Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра комп'ютерних наук

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему: «Інформаційна технологія визначення унікальності текстового документу»

Виконав: студент 2 курсу,
групи 2КН-20м
спеціальності 122 «Комп'ютерні науки»
Кучевський Ю.А.

Керівник: PhD, проф. кафедри КН
Савчук Т.О.

Рецензент: к.т.н., доц. кафедри ПЗ
Коваленко О.О.

Вінниця
2021

ЗАТВЕРДЖУЮ
Завідувач кафедри____КН____
д.т.н., проф. Яровий А.А.

_____
(підпис)
“____” _____2021 року

ЗАВДАННЯ

на магістерську кваліфікаційну роботу на здобуття кваліфікації магістра зі спеціальності: 122 – «Комп'ютерні науки»

08-22.МКР.031.18.000

Магістранта групи 2КН-20м Кучевського Юрія Андрійовича

Тема магістерської кваліфікаційної роботи: «Інформаційна технологія визначення унікальності текстового документу»

Вхідні дані: обсяг документу, що аналізується на унікальність, сторінки: не більше ніж 40; типи форматів документів, що підтримуються додатком: doc, docx; потужність множини правил перевірки на унікальність, не менше 15 правил; час перевірки однієї роботи на унікальність, не більше 20 секунд; допустимий відсоток плагіату: не більше 20 відсотків, мова програмування – ООП, повинна забезпечувати можливість маніпулювання даними і управління базами даних.

Короткий зміст частин магістерської кваліфікаційної роботи:

1. Графічна: UML-діаграма активності удосконаленого алгоритму, UML-діаграма взаємодії модулів системи, схема алгоритму перевірки текстів на плагіат, IDEF0 діаграма обробки клієнтського запиту, діаграма послідовності модуля обчислення унікальності текстового документу.

2. Текстова (пояснювальна записка): вступ, обґрунтування доцільності розробки інформаційної технології перевірки текстів на унікальність; математична модель процесу перевірки текстів на унікальність, структурна організація та особливості програмної реалізації технології перевірки текстів на унікальність, економічна частина, висновки, перелік використаних джерел, додатки.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ МКР

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		початок	кінець	
1	Обґрунтування доцільності розробки			Розділ 1
2	Моделювання інформаційної технології визначення унікальності текстового документу			Розділ 2
3	Структурна організація та особливості програмної реалізації розробки			Розділ 3
4	Підготовка економічної частини			Розділ 4
5	Апробація результатів дослідження			тези доповідей/акт впровадження
6	Оформлення пояснювальної записки, графічного матеріалу та презентації			Пояснювальна записка, графічний матеріал, презентація

Консультанти з окремих розділів магістерської кваліфікаційної роботи

1. Науковий керівник _____ PhD, професор кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
"____" _____20___ р. _____Т. О. Савчук_____
ініціали та прізвище

2. Економічна частина _____ канд. екон. наук, доц., доц. каф. ЕПВМ
(підпис) наук. ступінь, вчене звання (посада)
_____Небава М.І._____
ініціали та прізвище
"____" _____20___ р.

Дата попереднього захисту роботи "____" _____20___ р.

Рецензент _____ к.т.н., доц. кафедри ПЗ_____
(підпис) наук. ступінь, вчене звання (посада)
О.О. Коваленко
ініціали та прізвище

Завдання видав
науковий керівник _____ PhD, професор кафедри КН
(підпис) наук. ступінь, вчене звання (посада)
_____Т. О. Савчук_____
ініціали та прізвище
"____" _____20___ р.

Завдання отримав магістрант _____ ____Ю. А. Кучевський_____
(підпис) ініціали та прізвище
"____" _____20___ р.

АНОТАЦІЯ

У магістерській кваліфікаційній роботі запропоновано інформаційну технологію визначення унікальності текстового документу на підставі аналізу сучасних методів та технологій призначених для перевірки документів на унікальність.

Інформаційна технологія характеризується підвищеною точністю за рахунок використання коефіцієнту Жаккарда, також високою продуктивністю, відмовостійкістю та масштабованістю за рахунок використання розподіленої архітектури, розроблено програмне забезпечення, що реалізує запропонований алгоритм.

Інформаційну технологію визначення унікальності текстового документу реалізовано засобами мови програмування C# та інтегрованого середовища розробки Microsoft Visual Studio.

ABSTRACT

In the master's qualification the information technology for calculation document’s uniqueness was proposed based on analysis of modern methods and approaches for checking txt uniqueness.

The proposed information technology is accounted for having higher precision due to the use of Jaccard similarity and high productivity, fault tolerance, scalability by utilizing the concept of distributed architecture, developed software that implements the algorithm mentioned above.

The information technology had been written in C# programming language. Also Microsoft Visual Studio IDE was used.

ЗМІСТ

ВСТУП **7**

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ **10**

1.1 Аналіз існуючих антиплагіат методів **10**

1.2 Програмні засоби для перевірки текстів на унікальність **19**

1.3 Постановка задачі дослідження **20**

2 РОЗРОБКА УДОСКОНАЛЕНОГО МЕТОДУ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ **22**

2.1 Математична модель визначення унікальності текстового документу **22**

2.2 Удосконалення методу визначення унікальності текстового документу **24**

2.3 Розробка алгоритму визначення унікальності текстового документу для txt файлів **27**

2.4 Розробка алгоритму визначення унікальності текстового документу для doc файлів **30**

2.5 Висновки **32**

3 СТРУКТУРНА ОРГАНІЗАЦІЯ ТА ОСОБЛИВОСТІ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ **33**

3.1 Розробка модуля клієнтського інтерфейсу **35**

3.2 Розробка модуля обчислення унікальності документу **44**

3.3 Розробка серверного модуля **46**

3.4 Розробка модуля бази даних **49**

3.5 Аналіз результатів функціонування інформаційної технології визначення унікальності текстового документу **51**

3.6 Висновок **57**

4 ЕКОНОМІЧНА ЧАСТИНА **58**

4.1 Оцінювання комерційного потенціалу розробки **58**

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи 61
4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором 67
ВИСНОВКИ 73
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 75
ДОДАТКИ ..... Ошибка! Закладка не определена.
Додаток А ..... Ошибка! Закладка не определена.
Інструкція користувача ..... Ошибка! Закладка не определена.
Додаток Б ..... Ошибка! Закладка не определена.
Лістинг вихідного коду ..... Ошибка! Закладка не определена.
Додаток В ..... Ошибка! Закладка не определена.
Графічна частина ..... Ошибка! Закладка не определена.

ВСТУП

Актуальність теми. Стрімкий розвиток Інтернету разом із зростанням комп'ютерної грамотності сприяє проникненню плагіату в різні сфери людської діяльності: плагіат є насущою проблемою в промисловості, освіті та наукових спілках.

Відповідно до [1] під плагіатом вважають умисно вчинене окремою особою незаконне використання чи розпорядження охоронюваними результатами чужої творчості, що супроводжується донесенням до інших осіб неправдивих відомостей про себе як про реального автора. Плагіат може бути порушенням законодавства про авторські права та патенти і як такий може призвести до юридичної відповідальності. З іншого боку, плагіат можливий в областях, які не охоплені жодним типом інтелектуальної власності, наприклад, математика та інші базові наукові дисципліни.

Плагіат проявляється у виданні під власною назвою чужого твору, а також у запозичення фрагментів чужих творів без зазначення джерела запозичення. Обов'язковою ознакою плагіату є привласнення авторства, оскільки неправомірне використання, публікація, копіювання та захищений авторським правом твір сам по собі є не плагіатом, а іншим видом порушення авторських прав, який часто називають піратством. Піратство стає плагіатом, коли результати інтелектуальної роботи використовуються неправомірно, а атрибуція публікується особою автора.

Плагіат із появою Інтернету став серйозною проблемою. Потрапивши в Інтернет, знання стають надбанням усіх, захистити авторські права стає все важче, а іноді й неможливо. Тому перевірка унікальності документів є актуальним завданням.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напрямку наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Розробка спеціалізованих засобів штучного інтелекту на основі інтелектуального аналізу даних та машинного навчання».

Мета та завдання дослідження Метою даної магістерської роботи є підвищення точності перевірки текстів на унікальність.

Для досягнення поставленної мети необхідно розв'язати такі завдання:

- аналіз та обґрунтуваня доцільності розробки інформаційної технології перевірки текстів на унікальність;
- аналіз та вибір методів та технологій для перевірки текстів на унікальність;
- розробити математичну модель перевірки текстів на унікальність, удосконалений алгоритм перевірки текстів на унікальність, інформаційну технологію визначення унікальності текстового документу;
- провести тестування програми та проаналізувати отримані результати;
- економічно обґрунтувати доцільність перевірки текстів на унікальність.

Об'єкт дослідження – процес перевірки текстів на унікальність.

Предмет дослідження – технології перевірки текстових документів на унікальність.

Методи дослідження. У роботі використані такі методи наукових досліджень: аналіз інформаційної системи антиплагіат, методи оброблення текстової інформації, методи взаємодії між серверами, методи комунікації з базою даних, об'єктно-орієнтованого програмування для отримання подальших результатів роботи програми.

Наукова новизна одержаних результатів полягає в наступному:

- розроблено математичну модель процесу визначення унікальності текстового документу, що відрізняється наявність схожості Жаккарда, що дозволило підвищити точність процесу визначення унікальності текстового документу;
- удосконалено метод перевірки текстів на унікальність за рахунок обрахування схожості Жаккарда для документів з найбільшими співпадіннями, що дозволило підвищити точність методу визначення унікальності текстового документу. Також використовуються різні алгоритми початкової канонізації текстової інформації залежно від розширення файлу, що дозволяє обробляти різні формати текстів;

– удосконалено інформаційну технологію перевірки текстів на унікальність, яка відрізняється введенням коефіцієнту унікальності, що збільшує точність обрахунку .

Практичне значення одержаних результатів полягає у наступному:

1. Розроблено структуру організації системи;
2. Розроблено удосконалений алгоритм визначення унікальності текстового документу;
3. Розроблено алгоритм роботи серверверної частини;
4. Розроблено алгоритм роботи клієнтської частини.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується строгістю постановки задач, коректним застосуванням математичних методів під час доведення наукових положень, чітким та лаконічним виведенням аналітичних співвідношень, порівнянням результатів з відомими, та збіжністю результатів математичного моделювання з результатами, що отримані під час впровадження розроблених програмних засобів.

Особистий внесок магістранта. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані самостійно.

Апробація результатів роботи. Результати досліджень було апробовано на міжнародній науково-практичній конференції ІОН 2020 [1]; та на науково-технічній конференції факультету інформаційних технологій та комп'ютерної інженерії (2020) [2], науково-технічній конференції Вінницького національного технічного університету факультету інформаційних технологій та комп'ютерної інженерії м. Вінниці у 2021 р [3], міжнародній науково-практичній конференції RECENT SCIENTIFIC INVESTIGATION в Осло [4].

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано 4 тез доповідей на міжнародних конференціях. Також на розробку зареєстровано 2 авторських права [5, 6] та опубліковано статтю в журналі «Таврійський науковий вісник» [7].

1 ОБҐРУНТУВАННЯ ДОЦІЛЬНОСТІ РОЗРОБКИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ

1.1 Аналіз існуючих антиплагіат методів

Сучасні підходи перевірки документів на унікальність характеризуються по типу оцінки подібності. Глобальна оцінка використовує великі фрагменти тексту або документа, щоб знайти спільні риси, а локальні методи введення перевіряють обмежений сегмент тексту.

В даний час одним з найпоширеніших підходів є дактилоскопія, суть якого полягає в наступному: набір документів вибирається з декількох документів, які є відбитками. Цей документ буде порівнюватися з «відбитками» для всіх документів збірки. Знайдені відповідності коїться з іншими документами вказують на загальні частини тексту [6]. Перевірка документа з буквального накладання тексту – це класичне порівняння рядків. Перевірка підозрілих документів у цій ситуації вимагає підрахунку та зберігання ефективно порівнянних уявлень усіх документів у довідковій колекції, що порівнюються попарно [7]. Зазвичай є використовуваними моделі, як дерево суфіксів або перелік суфіксів, адаптованих для вирішення цієї задачі в контексті програмованого визначення плагіату. Перевагою означеного підходу є відносно проста модель реалізації [8]. Однак зіставлення підрядків є нежиттєздатним рішенням для тестування великих масивів документів (відповідно до досліджень алгоритм в середньому працює за 2h порівнянь, де h - довжина рядка, в якому виконується пошук), що є чималим недоліком [9].

Аналіз «кошика слів» - це спрощене уявлення, що використовується при обробці природної мови та пошуку інформації [10]. В даній моделі текст представлений у вигляді неупорядкованого набору слів. Документи представлені як один чи декілька векторів, використовуваних для попарного обрахування схожості [11]. Приклад реалізації: такі моделі створюють

текстовий документ із використанням кошика слів. Наприклад ці два простих текстових документи:

(1) Джон любить дивитися фільми. Марія теж любить фільми.
(2) Джон також любить дивитися футбольні матчі.

На основі цих двох текстових документів, для кожного документа будується список таким чином:

(1) «Джон», «любить», «дивитися», «фільми», «Марія», «любить», «фільми», «теж».

(2) «Джон», «також», «любить», «дивитися», «футбольні», «матчі».

Кожен масив пізніше стає об'єктом, де ключем виступатиме слово з кошику, а за значення буде використовуватись кількість разів, під яким ключ з'являється в рядку. При цьому порядок елементів може бути вільним. Насправді модель «корзини слів» використовується в основному як інструмент для генерації ознак, що корисно при порівнянні двох текстових фрагментів на унікальність [12, 13]. Перетворюючи текст на «кошик слів», можна формувати різні заходи, що характеризують текст. Найбільш поширеним типом характеристик або ознак, що розраховуються за моделлю «корзини слів», є частота використання термінів, а саме, скільки разів термін зустрічається в тексті [14, 15].

