

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «магістр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Розробка сервісів первинної перевірки
пояснювальних записок до дипломних робіт»**

Виконав студент групи ПЗМ-22-2
ЖЕЛІЗКО Віктор Вікторович

Керівник роботи:
САВІЦЬКИЙ Роман Святославович

Рецензент:
ЄФРЕМОВ Юрій Миколайович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

зав. кафедри інженерії

програмного забезпечення

Тетяна ВАКАЛЮК

«18» вересня 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ЖЕЛІЗКО Віктор Вікторович**

Керівник роботи: **САВІЦЬКИЙ Роман Святославович**

Тема роботи: **«Розробка сервісів первинної перевірки пояснювальних записок до дипломних робіт»**,

затверджена Наказом закладу вищої освіти від **«18» вересня 2023 р. № 479/с**

Вихідні дані для роботи: **академічні роботи студентів, а також вимоги до оформлення академічних робіт, алгоритми та методи первинної перевірки дипломних робіт у форматі PDF, елементи оформлення дипломних робіт, які підлягають автоматизованій перевірці.**

Консультанти з магістерської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1			
2			
3			

РЕФЕРАТ

Випускна кваліфікаційна магістерська робота включає аналіз та вибір інструментарію розробки сервісів первинної перевірки пояснювальних записок до дипломних робіт.

Мета роботи полягає в розробці сервісів первинної перевірки пояснювальних записок до дипломних робіт. Результати дослідження можуть використовуватись в подальших наукових розробках сервісів з перевірки дипломних робіт, статей, звітів студентів.

Пояснювальна записка складається зі вступу, трьох розділів, переліку умовних позначень та списку використаної літератури. Записка дипломної роботи містить 18 ілюстрацій, 5 таблиць, 2 формули та 41 посилання на джерела бібліографії. Повний обсяг роботи становить 72 сторінки, з них 59 сторінок основного тексту.

В першому розділі визначено елементи оформлення дипломних робіт, які підлягають автоматизованій перевірці, проведено аналіз аналогів та вибір програмних засобів реалізації. Досліджено поняття парсингу PDF-файлів та проведено вибір алгоритмів перевірки оформлення академічних робіт.

В другому розділі описано вимоги до сервісу та його складових, а також вимоги та розробка алгоритмів для реалізації мікросервісу перевірки рамок, аббревіатур, додатків та підписів до рисунків і таблиць.

В третьому розділі надано інструкції з встановлення та налаштування сервісу, описано тестування сервісу та представлені економічні результати дослідження.

КЛЮЧОВІ СЛОВА: ПАРСИНГ, PDF, АВТОМАТИЗОВАНА ПЕРЕВІРКА РОБІТ, JAVASCRIPT, PFDJS-DIST.

ABSTRACT

The graduation qualification master's thesis includes analysis and selection of tools for developing services for the primary verification of explanatory notes for the thesis.

The work aims to develop services of primary verification of explanatory notes to the thesis. The study results can be used in further scientific developments of services to verify dissertations, articles, and reports of students.

The explanatory note consists of an introduction, three chapters, a list of conditional abbreviations, and a list of used literature. The thesis note contains 18 illustrations, 5 tables, 2 formulas, and 41 references to bibliographic sources. The total volume of the work consists of 72 pages, of them 59 pages of the main text.

The first section defines the elements of the thesis certification, which are subject to automated verification, analysis of analogues, and software tools' selection for implementation are carried out. The concept of parsing PDF files has been studied, and the selection of algorithms for verifying the design of academic works has been carried out.

The second section describes the requirements for the service, and its' components, as well as the development of algorithms for implementing a microservice to verify frames, abbreviations, annexes and signatures to drawings and tables.

The third section contains a description of the service for verifying the certification of the thesis, instructions for installing and configuring the service, testing and presentation of the study results.

KEYWORDS: PARSING, PDF, AUTOMATED WORKS VERIFICATION, JAVASCRIPT, PDFJS-DIST.

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	6
ВСТУП	7
РОЗДІЛ 1. ФОРМУЛЮВАННЯ ЗАВДАННЯ ТА ВИВЧЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ.....	10
1.1 Визначення елементів оформлення дипломних робіт, які підлягають автоматизованій перевірці.	10
1.2 Аналіз аналогів та вибір програмних засобів реалізації.	13
1.3 Парсинг PDF-файлів	18
1.4 Вибір алгоритмів перевірки оформлення академічних робіт.....	26
Висновки до 1 розділу	30
РОЗДІЛ 2. ТЕХНІЧНЕ ВТІЛЕННЯ СЕРВІСУ ДЛЯ ПЕРЕВІРКИ ОФОРМЛЕННЯ ДИПЛОМНИХ РОБІТ З ВИКОРИСТАННЯМ TYPESCRIPT	31
2.1 Вимоги до сервісу та його складових	31
2.2 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки рамок.....	37
2.3 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки аббревіатур.....	40
2.4 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки підписів до рисунків та таблиць	44
2.5 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки додатків	47
Висновки до розділу 2	48
РОЗДІЛ 3. ОПИС ТА ЗАСТОСУВАННЯ СЕРВІСУ	49
3.1 Опис використання та налаштування сервісу	49
3.2 Тестування та представлення результатів роботи сервісу.....	53
3.3 Оцінка ефективності впровадження сервісу	56
Висновки до 3 розділу	63
ВИСНОВКИ.....	64
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	66

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – application programming interface

CSV – comma-separated values

HTML – HyperText Markup Language

JS – JavaScript

NLTK – natural language toolkit

OCR – optical character recognition

PDF – Portable Document Format

TS – TypeScript

XML – extensible markup language

АВП – автоматизована перевірка

АП – аналітичний парсинг

Б – бакалавр

БД – база даних

ВЗП – візуальний парсинг

ВП – виробнича практика

ІПЗ – інженерія програмного забезпечення

КР – кваліфікаційна робота

М – магістр

ОКР – освітньо-кваліфікаційний рівень

ПЗ – програмне забезпечення

ПП – переддипломна практика

СП – семантичний парсинг

ШІ – штучний інтелект

ВСТУП

Актуальність теми: У сучасному інформаційному суспільстві, з поширенням інформаційних технологій, автоматизована перевірка дипломних робіт набуває зростаючої вагомості, сприяє підвищенню ефективності та точності перевірки робіт, дозволяє автоматично виявляти відхилення від встановлених стандартів оформлення та забезпечує високий рівень якості оформлення. Крім того, автоматизована перевірка значно економить час викладачів та наукових керівників.

В сучасному світі, де наукова діяльність та освіта знаходяться в постійному розвитку, автоматизована перевірка дипломних робіт у форматі PDF стає особливо актуальною. Завдяки цьому підходу забезпечується швидка та ефективна перевірка документів з використанням передових технологій. Інформаційне суспільство вимагає швидкості, точності та доступності, тому автоматизація процесу перевірки дипломних робіт є критично важливою.

Окрім економії часу для викладачів і наукових керівників, автоматизована система перевірки дозволяє виявляти помилки та недоліки при оформленні дипломних записок та допомагає зрозуміти чи академічні роботи відповідають встановленим стандартам оформлення та науковим вимогам ще на етапі написання роботи.

Процес створення автоматизованої системи для перевірки дипломних робіт є важливим і динамічним напрямом досліджень. Технологія має потенціал суттєво вплинути на якість освіти, забезпечуючи зростання академічного рівня студентів та науковців. Розробка вищеописаної системи вимагає комплексного підходу та поєднання різних аспектів інформаційних технологій для досягнення максимальної ефективності та точності.

Таким чином, робота спрямована на розробку сервісів, які будуть не тільки полегшувати процес перевірки дипломних робіт, але й сприяти підвищенню їх якості. Використання розробленого сервісу допоможе створити зручний та

ефективний інструмент для академічної спільноти, сприяючи подальшому розвитку науки та освіти.

Мета роботи полягає в розробці сервісу для автоматизованої перевірки дипломних робіт у форматі PDF.

З метою досягнення поставленої задачі були сформульовані наступні **завдання**:

- визначення елементів оформлення дипломних робіт, які підлягають автоматизованій перевірці. Аналіз аналогів та вибір програмних засобів реалізації;
- вибір алгоритмів перевірки оформлення академічних робіт;
- визначення вимог до сервісу та відображення результатів перевірки;
- визначення вимог та розробка алгоритмів для реалізації мікросервісів перевірки рамок, аббревіатур, додатків і підписів до рисунків та таблиць;
- опис налаштування і тестування сервісу;
- оцінка ефективності впровадження сервісу.

Об'єктом дослідження виступають академічні роботи студентів, а також вимоги до оформлення академічних робіт.

Предметом дослідження є алгоритми та методи первинної перевірки дипломних робіт у форматі PDF, елементи оформлення дипломних робіт, які підлягають автоматизованій перевірці та вимоги до оформлення академічних робіт.

Наукова новизна даної роботи полягає в розробці технології, яка поєднує різні аспекти перевірки дипломних робіт, зокрема перевірку на наявність та заповнення рамок, підписів до рисунків і таблиць, перевірку аббревіатур. Використання розробленого рішення дозволить автоматизувати процес перевірки наукових робіт, покращити їх якість та сприяти подальшому науковому розвитку. На цей час досі не існує подібного сервісу і вищенаведена розробка ведеться вперше.

Практична цінність полягає в наданні користувачам зручного та ефективного інструменту для перевірки академічних робіт на відповідність академічним стандартам.

Апробація. За темою магістерської роботи було опубліковано тези конференції [1] та подано статтю до друку [2].

РОЗДІЛ 1. ФОРМУЛЮВАННЯ ЗАВДАННЯ ТА ВИВЧЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Визначення елементів оформлення дипломних робіт, які підлягають автоматизованій перевірці.

Завдяки постійному розвитку освітньо-наукового середовища, написання дипломних робіт стає важливою та неодмінною частиною навчального процесу. Забезпечення високого рівня граматичної коректності, структурованості та наукової виразності є вимогами, які висуваються до студентських робіт. Однак, попри завдану мету та об'єктивність виконання, можливість допущення орфографічних, граматичних та стилістичних помилок завжди існує. З метою підвищення якості академічних текстів та спрощення процесу їх редагування були розроблені спеціальні програми, спрямовані на автоматизовану перевірку дипломних робіт на наявність недоліків. Програми використовують алгоритми обробки природної мови, що дозволяє здійснити аналіз тексту на різних рівнях: від виявлення орфографічних неточностей до перевірки логічної структури.

Автоматизована система – це сукупність складників, призначених для досягнення визначеної мети з використанням засобів автоматизації [3].

Згідно з визначенням про автоматизовану систему можна підсумувати, що автоматизована перевірка (АВП) (також відома як автоматизована верифікація або автоматична перевірка) означає процес використання програмних засобів і алгоритмів для автоматичної перевірки або оцінки певних характеристик, які зазвичай вимагали б ручної перевірки або оцінки вручну. Комп'ютерна програма виконує процес перевірки або оцінки самостійно, без прямого втручання людини. Автоматизована перевірка елементів оформлення дипломних робіт відіграє важливу роль у забезпеченні наукової доброчесності та високої якості зданих робіт.

Одним із ключових етапів АВП є аналіз структури та відповідності документа вимогам установи чи університету. Система повинна перевіряти наявність та правильне заповнення рамок, правильність підписів до рисунків та таблиць, правильність оформлення абревіатур, оформлення джерел бібліографії в списку використаних джерел та в документі, оформлення елементів змісту в документі та змісті. Перевірка вищевказаних елементів забезпечує дотримання формальних вимог до документа. Наступним етапом АВП є те, що автоматизована система повинна ретельно аналізувати структуру вступу, висновків та інших розділів роботи відповідно до встановлених стандартів. Враховуючи актуальні потреби університетів та наукових установ, автоматизована перевірка може стати важливим інструментом для забезпечення якості та наукової доброчесності дипломних робіт.

Застосування автоматизованої перевірки дозволяє ефективно здійснювати контроль та перевірку без великих зусиль людських ресурсів, тим самим забезпечуючи більш точні та однорідні результати перевірки документів. Однак, необхідно врахувати, що автоматизована перевірка не замінює роль наукового співтовариства та експертної оцінки. Важливо пам'ятати про контекстуальні особливості досліджень, які можуть не бути враховані програмами перевірки. Тому системи АВП повинні сприяти, а не заміщати, науковій етиці та академічним стандартам.

Розвиток і вдосконалення вищезазначених систем забезпечує надійність та ефективність перевірки, сприяючи динамічному розвитку науки й освіти. Застосування автоматизованої перевірки елементів оформлення дипломних робіт може виявитися корисним і для самого процесу навчання студентів. Шляхом автоматичної перевірки та надання звіту про помилки чи недоліки у структурі та форматуванні роботи, студентам надається можливість отримати конструктивний фідбек та покращити свої академічні навички. Такий інструмент допомагає стимулювати самостійність та вдосконалення в науковій діяльності, що позитивно впливає на якість наукових праць у майбутньому.

Сприймаючи звіти та рекомендації, надані системами АВП, студенти можуть виявити практичні різниці між вимогами форматування наукових праць та своїм поточним підходом, що дозволить їм активно працювати над усуненням допущених помилок та недоліків та сприятиме підвищенню професіоналізму і точності у поданні результатів своїх досліджень.

Можливість порівняти результати своєї роботи з встановленими стандартами допомагає студентам більш точно розуміти вимоги наукової спільноти та виробляти необхідні навички для досягнення високих наукових стандартів. Однак, важливо пам'ятати, що автоматизована перевірка не повинна розглядатися як єдина міра для оцінки якості наукової роботи студентів. Результати перевірки мають інтерпретуватися в контексті дослідження та звітування, оскільки деякі аспекти наукових досліджень можуть бути складні для визначення або оцінки автоматичними системами.

Автоматизована перевірка не здатна виявити суттєві креативні або інноваційні аспекти досліджень студентів, що вимагають експертної оцінки та аналізу від зазначених учасників наукової спільноти. Окрім перевірки формальних аспектів, автоматизовані перевірки можуть виявляти потенційні проблеми з оригінальністю досліджень. Застосування алгоритмів антиплагіату дозволяє виявляти можливі випадки копіювання чи недостатньо самостійного наукового дослідження, що сприяє підвищенню довіри до наукових праць та запобіганню недобросовісному поведженню.

Проте, при застосуванні автоматизованої перевірки необхідно уникати абсолютизації та робити певні застереження. Системи можуть показувати результати залежно від вибраного алгоритму та налаштувань, тому необхідно розуміти їх обмеження та не використовувати їх результати, як єдиний показник якості дослідження. Важливо забезпечити баланс між АВП та експертною оцінкою, залучаючи викладачів та науковців до остаточного аналізу наукових праць.

Застосування програмних інструментів дозволяє автоматизувати процес перевірки дипломних робіт, звільняючи людські ресурси для інших важливих

завдань, пов'язаних з науково-дослідною та навчальною діяльністю. Можна зазначити, що автоматизована перевірка елементів оформлення дипломних робіт є лише однією зі складових частин навігації у сфері наукових досліджень.

Напрямки розвитку таких систем повинні орієнтуватися на надання комплексного та мультимодального аналізу, що охоплюватиме як формальні аспекти, так і змістову адекватність наукових праць.

Важливим аспектом розвитку автоматизованої перевірки є постійне оновлення та адаптація систем до змін у вимогах та стандартах оформлення. Сфера наукових досліджень та академічної практики постійно змінюється і з цим пов'язані нові вимоги до структури та оформлення документів. Тому важливо забезпечити гнучкість та легкість внесення змін у програмні рішення, щоб системи автоматизованої перевірки завжди були актуальними та ефективними.

1.2 Аналіз аналогів та вибір програмних засобів реалізації.

Сучасний швидкий темп розвитку технологій вимагає від бізнесу та індивідуальних користувачів надійних і ефективних програмних рішень. При виборі програмних засобів для реалізації конкретного завдання важливо провести аналіз аналогів, щоб забезпечити оптимальність та придатність обраного інструменту.

До найпопулярніших систем перевірки тексту відноситься Grammarly [4]. Платформа являє собою інноваційний інструмент, призначений для підтримки користувачів у вдосконаленні їхнього письмового спілкування. Grammarly використовує штучний інтелект та алгоритми машинного навчання для автоматичного аналізу текстів та надання рекомендацій щодо граматики, стилю, орфографії та пунктуації. Система надає широкий спектр можливостей і користується популярністю серед студентів, професіоналів і всіх, хто прагне зробити своє письмо більш точним і читабельним.

Grammarly має численні переваги. Зручний у користуванні інтерфейс значно полегшує роботу користувачів на платформі (рис. 1.1).

