**ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**
**МІСЬКОГО ГОСПОДАРСТВА ІМЕНІ О. М. БЕКЕТОВА**

Пояснювальна записка
до кваліфікаційної роботи бакалавра

на тему: Методи виявлення та запобігання порушенню авторських прав у веб-середовищі

Виконав: студент 4 курсу, групи КН 2022-1
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

Богдан ДВОРНІК
(прізвище та ініціали)

Керівник: Марія ВОЄВОДІНА
(прізвище та ініціали)

Рецензент: Олександр КОСТЕНКО
(прізвище та ініціали)

м. Харків – 2026 рік

Харківський національний університет міського господарства імені О. М. Бекетова
(повне найменування закладу вищої освіти)
Навчально-науковий Інститут енергетичної, інформаційної та транспортної інфраструктури
Кафедра комп'ютерних наук та інформаційних технологій
Рівень вищої освіти перший (бакалаврський)
Спеціальність 122 Комп'ютерні науки
(шифр і назва)

**ЗАТВЕРДЖУЮ**
Завідувач кафедри КНтаІТ

Марина НОВОЖИЛОВА
«_23___» __червня__2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ
Дворніку Богдану Валерійовичу
**(прізвище, ім'я, по батькові)**

1. Тема роботи Методи виявлення та запобігання порушенню авторських прав у веб-середовищі

керівник роботи Воєводіна М. Ю.
( прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від «22» травня 2026 р. № 440-03

2. Термін подання здобувачем роботи 15 червня 2026 р.

3. Вихідні дані до роботи Рекомендації щодо реалізації застосунку для управління бібліотечними архівними даними

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

– дослідити загальніх принципи правової та технічної бази захисту цифрового контенту;
– провести опис та порівняння існуючих алгоритмів і методів виявлення порушень авторського права з різними типами контенту (текст, медіаконтент, програмний код);
– розробити архітектуру система онлайн-моніторингу порушення авторських прав з диференційованим підходом до аналізу різнорідних цифрових об'єктів ;
– протестувати здатність системи коректно виявляти реальні факти порушення авторських прав у веб-середовищі;
– розглянути питання, пов'язані із організацією та забезпеченням охорони праці як важливої складової безпеки працівників.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація – 18 аркушів

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Розділ I | Воєводіна М.Ю., ст.викл. каф КНтаІТ | 11.05.2026 | 23.05.2026 |
| Розділ II | Воєводіна М.Ю., ст.викл. каф КНтаІТ | 24.05.2026 | 02.06.2026 |
| Розділ III | Воєводіна М.Ю., ст.викл. каф КНтаІТ | 03.06.2026 | 10.06.2026 |
| Розділ IV | Малишева В.В., доцент каф. ОПтаБЖ | 11.06.2026 | 14.06.2026 |
| | | | |
| | | | |
| | | | |

7. Дата видачі завдання 11.05. 2026 р.

**КАЛЕНДАРНИЙ ПЛАН**

| /п | Назва етапів кваліфікаційної роботи | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| | Вибір теми кваліфікаційної роботи | 11.05.2026 | |
| | Затвердження тем, наукових керівників, завдань та календарного плану підготовки кваліфікаційної роботи | 15.05.2026 | |
| | Написання I розділу | 23.05.2026 | |
| | Написання II розділу | 02.06.2026 | |
| | Написання III розділу | 10.06.2026 | |
| | Написання IV розділу | 14.06.2026 | |
| | Подання кваліфікаційної роботи керівнику | 15.06.2026 | |
| | Робота по усуненню зауважень керівника, уточнення і доповнення практичного матеріалу, оформлення додатків до роботи | 16.06.2026 | |
| | Подання доопрацьованого варіанту роботи керівнику | 16.06.2026 | |
| 0 | Захист матеріалів кваліфікаційної роботи на засіданні кафедри | 18.06.2026 | |
| 1 | Офіційний захист матеріалів кваліфікаційної роботи на засіданні Державної екзаменаційної комісії | 25.06.2026 | |

Студент _____ ( підпис ) Богдан Дворнік (прізвище та ініціали)

Керівник роботи _____ (підпис ) Марія ВОЄВОДІНА (прізвище та ініціали)"

# АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка кваліфікаційної роботи бакалавра студента групи КН 2022-1 спеціальності 122 Комп’ютерні науки Дворніка Богдана Валерійовича за темою «Методи виявлення та запобігання порушенню авторських прав у веб-середовищі» складається з 4 розділів, містить 6 рисунків, 5 таблиць, 29 джерел.

Кваліфікаційну роботу бакалавра присвячено розробці прототипа системи онлайн-моніторингу спроб порушення авторських прав, що сприятиме збереженню прав авторів в мережі інтернет.

У першому розділі описано предметне середовище, розглянуто концептуальні питання правового захисту авторів, стан в Україні, були наведені наявні аналоги та проведено їх порівняння. Також сформований об’єкт та предмет дослідження.

У аналітичному розділі описано існуючі алгоритми та методи виявлення та запобігання порушенню авторських прав в залежності від типу цифрового контенту (текст, медіа, програмний код), проведено їх порівняння.

У розділі програмного та технічного забезпечення розроблено трирівневу архітектуру системи, компонентну схему, UML-діаграму діяльності, розроблено програми для методів аналізу схожості та проведено тестування коректної роботи системи на тестових даних двох типів (текст, фото).

У розділі охорони праці визначені вимоги до організації робочого оточення та були запропоновані рекомендації щодо покращення умов праці та зниження рівня професійних ризиків.

Ключові слова: АВТОРСЬКЕ ПРАВО, ВЕБ-СЕРЕДОВИЩЕ, ЦИФРОВИЙ ПЛАГІАТ, АНТИПЛАГІАТ-СИСТЕМИ, МОНІТОРИНГ КОНТЕНТУ, МЕТРИЧНА СХОЖІСТЬ ЦИФРОВОГО КОНТЕНТУ

# SUMMARY

Structure and scope of work. The explanatory note of the bachelor's qualification work of the student of the group CS 2022-1 specialty 122 Computer Science Dvornik Bohdan Valerievich on the topic " Methods for detecting and preventing copyright infringement in the web environment " consists of 4 sections, contains 6 figures, 5 tables, 29 sources.

The bachelor's qualification work is dedicated to the development of a prototype of an online monitoring system for attempted copyright infringement, which will help preserve the rights of authors on the Internet.

The first section describes the subject environment, considers conceptual issues of legal protection of authors, the situation in Ukraine, presents existing analogues and compares them. The object and subject of the study are also formed.

The analytical section describes existing algorithms and methods for detecting and preventing copyright infringement depending on the type of digital content (text, media, program code), and compares them.

In the software and hardware section, a three-level system architecture, a component diagram, a UML activity diagram were developed, programs for similarity analysis methods were developed, and the correct operation of the system was tested on test data of two types (text, photo).

The section of labor protection defines the requirements for the organization of the working environment, taking into account harmful and dangerous production factors.

Keywords: COPYRIGHT, WEB ENVIRONMENT, DIGITAL PLAGIARISM, ANTI-PLAGIAMENT SYSTEMS, CONTENT MONITORING, DIGITAL CONTENT SIMILARITY METRIC.

ЗМІСТ

# ВСТУП

У сучасному цифровому світі обсяги контенту, що публікується в інтернеті, зростають лавиноподібно. Таке стрімке зростання обсягів цифрового контенту – це глобальний вибух інформації, спричинений розвитком штучного інтелекту, соціальних мереж та мобільних пристроїв. Це явище докорінно змінює те, як ми створюємо, зберігаємо та споживаємо дані, вимагаючи нових підходів до фільтрації та управління інформацією[1].

Причинами та рушійними силами цього процесу є наступні причини:

- Перехід на відеоформати. Переважну частку контенту зараз становлять короткі відео (TikTok, Instagram Reels, YouTube Shorts), що витісняють довгі тексти.
- Розповсюдження генеративного штучного інтелекту (ШІ). Нейромережі дозволяють створювати тексти, зображення, аудіо та відео за лічені секунди, швидко примножуючи загальний обсяг даних.
- Глобальна діджиталізація. Розвиток хмарних технологій, IoT (Інтернету речей) та онлайн-бізнесу.

Слід зазначити, що ключові наслідки є дуже неоднозначними і можуть мати серйозний продовжений вплив на різні сфери життя: освіту, культуру, бізнес, інтелектуальний розвиток суспільства, тощо.

Виникає, так звана проблема "цифрового сміття". Левова частка згенерованого контенту є дублікатами або не має практичної цінності. З іншого боку, надмірне зберігання даних на серверах має негативний вплив на екологію та енергетичну інфраструктуру.

Відбувається зміна уваги аудиторії. Через перевантаження інформацією (Information Overload) користувачі стали гірше фокусуватися, що стимулює подальше скорочення окремих одиниць контенту, і проблема тільки поглиблюється.

Зростання обсягів має і економічний вплив, воно диктує нові правила для ринку цифрового маркетингу. Успіх компаній зараз залежить від правильного просування, SEO та аналітики на фоні високої конкуренції[1,2].

І, звичайно, виклики для безпеки. Збільшення мультимедійних даних загострює питання захисту авторських прав в онлайн-середовищі.

Для успішного функціонування в умовах перенасичення, бізнеси та окремі користувачі змушені оптимізувати свої стратегії: робити ставку на персоналізацію, підвищувати якість замість кількості та активно впроваджувати інструменти автоматизації для аналізу трендів[2,3].

Разом з цим зростає кількість порушень авторських прав: копіювання, несанкціоноване поширення, плагіат тощо. Плагіат є глобальною проблемою академічної та професійної сфер, яка стрімко загострюється через доступність генеративного штучного інтелекту. Це призводить до зниження стандартів освіти, порушення авторських прав і знецінення оригінальних досліджень[3,4].

За законами про авторське право ШІ не копіює творчий продукт, а лише використовує його методи та техніки для створення нового. По суті, вважається, що доробок ШІ є унікальним, адже це не точна копія оригіналу. Теж саме стосується і зображень – внесок автора має бути більшим за роботу нейромережі.

Насправді ж серія нещодавніх досліджень показала, що великі мовні моделі від OpenAI, Google, Meta, Anthropic та xAI запам'ятовують набагато більше навчальних даних, ніж вважалося раніше, фактично дослівно відтворюють твори, на яких навчаються. Тим самим це вже не можна вважати «добросовісним використанням» творів, що захищені авторським правом.[5,6]

Для комп’ютерних наук актуальним є дослідження технічних методів виявлення та запобігання таким порушенням, адже вони є основою для автоматизованих систем захисту контенту.

# РОЗДІЛ 1 ЗАГАЛЬНІ ПОЛОЖЕННЯ

Стрімкий розвиток веб-технологій, міграція бізнесу в онлайн-простір та поява генеративного штучного інтелекту (Generative AI) призвели до безпрецедентного зростання обсягів цифрового контенту. Водночас масштаби порушень авторських прав у веб-середовищі досягли критичного рівня. Незаконне копіювання текстових матеріалів, несанкціоноване використання мультимедійного контенту, хотлінкінг та плагіат програмного коду завдають суттєвих фінансових та репутаційних збитків правовласникам[7,8].

Традиційні юридичні методи захисту інтелектуальної власності часто виявляються неефективними через транскордонну природу Інтернету та високу швидкість поширення інформації. У зв'язку з цим виникає гостра потреба у розробці та впровадженні автоматизованих, високопродуктивних технічних методів виявлення та запобігання порушенням авторських прав. Створення ефективних алгоритмів та програмних інструментів для моніторингу веб-простору, криптографічного маркування та блокування несанкціонованого доступу є актуальним науково-практичним завданням для сучасної IT-індустрії.

## 1.1 Аналіз діяльності та опис предметного середовища

Щосекунди в мережі створюється і передається близько 29 терабайтів інформації. Загальний обсяг інформації, що зберігається та передається, подвоюється кожні 2 роки (рис.1.1)[1].

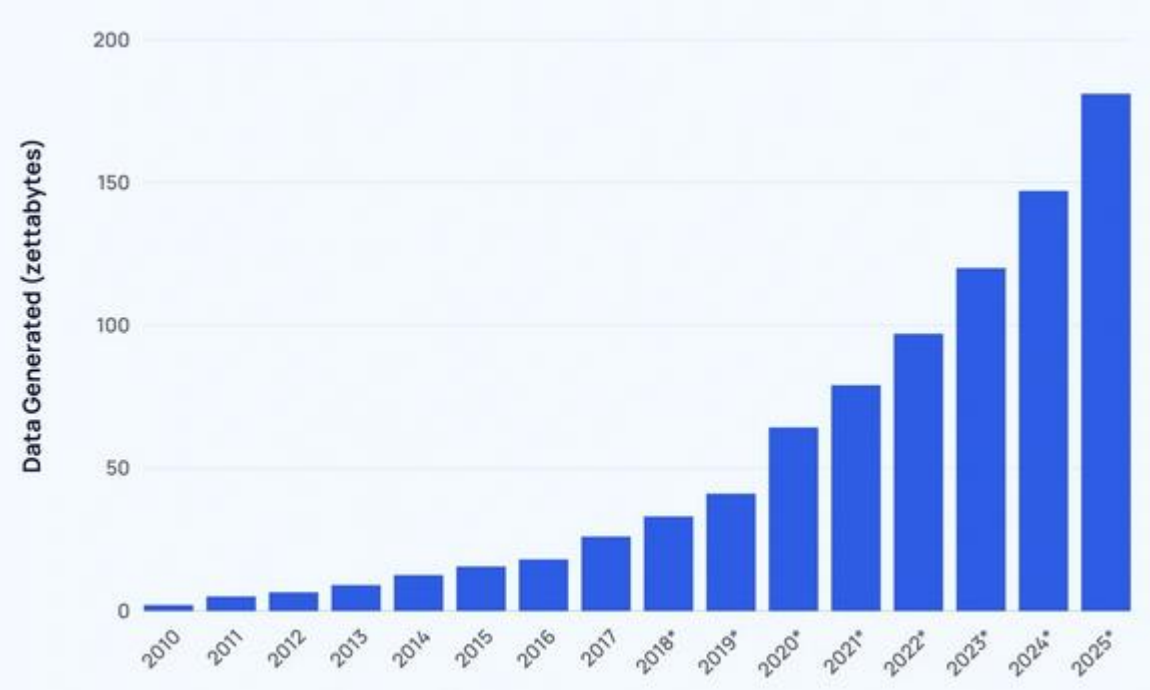


Рисунок 1.1 – Історична динаміка зростання обсягів інформації в мережі (зеттабайт = 1 трильйон гігабайт)

Через надстрімкий ріст обсягів інформації, який часто називають "інформаційним вибухом", користувачам стає дедалі складніше знаходити дійсно релевантні, якісні та достовірні дані. Ця ситуація кардинально змінює підходи до створення та споживання контенту:

- Перехід до персоналізації. Алгоритми соціальних мереж та пошукових систем (зокрема, еволюція Google Search) намагаються фільтрувати "інформаційний шум", пропонуючи користувачам матеріали, що відповідають їхнім інтересам та попереднім запитам.
- Ціна уваги. Оскільки контенту стало нескінченно багато, найціннішим ресурсом для авторів та брендів тепер є саме увага читача. Це вимагає створення більш лаконічних, структурованих та інтерактивних матеріалів.
- Фокус на якості, а не кількості. Експертність, достовірність та авторитетність стають головними критеріями, за якими алгоритми оцінюють та ранжують публікації.