Модель кошика слів - це невпорядкований документ, у якому важливо лише кількість слів. Наприклад, у наведеному вище прикладі «Джон любить дивитися фільми. Марія теж любить кіно», уявлення кошика слів не означатиме, що дієслово «любити» завжди слідує за ім'ям людини в цьому тексті [16]. Недолік цього алгоритму в тому, що він не враховує відносин між словами. Як альтернатива, модель n-грам може зберігати цю просторову інформацію. Застосовуючи цей приклад, модель біграм розбере текст на одиниці і збереже частоту кожної одиниці, як раніше [17].

Поширеною альтернативою використанню словників є хеш-трюк, коли слова пов’язані безпосередньо з індексами за допомогою хеш-функції. З таким підходом, для зберігання словника не потрібне використання пам'яті. Конфлікти хешування зазвичай виправляються шляхом вивільнення пам'яті і збільшення кількості сегментів хешування. На практиці підхід хешування значно спрощує програмну реалізацію моделі кошику слів та значно підвищує масштабованість.

Цитування - це автоматизований метод виявлення плагіату, розроблений для використання в наукових документах, що дозволяє цитувати та посилання. Визначає загальні цитати двох праць. Шаблон цитати - це підпослідовність, що містить не тільки загальні цитати для двох документів, але також аналогічний порядок та близькість цитат у тексті, які є основними критеріями визначення шаблону цитати. Перевагою цього є можливість вузької реалізації отримання інформації про автора висловлювань і висновків. Крім того, цей метод менш вимогливий до обчислень, ніж зняття відбитків пальців. Тиждень становить труднощі при масштабуванні великої кількості документів [9].

Стилометрія, або вивчення мовних стилів, – це статистичний метод атрибуції анонімних документів та комп'ютерних перевірок на плагіат. Стилометричні моделі будуються для різних фрагментів тексту, фрагментарно, стилістично відмінних від інших. А при порівнянні моделей можна виявити плагіат. Перевага цього методу – виявлення плагіату навіть у тому випадку, якщо текст був повністю змінено. Недоліком є велика складність реалізації та необхідність наявності в навчальній базі якнайбільшої кількості робіт конкретних авторів для досягнення прийнятної точності пошуку оригінального автора за стилем [10].

Аналіз на основі послідовностей мовних частин. Розглядається спосіб розбиття тексту однорідні фрагменти. Як параметри поділу приймаються різні послідовності частин мови. Потім фрагменти аналізуються. І в результаті з'являються послідовності для тексту, що витягують із текстів фрагменти, тобто

алгоритм, що витягує з тексту фрагменти неоднорідності, що мають різну частоту отримання обраної послідовності мовних частин, що вказує на можливий плагіат. тут [18].

Аналіз шинглів - це алгоритм нечіткого пошуку тексту, що повторюється. Слово нечіткий означає, що входження дублікатів шукається не точно, а розмито. Наприклад, ви можете дублювати не лише рядки, а й окремі фрази. Здебільшого модифікація алгоритму shingle використовується системами антиплагіату, пошуковими системами боротьби зі спамом, копіпастом, і навіть визначення унікальності перезапису [23].

Шингл - це окремі фрагменти (підрядки), які вибираються порівняння з основного текст, з певною кількістю слів у його послідовності для перевірки на унікальність [21]. Шингли можуть складатися з будь-якої кількості слів, чим коротший шингл, тим точніше буде результат перевірки, але в той же час тим більше обчислювальних ресурсів буде витрачено під час процесу перевірки текстового документу на унікальність. Наразі є такі методи розбиття інформації на шингли:

- послідовний, фрагменти взагалі не перетинаються (рис. 1.1).

- перетинний, коли частини рядка включають певну частину попереднього підрядка (рис. 1.2)

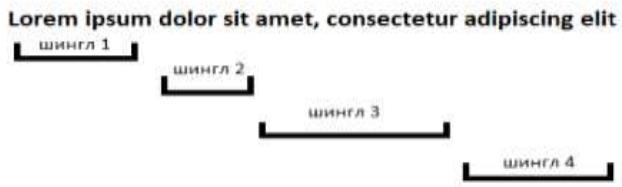

Рисунок 1.1 – Зображення розбиття текстової інформації на шингли послідовним шляхом

Рисунок 1.2 – Зображення розбиття текстової інформації на шингли перетиним шляхом

Спосіб утворення шинглів та кількість символів у шинглі, а також їх зсув (на яку кількість слів та символів зсувається наступна стрічка) значною мірою впливає на точність. При визначенні розмірності підрядка вибір великою мірою залежить від наявної обчислювальної потужності, розміру пам’яті та необхідної для досягнення точності результатів [25].

Після поділу рядка на шингли існують також різні підходи щодо обчислення контрольних сум та їх подальшого порівняння попарно для оцінки рівня схожості двох текстів. Контрольні чек-суми можна отримати шляхом хешування використовуючи різні алгоритми, такі як SHA1, SHA3, CRC32, MD5. Після цього необхідно оцінити збіг наявних чек-сум для двох текстів [26].

Для ефективного порівняння потрібно встановити правильні параметри алгоритму. Чим коротший шингл, тим більш точно будуть знайдені відповідні слова. Крім того, зі зсувом менша ймовірність «перестрибування» повторюваних фраз. Тим не менш, чим довший текст, тим легше знайти в ньому збіги, якщо такі наявні, і немає необхідності обирати невелике значення довжини шинглу. Варто підкреслити, що точність обробки текстової інформації пропорційно росте до кількості затрачених обчислювальних потужностей[27].

Часто пишуть, що алгоритм шинглів не може визначити ідентичність таких фраз, як «Вчитель дає матеріал учневі / Викладач дає матеріал учневі». Багато послуг перевірки на плагіат, засновані на алгоритмі шинглів, покажуть, що фрази унікальні, хоча для пошукових систем вони ідентичні. Якщо при канонізації використовується морфологія, іншими словами всі слова приводяться до їхньої нормальної форми, алгоритм без проблем здатний розпізнати фрази які є однаковими, не дивлячись на їх кінцівки [28].

Такий підхід підвищує точність аналізу текстових творів на плагіат, тому часто використовується при розробці сучасних програмних систем перевірки унікальності. Обмеження цього алгоритму є у тому, що за наявності чималої бази даних текстових документів попарне порівняння матиме невелику продуктивність.

Таким чином, цей підхід дозволяє налаштовувати точність алгоритму залежно від доступних ресурсів, а за рахунок зберігання хешованих неоднорідностей у високопродуктивних базах даних, таких як ключ-значення, може усунути основний недолік існуючих підходів та засобів перевірки текстових документів на унікальність, а саме неможливість оцінити дані документа із прийнятною швидкістю на великих обсягах інформації.

Використання методу шинглів для перевірки текстових документів на унікальність підвищує точність результатів аналізу та дає можливість налаштувати алгоритм для оптимізації використання обчислювальних ресурсів за рахунок збільшення кількості фрагментів неоднорідності, що містяться у шинглі. Цей метод не вимагає від користувача ніяких додаткових операцій крім надання відповідного документа для подальшого аналізу, що автоматизує процес аналізу.

Класичний алгоритм перевірки текстових робіт на унікальність методом шинглів включає наступні етапи:

1. Вибір файлу.

2. Зчитування файлу.

3. Нормалізація тексту шляхом конвертації до одного регістру та прибиранням знаків пунктуації.

4. Розбиття текстової інформації на шингли з фіксованою довжиною.

5. Калькуляція результату хеш-функції за певним шинглом для кожного окремо взятого фрагменту неоднорідності.

6. Витягнення існуючих документів з бази даних.

7. Для кожної чек-суми поточного документу провести перевірку чи є ідентичний в існуючі колекції шляхом попарного порівняння. В разі співпадіння потрібно зберегти джерело ідентичної чек-суми.

8. Побудова результату обчислення у вигляді користувацького звіту за допомогою довідника, що був побудований раніше.

9. Представлення результату.

Стандартний вид цього алгоритму передбачає порівняння кожного шинглу кожного документу попарно, що і є його основним недоліком. Нехай

n – кількість текстових документів в наявній базі даних,

m – кількість слів в документах (усереднена),

k– фіксована довжина шинглу.

Отже:

$\frac{m}{k}$– усереднена кількість шинглів в текстовому документі,

$\frac{n \times m}{k}$ – загальна кількість шинглів.

Тоді:

$\frac{n \times m^2}{k^2}$– праблизна загальна кількість попарних порівнянь.

Зазвичай k не є маленьким числом і є частиною множини чисел $Z = \{1 \dots 10\}$, в іншому разі точність алгоритму стала б настільки малою, що подальший аналіз не мав би жодного сенсу. У такому разі при обрахунку складності алгоритму доцільно буде вважати k як константу. Отож:

$O(n \times m^2)$ є складністю означеного вище алгоритму і підтверджує описаний недолік;

$O(\frac{n \times m}{k})$ є обсягом пам’яті необхідної для зберігання текстових файлів.

Наведені етапи відображаються в схемі алгоритму класичного алгоритму з використанням методу шинглів для верифікації текстових файлів на унікальність (рис. 1.3).

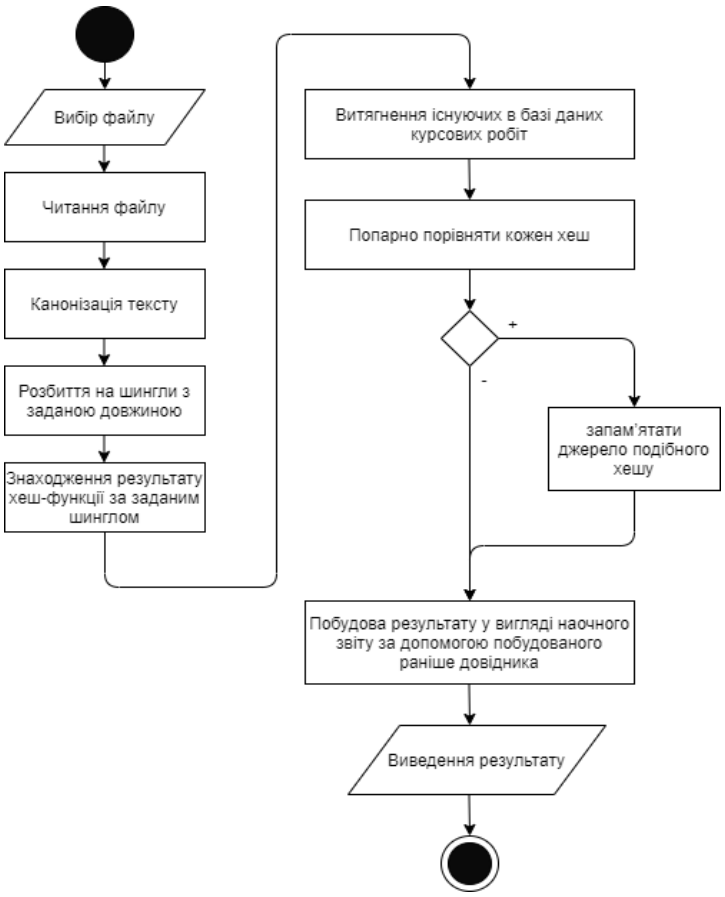

Рисунок 1.3 – Діаграма активності процесу перевірки текстів на плагіат

Таким чином в результаті порівняння наведених методів перевірки текстових документів на унікальність було виявлено їх характеристики, що

наведені у таблиці 1.1. Однією з важливих характеристик яку необхідно покращити є точність. Алгоритм шинглів є доцільним фундаментальним методом для подальших покращень.

Таблиця 1.1 – Переваги та недоліки розглянутих методів перевірки текстових документів на унікальність

Характеристика	Дактилоскопія	Кошик слів	Шаблон цитат	Шингли	Стильометрія
Простота реалізації	так	так	ні	так	ні
Висока точність	так	ні	ні	ні	ні
Здатність до обробки великих обсягів документів	ні	так	ні	так	ні
Виявлення плагіату в повністю зміненому тексті	ні	ні	ні	ні	так
Може працювати на невеликій базі	так	так	ні	так	ні
Врахування зв’язку між словами	ні	ні	так	ні	так

Програмні засоби для перевірки текстів на унікальність

Розглянемо існуючі програмні засоби перевірки текстових документів на унікальність.

Систему Antiplagiat розробила компанія Forexis [19]. Система здійснює онлайн-пошук по великій кількості документів, що зберігаються в базі системи, в базах даних партнерів, у тому числі: науковій електронній бібліотеці ELibrary.ru, Lexpro, а також у базі даних користувачів. Antiplagiat шукає в Інтернеті своїми засобами і тому менш ефективний, ніж системи, що використовують Yandex XML. У тріал версії системи доступна лише скорочена форма звіту. Перевагою системи є висока точність порівняння, велика база документів і партнерів, можливість пошуку в Інтернеті. Недоліком є обмежена можливість додавання власної бази документів.

Сервіс Unplag [20] може перевірити на плагіат як в режимі реального часу онлайн, так і порівняти документ із збереженою базою документів у бібліотеці користувача. Підтримує роботу з різними типами документів. Є персональна та корпоративна програма. Також працює з системою керування курсами Moodle, Canvas, Blackboard, Sakai. Цей інструмент має перевагу порівняння файлів в Інтернеті та варіювання типів підтримуваних документів. Водночас недоліком є менша документна база і, ймовірно, менша точність.

Система Плагіаінформ перевіряє документи на наявність запозичень як у локальній базі даних, так і в Інтернеті [21]. Перевага такого підходу полягає в тому, що система здатна знаходити плагіат у вигляді документів, що складаються із «змішаних» фрагментів тексту з кількох джерел. Перевірку можна виконати за допомогою швидкого або глибокого пошуку. Результати випробувань представлені у вигляді візуального звіту. Недоліками є відсутність перетворення букв і можливість вільного використання або тестування системи.

Таким чином, у результаті порівняння наведених вище програмних засобів перевірки текстів на унікальність були виявлені наступні переваги та недоліки, які наведено в таблиці 1.2. Слід зазначити, що підвищення точності є доцільним завданням при вдосконаленні засобів визначення унікальності текстового документа.