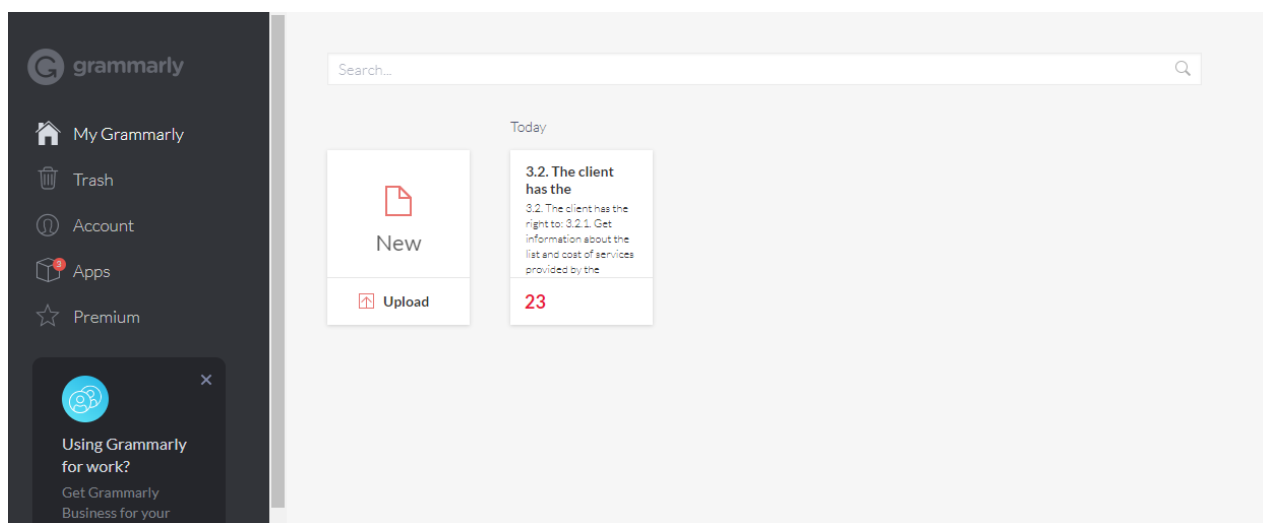


Рис. 1.1 Інтерфейс Grammarly

Даний застосунок забезпечує надійну перевірку граматики та правопису, допомагаючи уникнути простих помилок, які можуть зіпсувати враження від тексту. Система аналізує стиль та синтаксис, що допомагає підвищити читабельність тексту та зробити його більш професійним. Також, Grammarly пропонує рекомендації щодо використання різних лексичних одиниць, що сприяє збагаченню словникового запасу користувачів (рис. 1.2).

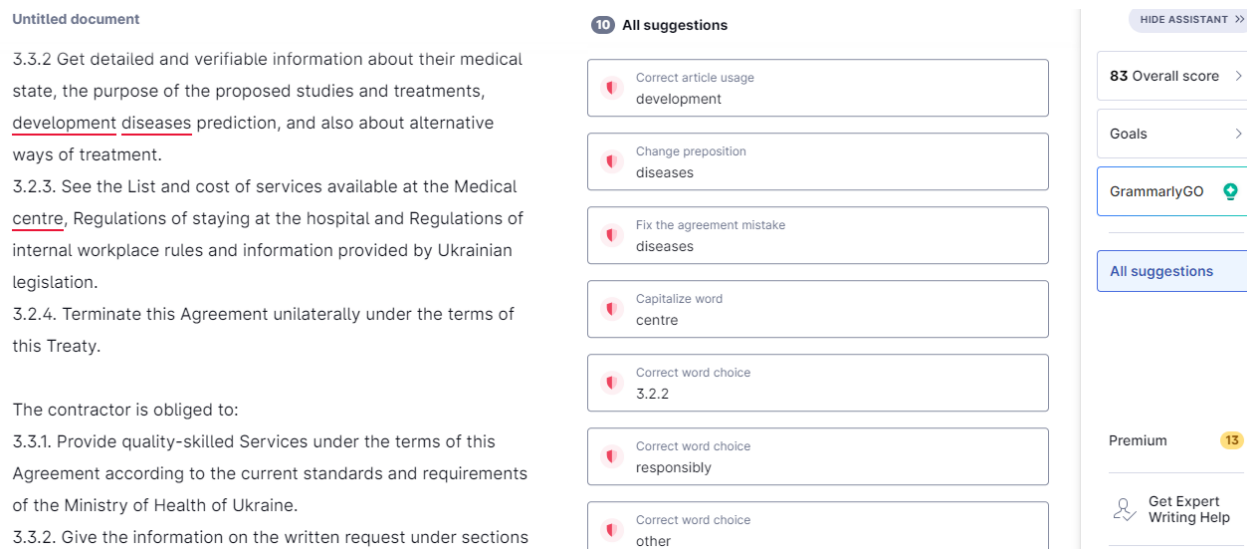


Рис. 1.2 Перевірка тексту в Grammarly

Крім цього, платформа надає можливість адаптації до конкретного типу контенту, наприклад, наукових досліджень, бізнес-літератури або художніх текстів.

Попри численні переваги, Grammarly має свої недоліки. Для повного доступу до всіх функцій платформи, включаючи розширену перевірку структури тексту та аналіз смислової навантаженості, потрібно придбати платний план, що може бути вартісним для багатьох користувачів. Хоча Grammarly здатна виявляти багато граматичних помилок, вона не завжди здатна визначити контекстуальні неточності та стилістичні відхилення. Користувачі повинні мати на увазі, що платформа надає рекомендації, але остаточне рішення про внесення змін до тексту залишається за користувачем. Не менш популярним є веб сервіс та додаток для редагування тексту, призначений для покращення якості письмового контенту – Hemingway Editor [5].

Платформа отримала свою назву на честь відомого американського письменника Ернеста Хемінгуея, відомого своєю лаконічністю та прямим стилем письма. Hemingway Editor пропонує інструменти для аналізу та виправлення структури речень, використання слова та загального стилю, з метою зробити тексти більш чіткими та лаконічними. Інтерфейс Hemingway Editor простий, інтуїтивний та спрямований на полегшення процесу редагування тексту, сприяючи поліпшенню якості письмового контенту (рис. 1.3).

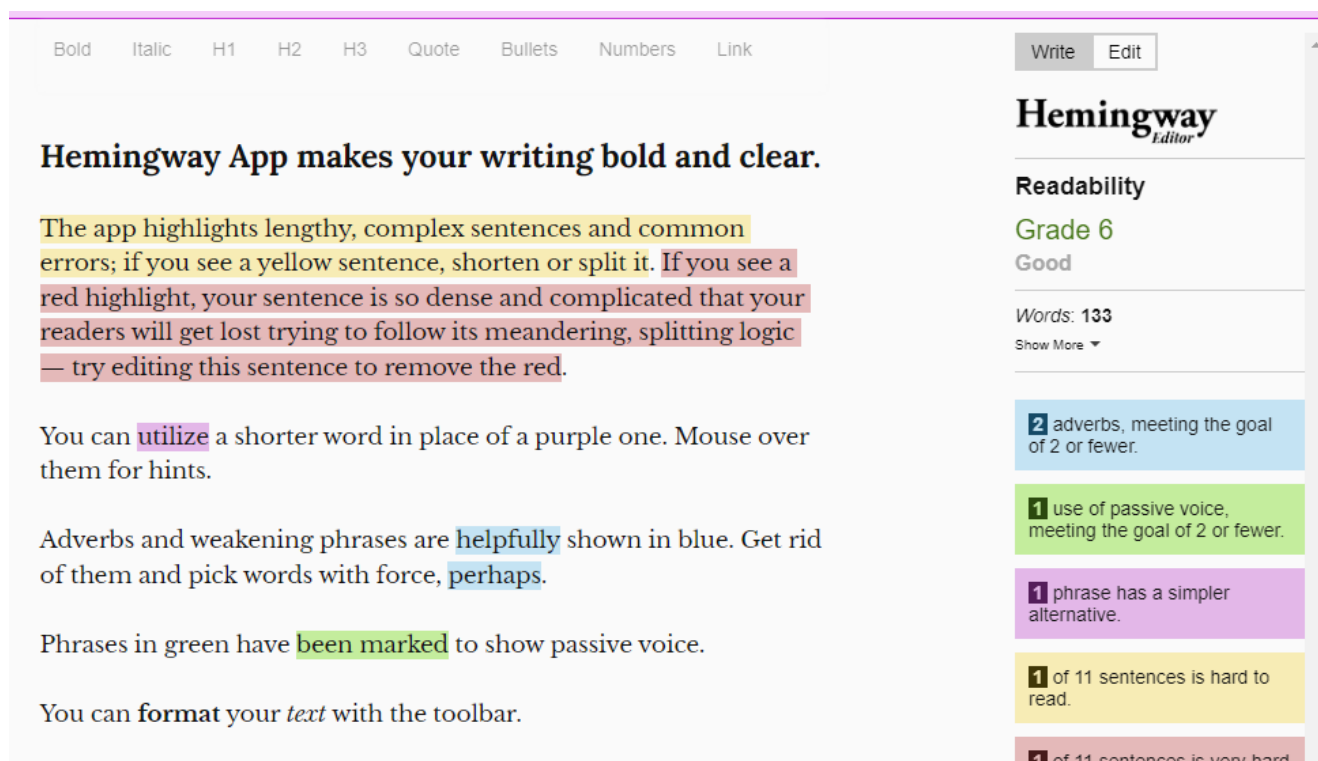


Рис. 1.3 Інтерфейс Hemingway Editor

Hemingway Editor має кілька значущих переваг. Застосунок надає детальний аналіз структури речень, допомагаючи ідентифікувати та виправляти складні конструкції, а також рекомендує заміну довгих речень більш короткими та зрозумілішими аналогами. Hemingway Editor надає оцінку загальної складності тексту за допомогою різних метрик, таких як: Flesch-Kincaid – читабельність, що допомагає авторам створювати контент, який легко зрозуміти аудиторії (рис. 1.4).

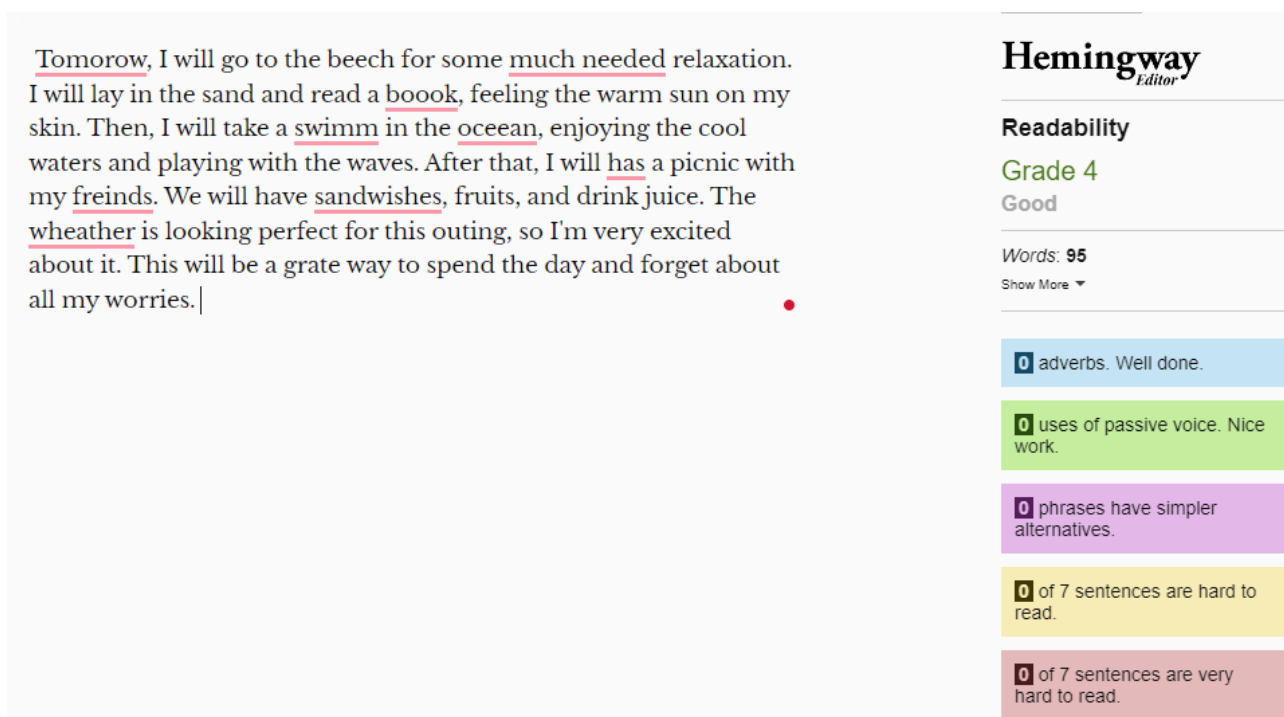


Рис. 1.4 Перевірка тексту за допомогою Hemingway Editor

Платформа підкреслює недоліки та пропонує альтернативні варіанти використання слів, що сприяє покращенню стилю та розширенню словникового запасу.

Незважаючи на свої переваги, Hemingway Editor має певні обмеження. Платформа спрямована на англomовний контент, тому її користь для користувачів, які пишуть іншими мовами, обмежена. Hemingway Editor не надає виправлень для граматичних або правописних помилок. Отже, користувачам може бути необхідно використовувати додаткові інструменти для повного редагування тексту.

Платформи, описані вище, надзвичайно корисні та можуть значно допомогти в написанні текстів, особливо англomовних, але вони не здійснюють перевірку академічних робіт на правильність оформлення. Застосунок має на меті перевіряти структуру PDF документів та надавати звіт по знайдених помилках. З цією метою було вирішено розробити інноваційний сервіс первинної перевірки пояснювальних записок до дипломних робіт.

Для реалізації сервісу з перевірки PDF документів було вирішено обрати мову програмування TypeScript (TS), яка виходить за межі звичайного відношення до JavaScript (JS) та надає всі можливості JS, а також додатковий рівень поверх цього: систему типів TS [6].

TS володіє статичною типізацією, що надає можливість визначення типів змінних, параметрів функцій та об'єктів на етапі написання коду, що дозволяє виявляти потенційні помилки ще до виконання програми, сприяючи вищому рівню безпеки та зрозумілості коду.

Також, слід відзначити високий рівень підтримки з боку інструментів розробки та середовища роботи. TS інтегрується з популярними редакторами, такими, як Visual Studio Code, та забезпечує розширену функціональність таку, як автодоповнення, перевірка типів у реальному часі та відладка коду.

Ще однією значущою перевагою TS є можливість використання новітніх фіч ECMAScript, оскільки вона є суперсетом JS. Сприяючи використанню стрілкових функцій, `async/await`, деструктуризації та інших синтаксичних покращень, TS дозволяє писати сучасний код, який легко читати та розширювати.

Крім того, TS активно підтримує концепції об'єктно-орієнтованого програмування такі, як: класи та інтерфейси, що сприяє створенню більш структурованих та підтримуваних кодових баз, особливо у великих проектах.

1.3 Парсинг PDF-файлів

В сучасному світі PDF-файли є одним з найпоширеніших форматів для зберігання й обміну документами. Вони можуть містити текст, зображення, таблиці, діаграми та інші дані. PDF-файли часто використовуються у науковій, технічній та діловій сфері. Зручні для перегляду та друку, але можуть бути складними для обробки за допомогою програмного забезпечення.

Парсинг – це процес, при якому відбувається перетворення даних у формат, більш зручний для читання користувачем [7].

PDF-парсер (іноді також званий PDF-скрепером) – програмне забезпечення, яке можна використовувати для вилучення даних з PDF-документів. Парсери можуть бути у вигляді бібліотек для розробників чи як автономні програмні продукти для кінцевих користувачів.

PDF-парсери, в основному, використовуються для вилучення даних з пакету PDF-файлів. Ручне введення даних (копіювання та вставка) є поширеною альтернативою, коли потрібно витягти дані з невеликої кількості документів [1].

PDF-файли є універсальним форматом для багатьох типів документів: від книг, презентацій, звітів, буклетів до рахунків-фактур та накладних. Хоча PDF пропонує можливість вбудувати багатий медіаконтент та вкладення, парсери зазвичай використовуються для вилучення різних видів даних.

Парсинг застосовується для окремих даних (дати, номери відстеження тощо). Якщо PDF-документ створено на основі набору даних або містить поля з окремими фрагментами даних, то застосунок може точно витягти їх для вас. Додаток може розташувати дані в певному полі відповідно до поля, щоб можна було скопіювати їх в інше місце. Наприклад, якщо документ містить таблицю з даними про клієнтів, то PDF-парсер може отримати ім'я, адресу, номер телефону та email кожного клієнта в окремі поля. Це може бути корисно, якщо ви хочете автоматизувати обробку даних або перетворити їх у базу даних [1].

Табульовані дані (таблиці та списки).

Більшість сучасних PDF-парсерів можуть ідентифікувати наявність таблиць у документі, що можна вважати значним прогресом, оскільки більшість старих PDF-парсерів розглядали всі типи даних як абзаци й перетворювали таблиці в нерозбірливий текст, в результаті чого користувачам доводилося копіювати дані вручну.

Зображення.