- Роль ШІ. Генеративний штучний інтелект стає як інструментом для швидкого створення контенту, так і основою для нових пошукових систем, що намагаються синтезувати відповіді на запити миттєво, економлячи час користувача.

Зважаючи на цю реальність, головним завданням для творців контенту сьогодні є не просто публікація великих обсягів тексту, а створення чіткої, структурованої та корисної інформації, яка вирішує конкретні проблеми аудиторії.

Фіксація авторства цифрових та інших творів - це публікація сайту в інтернеті на публічному домені, це є закріпленням факту його створення. Якщо він буде доступний для сканування пошуковими системами, то можна буде без проблем довести факт та час його створення. Причому, навіть якщо згодом сайт зникне, а за статистикою протягом 3-х років 30% сайтів, що не мають друкованих аналогів, стають недоступними.

Вирішувати проблеми збереження інформації в інтернеті допомагають сервіси, що називаються вебархівами. Вони зберігають у себе копії сайтів, просканованні пошуковими системами. Там є можливість побачити його збережені копії на ту чи іншу дату. Вебархіви зберігають не всі сайти, іноді не повністю та не за будь-який проміжок часу, і таке збереження теж є проблемою, тому що не всі вебархіви убезпечені від підробок [6].

## 1.2 Міжнародно-правова база захисту цифрового контенту

Сучасна архітектура цифрового авторського права базується на Договорі ВОІВ про авторське право (WIPO Copyright Treaty) 1996 року. Цей документ адаптував Бернську конвенцію до умов Інтернету. Головним нововведенням стало зобов'язання країн-учасниць забезпечити юридичний захист від обходу технічних засобів (наприклад, DRM та криптографічного шифрування). На основі цього договору США та Європейський Союз сформували свої

національні законодавчі системи, які сьогодні де-факто регулюють увесь глобальний веб-простір[7,8].

### 1.2.1 Закон США про авторське право в цифрову епоху (DMCA)

Digital Millennium Copyright Act (DMCA), ухвалений у 1998 році, є федеральним законом США. Його ключовим елементом для веб-середовища є положення Safe Harbor («Безпечна гавань»).

- Суть концепції: Інтернет-провайдери, хостинги та великі веб-платформи (наприклад, YouTube, GitHub, AWS) звільняються від юридичної відповідальності за порушення авторських прав їхніми користувачами.
- Умова звільнення: Платформа повинна оперативно реагувати на скарги правовласників за процедурою Notice and Takedown.
- Алгоритм процедури: Правовласник надсилає офіційне сповіщення (DMCA Notice) про порушення. Платформа зобов'язана негайно заблокувати або видалити цей контент. Користувач, який завантажив файл, має право надіслати зустрічне сповіщення (Counter-Notice), якщо вважає блокування помилковим. Якщо протягом 10-14 робочих днів автор не подає позов до суду, контент відновлюється.
- Глобальний ефект: Оскільки ключові IT-гіганти зареєстровані в США, процедура DMCA діє на користувачів та розробників з усього світу[7,9].

### 1.2.2 Директива ЄС про авторське право на Єдиному цифровому ринку

Ухвалена у 2019 році Директива 2019/790 (EU Copyright Directive) суттєво посилила правила у порівнянні з DMCA. Найбільш резонансною є Стаття 17 (раніше Стаття 13).

- Зміна відповідальності: Директива скасовує безумовну «безпечну гавань» для великих комерційних платформ. Тепер такі сервіси, як YouTube чи TikTok, несуть пряму відповідальність за контент, який завантажують користувачі.

- Превентивні фільтри: Платформи зобов’язані впроваджувати автоматизовані системи превентивного моніторингу та фільтрації (на зразок Content ID), щоб піратський контент взагалі не міг потрапити в мережу.
- Винятки: Суворі правила не поширюються на некомерційні платформи (наприклад, Вікіпедія) та молоді стартапи з малим обігом коштів.

## 1.3 Законодавство України у сфері захисту авторських прав в Інтернеті

Україна повністю адаптувала свою нормативну базу до європейських та міжнародних стандартів, ухваливши нову редакцію Закону України «Про авторське право і суміжні права» (набув чинності у 2023 році)[10].

Процедура припинення порушень (Український аналог DMCA)

Закон України спільно із Законом «Про державну підтримку кінематографії в Україні» регламентує чітку позасудову процедуру видалення піратського контенту в мережі. Вона має назву камдаун (Takedown) і працює за жорстким часовим графіком (рис. 1.2):

1. Звернення до власника сайту: Обмежити доступ до контенту може лише суб'єкт авторського права через сертифікованого адвоката. Власник сайту зобов'язаний заблокувати доступ протягом 48 годин з моменту отримання заяви.
2. Звернення до хостинг-провайдера: Якщо власник сайту ігнорує запит або його контакти приховані, правовласник звертається безпосередньо до постачальника послуг хостингу. Хостинг-провайдер також має 48 годин на повне блокування сайту чи сторінки.
3. Санкції: Невиконання хостингом або власником сайту цих вимог тягне за собою накладення значних адміністративних штрафів.

Порівняння законів різних країн представлено в таблиці 1.1.

Таблиця 1.1 – Порівняльний аналіз правових систем регулювання цифрового контенту

| Критерій порівняння | Модель США (DMCA) | Модель ЄС (Directive 2019/790) | Модель України (Закон № 2811-IX) |
|---|---|---|---|
| Базовий принцип | Реактивний (Notice and Takedown). | Превентивний (Пряма відповідальність платформ). | Реактивно-швидкий (Позасудове блокування). |
| Відповідальність провайдера | Відсутня, якщо запит виконано вчасно (Safe Harbor). | Присутня, якщо не доведено впровадження фільтрів захисту. | Відсутня, якщо запит хостингом виконано за 48 годин. |
| Хто ініціює блокування | Будь-який заявник (автор або представник). | Автоматичні алгоритми платформи + правовласники. | Тільки суб'єкт авторського права через адвоката. |
| Час на реакцію | Не визначено чітко («expeditiously» – оперативно). | Миттєво (на етапі спроби завантаження файлу). | Жорстко обмежений – 48 годин. |

Хоча юридично DMCA діє лише в США, його архітектура стала міжнародним стандартом «за замовчуванням». Попри локальний статус, DMCA задав глобальні стандарти через дві головні концепції, які запозичили інші країни[11]:

1. Безпечна гавань (Safe Harbor): Звільнення платформ від відповідальності за піратство користувачів, якщо вони швидко реагують на скарги.
2. Заборона обходу DRM (Digital Rights Management): Криміналізація інструментів для зламу цифрового захисту (ліцензійних ключів, шифрування тощо).

Алгоритм дії відповідних законів на рисунку 1.2.

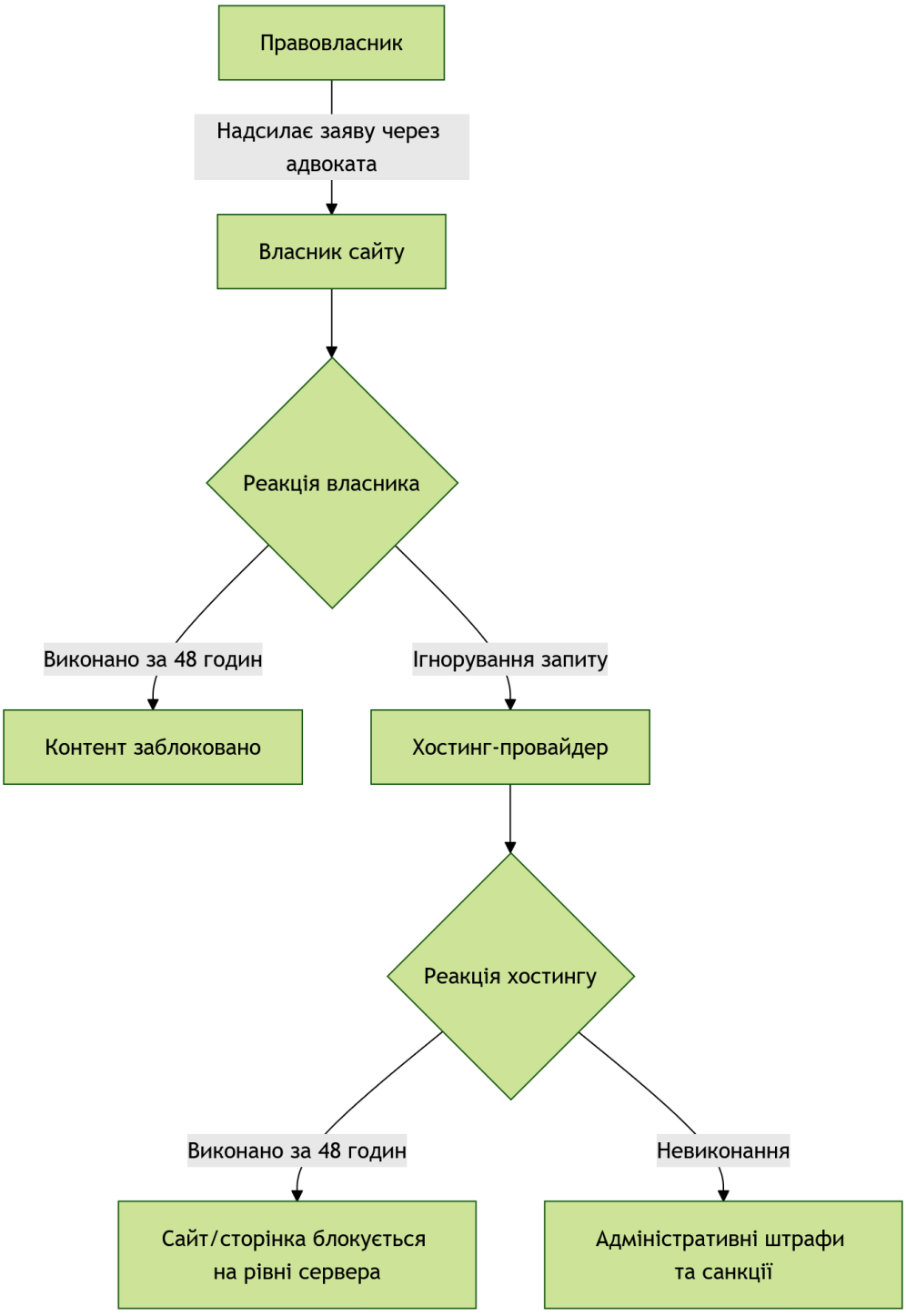


Рисунок 1. 2 – Алгоритм процедури припинення порушень

Міжнародний та національний правові контексти демонструють чітку тенденцію: юридична відповідальність за поширення піратського контенту все більше перекладається на технічних операторів та власників веб-ресурсів.

## 1.4 Технічний контекст та класифікація об'єктів веб-захисту

З інженерної точки зору, веб-середовище є розподіленою клієнт-серверною системою, де захист інформації ускладнюється відкритістю протоколів передачі даних (HTTP/HTTPS) та публічним доступом до вихідного коду клієнтської частини (HTML, CSS, JavaScript).

Для побудови архітектури захисту об'єкти інтелектуальної власності у Web поділяють на три технічні категорії:

1. Дискретні текстові масиви (статті, документація, бази даних). Захист ускладнюється використанням автоматизованих скриптів-парсерів (web scrapers).
2. Мультимедійні бінарні потоки (статичні зображення, аудіо- та відеоконтент). Захист потребує високих обчислювальних потужностей для криптографічного шифрування «на льоту» (DRM) та маркування.
3. Вихідний програмний код (скрипти, стилі, клієнтські фреймворки). Оскільки JavaScript виконується на стороні браузера користувача, він є повністю відкритим для копіювання, що потребує застосування методів обфускації та мініфікації.

Для оцінки ефективності розроблюваного програмного комплексу та визначення його функціональних вимог було проведено аналіз існуючих комерційних та відкритих систем перевірки унікальності тексту, які представлені на світовому та вітчизняному ринках.

1. ContentWatch (комерційний веб-сервіс)

- Опис: Популярний онлайн-інструмент для експрес-перевірки унікальності текстового контенту сайту.
- Технічні особливості: Працює через власний веб-інтерфейс, використовує алгоритми пошуку точних та модифікованих копій у пошуковій видачі (Google, Yandex).

- Переваги: Висока швидкість обробки невеликих текстів, зручне підсвічування знайдених збігів.
- Недоліки: Жорсткі обмеження безкоштовної версії (до 3 перевірок на день), закритий вихідний код, відсутність зручного мобільного або API-інтерфейсу для автоматизації без придбання дорогої підписки.

2. Copyleaks (платформа на базі ШІ)

- Опис: Хмарна система корпоративного рівня для виявлення плагіату та контенту, створеного штучним інтелектом.
- Технічні особливості: Використовує складні NLP-моделі (машинне навчання) для семантичного аналізу та перевірки вихідного коду програм.
- Переваги: Висока точність виявлення глибокого рерайтингу, розгалужене API, підтримка багатьох мов.
- Недоліки: Висока вартість інтеграції для малого бізнесу та стартапів, надлишкова складність інтерфейсу для швидкої перевірки окремих статей, великі затримки (latency) при обробці черги запитів.

3. PlagiarismChecker.bot (Telegram-аналог)

- Опис: Прості мобільні рішення у вигляді чат-ботів, що виконують парсинг або порівняння тексту за запитом користувача.
- Технічні особливості: Зазвичай виступають лише посередниками (обгортками), які перенаправляють запити користувача на сторонні платні API (наприклад, Text.ru або Advego).
- Переваги: Доступність з мобільних пристроїв, відсутність потреби у реєстрації на сайтах.
- Недоліки: Повна залежність від сторонніх сервісів, низький рівень безпеки та конфіденційності (текст користувача передається третім особам), неможливість кастомізації алгоритму під власну базу еталонних документів компанії.