Таблиця 1.2 – Переваги та недоліки розглянутих засобів перевірки текстових документів на унікальність

Характеристика	Антиплагіат	Unplag	Plagiatinform
Висока точність	так	ні	ні
Редагування бд документів	ні	ні	ні
Велика бд	так	ні	так
Підтримувані формати	так	так	так
Вільне використання і тестування	так	ні	ні
Аналіз заміни знаків	так	так	ні
Різні типи перевірки	ні	так	так
Відловлення комплексного плагіату	ні	ні	так
Пошук у веб	так	так	так

1.2 Постановка задачі дослідження

Нехай дано базу даних оброблених документів. Модифікація бази даних та додавання нових документів виконується користувачем системи.

Нехай задано вхідний вектор X(x1, x2, x3, x4, x5), де

X1 – база оброблених документів.

X2 – документ для оцінення на ступінь унікальності.

X3 – максимальний час обробки одного документу.

X4 – потужність множини правил для перевірки.

X5 – максимальний допустимий відсоток плагіату.

Тоді, задачу аналізу тексту на подібність

подається наступним чином:

$$Y = F(X),$$

В якій Y(y1, y2, y3) – вихідний вектор,

Y1 – коефіцієнт унікальності наданого документу у відсотках.

Y2 – множина документів, що є джерелами інформації.

Y3 – індикатор, що вказує чи документ успішно пройшов перевірку.

2 РОЗРОБКА УДОСКОНАЛЕНОГО МЕТОДУ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ

2.1 Математична модель визначення унікальності текстового документу

Оскільки інформаційна технологія визначення унікальності текстового документу використовує попарне порівняння фрагментів неоднорідності, розглянемо математичну модель визначення унікальності методом шинглів.

Текст документу розбивається на шингли довжиною 1. Проте при подальших обчисленнях ми не повинні враховувати пунктуаційні знаки, сполучники, прийменники тощо.Також при попарному порівнянні шинглу ми маємо оперувати його нормалізованим значенням, а не вхідним. Алгоритм має оперувати тільки тими шинглами, що впливають на якісну оцінку тексту при перевірці. Дану взаємодію можна описати функціональною залежністю r(X2) з двома підмножинати TiA (2.1).

$$r(R),\ T \subseteq R,\ B \subseteq R, \tag{2.1}$$

де X2 – множина всіх шинглів, якімістяться в документі, T – множина нормалізованих шинглів документу, A – множина всіх шинглів в системі.

Подавши множину T як детермінант, а множину A як залежну частину та звівши функціональну залежніть до тривіальної форми отримано формулу (2.2):

$$(T \rightarrow A) \Leftrightarrow ((\forall t_1, t_2 \in r : t_1(A) = t_2(A)) \Rightarrow (t_1(B) = t_2(B))). \tag{2.2}$$

Загальну кількість доступних шинглів можна отримати об’єднавши множини T та A за таким виразом:

$$X2 = A \cup T. \tag{2.3}$$

В даному випадку множина А багатозначно залежить від Т, тоді і тільки тоді коли множина значень А відповідає парі залежностей:

$$[t : T, a:A], r(X2). \tag{2.4}$$

Надалі відбувається процес порівняння шинглів. Алгоритм перевіряє чи шингл, який обробляється є якісно впливовим. Після підтвердження відбувається 2-й етап перевірки шингла, на цей раз вже в базі даний. Система звертається до БД та звіряє отриманий шингл з вмістом кортежівв таблиці фрагментів неоднорідності. Таким чином отриманий шингл, якщо він вже був в попередніх документах має міститись в 3-х підмножинах:

$$(A \rightarrow T \rightarrow V) \Leftrightarrow \forall t_1, t_2 \in r : \begin{cases} t_1(A), \\ t_1(T) = t_2(A), \\ t_1(V) = t_2(A) = t_3(T). \end{cases} \tag{2.5}$$

Після перевірки шинглу, програма відповідно до запиту, отримує від бази даних необхідну інформацію. Для обчислення наочного коефіцієнту унікальності текстового документу використаємо коефіцієнт Жаккарда для кожної пари вхідного документу і документу, що має найбільшку кількість однакових шинглів шляхом виконання наступних обчислень:

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

Отже, розроблено математичну модель процесу визначення унікальності текстового документу з використанням коефіцієнту Жаккарда, що підвищує точність результатів. При відображені результату користувачу необхідно вказати користувачу чи документ успішно пройшов перевірку на плагіат наступним чином:

Y3 = Y1 < X5, де Y1 – коефіцієнт Жаккарда для заданого документу, Y3 – максимально допустимий відсоток плагіату. Також, має задовольнятись рівність

t(X4, R) < X3, де t – функція що повертає час, затрачений на обрахунок унікальності для R (всіх шанглів наданого документу), X3 – максимальний час перевірки одного документу, X4 – потужність множини правил перевірки при визначенні унікальності текстового документу.

2.2 Удосконалення методу визначення унікальності текстового документу

Так як інформаційна технологія повинна обробляти як txt так і doc/docx файли, то методу визначення унікальності текстового документу необхідно обрати відповідний алгоритм залежно від типу документу, наданого користувачем. Це реалізується за допомогою програмного підходу стратегія. Суть стратегії полягає в використанні основного алгоритму який на певному етапі використовує різні алгоритми, що ведуть до одного і того самого результату, проте це досягається різними шляхами. Так як doc/docx файли використовують XML розмітку для опису документу, то для того щоб отримати канонізований текст, необхідно розробити алгоритм, відмінний від обробки txt документу. Отже, необхідно розробити наступні 2 алгоритми:

1. Алгоритм визначення унікальності текстового документу для txt файлів.
2. Алгоритм визначення унікальності текстового документу для doc/docx файлів.

Отже, метод визначення унікальності текстового документу складатиметься з наступних етапів:

1. Читання типу наданого файлу.

Отформатировано: русский

2. Якщо тип наданого файлу txt – перейти до алгоритму визначення унікальності текстового документу для txt файлів.
3. Якщо тип наданого файлу doc/docx – перейти до алгоритму визначення унікальності текстового документу для doc/docx файлів.
4. Виведення звіту користувачеві.

Для txt алгоритму немає необхідності попередньо аналізувати заданий текст, адже цей формат не містить ніяких додаткових метаданих про документ. Він є найпростішим з алгоритмів. Натомість, алгоритм визначення унікальності текстового документу для doc/docx файлу, що належить Office Open XML формату, є складнішим. В ранній формі цих форматів, перед стандартизацією ECMA, «Microsoft Office 2003 XML» використовували єдиний монолітний файл з вбудованими елементами такими як зображення як закодовані блоки всередині XML. Office Open XML більш цього не підтримує, але використовує стиснення файлів відповідно до «Open Packaging Convention». Зокрема, Open XML SDK дозволяє генерувати документи, отримувати інформацію з існуючих документів (у тому числі для перетворення в HTML) і модифікувати існуючі документи. Також в Open XML SDK присутній код перевірки якості, який може бути використаний в системі автоматизованого тестування для перевірки якості підтримки OOXML. Сирцевий код SDK написаний мовою C#.

Отже, даний алгоритм повинен використати Open XML SDK для попереднього перетворення заданого файлу в текст, готовий до аналізу.

Ці етапи зображені в схемі методу для перевірки текстів на унікальність (рис. 2.1).

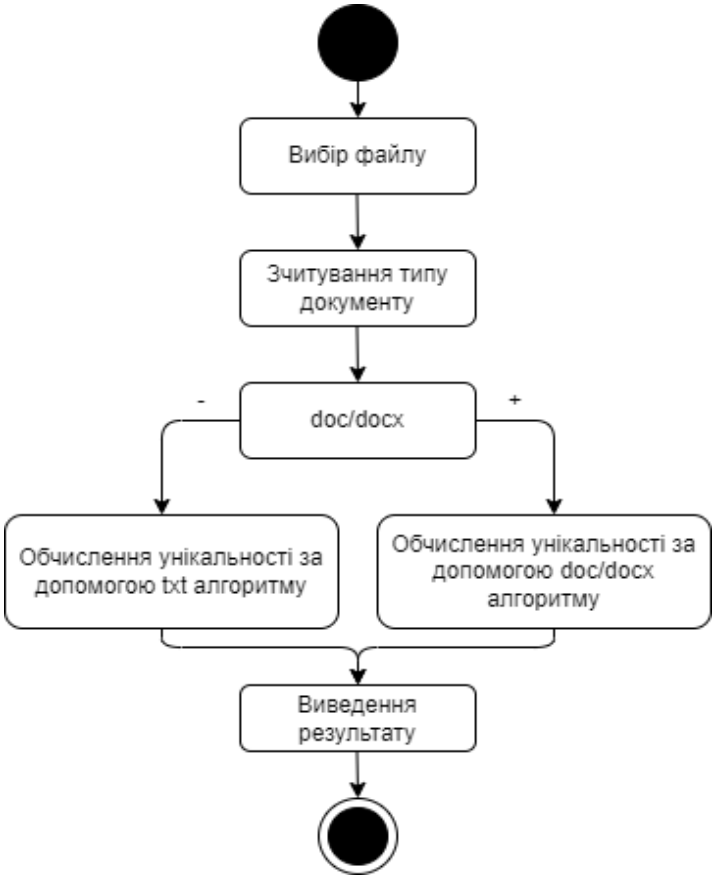

Рисунок 2.1 – Діаграма активності удосконаленого методу перевірки текстів на унікальність

Таким чином, запропоновано удосконалений метод перевірки текстових документів на унікальність, що базується на використанні різних алгоритмів обробки текстових файлів, залежно від типу документу. Інформаційна система визначення унікальності текстового документу підтримує алгоритму для txt та doc/docx файлів. При цьому, кожен з алгоритмів повинен удосконалювати точність відносно стандартного методу шинглів.

2.3 Розробка алгоритму визначення унікальності текстового документу для txt файлів

Розглянемо удосконалення означеного алгоритму з метою забезпечення підвищення точності. Так як файли типу txt не містять ніяких метаданих, то для того щоб отримати текст придатний до аналізу не потрібно вводити попередній етап аналізу текстових даних. Для мінімізації кількості необхідних процесорних потужностей потрібно ввести попередній етап обробки текстового документу, що додається в базу даних текстових документів. Кожен з хешованих фрагментів неоднорідності додається в нереляційне сховище типу "ключ-значення". В сховищі ключем виступатиме сама чек-сума, а значенням – множина всіх ідентифікаторів текстових документів що містять дану чек-суму.

Наприклад, якщо фрагмент word міститься в документах 4, 5, 6, то при аналізі файлу під номером 7 після обробки фрагменту 'word, матиме екземпляр сутності з ключем word, та значенням [4, 5, 6, 7]. Надалі, під час обробки текстового файлу на плагіат замість використання попарних порівнянь всіх шинглів з наданого користувачем файлу, пропонується для кожного з шинглів робити запит на кортеж за ключем і модифікувати змінні що ідентифікатори на джерела плагіату.

Беручи до уваги той факт, що сучасні нереляційні сховища типу ключ-значення гарантують доступ до кортежу за константний час, то можна стверджувати що після введення описаної оптимізації матимемо наступні значення:

$O(m)$ - складність удосконаленого алгоритму,

$O(\frac{n \times m}{k})$ – обсяг затраченої пам'яті.

Щоб підвищити точність, для обчислення наочного коефіцієнту унікальності текстового документу використаємо коефіцієнт Жаккарда для

кожної пари вхідного документу і документу, що має найбільшку кількість однакових шинглів шляхом виконання наступних обчислень:

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

Отже, алгоритм для txt файлів складатиметься з наступних етапів:

1. Розбиття вхідних даних на шингли довжиною 1.
2. Канонізація даних за допомогою морфологічних словників.
3. Видалення допоміжних слів таких як сполучники, прийменники тощо.
4. Для кожного шинглу виконання запиту для отримання множини ідентифікаторів документів що містять такий самий шингл. При цьому для кожного отриманого ідентифікатора оновлюємо довідник в пам’яті, інкрементуючи значення змінної під ключем ідентифікатора.
5. Сортування пар ключ-значення в порядку спадання за значенням.
6. Для перших k пар дістаємо з бази даних текстові документи з відповідними ідентифікаторами.
7. Для кожного документу з бази даних обраховуємо коефіцієнт Жаккарда використовуючи шингли вхідного документу та того, що був отриманий з бази даних.
8. Для відображення наочного коефіцієнту використовуємо коефіцієнт Жаккарда як показник рівня плагіату відносно певного існуючого документу і 1-jaccard як коефіцієнт унікальності.

Ці кроки зображені на діаграмі активності удосконаленого алгоритму перевірки txt файлів документів на унікальність (рис. 2.2).