При наявності зображень в PDF документах, PDF-парсер може вилучити окремі картинки та зберегти їх. Це допоможе відтворити зображення з цих документів деінде, оскільки позбавляє від необхідності робити кілька знімків екрана низької якості.

PDF-парсери використовуються у різних галузях: від документообігу, індексації документів до автоматизації бізнес-процесів з метою автоматичного вилучення даних з PDF-файлів. Можливість успішного парсингу файлів значною мірою залежить від природи документів, і не всі типи документів можна парсити [6].

Найбільш поширеними прикладами використання PDF-парсерів є [8-9]:

- пошукові системи;
- програми пошуку плагіату й аналізу семантики в текстах;
- інтернет-маркетинг;
- пошукове просування;
- парсинг валютних курсів і курсів криптовалют.

Технологія парсингу надає популярні рішення користувачам [10]:

- отримувати текст із файлів зображень;
- отримувати дані з PDF документів;
- отримувати текст з документів, отримувати таблиці з PDF документів.

Парсинг PDF-файлів справді дуже схожий на вибірку даних з вебсайтів. Деякі люди навіть використовують слово «PDF-скрепер» замість «PDF-парсер».

Однак, вибірка даних з вебсайтів має перевагу в тому, що вебсайти зазвичай представляють собою ієрархічно структуровані HTML-документи. Здатність отримувати доступ до HTML-тегів (наприклад, `<h1>`, `<h2>`, `<table>`, ...) робить набагато простіше для програмного забезпечення «розуміти» структуру документа. На жаль, специфікація PDF не містить ніяких структурних тегів. Наприклад, таблиця всередині PDF-файлу є просто текстом, який розташований у певному порядку. Відсутність структурних тегів ускладнює успішний парсинг PDF-файлів.

PDF-парсери можуть використовуватися для вилучення даних з PDF-файлів, які містять складні або неструктуровані дані. Для цього часто використовуються алгоритми машинного навчання або штучного інтелекту, які можуть навчитися розпізнавати різні типи даних у файлах, навіть якщо ці дані не мають чіткої структури.

Наприклад, PDF-парсер може бути навчений розпізнавати різні типи таблиць у файлах, навіть якщо таблиці не мають чітких меж або заголовків. Це може бути корисно для вилучення даних з файлів, які містять неформальні таблиці або таблиці, які були створені вручну.

PDF-парсери можуть використовуватися для вилучення даних з файлів, які містять графічні об'єкти такі, як зображення, діаграми та схеми. Для цього часто використовуються алгоритми оптичних розпізнавань символів (OCR). Ці алгоритми можуть навчитися розпізнавати текст з графічного об'єкта, навіть якщо текст не є чітко видимим.

Наприклад, парсер може бути навчений розпізнавати текст із зображенням штрих-коду на PDF-рахунку, що може бути корисно для автоматизації процесу обробки рахунків-фактур.

Сервіс може бути використаний з метою перевірки академічних робіт на правильність оформлення. В першу чергу необхідно розпарсити документ формату PDF з метою отримати дані того типу, з яким ми будемо працювати. В нашому випадку ми обрали тип стрінги, оскільки очікуємо, що вихідний результат буде повертатися масивом строк (рис. 1.5).

```

page 57 list : [
  'СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ',
  '1. Н.Р. Балик, В.І. Мандзюк Тернопільський національний педагогічний',
  'університет: Сучасні клієнт-серверні технології та їх застосування при',
  'вивченні систем управління базами даних [Електронний ресурс] – Режим',
  'доступу до ресурсу: https://fi.npu.edu.ua/files/Zbirnik\_KOSN/12/29.pdf',
  '2. Bumble [Електронний ресурс] – Режим доступу до ресурсу:',
  'https://bumble.com',
  '3. OkCupid [Електронний ресурс] – Режим доступу до ресурсу:',
  'https://www.okcupid.com',
  '4. Tinder [Електронний ресурс] – Режим доступу до ресурсу:',
  'https://tinder.com/',
  '5. Developer Mozilla: MVC [Електронний ресурс] – Режим доступу до',
  'ресурсу: https://developer.mozilla.org/en-US/docs/Glossary/MVC.',
  '6. TechTarget: MVC [Електронний ресурс] – Режим доступу до ресурсу:',
  'https://www.techtarget.com/whatis/definition/model-view-controller-MVC',
  '7. Developer Mozilla: MVC [Електронний ресурс] – Режим доступу до',
  'ресурсу: https://www.geeksforgeeks.org/mvc-framework-introduction/',
  '8. Microsoft Learn: Microservice Architecture style [Електронний ресурс] –',
  'Режим доступу до ресурсу: https://learn.microsoft.com/en-',
  'us/azure/architecture/guide/architecture-styles/microservices',
  '9. Microservice Architecture: Microservice Architecture Pattern [Електронний',
  'ресурс] – Режим доступу до ресурсу:',
  'https://microservices.io/patterns/microservices.html',
  '10. Microservice Architecture: Microservice Architecture Pattern [Електронний',
  'ресурс] – Режим доступу до ресурсу:',
  'https://microservices.io/patterns/microservices.html',
  '11. Medium: Microservices [Електронний ресурс] – Режим доступу до ресурсу:',
  'https://medium.com/hashmapinc/the-what-why-and-how-of-a-microservices-',
  'architecture-4179579423a9',
  'ІПЗ.КР.Б – 121 – 23 – ПЗ ',
  '57'
]

```

Рис. 1.5 Парсинг PDF файлу

PDF-парсери є потужним інструментом, який може бути використаний для вилучення даних з PDF-файлів. Вони можуть бути корисні для різних завдань, таких як: автоматизація бізнес-процесів, індексація документів або аналіз даних.

Підсумовуючи вищенаведені аргументи, можна сказати, що, парсинг PDF-файлів може бути складним завданням, але і дуже корисним. Завдяки парсерам користувачі можуть автоматично витягувати дані з PDF-файлів, що може заощадити час і зусилля.

Існує кілька різних алгоритмів парсингу файлів, які можна використовувати для різних цілей:

- аналітичний;
- візуальний;
- семантичний.

Аналітичний парсинг (АП) PDF-файлів належить до найпростішого типу парсингу, який полягає в аналізі структури файлу та вилучення інформації з нього. Для аналітичного парсингу можна використовувати такі інструменти, як Adobe Acrobat Reader або PDFMiner.

Аналітичний парсинг PDF-файлів виконується у два етапи:

- аналіз структури документа. На цьому етапі інструмент аналітичного парсингу аналізує структуру документа та витягує метадані;
- вилучення тексту документа. Інструмент аналітичного парсингу витягує текст документа з метаданих.

Аналітичний парсинг можна використовувати для різних цілей.

Наприклад, можна використовувати аналітичний парсинг для пошуку всіх випадків використання певного слова або фрази в документі, для перетворення PDF-файлів в інші формати, такі як текстові файли, CSV-файли або XML-файли.

АП можна використовувати для автоматизації завдань, пов'язаних з PDF-файлами. Наприклад, для автоматичного створення резюме PDF-файлів або для автоматичного завантаження файлів в базу даних.

Аналітичний парсинг має ряд переваг, даний тип парсингу є відносно простим, оскільки він не вимагає розуміння змісту файлу, може бути досить швидким, оскільки не вимагає розпізнавання тексту та вважається відносно надійним, оскільки не залежить від якості тексту.

Має ряд недоліків, таких як: відносні обмеження, оскільки він не може витягувати всю інформацію з файлу. Аналітичний парсинг може бути неточним, оскільки може не враховувати всі нюанси структури PDF-файлу.

Візуальний парсинг (ВЗП) полягає в розпізнаванні тексту та інших елементів зображення в файлі. Для ВЗП можна використовувати такі інструменти, як Tesseract або Cuneiform.

Візуальний парсинг PDF-файлів виконується у два етапи:

- розпізнавання тексту: на цьому етапі інструмент візуального парсингу розпізнає текст у файлі;

- виділення елементів оформлення: на цьому етапі інструмент візуального парсингу виділяє елементи оформлення в файлі, такі як заголовки, таблиці, зображення.

Візуальний парсинг PDF-файлів можна використовувати для різних цілей:

- вилучення тексту з PDF-файлів, які не містять структурну інформацію;
- виділення елементів оформлення;
- перетворення PDF-файлів в інші формати, такі як текстові файли, CSV-файли або XML-файли.

ВЗП має ряд переваг. Можна використовувати для обробки файлів з різним змістом і оформленням. PDF-скрепер може бути досить точним, якщо інструмент добре навчений.

Візуальний парсинг має ряд недоліків, оскільки він є більш складним, та може бути менш швидким, ніж аналітичний парсинг. ВЗП залежить від якості зображення в PDF-файлі.

Семантичний парсинг (СП) полягає в розумінні змісту PDF-файлу та вилучення інформації з нього. Для СП використовуються такі інструменти, як Stanford CoreNLP або spaCy.

Семантичний парсинг PDF-файлів виконується в кілька етапів:

- розпізнавання тексту: на цьому етапі інструмент семантичний парсинг розпізнає текст у файлі;
- розділення на речення: на цьому етапі інструмент семантичний парсинг розділяє текст на речення;
- розділення на слова: на цьому етапі інструмент СП розділяє речення на слова;
- розпізнавання частин мови: на цьому етапі інструмент СП розпізнає частини мови для кожного слова;
- визначення синтаксичних зв'язків: на цьому етапі інструмент семантичного парсингу визначає синтаксичні зв'язки між словами;

- визначення семантики: на цьому етапі інструмент семантичного парсингу визначає семантику речення.

Семантичний парсинг можна використовувати для різних цілей, таких як:

- пошук інформації: СП можна використовувати для пошуку інформації в PDF-файлах, яка не може бути знайдена за допомогою аналітичного або візуального парсингу;
- перетворення PDF-файлів: семантичний парсинг можна використовувати для перетворення файлів в інші формати, такі як JSON або XML, які містять семантичну інформацію;
- автоматизація завдань: семантичний парсинг можна використовувати для автоматизації завдань, пов'язаних з PDF, таких як: створення резюме або вилучення даних з PDF-файлів.

Семантичний парсинг володіє кількома вагомими перевагами. Підхід відзначається широким спектром застосування, оскільки він може успішно обробляти PDF-файли різних типів та різного формату. СП може бути досить точним, якщо модель семантичного парсингу має відповідне навчання. Метод може бути ефективним для обробки великих PDF-файлів.

Проте, семантичний парсинг також супроводжується певними недоліками. Впровадження СП є складнішим у порівнянні з аналітичним або візуальним парсингом. Другою важливою проблемою є можлива менша швидкість обробки, порівняно з іншими методами парсингу, а ефективність значно залежить від якості тексту в оброблюваних PDF-файлах.

Для реалізації сервісу необхідно було обрати бібліотеку для аналізу даних. З цією метою було проведено ознайомлення з найпопулярнішими бібліотеками для роботи з файлами. В статті [11] наведене порівняльне дослідження шести різних бібліотек для роботи на JS. Оскільки зазначені в джерелі [11] бібліотеки не відповідали всім вимогам для розробки сервісу, було вирішено використовувати pdfjs-dist, оскільки вона надає більше можливостей для аналізу даних.

Pdfjs-dist [12] – бібліотека, яка надає різноманітні інструменти для роботи з PDF-файлами та особливо добре підходить для нашої магістерської роботи, оскільки дозволяє легко витягувати текст, зображення та інші дані з PDF-файлів, а також генерувати нові PDF-файли з нуля. Дана бібліотека є дуже зручною у використанні, з простим та інтуїтивно зрозумілим API. Крім того, pdfjs-dist надає широкий спектр функцій для роботи з PDF-файлами, включаючи вилучення тексту, вилучення зображень, генерування PDF-файлів та маніпулювання PDF-файлами. Pdfjs-dist є добре документованою, з чіткою та лаконічною документацією, яка полегшує початок роботи з бібліотекою. Надзвичайно важливим фактом є те, що pdfjs-dist є бібліотекою з відкритим вихідним кодом, що дозволяє вільно її використовувати та модифікувати. Згідно з інформацією на офіційному сайті можна зробити висновок, що дана бібліотека надзвичайно популярна серед користувачів, оскільки має досить високу кількість завантажень за тиждень (рис. 1.6).

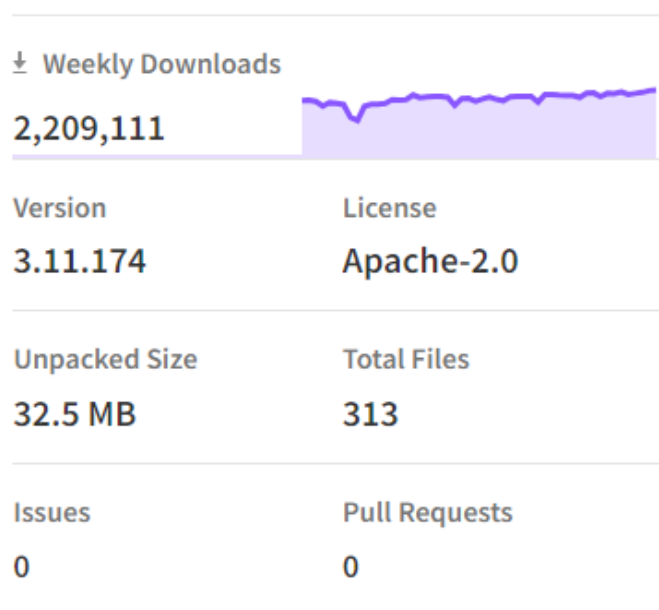


Рис. 1.6 Статистика бібліотеки pdfjs-dist [12]

Проаналізувавши аналоги сервісів та вибравши бібліотеку, з якою ми будемо надалі працювати, ми можемо переходити до аналізу та вибору алгоритмів перевірки оформлення.

1.4 Вибір алгоритмів перевірки оформлення академічних робіт

Алгоритм – точний набір інструкцій, що описують порядок дій виконавця для досягнення результату розв'язання задачі за кінцевий час [13].

В підручнику з базових алгоритмів та основ програмування наведене наступне поняття алгоритму: «Алгоритм – це кінцевий набір правил, який визначає послідовність операцій для вирішення конкретної множини завдань і має володіти п'ятьма важливими рисами: кінцевість, визначеність, введення, висновок, ефективність» [13].

Алгоритми перевірки оформлення дипломних робіт є програмними або автоматизованими методами, які застосовуються для оцінки та перевірки різних аспектів форматування, структури та стилю написання дипломних робіт, що допомагає авторам уникнути помилок та покращити професійний вигляд своїх робіт, забезпечуючи відповідність академічним стандартам та вимогам університетів та наукових установ. В програмуванні реалізацією алгоритму називається запис алгоритму за допомогою мови програмування для його виконання комп'ютером. Після того, як алгоритм реалізований, проводять його аналіз. У результаті цього можна отримати відповідь на питання: чи підходить алгоритм для розв'язання поставленої задачі. Аналіз полягає у визначенні та аналізі таких критеріїв як:

- час роботи програми, як функцію від вхідних даних;
- загальну кількість пам'яті, що необхідна для даних програми;
- загальний об'єм програмного коду;
- чи програма розв'язує коректно (правильно) поставлену задачу;
- чи стресостійка програма, тобто як добре буде поводити себе програма з некоректними вхідними даними;
- комплексність програми, тобто чи легко програму читати, розуміти та модифікувати.

І якщо останні чотири пункти частіше за все покладаються на зовнішні аналізатори (інженери-тестувальники, автоматичні системи тестування, рецензування коду), то перші два цілковито знаходяться у зоні відповідальності програміста, який створює програму [14].

Перед тим, як розпочати розробку нового алгоритму, необхідно детально дослідити вже існуючі методи та їхні обмеження, що дозволить уникнути повторення існуючих рішень і знайти точки росту для подальших вдосконалень. Ми провели дослідження по існуючим алгоритмам та занесли результати в таблицю (табл. 1.1).

Таблиця 1.1

Аналіз існуючих алгоритмів

Алгоритм	Опис	Переваги	Недоліки	Джерело
Аналіз стилів	Порівнюють стильову структуру документів.	Швидкий і простий у реалізації.	Не може виявити помилки в змісті тексту.	[15]
Аналіз англomовних текстів за допомогою парсерів	Обробляють англomовні тексти за допомогою парсерів.	Здійснення максимально швидкого розбору великих обсягів текстів.	Для роботи та використання парсера необхідна ліцензія та дистрибутив.	[16]
Алгоритми пошуку та сортування	Алгоритми сортування даних засновані на використанні масивів.	Простота реалізації, ефективність на невеликих наборах даних, ефективність на частково впорядкованих даних, стійкість можливості виконання сортування вхідної послідовності даних.	Інформація не вказана.	[17]

В 1901 році відомий швейцарський вчений, професор Поль Жаккар запропонував коефіцієнт флористичної спільності, який спочатку використовували лише в ботаніці, але згодом метод здобув поширення і нині використовується для оцінки подібності скінченних множин, в інформатиці, для пошуку подібних документів, плагіату тощо. Згодом метод був названий мірою Жаккара на честь винахідника. Міра Жаккара двох множин A_k та A_l обчислюється за формулою 1.1 [18].

$$sim_2(A_k, A_l) = \frac{|A_k \cap A_l|}{|A_k \cup A_l|} \quad (1.1)$$

Коефіцієнт Жаккара дорівнює відношенню кількості елементів перетину множин до кількості елементів їхнього об'єднання [18].