## 1.5 Порівняльна характеристика аналізованих систем

Для наочності результати аналізу зведено у кваліметричну таблицю.

Оцінювання характеристик проведено за 3-бальною шкалою (0 – відсутній/незадовільний рівень, 1 – середній/обмежений, 2 – високий/повний).

Таблиця 1.2 –Порівняльна характеристика існуючих аналогів

| Критерій порівняння | Content Watch | Copy leaks | Plagiarism Checker.bot | Розроблювана система |
|---|---|---|---|---|
| Доступність API та автоматизації | 1 | 2 | 0 | 2 |
| Кросплатформність (Telegram) | 0 | 0 | 2 | 2 |
| Швидкість обробки запиту (Latency) | 2 | 1 | 1 | 2 |
| Автономність (власна база даних) | 0 | 2 | 0 | 2 |
| Економічна доступність (Open-Source) | 1 | 0 | 1 | 2 |
| Конфіденційність даних | 1 | 1 | 0 | 2 |
| Загальна сума балів | 5 | 6 | 4 | 12 |

Аналіз існуючих аналогів показав, що на ринку присутній технологічний розрив між складними, дорогими корпоративними системами (Copyleaks) та простими, але залежними рішеннями без власної логіки (більшість Telegram-ботів).

Обґрунтування доцільності розробки:

1. Локальний контроль даних: Розроблювана система використовує власне аналітичне ядро на базі алгоритму шинглів. Тексти не передаються стороннім компаніям, що забезпечує повну конфіденційність (критично для

комерційних фірм, які захищають свої внутрішні документи). Локальний контроль даних суттєво підвищує безпековий аспект проєкту.

2. Гнучкість архітектури: Поєднання Core Engine на Python з мінімалістичним інтерфейсом дозволяє отримати швидкість комерційного сервісу та мобільну доступність без витрат на хостинг складних веб-інтерфейсів.

3. Оптимізація під власну базу: Система порівнює тексти з джерелом еталонного контенту, що усуває потребу у платному парсингу пошукових систем для внутрішніх потреб захисту авторського права.

## 1.5 Формулювання задач дослідження та актуальність розробки

Об’єкт дослідження: веб-середовище та цифровий контент (тексти, зображення, відео, аудіо).

Предмет дослідження: технічні методи виявлення та запобігання порушенню авторських прав.

Ціль роботи: проаналізувати існуючі технічні підходи, оцінити їхню ефективність та запропонувати шляхи покращення. Розробити прототип системи онлайн-моніторингу порушення авторських прав в мережі інтернет.

Для досягнення цілей треба вирішити наступні завдання:

- Зробити огляд предметної області.
- Ознайомитись з нормативно-правовою базою захисту цифрового контенту.
- Дослідити існуючі технічні методи виявлення порушень авторських прав у веб-середовищі.
- Зробити порівняльний аналіз їхньої ефективності.
- Розробити прототип системи онлайн-моніторингу порушення авторських прав для виявлення порушень в контексті типу контенту.

- Оцінити ефективність запропонованого рішення на тестових даних.

Висновки до розділу

В першому розділі кваліфікаційної роботи представлено:

аналіз діяльності та предметної області;

розглянуто і проведенно порівняння законів проавторське право в різних країнах;

проаналізовано існуючі системи виявлення порушень авторського права для різних типів цифрового контенту, зроблено їх порівняння;

обґрунтовано доцільність розробки, сформульовано цілі і задачі дослідження.

Також сформульвано об’єкт та предмет дослідження.

# РОЗДІЛ 2 АНАЛІТИЧНИЙ РОЗДІЛ

## 2.1 Аналіз предметної області

Стрімкий розвиток веб-середовища та спрощення процесів обміну інформацією призвели до масштабування проблеми несанкціонованого копіювання, плагіату та цифрового піратства. Ефективна протидія цим явищам неможлива виключно у правовому полі, оскільки юридичні механізми захисту авторського права мають переважно реактивний характер і застосовуються вже після фіксації факту правопорушення. Відтак, першочерговим завданням забезпечення інтелектуальної власності в Інтернеті є розробка та впровадження надійних технічних методів і програмних технологій для автоматизованого виявлення дублікатів контенту.

Сучасний інструментарій моніторингу цифрового простору базується на розгалуженому математичному апараті та алгоритмах аналізу даних[12]. У сучасному веб-середовищі автоматизоване виявлення плагіату базується на обробці трьох основних типів даних: текстового контенту, мультимедіа (зображення, аудіо, відео) та вихідного коду програмного забезпечення.

Залежно від типу контенту, архітектура антиплагіат-систем використовує різні підходи до обробки інформації. Основним технологічним викликом при цьому є потреба у балансуванні між обчислювальною складністю алгоритмів та точністю розпізнавання модифікованого (рерайтнутого, масштабованого чи закодованого) контенту.

Метою цього розділу є комплексний техніко-технологічний аналіз фундаментальних алгоритмів, що становлять основу сучасних систем виявлення плагіату та несанкціонованого копіювання. У межах розділу буде досліджено математичні моделі обробки текстових масивів, проведено порівняльну оцінку їхньої ефективності за критеріями швидкодії та стійкості

до навмисних спотворень (обфускації), а також визначено оптимальні сценарії їхньої інтеграції у локальні репозиторії для захисту авторських прав без залучення сторонніх комерційних платформ.

Нижче наведено детальний аналіз математичних моделей та алгоритмів, що застосовуються для розв'язання цих задач.

### 2.1.1 Функціонал виявлення та запобігання порушенню авторських прав

Функціонал системи виявлення та запобігання порушенню авторських прав залежить від конкретного запиту:

- Автор бажає зробити публікацію свого матеріалу має довести авторство через процедуру перевірки на плагіат.
- Автор опублікованого матеріалу бажає виявити випадки його несанкціонованого використання і вжити певні заходи проти порушників, якщо вони будуть виявлені.

В кваліфікаційній роботі розглядається саме другий варіант: створення онлайн ресурсу, який за допомогою ефективних алгоритмів буде виявляти випадки порушення авторських прав в мережі для конкретних випадків.

## 2.2 Огляд існуючих алгоритмів та методів виявлення та запобігання порушень авторського права при роботі з текстом

### 2.2.1 Алгоритм шинглів (Shingle Algorithm) та коефіцієнт Жаккара

«Алгоритм черепиці (shingles)» – це поширена техніка обробки природної мови (NLP), розроблена для виявлення дублікатів тексту та плагіату. Алгоритм шинглів є класичним методом визначення ступеня схожості веб-документів на основі $N$-грамного аналізу[13,14].

Він працює, розбиваючи текст на послідовності слів (шинглів), що накладаються, і порівнянні відсотка спільних шинглів між документами для

розрахунку їхньої схожості.

- Канонізація: прибирають HTML-теги, пунктуацію, великі літери та часто поширені стоп-слова, отримують «сирий» рядок тексту.
- Розбиття на шингли (черепиці): текст поділяється на накладені послідовності N слів (токенів), де N – це «розмір черепиці».
- Хешування: Щоб заощадити пам'ять і значно пришвидшити порівняння, кожна послідовність слів конвертується у числове значення (хеш).
- Вибірка та порівняння: Замість порівняння кожного окремого хешу, алгоритми часто вибирають випадкову вибірку цих значень. Потім порівнюються множини хешів із двох різних текстів, щоб визначити їхнє співвідношення схожості. Схожість двох текстів A та B визначається за допомогою коефіцієнта подібності Жаккара (Jaccard Similarity Index):

$$J(A,B) = \frac{|H(A) \cap H(B)|}{|H(A) \cup H(B)|}$$

де H(A) та H(B) – множини хешів шинглів для текстів A та B відповідно;

$|H(A) \cap H(B)|$ – потужність (кількість елементів) перетину множин (спільні шингли);

$|H(A) \cup H(B)|$ – потужність об'єднання множин (всі унікальні шингли з обох текстів).

Межі значень:

J(A, B) = 1 (100%) – множини абсолютно ідентичні.

J(A, B) = 0 (0%) – множини не мають жодного спільного елемента.

«Алгоритм черепиці (shingles)» використовується для:

- Виявлення плагіату: Сканери використовують це для виявлення випадків, коли абзаци або цілі есе були скопійовані та переставлені.
- Оптимізація для пошукових систем (SEO): допомагає вебсайтам виявляти дублікатний контент і уникати штрафів з боку пошукових систем.

- Управління документами: Великі бази даних використовують «Алгоритм черепиці (*shingles*)» для кластеризації пов'язаних документів або фільтрації дублювання веб-сторінок.

2.2.2 Метрика Левенштейна (редакційна відстань) – це метрика, яка визначає ступінь відмінності між двома послідовностями символів (словами або рядками) [stem-calculative-problem-solving]. Вона дорівнює мінімальній кількості односимвольних операцій, необхідних для перетворення одного рядка в інший[15].

Ця метрика є ідеальним доповненням до алгоритму шинглів. Якщо шингли шукають перестановку великих блоків тексту, то відстань Левенштейна виявляє дрібний рерайт, заміну літер (микроплагіат), маскування слів та використання синонімів.

Математично відстань Левенштейна між двома рядками $S_1$ (довжиною M) та $S_2$ (довжиною N) обчислюється через динамічне програмування за допомогою рекурентного співвідношення:

$$D(i,j) = \min \begin{Bmatrix} D(i-1,j)+1 & (\text{видалення}) \\ D(i,j-1)+1 & (\text{вставка}) \\ D(i-1,j-1)+m\left(S_{1[i]},S_{2[j]}\right) & (\text{заміна}) \end{Bmatrix}$$

де $m(S_1[i], S_2[j]) = 0$, якщо символи однаві, і 1, якщо вони різні.

Використання метрики для виявлення порушень авторських прав.

Боротьба зі спуфінгом та обходом антиплагіату: Порушники часто замінюють українські літери схожими за виглядом англійськими (наприклад, «о», «е», «а», «і»). Відстань Левенштейна миттєво підсвітить таку аномалію, оскільки з погляду кодування це абсолютно різні символи.

Аналіз вихідного коду (Платні програми/Скрипти): Якщо розробник

скопіював чужий код і просто змінив назви змінних, метрика Левенштейна покаже дуже високий відсоток схожості між рядками коду.

Виявлення кіберсквотингу: Захист брендів від реєстрації схожих доменних імен. Наприклад, якщо оригінальний сайт – google.com, а зловмисник створив g00gle.com або goog1e.com, алгоритм зафіксує мінімальну редакційну відстань, що є технічним доказом наміру введення користувачів в оману.

2.2.3 Косинусна схожість

Косинусна схожість (Cosine Similarity) – це метрика, яка оцінює ступінь схожості між двома векторами у багатовимірному просторі, вимірюючи косинус кута між ними [stem-calculative-problem-solving].

На відміну від індексу Жаккара та відстані Левенштейна, косинусна схожість оцінює семантичний та частотний склад тексту. Вона ідеально працює для порівняння документів різної довжини (наприклад, великої статті та короткої тези)[16,17,18].

Тексти перетворюються на вектори, де кожен унікальний токен (слово або шингл) є окремим виміром, а координатою є частота появи цього слова (метод TF-IDF або Bag of Words).

Формула обчислення косинуса кута $\theta$між двома векторами A та B:

$$\text{Cosine Similarity} = \cos(\theta) = \frac{\left(\vec{A} \cdot \vec{B}\right)}{\left\|\vec{A}\right\| \left\|\vec{B}\right\|} = \frac{\left(\sum_{i=1}^{n} A_i B_i\right)}{\sqrt{\left(\sum_{i=1}^{n} A_i^2\right)} \cdot \sqrt{\left(\sum_{i=1}^{n} B_i^2\right)}}$$

де $\vec{A} \cdot \vec{B}$ – скалярний добуток векторів,

$\left\|\vec{A}\right\| \left\|\vec{B}\right\|$ – евклідові норми (довжини) векторів.

Інтерпретація результату:

- 1 (кут 0°): Вектори збігаються за напрямком. Тексти мають однакові пропорції слів (абсолютна схожість).

- 0 (кут 90°): Вектори ортогональні. Тексти не мають жодного спільного слова.

Метрика косинусної схожості демонструє стійкість до зміни об'єму матеріалу: якщо зловмисник візьме чужий абзац тексту і розбавить його своїми п'ятьма абзацами «води», індекс Жаккара різко впаде до критичного мінімуму. Косинусна схожість все одно зафіксує аномальну схожість векторів, оскільки напрямок вектора у просторі ключових слів зміниться незначно.

Косинусна схожість – це база для сучасних нейромережевих моделей (BERT, OpenAI Embeddings). Нейромережі переводять тексти у вектори за змістом, а порівнюють їх між собою саме через косинусну відстань.

## 2.3 Алгоритми та методи роботи з фото, аудіо та відеоматеріалами

### 2.3.1 pHash

pHash (Perceptual Hash / Перцептивний хеш) – це один із найефективніших алгоритмів комп'ютерного зору, що використовується для створення «цифрових відбитків» (fingerprints) медіаконтенту (зображень, аудіо, відео) з метою виявлення порушень авторських прав[19].

Призначений для захисту графічного та мультимедійного контенту. На відміну від тексту, медіафайли дуже легко модифікувати (змінити розмір, обрізати краї, накласти водяний знак), що повністю ламає класичні криптографічні хеші (як-от MD5 чи SHA-256). pHash вирішує цю проблему, оскільки оцінює структуру контенту, яку бачить людське око, а не конкретні байти файлу.

Наведемо покроковий алгоритм роботи pHash (для зображень). Процес генерації перцептивного хешу складається з 5 основних інженерних кроків:

1. Зменшення роздільної здатності: Зображення стискається до фіксованого розміру, зазвичай 32×32 пікселі. Це прибирає дрібні шуми та деталі, залишаючи лише загальну структуру.

2. Канонізація кольору (Grayscale): Зображення переводиться у відтінки сірого (один канал яскравості замість трьох каналів RGB). Людський мозок розпізнає об'єкти за формами та контрастом, а не лише за кольором.

3. Дискретне косинусне перетворення (DCT – Discrete Cosine Transform): Зображення переводиться з просторової області у частотну. DCT розбиває картинку на сукупність частот.