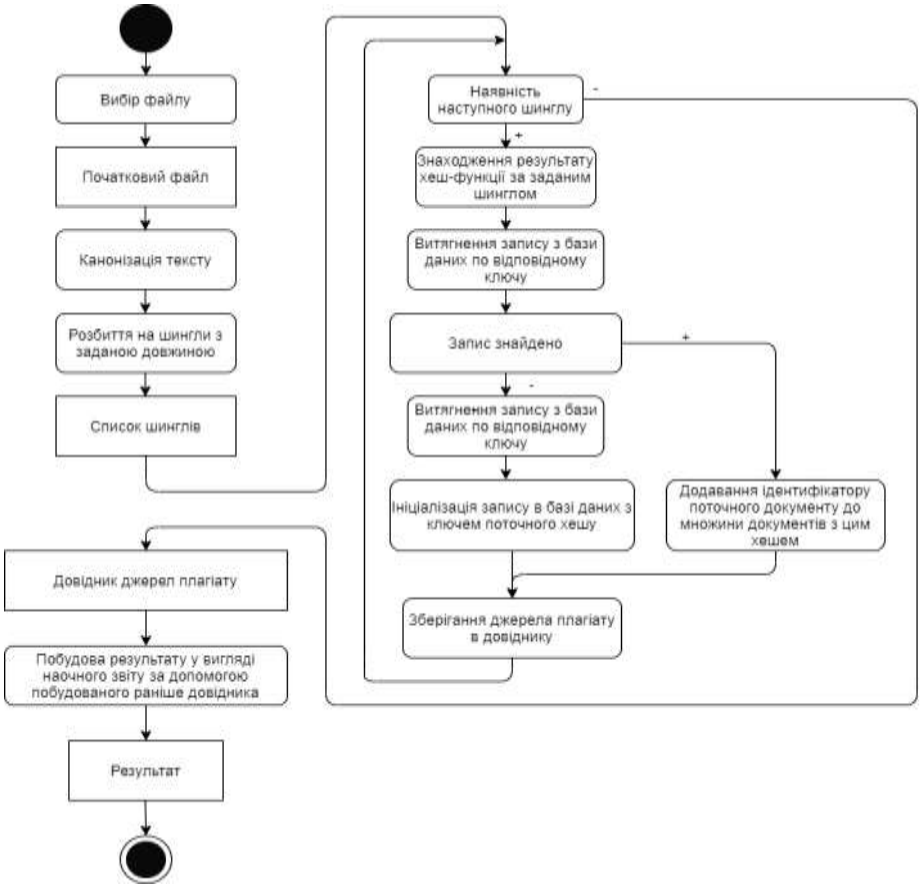

Рисунок 2.2 – Діаграма активності удосконаленого процесу перевірки txt файлів на унікальність

Таким чином, запропоновано удосконалений алгоритм перевірки текстової інформації з txt файлів на унікальність, що базується на використанні попереднього зберігання хешованих частин в нереляційному сховищі типу ключ-значення. Це, в свою чергу, забезпечить чималу швидкість обробки значних обсягів документів. При цьому, обсяг затраченої пам’яті залишатиметься незмінним [1].

2.4 Розробка алгоритму визначення унікальності текстового документу для doc файлів

У випадку, коли використовується файл з розширенням doc чи docx необхідно додати етап попереднього витягнення текстової інформації з документу, адже там використовується XML розмітка для опису документу. Отже, алгоритм для doc та docx файлів складатиметься з наступних етапів:

1. Читання файла та парсинг тексту з doc/docx документу.
2. Розбиття вхідних даних на шингли довжиною 1.
3. Канонізація даних за допомогою морфологічних словників.
4. Видалення допоміжних слів таких як сполучники, прийменники тощо.
5. Для кожного шинглу виконання запиту для отримання множини ідентифікаторів документів що містять такий самий шингл. При цьому для кожного отриманого ідентифікатора оновлюємо довідник в пам’яті, інкрементуючи значення змінної під ключем ідентифікатора.
6. Сортування пар ключ-значення в порядку спадання за значенням.
7. Для перших *k* пар дістаємо з бази даних текстові документи з відповідними ідентифікаторами.
8. Для кожного документу з бази даних обраховуємо коефіцієнт Жаккарда використовуючи шингли вхідного документу та того, що був отриманий з бази даних.
9. Для відображення наочного коефіцієнту використовуємо коефіцієнт Жаккарда як показник рівня плагіату відносно певного існуючого документу і 1-jaccard як коефіцієнт унікальності.

Цей алгоритм перевірки doc та docx документів на унікальність зображень на UML діаграмі активності (рис. 2.3).

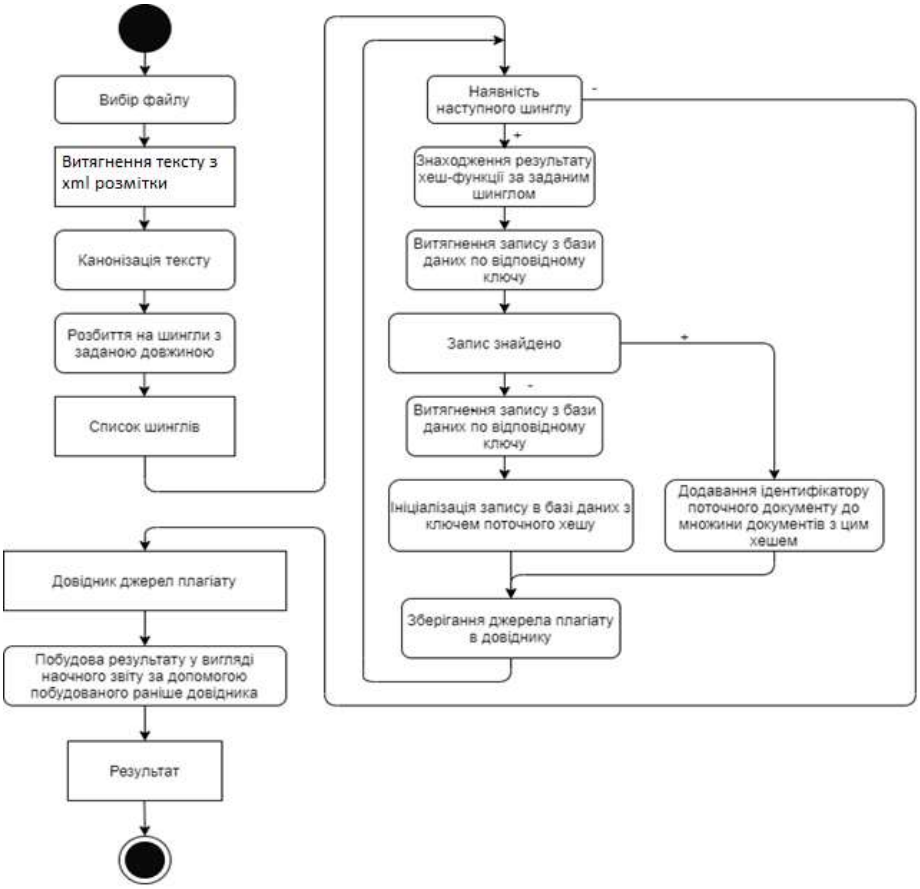

Рисунок 2.3 – Діаграма активності удосконаленого процесу перевірки текстів на унікальність

Таким чином, запропоновано удосконалений алгоритм перевірки doc та docx файлів на унікальність, що базується на використанні попереднього зберігання хешованих частин в нереляційному сховищі типу ключ-значення. Це, в свою чергу, забезпечить чималу швидкість обробки значних обсягів документів. При цьому, обсяг затраченої пам’яті залишатиметься незмінним [1].

2.5 Висновки

У даному розділі розроблену математичну модель процесу визначення унікальності текстового документу. Підвищення точності перевірки було реалізовано за рахунок введення коефіцієнту Жаккарда як фінальний етап обрахунку унікальності.

Розроблено удосконалений алгоритм визначення унікальності для txt та doc/docx документів, що базується на запропонованій математичній моделі. Фундаментальним підходом для удосконалень був метод шинглів. Пораховано складність алгоритму для обсягів пам’яті, що використовуються алгоритмом - $O(m)$. Пораховано складність алгоритму для необхідної процесорної потужності - $O(\frac{n \times m}{k})$. Наведена складність також підтверджує, що швидкодія не погіршиться.

Зазначено, що при обробці файлів з розширеннями doc і docx необхідно ввести додатковий попередній етап отримання текстової інформації. Це зумовлено особливістю використання розмітки XML в doc і docx документах.

3 СТРУКТУРНА ОРГАНІЗАЦІЯ ТА ОСОБЛИВОСТІ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ВИЗНАЧЕННЯ УНІКАЛЬНОСТІ ТЕКСТОВОГО ДОКУМЕНТУ

Для того щоб коректно ідентифікувати модулі, що необхідно розробити, потрібно описати процес визначення унікальності текстового документу у вигляді функціональної моделі. Методологію IDEF0 є доцільним інструментарієм для цього. Функціональні моделі розроблені для опису існуючих бізнес-процесів, які використовують як природні, так і графічні мови. Джерелом графічної мови для надсилання інформації про певну систему є сама техніка IDEF0. Функціональна модель процесу визначення унікальності текстового документу зображена на рисунку 3.1.

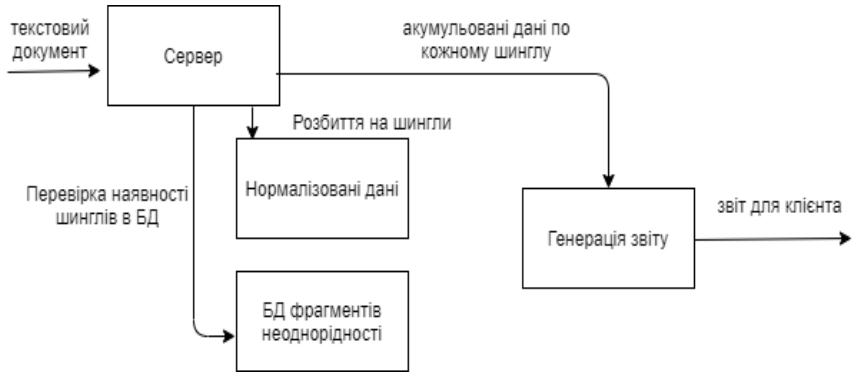

Рисунок 3.1 – IDEF0 діаграма процесу визначення унікальності текстового документу

Задля досягнення поставлених цілей необхідно реалізувати модулі графічного інтерфейсу, серверний модуль, модуль обчислення унікальності, модуль бази даних.

За графічне представлення всіх функцій кінцевого користувача відповідає клієнтський модуль. Він буде представлений у вигляді віконної програми для

операційної системи Windows. Цей модуль обробляє всі дії та запити користувача, наприклад: дозволяє вибрати файл текстового документа для подальшої перевірки на унікальність; Візуальні результати перевірки вибраного документа на унікальність Слід зазначити, що цей модуль не реалізує покращений алгоритм перевірки текстів на унікальність, а лише перенаправляє запити на серверний модуль, надсилаючи веб-запити через комп'ютерну мережу.

Поділ наданого текстового документа на фрагменти неоднорідності та порівняння з фрагментами, що є в основі, відноситься до модуля розрахунку унікальності текстового документа. Цей модуль взаємодіє із серверним модулем, який ініціює запити до модуля обчислення унікальності та з модулем бази даних для отримання фрагментів неоднорідності у потрібному форматі. Також цей модуль формує числовий звіт за наслідками перевірки текстового документа на унікальність.

отримання запитів від клієнта через комп'ютерну мережу, їх перевірка та перевірка, збереження та надання доступу до файлів метаданих, що зберігаються у системі; з'єднання з модулем розрахунку унікальності текстового документа та модулем бази даних відповідає серверному модулю. Крім того, цей модуль генерує відповідь у форматі, зрозумілому клієнту, та надсилає результат.

Модуль бази даних буде відповідати за зберігання даних, який включає дві фізичні бази даних: базу даних для зберігання фрагментів неоднорідності і базу даних для зберігання метаданих про текстові документи в системі. Цей поділ необхідний, оскільки сучасні реляційні бази даних недостатньо оптимізовані для багатьох запитів ключ-значення. Саме ці запити складають ядро алгоритму отримання та зберігання фрагментів неоднорідності та ідентифікаторів завдань, що містять їх. У той же час бази даних типу ключ-значення не дозволяють виконувати запити з різних атрибутів, що є

обов'язковою умовою для отримання та зберігання метаданих текстових документів. Отже, цього потрібно окрема реляційна база даних.

Виходячи з цього структура системи для визначення унікальності текстового документу матиме такий вигляд (рис. 3.1).

Рисунок 3.2 – UML діаграма взаємодії модулів системи визначення унікальності текстового документу

3.1 Розробка модуля клієнтського інтерфейсу

Для реалізації запропонованої програми необхідно вибрати мову програмування, яка підтримує розробку графічних компонентів для операційної системи Windows. Найпоширенішими мовами програмування для таких вимог є C# і Java.

C # є широко прийнятним, багатопарадигмальним, строго типізованим, лексичними обмеженнями, обов'язковим, декларативним, функціональним, з підтримкою об'єктно-орієнтованого та компонентно-орієнтованого підходів програмування. Він був розроблений компанією Microsoft приблизно в 2000 році як частина її ініціативи платформи .NET, а потім затверджений як міжнародний стандарт Ecma (ECMA-334) у 2002 році та ISO (ISO / IEC 23270) у 2003 році. C # — мова програмування, яка була розроблена для загальномовної рамки CLR (CommonLanguageRuntime).

.NET Framework — це програмне забезпечення, розроблене Microsoft і в основному працює на операційній системі Windows. Він включає велику бібліотеку класів під назвою Framework Class Library (FCL) і пропонує взаємодію мов програмування (кожна мова програмування є частиною платформи .NET, може використовувати код, написаний іншими мовами програмування) для багатьох мов програмування, таких як B. F #, VisualBasic, IronPython та інші. Програми, написані для .NET Framework, виконуються в програмному середовищі (на відміну від апаратного середовища), що називається CLR. CLR — це програмна віртуальна машина, яка відповідає за безпеку, керування пам’яттю та обробку винятків для програмних компонентів. Тому код, написаний за допомогою .NET Framework, називається керованим кодом. FCL і CLR разом утворюють .NET Framework.

FCL забезпечує користувальницький інтерфейс, доступ до даних, підключення до зберігання, шифрування даних, розробку веб-додатків, різноманітні алгоритми та комунікації комп’ютерної мережі. Програмісти розробляють програмне забезпечення, поєднуючи вихідний код з .NET Framework та іншими бібліотеками. FCL призначений для використання в більшості нових програм, написаних для платформи Windows.

.NET Framework працює як рідне програмне забезпечення, хоча Microsoft працювала спеціально для стандартизації програмного стеку до його першого випуску. Незважаючи на зусилля стандартизувати стек програмного

забезпечення, розробники, особливо в спільнотах вільного та відкритого програмного забезпечення, висловлювали занепокоєння щодо обраних умов і перспектив вільного та відкритого програмного забезпечення, зокрема, патентів на програмне забезпечення. Відтоді Microsoft змінила дизайн .NET, щоб краще відповідати її поточній моделі розробки програмного забезпечення на основі спільноти, включаючи випуск оновлення свого патенту, яке обіцяє вирішити проблеми.