Коефіцієнт Жаккара та алгоритм шинглів використовують для пошуку схожих текстів у великому корпусі документів, а також для виявлення плагіату. Для ефективного обчислення оцінки значення коефіцієнта Жаккара використовують алгоритм MinHash [19].

Евклідова відстань - це найуживаніша міра відстані між об'єктами, яка являє собою геометричну відстань між об'єктами у багатомірному просторі [20]. Ця відстань названа на честь давньогрецького математика Евкліда, який розвивав геометрію. Для визначення Евклідової дистанції між двома точками x та y ми можемо використовувати позначення $d_{x,y}$. В такому випадку відстань між x та y задається формулою 1.2 [21]:

$$d_{x,y} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2} \quad (1.2)$$

Евклідова відстань вимірює пряму лінійну відстань між точками у просторі. Вона є найпоширенішою метрикою для вимірювання відстаней і широко використовується в різних галузях, включаючи геометрію, статистику, машинне навчання та інші області.

У інформатиці евклідова відстань часто використовується для кластеризації даних, розрахунку схожості між об'єктами або для пошуку найближчих сусідів у просторі даних. Вона допомагає визначити, наскільки близькі одне до одного об'єкти або точки в евклідовому просторі.

Евклідова відстань може бути використана в контексті пошуку плагіату для визначення схожості між двома текстами чи документами. У цьому використанні вона допомагає виміряти відстань або схожість між структурою документів чи текстами на основі внутрішнього вмісту.

Розглядаючи вищезазначену інформацію, необхідно відзначити важливість розробки нових алгоритмів для вирішення конкретних завдань, зокрема у випадку нашого експериментального дослідження, де саме цим шляхом ми визначатимемо параметри та функції наших алгоритмів.

Зазначивши, що на сьогодні існують сервіси для перевірки орфографії та плагіату в текстах, проте відсутність алгоритмів для перевірки підписів у рисунках і таблицях, правильності оформлення аббревіатур та перевірка наявності та заповнення рамок, свідчить про актуальність розробки нових методів. Зокрема, у випускних роботах важливо дотримуватися нормативноправових вимог, де наші алгоритми мають великий потенціал.

Експериментальні дослідження – один з основних способів одержання нових наукових знань. В його основу покладено експеримент, що представляє собою науково поставлений дослід або спостереження явища в точно обумовлених умовах, що дозволяє спостерігати за його ходом, керувати ним, відтворювати його кожного разу при повторенні первинних умов. Від звичайного пасивного спостереження експеримент відрізняється активним впливом дослідника на явище, що вивчається [22].

Створення нового алгоритму вимагає глибокого розуміння конкретної задачі, для якої він призначений. Розробник повинен чітко сформулювати ціль та обсяг роботи алгоритму, а також визначити ключові критерії ефективності. На важливому етапі розробки стоїть вибір оптимальної структури даних та математичних методів. Передбачення обсягу обчислень, оптимізація використання пам'яті та мінімізація часових затрат стають основоположними завданнями для девелопера.

Під час розробки важливо пам'ятати про можливості паралельності та розподіленого обчислення. Сучасні технології дозволяють використовувати

потужності багатоядерних процесорів та хмарних сервісів для поліпшення продуктивності нового алгоритму.

Завершуючи процес розробки, необхідно вивести алгоритм на тестування в реальних умовах. Відгуки та результати тестів допоможуть виявити потребу в подальших модифікаціях та доповненнях.

Шляхом безлічі вдач, невдач і інтенсивного тестування ми будемо постійно вдосконалювати алгоритми та деталі застосунку. Експериментальний підхід є надзвичайно корисним, дозволяючи нам усвідомити необхідні алгоритми, виявити пропуски й визначити слабкі місця в системі. Власне завдяки цьому етаповому вдосконаленню, ми можемо досягти високого рівня точності та ефективності наших алгоритмів, враховуючи усі основні аспекти, такі, як: перевірка правильності оформлення аббревіатур, перевірка наявності та заповнення рамок, а також правильність підписів у рисунках і таблицях. Такий експериментальний підхід є необхідним етапом в розробці, щоб забезпечити високу якість та відповідність стандартам.

Висновки до 1 розділу

В першому розділі було визначено елементи оформлення дипломних робіт, проаналізовано існуючі аналоги та алгоритми. В результаті було обрано найбільш актуальні методи оформлення академічних робіт, такі як: правильність перевірки рамок, їх наявність та заповнення, перевірка оформлення рисунків і таблиць, додатків та аббревіатур. На основі проведених досліджень в 1 розділі можна підсумувати, що на цей момент сервісів, здійснюючих такої перевірки не існує.

На основі проведених досліджень було підкреслено переваги та недоліки існуючих платформ, що допоможе нам при реалізації застосунку.

РОЗДІЛ 2. ТЕХНІЧНЕ ВТІЛЕННЯ СЕРВІСУ ДЛЯ ПЕРЕВІРКИ ОФОРМЛЕННЯ ДИПЛОМНИХ РОБІТ З ВИКОРИСТАННЯМ TYPESCRIPT

2.1 Вимоги до сервісу та його складових

Сервіси первинної перевірки пояснювальних записок до дипломних робіт важливі не лише для викладачів, але і для студентів, адже сприяють забезпеченню якості та відповідності академічним стандартам.

Вимоги до програмного забезпечення – набір вимог щодо властивостей, якості та функцій програмного забезпечення, що буде розроблено, або знаходиться у розробці. Вимоги визначаються в процесі аналізу вимог та фіксуються в специфікації вимог, діаграмах прецедентів та інших артефактах процесу аналізу та розробки вимог [23].

Функціональні вимоги (functional requirements) визначають функціональність програмного забезпечення, яку розробники повинні забезпечити, щоб користувачі змогли виконати свої завдання в межах бізнес вимог [24]. Інколи їх називають вимоги поведінки (behavioral requirements). Вони містять положення з традиційним «повинна». Наприклад, «Система повинна електронною поштою відправляти користувачу підтвердження замовлення». Функціональні вимоги описують, що розробнику необхідно реалізувати [24].

Окрім функціональних вимог існують ще нефункціональні, які визначають умови виконання функцій (наприклад, захист інформації у БД, аутентифікація доступу до ПС тощо), середовища, що безпосередньо не пов'язані з функціями, а відбивають потреби користувачів щодо їх виконання. Ці вимоги характеризують принципи взаємодії із середовищами або іншими системами, а також визначають показники часу роботи, захисту даних і досягнення якості з урахуванням рекомендацій використовуваного стандарту [25].

Розібравшись із термінологією, ми можемо перейти до опису вимог до сервісу з перевірки пояснювальних записок. Сервіс має працювати швидко та

якісно, без затримок та помилок. Головна мета сервісу - забезпечити зручність при перевірці робіт для викладачів, заощадити їх час та покращити написання академічних робіт серед студентів. Тому, надзвичайно важливо, щоб застосунок надавав результат за максимально короткий проміжок часу.

Сервіс повинен надавати докладні звіти щодо результатів перевірки, вказуючи на будь-які знайдені недоліки чи помилки. Це допомагає користувачам легше розуміти, як поліпшити свої академічні праці та уникнути порушень в майбутньому.

Одним з найважливіших аспектів розробки сервісу з перевірки академічних робіт є гнучкість інструментів перевірки, що означає, що сервіс повинен дозволяти проводити конфігурацію відповідно до потреб конкретного користувача або організації.

З метою наглядності та ясного розуміння структури сервісу, в процесі розробки була створена діаграма архітектури (рис. 2.1), на якій зображено роботу мікросервісів.

Діаграма складається з кількох типів елементів:

- круги – вхідні дані, які передаються мікросервісам;
- овали – вихідні дані мікросервісів у вигляді допоміжних конфігурацій;
- квадрати з закругленими кутами – результати перевірки (знайдені помилки);
- прямокутники – мікросервіси. Різні види перевірок, які виконуються над PDF-файлами.

Відповідними кольорами на діаграмі позначено зв'язки взаємодії між мікросервісами. Філетовим кольором позначено вхідні дані, а саме PDF і конфігурації для сервісів. Зеленим і синім – додаткові конфігурації, які ми отримуємо як результат роботи сервісів перевірки рамок і змісту.

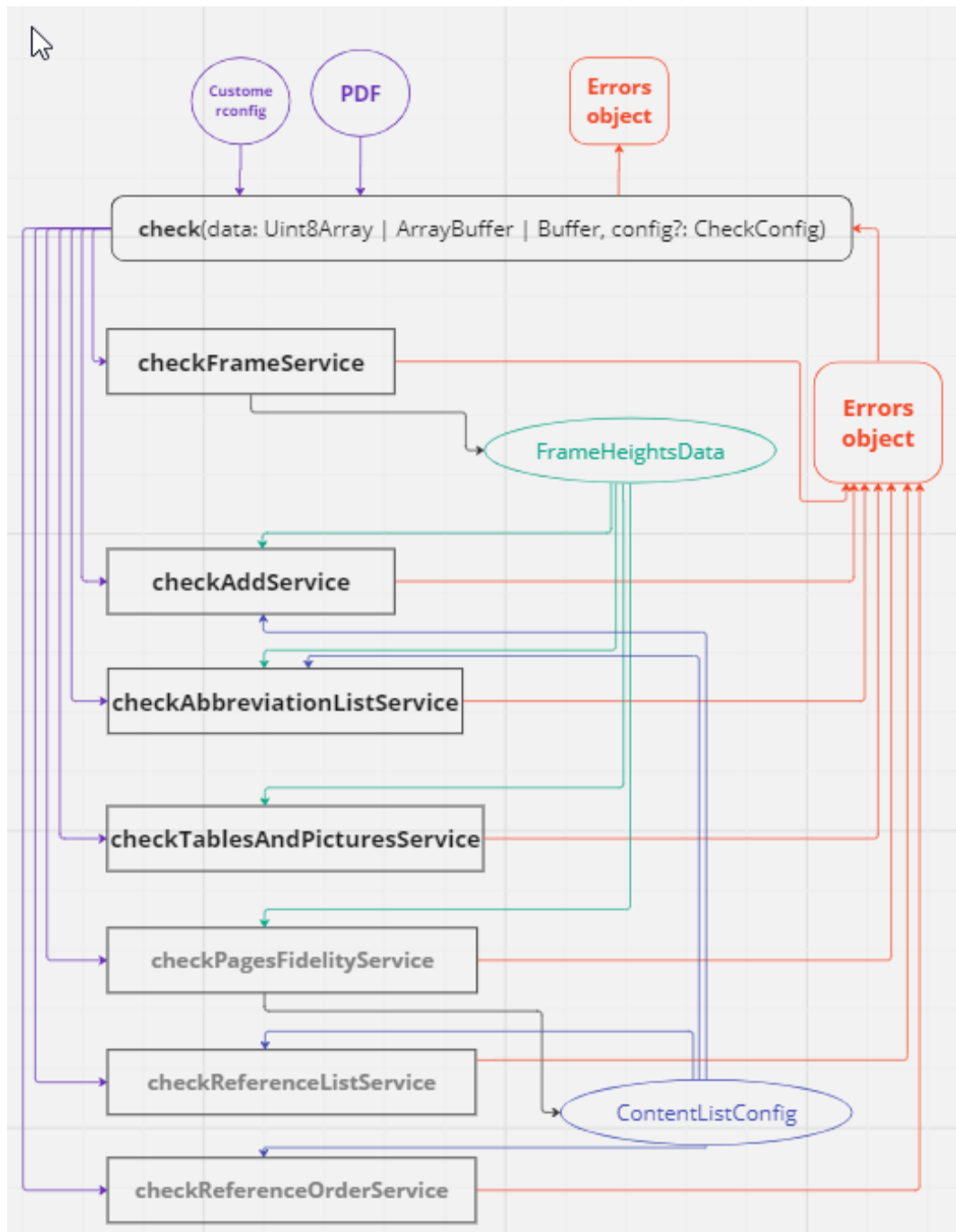


Рис. 2.1 Діаграма архітектури

При розробці застосунку важливо враховувати ряд вимог, які гарантують його ефективну та надійну роботу.

Для виявлення помилок у використанні абревіатур сервіс повинен мати чітке визначення того, які абревіатури є допустимими. Користувачі повинні мати можливість налаштувати визначення абревіатур у сервісі відповідно до їхніх потреб. Важливо, щоб сервіс міг розпізнавати абревіатури відповідно до

встановлених стандартів. Якщо в галузі програмування аббревіатура «ПЗ» означає «програмне забезпечення», то сервіс повинен розпізнавати її саме так.

У різних навчальних закладах існують свої правила щодо оформлення підписів у академічних роботах. Державний університет «Житомирська політехніка» очікує, що підписи до рисунків будуть розташовані по центру під кожним рисунком. Тому важливо, щоб сервіс міг перевіряти підписи відповідно до встановлених правил.

Іншим прикладом гнучкості є можливість задавати необхідність рамки. Рамка розглядається як короткий виклад теми академічної роботи, який зазвичай охоплює основні питання, які будуть розглянуті в роботі. Сервіс перевірки академічних робіт повинен дозволяти користувачам задавати параметр, який вимагає від робіт наявності рамки, що допоможе запобігти тому, щоб студенти писали роботи, які не відповідають темі.

Можна налаштувати параметри, які перевіряють рамки на наявність та заповнення полів, обирати групу або кваліфікацію.

Гнучкість інструментів перевірки академічних робіт є важливим фактором, який впливає на ефективність сервісу, який при наявності можливості конфігурації відповідно до потреб користувача, буде більш корисним і точним.

До переваг гнучких інструментів перевірки академічних робіт відносять:

- покращену точність перевірки: гнучкі інструменти дозволяють користувачам налаштувати сервіс відповідно до своїх конкретних потреб, що може допомогти поліпшити точність перевірки, оскільки сервіс буде перевіряти роботи відповідно до вимог користувача;
- збільшену ефективність перевірки: гнучкі інструменти дозволяють користувачам швидко і легко налаштувати сервіс, що може допомогти підвищити ефективність перевірки, оскільки користувачі не будуть витрачати час на те, щоб налаштовувати сервіс вручну;
- підвищення задоволеності користувачів: гнучкі інструменти дозволяють користувачам налаштувати сервіс відповідно до своїх потреб,

що може допомогти підвищити задоволеність користувачів, оскільки вони будуть відчувати, що сервіс відповідає їхнім вимогам.

Покриття тестами допомагає гарантувати, що нові фічі, які ми додаємо до сервісу, працюють належним чином, адже в такому випадку ми змушені думати про те, як вона повинна працювати, що допомагає нам виявляти помилки в дизайні та реалізації фічі.

Наприклад, припустимо, що ми хочемо додати до сервісу можливість перевірки наявності та заповнення рамок в академічних роботах. Ми можемо написати тест, який перевіряє, чи всі обов'язкові поля в формі були заповнені, і чи заповнені вони правильними даними, що включає такі поля, як:

- назва роботи: повинна бути чіткою та зрозумілою;
- автор: інформація повинна бути вказана;
- керівник: ім'я керівника повинно бути вказане;
- дата: повинна бути вказана;
- тема: повинна бути вказана;
- шифр групи: повинен бути вказаним.

Тест може бути написаний таким чином, щоб він перевіряв наявність кожного поля, а також правильність даних, які були в нього введені. Наприклад, тест може перевірити, чи поле «Назва роботи» не пусте, і чи довжина назви не перевищує певного значення. Якщо тест пройде успішно, ми можемо бути впевнені, що наша нова фіча працює належним чином у простих випадках. Однак, якщо тест провалиться, ми будемо знати, що нам потрібно внести зміни у нашу реалізацію, щоб фіча працювала належним чином.

Отримання об'єктивних результатів перевірки допомагають забезпечити відповідність академічних робіт встановленим стандартам, результати перевірки повинні бути несуперечливими та незалежними від суб'єктивних думок чи уподобань перевіряючого. Для досягнення цієї мети використовують чітко визначені правила перевірки, які визначаються в наборі правил, що розробляється експертами в галузі академічного письма. Правила повинні бути чіткими та зрозумілими, щоб їх можна було легко застосувати до будь-якого

документа. Автоматизовані методи перевірки дозволяють перевірити документи швидко та ефективно. Вони також допомагають забезпечити послідовність результатів перевірки.

