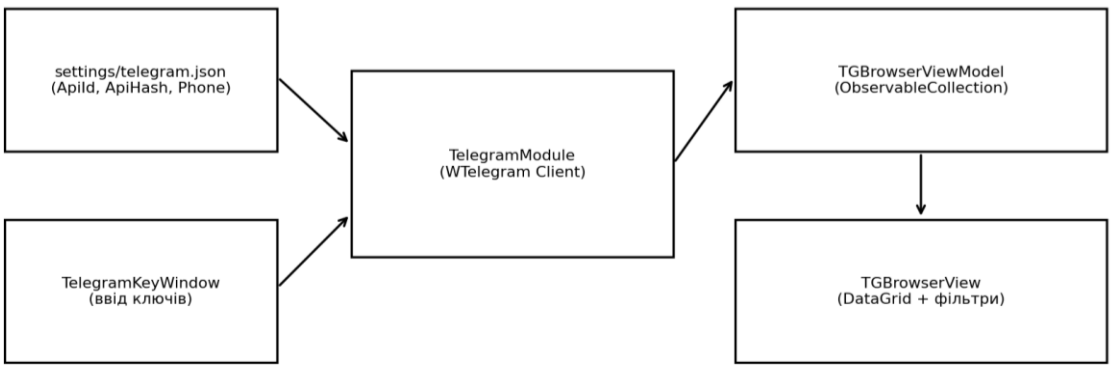

Рисунок 3.4. Логічний ланцюжок: telegram.json → TelegramModule → ViewModel → інтерфейс

• TelegramKeyWindow: інтерфейс введення ApiId/ApiHash/Phone

Вікно TelegramKeyWindow реалізовано як окремий Window з власним DataContext (TelegramKeyViewModel). Користувач вводить ApiId, ApiHash і

номер телефону, після чого натискає "Зберегти". ViewModel зберігає дані у settings/telegram.json (рис. 11). Для UX додано StatusText, де відображається короткий результат дії (наприклад, "Збережено" або повідомлення про помилку), це показано на рисунку 3.5.

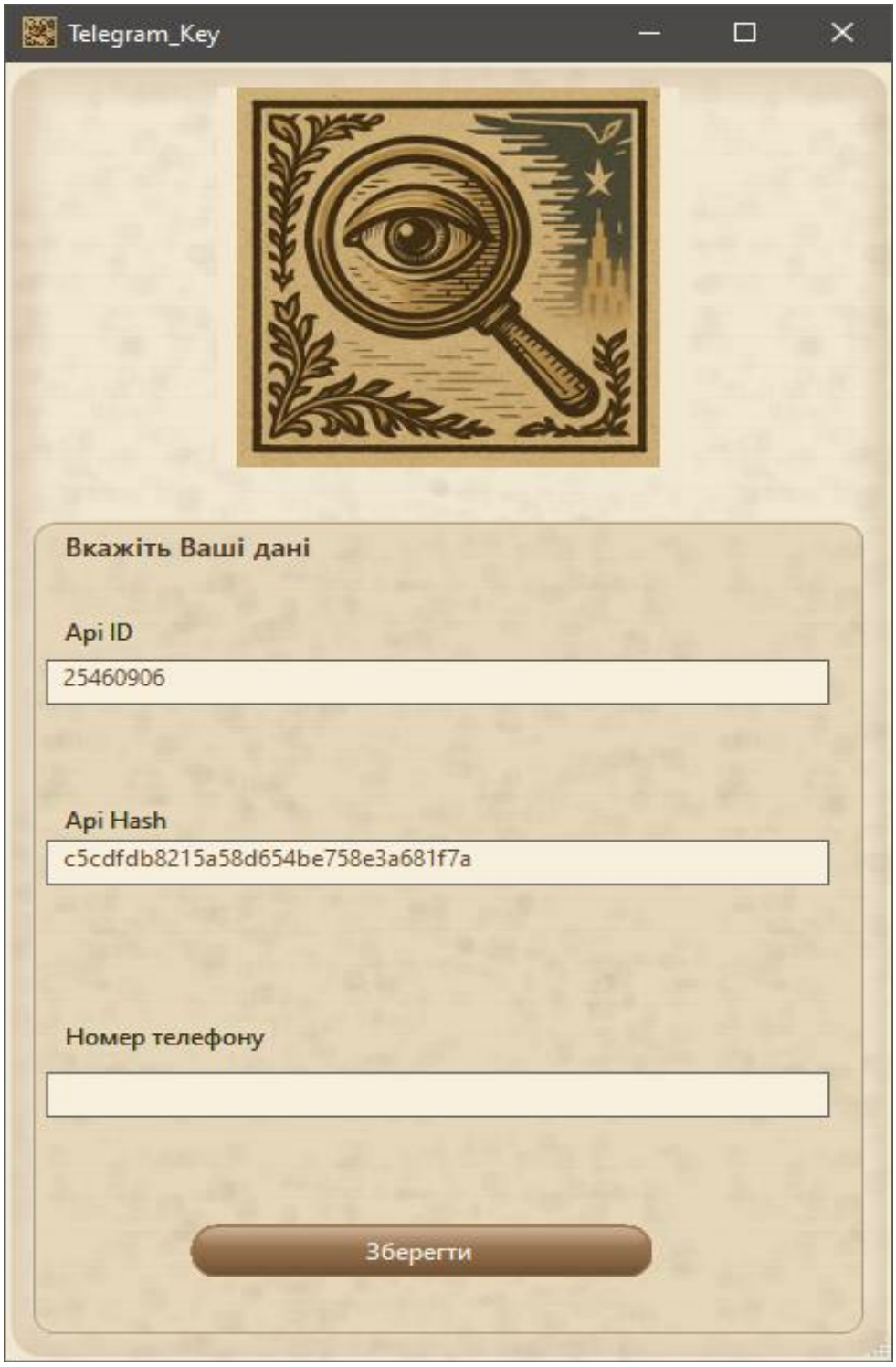

Рисунок 3.5. Вікно налаштування Telegram

• TGBrowserView: структура як у RSS, але з Telegram-специфікою

TGBrowserView повторює принцип RSS: таблиця зліва, панель опцій справа. Однак для Telegram потрібно показувати: назву каналу/групи, тип (канал чи група), а також посилання на канал (t.me) і на конкретне повідомлення.

Тому в таблиці передбачені додаткові колонки: "Канал/група", "Тип", "t.me". У панелі опцій додається фільтр за типом чату (ComboBox: Усі/Канали/Групи). Далі, як і в RSS, є фільтр за датою, поле пошуку і статус оновлення.

• Об’єднана стрічка RSS + Telegram (інтерфейсний сценарій)

Під час роботи над проєктом з’явилась ідея звести RSS і Telegram в одну спільну стрічку. На рівні інтерфейсу це означає, що користувач бачить один DataGrid, а джерело позначено окремою колонкою (тип або іконка). Це дає змогу порівнювати інформацію в одному місці й швидше знаходити підтвердження або суперечності.

У дипломі цей пункт доречно показати як "розширення функціоналу", бо він логічно витікає з модульної побудови інтерфейсу.

• Візуалізація активності Telegram-каналів (Live-heatmap)

Ще одна ідея, яка виросла з практики: у Telegram багато каналів, і корисно бачити, які з них зараз активні. Для цього запропоновано heatmap-візуалізацію: рядки - канали/групи, стовпці - години або інші інтервали часу, а інтенсивність показує кількість повідомлень (рис. 12).

Нижче на рисунку 3.6 наведено ескіз такої візуалізації, який можна реалізувати у WPF як окремий таб або як додаткову панель:

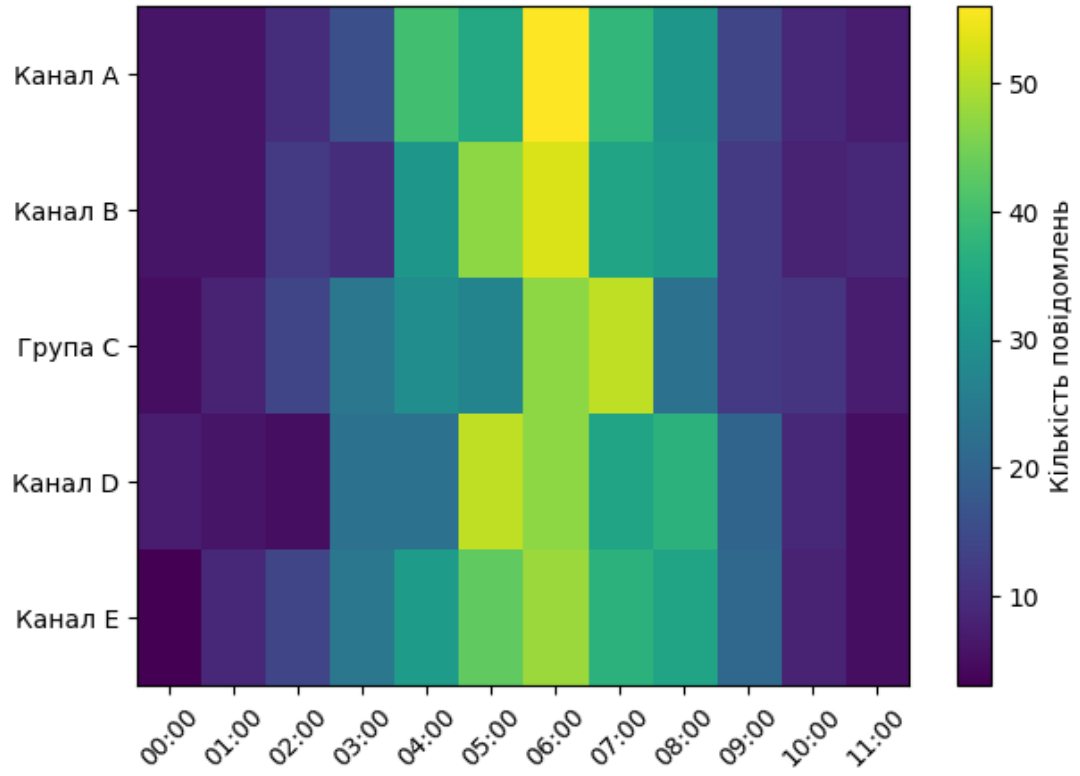

Рисунок 3.6. Ескіз heatmap активності Telegram-каналів

• Інтеграція AI-перевірки у користувацький сценарій

Для OSINT-додатку важливо не тільки зібрати контент, а й мати швидку перевірку. У проєкті передбачено, що користувач може виділити елемент у стрічці і запустити AI-аналіз (наприклад, детекцію пропаганди або маніпулятивної риторики). Результат відображається або в окремому модулі, або в правій частині інтерфейсу - залежно від обраної архітектури (рис. 3.7).