4. Виділення низьких частот: Оскільки основна візуальна інформація (форми, великі об'єкти) лежить у низькочастотному спектрі, алгоритм бере лише верхній лівий квадрат матриці DCT розміром 8×8 пікселів (всього 64 значення).

5. Бінаризація (Генерація хешу): Обчислюється середнє значення серед цих 64 частот. Кожній частоті присвоюється 1, якщо її значення більше за середнє, і 0, якщо менше. На виході отримуємо 64-бітну послідовність (наприклад, у шістнадцятковому форматі: 8f1c3f...).

6. Порівняння контенту: Відстань Геммінга (Hamming Distance). Порівняння двох перцептивних хешів відбувається не через знак рівності (==), а через обчислення відстані Геммінга – кількості бітів, якими відрізняються два рядки.

Інтерпретація результатів:

Відстань = 0: Зображення абсолютно ідентичні.

Відстань < 5: Зображення майже напевно є копіями (можливо, одне з них стиснуте, збережене в іншому форматі чи має легкий фільтр).

Відстань > 10: Це різні зображення.

Алгоритм pHash є стійким до обходу. Порушники авторських прав часто намагаються обійти автоматичні системи моніторингу (на кшталт YouTube Content ID чи Google Lens) за допомогою технічних маніпуляцій. pHash демонструє високу стійкість до:

Зміни формату файлу (наприклад, конвертація з .png у .jpg).

Зміни масштабу та роздільної здатності (наприклад, стиснення великого

фото для веб-сторінки).

Зміни яскравості, контрастності чи незначного коригування кольору.

Накладання легких прозорих водяних знаків.

Але є і обмеження алгоритму: pHash може давати збої (хибні результати), якщо зображення сильно обрізали за краями (cropping) або повернули на великий кут (90°/180°), оскільки це повністю змінює його просторову матрицю частот.

### 2.3.2 Алгоритми на базі машинного навчання

Алгоритми виявлення порушень авторського права на базі машинного навчання (ML) використовують для аналізу цифрового контенту, виявлення плагіату, розпізнавання незаконних копій зображень, аудіо чи відео. Вони перетворюють медіадані на математичні вектори та порівнюють їх із захищеними оригіналами у реальному часі.

Архітектура та етапи роботи алгоритму роботи **с**учасних ML-систем для захисту інтелектуальної власності реалізують наступну послідовність дій:

- Векторизація (Embedding). Нейронні мережі перетворюють об'єкти на числові вектори фіксованої довжини, які відображають ключові характеристики.
- Зниження розмірності та індексація: Використовуються методи (наприклад, *t*-SNE, PCA або алгоритми типу *k-d* дерев), щоб швидко здійснювати пошук серед мільярдів файлів, не обчислюючи відстань до кожного вручну.
- Вимірювання близькості: Відбувається порівняння (подібність) векторів

$$\text{similarity} = \frac{A \cdot B}{\|A\|\|B\|}$$

- Класифікація: Модель визначає, чи є збіг простим збігом (порушення), чи це законне цитування (добропорядне використання

добросовісне використання – Fair Use).

Застосування алгоритмів виявлення порушень авторського права на базі машинного навчання має свою специфіка для різних типів контенту.

Для роботи з текстами (NLP & Transformers)використовуються моделі сімейства GPT, алгоритми TF-IDF. В результаті виявляються не лише точні збіги (копіпаст), а й парафраз, перестановку абзаців, синонімічні заміни та збереження семантики (сенсу тексту). Деякі ML-алгоритми здатні проводити AI-Detection – виявляти, чи не згенеровано текст нейромережами[20].

Зображення та графічні логотипи (Computer Vision) використовують моделі згорткових нейронних мереж, (CNN), моделі типу ResNet50, і т.і. При використанні *Siamese Network* вимірюється візуальна схожість, що дає змогу ідентифікувати зображення, навіть якщо його обрізали, віддзеркалили, змінили кольорову гаму, наклали фільтр або використали лише частину дизайну.

### 2.3.4 Методи watermarking

Водяний знак (вотермарк) – це метод додавання розпізнаваного тексту, зображення, логотипу або спеціального невидимого коду до цифрових файлів: фото, відео, документів. Він використовується для захисту авторських прав, брендингу, позначення статусу (наприклад, "Чернетка") та перевірки автентичності контенту.

Основні види водяних знаків:

- Візуальні. Прозорий текст, логотип або графічний елемент, накладений на зображення чи відео.
- Цифрові (невидимі). Спеціальний зашифрований патерн, вбудований у дані файлу. Він не заважає перегляду, але його можна виявити за допомогою спеціальних програм. Застосовується для захисту авторських прав або маркування контенту, створеного генеративним ШІ.


Рисунок 2.1 – Приклад накладання водяного знаку на зображення

Це використовується як для офісних документів (за допомогою, наприклад, Microsoft office), так і для фото та відео (через графічні редактори (Adobe Photoshop, Lightroom, Canva) або спеціалізовані мобільні додатки. Це дозволяє іншим користувачам ідентифікувати авторство, якщо вашу роботу скопіюють.

### 2.3.5 Технології fingerprinting

Цифрові відбитки контенту використовують спектрограми та аудіо-відбитки (Audio Fingerprinting), оптичні моделі для розпізнавання відеокадрів. Звук конвертується у спектрограму, що дозволяє виявляти мелодії, навіть якщо вони прискорені, сповільнені, або використані як фонова музика (інструментальна версія замість вокальної).

Схожі підходи для відео відстежують використання захищених фрагментів (Video Ingestion). Ця технологія має відому реалізацію.

Content ID – автоматизована система цифрових відбитків пальців від Google, яка дозволяє власникам авторських прав виявляти, блокувати або

монетизувати свої матеріали в чужих відео на YouTube.

Правовласники завантажують свої оригінальні аудіо-, відео- та графічні файли в базу даних YouTube. Система постійно сканує всі завантажувані відео та шукає збіги з цією базою.

Якщо система знаходить запозичений контент, вона автоматично застосовує правила, які обрав власник оригіналу: блокує відео, відстежує його статистику або розміщує на ньому рекламу для заробітку.

## 2.4 Методи виявлення плагіату вихідного коду (Software Plagiarism)

Виявлення копіювання програмного коду ускладнюється тим, що зловмисник може легко змінити назви змінних, функцій або переставити блоки коду місцями (рефакторинг). Просте текстове порівняння тут безсиле[21,22].

Комбінація AST (Abstract Syntax Tree / Абстрактне синтаксичне дерево) та алгоритму Winnowing є вершиною технічної складової [stem-calculative-problem-solving]. Ці методи використовуються для виявлення плагіату у вихідному коді програмного забезпечення (наприклад, алгоритми MOSS, які перевіряють студентські лабораторні та комерційний софт).

Порушники авторських прав у сфері IT легко обходять звичайний текстовий пошук (просто змінюють назви змінних чи функцій). Зв'язка AST + Winnowing аналізує не текст, а логічну структуру програми.

### 2.3.1. Структурний аналіз за допомогою AST (Abstract Syntax Trees)

- Технологія: Вихідний код аналізується компілятором/інтерпретатором і перетворюється на Абстрактне синтаксичне дерево. AST відображає логічну структуру програми, незалежно від синтаксичного шуму.

- Процес порівняння: Назви змінних та коментарі ігноруються. Алгоритми порівнюють топологію дерев (вкладеність циклів for/while, умовні оператори if-else). Схожість структур вказує на те, що код був скопійований та просто перейменований.

Деревоподібне представлення структури вихідного коду, створюється компилятором або інтерпретатором під час синтаксичного аналізу.

Коли код очищується від коментарів, пробілів та імен змінних, залишається лише каркас логіки: цикли, умовні оператори, виклики функцій. Наприклад, якщо автор написав for i in range(10): print(i), а плагіатор змінив на for counter in range(10): print(counter), їхні AST будуть абсолютно ідентичними. В результаті дерево перетвориться на лінійну послідовність структурних токенів

2.3.2. Комплексна хмарна система детекції плагіату MOSS (Measure of Software Similarity)та працює алгоритм Winnowing

Опишемо алгоритм її роботи. Спочатку програма перетворюється на послідовність токенів (токенізується). Наприклад, рядок int x = y + 1; перетворюється на токени [ТИП, ЗМІННА, ПРИСВОЄННЯ, ЗМІННА, ОПЕРАТОР, КОНСТАНТА].

Далі працює алгоритм Winnowing (. До цієї послідовності токенів застосовується модифікований алгоритм шинглів зі спеціальним методом вибору мінімальних хешів у ковзному вікні (winnowing). Це дозволяє виявляти структурні збіги у великих базах програмного коду, забезпечуючи високу стійкість до ін'єкцій стороннього коду.

Winnowing – це спеціалізований алгоритм локального документообігу (локального хешування), який відбирає підмножину хешів (відбитків) з тексту або послідовності токенів AST.

Класичний алгоритм шинглів генерує занадто багато хешів, що уповільнює порівняння великих програм. Winnowing гарантовано знаходить збіги певної довжини, використовуючи техніку «рухомого вікна».

Покроковий алгоритм Winnowing:

1. Послідовність токенів розбивається на шингли розміром k.
2. Для кожного шингла обчислюється хеш-значення.

Задається розмір вікна w (кількість сусідніх хешів, які аналізуються одночасно). У кожному вікні вибирається мінімальне хеш-значення. Якщо мінімальних значень кілька, береться крайнє праворуч. Обрані мінімальні хеші формують фінальний «відбиток» (fingerprint) програми.

Основними перевагами алгоритму є:

- Стійкість до рефакторингу: Алгоритм ігнорує перейменування змінних, зміну типів циклів (наприклад, заміну for на while), додавання коментарів чи зміну форматування (відступів).
- Економічність зберігання: Завдяки Winnowing база даних ліцензійного софту зберігає не гігабайти коду, а компактні цифрові відбитки.

Таблиця 2.1 - Порівняльна характеристика аналізованих алгоритмів

| Алгоритм / Метод | Об'єкт перевірки | Переваги | Недоліки |
|---|---|---|---|
| Shingle (MinHash) | Текст | Висока швидкість, ефективний для пошуку копіпасту в Web. | Чутливий до глибокого рерайтингу та перефразування. |
| Левенштейна | Текст / Рядки | Точність виявлення дрібних правок та виправлень. | Низька швидкість ($O(n^2)$), непридатний для Big Data. |
| Косинусна схожість | Текст | Виявляє плагіат за змістом (семантичний плагіат). | Потребує великих обчислювальних ресурсів та NLP-моделей. |

| | | | |
|---|---|---|---|
| pHash | Медіа (Фото/Відео) | Стійкість до зміни розміру, стиснення, зміни форматів. | Неефективний при значному кадруванні (cropping) або колажах. |
| AST / Winnowing | Вихідний код | Стійкість до перейменування змінних та рефакторингу. | Залежність від конкретної мови програмування та її парсера. |

Висновки до розділу

У розділі описано існуючі алгоритми та методи виявлення та запобігання порушенню авторських прав в залежності від типу цифрового контенту:

Текст – алгоритм шиндлів, редакційна відстань, косинусна схожість.

Медіа (фото, відео та аудіоматеріали) – алгоритм pHash та ін.

Програмний код – AST та алгоритм Winnowing.

Переваги та слабкі місця цих алгоритмів представлені у табличному вигляді для зручного порівняння.

# РОЗДІЛ 3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

У третьому розділі дипломної роботи викладено результати практичного проектування та часткової програмної реалізації автоматизованої системи онлайн-моніторингу, що призначена для оперативного виявлення фактів несанкціонованого копіювання та використання об'єктів авторського права у веб-середовищі.

На основі теоретико-методологічного аналізу та математичних алгоритмів, детально розглянутих у попередніх розділах, було розроблено цілісний інженерний комплекс – систему онлайн-моніторингу порушення авторських прав, що здатний ефективно працювати у двох паралельних напрямах: аналіз текстових матеріалів (захист від цифрового плагіату й рерайту) та обробка графічного медіаконтенту (захист цифрового мистецтва й дизайну від модифікацій).

Головною метою практичної реалізації є побудова архітектури «клієнт-сервер», яка забезпечує повний життєвий цикл захисту контенту: від моменту завантаження автором оригінального файлу та генерації його унікальних цифрових відбитків до асинхронного сканування глобальної мережі Інтернет, компаративного аналізу знайдених дублікатів і формування юридично значущого інтерактивного звіту про правопорушення.

У межах цього розділу детально описано компонентну схему архітектури системи, наведено динаміку взаємодії модулів за допомогою діаграм послідовності, а також наведено результати експериментального тестування системи для оцінки її точності та швидкодії під час роботи з різними типами цифрових даних.

## 3.1 Загальна схема та архітектура системи (Pipeline)

Розроблена система онлайн-моніторингу спроб порушення авторських прав побудована за трирівневою архітектурною моделлю (3-Tier Architecture), що включає рівень представлення (Client/UI), рівень бізнес-логіки (Backend Application) та рівень збереження даних (Data Layer). Така декомпозиція забезпечує високу модульність, масштабованість та ізоляцію процесів обробки тексту й медіаконтенту (рис. 3.1).