Система відстеження помилок дозволяє зберігати інформацію про кожну знайдену помилку. Це дозволяє перевіряючому швидко знайти та усунути помилки. Правило перевірки може вимагати, щоб всі рисунки в академічній роботі були оформлені згідно вимог. Автоматизований метод перевірки може перевірити, чи відповідає кожний рисунок вимогам. Система відстеження помилок може зберегти інформацію про будь-які знайдені невідповідності та забезпечує зручний доступ до зібраної інформації, сприяючи оперативному виправленню помилок та вдосконаленню якості академічних робіт. В таблиці (табл. 2.1) представлені схеми результатів перевірки академічних робіт.

Таблиця 2.1

Результати перевірки академічних робіт

Тип помилки	Позиція помилки	Пояснення помилки
Помилка в назві рисунка.	Рисунок: «{str1}»	Назва рисунка має бути з великої літери.
Помилка в підписі до таблиці	Помилка в таблиці: «{title}»	Підпис таблиці має містити порядковий номер таблиці та заголовок

Можливість застосування сервісу на бекенді та фронтенді є однією з ключових переваг сервісу, які роблять його універсальним та гнучким рішенням. Сервіс може використовуватися для перевірки академічних робіт різного типу, включаючи курсові роботи, дипломні роботи, статті та інші наукові публікації, що може бути корисним для студентів, викладачів, аспірантів та інших осіб, які займаються академічною діяльністю. На даний момент сервіс перевірятиме лише академічні роботи рівня бакалаврів, магістрів, в майбутньому планується розширити його до вищезгаданих перевірок.

Виконана робота передбачає:

- завантаження PDF-файлу;
- вибір критеріїв перевірки;
- перегляд результатів перевірки.

Сервіс може бути корисним для студентів, які хочуть самостійно перевірити свої академічні роботи, а також для викладачів, які хочуть спростити процес перевірки робіт студентів.

Створення сервісу з перевірки дипломних робіт на правильність оформлення вимагає ретельного та стратегічного підходу до розробки. Ключовим пріоритетом повинно бути створення легкозрозумілого та масштабованого коду, щоб будь-який майбутній девелопер міг швидко вникнути в суть проекту.

Забезпечення підтримки на фронті та бекенді включає в себе систематичне виправлення помилок, регулярне оновлення залежностей та підтримку нових функцій. Крім того, створення тестів є обов'язковим етапом для забезпечення стабільності та надійності сервісу.

2.2 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки рамок

Сучасний підхід до розробки програмного забезпечення, зокрема мікросервісна архітектура, вимагає не лише високої функціональності, але й належного оформлення його компонентів. Наявні вимоги до оформлення основних компонентів мікросервісів включають ряд аспектів, які сприяють якісному аналізу, зрозумілості та ефективному спілкуванню в рамках розробки та управління.

Перш ніж починати розробку мікросервісів, необхідно знайти основні вимоги до оформлення тих елементів академічних робіт, які будуть перевірятися сервісом. Приймаючи до уваги той факт, що рамки не входять до магістерської атестаційної роботи, вимоги слід шукати у вимогах до випускної кваліфікаційної роботи бакалавра 2023.

Згідно з вимогами університету «Житомирська політехніка», факультету ФІКТ [26], рамки розміщують у колонтитулах як графічні об'єкти. Відступи рамок від країв паперу: зліва 2 см, згори, знизу, справа 5 мм.

Пояснювальні записки випускних кваліфікаційних робіт ОКР “Бакалавр”, та всіх видів звітів з практики оформляються у рамках з основним написом. Шифр документа складається з 2-х або 3-літерного коду кафедри, де виконується розробка (ПЗ), розділової крапки, 2 розрядного коду виду роботи (КР – кваліфікаційна робота, ВП – виробнича практика, ПП – переддипломна практика), розділової крапки, 1-розрядного позначення рівня (Б – бакалавр, М – магістр), розділового дефіса, цифрового коду спеціальності (121), розділового дефіса, 2-розрядного коду року захисту, риски (знак «мінус») 2-розрядної позначки типу документа (ПЗ – пояснювальна записка) [26].

Основний напис (рис. 2.2) виконується на аркуші із рефератом, на інших аркушах записки основний аркуш має вигляд, як на рисунку (рис. 2.3). На титульному аркушеві та завданні основний напис не виконується [26].

					ПЗ.КР.Б – 121 – 20 – ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розроб.		А.Г. Проценко			Побудова адаптивного алгоритму розпізнавання рукописних символів за допомогою нейронної мережі. Адаптація на рівні структури Пояснювальна записка	Літ.	Арк.	Аркуші
Керівник		Л.В. Рудюк					4	89
Рецензент		Г.В. Марчук				Житомирська політехніка, група ПП-50		
Зав. каф.		І.В. Пузехо						

Рис. 2.2 Рамка з основним написом першої сторінки [26]

					ПЗ.КР.Б – 121-20 – ПЗ			
								5

Рис. 2.3 Рамка з основним написом для інших сторінок [26]

У процесі розгляду впровадження сервісів первинної перевірки пояснювальних записок до дипломних робіт, зокрема перевірки рамок, виникла необхідність докладно розглянути аспекти, пов'язані зі структурою та зміщенням на сторінці. Важливо враховувати різноманітні аспекти такі, як: заповненість полів та відповідність визначеним стандартам.

Одним з ключових аспектів є детальна перевірка заповненості полів рамки. На даному етапі розробки ми встановлюємо за ціль перевіряти шифр спеціальності, номер сторінки та наявність обов'язково заповнених полів

«виконав» і «перевірив», які не мають бути пустими. Детальна перевірка всіх полів рамки можлива лише за умови наявності даних, таких як:

- автор;
- керівник;
- назва роботи.

Вищенаведені дані теоретично можливо отримати за допомогою конфігурацій, які передаються безпосередньо до сервісу. Щоб не ускладнювати роботу сервісу та зробити швидку перевірку було прийнято рішення відмовитись від детальної перевірки на даному етапі.

Однією з головних проблем перевірки рамок є неправильне положення рамки на сторінці. Визначення положення та розмірів рамки критично необхідне для розділення даних на дані контексту роботи та дані рамки.

Дані про розміри рамки мають бути використані іншими сервісами для правильного парсингу контексту сторінок і виключення даних рамки в інших сервісах. Отже, сервіс перевірки рамок має не тільки виводити помилки, а й надавати об'єкт даних, в якому буде вказано висоту рамки для кожної сторінки на якій є рамка.

Рамка має перед заповнені поля та поля, в які користувач вносить дані про роботу. Передзаповнені поля (ПП) можуть бути використані для знаходження зміщення рамки, а знаючи ці величини можна визначити координати полів рамки, які потребують валідації.

З метою розв'язання вищеописаної проблеми було розроблено та впроваджено новий алгоритм пошуку зміщення рамки відносно її правильного положення. Основною перевагою алгоритму є його здатність отримувати дані всіх текстових об'єктів з парсера, враховуючи координати порога сторінки відносно нижнього лівого кута. Оскільки, рамка завжди має зміщення та перед заповнені текстові поля, парсер надає необхідні дані для визначення цього зміщення. Маючи зони всіх текстових полів рамки та об'єкт парсингу PDF, ми визначаємо текстові дані в даних зонах та перевіряємо їх валідність. Поля вважаються валідними, якщо вони заповнені. Для розрахунків положення полів

та обов'язкових елементів в рамках ми регламентувалися стандартом ЄСКД (ГОСТ 2.104–2006) [27].

Алгоритм пошуку заміщення передбачає:

1. визначити положення перед заповнених полів рамки, а саме «Арк.», «Підпис», «Дата»;
2. визначити різницю зміщення ПП відносно їх правильного положення, яке визначено в стандартах та описано вище;
3. визначити середнє арифметичне значення зміщення, яке і буде очікуваною величиною. Зміщення потрібно визначати для кожної сторінки, оскільки воно може бути різним.

Алгоритм перевірки полів рамки передбачає:

1. визначення координат текстових полів рамки, які потребують перевірки;
2. визначення текстових даних в областях перевірки з урахуванням зміщення;
3. безпосередньої валідації даних.

Поля «розробник» та «керівник» не мають бути пустими, поле «шифр групи» має відповідати змінній, яку вводить користувач, поле «номер сторінки» має збігатися з номером сторінки.

2.3 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки аббревіатур

Абревіатури є поширеним способом скорочення довгих слів або фраз. Вони можуть використовуватися в PDF-файлах для полегшення читання і розуміння тексту. Читачу може бути важко читати текст у випадку, якщо аббревіатури не розшифровані.

У лінгвістичному енциклопедичному словнику під аббревіатурою розуміється «іменник, що складається з усічених слів, що входять у вихідне словосполучення, або з усічених компонентів вихідного складного слова, де

останній компонент аббревіатури може бути також цілим (не усіченим) словом» [28].

Варто зазначити різні види аббревіатур [29], які можуть зустрічатися в тексті:

- ініціалізми;
- акроніми;
- скорочення наукових ступенів та ввічливих звертань до осіб;
- латинські аббревіатури;
- мішані аббревіатури.

Акронім вимовляється як єдине слово, а не як послідовність літер. Наприклад, NASA є акронімом та означає Національне управління з Аеронавтики та Космічного Простору. Іноді акронім стає настільки поширеним, що перетворюється на звичайне слово, яке люди більше не вважають акронімом. Слова «scuba» та «laser», наприклад, походять від акронімів (self contained underwater breathing apparatus, що означає «автономний дихальний апарат під водою» та light amplification by stimulated emission of radiation, що означає «підсилення світла за допомогою вимушеного випускання випромінювання») [29].

Ініціалізми схожі на акроніми тим, що вони також утворюються за допомогою першої літери кожного слова в довшій фразі. Однак, на відміну від акронімів, ініціалізми вимовляються як послідовність літер. NFL (National Football League), наприклад, вимовляється як «ен-еф-ел» [29].

Титули, такі як «пан», «пані» та «доктор», а також назви академічних ступенів, такі як «бакалавр мистецтв» та «доктор філософії», майже завжди скорочуються. В американській англійській мові за скороченнями титулів стоїть крапка, в британській англійській мові крапка опускається [29].

Найпоширеніші скорочення титулів включають:

- Mr. = пан;
- Mrs. = пані (вимовляється «місіс»);

- Ms. = (вимовляється «міс» або «міз»);
- Sr. = старший;
- Jr. = молодший;
- Dr. = доктор.

Найпоширеніші скорочення академічних ступенів включають:

- B.S. = бакалавр наук;
- B.A. = бакалавр мистецтв;
- M.A. = магістр мистецтв;
- M.B.A. = магістр ділового адміністрування;
- Ph.D. = доктор філософії [29].

Латинських аббревіатур існує небагато і вони часто використовуються з помилками в англійській письмовій мові. *Exempli gratia*, що означає «наприклад», використовується як «e.g.», коли необхідно навести конкретні приклади узагальнення [29].

До змішаних аббревіатур відносять ті, в яких частина слова може бути з великої літери, а частина – з малої (наприклад, «iTunes», «eBook»).

Іноді в аббревіатурах можуть використовуватися як великі, так і малі літери в залежності від слова чи складу, наприклад, «iOS» або «eLearning».

Перед розробкою алгоритмів для перевірки коректності використання аббревіатур в академічних роботах, науковий підхід вимагає вивчення та узагальнення вимог до оформлення. Важливим етапом цього процесу є чітке формулювання вказівок щодо місця розташування переліку аббревіатур – від плану до вступу [30].

Окрім цього, необхідно наголосити на необхідності дотримання стандартів в написанні аббревіатур, де основним правилом є використання великих літер. Запропоновано також враховувати алфавітний порядок при розташуванні аббревіатур, що сприяє систематизації та зручності в їх сприйнятті.

Зазначаючи, що між самою аббревіатурою та її розшифруванням необхідно вставляти тире, слід підкреслити, що цей елемент оформлення відіграє важливу

роль у забезпеченні чіткості та зрозумілості тексту [30]. Такий підхід не лише визначає структурованість, а й сприяє полегшенню процесу аналізу та інтерпретації змісту.

Алгоритми розробки сервісу для перевірки правильності використання аббревіатур в дипломних роботах може бути розділений на два основні завдання, що визначають ефективність та точність процесу валідації. Перша задача полягає в перевірці наявності вказаних аббревіатур у тексті роботи, водночас друга – стосується перевірки того, чи всі вказані аббревіатури входять до переліку аббревіатур, наведеного в загальному списку.

Для виконання вищезазначеної перевірки існують цілі бази аббревіатур, а також є різні бібліотеки, які можуть впоратися із цим завданням, але вони не є ідеальними, оскільки аббревіатури можуть бути дуже складними.

Однією з таких бібліотек є Natural Language Toolkit (NLTK) для Python, яка може допомогти в роботі з текстовим аналізом.

Використовуючи NLTK, можна використовувати частину мови «POS tagging» (частини мови мітки) для визначення частини мови кожного слова в тексті. З потрібними тегами частин мови можна легко виявити аббревіатури.

На даному етапі розробки ми прийшли до висновку, що доцільно визначати лише прості аббревіатури шляхом використання шаблонних виразів. Зокрема, ми реалізуємо перевірку послідовності великих літер, враховуючи випадок, коли кількість таких літер перевищує 2 підряд. У випадку виявлення цієї характеристики, система виводитиме попередження, що вказане слово, можливо, є аббревіатурою. Важливо відзначити, що на даний момент ми не можемо категорично стверджувати, що виявлення цієї особливості є помилковим результатом, однак ми забезпечимо користувача можливістю отримати список знайдених виразів для подальшої перевірки.

2.4 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки підписів до рисунків та таблиць

Перевірка підписів до рисунків і таблиць відіграє важливу роль при валідації академічних робіт та вимагає значних часових затрат та уважності. Підписи до рисунків та таблиць визначаються як ключові елементи наукової роботи, які роз'яснюють та конкретизують представлені дані та повинні бути чіткими, інформативними й відповідати вимогам стандартів наукового видання. Їх перевірка допомагає уникнути можливих непорозумінь та невірної тлумачення результатів.

Згідно з вимогами державного університету «Житомирська політехніка» до оформлення підписів до рисунків і таблиць [31] номер рисунка формується з номера розділу та порядкового номера самого рисунка, розділених крапкою. Номер рисунка та його назву розміщують під самим рисунком без абзацного відступу по центру, курсив. Крапка вкінці не ставиться. Підпис рисунка має шрифт Times New Roman 14, інтервал між підписом рисунка і наступним текстом 6 пт., пишеться з великої літери (рис. 2.4).

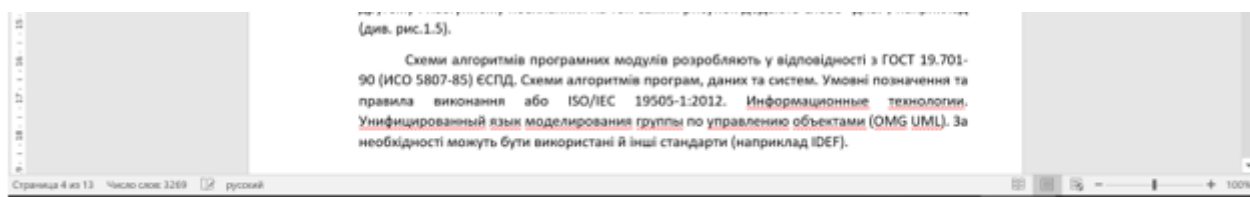


Рис 3.4 Головне вікно програми

Рис. 2.4 Приклад підпису рисунка

Окремої уваги вимагає вказівка на рисунок у реченні. Якщо рисунок є частиною фрази, його слід інтегрувати так, як у прикладі: «Аналіз використання робочого часу працівниками представлено на «рис. цифра.цифра». У випадку, коли посилання самотійно стоїть у реченні, використовуються круглі дужки: «Аналіз використання робочого часу свідчить про низьку ефективність його використання працівниками підприємства (рис. цифра.цифра)».

При другому і наступному посиланнях на той самий рисунок додають слово «див.».

У сучасному науковому письмі велика увага приділяється не лише якісному поданню інформації, але і правильному оформленню таблиць, як одному з ключових елементів наукової праці. Відповідно до встановлених стандартів та вимог [31], таблиці оформлюються згідно з наступними критеріями:

- таблицю подають безпосередньо після тексту, у якому її згадано вперше, або на наступній сторінці;
- заголовок пишуть у називному відмінку однини та розміщують над таблицею, підкреслювати його не слід. Крапку після не ставлять, заголовки підлеглих ступенів пишуть з малої літери;
- всі таблиці нумерують в межах розділу. Над тематичним заголовком, праворуч з прописної букви цілком пишуть слово «таблиця» і проставляють її порядковий номер;
- номер таблиці складається з номера розділу і порядкового номера таблиці, розділених крапкою, наприклад, «таблиця число.число» (друга таблиця другого розділу);
- при посиланні на таблицю в тексті пишеться «табл.» і її порядковий номер (наприклад, табл.число.число., див.табл.число.число), слово «таблиця» пишеться повністю лише при відсутності номера;
- номер таблиці та назву розміщують над таблицею: спочатку слово «таблиця номер.номер» (вирівнювання по правому краю рядка, курсив), а тоді назва таблиці (без абзацного відступу по центру, жирним).