.NET Framework також створила сімейство платформ .NET, які зосереджені на мобільних комп'ютерах, вбудованих пристроях, альтернативних операційних системах і плагінах для веб-браузерів. Скорочена версія платформи, .NET Compact Framework, доступна на платформі Windows CE, включаючи пристрої Windows Mobile, такі як смартфони. .NET Micro Framework зосереджено на вбудованих пристроях з обмеженими обчислювальними ресурсами. Silverlight був доступний у вигляді програми для веб-браузера. Mono доступний для багатьох операційних систем і може бути легко адаптований до поширених операційних систем Android та iOS для смартфонів та ігрових движків. .NET Core зосереджено на універсальній платформі Windows (UWP), кросплатформних і хмарних обчисленнях.

Екосистема платформи .NET має три інструменти для створення графічних компонентів: UWP, WindowsForms, WPF.

Універсальна платформа Windows (UWP) — це комп'ютерна платформа, розроблена Microsoft і вперше представлена в Windows 10. Метою цієї платформи є створення універсальних програм, які працюють на Windows 10, Windows 10 Mobile, Xbox One і HoloLens без написання будь-якого коду. . Специфічна для системи Підтримує розробку програм для операційної системи Windows з використанням C++, C#, VB.NET і XAML. API реалізовано на C++ і підтримується в C++, VB.NET, C#, F# і JavaScript. Розроблений як розширення платформи Windows Runtime Platform (WindowsRT) і вперше представлений у

Windows Server 2012 і Windows 8, UWP дозволяє розробникам створювати програми, які можна запускати на різних типах пристроїв.

Windows Forms (WinForms) — це безкоштовна графічна бібліотека з відкритим вихідним кодом (GUI), яка є частиною Microsoft .NET Framework або Mono Framework і забезпечує основу для написання потужних графічних клієнтських програм для настільних комп'ютерів, ноутбуків і планшетів. Хоча розглядається як заміна більш ранньої та більш складної бібліотеки класів Microsoft Foundation, розробленої на C ++, вона є платформою Microsoft .NET Framework. На цьому від пакету програм вона витрачає більшу частину свого часу, просто чекаючи, коли користувач щось зробить, наприклад, заповни текстове поле або натисне кнопку. Форми Windows надають доступ до загальних елементів власного користувацького інтерфейсу Windows, загортаючи наявний API Windows у керований код. За допомогою Windows Forms, .NET Framework забезпечує більш повну абстракцію над APIWin32, крім того, що було створено VisualBasic або MFC. WindowsForms подібний до бібліотеки MicrosoftFoundationClass (MFC) при розробці клієнтських застосунків. Він пропонує обгортку, що складається з набору класів C++ для розробки програм; однак він не забезпечує рамки програм для замовчуванням, як MFC. Кожен елемент управління в додатку Windows Forms - це конкретний екземпляр відповідного класу.

Windows Presentation Foundation (WPF) — це вільна графічна підсистема з відкритимвихідним кодом (схожа на WinForms), спочатку створена Microsoft для надання інтерфейсу користувача в графічних додатках на базі Windows. WPF, раніше відомий як Avalon, спочатку був представлений як частина .NET Framework 3.0 у 2006 році. WPF використовує DirectX і намагається надати послідовну модель програмування для побудови додатків. Він відокремлює інтерфейс користувача від логіки додатку та нагадує подібні об'єкти, орієнтовані на XML, такі як реалізовані в XUL та SVG.

WPF використовує XAML, мову розмітки на основі XML, для визначення та зв’язку різних елементів інтерфейсу. Програми WPF можна розглянути як окремі настільні програми або розглянути як вбудований об’єкт на веб-сайті. WPF має об’єднані ряди загальних елементів інтерфейсу користувача, таких як 2D та 3D візуалізація, фіксовані та адаптивні документи, типографія, векторна графіка, анімація виконання та попередні надані носії інформації. Ці елементи потім можуть бути пов’язані та маніпульовані на основі різних підходів, взаємодії користувачів та зв’язків із даними.

Бібліотеки виконання WPF включені у всі версії Microsoft Windows з Windows Vista та Windows Server 2008. Користувачі Windows XP SP2 та SP3 та Windows Server 2003 можуть додатково встановлювати необхідні компоненти. Microsoft Silverlight надає функціональні можливості, які в основному є підмножиною WPF для забезпечення вбудованих веб-елементів управління.

Java – цеоб’єктно-орієнтована мова програмування загального призначення, заснована на класах і покликана мати якомога менше залежностей від реалізації. Наведена мова призначена для того, щоб розробники додатків могли писати один раз і запускати на будь-якій платформі, тобто скомпільований код Java може працювати на всіх платформах, які підтримують Java без необхідності перекомпіляції програм вихідного коду. Програми Java зазвичай компілюються в байт коду, який може працювати на будь-якій віртуальній машині Java (JVM) незалежно від фізичної архітектури комп’ютера. Синтаксис мови Java схожий на C та C ++, але в ньому значно менше об'єктів низького рівня, ніж в будь-якому з них. Станом на 2019 рік Java була виконана з найпопулярніших мовних програм, відповідно до Git, особливо для клієнт-серверних веб-додатків. Java була спочатку розроблена Джеймсом Гослінгом у компанії Sun Microsystems, яка з тих пір була придбана в Oracle, випущена в 1995 році як основний компонент платформи Java Sun Microsystems. Оригінальні та довідкові Java-компілятори, віртуальні машини та бібліотечні класи були спочатку випущеними компанією Sun під власною ліцензією.

Станом на травень 2007 року, відповідно до специфікацій процесу спільної Java, Sun передала найкращі технології Java відповідно до загальної публічної ліцензії GNU.

Віртуальна машина Java (JVM) - це віртуальна машина, яка дозволяє запустити комп'ютер Java, а також програми, написані іншими мовами, які також комп'ютерні в Java байт-код. JVM деталізується специфікацією, яка формально описує, що необхідно для реалізації JVM. Наявність специфікації Java Development Kit (JDK), необхідна для турботи про базову апаратну платформу. JVM розроблений проектом OpenJDK як відкритий код і включає компілятор JIT під назвою HotSpot. Комерційно підтримувані версії Java, доступні від корпорації Oracle, базуються на режимі виконання OpenJDK. Eclipse OpenJ9 - це один відкритий вихідний код для OpenJDK.

Головною графічною бібліотекою для побудови графічних додатків за допомогою Java є Swing. Swing - це інструментарій графічних інструментів GUI для Java. Він є частиною класу Java Foundation Oracle (JFC) - API для надання графічного інтерфейсу користувача для програми Java. Swing розроблено, щоб забезпечити більший складний набір компонентів графічного інтерфейсу, ніж попередній інструментарій абстрактних вікон (AWT). Swing надає зовнішній вигляд, який імітує зовнішній вигляд на платформі Windows, в основному, Linux та MacOS. Він має більш потужні та гнучкі компоненти, ніж AWT. Окрим звичних компонентів, таких як кнопки, мітки та прапорці, Swing пропонує декілька вдосконалених компонентів, таких як панель із вкладками, панелі прокрутки, таблиці, дерева та списки. На вибір від компонентів AWT, компоненти Swing не реалізовані кодом, визначеним конкретною платформою. Натомість вони повністю написані на Java і тому не залежать від платформи. У грудні 2008 року компанія Sun Microsystems випустила бібліотеку JavaFX на основі CSS / FXML, яка мала намір стати наступником Swing.

Розглянувши основні платформи та засоби, які вони надають для створення клієнтських додатків із використанням вбудованих графічних

компонентів, можемо зробити висновок, що для реалізації поставленої задачі доцільно використати для використання мови програмування C# та графічної бібліотеки WPF.

Для гнучкої реалізації архітектури для можливого подальшого розширення додатків та для чіткого розділення відповідальності програмних компонентів використовується патерн проектування MVVM.

Model-View-Viewmodel (MVVM) - це програмний архітектурний шаблон, який забезпечує розмежування графічного інтерфейсу користувача, тобто подання, будь-яке за допомогою мови розмітки чи коду графічного інтерфейсу - від логіки чи додатку (модель для представлення будь-яких даних не залежали від конкретної реалізації моделі. Модель перегляду в MVVM є перетворювачем, тобто модель перегляду відповідає перетворенню об'єктів даних із моделлю таким чином, що об'єктами можна було легко створювати та відображати за допомогою графічного інтерфейсу. У цьому відношенні моделі перегляду є більшою моделлю, виглядом, і обробляє більшу частину, якщо не всю логіку відображення перегляду. Модель перегляду може реалізувати модель посередника, організуючи доступ до зворотної логіки, що підтримує вид.

MVVM також називається сполучною моделлю-перегляд-модель, особливо в реалізації, що не включає платформу .NET. ZK (рам веб-додатків, написані на Java) та KnockoutJS (бібліотека JavaScript) використовують модель-перегляд – палітурку. Модель посилається або на доменну модель, яка представляє реальний стан (об'єктно-орієнтований підхід), або до рівня доступу до даних, який представляє контент орієнтований на. Як і у моделях контролера перегляду моделей (MVC) та моделей перегляду презентацій (MVP), представлення - це структура, компонування та зовнішній вигляд того, що користувач бачить на екрані. Він створює модель представлення та обробляє взаємодію користувача з поданням, і передає обробку їх моделі перегляду через

прив'язку даних (прив'язку), що визначено для зв'язку моделі перегляду та перегляду.

Модель перегляду - це абстракція подання, що відкриває загальнодоступні властивості та команди. Замість контролера шаблону MVC або шаблону презентатора MVP, MVVM має зв’язуючу структуру, яка автоматизує зв’язок між поданням та його пов’язаними властивостями в моделі перегляду. Основна відмінність моделі перегляду від презенту в шаблоні MVP є тому, що презентатор має посилання на вигляд, тоді як модель перегляду не має. Натомість представлена прив'язується до моделі даних, що використовується для відправлення та отримання оновлень. Щоб ефективно функціонувати, для виконання прив’язки потрібна технологія зв’язків або генерування готового коду. Дані та прив'язка команди неявні в шаблоні MVVM. У стеці рішень Microsoft зв’язуючу структуру є мовою розмітки під назвою XAML. Схематичну візуалізацію шаблону проектування MVVM зображено на рисунку 3.3.

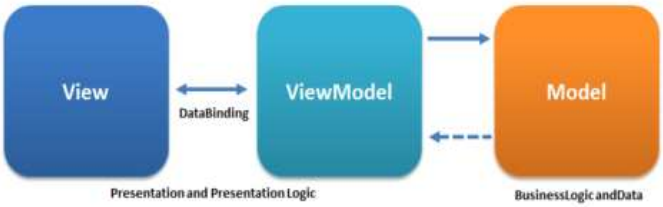

Рисунок 3.3 – Візуалізація шаблону проектування MVVM

Отож, за допомогою використання технології WPF та паттерну проектування MVVM програмна реалізація програмного модуля, відповідального за графічне представлення усього функціоналу кінцевому користувачеві реалізована в наступному вигляді (рис. 3.4).

Рисунок 3.4 – Діаграма активності клієнтського додатку

3.2 Розробка модуля обчислення унікальності документу

Шаблон дизайну ітератор використовується для розробки модуля для обчислення унікальності текстового документа.

В об’єктно-орієнтованому програмуванні шаблон дизайну ітератор — це модель проектування, яка використовує ітератор для переміщення по контейнеру та доступу до елементів контейнера. Ітераційний підхід відокремлює контейнери від алгоритмів обходу; в деяких випадках алгоритми обов'язково залежать від контейнера і тому навряд чи їх можна розділити. Наприклад, гіпотетичний алгоритм пошуку елемента в послідовності може бути реалізований насамперед з використанням ітератора певного типу, а не як специфічний для контейнера алгоритм. Таким чином, ви можете використовувати пошук елементів для будь-якого контейнера, який підтримує потрібний вам тип ітератора.

Шаблон дизайну ітератора є однією з 23 добре відомих моделей проектування ГОФ, що описують, як вирішувати повторювані проблеми проектування для створення гнучких і повторно використовуваних об’єктно-орієнтованих систем, тобто об’єктів, які легше впроваджувати, модифікувати, тестувати та повторно використовувати. Визначення операцій доступу та обходу в агрегованому інтерфейсі є негнучким, оскільки він стає вразливим до конкретних операцій доступу та обходу та робить неможливим додавання нових операцій без зміни інтерфейсу.

Шаблон дизайну Ітератор надає наступний підхід. Визначте один об’єкт, який інкапсулює доступ до контейнера та обхід контейнера. Клієнти використовують ітератор для доступу та навігації по блоку, не знаючи його представлення (структури даних). Ви можете використовувати різні ітератори для доступу та переміщення блоку різними способами. Нові операції доступу та обходу можуть бути визначені незалежно шляхом створення нових ітераторів.

Шаблон ітератора архітектури є невід'ємною частиною модуля для розрахунку унікальності текстового документа і забезпечує механізм обходу фрагментів неоднорідності з подальшим порівнянням із записами даних, наявними в базі даних. Принцип роботи модуля обчислення унікальності текстового документу зображено UML діаграмі послідовності (рис. 3.5).