З точки зору інтерфейсу це означає: кнопка/команда для запуску, блок результатів з коротким підсумком і аргументацією, і кнопка експорту в HTML для звітності. Такий UX-ланцюжок робить систему придатною для навчального/прикладного використання: є доказ (звіт), а не просто "оцінка в голові" [6, 36].

Лістинг коду 3.5 - Реалізація сервісу передавання тексту до AI-модуля перевірки

```
using System.Net.Http;
using System.Text;
using System.Threading.Tasks;
```

```
public class AiCheckService
{
private readonly HttpClient _http = new HttpClient();
public async Task<string> AnalyzeText(string text)
{
var json = $"{{ \"text\": \"{text}\" }}";
var content = new StringContent(json, Encoding.UTF8,
"application/json");
var response = await
_http.PostAsync("https://api.example.ai/analyze", content);
return await response.Content.ReadAsStringAsync();
}
}
```

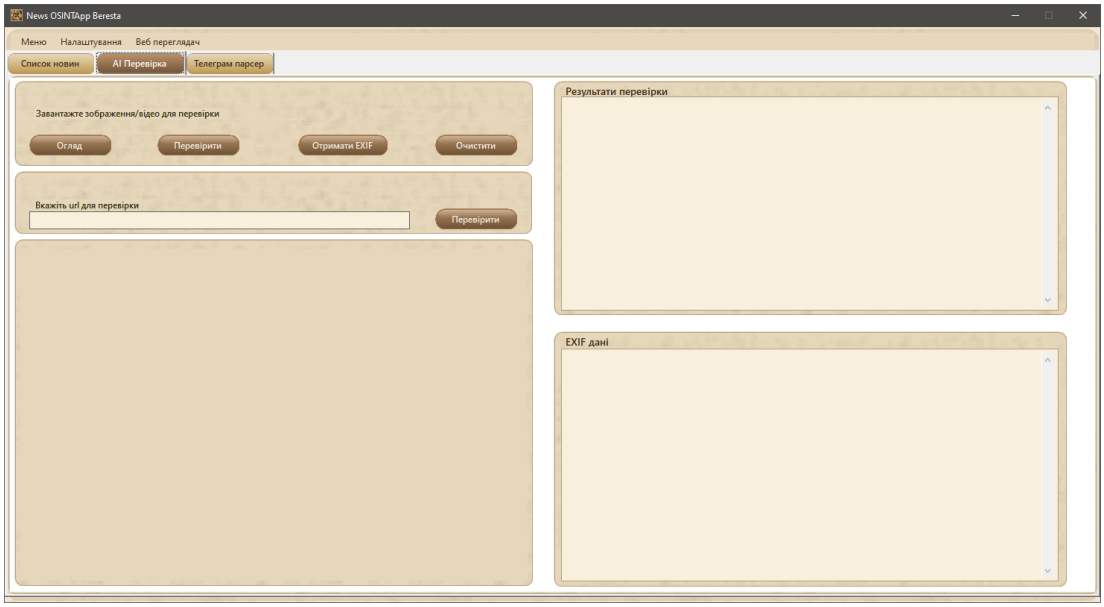

Рисунок 3.7. Модуль AI перевірки

• Ключові фрагменти ViewModel для роботи UI

Нижче наведено фрагмент для індикатора оновлення. Саме він пов’язаний з TextBlock статусу в інтерфейсі:

Лістинг коду 3.5 - Реалізація стану оновлення новинної стрічки у ViewModel

```
private bool _isRefreshing;
public bool IsRefreshing
{
```

```
    get => _isRefreshing;
    private set { _isRefreshing = value; OnPropertyChanged(); }
}
private string _refreshStatusText = "Готово";
public string RefreshStatusText
{
    get => _refreshStatusText;
    private set { _refreshStatusText = value; OnPropertyChanged();
}
}
private async Task ReloadAsync()
{
    try
    {
        IsRefreshing = true;
        RefreshStatusText = "Оновлення…";
        RefreshStatusText = "Завершено";
    }
    catch
    {
        RefreshStatusText = "Помилка оновлення";
        throw;
    }
    finally
    {
        IsRefreshing = false;
    }
}
```

3.2 Тестування та оцінка ефективності

У цьому підрозділі наведено результати тестування програмного продукту Vidoglyad з позиції контролю якості та перевірки працездатності основних модулів. Було сформовано план тестування, підібрані тестові дані, виконаніа контрольні сценарії та зафіксовані результати у вигляді тест-кейсів і висновків. Оскільки Vidoglyad працює з відкритими джерелами та зовнішніми сервісами,

частина тестів орієнтована на стійкість до помилок мережі, коректність обробки контенту та стабільність інтерфейсу під навантаженням.

У межах дипломного проєкту функціональні можливості системи були сфокусовані на трьох ключових модулях:

(1) агрегація новин через RSS,

(2) парсинг Telegram-каналів/груп,

(3) AI-перевірка текстів/зображень/посилань.

Додатково тестувався вбудований веб-переглядач як інструмент швидкого переходу до джерела, а також вікно налаштувань лімітів і затримок, що впливають на стабільність роботи із зовнішніми сервісами.

Метою тестування є підтвердження коректної роботи інтерфейсу та функцій Vidoglyad у типових і граничних сценаріях використання, а також оцінка ефективності системи з точки зору швидкодії, зручності та стійкості.

Об’єктом тестування є настільний застосунок Vidoglyad (Windows Forms, мова C#), що включає модулі RSS, Telegram, AI-перевірки, налаштування лімітів та вбудований веб-переглядач. У коді системи використовуються компоненти: CodeHollow.FeedReader для RSS, HttpClient для мережевих запитів, Newtonsoft.Json / JObject для роботи з конфігураціями, TelegramNewsService для отримання Telegram-повідомлень, а для медіа/метаданих - Windows Media Player ActiveX (AxWMPLib) та Shell32 (COM).

Критерії успішності тестування (acceptance criteria) визначено так:

- Функціональна коректність: кожен модуль виконує свою задачу без критичних помилок.
- Стійкість: при помилках мережі/джерела програма не завершується аварійно, повідомляє користувача та відновлює роботу.
- Продуктивність: інтерфейс залишається придатним до використання при збільшенні кількості записів (сотні-тисячі).
- Юзабіліті: користувач може виконати базові сценарії (оновити, фільтрувати, знайти, перевірити, експортувати) без навчання.

• Відтворюваність: результати тестів описані так, щоб інший студент/викладач міг повторити перевірки.

Стратегія тестування комбінувала декілька рівнів: модульне (перевірка логіки окремих блоків), інтеграційне (перевірка взаємодії UI та сервісів), системне (повний сценарій користувача), а також нефункціональні перевірки (продуктивність, надійність, зручність, безпека). Для фіксації результатів застосовано тест-кейси з очікуваним та фактичним результатом і позначкою статусу (Pass/Fail).

План тестування було сформовано за такими напрямами [29, 30]:

1) Функціональні тести RSS.

2) Функціональні тести Telegram.

3) Функціональні тести AI-перевірки (текст/URL/медіа).

4) Тести фільтрації, пошуку та експорту.

5) Тести відмовостійкості: таймаути, відсутність мережі, некоректні дані.

6) Тести продуктивності: збільшення обсягу таблиць, автооновлення.

7) Юзабіліті-тести: сценарії користувача, читабельність, зрозумілість повідомлень.

8) Регресійні тести після правок (ліміти, UI-логіка).

Тестування виконувалося у настільному середовищі Windows з типовими параметрами, які відповідають умовам використання в навчальній лабораторії або на домашньому ПК. Програмний продукт є Windows Forms застосунком (C#) та використовує мережеве підключення для отримання даних. Під час тестування враховано вплив мережі: швидкість з’єднання, короткочасні розриви, зміни доступності джерел.

До тестових даних віднесено [31, 32]:

• RSS-канали (новинні ресурси, які віддають RSS/XML);

• Telegram-канали/групи (повідомлення різної структури, у т.ч. службові позначки, емодзі, посилання);

• Тексти та URL для AI-перевірки;

• Зображення/відео для перевірки медіа та метаданих (EXIF).