Відповідно до вимог оформлення таблиць і рисунків методом експерименту було побудовано наступний алгоритм перевірки підписів до рисунків і таблиць:

1. визначення даних в межах розділів які підлягають перевірці;
2. розділення даних на дані підписів і посилань таблиць та рисунків;

3. валідація підписів таблиць;
4. валідація підписів рисунків.

Визначення даних в межах розділів які підлягають перевірці.

Сервіс перевірки змісту надає зручний інструмент – масив елементів змісту. За допомогою масиву елементів змісту визначаємо початок та кінець розділів та зберігаємо. Використовуючи початок і кінець розділів, парсимо сторінки, як масив рядків в межах розділу, які будемо перевіряти на наявність посилань та підписів до рисунків і таблиць.

Розділення даних на дані підписів і посилань таблиць та рисунків.

За допомогою регулярного виразу знаходимо всі рядки, в яких містяться посилання на рисунки та таблиці. Для цього шукаємо відповідно значення, які співпадають з «рис» або «табл» та зберігаємо в окремі масиви даних для подальшого аналізу.

Валідація підписів таблиць.

Послідовно перевіряємо масив рядків посилань та підписів таблиць, розділяючи їх на рядки підписів та рядки посилань.

Рядок підпису має починатися зі слова «таблиця» та містити її порядковий номер, наприклад, «Таблиця число.число». Перевіряємо відповідність правильності формату підпису. Перевіряємо чи є посилання на дану таблицю, яке має розташовуватися перед посиланням на неї. Для перевірки посилання використовуємо номер розділу і порядковий номер знайденої таблиці. Відповідно перевіряємо послідовність заголовків та посилань на таблиці.

Валідація підписів рисунків.

Перевіряється аналогічно до валідації підписів і посилань таблиць з урахуванням особливостей оформлення рисунків. Рисунки мають містити назву рисунка з великої літери після порядкового номера рисунка через пробіл, наприклад «рис – крапка – пробіл – номер розділу – крапка – номер рисунку».

2.5 Вимоги та розробка алгоритмів для реалізації мікросервісу перевірки додатків

Додатки є невід'ємною частиною дипломної роботи, вони містять допоміжну інформацію, яка може бути корисною для читача. Додатки повинні бути добре оформленими, структурованими та містити вірогідну інформацію. Це допоможе отримати повне уявлення про тему дослідження.

На сайті державного університету «Житомирська політехніка» прописано вимоги до оформлення додатків. Після останньої сторінки списку джерел перед додатками необхідно помістити чистий лист паперу, на якому посередині великими буквами написати «ДОДАТКИ». Номер сторінки на цьому аркуші не ставиться. У змісті роботи можна вказувати як кожне з наявних у роботі додатків з відповідними їм сторінками, так і перший лист додатків [31].

Додатки позначаються заголовними буквами за абеткою, за винятком літер Є, С, З, Ї, Е, І, Й, О, Ч, Ь. При необхідності текст додатка може бути розділений на розділи, у цьому випадку номер розділу додатка складається з букви додатка і поточного номера розділу додатка, розділених крапкою [31].

Додаток повинен мати узагальнюючий заголовок, надрукований угорі малими літерами з першої прописної симетрично щодо тексту сторінки. Праворуч над заголовком, вирівняним посередині малими літерами з першої прописної повинне бути написане слово «Додаток _» і прописна буква, що позначає додаток [31].

Додатки повинні мати загальну з іншою частиною роботи наскрізну нумерацію сторінок [31].

Детально проаналізувавши вимоги до оформлення додатків було вирішено проводити перевіряти на розташування додатків в алфавітному порядку, регістр, нумерацію з великої букви та на розташування назви з наступного рядка.

При оформленні дипломної роботи студенти дуже часто помиляються в регістрі додатків. Більшість випускників не звертає увагу на те, що сам розділ «додатки» має бути написане усіма великими літерами, а вже підрозділи додатків

з нумерацією мають бути з великих літер, а далі малими, окрім нумерації, наприклад «Додаток А». Ще однією частою проблемою є неправильне розташування додатків, а саме, може бути спочатку «Додаток Б», а потім «Додаток А».

Вирішивши основні задачі для виконання, можна переходити до розробки алгоритму. Якщо в змісті є додатки, необхідно розпарсити всі сторінки з наступної, після заголовку «додатки». За допомогою шаблонного виразу, треба знайти строки в яких є слово «Додаток». Наступним кроком, перевіримо, щоб заголовки відповідали шаблону: слово «додаток» з великої літери, після якого пробіл і велика літера. Останнім кроком є перевірка розташування додатків за алфавітним порядком зростання.

Таким чином, керівнику дипломної роботи не доведеться витратити час на перевірку оформлення додатків з урахуванням регістру та розташування, адже дана робота буде виконана сервісом.

Висновки до розділу 2

В другому розділі було проведено аналіз проблеми проведення первинної перевірки дипломних робіт, описано основні вимоги до сервісів перевірки на правильність оформлення, визначено основні вимоги до мікросервісів перевірки рамок, аббревіатур, додатків та підписів до рисунків і таблиць. На основі визначених вимог було розроблено інноваційні алгоритми для розробки вищезгаданих сервісів.

РОЗДІЛ 3. ОПИС ТА ЗАСТОСУВАННЯ СЕРВІСУ

3.1 Опис використання та налаштування сервісу

Дипломна робота є важливою частиною навчання в університеті. Вона є підсумком набутих знань і навичок студента, а також демонструє його здатність до самостійного дослідження. Однак, оформлення дипломної роботи також є важливим аспектом, оскільки воно має відповідати вимогам університету.

Для того, щоб допомогти студентам уникнути помилок при оформленні дипломних робіт, було створено спеціальний сервіс, який дозволяє перевірити оформлення академічної роботи на відповідність вимогам університету.

Сервіс складається з чотирьох мікросервісів:

- перевірки на правильність оформлення абревіатур;
- перевірки на наявність та заповнення рамок при необхідності;
- перевірку на правильність підписів до рисунків і таблиць;
- перевірку на правильність оформлення додатків.

Для того, щоб скористатися сервісом, необхідно буде завантажити дипломну роботу у форматі PDF. Після цього сервіс перевірить оформлення дипломної роботи й видасть звіт про результати перевірки.

Встановлення сервісу та залежностей, виконання перевірки.

Після базового розгортання проекту з бібліотекою React, створюємо всередині проекту папку `services`, в яку поміщаємо репозиторій сервісу перевірки академічних робіт. Встановлюємо додаткові залежності: `«npm»: «>=8.0.0»`, `«node»: «>=18.0.0»`, `«pdfjs-dist»: «^3.11.174»`, `«typescript»: «^4.9.5»`, `«@types/pdfjs-dist»: «^2.10.378»`. Для виконання перевірки потрібно імпортувати функцію `check` із сервісу, викликати її та передати їй два аргументи: PDF файл та конфігурації, як 2 аргумент. Файл має бути в одному із наступних форматів `ArrayBuffer`, `Uint8Array`, `Buffer`. Результатом виконання функції є об'єкт

знайдених помилок під час перевірки, які можуть бути представлені як вкладений список.

Налаштування відображення результатів перевірки.

Об'єкт результату перевірки містить ключі – назви вкладених сервісів, в яких знайдено помилки, та значення – об'єкт знайдених помилок, згрупованих за певними правилами. Для зручного відображення назв сервісів та груп помилок потрібно створити конфігурацію для підстановки назв. Ми отримаємо можливість зручного конфігурування повідомлення заголовків для помилок. На рисунку (рис. 3.1) зображено приклад конфігурації помилок.

```
export const errorMapper: ErrorMapper = {
  title: {
    [ERROR_TYPE.frame]: 'Помилки оформлення рамки',
    [ERROR_TYPE.referenceList]: 'Помилки в списку використаних джерел',
    [ERROR_TYPE.pagesFidelity]: 'Помилки в змісті',
    [ERROR_TYPE.picturesAndTables]: 'Помилки надписів рисунків та таблиць',
    [ERROR_TYPE.referenceOrder]: 'Помилки порядку літературних джерел в роботі',
    [ERROR_TYPE.abbreviation]: 'Помилки в абривіатурах',
    [ERROR_TYPE.addition]: 'Помилки в додатках',
  },
  [ERROR_TYPE.referenceList]: {
    rule0: 'Неправильна назва у СПИСКУ ВИКОРИСТАНИХ ДЖЕРЕЛ',
    rule1: 'Кількість використаних джерел',
    rule2: 'Неправильний порядок джерел',
    rule3: 'Неправильна нумерація джерел',
    rule4: 'Відсутні обов'язкові атрибути у джерелах',
    rule5: 'Відсутні обов'язкові атрибути у джерелах',
  },
  [ERROR_TYPE.referenceOrder]: {
    rule1: 'Неправильний формат чисел',
    rule2: 'Помилка в розділових знаках перед джерелом',
    rule3: 'Не може бути джерело після крапки або коми',
    rule4: 'Помилка в розділових знаках після джерела',
    rule5: 'Порядок вживання літературних джерел',
    rule6: 'Літературні джерела без посилань',
  },
}
```

Рис. 3.1 Конфігурація відображення помилок

Налаштування перевірки.

Сервіс передбачає встановлення конфігурацій, для правильної перевірки. Дані конфігурації містять додаткову інформацію для мікросервісів, які передбачають додаткові перевірки. Конфігурація дуже важлива, оскільки встановлює залежності під час перевірки, які використовуються для розпізнавання контенту в PDF. Наприклад, сервіс перевірки рамок може перевіряти на обов'язково заповнені поля або на точне співпадіння контенту в

полях. Якщо в роботі обов'язкова рамка, потрібно вказати, які поля мають перевірятися на точне співпадіння або не бути пустими.

Дані про наявність рамки в академічній роботі є обов'язковими, оскільки парсер має окремо перевіряти контент рамки та контент документа.

Налаштування конфігурації сервісу дає гнучкий механізм перевірки, який дозволяє отримати потрібні результати. Використання сервісу є універсальним рішенням для використання на стороні сервера та стороні клієнту. Застосування сервісу перевірки академічних робіт в кабінетах студентів дозволить використати дані студентів для формування конфігураційних вимог для перевірки їх академічних робіт, що підвищить рівень якості оформлення та в свою чергу зменшить навантаження на керівників робіт та перевірку норм контролю.

У разі виявлення недоліків у наявності або заповненні рамок, мікросервіс генерує докладний звіт, в якому вказані місця в документі, де виявлені проблеми, що забезпечує користувачам чітку інформацію про те, як саме можна поліпшити оформлення академічних робіт (рис. 3.2).

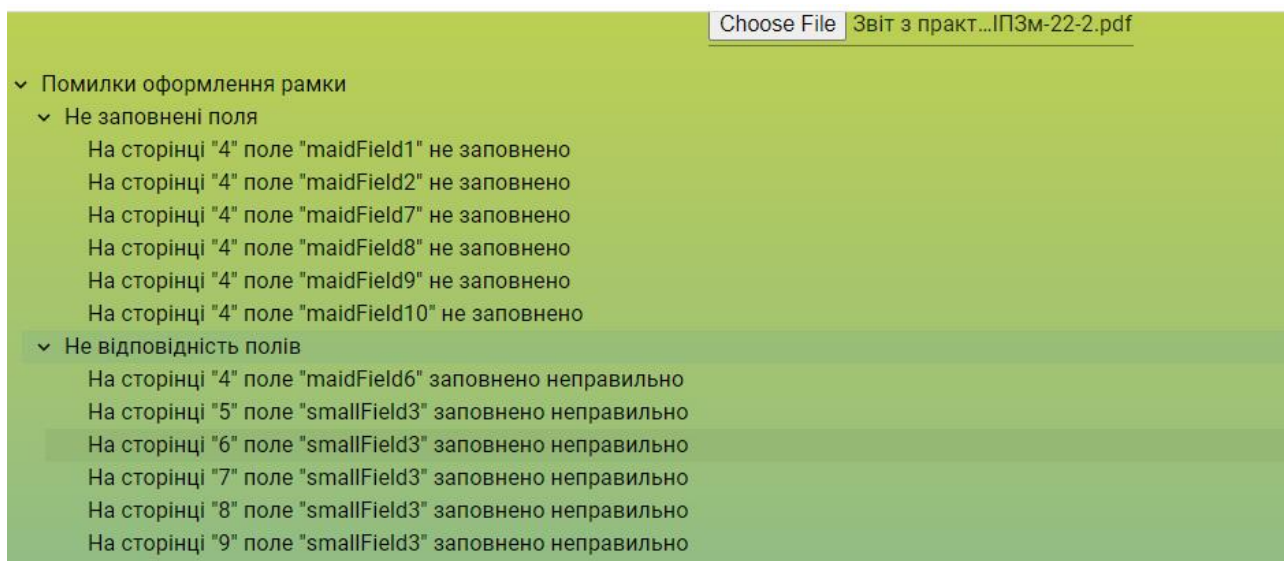


Рис. 3.2 Приклади помилок, які виводитиме сервіс з перевірки рамок

Функціонал мікросервісу з перевірки академічних робіт на правильність оформлення рисунків та підписів спрямований на забезпечення відповідності графічних елементів стандартам та вимогам оформлення.

Мікросервіс використовує алгоритми обробки графіки для виявлення рисунків та інших графічних елементів, аналізує їхнє розташування, розмір та формат, перевіряючи відповідність стандартам оформлення, що включає в себе перевірку розташування рисунків у тексті згідно вимог, тобто по центру.

Мікросервіс перевіряє правильність підписів до рисунків, аналізує наявність підписів, розташування та текст, переконуючись, що інформація, надана у підписах, відповідає стандартам та є чіткою та зрозумілою.

Якщо під час аналізу виявляються невідповідності чи помилки, мікросервіс генерує докладний звіт, який включає в себе інформацію про місця у документі, де виявлені проблеми з оформленням графічних елементів та підписів, а також надає рекомендації щодо їх виправлення (рис. 3.3).

```
[ERROR_TYPE.picturesAndTables]: {
  rule1: 'Помилка в нумерації! Нумерація рисунків та таблиць має бути арабськими цифрами без знака "№". ',
  rule2: 'Помилка в розташуванні! Рисунки й таблиці мають бути після тексту, де вони згадані вперше, або на наступній сторінці.',
  rule3: 'Помилка в розташуванні! Рисунки або таблиці, які займають більше 2/3 сторінки рекомендується виносити в додатки',
  rule4: 'Помилка в позначенні! Рисунки мають бути позначені словом «рис.» та нумеровані послідовно в межах розділу.', //за винятком рисунків, ви
  rule5: 'Помилка в нумерації! Номер рисунку складається з номера розділу та порядкового номера рисунку, між якими ставиться крапка', //напри
  rule6:
    'Помилка в підписі рисунку! Номер рисунку та його назву розміщують під самим рисунком без абзацного відступу по центру, курсив. Крапка вкін
  rule7: 'Розрив рисунка не допускається',
  rule8: 'Назва рисунка має бути з великої літери. ',
  rule9: 'Підпис таблиці має містити порядковий номер таблиці та заголовок',
  rule10:
    'Номер таблиці та його назва мають бути над таблицею: спочатку слово «Таблиця 1.1» (вирівнювання по правому краю рядка, курсив), а тоді наз
  rule11:
    'Над заголовком, праворуч з великої літери цілком пишуть слово «Таблиця» та проставляють її порядковий номер.',
  rule12: 'Номер таблиці має складатися з номера розділу та порядкового номера таблиці, розділених крапкою', //наприклад; «Таблиця 2.2»
  rule13: 'При посиланні на таблицю в тексті має бути табл. та її порядковий номер',
  rule14: 'Над продовженням таблиці на новій сторінці має бути "Продовження табл. 2.2". ',
  rule15:
    'Продовження таблиці займає більше 1 сторінки, перед останнім фрагментом має бути написано Закінчення табл. 2.2, заголовок не повторюють',
}
```

Рис. 3.3 Приклади помилок, які виводитиме сервіс з перевірки рисунків і таблиць

Завдання мікросервісу з перевірки аббревіатур включає аналіз текстового контенту документа та перевірку використання аббревіатур у відповідності до встановлених норм.

Початково мікросервіс ідентифікує всі можливі аббревіатури в тексті документа, що може включати аббревіатури наукових термінів, власних назв, стандартних скорочень та інших виразів, які часто використовуються у дипломних роботах. Для цього використовуються розширені алгоритми обробки тексту та мовної аналітики.

Після ідентифікації аббревіатур мікросервіс проводить перевірку правильності їхнього написання та використання відповідно до встановлених стандартів та правил, аналізує контекст, в якому використовуються аббревіатури, щоб визначити, чи вони використані вірно та, чи відповідають прийнятим граматичним та стилістичним нормам.

Додатково, мікросервіс може враховувати різні варіації аббревіатур та їхній контекст, щоб уникнути неправильного розуміння або використання термінів. Додаток може розрізняти аббревіатури, що мають декілька можливих значень, і враховувати контекст для правильного інтерпретування.