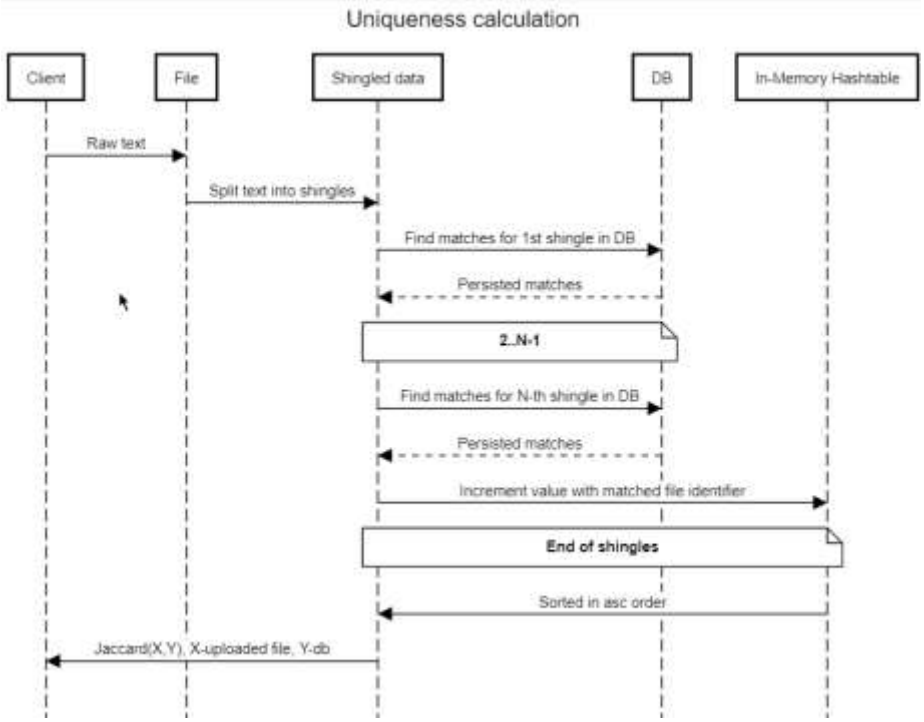

Рисунок 3.5 – Діаграма послідовності модуля обчислення унікальності текстового документу

Отже, було реалізовано модуль обчислення унікальності текстового документу, що відповідає за розбиття наданого текстового документу на

хешовані фрагменти і порівняння з чек-сумами, що наявні в нереляційному сховищі.

3.3 Розробка серверного модуля

Для розробки серверного модуля використовуються фреймворк ASP.NET і команда шаблону архітектури.

ASP.NET — це система сервера веб-додатків з відкритим вихідним кодом для розробки статичних і динамічних веб-додатків. Він був розроблений компанією Microsoft, щоб дати можливість розробникам створювати динамічні веб-сайти, програми та служби. ASP.NET вперше був опублікований у січні 2002 року з версією 1.0 .NET Framework і є наступником технології Microsoft Active Server Pages. ASP.NET заснований на загальній мові виконання (CLR), яка дозволяє програмістам писати код ASP.NET будь-якою мовою, що підтримується платформою .NET. Структура розширення ASP.NET SOAP дозволяє компонентам ASP.NET обробляти повідомлення у форматі SOAP. Наступником ASP.NET є ASP.NET Core. Це нова реалізація платформи ASP.NET як модульної веб-платформи разом з іншими структурами, такими як Entity Framework. Нова платформа використовує нову платформу компілятора з відкритим вихідним кодом .NET (кодова назва Roslyn) і є кросплатформною.

У ASP.NET управління автентифікацією користувачів сайту було реалізовано через API Membership API, який є засобом входу, зберігання та керування обліковими записами користувачів системи. Архітектура API членства розроблена для керування користувачами, що зберігаються в різних джерелах: Microsoft SQL Server, Microsoft Active Directory або сховище користувачів. У ASP.NET автентифікацію можна виконати за допомогою форм або за допомогою Windows через IIS.

Internet Information Services (IIS) — це власний набір серверів для Інтернет-сервісів від Microsoft. IIS поширюється разом із WindowsNT. Основним компонентом IIS є веб-сервер, який можна використовувати для розміщення веб-сайтів в Інтернеті. IIS має вбудовану підтримку таких протоколів: HTTP, HTTPS, FTP, POP3, SMTP, NNTP.

Веб-сервер IIS пропонує кілька способів розрізняти доступ до сайтів і веб-програм. Щоб отримати доступ до захищеного ресурсу, користувач повинен ввести логін та пароль користувача під Windows, на якій встановлено IIS (або в домені ActiveDirectory, якщо сервер є частиною домену). Після цього користувач працює з сайтом так, ніби здійснив інтерактивний вхід на сервер.

Команда — це шаблон дизайну поведінки, який використовується в об’єктно-орієнтованому підході, який представляє дію. Об’єкт команди містить саму дію та її параметри.

В об’єктно-орієнтованому програмуванні шаблон команди — це шаблон поведінки, в якому об’єкт використовується для інкапсуляції всієї інформації, необхідної для виконання дії або для пізнішого виклику події. Ця інформація включає назву методу, об’єкт, який володіє методом, і значення параметрів методу. Шаблон команди завжди має чотири терміни: команда, одержувач, абонент і клієнт. Об’єкт команди містить інформацію про одержувачів і викликає метод одержувача. Значення параметрів приймача зберігаються в команді. Абонент знає, як виконується команда, записує та записує виконані команди. Художник нічого не знає про конкретну команду, він знає лише її інтерфейс. Обидва об'єкти належать об'єкту клієнта. Клієнт вирішує, які команди виконувати і коли. Щоб виконати команду, він передає об’єкт команди виконавцю. Використання командних об’єктів спрощує побудову загальних компонентів, які мають бути делеговані або виклики методів виконуватися в будь-який час без необхідності знати методи класу або параметри методу. За допомогою виконавця ви можете ввести рахунки за виконані команди без того,

щоб замовник знати цю модель розрахунку. Принцип роботи серверного модулю зображено на UML діаграмі активності (рис 3.6).

Рисунок 3.6 – Діаграма активності серверної частини додатку

Отож, було реалізовано модуль серверної частини, який відповідальний за обробку клієнтських запитів, а також їх валідацію, верифікацію і зберігання, надання доступу до метаданих файлів, що зберігаються в системі; комунікацю між модулем обчислення унікальності текстового файлу та модулем бази даних. Також відповідальний за формування відповіді для клієнта та відправлення результату.

3.4 Розробка модуля бази даних

Для розробки модуля бази даних використовуються дві бази даних: MSSQL для зберігання метаданих в текстових документах і Redis для ефективного зберігання фрагментів неоднорідності.

Microsoft SQL Server — це система керування базами даних, розроблена компанією Microsoft. Основною мовою запитів є T-SQL, яка була розроблена спільно Microsoft і Sybase. T-SQL — це реалізація стандарту для мови структурованих запитів SQL з певними розширеннями. Використовується для роботи з базами даних різного розміру від персональних до великих та корпоративних.

Redis (віддалений сервер словників) — це система керування базами даних NoSQL з відкритим вихідним кодом, яка працює зі структурами даних ключ-значення. Він використовується як для баз даних, так і для реалізації механізму кешування, а іноді і для брокерів повідомлень. Орієнтований на досягнення максимальної продуктивності в ядерних операціях. Інтерфейси доступу, написані на C, розроблені для більшості поширених мов програмування, включаючи C#. Вся база даних зберігається в RAM. Тому існують механізми для реєстрації дій і для збереження перетворення бази даних на жорсткому диску. Redis також забезпечує операції з реалізації механізму обміну повідомленнями в шаблоні передплатника публікації: він дає можливість програмам створювати канали, підписуватися на них і розміщувати повідомлення на каналах, які отримають усі передплатники каналу. Підтримує реплікацію даних з головного вузла на кілька підлеглих вузлів, тобто реплікацію провідний-підпорядкований. Також підтримує транзакцію та пакетну обробку команд, виконання командного пакету, отримання підсумкового пакету. Redis зберігає всі дані у вигляді словника, в якому ключі пов’язані зі своїми значеннями. Однією з основних відмінностей між Redis та іншими сховищами даних є те, що значення цих ключів не обмежуються

рядками. Підтримуються такі абстрактні типи даних: рядки, списки, набори, хеш-таблиці, упорядковані набори. Тип даних значення визначає, які операції (команди) йому доступні; Підтримуються високорівневі операції, такі як агрегація та розрізнення наборів, а також сортування наборів. Модель реляційного зберігання даних також дозволяє легко логічно масштабувати структуру бази даних у майбутньому, якщо функціональні вимоги системи зміняться. Принцип роботи модуля бази даних зображено на UML діаграмі активності (рис. 3.7).

Рисунок 3.7 – Діаграма активності модуля бази даних

Отже, реалізовано модуль бази даних, що є відповідальним за зберігання хешованих фрагментів та метаданих про файли в системі, також відповідальний за формування та виконання запитів.

3.5 Аналіз результатів функціонування інформаційної технології визначення унікальності текстового документу

Розроблена інформаційна технологія перевірятиметься на відповідність поставленим задачам, безпечність та коректність роботи у середовищі віндовс шляхом впровадження технології тестування чорна скринька, тобто тестувальник тестує систему нічого не знаючи про деталі її реалізації.

При завантаженні документу в якому дуже низький рівень плагіату запропоноване програмне забезпечення вказує, що файл успішно пройшов верифікацію.

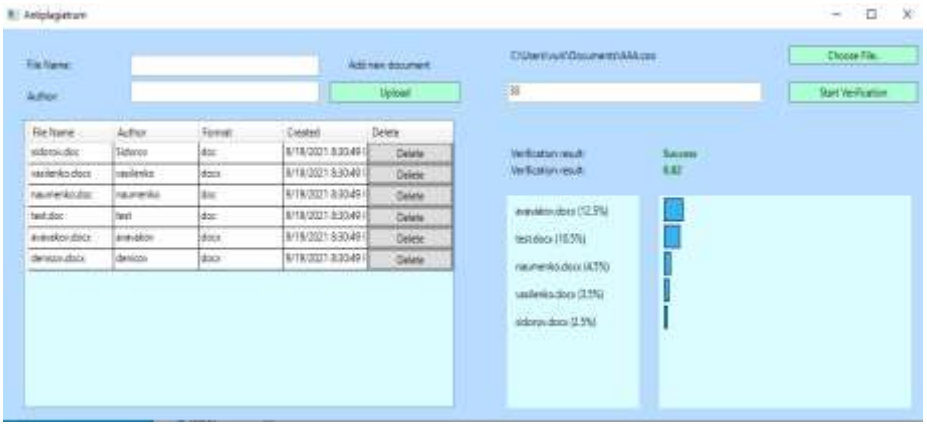

Рисунок 3.8 – Успішна верифікація документу

При завантажені документу, майже ідентичного до такого що існує в запропонованому ПЗ, додаток вказує, що верифікація пройшла невдало.

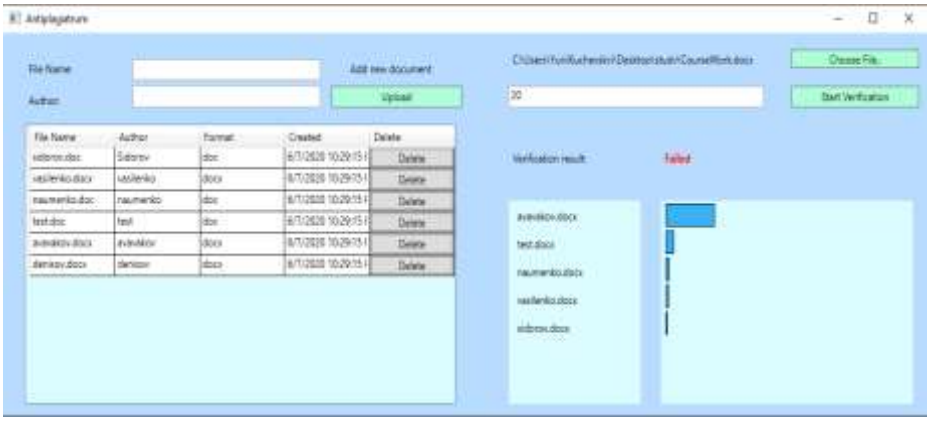

Рисунок 3.9 – Невдала верифікація документу

Також маємо можливість самостійно додати файл до бази документів. Для цього заповнюємо обов’язкові поля File Name та Author.

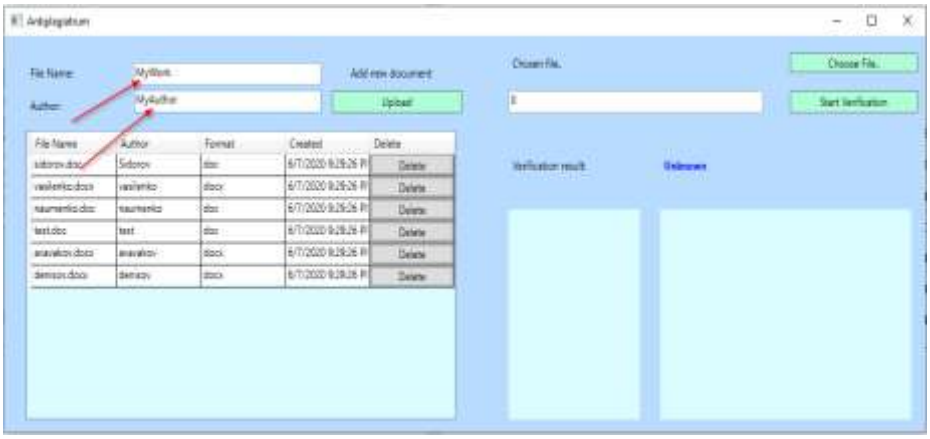

Рисунок 3.10 – Заповнення полів File Name та Author

Після натиснення кнопки Upload необхідно обрати бажаний файл в провіднику.

Рисунок 3.11 – Вибір бажаного файлу в провіднику

Після вибору файл завантажується на сервер та з’являється в списку що відображає базу документів.

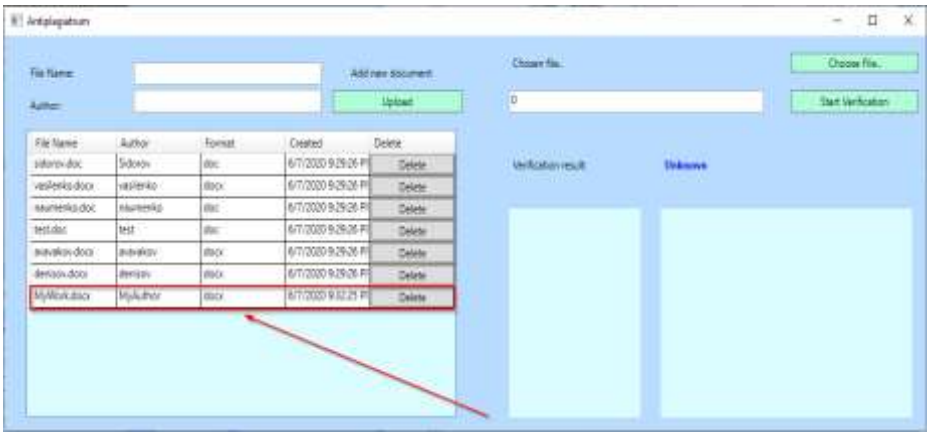

Рисунок 3.12 – Завантажений документ

Після натиснення клавіші Delete відповідний документ повинен бути видалений, що і спостерігається під час тестування.