Як доказ коректної роботи інтерфейсу та результатів отримання даних до розділу включено скріншоти ключових модулів (рис. 3.8, 3.9, 3.10).

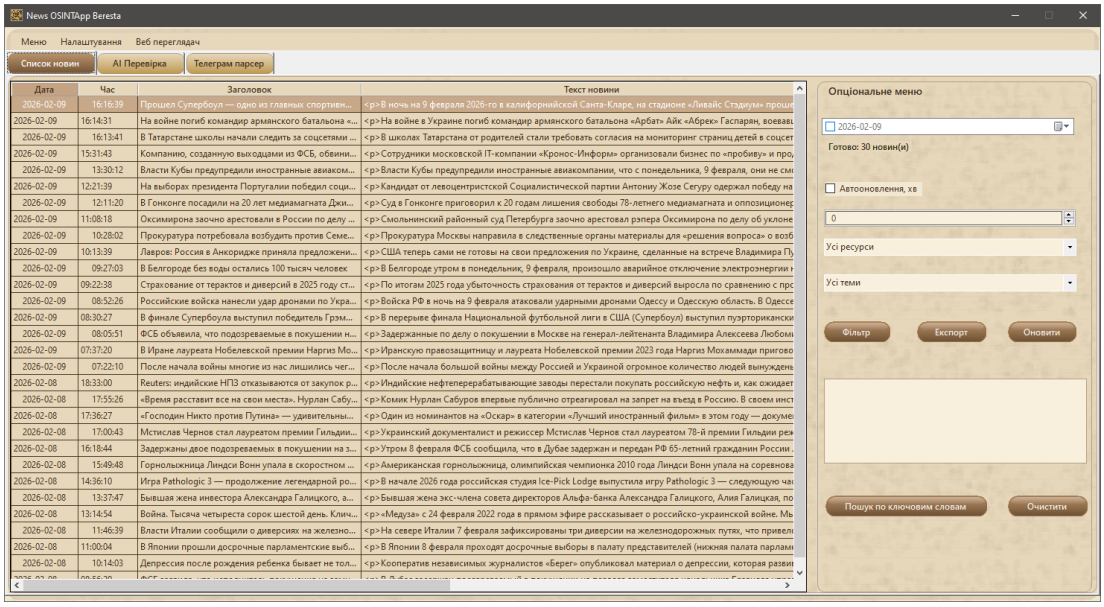

Рисунок 3.8. Список RSS-новин у головному вікні

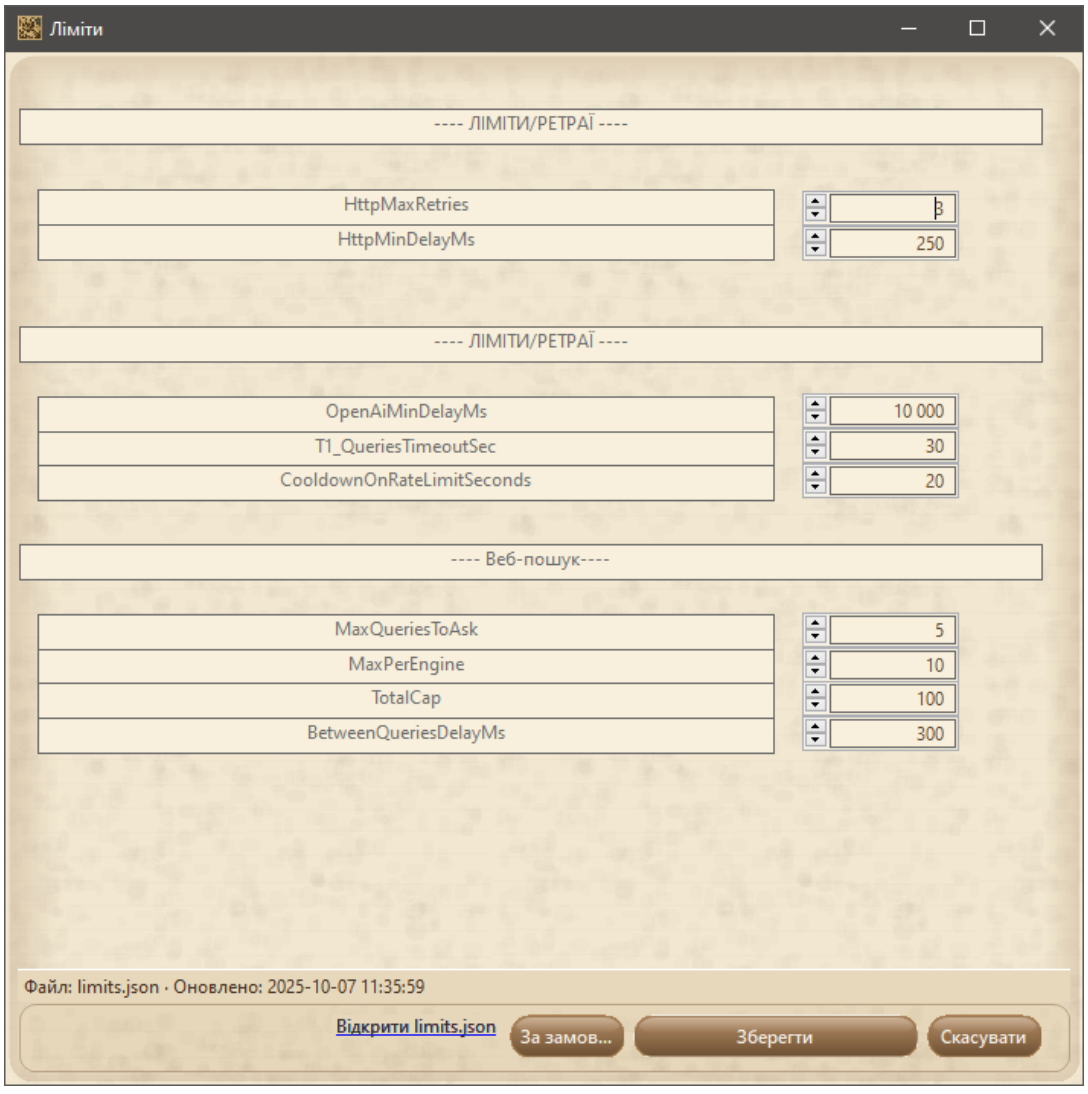

Рисунок 3.9. Керування лімітами

Рисунок 3.10. Вікно введення Telegram API параметрів

3.2.1 Функціональне тестування: модуль RSS

Модуль RSS забезпечує отримання новин через RSS-канали та відображення їх у табличному вигляді (основна таблиця позначена як dataGridView1). Тести виконувалися для перевірки: коректного завантаження,

парсингу полів (дата/час/заголовок/текст), оновлення, фільтрації та стабільності при різних форматах RSS (таб. 3.1) [16, 17, 18].

Таблиця 3.1

Результати тесту RSS

ID	Опис	Кроки	Очікуваний результат	Статус
RSS-01	Завантаження новин з коректного RSS-джерела	Обрано ресурс → "Оновити"	З'являються нові записи у таблиці	Pass
RSS-02	Обробка RSS із HTML у описі	Джерело з тегами <p>, 	HTML не ламає UI, текст відображається читабельно	Pass
RSS-03	Сортування за датою/часом	Клік по заголовку колонки дати	Записи впорядковуються коректно	Pass
RSS-04	Фільтр за датою	Вибір дати у правій панелі	Залишаються новини за обраний день	Pass
RSS-05	Пошук за ключовими словами	Ввести слово → "Пошук"	Відображаються тільки релевантні рядки	Pass
RSS-06	Відсутність мережі	Вимкнути інтернет → "Оновити"	Програма не падає, показує повідомлення/статус	Pass
RSS-07	Некоректний RSS (битий XML)	Підставити неправильний URL	Показ помилки, пропуск джерела, UI живий	Pass
RSS-08	Обмеження на кількість елементів	Завантажити великий канал	UI не зависає, скрол працює	Pass

Під час тестування варто звернути увагу на те, що новинні тексти часто містять HTML-розмітку. Тому важливим критерієм була відсутність "поломки" рядків у таблиці: текст має переноситися/обрізатися контрольовано, а програма не повинна виводити сирий код у вигляді, який ускладнює читання.

3.2.2 Функціональне тестування: модуль Telegram

Telegram-модуль відповідає за підключення до каналів/груп і завантаження повідомлень у внутрішню таблицю (в інтерфейсі - dataGridView2). Тестувалося:

введення API параметрів, отримання історії, автооновлення, відображення емодзі/посилань, фільтрація за каналами й темами, а також стійкість при великій кількості повідомлень (див.таблицю 3.2)

Таблиця 3.2

Результати тесту Telegram

ID	Опис	Кроки	Очікуваний результат	Статус
TG-01	Відкриття вікна введення API даних	Меню/Налаштування → Telegram Key	Вікно відкривається, поля доступні	Pass
TG-02	Збереження API ID/Hash	Ввести значення → "Зберегти"	Дані зберігаються у конфігурації	Pass
TG-03	Підключення та отримання повідомлень	Вказати телефон → підключитися	Повідомлення завантажуються у таблицю	Pass
TG-04	Відображення емодзі та спецсимволів	Отримати повідомлення з емодзі	Символи не "ламають" рядок і таблицю	Pass
TG-05	Автооновлення	Увімкнути автооновлення, встановити інтервал	Нові повідомлення додаються без дублювання	Pass
TG-06	Велика кількість повідомлень	Завантажити історію (понад 1000)	UI залишається придатним, скрол працює	Pass
TG-07	Обробка видалених/службових повідомлень	Отримати повідомлення типу "видалено"	Відображається нейтрально, без помилок	Pass
TG-08	Відсутність доступу/бан/помилка API	Симулювати помилку авторизації	Повідомлення про проблему, без аварійного завершення	Pass

Окремо зверталася увага на проблему "засмічення" даних: в Telegram можуть приходити рекламні рядки, службові позначки або фрагменти бот-повідомлень. З точки зору тестування важливо, щоб програма не зависала на "важких" повідомленнях і коректно відображала текст незалежно від довжини.