Якщо мікросервіс виявляє неправильне використання аббревіатур або їхні невідповідності вимогам, він створює докладний звіт, який вказує місця у тексті документа, де виявлені проблеми, та надає рекомендації щодо їх виправлення (рис. 3.4).

```
[ERROR_TYPE.mainPage]: {
  rule1: 'Пропущена фраза "МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ" або неправильне вирівнювання',
  rule2: 'Пропущена назва університету або неправильний регістр. Перевірте вирівнювання',
  rule3: 'Неправильний регістр або вирівнювання',
  rule4: 'Пропущена фраза "ПОЯСНЮВАЛЬНА ЗАПИСКА" або неправильний регістр',
  rule5: 'Пропущена фраза "до кваліфікаційної роботи освітнього ступеня «магістр» (або «бакалавр»)''',
  rule6: 'Пропущена назва спеціальності (немає фрази "за спеціальністю" ІІІ коду з трьох цифр)',
  rule7: 'Не вказана освітня програма (немає фрази "освітня програма", відсутні круглі дужки)',
  rule8: 'Не вказана фраза ("на тему:")',
  rule9: 'Неправильний регістр в назві роботи. Має бути написана великими літерами',
  rule10: 'Неправильний регістр, тема роботи пишеться з великої літери, далі маленькими',
  rule11: 'Неправильне вирівнювання, виконавець роботи, керівник та рецензент має вказуватися справа',
  rule12: 'Помилка в місті та році, місто має писатися з великої літери (далі маленькими). Перевірте, щоб після міста був пробіл, тире, пр
},
```

Рис. 3.4 Приклади помилок, які виводитиме сервіс з перевірки аббревіатур

3.2 Тестування та представлення результатів роботи сервісу

Розробка програмного забезпечення неможлива без ефективного тестування, яке гарантує стабільність та надійність сервісу.

Тестування програмного забезпечення – техніка контролю якості, що перевіряє відповідність між реальною й очікуваною поведінкою програми завдяки кінцевому набору тестів, які обираються певним чином [32].

Тестування ПЗ поділяється на декілька видів:

- інтеграційне тестування;

- системне тестування;
- навантажувальне тестування;
- юніт тестування.

Юніт тестування – це практика в розробці програмного забезпечення для перевірки того, що одиниця коду буде працювати як очікується, і що протестований код буде виробляти однаковий результат щоразу, коли тест виконується. Юніт тест повинен тестувати лише функціональність одного методу, також званого методом, що тестується, або системою, що тестується. Це легше сказати, ніж зробити, через різні залежності в методі та як вони контролюються. Такі залежності включають аргументи, використання результатів виклику інших методів, створення нових екземплярів класів та роботу з ними тощо. Поведінка цих залежностей повинна контролюватися, і вони повинні виробляти контрольні дані [33].

Інтеграційне тестування є типом тестування ПЗ, яке прагне перевірити інтерфейси між компонентами від програмного дизайну. Програмні компоненти можуть бути інтегровані як в рамках ітеративного підходу, так і всі разом. Інтеграційне тестування працює над виявленням дефектів у інтерфейсах та взаємодії інтегрованих компонентів (модулів). Воно проводиться до тих пір, поки великі групи тестованих компонентів ПЗ, які відповідають потрібній архітектурі, починають працювати, як система [34].

Системне тестування – це тип тестування програмного забезпечення, який виконує перевірку системи в цілому та передбачає інтеграцію всіх окремих модулів і компонентів програмного забезпечення, яке ви розробили, щоб перевірити, чи працює система разом, як очікувалося [35].

Навантажувальне тестування – форма тестування, яка визначає працездатність системи для того, щоб оцінити поведінку додатка (сайту) із заданим наперед очікуваним навантаженням [36].

Для тестування сервісів первинної перевірки дипломних робіт ми вирішили обрати саме юніт тестування в поєднанні з ручним тестуванням. Вибір

пов'язаний зі специфікою функціонала сервісів, адже кожний наш мікросервіс функціонує як окремий сервіс.

Для перевірки роботи мікросервісів було написано юніт тести, виконання яких продемонстровано на рисунку (рис. 3.5).

File	% Stmts	% Branch	% Funcs	% Lines	Un
All files	98.82	92.64	100	98.67	
config	100	100	100	100	
enums.ts	100	100	100	100	
config/constants	100	100	100	100	
content.ts	100	100	100	100	
frame.ts	100	100	100	100	
reference.ts	100	100	100	100	
services/checkAbbreviationList/src	100	100	100	100	
checkAbbreviationListStructure.ts	100	100	100	100	
defineAbbr.ts	100	100	100	100	
utils/pdfFormatters/helpers	100	100	100	100	
removePageNumberRow.ts	100	100	100	100	
Test Suites: 41 passed, 41 total Tests: 145 passed, 145 total Snapshots: 0 total Time: 32.934 s Ran all test suites.					

Рис. 3.5 Юніт тестування сервісу

Окрім автоматизованого тестування, було також проведено ручне для досягнення точніших результатів. Багів знайдено не було, адже всі недоліки були виправлені при автоматизованому тестуванні.

Спочатку було протестовано мікросервіс перевірки рамок (рис. 3.6)

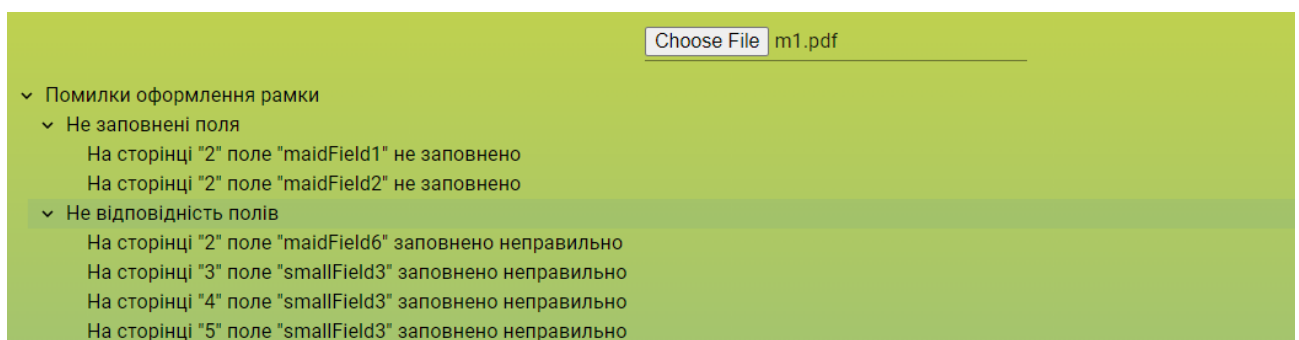


Рис. 3.6 Тестування мікросервісу перевірки рамок

Порівняємо результати перевірки мікросервісу з PDF файлом дипломної роботи (рис. 3.7).

					ДУ «Житомирська політехніка».			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.					Переддипломна практика	Літ.	Арк.	Аркушів
Перевір.							2	22
Керівник	Гришкун Є.С.					ФІКТ Гр. ІПЗм-22-2		
Н. контр.								
Зав. каф.								

Рис. 3.7 Заповнення полів рамки

Як бачимо, поле 1 та 2 в рамці на другій сторінці дійсно не заповнені, а поле 6 заповнене неправильно – має бути шифр спеціальності.

3.3 Оцінка ефективності впровадження сервісу

Інновації відіграють важливу роль у науковій діяльності вищих навчальних закладів. Розвиток нових технологій, таких як: штучний інтелект, машинне навчання та хмарні обчислення, відкриває нові можливості для наукових досліджень та дозволяє науковцям швидше збирати та аналізувати дані, а також проводити складні експерименти.

Цифрова трансформація вищої освіти призвела до появи нових форм і методів навчання, таких як: дистанційне навчання, віртуальні лабораторії та інтерактивні навчальні модулі. Дані технології дозволяють студентам здобувати освіту незалежно від місця розташування та часу доби, а також підвищують ефективність навчання.

Спрямовані на сприяння навчальному процесу, науковці впроваджують різноманітні інструменти та платформи, що допомагають викладачам та студентам. У цьому контексті було розроблено інноваційний сервіс, спрямований на покращення оформлення дипломних робіт.

Розроблений сервіс має на меті автоматизувати процес перевірки правильності оформлення дипломних робіт. Основні аспекти, на які зосереджено

увагу, включають в себе перевірку відповідності рамок, використання аббревіатур та належних підписів до рисунків і таблиць, що особливо актуально в умовах сучасного онлайн навчання, де студенти та викладачі часто взаємодіють через віртуальні платформи.

Спеціалізований алгоритм перевірки визначає відхилення від стандартів та надає детальний звіт задля зручності виправлення. Студенти отримують можливість швидше та ефективніше коригувати свої роботи, враховуючи рекомендації системи.

Однією з ключових переваг сервісу є його спроможність ефективно економити час як для викладачів, так і для студентів. Викладачам не доведеться витрачати безліч годин на ручну перевірку оформлення кожної роботи, оскільки весь процес стає автоматизованим. Враховуючи кількість випускників у наукового керівника, викладачу доводиться витрачати чимало часу на перевірку дипломних робіт.

З метою порівняння ручної та автоматизованої перевірки дипломних робіт було зроблено таблицю (табл. 3.1) [2].

Таблиця 3.1

Порівняння ручної та автоматизованої перевірки дипломних робіт

Параметри перевірки	Ручна перевірка	Автоматизована перевірка
Час	Від 2 до 23 годин на одну роботу бакалавра, від 3 до 5 годин на одну роботу магістра.	До 1 хвилини на одну роботу.
Ефективність	Невисока, вимагає багато часу та зусиль.	Дуже висока, дозволяє перевірити велику кількість робіт за короткий час.
Обсяг робіт	Будь-який обсяг робіт в день.	Не більше 3 робіт ступеня магістр або бакалавр за день.
Об'єктивність	Залежить від досвіду, уважності, ступеня виснаженості та втомленості викладача, наявності часу.	Більш об'єктивна, оскільки заснована на алгоритмах.
Переваги	можливість враховувати індивідуальні вимоги	швидкий процес перевірки, оскільки

Продовження таблиці 3.1

	конкретного закладу вищої освіти; перевірка всіх аспектів дипломної роботи включаючи стилістику, лексику та орфографію.	сервіс може перевірити велику кількість дипломних робіт за короткий час на основі чітко визначених правил.
Недоліки	тривалий процес перевірки, оскільки керівник має перевірити кожен дипломну роботу окремо. Можливість пропуску помилок при оформленні, через втому або виснаження.	вірогідність пропуску помилок, які не передбачені правилами перевірки сервісу. Неможливість врахування індивідуальних вимог до оформлення дипломної роботи.

Відносини, які формуються з приводу економічного та раціонального використання робочого часу в процесі виробництва матеріальних благ, регулюються об'єктивним загальним економічним законом економії часу. Сутність закону економії часу виявляється в його якісній характеристиці, а також у визначенні кількісних меж економії часу [37].

Якісна характеристика закону полягає в тому, що внаслідок трудової діяльності між працюючими складаються такі причинно-наслідкові зв'язки, які відображають скорочення витрат часу відносно результатів праці за рахунок раціонального та економного використання робочого часу.

Максимальною межею економії часу є повне використання всього фонду часу та сукупної праці при досягнутому в суспільстві максимальному рівні продуктивності праці. Мінімальною межею закону економії часу є фактично досягнутий рівень використання сукупної праці та фонду часу. Форми прояву закону економії часу різні, оскільки пов'язані з різноманітністю життєдіяльності людей.

Основою економного та раціонального використання часу у сфері освіти є використання кожної робочої хвилини та підвищення продуктивності праці.

Навантаження викладача закладу вищої освіти – це обсяг роботи у годинах за визначеними законом видами діяльності (навчальна, методична, наукова, організаційна), який дорівнює встановленій кількості годин.

Відповідно пункту 1 статті 56 закону України «Про вищу освіту» від 01.07.2014 № 1556-VII робочий час науково-педагогічних, наукових і педагогічних працівників робочий час науково-педагогічних працівників становить 36 годин на тиждень (скорочена тривалість робочого часу) [38].

Пунктом 2 вищезазначеного закону [38] передбачено, що робочий час науково-педагогічного працівника включає час виконання ним навчальної, методичної, наукової, організаційної роботи та інших трудових обов'язків. Робочий час наукового працівника включає час виконання ним наукової, дослідницької, консультативної, експертної, організаційної роботи та інших трудових обов'язків.

Поряд з тим, відповідно до закону України «Про вищу освіту» від 01.07.2014 № 1556-VII [38] максимальне навчальне навантаження на одну ставку науково-педагогічного працівника не може перевищувати 600 годин на навчальний рік. При цьому, норми часу навчальної, методичної, наукової, організаційної роботи визначаються закладом вищої освіти.

Для визначення ефективності впровадження сервісу проведено аналіз витрат часу при плануванні навчальної роботи науково-педагогічних працівників на керівництво, консультування здобувача першого (бакалаврського) та другого (магістерського) рівнів вищої освіти щодо виконання кваліфікаційної (дипломної) роботи. Для аналізу використано дані з вищих навчальних закладів (рис. 3.8).

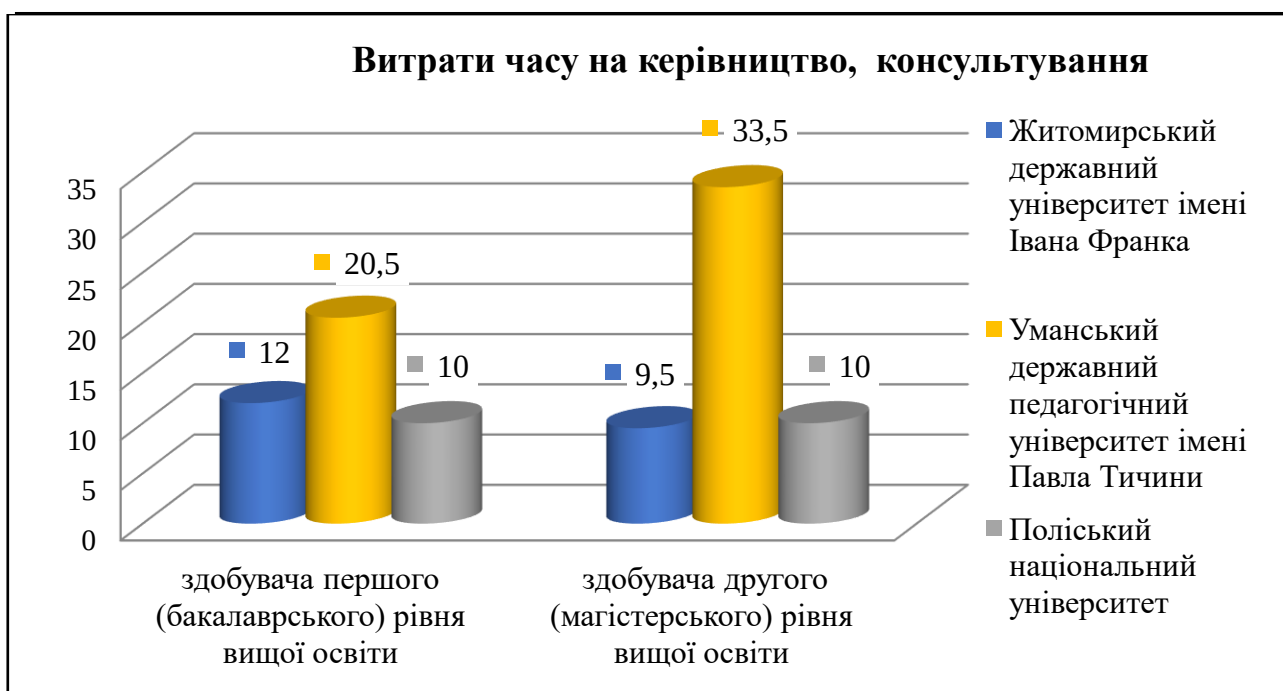


Рис. 3.8 Аналіз витрат часу при плануванні навчальної роботи науково-педагогічних працівників на керівництво, консультування здобувача першого (бакалаврського) та другого (магістерського) рівнів вищої освіти щодо виконання кваліфікаційної (дипломної) роботи, год

Керівництво, консультування здобувача першого (бакалаврського) рівня вищої освіти щодо виконання кваліфікаційної (дипломної) роботи у різних закладах вищої освіти займає від 10 до 20,5 годин (див. рис. 3.8) [39-41].

Поряд з тим, керівництво, консультування здобувача другого (магістерського) рівня вищої освіти щодо виконання кваліфікаційної (дипломної) роботи у різних закладах вищої освіти займає від 9,5 до 33,5 годин [39-41]. Слід зазначити, що в Поліському університеті на керівництво кваліфікаційними роботами передбачено 12 годин, в тому числі 10 годин на керівництво та 2 години на рецензування [39].