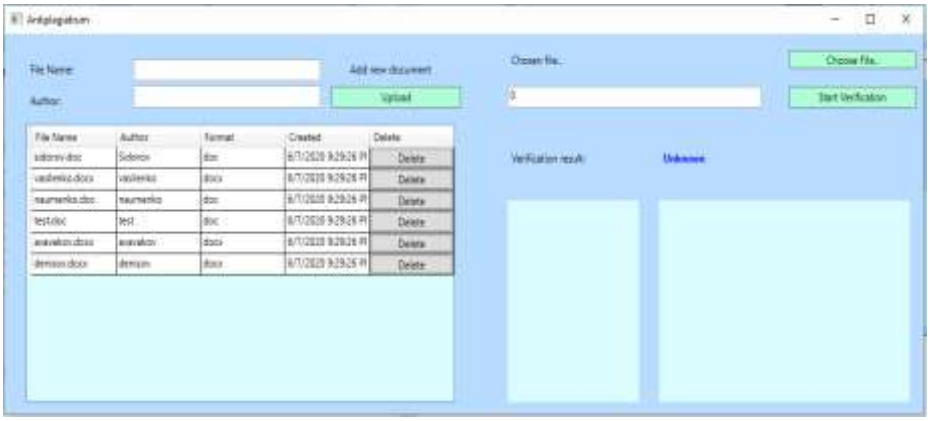

Рисунок 3.13 – Видалення документа з бази

Доречність використання покращеного методу перевірки текстової інформації на унікальність було доведено детальним обрахунком складності алгоритму та його подальшим порівнянням зі складністю аналогу. Результати пророблених досліджень щодо залежності необхідних ресурсів по перевірці текстової інформації на унікальність від обсягу пам’яті необхідної для зберігання чек-сум наведені у таблиці 1.

Таблиця 3.1 – необхідних ресурсів по перевірці текстової інформації на унікальність від обсягу пам’яті необхідної для зберігання чек-сум

К-сть шинглів	К-сть текстів	К-сть операцій (стандарт)	К-сть операцій (удосконалений)	Затрачена пам’ять (стандарт)	Затрачена пам’ять (удосконалений)
10	10	1000	10	100	100
10	50	25000	10	500	500
50	10	5000	50	500	500

50	50	125000	50	2500	2500
100	100	1000000	100	10000	10000
100	200	4000000	100	20000	20000
200	200	8000000	200	40000	40000

Таким чином, наведений алгоритм удосконалює точність аналізу текстових робіт на унікальність при збережені швидкодії обробки документу, тому є частим вибором при розробці сучасних систем перевірки на унікальність. Використання коефіцієнту Жаккарда дає більш точну оцінку при обчислені унікальності текстового документу. При цьому, потреби в обсягах пам'яті та вимоги до процесорної потужності залишились незмінними.

Розроблена інформаційна технологія була перевірена на відповідність поставленим вимогам та безпечність та коректність роботи у ОС віндовс. Аналіз результатів функціонування було виконано шляхом використання технології тестування чорна скринька, тобто тестувальник перевіряє систему нічого не знаючи про деталі її внутрішньої реалізації. В ході аналізу технологія показала себе, як стабільний програмний комплекс, що відповідає усім вимогам, поставлених на початку дослідження. Додаток було проаналізовано при допустимому відсотку повторень 20%, використовуючи документи розміром від 5 до 40 сторінок формату doc, docx, txt. Результати аналізу функціонування технології наведені в таблиці 3.2.

Таблиця 3.2 – Результати тестування додатку

Кількість сторінок	Час обробки docx документу запропонованим алгоритмом (с)	Коефіцієнт унікальності стандартного алгоритму	Коефіцієнт унікальності удосконаленого алгоритму
5	2.34	0.13	0.14
10	4.58	0.15	0.16
15	6.42	0.14	0.15
20	8.55	0.135	0.144
25	10.9	0.137	0.139
30	12.83	0.141	0.15
35	15.1	0.12	0.13
40	17.12	0.15	0.15

З результатів тестування додатку бачимо, що запропонований алгоритм успішно обчислює коефіцієнт унікальності за менш ніж 20 секунд для одного документу і в середньому на 8% точніший ніж класичний аналог.

3.6 Висновок

В даному розділі було програмно реалізовано інформаційну технологію, що несе собою сукупність складових інтелектуального модуля перевірки текстових документів на унікальність, що комунікують відповідно до спроектованої структури модулів; обгрунтовано вибір клієнтської та серверної мови програмування C# та інтегрованого середовища розробки Visual Studio задля вирішення поставленої задачі, обгрунтовано використання клієнтського фреймворку WPF та архітектури проектування MVVM для розробки клієнтського інтерфейсу, використання серверного фреймворку ASP.NET та архітектурного шаблону проектування command для серверного модуля, використання шаблону проектування iterator для модуля обчислення унікальності текстового документу, використання реляційної бази даних MS SQL для зберігання метаданих про текстові документи та використання NoSQL бази даних типу ключ-значення Redis для зберігання хешованих фрагментів неоднорідності. Тестування системи довело відповідність розробленого програмного продукту до поставлених вимог. Розроблений аналог показав на 8% точніші результати ніж стандартний алгоритм.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Оцінювання комерційного потенціалу розробки

Метою даного розділу є проведення технологічного аудиту, в даному випадку нового програмного продукту інформаційної технології Інформаційна технологія визначення унікальності текстового документу. Продукт надає змогу розпізнавати виявлення плагіату в текстових документах. Особливістю програми є те, що данна технологія надає змогу підвищити точність програмного продукту.

Аналогом може бути сервіс «Антиплагіат.ру», ціна 270 грн. на місяць.

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, у відповідності із табл. 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздат-ність продукту в реальних умовах
2	Багато аналогів на малому ринку	Ринкові п Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку

Продовження табл. 4.1

Ринкові переваги					
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практик на здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так із комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження табл. 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Усі дані по кожному параметру занесено в таблиці 4.2

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії оцінювання	ПІБ експертів		
	Експерт 1	Експерт 2	Експерт 3
	Бали		
Технічна здійсненність концепції	2	3	2
Наявність аналогів на ринку	3	4	4
Цінова політика	3	3	4
Технічні та споживчі властивості виробу	4	3	4
Експлуатаційні витрати	2	3	2
Ринок збуту	3	3	2
Конкурентоспроможність	3	4	4
Фахівці з технічної і комерційної реалізації	4	3	4
Фінансування	4	4	3
Матеріально-технічна база	2	2	3
Термін реалізації ідеї	4	3	3
Супровідна документація	3	3	4
Сума	37	38	39
Середньоарифметична сума балів	(37+38+39) / 3 = 38		

За даними таблиці 4.2 можна зробити висновок щодо рівня комерційного потенціалу даної розробки. Для цього доцільно скористатись рекомендаціями, наведеними в таблиці 4.3.

Таблиця 4.3 - Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

Як видно з таблиці, рівень комерційного потенціалу розроблюваного нового програмного продукту є вище середнього, що досягається за рахунок того, що інформаційна модель відрізняється від існуючих покращеним алгоритм перевірки на унікальність, який пришвидшує точність програмного продукту.

4.2 Прогнозування витрат на виконання науково-дослідної (дослідно-конструкторської) роботи

4.2.1 Основна заробітна плата розробників, яка розраховується за формулою:

$$З_о = \frac{M}{T_р} \cdot t, \quad (4.1)$$

де М – місячний посадовий оклад конкретного розробника (дослідника), грн.;

$T_р$ – число робочих днів в місяці, 21 днів;

t – число днів роботи розробника (дослідника).

Результати розрахунків зведемо до таблиці 4.1.

Таблиця 4.1 – Основна заробітна плата розробників

Найменування посади	Місячний посадовий	Оплата за робочий день,	Число днів роботи	Витрати на заробітну плату,

	оклад, грн.	грн.		грн.
Керівник проекту	25000	1190,48	35	41666,667
Інженер	22000	1047,62	35	36666,667
Всього				78333,33

Так як в даному випадку розробляється програмний продукт, то розробник виступає одночасно і основним робітником, і тестувальником розроблюваного програмного продукту.

4.2.2 Додаткова заробітна плата розробників, які приймали участь в розробці обладнання.

Додаткова заробітна плата прийнято розраховувати як 12 % від основної заробітної плати розробників та робітників:

$$З_д = З_о \cdot 12\ \% / 100\ \% \qquad (4.2)$$

$З_д = (78333,33 \cdot 12\ \% / 100\ \%) = 9400,00$ (грн.)

4.2.3 Нарахування на заробітну плату розробників.

Згідно діючого законодавства нарахування на заробітну плату складають 22 % від суми основної та додаткової заробітної плати.

$$Н_з = (З_о + З_д) \cdot 22\ \% / 100\% \qquad (4.3)$$

$Н_з = (78333,33 + 9400,00) \cdot 22\ \% / 100\ \% = 19301,33$ (грн.)

4.2.4. Оскільки для розроблювального пристрою не потрібно витрачати матеріали та комплектуючи, то витрати на матеріали і комплектуючі дорівнюють нулю.

4.2.5 Амортизація обладнання, яке використовувалось для проведення розробки.

Амортизація обладнання, що використовувалось для розробки в спрощеному вигляді амортизація обладнання, що використовувалась для розробки розраховується за формулою:

$$A = \frac{Ц}{Тв} \cdot \frac{t_{вик}}{12} \text{ [грн.]}. \quad (4.4)$$

де Ц – балансова вартість обладнання, грн.;

Т – термін корисного використання обладнання згідно податкового законодавства, років

$t_{вик}$ – термін використання під час розробки, місяців

Розрахуємо, для прикладу, амортизаційні витрати на комп'ютер балансова вартість якого становить 20000 грн., термін його корисного використання згідно податкового законодавства – 2 роки, а термін його фактичного використання – 1,67 міс.

$$A_{обл} = \frac{20000}{2} \times \frac{1{,}67}{12} = 1388{,}889 \ грн.$$

Аналогічно визначаємо амортизаційні витрати на інше обладнання та приміщення. Розрахунки заносимо до таблиці 4.2. Для розрахунку амортизації нематеріальних ресурсів використовується формула:

$$A_{н.р.} = Ц_{н.р.} * Н_а * \frac{t_{вик.}}{12} \quad (4.5)$$

Норму амортизації $Н_а$ приймемо за 12 %.

Таблиця 4.2 – Амортизаційні відрахування матеріальних і нематеріальних ресурсів для розробників

Найменування обладнання	Балансова вартість, грн.	Строк корисного використання, років	Термін використання обладнання, місяців	Амортиза-ційні відрахуван-ня, грн.
Комп'ютер (Laptop HP Probook)	20000	2	1,67	1388,889
Приміщення	550000	20	1,67	3819,444
Ліцензійна ОС, та спеціалізовані ліцензійні нематеріальні ресурси (Microsoft Visual Studio, Resharper)	6200	-	1,67	103,333
Всього				5311,67

5.2.6 Тарифи на електроенергію для непобутових споживачів (промислових підприємств) відрізняються від тарифів на електроенергію для населення. При цьому тарифи на розподіл електроенергії у різних постачальників (енергорозподільних компаній), будуть різними. Крім того, розмір тарифу залежить від класу напруги (1-й або 2-й клас). Тарифи на розподіл електроенергії для всіх енергорозподільних компаній встановлює Національна комісія з регулювання енергетики і комунальних послуг (НКРЕКП). Витрати на силову електроенергію розраховуються за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_п, \quad (4.6)$$

де В – вартість 1 кВт-години електроенергії з ПДВ для 1 класу підприємства, В = 2,01 грн./кВт;

П – встановлена потужність обладнання, кВт. П = 0,5 кВт;

Ф – фактична кількість годин роботи обладнання, годин.

$К_п$ – коефіцієнт використання потужності, $К_п$ = 0,9.

$В_е = 0,9 \cdot 0,5 \cdot 8 \cdot 35 \cdot 2,01 = 253,26$ (грн.)

5.2.7 Інші витрати та загальновиробничі витрати.

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками. Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.7)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

$I_в$ = 78333,33 * 112% / 100% = 87733,33 (грн.)

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін. Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників:

$$Н_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.8)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

$Н_{нзв}$ = 78333,33 * 120 % / 100 % = 94000 (грн.)

5.2.9 Витрати на проведення науково-дослідної роботи.

Сума всіх попередніх статей витрат дає загальні витрати на проведення науково-дослідної роботи:

$B_{заг}$ = 78333,33+9400,00+19301,33+5311,67+253,26+87733,33+94000 =
= 294332,93 грн.

5.2.11 Розрахунок загальних витрат на науково-дослідну (науково-технічну) роботу та оформлення її результатів.

Загальні витрати на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються *ЗВ*, визначається за формулою:

$$ЗВ = \frac{B_{заг}}{\eta} \text{ (грн)}, \qquad (5.9)$$

де η – коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи.

Так, якщо науково-технічна розробка знаходиться на стадії: науково-дослідних робіт, то η=0,1; технічного проектування, то η=0,2; розробки конструкторської документації, то η=0,3; розробки технологій, то η=0,4; розробки дослідного зразка, то η=0,5; розробки промислового зразка, то η=0,7; впровадження, то η=0,9. Оберемо η = 0,7, так як розробка, на даний момент, знаходиться на стадії промислового зразка:

ЗВ = 294332,93 / 0,7 = 420476 грн.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнювальним позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку. Саме зростання чистого прибутку забезпечить потенційному інвестору надходження додаткових коштів, дозволить покращити фінансові результати його діяльності, підвищить конкурентоспроможність та може позитивно вплинути на ухвалення рішення щодо комерціалізації цієї розробки.