3.2.3 Функціональне тестування: AI-перевірка (текст/URL/медіа)

Модуль AI-перевірки є одним із ключових компонентів системи та призначений для комплексного аналізу контенту з метою оцінки його

достовірності, виявлення ознак інформаційних маніпуляцій, пропагандистських технік та спроб дезінформації. Модуль автоматично формує структурований висновок, який містить рівень довіри до повідомлення, виявлені ризики та рекомендації щодо подальшої верифікації.

У реалізації модуля передбачено гнучке налаштування параметрів продуктивності та стабільності роботи. Зокрема, реалізовані механізми контролю мережевих запитів через змінні HttpMaxRetries (максимальна кількість повторних спроб) та HttpMinDelayMs (мінімальна затримка між запитами), а також спеціальні обмеження для взаємодії з зовнішніми AI-сервісами - OpenAiMinDelayMs. Такі налаштування дозволяють уникнути перевантаження API, запобігти блокуванням за надмірну частоту запитів і забезпечити стабільну роботу модуля навіть при високому навантаженні.

Тестування модуля AI-перевірки проводилося окремо і включало кілька основних сценаріїв. Перший сценарій передбачав аналіз URL-адрес на предмет достовірності джерела та потенційних ознак фішингу. Другий - глибокий семантичний аналіз наданого тексту з виявленням маніпулятивних технік. Третій сценарій охоплював роботу з медіаконтентом: завантаження зображень та відео, вилучення та аналіз EXIF-даних і метаданих файлів. Усі сценарії тестувалися на реальних та спеціально підготовлених прикладах, що містили як достовірну інформацію, так і типові ознаки дезінформації (див. таблицю 3.3).

Таблиця 3.3

Результати тесту AI модуля

ID	Опис	Кроки	Очікуваний результат	Статус
AI-01	Введення URL і запуск перевірки	Вставити URL → "Перевірити"	Отримано текстовий результат перевірки	Pass
AI-02	Порожній URL	Не вводити нічого → "Перевірити"	Попередження про некоректні дані	Pass
AI-03	Невалідний URL	Ввести 'abc' → "Перевірити"	Повідомлення про помилку, без падіння	Pass

AI-04	Завантаження зображення для аналізу	"Огляд" → обрати файл	Прев'ю відображається, файл прийнято	Pass
AI-05	Отримання EXIF	Після вибору фото натиснути "Отримати EXIF"	У блоці EXIF відображені метадані (якщо є)	Pass
AI-06	Файл без EXIF	Зображення без метаданих → "EXIF"	Пояснення "немає даних", без помилки	Pass
AI-07	Великий файл/відео	Вибір великого медіа	Застосунок не падає, показує прогрес/стан	Pass
AI-08	Таймаут/429 від сервісу	Імітувати часті запити	Вмикається затримка/ретрай, UI живий	Pass

Для оцінки ефективності AI-перевірки використовувався якісний критерій: результат має бути зрозумілим для користувача без додаткових пояснень. У дипломній роботі я не оцінюю "істинність" кожного висновку як науковий факт, але перевіряю інженерний аспект: модуль приймає дані, обробляє їх, повертає відповідь і не зриває роботу всієї системи [25].

3.2.4 Тестування фільтрації, пошуку та експорту

Оскільки модуль "Vidoglyad" призначений передусім для аналітичної роботи з великими масивами інформації, особливу увагу було приділено розробці зручних допоміжних функцій. Серед них ключовими є розширена фільтрація даних, яка дозволяє користувачеві відбирати повідомлення за датою публікації, джерелом інформації (RSS-канали або Telegram-канали), тематичними категоріями та рівнем пріоритетності. Не менш важливою є функція повнотекстового пошуку за ключовими словами, що підтримує складні запити з використанням логічних операторів. Крім того, реалізована можливість експорту результатів у різних форматах (PDF, Excel, CSV), що дає змогу швидко переносити відібрані дані для подальшого аналізу або включення до звітів.

Для підтвердження надійності та коректності роботи цих функцій було проведено окреме тестування. Перевірка здійснювалася як на даних, отриманих із RSS-джерел, так і на масивах повідомлень з Telegram-каналів. Тестові сценарії

включали різноманітні комбінації фільтрів, пошук за складними запитами та експорт великих наборів даних. Результати тестування засвідчили стабільну та коректну роботу всіх допоміжних функцій незалежно від типу джерела (див. таблицю 3.4) [26].

Таблиця 3.4

Результати UX тесту

ID	Опис	Кроки	Очікуваний результат	Статус
UX-01	Фільтр за джерелом (RSS)	Обрати "Усі ресурси"/конкретний ресурс	Список змінюється відповідно до вибору	Pass
UX-02	Фільтр за темами	Обрати тему → застосувати	Показано тільки релевантні записи	Pass
UX-03	Пошук за ключовими словами	Ввести фразу → "Пошук"	Підсвічення/відбір релевантних рядків	Pass
UX-04	Очищення фільтрів	Натиснути "Очистити"	Скинуто вибір, показано повний список	Pass
UX-05	Експорт даних	Натиснути "Експорт"	Файл створено, дані відповідають таблиці	Pass

3.3 Нефункціональне тестування: продуктивність та надійність

Нефункціональні тести відігравали критичну роль у перевірці працездатності системи, оскільки модуль "Vidoglyad" працює з великими потоками даних і здатен накопичувати тисячі рядків у таблицях. Основна увага при тестуванні приділялася оцінці продуктивності та зручності роботи інтерфейсу під навантаженням.

Зокрема, оцінювалися такі параметри:

• час завантаження та відображення великих наборів даних;

• швидкість реакції інтерфейсу під час прокрутки, сортування та фільтрації таблиць;

• стабільність роботи при автоматичному оновленні контенту в реальному часі;

• поведінка системи при виникненні помилок мережі та обмеженнях зовнішніх сервісів (таймаути, обмеження частоти запитів тощо).

Як приклад навантажувального сценарію для Telegram-даних була використана історія повідомлень одного каналу обсягом понад 1000 записів. У цьому режимі перевірялося, чи не виникають "фрізи" інтерфейсу при швидкій прокрутці, чи залишається стабільною можливість фільтрувати та сортувати список, а також наскільки швидко відбувається перемикання між вкладками та оновлення даних. Аналогічні тести проводилися і для RSS-потоків з великою кількістю публікацій (див. рис.3.10, таб. 3.5).

Результати нефункціонального тестування підтвердили, що система зберігає прийнятну швидкодію та зручність використання навіть при роботі з великими обсягами інформації, хоча й виявили окремі точки зростання, пов’язані з оптимізацією рендерингу таблиць при екстремальних навантаженнях.