При використанні сервісу первинної перевірки зазначених робіт щоразу викладач витратить на 1 годину менше. Таким чином, витрати часу мають тенденцію до зменшення, що дасть можливість **скоротити витрати часу та перерозподілити його на більш необхідні завдання.**

Якщо припустити, що викладач проводить керівництво та консультування одночасно у 5 здобувачів другого (магістерського) рівня вищої освіти та

8 здобувачів першого (бакалаврського) рівня вищої освіти, то навантаження викладача закладу вищої освіти, при використанні сервісу, буде зменшене на 13 годин.

Крім того, якщо припустити, що в університеті може навчатися від 300 до 700 здобувачів освіти, то при застосуванні сервісу економія часу для викладачів, які консультують здобувачів першого (бакалаврського) рівня вищої освіти становитиме від 300 годин до 700 годин відповідно, що зображено в таблиці (табл. 3.2).

Таблиця 3.2

Аналіз витрат часу на керівництво, консультування здобувачів першого (бакалаврського) рівня вищої освіти

<i>без застосування сервісу</i>			
Кількість здобувачів, чол	300	500	700
Кількість годин, год	4260	7100	9940
<i>при застосуванні сервісу</i>			
Кількість здобувачів, чол	300	500	700
Кількість годин, год	3960	6600	9240
різниця (економія), год	300	500	700

* - при обрахунку кількості годин використано середнє значення витрат часу на керівництво, консультування здобувачів першого (бакалаврського) рівня вищої освіти трьох університетів (14,2 год).

Аналогічна ситуація спостерігається і для викладачів, які консультують здобувачів другого (магістерського) рівня вищої освіти та становитиме від 300 годин до 700 годин відповідно, що зображено в таблиці (табл. 3.3).

Таблиця 3.3

**Аналіз витрат часу на керівництво, консультування здобувачів другого
(магістерського) рівня вищої освіти**

<i>без застосування сервісу</i>			
Кількість здобувачів, чол	300	500	700
Кількість годин, год	5310	8850	12390
<i>при застосуванні сервісу</i>			
Кількість здобувачів, чол	300	500	700
Кількість годин, год	5010	8350	11690
різниця (економія), год	300	500	700

** - при обрахунку кількості годин використано середнє значення витрат часу на керівництво, консультування здобувачів другого (магістерського) рівня вищої освіти трьох університетів (17,7 год).*

Враховуючи вищевикладене, встановлено, що під час використання сервісу виникає економія часу, яка полягає у прискоренні проходження первинної перевірки створеної здобувачем освіти кваліфікаційної роботи відповідно до вимог методичних рекомендацій до виконання дипломної роботи. При цьому, скорочуються витрати часу на перевірку всіх робіт та підвищується ефективність роботи викладача.

Таким чином, застосування сервісу первинної перевірки кваліфікаційних робіт може значно полегшити та покращити роботу викладача та може допомогти студентам уникнути помилок при оформленні їхніх робіт.

Висновки до 3 розділу

В третьому розділі було описано використання та рекомендації щодо налаштування сервісу, а саме: послідовність встановлення сервісу та залежностей, налаштування відображення результатів перевірки та налаштування безпосередньо перевірки. Описано тестування сервісу та представлення результатів роботи сервісу.

Сервіс має стати універсальним швидким рішенням для перевірки робіт студентів. Впровадження сервісу в кабінетах студентів автоматично надасть сервісу всі потрібні вхідні дані для його налаштування, а саме дані про тип роботи яка перевіряється, групу студента та інші. На основі даних студента можуть бути автоматично сформовані дані для перевірки, наприклад для заповнення рамки буде згенеровано шифр роботи.

Було проведено оцінку ефективності впровадження сервісу на основі декількох університетів. Встановлено, що під час використання сервісу виникає економія часу, яка полягає у прискоренні проходження первинної перевірки створеної здобувачем освіти кваліфікаційної роботи відповідно до вимог методичних рекомендацій до виконання дипломної роботи. При цьому, скорочуються витрати часу на перевірку всіх робіт та підвищується ефективність роботи викладача.

Таким чином, застосування сервісу первинної перевірки кваліфікаційних робіт може значно полегшити та покращити роботу викладача та може допомогти студентам уникнути помилок при оформленні їхніх робіт.

ВИСНОВКИ

У магістерській роботі включено процес розробки сервісів первинної перевірки пояснювальних записок до дипломних робіт.

З метою розробки застосунку було проаналізовано і визначено елементи оформлення дипломних робіт, які підлягають автоматизованій перевірці, проведено аналіз аналогів та обрано програмні засоби реалізації. Враховуючи роботу з PDF файлами, було проаналізовано та проведено огляд парсингу, а також обрано та обґрунтовано вибір програмних засобів та реалізацій. На основі спостережень, аналізу, досліджень і ознайомлення з вимогами до оформлення було реалізовано мікросервіси перевірки рамок, аббревіатур, підписів до рисунків і таблиць і додатків. На останньому етапі розробки було проведено ручне та юніт тестування, описано роботу інноваційних сервісів первинної перевірки дипломних робіт на правильність оформлення, надано рекомендації з встановлення та налаштування та проведено оцінку ефективності впровадження сервісу.

В результаті виконання магістерського завдання було досягнуто поставленої мети та розроблено сервіс первинної перевірки дипломних робіт на правильність оформлення згідно вимог. На даний момент сервіс здатний перевіряти дипломні роботи у форматі PDF на правильність оформлення підписів до рисунків і таблиць, правильне оформлення аббревіатур і додатків, а також наявність та заповнення рамок в роботах ступеня бакалавр згідно вимог університету, а саме факультету ФІКТ.

Інноваційний сервіс пропонується до використання студентами та науковими керівниками з метою перевірки академічних праць перед захистом. Практична цінність полягає в наданні користувачам зручного та ефективного інструменту для перевірки академічних робіт на відповідність академічним стандартам, що підвищить якість академічних робіт на етапі написання студентами, зменшить навантаження на наукових керівників та перевірку норм контролю робіт.

В майбутньому планується покращити сервіс та додати більше функціоналу, який значно скоротить час та якість перевірки. До прикладу, можна накопичувати, зберігати та аналізувати статистику помилок з дипломних робіт для вдосконалення сервісу. Крім того, можна додати побудову ІШ для розпізнавання орфографічних та стилістичних помилок на базі статистики.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Желізко В. В., Кучерук О.В., Савіцький Р.С. Парсинг PDF файлів // Теоретичні та практичні дослідження молодих вчених : тези доповідей Міжнародної науково-практичної конференції, Online, 28 – 30 листопада 2023 / МОН України, Харківський політехнічний інститут, 2023. – С. 360.
2. Кучерук О. В., Желізко В.В., Савіцький Р.С. Система перевірки дипломних робіт як інструмент автоматизації роботи студента та викладача // редакційний журнал «Житомирська політехніка»(подано до друку) – Житомир.
3. Гладун А. Я. Автоматизована система [Електронний ресурс] / А. Я. Гладун – Режим доступу до ресурсу: <https://vue.gov.ua/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B0%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0>.
4. Grammarly: Free Writing AI Assistance [Електронний ресурс] – Режим доступу до ресурсу: <https://app.grammarly.com/>.
5. Hemingway Editor [Електронний ресурс] – Режим доступу до ресурсу: <https://hemingwayapp.com/>.
6. TypeScript for JavaScript Programmers [Електронний ресурс] – Режим доступу до ресурсу: <https://www.typescriptlang.org/docs/handbook/typescript-in-5-minutes.html>.
7. Парсинг даних з сайтів: що це та на чиєму боці закон.. [Електронний ресурс] – Режим доступу до ресурсу: <https://highload.today/uk/parsing/>.
8. Кащенко Д.О. Дослідження характеристик веб-застосунків ВНЗ за допомогою робота-парсера: автореф. маг. н. р. на здобуття освіт. кваліфікації «Маг. сист. ан.»: 004.89. Миколаїв, 2020. 83 с.
9. Вакулєнко Ю.О. Впровадження та перспективи використання програм-парсерів в закладах вищої освіти [Електронний ресурс] / Юлія Олегівна

Вакуленко. – 2021. – Режим доступу до ресурсу:
<https://jarch.donnu.edu.ua/article/view/11534/11415>.

10. PDF Parser [Електронний ресурс]. – 2021. – Режим доступу до ресурсу:
<https://medium.com/nanonets/pdf-parser-aa6555c324ad>

11. A full comparison of 6 JS libraries for generating PDFs [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: <https://dev.to/handdot/generate-a-pdf-in-js-summary-and-comparison-of-libraries-3k0p>.

12. Documentation pdfjs-dist [Електронний ресурс] – Режим доступу до ресурсу: <https://www.npmjs.com/package/pdfjs-dist>.

13. Тверитникова О. Є., Васильченков О.Г., Крилова В.А. Базові алгоритми та основи програмування. Теорія і практика. Навчальний посібник для студентів спеціальностей «Автоматизація та комп'ютерно-інтегровані технології», «Метрологія та вимірювальна техніка» усіх форм навчання вищих навчальних закладів. – Харків, 2020. –264 с.

14. Крєневич А. П. Електронний підручник закладу вищої освіти // Алгоритми і структури даних. Підручник. – К.: ВПЦ "Київський Університет", 2021. – 200 с.

15. Hlukhov V. S., Sydorko D. S. «Algorithmes and software for verification of scientific and technical text documents». Applied Aspects of Information Technology. 2023; Vol. 6 No. 3: с. 317.

16. Парнус, К., Свісюк, О. (2022). The algorithm of analything English texts with the help of parsers. SWorldJournal, 1(14-01), 108–113.

17. Тверитникова О. Є., Васильченков О.Г., Крилова В.А. Базові алгоритми та основи програмування. Теорія і практика. Навчальний посібник для студентів спеціальностей «Автоматизація та комп'ютерно-інтегровані технології», «Метрологія та вимірювальна техніка» усіх форм навчання вищих навчальних закладів. – Харків, 2020. –264 с.

18. Основні міри подібності та нові підходи до їх застосування при класифікуванні гіперспектральних космічних зображень / С.І. Альперт //

Математичні машини і системи. — 2019. — № 1. — С. 143–151. — Бібліогр.: 22 назв.

19. Джинджиристий А. З. Математичне та програмне забезпечення автоматизованої системи підбору команди розробників [Електронний ресурс] / А. З. Джинджиристий, М. І. Паламар. – Тернопіль, 2019. – 47 с. – Режим доступу до ресурсу:

https://elartu.tntu.edu.ua/bitstream/lib/29853/5/Magister_work_Dzyndzyrystyi.pdf.

20. Климчук В. О. Кластерний аналіз: використання у психологічних дослідженнях / В. О. Климчук. – Житомир. – 6 с. – (Житомирського державного університету імені Івана Франка).

21. Cohen, David (2004), Precalculus: A Problems-Oriented Approach (6th ed.), Cengage Learning, p. 698.

22. Методи та засоби експериментальних досліджень : навч. посіб. / Г.Б. Параска, Д.В. Прибега, П.С. Майдан. – Київ : Кондор-Видавництво, 2017. – 138 с.

23. Вимоги до програмного забезпечення [Електронний ресурс] – Режим доступу до ресурсу: <https://studfile.net/preview/5130988/page:2/>.

24. Козак О.Л. Опорний конспект лекцій з курсу —Аналіз вимог до програмного забезпечення для студентів напрямку підготовки — Програмна інженерія [Електронний ресурс] / О.Л. Козак. – Тернопіль, 2011. – 56 с. – Режим доступу до ресурсу:

http://dspace.wunu.edu.ua/retrieve/14135/FCIT_kKN_sPZS_dAVPZ_%20LEC.pdf

25. Специфікація вимог до програмного забезпечення: Методичні вказівки до лабораторних робіт № 1, 2 з дисципліни «Аналіз вимог до програмного забезпечення» для студентів базового напрямку 6.050103 «Програмна інженерія» / Укл.: О.Є. Білас, Н. Б. Мельник – Львів: Видавництво Національного університету «Львівська політехніка», 2015. – 16 с.

26. Випускна кваліфікаційна робота бакалавра [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: <https://learn.ztu.edu.ua/course/view.php?id=4901>.

27. ДСТУ ГОСТ 2.610:2006 - Єдина система конструкторської документації. Основні написи – Україна, 2006.
28. Гаврилюк О. О. Аббревіатури у сфері віртуальної комунікації: прагматичний аспект (на матеріалі англійської мови) – Київ: Видавництво Київського національного університету імені Тараса Шевченка, 2018. – №4 (23). – 50 с.
29. Abbreviation: Definition and Examples [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: <https://www.grammarly.com/blog/abbreviations/#:~:text=Typically%2C%20acronyms%20and%20initialisms%20are,they%20entail%20a%20proper%20noun.>
30. Правила оформлення списку скорочень у дипломній роботі [Електронний ресурс] – Режим доступу до ресурсу: <https://diploms.com.ua/ua/pravila-oformlennya-spisku-skorochen-u-diplomnij-roboti/>.
31. Випускна кваліфікаційна робота(ІІІ+УІІІ). Рівень: магістр [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: <https://learn.ztu.edu.ua/course/view.php?id=4902>.
32. Конспект лекцій з навчальної дисципліни «Методологія тестування програмного забезпечення» [Електронний ресурс] / Укладач Старух А.І. – Львів: Львівський національний університет імені Івана Франка, 2020. – с.3 – Режим доступу до ресурсу: <https://financial.lnu.edu.ua/wp-content/uploads/2019/09/konspekt-testuvannia.pdf>.
33. Unit testing: the complete guide. – вид-во Progress – 2020. – 24 с.
34. Конспект лекцій з навчальної дисципліни «Методологія тестування програмного забезпечення» / Укладач Старух А.І. – Львів: Львівський національний університет імені Івана Франка, 2020. – с.3.
35. Що таке системне тестування? Глибоке занурення в підходи, типи, інструменти, поради та підказки і багато іншого! [Електронний ресурс] – Режим доступу до ресурсу: <https://www.zaptest.com/uk/%D1%89%D0%BE-%D1%82%D0%B0%D0%BA%D0%B5-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%BD%D0%B5->

[%D1%82%D0%B5%D1%81%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F-](#)

[%D0%B3%D0%BB%D0%B8%D0%B1%D0%BE%D0%BA%D0%B5-%D0%B7.](#)

36. Дякович Т. Навантажувальне тестування для перевірки стабільності роботи WEB-ресурсу // Магістр. Тернопіль : Тернопільський національний педагогічний університет ім. В. Гнатюка, 2015. Вип. 23. С. 91-93.

37. Економічна енциклопедія: У трьох томах. Т. 1. / Редкол.: ...С. В. Мочерний (відп. ред.) та ін. — К.: Видавничий центр «Академія», 2000. — 864 с.

38. Закон України «Про вищу освіту» від 01.07.2014 № 1556-VII : Закон України від 01.07.2014 р. № № 1556-VII.

39. Положення про планування та облік роботи науково-педагогічних працівників, затверджене рішенням Вченої ради Поліського національного університету: ст. 11 Дод. А від 28.08.2021 р. протоколом №1.

40. Вчена рада Житомирського державного університету імені Івана Франка. Тимчасового положення про планування, норми, облік робочого часу й рейтингування науково-педагогічних працівників за результатами роботи в Житомирському державному університеті імені Івана Франка / Вчена рада Житомирського державного університету імені Івана Франка. – 2021. – С. протокол №16 : ст. 17 Дод. А від 30.08.2021 р. №16.

41. Положення про планування та облік основних видів роботи науково-педагогічних працівників Уманського державного педагогічного університету імені Павла Тичини : ст. 11 Дод. 1 від 25.05.2021 р. № 17 [Електронний ресурс] –

Режим доступу до ресурсу:

<https://udpu.edu.ua/documents/doc/%D0%94%D0%BE%D0%BA%D1%83%D0%B%D0%B5%D0%BD%D1%82%D0%B8%20%D0%B7%20%D0%BE%D1%80%D0%B3%D0%B0%D0%BD%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%97%20%D0%BE%D1%81%D0%B2%D1%96%D1%82%D0%BD%D1%8C%D0%BE%D0%B3%D0%BE%20%D0%BF%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%83/%D0%9D%D0%B0%D0%B2%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE->

[%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%B8%D1%87%D0%BD%D0%B0%20%D0%B4%D0%BE%D0%BA%D1%83%D0%BC%D0%B5%D0%BD%D1%82%D0%B0%D1%86%D1%96%D1%8F/%D0%9F%D0%BE%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F%20%D0%BF%D1%80%D0%BE%20%D0%BF%D0%BB%D0%B0%D0%BD%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F%20%D1%82%D0%B0%20%D0%BE%D0%B1%D0%B%D1%96%D0%BA%20%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%BD%D0%B8%D1%85%20%D0%B2%D0%B8%D0%B4%D1%96%D0%B2%20%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B8%20%D0%9D%D0%9F%D0%9F.pdf?ver=1623303251](#)