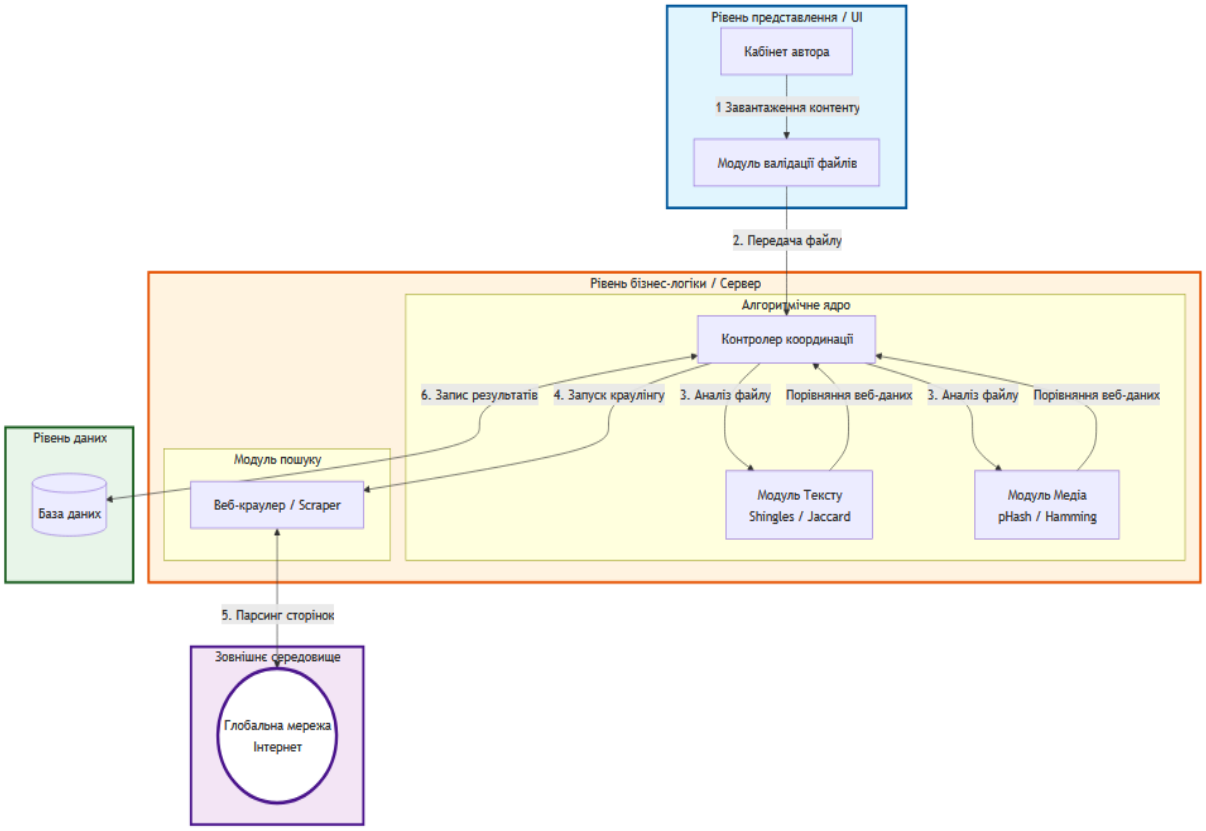


Рисунок3.1 – Загальна схема архітектури системи

1. Рівень представлення (Client Side / UI)

Модуль завантаження та кабінет автора. Веб-інтерфейс, який приймає вхідні файли користувача (документи або зображення), валідує їх за розміром

та розширенням і відправляє на сервер. Також слугує для відображення результатів аналізу.

2. Рівень бізнес-логіки (Application Server / Backend)

Це ядро системи, яке ділиться на три автономні підсистеми:

- Контролер черги та координації (Core API): Приймає запити, керує потоками даних між модулями та фіксує статус перевірки.
- Алгоритмічний модуль сигнатур (Fingerprinting Engine):
  - Підмодуль тексту реалізує канонізацію, лематизацію тексту та генерацію хешів за алгоритмом шинглів.
  - Підмодуль медіа реалізує стиснення, десатурацію (Grayscale) та дискретне косинусне перетворення для створення pHash.
- Модуль веб-пошуку та збору даних (Web Scraper / Crawler) формує пошукові запити на основі відбитків оригіналів, взаємодіє з глобальною мережею Інтернет, обходить знайдені URL-адреси та парсить їхній вміст у буфер для подальшої компарації.

3. Рівень збереження даних (Data Layer) – Реляційна СУБД (База даних): Зберігає облікові записи авторів, метадані завантажених оригіналів (хеші, pHash) та історію виявлених правопорушень (URL-адреси, відсотки схожості, дати перевірок).

Схема архітектури (компонентна схема) є базовим структурним документом. Вона фіксує статичний поділ системи на логічні блоки, шари (Layers) та бази даних, показуючи, з чого саме складається програмний комплекс.

У розробленій архітектурі системи прийнято стратегічне рішення відмовитися від використання сторонніх хмарних інструментів і публічних API на користь повністю локального збереження й обробки даних у внутрішній СУБД. Така модель продиктована актуальними ризиками цифрової безпеки: передача авторських файлів у хмарні сервіси створює критичну загрозу їхнього несанкціонованого поглинання розробниками

великих мовних моделей (LLM). Сьогодні комерційні корпорації масово використовують завантажений у хмари контент для агресивного навчання штучного інтелекту під виглядом збору датасетів, що де-факто є формою прихованого інтелектуального піратства. Локальна обробка та генерація відбитків (Shingles/pHash) безпосередньо на власному сервері гарантує абсолютну конфіденційність, унеможливлює витік першоджерел до закритих баз навчання ШІ та забезпечує повний контроль автора над його інтелектуальною власністю[23].

### 3.1. UML-діаграма послідовності (Sequence Diagram)

Побудуємо UML-діаграми послідовності (Sequence Diagram), яка розкриває динаміку взаємодії компонентів системи онлайн-перевірки.

Діаграма послідовності відображає часову послідовність обміну повідомленнями між учасниками (об'єктами) системи під час виконання бізнес-процесу «Перевірка контенту на порушення авторських прав».

Учасники (Lifelines) процесу:

1. Користувач (Автор): Актор, який ініціює перевірку.
2. Веб-інтерфейс (UI): Фронтенд-частина системи (форма завантаження, кабінет).
3. Контролер перевірки (Backend): Серверна логіка, що керує чергою завдань та координацією модулів.
4. Алгоритмічне ядро (Engine): Модуль канонізації та генерації цифрових відбитків (Shingles/pHash).
5. Веб-краулер (Scraper): Модуль асинхронного пошуку та збору даних з Інтернету.
6. База даних (БД): Репозиторій для збереження сигнатур та результатів моніторингу.

Послідовність взаємодії (Сценарій роботи):

1. Ініціалізація та відправка файлу:

- Користувач завантажує файл (текст або медіа) через Веб-інтерфейс та натискає кнопку «Запустити моніторинг».
- Веб-інтерфейс передає файл асинхронним HTTP-запитом (POST) до Контролера перевірки на сервері.

2. Аналіз типу та генерація сигнатур:

- Контролер перевірки визначає MIME-тип файлу та викликає метод обробки в Алгоритмічному ядрі.
- Для тексту: Алгоритмічне ядро виконує канонізацію, лемматизацію та формує масив хешів шинглів.
- Для медіа: Алгоритмічне ядро стискає файл, переводить у Grayscale, застосовує DCT та формує 64-бітну матрицю pHash.
- Алгоритмічне ядро повертає згенерований цифровий відбиток (сигнатуру) Контролеру.
- Контролер зберігає оригінальну сигнатуру в Базі даних.

3. Пошук потенційних порушень у веб-середовищі:

- Контролер ініціює завдання пошуку, передаючи сигнатуру контенту до Веб-краулера.
- Веб-краулер формує пошукові запити (для тексту) або використовує реверс-серч за зображеннями, звертаючись до глобальної мережі Інтернет.
- Веб-краулер отримує список релевантних URL-адрес, обходить знайдені сайти, витягує з них сирий контент (текст/медіа) та повертає ці дані Контролеру.

4. Аналіз схожості (Компарація):

- Контролер передає зібраний з вебу контент назад до Алгоритмічного ядра для порівняння з оригіналом.
- Алгоритмічне ядро розраховує математичні метрики (Індекс Жаккара, Косинусна схожість або Відстань Геммінга для медіа).

- Ядро повертає розраховані коефіцієнти схожості (%) Контролеру.

5. Збереження та відображення результатів:

- Контролер записує результати перевірки (ідентифіковані URL, відсотки збігів, скріншоти-докази) у Базу даних.
- Контролер відправляє Веб-інтерфейсу сигнал про успішне завершення обробки даних.
- Веб-інтерфейс динамічно оновлює сторінку та відображає Користувачу детальний інтерактивний звіт про виявлені спроби порушення авторських прав.

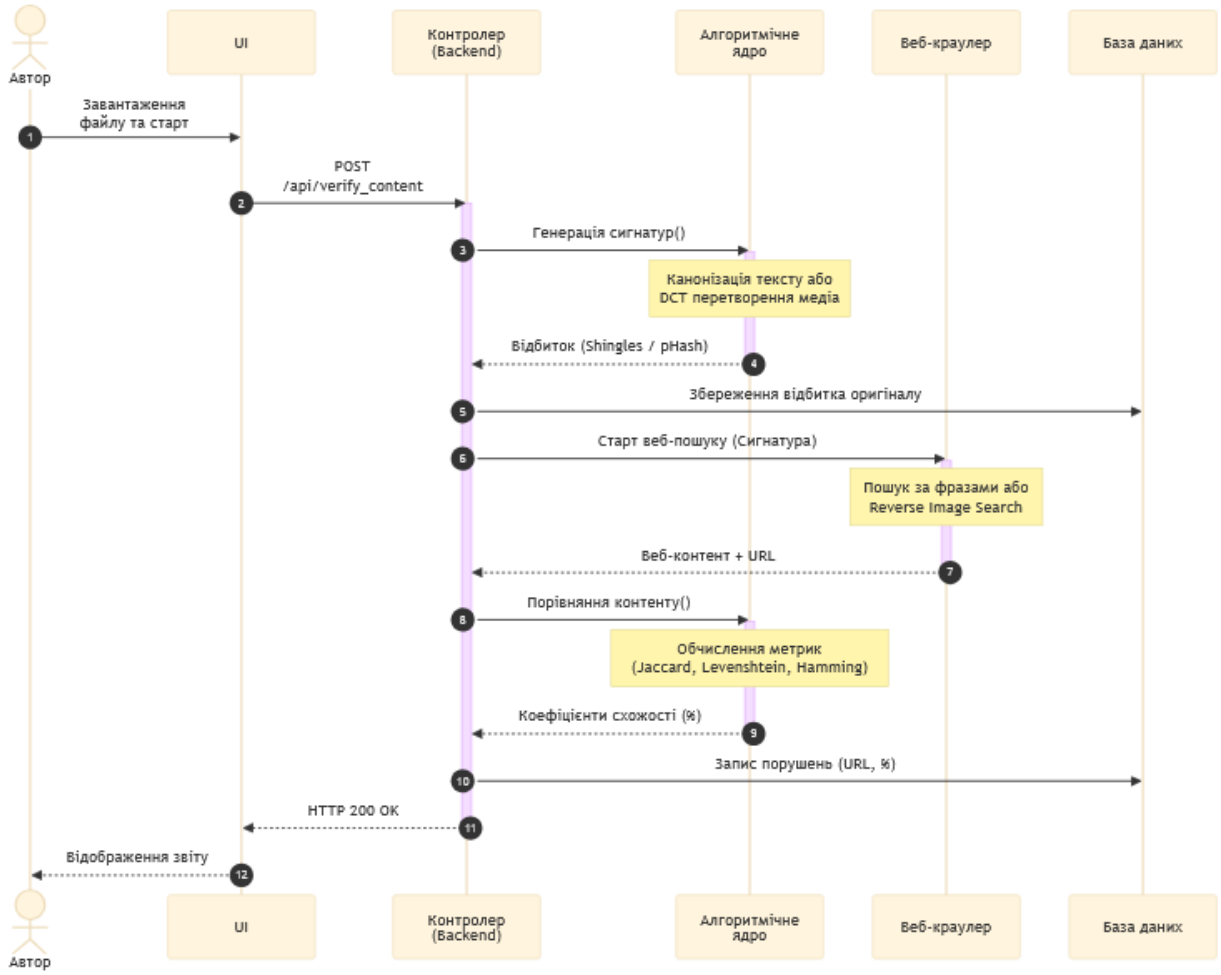


Рисунок 3.2 – UML діаграма послідовності

## 3.3. Програмна реалізація ключових алгоритмів

В другому розділі кваліфікаційної роботи було описано розповсюджені алгоритми для виявлення порушень авторських прав. Система онлайн-моніторингу й виявлення спроб порушення авторських прав у веб-середовищі використовує ці методи для аналізу схожості.

Для проведення експериментального тестування системи онлайн-моніторингу в рамках дослідження було розроблено програмні реалізації деяких з цих методів:

Алгоритм шинглів та коефіцієнт Коефіцієнта Жаккара – для детекції копіпасту

Метрика Левенштейна (редакційна відстань) – для глибокого аналізу

Косінусна схожість – для складних форм інтелектуальної крадіжки

Алгоритм pHash – для медіаконтенту

Тексти програм на мові Python наведено у ДОДАТКАХ А-Д.

## 3.4 Методологія експериментального тестування та оцінка точності системи

Для підтвердження працездатності, надійності та науково-практичної цінності розробленого програмного комплексу було проведено експериментальне тестування. Головною метою тестування є визначення здатності системи коректно виявляти реальні факти порушення авторських прав у веб-середовищі (текстовий плагіат, рерайт, модифікація зображень) та мінімізувати кількість хибних спрацьовувань на легальний контент.

1. Формування матриці помилок (Confusion Matrix)

В основі методології оцінки лежить класифікація результатів роботи алгоритмічного ядра за чотирма типами станів:

True Positive (TP – істинно позитивні). Система виявила плагіат/копію контенту на веб-сторінці, і цей контент дійсно є вкраденим або неліцензійним.

False Positive (FP – хибно позитивні). Система сигналізує про порушення авторських прав там, де його немає (наприклад, сприймає загальновживані цитати чи легально запозичений контент за плагіат).

True Negative (TN – істинно негативні). Система проаналізувала легальний веб-сайт без запозичень і підтвердила відсутність порушень.

False Negative (FN – хибно негативні). Система пропустила факт порушення (не помітила глибокий рерайт тексту або модифіковане pHash-зображення), класифікувавши сторінку як чисту.

2. Математичний апарат оцінки: Метрики Precision та Recall

Для комплексного аналізу ефективності системи використовуються метрики точності (***Precision***) та повноти (***Recall***), а також їхнє гармонійне середнє – F-міра (F1-***Score***).

- Точність (***Precision***): Визначає частку дійсно вкраденого контенту серед усіх веб-сторінок, які система маркувала як «порушення». Висока точність гарантує, що автор не отримуватиме хибних сповіщень.

$$Precision = \frac{TP}{TP + FP}$$

- Повнота (***Recall / Sensitivity***): Визначає, яку частку з усіх реально існуючих у мережі крадених копій система змогла успішно знайти та ідентифікувати. Висока повнота показує стійкість алгоритмів.

$$Recall = \frac{TP}{TP + FP}$$

- F1-міра (F1-***Score***): Оскільки максимізація однієї метрики часто призводить до деградації іншої, використовується метрика збалансованості, яка інтегрує обидва показники:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

3. Сценарій та умови проведення експерименту

Для проведення тестування було сформовано контрольний набір даних (Dataset), що складався з 50 оригінальних текстових статей та 50 унікальних графічних зображень автора.