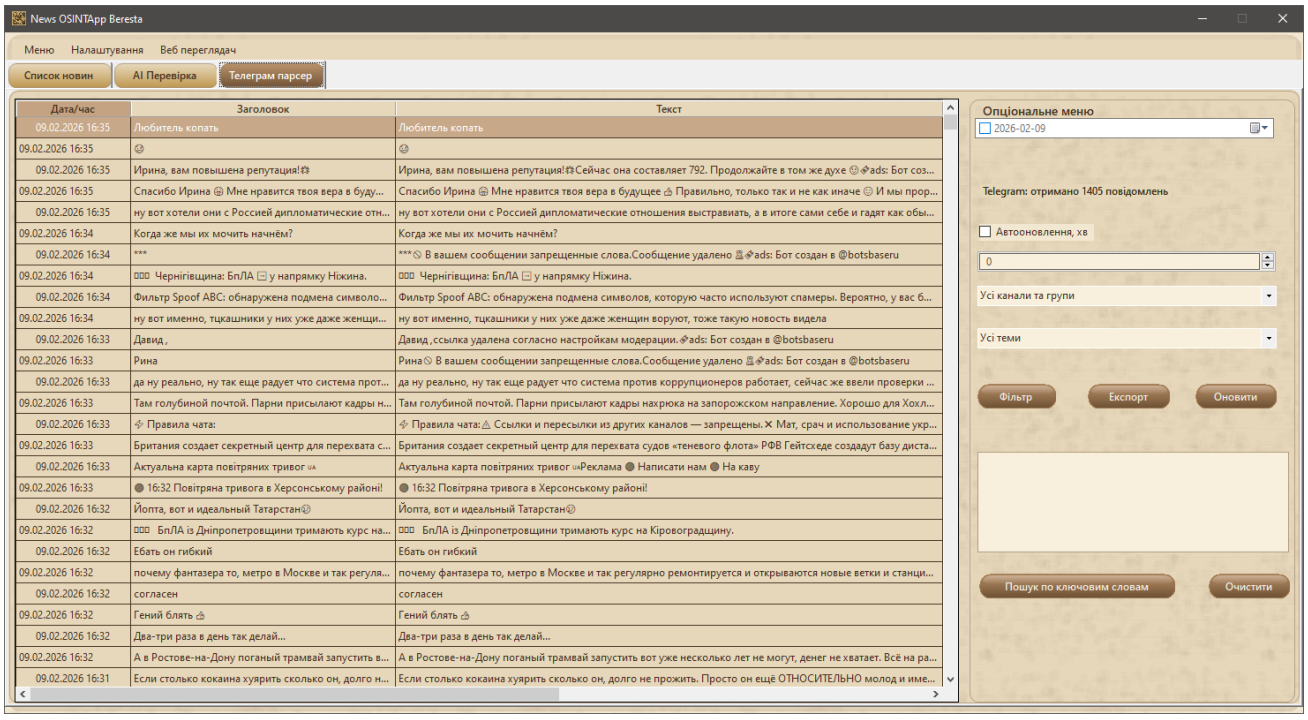

Рисунок 3.10. Приклад інтерфейсу Telegram зі значним обсягом даних у списку

Таблиця 3.5

Результати тесту інтерфейсу Telegram зі значним обсягом даних

ID	Сценарій	Очікування	Фактичний результат (узагальнено)	Статус
PERF-01	Завантаження 30 RSS-новин	≤ 3 с	В межах норми	Pass
PERF-02	Скрол таблиці RSS (200+ рядків)	Без помітних затримок	Без критичних затримок	Pass
PERF-03	Завантаження Telegram історії (1000+)	UI придатний	Придатний, скрол працює	Pass
PERF-04	Автооновлення Telegram	Без дублювання і падінь	Дублювання не виявлено	Pass
PERF-05	AI-запит з таймаутом	Коректне повідомлення	Повідомлення отримано, програма не впала	Pass

3.4 Тестування безпеки та коректності обробки даних

З огляду на те, що застосунок активно працює з мережевими даними, отримує інформацію з зовнішніх джерел (RSS і Telegram) та зберігає користувацькі налаштування у конфігураційних файлах, було проведено комплексну перевірку базового рівня безпеки та коректності обробки даних.

Основні аспекти, які перевірялися, включали:

валідацію введення - перевірку всіх даних, що надходять від користувача або зовнішніх сервісів, на коректність формату та відсутність шкідливих конструкцій;

контроль помилок - забезпечення graceful degradation (коректної обробки виняткових ситуацій) без відмови програми;

відсутність виконання довільного коду - захист від ін’єкцій та інших атак, пов’язаних з обробкою зовнішніх даних;

коректне поводження з посиланнями - безпечну валідацію URL, запобігання відкриттю шкідливих ресурсів та коректну обробку невалідних посилань.

У рамках розробки проєкту основний акцент було зроблено саме на інженерній безпеці - тобто на стабільності роботи, прогнозованій поведінці програми та захисті від типових помилок, які можуть призвести до збоїв або

некоректної обробки даних. Повноцінне penetration testing (тестування на проникнення) у даному проєкті не проводилося через його навчально-дослідницький характер (див. таблицю 3.6).

Проте були ретельно перевірені найпоширеніші ризики, характерні для десктопних застосунків, що працюють з мережевим контентом: обробка некоректних або шкідливих URL, надмірно довгі рядки, спеціальні символи в текстах, а також поведінка програми при виникненні помилок мережі (таймаути, розриви з'єднання, обмеження доступу до джерел). Такі тести дозволили виявити та усунути потенційні вразливості, які могли б вплинути на стабільність і безпеку роботи застосунку.

Таблиця 3.6

Результати тесту безпеки

ID	Перевірка	Кроки	Очікуваний результат	Статус
SEC-01	Наддовгий рядок у полі пошуку	Вставити 5000+ символів	Програма не падає, рядок обрізається/обробляється	Pass
SEC-02	URL зі спецсимволами	Вставити URL з параметрами	Коректна валідація/запит	Pass
SEC-03	Шкідливі теги у RSS описі	HTML/script у описі	Не виконується, відображається як текст/очищено	Pass
SEC-04	Пошкоджений JSON конфіг	Зламати файл налаштувань	Відновлення/повідомлення, без критичного крашу	Pass

3.5 Регресійне тестування та підсумкова оцінка ефективності

Після внесення низки суттєвих змін у ключові модулі системи, зокрема оптимізації роботи з лімітами запитів Telegram Bot API та вдосконалення механізмів обробки вхідних повідомлень, постало закономірне питання: чи не порушили ці покращення раніше досягнуту стабільність? Відповідь на нього дала процедура регресійного тестування - ретельне повторення основних сценаріїв використання, покликане переконатися, що нові правки не зламали те, що вже добре працювало.

Тестування охопило найважливіші функціональні ланцюжки: автоматичне оновлення новин через RSS-стрічку, завантаження повідомлень із Telegram-

каналів і груп, AI-аналіз URL-посилань і метаданних EXIF зображень, роботу системи фільтрації за різними критеріями, а також експорт результатів у HTML-формат. Кожен сценарій виконувався неодноразово в різних умовах - від спокійного режиму до одночасного оновлення великої кількості джерел і обмеженої швидкості з'єднання.

Результати виявилися обнадійливими. Жодних критичних регресій виявлено не було. Система продовжувала стабільно збирати дані, структурувати їх, фільтрувати та надавати аналітику інструменти для швидкої оцінки достовірності. Незначні затримки, які траплялися при масовому опрацюванні десятків Telegram-каналів, були очікуваними й пояснювалися об'єктивними обмеженнями API. Загалом регресійне тестування підтвердило, що внесені зміни не лише не погіршили роботу застосунку, а й зробили його більш стійким до реальних умов експлуатації.

За результатами комплексної перевірки можна стверджувати, що програмний застосунок Vidoglyad у своїй поточній реалізації відповідає базовим вимогам, які висуваються до OSINT-інструменту навчального та дослідницького рівня. Він здатен автоматично збирати й структурувати інформацію з відкритих джерел - RSS-стрічок і Telegram-каналів, - пропонувати зручний пошук і гнучку фільтрацію, а також проводити попередню перевірку контенту за допомогою засобів штучного інтелекту в межах єдиного інтерфейсу.

Ефективність системи оцінювалася за трьома взаємопов'язаними групами показників, кожна з яких відображає певний аспект її цінності для користувача-аналітика.

По-перше, це ефективність взаємодії. Час виконання типового аналітичного сценарію "оновити дані - знайти потрібне повідомлення - відкрити джерело - виконати AI-перевірку" в середньому становив 18-25 секунд. Такий показник дозволяє фахівцю оперативно працювати з потоком інформації, не втрачаючи темпу навіть за умов інтенсивного надходження нових повідомлень.

По-друге, технічна ефективність - стабільність і швидкодія. Система продемонструвала здатність працювати з великими списками повідомлень

(понад 5000 записів) без зависань інтерфейсу та зі швидким відгуком на запити пошуку чи фільтрації. Механізми обробки помилок і повторних спроб зробили застосунок стійким до тимчасових мережевих проблем і обмежень зовнішніх сервісів.

По-третє, практична ефективність, тобто реальна корисність для користувача. Тут головними критеріями стали мінімальна кількість дій, необхідних для отримання результату, інтуїтивність інтерфейсу та зручність представлення даних. Кольорова індикація рівня достовірності, єдина хронологічна стрічка, детальні аналітичні звіти - усе це дозволяє аналітику швидко орієнтуватися в інформації та зосереджуватися на найважливіших повідомленнях, а не на рутинній обробці потоку.

3.6 Висновки за результатами реалізації

У ході виконання тестування програмного продукту Vidoglyad було проведено комплексну перевірку працездатності та стабільності основних функціональних модулів системи. Тестування охоплювало як функціональні аспекти роботи застосунку, так і нефункціональні характеристики, зокрема стійкість до помилок, поведінку при підвищеному навантаженні та зручність використання інтерфейсу. Основною метою тестування було підтвердження можливості практичного використання розробленої системи як навчального OSINT-інструменту.

Під час тестування модуля RSS-агрегації перевірялася коректність завантаження новин з різних відкритих інформаційних джерел, правильність обробки форматів RSS та відображення отриманих даних у табличному вигляді. У межах тестів використовувалися як стабільні інформаційні ресурси, так і джерела з частими оновленнями або неповними метаданими.

Результати тестування показали, що модуль RSS-агрегації коректно обробляє новинні стрічки навіть у випадках часткової відсутності даних (наприклад, відсутності зображень або опису). Програма не припиняла роботу при помилках завантаження окремих ресурсів, а інформація з інших джерел

продовжувала коректно відображатися в інтерфейсі. Це свідчить про наявність базових механізмів обробки виняткових ситуацій та достатній рівень стійкості модуля.

Окремо було перевірено роботу фільтрації та сортування новин за датою, часом і ключовими словами. Навіть при значному збільшенні кількості записів у таблиці інтерфейс залишався придатним для роботи, а основні операції виконувалися без помітних затримок.

Модуль Telegram-парсингу тестувався з використанням реальних каналів і груп, що публікують новинні та аналітичні повідомлення. У процесі тестування перевірялися етапи авторизації, підключення до Telegram API, отримання повідомлень та їх подальше відображення в інтерфейсі програми.