Для того, щоб розрахувати можливе зростання чистого прибутку у потенційного інвестора від можливого впровадження науково-технічної розробки необхідно:

а) вказати, з якого часу можуть бути впроваджені результати науково-технічної розробки;

б) зазначити, протягом скількох років після впровадження цієї науково-технічної розробки очікуються основні позитивні результати для потенційного інвестора (наприклад, протягом 3-х років після її впровадження);

в) кількісно оцінити величину існуючого та майбутнього попиту на цю або аналогічні чи подібні науково-технічні розробки та назвати основних суб'єктів (зацікавлених осіб) цього попиту;

г) визначити ціну реалізації на ринку науково-технічних розробок з аналогічними чи подібними функціями.

При розрахунку економічної ефективності потрібно обов'язково враховувати зміну вартості грошей у часі, оскільки від вкладення інвестицій до отримання прибутку минає чимало часу. При оцінюванні ефективності інноваційних проектів передбачається розрахунок таких важливих показників:

- абсолютного економічного ефекту (чистого дисконтованого доходу);
- внутрішньої економічної дохідності (внутрішньої норми дохідності);

• терміну окупності (дисконтованого терміну окупності).

Аналізуючи напрямки проведення науково-технічних розробок, розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором можна об'єднати, враховуючи визначені ситуації з відповідними умовами.

4.3.1 Розробка чи суттєве вдосконалення програмного засобу (програмного забезпечення, програмного продукту) для використання масовим споживачем.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

$$\Delta П_i = (\pm \Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (4.10)$$

де $\pm\Delta Ц_o$ – зміна вартості програмного продукту (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки;

$Ц_o$ – основний оціночний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки, $Ц_o = Ц_б \pm \Delta Ц_o$;

$Цб$ – вартість програмного продукту у році до впровадження результатів розробки;

ΔN – збільшення кількості споживачів продукту, в аналізовані періоди часу, від покращення його певних характеристик;

λ – коефіцієнт, який враховує сплату податку на додану вартість. Ставка податку на додану вартість дорівнює 20%, а коефіцієнт $\lambda = 0{,}8333$.

ρ – коефіцієнт, який враховує рентабельність продукту;

ϑ – ставка податку на прибуток, у 2021 році ϑ =18%.

Припустимо, що при прогнозованій ціні 130 грн. за одиницю виробу, термін збільшення прибутку складе 3 роки. Після завершення розробки і її вдосконалення, можна буде підняти її ціну на *25* грн. Кількість одиниць реалізованої продукції також збільшиться: протягом першого року – на 30000 шт., протягом другого року – на 50000 шт., протягом третього року на 75000 шт. До моменту впровадження результатів наукової розробки реалізації продукту не було:

*ΔП₁ = (0*25 + (*130 + 25 *)*30000)* 0,8333* 0,32) * (1 - 0,18) = 852799,966 грн.*
*ΔП₂ = (0*25 + (*130 + 25 *)*(30000+50000)* 0,8333* 0,32) * (1 - 0,18) = 2711466,558 грн.*
*ΔП₃ = (0*25 + (*130 + 25 *)*(30000+50000+75000)* 0,8333* 0,32) * (1 - 0,18) = 5253466,457 грн.*

Отже, комерційний ефект від реалізації результатів розробки за три роки складе 8817732,98 грн.

4.3.2 Розрахунок ефективності вкладених інвестицій та періоду їх окупності.

Розраховуємо приведену вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.11)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої науково-дослідної (науково-технічної) роботи, грн;

Т – період часу, протягом якою виявляються результати впровадженої науково-дослідної (науково-технічної) роботи, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, $\tau = 0,05...0,15$;

t – період часу (в роках).

Збільшення прибутку ми отримаємо починаючи з першого року:

ПП= $(852799,966/(1+0,1)^1)+(2711466,558/(1+0,1)^2)+(5253466,457/(1+0,1)^3)$ = 775272,70 + 2240881,45 + 3947007,1 = 6963161,254 грн.

Далі розраховують величину початкових інвестицій *PV*, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} * ЗВ, \quad (4.12)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай $k_{інв}=2...5$, але може бути і більшим;

ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

PV = 2 * 420476 = 840951,22 *грн.*

Тоді абсолютний економічний ефект $E_{абс}$ або чистий приведений дохід *(NPV, Net Present Value)* для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП – PV, \quad (4.13)$$

$E_{абс}$ = 6963161,254 – 840951,22 = 6122210,04 грн.

Оскільки $E_{абс} > 0$ то вкладання коштів на виконання та впровадження результатів даної науково-дослідної (науково-технічної) роботи може бути доцільним.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність або показник внутрішньої но-рми дохідності *(IRR, Internal Rate of Return)* вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.14}$$

$T_ж$ – життєвий цикл наукової розробки, роки.

$E_в = \sqrt[3]{(1 + 6122210{,}04/840951{,}22} - 1 = 1{,}023$

Визначимо мінімальну ставку дисконтування, яка у загальному вигляді визначається за формулою:

$$\tau = d + f, \tag{4.15}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2021 році в Україні d = (0,09…0,14);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,5).

$$\tau_{\min} = 0{,}14 + 0{,}05 = 0{,}19.$$

Так як $E_в > \tau_{min}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Розрахуємо термін окупності вкладених у реалізацію наукового проекту інвестицій за формулою:

$$T_{ок} = \frac{1}{E_в}, \quad (4.16)$$

$T_{ок}$ = 1 / 1,023 = 0,98 р.

Оскільки $T_{ок}$ < 3-х років, а саме термін окупності рівний 0,98 роки, то фінансування даної наукової розробки є доцільним.

Висновки до розділу: економічна частина даної роботи містить розрахунок витрат на розробку нового програмного продукту, сума яких складає 420476 гривень. Було спрогнозовано орієнтовану величину витрат по кожній з статей витрат. Також розраховано чистий прибуток, який може отримати виробник від реалізації нового технічного рішення, розраховано період окупності витрат для інвестора та економічний ефект при використанні даної розробки. В результаті аналізу розрахунків можна зробити висновок, що розроблений програмний продукт за ціною дешевший за аналог і є висококонкурентоспроможним. Період окупності складе близько 0,98 роки.

ВИСНОВКИ

В магістерській кваліфікаційній роботі проведено розробку програмного модуля, інформаційної системи перевірки на унікальність текстового документу.

В першому розділі проведено опис предметної області та сфери застосування програмного продукту, що розробляється, доведено його технічну доцільність шляхом співставлення та порівняння його з аналогами. Для створення та впровадження інформаційної технології програмного продукту розроблено технічні вимоги до об'єкта проектування та визначено шляхи підвищення технічного рівня розв’язку поставленої задачі, а саме, підвищення точності процесу визначення унікальності текстового документу. Проаналізовано способи вирішення технічної проблеми а також їх недоліки. Обрано метод шинглів є фундаментальним алгоритмом для покращення точності перевірки на унікальність.

Розроблено математичну модель процесу визначення унікальності текстового документу.

В роботі розроблено алгоритм функціонування системи. На основі даного алгоритму розроблено додаток що визначає унікальність документу. З допомогою даного програмного засобу можна оперативно перевірити що документ не є плагіатом.

На базі функціональної моделі процесу визначення унікальності текстового документу запропоновано структуру системи та реалізовано її використовуючи мову програмування C#.

Проведене тестування системи і показано її функціональність. Розроблений аналог має на 8% більшу точність перевірки документу. Під час тестування обсяг документу, що аналізується на унікальність не перевищував 40 сторінок; типи форматів документів, що тестувались: doc, docx, txt, потужність множини використовуваних правил перевірки на унікальність не - 15 правил, що відповідає завданню на МКР.

Проведено технологічний аудит з залученням двох незалежних експертів. Визначено, що рівень комерційного потенціалу розробки вище середнього. Згідно з проведеним оцінюванням запропонована розробка більш якісна і конкурентоспроможна.

Чистий прибуток підприємства від впровадження розробки за три роки становить 504532,11 грн. Також здійснено розрахунок ефективності вкладених інвестицій та періоду їх окупності. Життєвий цикл наукової розробки становить 3 роки. Абсолютна ефективність вкладених інвестицій дорівнює 451938 грн.

Результати досліджень було апробовано на міжнародній науково-практичній конференції ІОН (Інтернет Освіта Наука) 2020 [1]; та на науково-технічній конференції факультету інформаційних технологій та комп'ютерної інженерії (2020) [2], науково-технічній конференції Вінницького національного технічного університету факультету інформаційних технологій та комп'ютерної інженерії м. Вінниці у 2021 р [3], міжнародній науково-практичній конференції RECENT SCIENTIFIC INVESTIGATION в Осло [4].

За результатами магістерської кваліфікаційної роботи опубліковано 4 тез доповідей на міжнародних конференціях. Також на розробку зареєстровано 2 авторських права [5, 6] та опубліковано статтю в журналі «Таврійський науковий вісник» [7].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Савчук Т.О., Кучевський Ю.А. Удосконалений алгоритм перевірки текстів на унікальність [Електронний ресурс] – Режим доступу до ресурсу: http://ies.vntu.edu.ua/reports/program/WORK-IES-2020.pdf

2. Савчук Т.О., Кучевський Ю.А. Підхід до аналізу на унікальність курсових розробок [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2020/paper/view/8929/7739

3. Савчук Т.О., Кучевський Ю.А. Удосконалений алгоритм перевірки текстів на подібність [Електронний ресурс] – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2021/paper/view/12511/10469

4. Савчук Т.О., Кучевський Ю.А. Реалізація алгоритму перевірки текстів на унікальність [Електронний ресурс] – Режим доступу до ресурсу: https://www.interconf.top/documents/2021.06.11-12.pdf

5. Свідоцтво про реєстрацію авторського права на твір № 109189 UA. Комп`ютерна програма «Інформаційна система визначення коефіцієнту унікальності текстового документу»] / Т. О. Савчук, Ю. А. Кучевський (Україна), Міністерство економічного розвитку і торгівлі України. – Дата реєстрації 08-11-2021.

6. Свідоцтво про реєстрацію авторського права на твір № 109189 UA. Комп`ютерна програма «Інформаційна система визначення коефіцієнту унікальності текстового документу»] / Т. О. Савчук, Ю. А. Кучевський (Україна), Міністерство економічного розвитку і торгівлі України. – Дата реєстрації 08-11-2021.

7. Савчук Т.О., Кучевський Ю.А. Визначення коефіцієнту унікальності текстового документу з використанням коефіцієнту Жаккарда. Таврійський науковий вісник №5 - 2021.

8. Monostori K., Zaslavsky A., Schmidt H. Document Overlap Detection System for Distributed Digital Libraries // ACM. — 2000.

9. Leong A., Lau H., Rynson W. H. Check: A Document Plagiarisment Detection System // ACM. — 1997.

10. Dreher H. Automatic Conceptual Analysis for Plagiarism Detection // Information and Beyond: The Journal of Issues in Informing Science and Information Technology. — 2007.

11. Meyer zu Eissen S., Stein B. Intrinsic Plagiarism Detection. // Springer. — 2006.

12. Седов А. В., Рогов А. А. Анализ неоднородностей в тексте на основе последовательностей частей речи. // Современные проблемы науки и образования.. — 2013. — Вып. 1.

13. Susan D. Blum (2010). My Word!: Plagiarism and College Culture. Cornell University Press. ISBN 9780801447631.

14. Cully, P. "Plagiarism Avoidance in Academic Submissions". ARROW@TU Dublib. Dublin Institute of Technology - 2013.

15. Kock, N. "A case of academic plagiarism". Communications of the ACM - 1999.

16. BrinS., DavisJ., Garcia-MolinaH. CopyDetectionMechanismsforDigitalDocuments — 2001.

17. O'Connor, Z (2015) Extreme plagiarism: The rise of the e-Idiot?, International Journal of Learning in Higher Education, ISSN 2327-7955.

18. Gabriel, Trip. "Plagiarism Lines Blur for Students in Digital Age". The New York Times - 2010.

19. Hutcheon, Linda (1985). "3. The Pragmatic Range of Parody". A Theory of Parody: The Teachings of Twentieth-Century Art Forms. New York: Methuen. ISBN 978-0-252-06938-3.

20. Joachimides, Christos M. and Rosenthal, Norman and Anfam, David and Adams, Brooks American art in the 20th century: painting and sculpture - 1993.

21. Антиплагиат за повышение качества образования [Електронний ресурс] – Режим доступу до ресурсу: https://www.antiplagiat.ru/corporate/education

22. Plagiarism Detection in Schools & Universities [Електронний ресурс] – Режим доступу до ресурсу: https://unicheck.com/plagiarism-check-for-k-12-and-higher-education

23. Plagiatinformкак система антиплагиат в научных работах[Електронний ресурс] – Режим доступу до ресурсу: https://cyberleninka.ru/article/n/plagiatinform-izbavit-ot-plagiata-v-nauchnyh-rabotah/viewer

24. Clarke, Roger (2006). "Plagiarism by academics: More complex than it seems". Journal of the Association for Information Systems. ISSN 1536-9323.

25. Eaton, S. E. Comparative Analysis of Institutional Policy Definitions of Plagiarism: A Pan-Canadian University Study. Interchange: A Quarterly Review of Education - 2017.

26. Colella-Sandercock, J. A.; Alahmadi, H. W. "Plagiarism education: Strategies for instructors". International Journal of Learning, Teaching and Educational Research - 2015.

27. Sattler, Sebastian; Wiegel, Constantin; Veen, Floris van. "The use frequency of 10 different methods for preventing and detecting academic dishonesty and the factors influencing their use". Studies in Higher Education - 2017.

28. Dawes, John (20 July 2018). "Practical Prevention of Plagiarism for University Faculty & Management – 14 Tactics". SSRN 3209034.

29. Ireland, Chris; Huddersfield, University of; UK; English, John; Huddersfield, University of; UK. "Let Them Plagiarise: Developing Academic Writing in a Safe Environment". Journal of Academic Writing – 2011.

30. Leung, C. H., & Cheng, S. C. L. An instructional approach to practical solutions for plagiarism. Universal Journal of Educational Research - 2017.