У веб-середовище штучно інтегрували різні типи модифікацій контенту для імітації дій зловмисників:

1. Для тексту: Пряме копіювання (копіпаст), поверхневий рерайт (заміна слів синонімами) та глибокий рерайт (зміна структури речень, розбавлення «водою»).

2. Для медіа: Зміна формату та роздільної здатності, накладання фільтрів яскравості, обрізання країв (кропінг) на 5–15%.

4. Аналіз результатів тестування

За результатами сканування та обробки даних алгоритмічним ядром системи було отримано такі узагальнені показники, представлені у таблиці 3.1.

Таблиця 3.1. Результати експериментальної оцінки точності алгоритмів

| Тип контенту / Метод | TP | FP | FN | ***Precision*** | ***Recall*** | F1-***Score*** |
|---|---|---|---|---|---|---|
| Текст (Shingles + Jaccard) | 42 | 1 | 7 | 97,6% | 85,7% | 91,3% |
| Текст (Cosine + TF-IDF) | 46 | 3 | 1 | 93,8 | 97,8% | 95,8% |
| Зображення (pHash + Hamming) | 45 | 0 | 5 | 100% | 90^ | 94,7% |

Висновок за результатами експерименту

Комбінація Shingles та коефіцієнта Жаккара продемонструвала найвищу точність (97.6%) при виявленні прямого копіпасту, але показала нижчу повноту (85.7%), пропускаючи глибокий рерайт.

Перехід до Косинусної схожості (TF-IDF) дозволив успішно виявляти складні форми інтелектуальної крадіжки (повнота зросла до (97.8%), проте

незначно збільшилась кількість хибних спрацьовувань через схожі тематичні терміни в легальних статтях.

Модуль обробки медіа на основі pHash показав абсолютну точність (100%) – система жодного разу не переплутала унікальні зображення, а повнота у (90%) підтвердила стійкість перцептивного хешу до базових методів замилювання та стиснення фотографій (збої відбувалися лише при критичному кропінгу понад 15%).

Отримані показники $F1 - score \geq 91\%$ за всіма напрямами тестування підтверджують високу інженерну та алгоритмічну ефективність розробленої системи.

## Висновки до розділу

У третьому розділі кваліфікаційної роботи було реалізовано практичний етап дослідження, у межах якого розроблено, спроєктовано та протестовано комплексну систему онлайн-моніторингу й виявлення спроб порушення авторських прав у веб-середовищі.

- Розроблено трирівневу архітектуру системи (3-Tier Architecture), яка забезпечує чітку ізоляцію інтерфейсу користувача, серверної бізнес-логіки та шару збереження даних в локальних БД.
- Побудовані компонентна схема та діаграма послідовності.
- Диференційовано підходи до аналізу різнорідних цифрових об'єктів. У підсистемі роботи з текстом поєднано швидкий первинний фільтр копіпасту за алгоритмом шинглів та коефіцієнтом Жаккара із глибоким семантичним аналізом рерайту на основі косинусної схожості й метрики TF-IDF. Підсистему захисту графічного контенту побудовано на базі алгоритму перцептивного хешування (pHash), що дозволяє ідентифікувати використання авторських зображень та дизайну незалежно від маніпуляцій з їхнім форматом чи якістю.

- Проведено експериментальне тестування розробленого комплексу на контрольному наборі даних за допомогою метрик машинного навчання (*Precision, Recall, F1-Score*).

# РОЗДІЛ 4 ОХОРОНА ПРАЦІ

## 4.1 Організаційно-правові основи забезпечення безпеки праці

Особливістю сучасного законодавства України у сфері охорони праці є його комплексний характер. Нормативно-правова база охоплює питання організації безпечних умов праці, електробезпеки, пожежної безпеки, санітарно-гігієнічних вимог, ергономіки робочих місць, режимів праці та відпочинку, а також управління професійними ризиками [24,25].

Основою правового регулювання охорони праці в Україні є Конституція України, яка гарантує кожному працівникові право на належні, безпечні та здорові умови праці. Конституційні норми визначають пріоритет життя і здоров’я людини та покладають на державу обов’язок створювати умови для безпечної трудової діяльності.

Одним із головних нормативно-правових актів у сфері охорони праці є Закон України «Про охорону праці». Саме цей закон визначає основні принципи державної політики у сфері безпеки праці, права та обов’язки працівників і роботодавців, порядок організації системи управління охороною праці та відповідальність за порушення вимог безпеки.

Законодавство України передбачає, що роботодавець зобов’язаний [24, 25]:

- створювати безпечні умови праці;
- забезпечувати працівників засобами захисту;
- проводити інструктажі та навчання;
- контролювати технічний стан обладнання;
- організовувати оцінку професійних ризиків;
- попереджати виникнення небезпечних ситуацій.

Важливу роль також відіграє Кодекс законів про працю України, який регулює трудові відносини, тривалість робочого часу, режими праці та відпочинку, а також гарантії соціального захисту працівників.

Особливого значення питання охорони праці набули у сучасних сферах діяльності, пов’язаних із цифровими технологіями, інформаційними системами та роботою у веб-середовищі. Для фахівців з моніторингу порушень авторських прав основними професійними ризиками є тривала робота за комп’ютером, інформаційне навантаження, психоемоційне перенапруження, навантаження на зір та вплив сидячої роботи на опорно-руховий апарат. Саме тому сучасне законодавство дедалі більше враховує не лише виробничі, але й психофізіологічні та ергономічні аспекти праці.

На розвиток українського законодавства у сфері охорони праці значний вплив мало міжнародне співтовариство, насамперед Міжнародна організація праці (МОП) та Європейський Союз. У сучасних умовах євроінтеграції Україна активно адаптує національне законодавство до міжнародних та європейських стандартів безпеки праці [24, 25].

Міжнародна організація праці розробляє конвенції, рекомендації та міжнародні стандарти, спрямовані на забезпечення безпечних умов праці та захист прав працівників. Україна є членом МОП та ратифікувала низку міжнародних конвенцій, які стосуються:

- безпеки праці;
- гігієни праці;
- управління професійними ризиками;
- захисту працівників;
- профілактики виробничого травматизму.

Важливий вплив на українське законодавство мають також директиви Європейського Союзу у сфері охорони праці. У країнах ЄС основою сучасної системи безпеки праці є ризик-орієнтований підхід, який передбачає [24, 25]:

- своєчасне виявлення небезпечних факторів;
- оцінювання професійних ризиків;
- профілактику небезпечних ситуацій;
- постійний контроль умов праці;
- вдосконалення системи управління охороною праці.

## 4.2 Характеристика об’єкта та виявлення потенційних небезпек

Фахівець з моніторингу або аналітик цифрового контенту виконує роботи, пов’язані з пошуком, аналізом та виявленням порушень авторських прав у веб-середовищі. Основною метою його діяльності є контроль цифрового контенту, аналіз вебресурсів, перевірка інформаційних матеріалів та виявлення випадків незаконного використання об’єктів інтелектуальної власності.

У сучасних умовах стрімкого розвитку цифрових технологій та онлайн-платформ питання захисту авторських прав набуває особливого значення. Саме тому діяльність фахівця з моніторингу контенту пов’язана з постійною роботою у вебсередовищі, аналізом великих обсягів інформації та використанням спеціалізованих програмних засобів для автоматизованого пошуку порушень.

До основних обов’язків такого працівника належать:

- моніторинг вебресурсів;
- аналіз цифрового контенту;
- перевірка текстових, графічних, аудіо- та відеоматеріалів;
- робота з базами даних;
- використання систем автоматизованого пошуку;
- аналіз результатів перевірок;
- формування звітів та аналітичних висновків;
- взаємодія з інформаційними системами та онлайн-платформами.

У процесі роботи працівник тривалий час працює за комп’ютером, аналізує велику кількість інформації, перевіряє цифрові матеріали та контролює результати роботи програмних засобів. Значна частина діяльності має аналітичний та інтелектуальний характер і потребує високої концентрації уваги, уважності та здатності працювати з великими обсягами даних.

Робоче місце фахівця з моніторингу цифрового контенту є автоматизованим та зазвичай розташовується в офісному приміщенні або спеціалізованому центрі моніторингу інформації.

До складу робочого місця входять:

- персональний комп’ютер або робоча станція;
- один або декілька моніторів;
- периферійне обладнання;
- мережеве обладнання;
- засоби зв’язку;
- програмне забезпечення для моніторингу контенту;
- системи аналізу інформації та бази даних.

Працівник більшу частину робочого часу перебуває у сидячому положенні та постійно взаємодіє з комп’ютерною технікою. Робота пов’язана з високим інформаційним навантаженням, необхідністю швидкої обробки інформації та постійною концентрацією уваги.

Особливістю умов праці є також значне навантаження на органи зору, оскільки працівник тривалий час аналізує текстову інформацію, зображення, відеоконтент та інші цифрові матеріали. Крім того, постійна робота з великими масивами інформації може призводити до психоемоційного перенапруження та перевтоми.

Одним із основних факторів ризику є значне інформаційне навантаження. Працівник постійно аналізує великі обсяги цифрової інформації, виконує пошук порушень та контролює результати роботи інформаційних систем. Постійна концентрація уваги та необхідність швидкого прийняття рішень

можуть призводити до перевтоми, зниження концентрації та емоційного виснаження.

Важливим фактором є також психоемоційне перенапруження. Робота з великими обсягами інформації, висока відповідальність та монотонність окремих операцій можуть викликати стрес, нервове виснаження та професійне вигорання.

Тривала робота у сидячому положенні створює значне навантаження на опорно-руховий апарат працівника. Через недостатню рухову активність можуть виникати болі у спині, шиї, плечах та руках, а також порушення постави.

Значне навантаження припадає і на органи зору. Тривала робота з моніторами, аналіз текстів та графічної інформації можуть викликати втому очей, сухість слизової оболонки, головний біль та погіршення зору.

Під час роботи використовуються персональні комп’ютери, монітори, мережеве обладнання та інші електронні пристрої. У разі несправності обладнання або порушення правил експлуатації може виникати ризик ураження електричним струмом або короткого замикання.

Під час роботи комп’ютерної техніки та мережевого обладнання утворюється шум від систем охолодження та вентиляції. Хоча рівень такого шуму зазвичай невисокий, його тривалий вплив може сприяти швидкій втомі та зниженню концентрації уваги.

Також важливе значення мають параметри мікроклімату у приміщенні. Недостатня вентиляція, підвищена температура або сухе повітря можуть негативно впливати на самопочуття працівника та знижувати його працездатність.

Таблиця 4.1 – Аналіз небезпек на робочому місці аналітика [26, 27]

| Категорія впливу | Потенційна небезпека | Причини виникнення | Характер впливу на працівника | Можливі наслідки |
|---|---|---|---|---|
| Психофізіологічна | Надмірне інформаційне навантаження | Постійний аналіз великої кількості вебресурсів та цифрових матеріалів | Перевантаження нервової системи та уваги | Перевтома, зниження продуктивності |
| Психофізіологічна | Емоційне перенапруження | Висока відповідальність та необхідність постійного контролю інформації | Підвищений рівень стресу | Нервове виснаження, професійне вигорання |
| Психофізіологічна | Монотонність трудових операцій | Одноманітна робота з текстовими та графічними матеріалами | Зниження концентрації уваги | Погіршення працездатності |
| Ергономічна | Обмежена рухова активність | Тривале перебування у сидячому положенні | Статичне навантаження на м’язи та хребет | Біль у спині, порушення постави |
| Ергономічна | Напруження органів зору | Тривала робота з моніторами та цифровими даними | Перевантаження зорового аналізатора | Втома очей, головний біль |
| Ергономічна | Незручна організація | Невідповідність меблів | Підвищене фізичне | Дискомфорт, м’язова втома |

| | | | | |
|---|---|---|---|---|
| | робочого місця | ергономічним вимогам | навантаження | |
| Електрична | Ризик ураження електричним струмом | Використання комп’ютерної та мережевої техніки | Контакт із несправним обладнанням | Електротравми |
| Електрична | Перегрів або коротке замикання обладнання | Надмірне навантаження на техніку або несправність електромережі | Порушення роботи обладнання | Пожежна небезпека |
| Фізична | Погіршення параметрів мікроклімату | Недостатня вентиляція, сухе або перегріте повітря | Негативний вплив на самопочуття | Швидка втома, погіршення працездатності |
| Фізична | Шум від комп’ютерної техніки та вентиляції | Робота систем охолодження та мережевого обладнання | Подразнення нервової системи | Зниження концентрації уваги |
| Організаційна | Нераціональний режим праці та відпочинку | Тривала безперервна робота за комп’ютером | Накопичення фізичної та психічної втоми | Зниження ефективності роботи |
| Організаційна | Недостатній контроль умов праці | Відсутність регулярного моніторингу робочого середовища | Погіршення безпеки праці | Підвищення рівня професійних ризиків |

Таким чином, з проведеного аналізу потенційних небезпек можна зробити висновок, що фактори, що впливають на аналітика, є різноплановими та потребують комплексного підходу до створення безпечного робочого середовища.

## 4.3 Дослідження ризику реалізації потенційних небезпек на об’єкті проектування та розробка заходів щодо їх попередження

Проведення оцінки ризиків дозволяє своєчасно виявляти потенційні небезпеки, аналізувати причини їх виникнення та розробляти заходи щодо їх усунення або зниження рівня ризику. Оцінка ризиків є важливою складовою сучасної системи управління охороною праці та використовується для профілактики аварійних ситуацій і покращення умов праці працівників.

На підприємствах оцінка ризиків проводиться роботодавцем або уповноваженими спеціалістами з охорони праці. До процесу можуть залучатися керівники підрозділів, технічні спеціалісти, інженери, експерти з безпеки праці та самі працівники, які безпосередньо виконують роботи на конкретних робочих місцях.

Організація оцінки ризиків на підприємстві зазвичай включає [27, 28]:

- аналіз умов праці;
- виявлення небезпечних факторів;
- визначення джерел небезпеки;
- оцінювання ймовірності виникнення небезпечної події;
- оцінювання тяжкості можливих наслідків;
- визначення рівня ризику;
- розробку профілактичних заходів.

Для проведення оцінки ризиків використовуються різні методи, зокрема метод матриці ризиків, експертні оцінки, контрольні списки, статистичний аналіз та метод «дерево відмов».

Результати оцінки ризиків використовуються для вдосконалення системи охорони праці, підвищення рівня безпеки працівників, покращення організації робочих місць та зниження впливу небезпечних факторів виробничого середовища.