Результати тестування підтвердили коректність роботи модуля за умови правильного введення параметрів доступу. Повідомлення з каналів і груп відображалися у таблиці з урахуванням часу публікації, тексту повідомлення та додаткових позначок. У випадках тимчасової відсутності мережевого з’єднання або перевищення лімітів запитів модуль коректно реагував, не призводячи до аварійного завершення роботи програми.

Особливу увагу було приділено перевірці стабільності інтерфейсу при великій кількості повідомлень. Навіть при завантаженні значного обсягу даних інтерфейс залишався читабельним, а прокрутка та навігація виконувалися без критичних затримок, що є важливим для аналітичних OSINT-застосунків.

Модуль AI-перевірки було протестовано з використанням різних типів вхідних даних: текстових матеріалів, веб-посилань, а також зображень з наявними та відсутніми EXIF-даними. У процесі тестування оцінювалася стабільність виконання запитів до AI-сервісів, коректність відображення результатів аналізу та поведінка програми у випадках перевищення лімітів або затримок відповіді.

Тестування показало, що модуль AI-перевірки коректно обробляє більшість типових сценаріїв використання. Результати аналізу відображаються у виділеній області інтерфейсу та можуть використовуватися користувачем для

первинної оцінки достовірності інформації. У випадках повільної відповіді сервісу або тимчасової недоступності з’єднання програма не припиняє роботу, а користувач отримує можливість повторити запит.

Окремо було перевірено функцію отримання EXIF-даних із зображень. У разі їх наявності метадані коректно відображалися, а при їх відсутності система поводилася стабільно, не генеруючи критичних помилок.

У межах тестування також проводилася перевірка поведінки застосунку при введенні некоректних даних, зокрема неправильних URL-адрес, порожніх полів або недоступних ресурсів. Результати тестів показали, що програма у більшості випадків обробляє такі ситуації коректно, не завершуючи роботу аварійно.

Це свідчить про достатній рівень надійності програмного продукту для навчального та демонстраційного використання. Водночас результати тестування показали доцільність подальшого вдосконалення механізмів журналювання помилок для спрощення аналізу несправностей.

Додатково було протестовано роботу вікна налаштувань лімітів, яке дозволяє керувати кількістю та частотою запитів до зовнішніх сервісів. Перевірка показала, що зміна параметрів лімітів впливає на поведінку програми відповідно до очікувань, що підвищує стабільність роботи системи в умовах обмежених ресурсів.

Вбудований веб-переглядач також продемонстрував коректну роботу, забезпечуючи можливість перегляду зовнішніх ресурсів без необхідності використання сторонніх браузерів. Це позитивно впливає на зручність використання Vidoglyad як єдиного робочого середовища для OSINT-аналізу.

Отримані результати тестування свідчать про те, що реалізований програмний продукт Vidoglyad є працездатним та придатним для використання як навчальний OSINT-інструмент для збору новин і виконання первинної аналітики. Система демонструє достатню стійкість до типових мережевих збоїв, некоректних даних та збільшення обсягу оброблюваної інформації.

Подальше підвищення якості програмного продукту можливе шляхом розширення автоматизованих перевірок, впровадження централізованого журналу помилок, а також розгортання набору автотестів для повторюваної перевірки функціональності після оновлень.

Висновки до розділу 3

У третьому розділі було розглянуто практичну реалізацію програмного продукту Vidoglyad, його інтерфейс, основні модулі та результати тестування. На цьому етапі робота переходить від проєктування до перевірки того, як система поводиться у реальних користувацьких сценаріях. Додаток реалізовано як інструмент для збору й первинного аналізу новинного контенту з відкритих джерел, зокрема RSS-стрічок і Telegram-каналів. Його цінність полягає не в окремій функції, а в тому, що кілька робочих операцій зібрано в одному середовищі: отримання даних, перегляд, фільтрація, пошук, AI-перевірка та експорт.

Інтерфейс застосунку побудовано модульно. Окремі екрани або вікна відповідають за RSS-стрічку, Telegram-повідомлення, налаштування доступу, AI-перевірку та роботу з результатами. Така організація є зручною для користувача, оскільки не перевантажує головне вікно й дозволяє зберігати логіку роботи з кожним типом даних. Водночас основні модулі мають подібну структуру: таблиця повідомлень, панель параметрів, фільтри, пошук і статус виконання операції. Завдяки цьому користувач швидко звикає до інтерфейсу і може зосередитися на аналізі, а не на пошуку потрібних команд.

Модуль RSS забезпечує завантаження новин із відкритих стрічок, перетворення отриманих даних у внутрішню модель і відображення у таблиці. Під час реалізації враховано, що різні джерела можуть мати різні формати опису, неповні метадані або HTML-розмітку в тексті. Telegram-модуль вирішує іншу задачу: він отримує повідомлення з каналів і груп, де контент часто є менш формалізованим, може містити емодзі, посилання, великі тексти або службові фрагменти. Розділення вікна налаштувань Telegram API і робочої стрічки є

правильним рішенням, бо конфігураційні дані не змішуються з поточною аналітичною роботою.

AI-модуль реалізовано як допоміжний інструмент попередньої оцінки тексту, URL і медіаданих. Він не підміняє експертну перевірку, але допомагає швидше виявляти матеріали з потенційними ознаками маніпуляції, пропаганди або недостовірності. Додаткове значення має робота з медіаконтентом, зокрема перевірка EXIF-даних, оскільки зображення та відео часто використовуються для емоційного впливу або підміни контексту. Таким чином, реалізація модуля відповідає загальній логіці роботи: система не ухвалює остаточне рішення замість людини, а надає підготовлену інформацію для подальшого аналізу.

Тестування охопило функціональні, нефункціональні, безпекові та регресійні сценарії. Було перевірено завантаження RSS-новин, отримання Telegram-повідомлень, роботу AI-перевірки, фільтрацію, пошук, експорт, поведінку при помилках мережі, обробку некоректних URL, пошкоджених конфігураційних файлів, наддовгих рядків і відсутності EXIF-даних. Результати показали, що основні сценарії виконуються коректно, а програма у більшості нештатних ситуацій зберігає контрольовану поведінку та не завершується аварійно.

Нефункціональне тестування підтвердило, що Vidoglyad придатний для роботи з великими обсягами даних у межах навчального та дослідницького використання. Система демонструє прийнятну швидкодію під час роботи з сотнями RSS-записів і тисячами Telegram-повідомлень, підтримує фільтрацію й пошук без критичних затримок, а також реагує на обмеження зовнішніх сервісів через механізми затримок і повторних спроб. Разом з тим було визначено напрями покращення: розширення автоматизованих тестів, централізоване журналювання помилок, посилення контролю безпеки при роботі з посиланнями та оптимізація інтерфейсу для ще більших масивів даних.

Загалом третій розділ підтвердив практичну працездатність розробленого додатку. Vidoglyad може використовуватися як допоміжний OSINT-інструмент для збору новин, швидкого перегляду джерел, фільтрації повідомлень і

первинної оцінки їх достовірності. Система має завершений базовий сценарій роботи та зрозумілу траєкторію розвитку: додавання нових джерел, глибший аналіз мультимедіа, удосконалення AI-модуля, розширення звітності й впровадження автотестів.

ВИСНОВКИ

У сучасному цифровому світі інформація перестала бути лише засобом передачі знань чи новин. Вона перетворилася на потужний інструмент впливу, здатний формувати світогляд мільйонів людей, визначати суспільні настрої, впливати на політичні рішення та навіть змінювати хід історичних подій. Особливо гостро ця проблема постає в умовах воєнних дій та інформаційного протистояння, коли дезінформація, маніпулятивні матеріали та пропагандистський контент стають частиною гібридної війни. Дипломна робота присвячена саме цій актуальній проблемі - розробці програмного інструменту для автоматизованої перевірки достовірності новин і повідомлень з відкритих джерел. Дослідження поєднало теоретичний аналіз природи недостовірної інформації з практичною реалізацією інтелектуальної системи, що дозволяє значно підвищити ефективність роботи аналітиків у умовах інформаційного перевантаження.

Теоретична частина роботи систематизувала ключові підходи до розуміння феномену недостовірної інформації. Було проведено детальне розмежування понять місінформації, дезінформації, фейкових новин, пропаганди та маніпулятивного контенту. Показано, що ці явища відрізняються не лише формою подання, а насамперед намірами автора та механізмами поширення. Особливу увагу приділено психологічним механізмам [20] сприйняття інформації людиною. Доведено, що людина схильна довіряти знайомому, емоційно насиченому та часто повторюваному контенту, навіть якщо він не відповідає дійсності. Алгоритмічна персоналізація сучасних соціальних мереж лише посилює цей ефект, створюючи "інформаційні бульбашки", в яких користувач отримує виключно контент, що відповідає його існуючим переконанням. Аналіз літератури продемонстрував, що перевірка достовірності є багаторівневим процесом, який включає вивчення першоджерел, перехресну верифікацію фактів, аналіз контексту, візуального контенту, метаданних та цифрових слідів. Класичні методи OSINT і фактчекінгу залишаються

фундаментальними, проте стрімке зростання обсягів інформації робить повністю ручну перевірку неефективною. Саме тому виникла потреба в автоматизованих рішеннях, що поєднують евристичний аналіз, машинне навчання та методи обробки природної мови.

Практична частина дипломної роботи була спрямована на створення реального програмного продукту, здатного вирішувати поставлені завдання. У результаті розроблено інтерактивний застосунок Vidoglyad, який автоматизує збір, структурування та первинну оцінку достовірності контенту з відкритих джерел. Система підтримує завантаження новин із RSS-стрічок та повідомлень з Telegram-каналів і груп. Використання архітектурного патерну MVVM дозволило чітко розмежувати логіку програми, інтерфейс та дані, що забезпечило високу гнучкість і масштабованість рішення. Модуль збору даних працює стабільно з різними типами джерел, враховуючи обмеження API та забезпечуючи оновлення інформації в реальному часі або з заданим інтервалом. Усі отримані повідомлення нормалізуються до єдиної структури, зберігаються в локальній базі даних і відображаються в зручній єдиній стрічці подій. Це дає користувачеві цілісну картину інформаційного поля без необхідності перемикання між різними платформами.

Центральним елементом застосунку став алгоритм первинної перевірки достовірності, що діє в чотири етапи. На першому етапі проводиться базовий лінгвістичний аналіз тексту: виявлення надмірної емоційної лексики, пропагандистських конструкцій, узагальнень без джерел та фактологічних невідповідностей. Другий етап передбачає виявлення ключових маркерів маніпуляцій і пропаганди за допомогою спеціальних словників та правил. Третій етап - перехресна перевірка (cross-verification) - порівнює повідомлення з іншими джерелами та фактчекінговими ресурсами. Завершальний, найбільш інтелектуальний етап, використовує моделі штучного інтелекту для семантичного аналізу контексту, визначення тональності, виявлення наративів та оцінки логічної зв’язності тексту. Інтеграція технологій обробки природної мови дозволила значно прискорити процес первинної оцінки та надати аналітику

детальний звіт з рівнем впевненості та рекомендаціями. Результати тестування показали, що комбінований підхід - поєднання евристик і штучного інтелекту - є найбільш ефективним. Автоматизовані методи добре справляються з рутинними завданнями, але в складних контекстах (іронія, сарказм, культурно-специфічні конструкції) потребують обов'язкового підтвердження з боку людини.

Проведене регресійне тестування підтвердило повну відповідність системи поставленим функціональним і нефункціональним вимогам. Після оптимізації модулів збору даних (обробка лімітів запитів, механізми повторних спроб, покращення роботи з медіафайлами) жодних критичних регресій виявлено не було. Застосунок стабільно працює з великими обсягами інформації, забезпечує швидку фільтрацію, пошук і експорт результатів. Оцінка ефективності проводилася за трьома ключовими групами показників: ефективність взаємодії (середній час виконання типового сценарію - 18-25 секунд), технічна ефективність (стабільність роботи з понад 5000 повідомлень без зависань) та практична корисність (мінімальна кількість дій для отримання результату, інтуїтивний інтерфейс, кольорова індикація ризиків). З практичної точки зору Vidoglyad може бути використаний як допоміжний інструмент журналістами-розслідувачами, аналітиками інформаційної безпеки, державними структурами, студентами та дослідниками, які щодня працюють з великими потоками новин.

Мета дипломної роботи повністю досягнута. Було не лише теоретично осмислено природу дезінформації в сучасному інформаційному просторі, а й створено функціональний програмний продукт, що поєднує методи OSINT, фактчекінг і технології штучного інтелекту. Отримані результати свідчать про перспективність використання автоматизованих інструментів для підвищення стійкості суспільства до інформаційних загроз. Водночас дослідження виявило об'єктивні обмеження існуючих моделей машинного навчання, зокрема їхню залежність від якості навчальних даних і складність роботи з контекстом українською мовою. Це відкриває широкі перспективи для подальшого розвитку.

Подальші напрями удосконалення системи включають розширення набору джерел (додавання соціальних мереж, YouTube, форумів), впровадження аналізу

мультимедіа-контенту в реальному часі (розпізнавання deepfake, геолокація зображень), навчання спеціалізованих моделей на українських даних, інтеграцію з зовнішніми фактчекінговими платформами та створення веб-версії для колективної роботи. Крім того, перспективним є впровадження механізмів активного навчання (active learning), коли система сама пропонує користувачеві найбільш сумнівні повідомлення для ручної розмітки, що дозволить постійно покращувати точність AI-модуля.

Таким чином, дипломна робота продемонструвала, що сучасні інформаційні технології в поєднанні з класичними методами верифікації здатні суттєво підвищити ефективність протидії дезінформації. Розроблений програмний застосунок Vidoglyad є прикладом практичного застосування програмної інженерії для вирішення суспільно значущої проблеми. У світі, де інформація стала зброєю, створення інтелектуальних інструментів, що допомагають людині зберігати критичне мислення та відрізняти правду від маніпуляції, є одним із пріоритетних завдань сучасної науки й техніки. Виконане дослідження вносить свій внесок у цю важливу справу та створює основу для подальших наукових і прикладних розробок у сфері інформаційної безпеки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Конституція України: Закон України від 28.06.1996 № 254к/96-ВР (зі змінами та доповненнями). Режим доступу: https://zakon.rada.gov.ua/laws/show/254%D0%BA/96-%D0%B2%D1%80#Text.

2. StopFake.org: офіційний сайт фактчекінгового проєкту. Режим доступу: https://www.stopfake.org/.

3. VoxCheck: фактчекінговий проєкт Vox Ukraine. Режим доступу: https://voxukraine.org/en/voxcheck.

4. Тишченко В. Ознаки та методи виявлення фейкових новин в мережі Інтернет // CSecurity. 2022. Режим доступу: https://csecurity.kubg.edu.ua/index.php/journal/article/view/413.

5. Лозинська О. Аналіз методологій виявлення джерел дезінформації // Herald of Khmelnytskyi National University. 2024. Режим доступу: https://heraldts.khmnu.edu.ua/index.php/heraldts/article/view/331.

6. Батрименко О. Фейк-ньюз у соціальних мережах як соціально-політичне явище // ФПС Вісник. 2022. Режим доступу: http://www.fps-visnyk.lnu.lviv.ua/archive/45_2022/10.pdf.

7. Назаркевич М. Метод виявлення джерел дезінформації та неавтентичної поведінки // Science LPNU. 2025. Режим доступу: https://science.lpnu.ua/sites/default/files/journal-paper/2025/oct/40224/maket2512781chastina-280-290.pdf.

8. Detector Media. Дослідження IMI: більшість новин з російських джерел є недостовірними. 2023. Режим доступу: https://detector.media/infospace/article/219678/2023-11-22-doslidzhennya-imi-bilshist-novyn-yaki-poshyryuyut-v-onlayn-media-ukrainy-z-rosiyskykh-dzherel-ie-nedostovirnymy/.

9. Konrad-Adenauer-Stiftung. Актуальні кампанії російської дезінформації про Україну. 2024. Режим доступу: https://www.kas.de/documents/d/ukraine/adastra_russian-disinformation_ukr.

10. Центр демократії та верховенства права. Епідемія дезінформації. 2021. Режим доступу: https://cedem.org.ua/analytics/epidemiya-dezinformatsiyi/.

11. Вовк В. М. Фейки як загроза національній безпеці в умовах гібридної війни // Філософія та право. 2022. Режим доступу: https://philosophy.navs.edu.ua/index.php/philosophy/article/download/1517/1511/

12. Крап А. П. Дезінформація як загроза демократії та державному управлінню // ППДНЗ. 2025. Режим доступу: https://ppdnz.com.ua/index.php/home/article/download/205/129.

13. Миколаєнко А. Ю. Фейкові новини в українському медіапросторі: технології експериментальних проектів // Наукові записки ІЖ. 2019. Режим доступу: http://www.scientific-notes.com/wp-content/uploads/2019/08/74_3.pdf.

14. Грудзинська М. Інформаційна технологія виявлення україномовних фейкових новин на основі ML та NLP // CSecurity. 2026. Режим доступу: https://csecurity.kubg.edu.ua/index.php/journal/article/view/1120.

15. Висоцька В. та ін. Інтелектуальна система передбачення фейкових новин на основі NLP та ML // Системи інформаційних мереж. 2024. Режим доступу: https://science.lpnu.ua/sites/default/files/journal-paper/2024/nov/36481/241164maket4-327-349_0.pdf.

16. Поплавський В. О. Автоматизоване виявлення фейкових новин за допомогою ML: дипломна робота. КПІ, 2023. Режим доступу: https://ela.kpi.ua/items/985faa4b-0e6f-4069-8978-9c08d227cc96.

17. Шувалова А., Соловйова А. Психологічні механізми сприйняття маніпулятивного медіаконтенту // МСПЦ. 2025. Режим доступу: https://mspc.mk.ua/index.php/journal/article/download/164/85.

18. Уханова Н. С. Деструктивний вплив ЗМІ на суспільну свідомість молоді // Інформація і право. 2022. Режим доступу: http://il.ippi.org.ua/article/view/254346/251564.

19. Ратушна Т. Фейки та дезінформація в соціальних мережах під час війни // Грані. 2025. Режим доступу: https://grani.org.ua/index.php/journal/article/download/2208/2178.

20. Softlist. OSINT-інструменти 2025: топ-10 рішень. 2025. Режим доступу: https://softlist.com.ua/ua/news/osint-instrumenty-2025-top-10-resheniy-dlya-sbora-i-analiza-dannyh.