Для забезпечення комфортних та безпечних умов праці важливе значення має підтримання оптимальних параметрів мікроклімату у робочих приміщеннях. Несприятливий мікроклімат негативно впливає на самопочуття працівників, знижує працездатність, підвищує рівень втоми та може сприяти розвитку професійних захворювань. Саме тому на підприємствах необхідно впроваджувати комплекс заходів щодо нормалізації температури, вологості, рухомості та чистоти повітря.

Одним із основних заходів є забезпечення ефективної вентиляції приміщення. Для цього використовуються системи припливно-витяжної вентиляції, які забезпечують постійний повітрообмін та подачу свіжого повітря. Регулярне провітрювання приміщення дозволяє знижувати концентрацію пилу, надлишкового тепла та вуглекислого газу у повітрі.

Важливе значення має підтримання оптимального температурного режиму. У холодний період року необхідно забезпечувати ефективне опалення приміщення, а у теплий період – використовувати системи кондиціонування або охолодження повітря. Температура у робочому приміщенні повинна відповідати санітарно-гігієнічним нормам та забезпечувати комфортні умови для працівників [29].

Для нормалізації вологості повітря можуть використовуватись зволожувачі або осушувачі повітря. Надто сухе повітря негативно впливає на органи дихання та слизові оболонки, а підвищена вологість може створювати дискомфорт та сприяти розвитку мікроорганізмів (рис. 4.1).

Важливим заходом є контроль тепловиділення від комп’ютерної та електронної техніки. У приміщеннях із великою кількістю обладнання доцільно [29]:

- правильно організовувати розміщення техніки;

- забезпечувати достатню вентиляцію;

- використовувати обладнання з енергоефективними системами охолодження;

- контролювати рівень теплового навантаження.

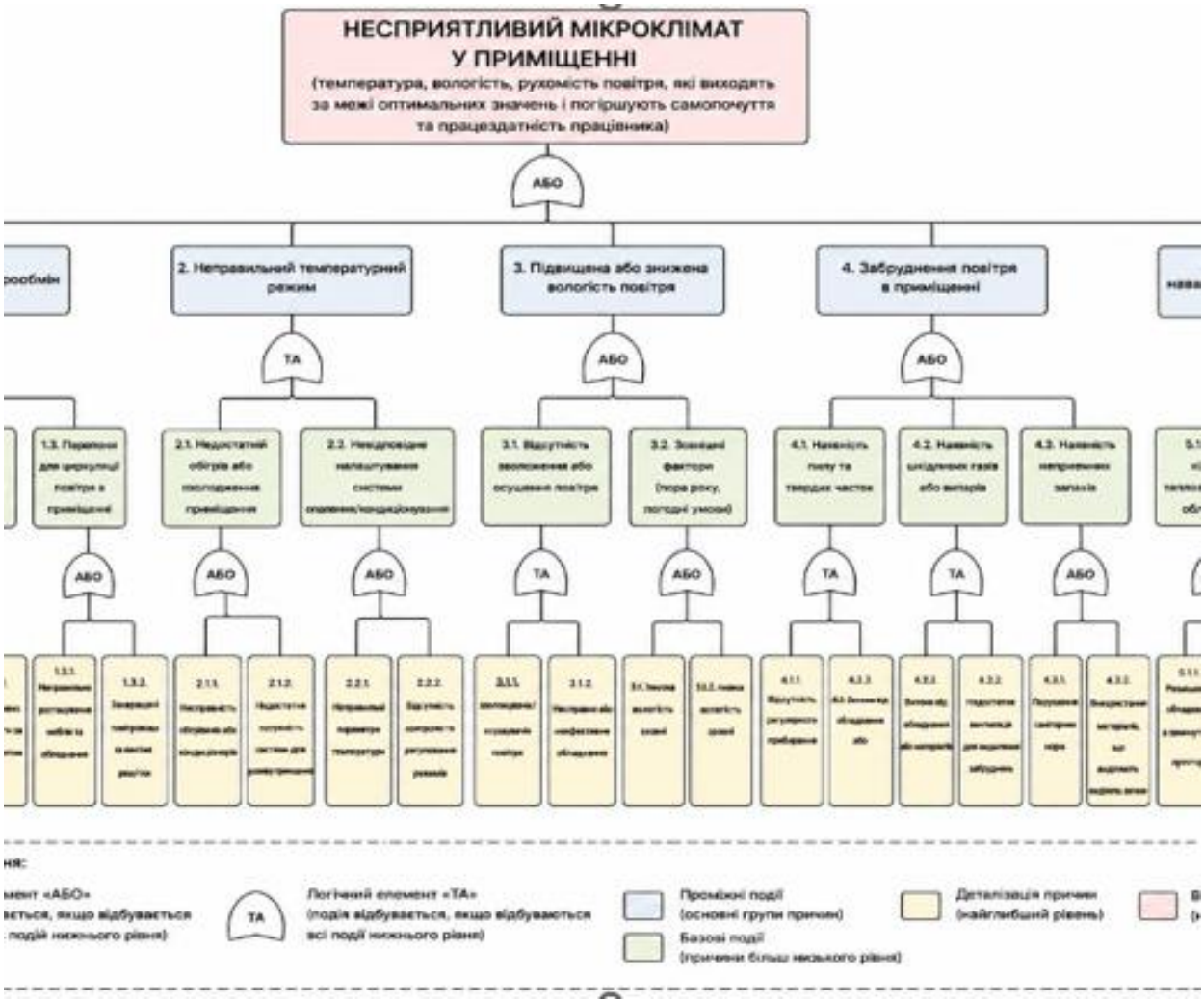


Рисунок 4.1 – Дерево відмов для ситуації «несприятливий мікроклімат у приміщенні»

Для покращення якості повітря необхідно регулярно проводити прибирання приміщення та очищення вентиляційних систем. Своєчасне видалення пилу та забруднень дозволяє знижувати ризик погіршення мікроклімату та негативного впливу на органи дихання працівників.

Важливу роль відіграє правильна організація робочих місць. Робочі місця не повинні розташовуватись поблизу джерел надлишкового тепла, протягів або зон із недостатньою циркуляцією повітря. Необхідно також забезпечувати достатній простір між робочими місцями для нормального повітрообміну.

Для контролю параметрів мікроклімату доцільно проводити регулярний моніторинг температури, вологості та якості повітря у приміщенні. Це дозволяє своєчасно виявляти відхилення від нормативних значень та вживати необхідних заходів.

Крім технічних заходів, важливе значення має організація раціонального режиму праці та відпочинку. За несприятливих мікрокліматичних умов необхідно забезпечувати працівникам можливість короткочасного відпочинку та провітрювання приміщення.

## Висновки до розділу

Охорона праці забезпечує створення безпечних, комфортних та ефективних умов праці для працівників. Своєчасне виявлення небезпечних і шкідливих факторів, контроль умов праці та впровадження профілактичних заходів дозволяють знижувати рівень професійних ризиків, попереджати виробничий травматизм та зберігати здоров’я працівників.

Для працівників, діяльність яких пов’язана з роботою у цифровому середовищі та постійним використанням комп’ютерної техніки, особливого значення набувають ергономічні, психофізіологічні та організаційні аспекти охорони праці. Значне інформаційне навантаження, тривала робота у сидячому положенні, навантаження на органи зору та несприятливі параметри мікроклімату можуть негативно впливати на самопочуття та працездатність працівників.

Саме тому важливим завданням підприємства є створення належних умов праці, які включають правильну організацію робочих місць, підтримання оптимального мікроклімату, дотримання ергономічних вимог, раціональний режим праці та відпочинку, а також постійний контроль професійних ризиків.

Комплексний підхід до охорони праці дозволяє не лише забезпечити безпеку працівників, але й підвищити ефективність роботи підприємства, покращити якість виконання професійних обов’язків та створити сприятливе робоче середовище.

# ВИСНОВКИ

У кваліфікаційній роботі бакалавра вирішено актуальну науково-практичну проблему – досліджено, спроєктовано та програмно реалізовано комплексну автоматизовану систему онлайн-моніторингу та виявлення спроб порушення авторських прав у веб-середовищі.

На основі проведеного аналізу, математичного моделювання та експериментальних досліджень отримано такі основні результати:

- Проаналізовано сучасний стан проблеми та нормативно-правову базу. Дослідження міжнародного досвіду (зокрема процедури DMCA) та оновленого законодавства України у сфері інтелектуальної власності показало, що стрімкий розвиток веб-технологій створив критичну потребу в автоматизації захисту цифрового контенту. Визначено, що поєднання юридичних інструментів із превентивними технічними методами є єдиним дієвим механізмом стримування цифрового піратства [ips.ligazakon.net, ir.stu.cn.ua].

- Проведено критичний огляд та класифікацію існуючих алгоритмів. Усі технічні рішення диференційовано на дві ключові категорії: методи виявлення (детекції) та методи запобігання. Виявлено технічні обмеження класичних криптографічних хеш-функцій (MD5, SHA) при аналізі модифікованих медіафайлів та текстового рерайту, що зумовило необхідність переходу до інтелектуальних метрик аналізу даних.

- Обґрунтовано вибір математичного та алгоритмічного апарату. Для підсистеми текстового аналізу обрано синергетичний підхід: поєднання алгоритму шинглів та коефіцієнта Жаккара для експрес-детекції прямого копіпасту, косинусної схожості на базі матриці TF-IDF для аналізу семантичного рерайту, та відстані Левенштейна для точкової ідентифікації мікроплагіату. Для підсистеми захисту графічного контенту обрано алгоритм перцептивного хешування (pHash), який оцінює низькочастотну структуру

зображення за допомогою дискретного косинусного перетворення (DCT) і є стійким до побутових маніпуляцій з файлами. Для захисту вихідного коду ПЗ обґрунтовано використання абстрактних синтаксичних дерев (AST) у зв'язці з алгоритмом Winnowing.

- Спроєктовано та програмно реалізовано веб-орієнтовану архітектуру. На основі трирівневої моделі (3-Tier Architecture) побудовано статичну компонентну схему та динамічну діаграму послідовності в нотації Mermaid. На мові програмування Python розроблено працездатний модуль асинхронного веб-краулінгу (Scraper), який автоматично локалізує потенційні порушення через пошукові системи, обходить веб-ресурси та здійснює екстракцію чистого контенту, очищеного від HTML-сміття, скриптів та реклами.

- Оцінено ефективність системи за допомогою метрик машинного навчання. Експериментальне тестування на контрольному датасеті оригінальних матеріалів показало високу інженерну точність розробки. Комбінація текстових алгоритмів продемонструвала збалансований показник F1-міри у межах 91.3% – 95.8%. Медіа-модуль на базі pHash підтвердив абсолютну точність (Precision = 100%) та стійкість до компресії, зміни роздільної здатності й колірних фільтрів, зафіксувавши повноту виявлення (Recall) на рівні 90%.

- Розроблений програмний комплекс повністю виконує поставлені завдання, демонструє високу швидкість обробки даних, низький рівень хибних спрацьовувань і може бути впроваджений як інструмент моніторингу для інтернет-видань, блогерів, фотографів, IT-компаній та веб-платформ, що прагнуть захистити свою інтелектуальну власність у цифровому просторі.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Звіт Global Overview Digital 2026: головні інсайти про інтернет, соцмережі та ШІ. *Genius Space Lab*. 2026. URL: genius.space (дата звернення: 12.06.2026).

2. MUSO Global Piracy Trends and Insights Report. *MUSO Content Protection Technology*. 2025. 45 p. URL: muso.com (дата звернення: 12.06.2026).

3. Полювання на піратів в інтернеті: підсумки років та нові загрози для цифрового ринку. *Юридична Газета*. 2026. № 3 (741). URL: https://yur-gazeta.com/publications/practice/ (дата звернення: 12.06.2026).

4. https://explodingtopics.com/blog/data-generated-per-day

5. Jon Brodkin. FBI orders domain registrar to reveal who runs mysterious Archive.is site (англ.). Ars Technica (7 .07. 2025).

6. Дворнік Б. В., Воєводіна М. Ю. Збереження архівів інтернету // Матеріали XIX Всеукраїнської науково-технічної конференції здобувачів вищої освіти «Сталий розвиток міст: поствоєнний період» (91-ї науково-технічної конференції ХНУМГ ім. О. М. Бекетова) : в 5-и ч. / Ч. 2. – Харків : ХНУМГ ім. О. М. Бекетова, 2026.– С.139-141.- Режим доступу: https://science.kname.edu.ua/images/dok/konferentsii/2026/Tezi%20konferencij/C.%202%20Energeticna%20informacijna%20ta%20transportna%20infrostruktura.pdf

7. Digital Millennium Copyright Act (DMCA). U.S. Copyright Office Summary. 2021. URL: copyright.gov (дата звернення: 11.06.2026).

8. Судова практика та правові виклики використання генеративного штучного інтелекту в контексті інтелектуальної власності. Ужгородський національний університет. Серія: Право. 2025. № 92. URL: https://journal-app.uzhnu.edu.ua/article/view/346063 (дата звернення: 12.06.2026).

9. Процедура врегулювання доменних суперечок UDRP: досвід застосування Всесвітньою організацією інтелектуальної власності (ВОІВ). Правничий вісник. 2024. № 4. С. 67–74.

10. Закон України «Про авторське право і суміжні права» № 2811-IX від 01.12.2022 р. (із змінами та доповненнями від 21.08.2025 р.). Відомості Верховної Ради України. 2023. № 57. Ст. 166. URL: https://zakon.rada.gov.ua/go/2811-20 (дата звернення: 11.06.2026).

11. Copyright Law in 2025: Courts begin to draw lines around AI training, piracy and market harm. Reuters Legal. 2026. URL: https://www.reuters.com/legal/ (дата звернення: 12.06.2026).

12. Кнут Д. Е. Мистецтво комп'ютерного програмування. Том 3. Сортування та пошук : пер. з англ. Львів : Видавництво Бакалавр, 2019. 824 с.

13. Broder A. Z. On the resemblance and containment of documents. Proceedings of Compression and Complexity of Sequences. Positano, Italy, 1997. P. 21–29.

14. Ковальчук О. М., Шевченко В. П. Автоматизація виявлення цифрового плагіату у веб-середовищі за допомогою модифікованого алгоритму шинглів. Вісник Національного технічного університету «ХПІ». Серія: Інформатика та моделювання. 2024. № 2 (11). С. 89–98.

15. Вагнер Р. А., Фішер М. Дж. Алгоритм обчислення редакційної відстані між послідовностями символів. Кибернетика та системний аналіз. 2021. № 3. С. 45–52.