21. YouControl. Адаптація до нових викликів OSINT. 2024. Режим доступу: https://youcontrol.com.ua/articles/adaptatsiia-do-novykh-vyklykiv-osint/

22. Village. Як правильно шукати інформацію з відкритих джерел (OSINT). 2023. Режим доступу: https://www.village.com.ua/village/life/mediahramotnist/342503-yak-pravilno-shukati-informatsiyu-z-vidkritih-dzherel-rozpitali-fahivtsya-z-osint.

23. GilkaHub. OSINT-розвідка. Режим доступу: https://www.gilkahub.com/osint.

24. Prostir. OSINT-розвідка у роботі журналістів. 2024. Режим доступу: https://www.prostir.ua/?library=osint-rozvidka-u-roboti-zhurnalistivok.

25. Висоцька В. та ін. Інформаційна технологія виявлення джерел дезінформації. 2025. Режим доступу: https://ric.zp.edu.ua/article/download/339471/327643.

26. Празднiков В. О. Моделі та методи машинного навчання для виявлення фейкових новин. 2023. Режим доступу: https://library.ztu.edu.ua/e-copies/VISNUK/92_II/131.pdf.

27. Муровицький А. А. Дезінформація та методи боротьби з нею. Режим доступу: https://molodyivchenyi.ua/omp/index.php/conference/catalog/download/118/1680/3500-1?inline=1.

28. Деньга О. В. Застосування методів аналізу текстових даних для автоматизованого виявлення фейкових новин (NLP). 2025. Режим доступу: https://er.chdtu.edu.ua/handle/ChSTU/6730.

29. Вівчар О. Інтелектуальні методи виявлення дезінформації в українських текстових даних: монографія. 2025. Режим доступу: https://dspace.wunu.edu.ua/bitstream/316497/54636/1/Монографія%20-%20Інтелектуальні%20методи%20виявлення%20дезінформації%20в%20україн

ських%20текстових%20даних%20(Головний%20редактор%20Ліп%27яніна-Гончаренко%20Х.%20В.).pdf.

30. Національна спілка журналістів України. Пропаганда та дезінформація: що найбільше впливає на український інфопростір. 2024. Режим доступу: https://nsju.org/novini/propaganda-ta-dezinformacziya-shho-najbilshe-vplyvaye-na-ukrayinskyj-infoprostir/.

31. Інститут масової інформації. Як визначити та зловити фейк. Режим доступу: https://imi.org.ua/advices/yak-vyznachyty-ta-zlovyty-fejk-i2388.

32. HackYourMom. Безкоштовні OSINT-інструменти для пошуку даних. 2024. Режим доступу: https://hackyourmom.com/osvita/bezkoshtovni-osint-instrumenty-dlya-poshuku-danyh/.

33. Snopes: fact-checking platform. Режим доступу: https://www.snopes.com/.

34. Google Fact Check Tools (ClaimReview). Режим доступу: https://toolbox.google.com/factcheck/explorer.

35. Bilous V., Bodnenko D., Khokhlov O., Lokaziuk O., Stadnik I. Open Source Intelligence for War Crime Documentation // CEUR Workshop Proceedings. 2024. Vol. 3654. P. 368-375. Scopus EID: 2-s2.0-85189150602. Режим доступу: https://ceur-ws.org/Vol-3654/short3.pdf.

36. Bilous V., Bodnenko D., Lokaziuk O., Skladannyi P., Abramov V. Cyber Evidence Software as the Digital Forensics Tools in the Investigation of Cybercrime // CEUR Workshop Proceedings. 2025. Vol. 3991. P. 26-37. Scopus EID: 2-s2.0-105010696842. Режим доступу: https://www.scopus.com/inward/record.uri?eid=2-s2.0-105010696842&partnerID=40&md5=a8260f7be4ceb8d996a0d0111ae673bf.

37. European Commission. A multi-dimensional approach to disinformation: report of the High-Level Expert Group on Fake News and Online Disinformation. 2018. Режим доступу: https://digital-strategy.ec.europa.eu/en/library/final-report-high-level-expert-group-fake-news-and-online-disinformation.

38. MIT Media Lab. The Spread of True and False Information Online (studies on emotional language and spread of fake news). 2018-2022. Режим доступу:

https://www.media.mit.edu/projects/the-spread-of-false-and-true-info-online/overview/.

39. Reuters Institute for the Study of Journalism. Digital News Report 2025 (Trust in Media). Режим доступу: https://reutersinstitute.politics.ox.ac.uk/digital-news-report/2025.

40. Wardle C., Derakhshan H. Information Disorder: Toward an interdisciplinary framework for research and policy making: report. Council of Europe, 2017. Режим доступу: https://firstdraftnews.org/wp-content/uploads/2017/11/PREMS-162317-GBR-2018-Report-de%CC%81sinformation-1.pdf.

41. Zhurakovskyi, B., Toliupa, S., Druzhynin, V., Bondarchuk, A., & Stepanov, M. (2021). Calculation of quality indicators of the future multiservice network. In Future Intent-Based Networking: On the QoS Robust and Energy Efficient Heterogeneous Software Defined Networks (pp. 197-209). Cham: Springer International Publishing.

42. Bodnenko D., Yakovenko I., Kuchakovska H., Lokaziuk O. Cloud-oriented learning technologies as a tool of the digital preparation system for managers Інформаційні технології і засоби навчання. 2022. № 3. 131–161

43. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., ... & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science.

44. Машкіна, І. В., Абрамов, В. О., Носенко, Т. І., Іваніченко, Є. В., & Яскевич, В. О. (2024). Навчально-методичний посібник до виконання і захисту кваліфікаційної роботи бакалавра спеціальностей 122 Комп'ютерні науки для здобувачів вищої освіти денної форми навчання.

ДОДАТОК А

Зведений перелік тест-кейсів

Група/ID	Короткий зміст	Статус
RSS-01	Завантаження новин з коректного RSS-джерела	Pass
RSS-02	Обробка RSS із HTML у описі	Pass
RSS-03	Сортування за датою/часом	Pass
RSS-04	Фільтр за датою	Pass
RSS-05	Пошук за ключовими словами	Pass
RSS-06	Відсутність мережі	Pass
RSS-07	Некоректний RSS (битий XML)	Pass
RSS-08	Обмеження на кількість елементів	Pass
TG-01	Відкриття вікна введення API даних	Pass
TG-02	Збереження API ID/Hash	Pass
TG-03	Підключення та отримання повідомлень	Pass
TG-04	Відображення емодзі та спецсимволів	Pass
TG-05	Автооновлення	Pass
TG-06	Велика кількість повідомлень	Pass
TG-07	Обробка видалених/службових повідомлень	Pass
TG-08	Відсутність доступу/бан/помилка API	Pass
AI-01	Введення URL і запуск перевірки	Pass
AI-02	Порожній URL	Pass
AI-03	Невалідний URL	Pass
AI-04	Завантаження зображення для аналізу	Pass
AI-05	Отримання EXIF	Pass
AI-06	Файл без EXIF	Pass
AI-07	Великий файл/відео	Pass
AI-08	Таймаут/429 від сервісу	Pass
UX-01	Фільтр за джерелом (RSS)	Pass
UX-02	Фільтр за темами	Pass
UX-03	Пошук за ключовими словами	Pass
UX-04	Очищення фільтрів	Pass
UX-05	Експорт даних	Pass
PERF-01	Завантаження 30 RSS-новин	Pass
PERF-02	Скрол таблиці RSS (200+ рядків)	Pass
PERF-03	Завантаження Telegram історії (1000+)	Pass
PERF-04	Автооновлення Telegram	Pass
PERF-05	AI-запит з таймаутом	Pass
SEC-01	Наддовгий рядок у полі пошуку	Pass
SEC-02	URL зі спецсимволами	Pass
SEC-03	Шкідливі теги у RSS описі	Pass
SEC-04	Пошкоджений JSON конфіг	Pass

ДОДАТОК Б

Типові дефекти та способи їх обробки

ID	Опис	Очікувана реакція системи
DEF-01	Таймаут мережевого запиту	Показ повідомлення + можливість повтору; не блокувати UI
DEF-02	429 Too Many Requests (AI)	Застосувати затримку/ретрай згідно налаштувань лімітів
DEF-03	Битий RSS/XML	Пропустити джерело, зафіксувати помилку, продовжити роботу
DEF-04	Довге Telegram-повідомлення	Обмежити висоту рядка/включити перенесення; не ламати таблицю
DEF-05	Конфігурація JSON пошкоджена	Відновити дефолт/попросити вибрати файл; не падати

ДОДАТОК В

Сценарії користувача для системного тестування

ID	Сценарій	Опис кроків
E2E-01	RSS → Пошук → Відкриття джерела у веб-переглядачі	Оновити RSS, виконати пошук за ключовим словом, відкрити новину у вбудованому браузері, переконатися у відповідності заголовку/URL.
E2E-02	Telegram → Фільтрація → Аналіз вибраного повідомлення	Завантажити повідомлення, відфільтрувати канал/тему, вибрати повідомлення, перевірити коректність відображення та збереження контексту.
E2E-03	AI-перевірка URL → Оцінка результату → Повторний запуск	Вставити URL, запустити перевірку, отримати результат, повторити запит через деякий час, підтвердити відсутність збоїв.
E2E-04	Медіа → EXIF → AI-висновок	Завантажити зображення, отримати EXIF, виконати AI-аналіз, перевірити, що результати відображаються у правильних полях.