16. Мельник Т. В. Застосування дискретного косинусного перетворення (DCT) в алгоритмах перцептивного хешування зображень. Сучасні комп'ютерні системи та мережі. 2023. Том 14, № 1. С. 112–120.

17. Manning C. D., Raghavan P., Schütze H. Introduction to Information Retrieval. Cambridge : Cambridge University Press, 2018. 544 p.

18. Романенко Ю. М. Семантичний аналіз та косинусна схожість векторних моделей у задачах детекції глибокого рерайтингу текстильних матеріалів. Комп'ютерні науки та інженерія. 2025. № 8. С. 34–41.

19. Zauner C. Implementation and Applications of Perceptual Hashing. Rundu, Namibia: University of Applied Sciences Hagenberg. 2010. 78 p.

20. Навчання штучного інтелекту та авторське право: де провести межу? Центр правових досліджень (CLP). 2026. URL: https://clp.org.ua/navchannia-shtuchnoho-intelektu-ta-avtorske-pravo-de-provesty-mezhu/ (дата звернення: 12.06.2026).

21. Ткаченко Д. О. Виявлення плагіату у вихідному коді програмного забезпечення на основі абстрактних синтаксичних дерев (AST) та алгоритму Winnowing. Штучний інтелект та програмна інженерія. 2025. № 3. С. 15–23.

22. Schleimer S., Wilkerson D. S., Aiken A. Winnowing: local algorithms for document fingerprinting. Proceedings of the 2003 ACM SIGMOD international conference on Management of data. San Diego, California, USA, 2003. P. 76–85.

23. Шутеєв Д. С., Левіков Ю.В. Порушення авторських прав великими мовними моделями // Інформаційні технології: теорія і практика: Матеріали III (IX) Міжнародної інтернет-конференції здобувачів вищої освіти і молодих учених (Харків-Запоріжжя-Дніпро, 25–27 березня 2026 р.) / М-во освіти і науки України, ХНУМГ ім. О. М. Бекетова [Електронний ресурс] Електрон. дані. – Дніпро : Свідлер А.Л., 2026. - С.406-409. - Режим доступу: https://knit.kname.edu.ua/en/2-uncategorised/324-conference-2026-en

24. Липовська О.Г., Завада О.Г. Державне управління у сфері охорони праці в Україні. – Електронне видання «Державне управління: удосконалення та розвиток» – №9, 2021. – 10.32702/2307-2156-2021.9.71.

25. Ліщук М.Є., Московчук А.Т. (2020). Система управління охороною праці в Україні: аналіз стану та перспектив її реформування. Економічний форум. 1. 66-74. 10.36910/6775-2308-8559-2020-2-8.

26. Класифікація небезпечних та шкідливих виробничих факторів. – Режим доступу: https://oppb.com.ua/articles/klasyfikaciya-nebezpechnyh-i-shkidlyvyh-vyrobnychyh-faktoriv.

27. Посібник з керування ризиками на робочому місці. – Режим доступу: https://ohoronapraci.kiev.ua/article/news/posibnik-z-keruvanna-rizikami-na-robocomu-misci.

28. Управління ризиками на виробництві: стратегії зниження, усунення та контролю, розгляд конкретних прикладів. – Режим доступу: https://oppb.com.ua/articles/upravlinnya-ryzykamy-na-vyrobnytstvi-strategiyi-znyzhennya-usunennya-ta-kontrolyu-rozglyad-konkretnyh-prykladiv.

29. Загальні заходи та засоби нормалізації параметрів мікроклімату. – Режим доступу: https://oppb.com.ua/news/zagalni-zahody-ta-zasoby-normalizaciyi-parametriv-mikroklimatu.

# ДОДАТКИ

# ДОДАТОК А

Модуль алгоритму шинглів та коефіцієнта Жаккара на Python

```
import hashlib
import re


class ShingleAntiPlagiarism:
    """
    Модуль алгоритму шинглів та коефіцієнта Жаккара для виявлення плагіату.
    Забезпечує очищення, токенізацію, хешування та компаративний аналіз текстів.
    """
    def __init__(self, shingle_size=3):
        """
        Ініціалізація алгоритму.
        :param shingle_size: Кількість слів в одному шинглі (рекомендовано від 3 до 5).
        """
        self.shingle_size = shingle_size
        # Список стоп-слів української мови для канонізації тексту
        self.stop_words = {
            'і', 'та', 'в', 'на', 'з', 'що', 'як', 'це', 'до', 'у', 'для',
            'про', 'але', 'чи', 'або', 'від', 'за', 'під', 'по', 'при'
        }

    def _canonicalize_text(self, text):
        """
        Внутрішній метод канонізації: очищення від пунктуації,
        нижній регістр та видалення стоп-слів.
        """
    # Приведення до нижнього регістру
        text = text.lower()
        # Видалення всіх розділових знаків та спецсимволів
        text = re.sub(r'[^\w\s]', '', text)
        words = text.split()

        # Фільтрація стоп-слів
```

```
        cleaned_words = [w for w in words if w not in self.stop_words]
        return cleaned_words

    def generate_shingle_hashes(self, text):
        """
        Генерація множини MD5-хешів шинглів для вхідного тексту.
        """
        words = self._canonicalize_text(text)
        shingles = set()

        num_shingles = len(words) - self.shingle_size + 1
    # Якщо текст коротший за розмір шингла, беремо його повністю як один шингл
        if num_shingles <= 0:
            if words:
                shingle_str = " ".join(words)
                shingle_hash = hashlib.md5(shingle_str.encode('utf-8')).hexdigest()
                shingles.add(shingle_hash)
            return shingles

        # Алгоритм рухомого вікна (Windowing)
        for i in range(num_shingles):
            # Вирізаємо вікно довжиною shingle_size слів
            shingle_str = " ".join(words[i:i + self.shingle_size])
            # Хешуємо рядок через MD5 для швидкого порівняння та економії пам'яті
            shingle_hash = hashlib.md5(shingle_str.encode('utf-8')).hexdigest()
            shingles.add(shingle_hash)
    return shingles

    def calculate_jaccard_similarity(self, text_a, text_b):
        """
        Розрахунок коефіцієнта схожості двох текстів за формулою Жаккара.
        :return: Відсоток схожості від 0.0 до 100.0
        """
        # Генерація відбитків (hashes) для обох текстів
        hashes_a = self.generate_shingle_hashes(text_a)
        hashes_b = self.generate_shingle_hashes(text_b)
```

```
        # Якщо обидва тексти порожні після очищення
            if not hashes_a and not hashes_b:
                return 100.0
            # Якщо один з текстів порожній
            if not hashes_a or not hashes_b:
                return 0.0

            # Обчислення математичного перетину множин (Intersection: A ∩ B)
            intersection = hashes_a.intersection(hashes_b)

            # Обчислення математичного об'єднання множин (Union: A ∪ B)
            union = hashes_a.union(hashes_b)

            # Формула Жаккара: (Потужність Перетину / Потужність Об'єднання) * 100%
            jaccard_index = len(intersection) / len(union)

            return round(jaccard_index * 100, 2)
        # --- ДЕМОНСТРАЦІЯ ТА ТЕСТОВИЙ СЦЕНАРІЙ ---
        if __name__ == "__main__":
            # Створюємо екземпляр класу із розміром шингла N=3 слів
            detector = ShingleAntiPlagiarism(shingle_size=3)

            # 1. Оригінальний текст автора
            original_text = (
                "Методи виявлення та запобігання порушенню авторських прав у веб-середовищі "
                "вимагають комплексного підходу. Необхідно поєднувати сучасні математичні "
                "алгоритми аналізу даних та локальні бази даних для захисту конфіденційності."
            )
        # 2. Текст порушника (спроба обходу шляхом перестановки речень та слів місцями)
            pirated_text = (
                "Комплексного підходу вимагають методи виявлення та запобігання порушенню "
                "авторських прав у веб-середовищі. Для захисту конфіденційності необхідно "
```

```
        "поєднувати локальні бази даних та сучасні математичні алгоритми аналізу даних."
        )

        # 3. Абсолютно інший легальний текст для перевірки відсутності хибних спрацьовувань
        clean_text = (
        "Сьогодні студенти четвертого курсу активно працюють над написанням "
        "своєї бакалаврської дипломної роботи та готуються до успішного захисту."
        )
    # Виклик розрахунків
        similarity_plagiarism = detector.calculate_jaccard_similarity(original_text, pirated_text)
        similarity_clean = detector.calculate_jaccard_similarity(original_text, clean_text)

        # Виведення результатів
        print("=== РЕЗУЛЬТАТИ ТЕСТУВАННЯ АЛГОРИТМУ ===")
        print(f"Схожість з модифікованим текстом (рерайт/плагіат): {similarity_plagiarism}%")
        print(f"Схожість з унікальним стороннім текстом: {similarity_clean}%")
```

# ДОДАТОК Б

Метрика Левенштейна на Pyton

```
def levenshtein_distance(s1, s2):
    """
    Обчислення редакційної відстані Левенштейна між двома рядками.
    """
    m, n = len(s1), len(s2)

    # Створення матриці розміром (m+1) x (n+1)
    matrix = [[0] * (n + 1) for _ in range(m + 1)]

    # Ініціалізація першого рядка та першого стовпця
    for i in range(m + 1):
        matrix[i][0] = i
    for j in range(n + 1):
        matrix[0][j] = j

    # Заповнення матриці
    for i in range(1, m + 1):
        for j in range(1, n + 1):
            if s1[i-1] == s2[j-1]:
                cost = 0
            else:
                cost = 1

            matrix[i][j] = min(
                matrix[i-1][j] + 1,      # Видалення
```

```
                matrix[i][j-1] + 1,      # Вставка
                matrix[i-1][j-1] + cost  # Заміна
            )

    return matrix[m][n]


def similarity_percentage(s1, s2):
    """
    Переведення відстані Левенштейна у відсоток схожості рядків.
    """
    distance = levenshtein_distance(s1, s2)
    max_len = max(len(s1), len(s2))
    if max_len == 0:
        return 100.0
    return round((1 - distance / max_len) * 100, 2)


# Демонстрація
word1 = "авторське"
word2 = "авторська"
print(f"Відстань Левенштейна: {levenshtein_distance(word1, word2)}")
print(f"Схожість: {similarity_percentage(word1, word2)}%")  # Повинно бути 88.89%
```

# ДОДАТОК В

Косінусна схожість. Реалізація на Python (з використанням бібліотеки scikit-learn)

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

def calculate_cosine_similarity(text1, text2):
    """
    Обчислення косинусної схожості між двома текстами за допомогою TF-IDF.
    """
    # Створюємо векторизатор, який рахує важливість слів (TF-IDF)
    vectorizer = TfidfVectorizer()

    # Перетворюємо тексти у матрицю фіч (вектори)
    tfidf_matrix = vectorizer.fit_transform([text1, text2])

    # Обчислюємо косинусну схожість між першим та другим вектором
    similarity_matrix = cosine_similarity(tfidf_matrix[0:1], tfidf_matrix[1:2])

    return round(similarity_matrix[0][0] * 100, 2)

# Тестування
txt_original = "Алгоритми виявлення плагіату ефективно захищають інтелектуальну власність авторів у мережі."
txt_rephrase = "В інтернет-мережі інтелектуальну власність авторів ефективно захищають алгоритми пошуку плагіату."

similarity = calculate_cosine_similarity(txt_original, txt_rephrase)
print(f"Косинусна схожість перефразованого тексту: {similarity}%")
```

# ДОДАТОК Г

## Алгоритм pHash. Реалізація на Python за допомогою бібліотеки imagehash

```
from PIL import Image
import imagehash

def detect_image_infringement(original_path, suspected_path, threshold=5):
    """
    Виявлення несанкційованого використання зображення за допомогою pHash.
    """
    # Завантаження зображень
    img_original = Image.open(original_path)
    img_suspected = Image.open(suspected_path)

    # Генерація перцептивних хешів (на основі DCT)
    phash_original = imagehash.phash(img_original)
    phash_suspected = imagehash.phash(img_suspected)

    # Обчислення відстані Геммінга (різниця між хешами)
    hamming_distance = phash_original - phash_suspected

    print(f"Хеш оригіналу:  {phash_original}")
    print(f"Хеш підозрілого: {phash_suspected}")
    print(f"Відстань Геммінга: {hamming_distance}")

    # Перевірка на порушення авторських прав
    if hamming_distance <= threshold:
```

```
            return f"УВАГА: Виявлено плагіат! Зображення схожі (Дистанція: {hamming_distance})"
        else:
            return "Зображення різні. Порушень не виявлено."

    # Приклад виклику (для тестування у вашому дипломі)
    # print(detect_image_infringement("original.jpg", "pirated_compressed.jpg"))
```

# ДОДАТОК Д

Аналіз AST + Winnowing. Реалізація на Python

```
import ast

def code_to_tokens(code_string):
    """
    Етап 1: Парсинг коду в AST та генерація структурних токенів.
    """
    root = ast.parse(code_string)
    tokens = []

    # Рекурсивно обходимо всі вузли дерева
    for node in ast.walk(root):
        # Ігноруємо назви змінних (Name) та завантаження (Load)
        # Зберігаємо лише типи операцій (цикли, функції тощо)
        node_type = type(node).__name__
        if node_type not in ['Name', 'Load', 'Store']:
            tokens.append(node_type)

    return tokens

def winnowing(hashes, w=4):
    """
    Етап 2: Реалізація алгоритму Winnowing для вибірки мінімальних хешів.
    """
    fingerprints = set()
```

```
    num_hashes = len(hashes)

    if num_hashes < w:
        return fingerprints

    for i in range(num_hashes - w + 1):
        window = hashes[i:i + w]
        min_val = min(window)
        # Знаходимо індекс мінімального значення (крайнього праворуч)
        min_index = i + len(window) - 1 - window[::-1].index(min_val)
        fingerprints.add((min_val, min_index)) # Зберігаємо хеш та його позицію

    return fingerprints

# --- ДЕМОНСТРАЦІЯ ---
code_author = "for i in range(10):\n    print(i)"
code_pirate = "for counter in range(10):\n    print(counter)"

tokens_auth = code_to_tokens(code_author)
tokens_pirt = code_to_tokens(code_pirate)

print(f"Токени автора: {tokens_auth}")
print(f"Токени пірата: {tokens_pirt}")
print(f"Чи збігаються структури AST? {tokens_auth == tokens_pirt}")
```