

ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та
інформаційних технологій
Кафедра інформаційних систем та технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавр

на тему: **«Розроблення системи інтелектуальної підтримки створення
силабусів освітніх компонентів шляхом агрегації даних та використання API
моделей штучного інтелекту»**

Виконала: здобувачка вищої освіти
за освітньою програмою
Інформаційні управляючі системи
спеціальності 126 Інформаційні системи та
технології ступеня вищої освіти бакалавр
групи 126ІСТ_бд_2022
Горда Віталіна Віталіївна
Керівник: Флегантов Леонід Олексійович
Рецензент: Брикун Олександр Миколайович

Полтава – 2026 року

ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та
інформаційних технологій
Кафедра інформаційних систем та технологій

Освітня програма Інформаційні управляючі системи
Спеціальність 126 Інформаційні системи та технології
Рівень вищої освіти перший (бакалаврський)

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Юрій УТКІН

«10» червня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Горди Віталіни Віталіївни

1. Тема кваліфікаційної роботи:

«Розроблення системи інтелектуальної підтримки створення силабусів освітніх компонентів шляхом агрегації даних та використання API моделей штучного інтелекту»

Керівник роботи: к.ф.-м.н., доцент, професор кафедри інформаційних систем та технологій Флегантов Леонід Олексійович.

Затверджено наказом закладу вищої освіти від «10» квітня 2026 року № 383-ст

2. Строк подання роботи здобувачем вищої освіти «08» червня 2026 року.

3. Вихідні дані до роботи: нормативно-правові акти України у сфері вищої освіти, освітньо-професійні програми (ОПП) та методичні рекомендації закладів вищої освіти, офіційна технічна документація прикладних програмних інтерфейсів (API) великих мовних моделей, специфікації та документація технологічного стеку розробки, наукові публікації вітчизняних і зарубіжних вчених у галузі штучного інтелекту та автоматизації текстових процесів.

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

Розділ 1. Аналіз предметної області та постановка задачі автоматизованого створення силабусів

Розділ 2. Проектування системи інтелектуальної підтримки створення силабусів

Розділ 3. Реалізація та дослідження роботи програмного забезпечення

5. Перелік графічного матеріалу: схеми, рисунки, діаграми за темою та об'єктом дослідження.

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання отримав
Оцінювання економічної ефективності результатів дослідження	Дивнич О. Д., к. е. н., доцент, доцент кафедри економіки та публічного управління	01.04.2026 р.	29.05.2026 р.

7. Дата видачі завдання «10» червня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1.	Вибір і затвердження теми роботи	02.06.2025 р. – 04.06.2025 р.	
2.	Складання і затвердження розгорнутого плану та завдання на кваліфікаційну роботу	05.06.2025 р. – 10.06.2025 р.	
3.	Опрацювання джерел інформації	11.06.2025 р. – 15.06.2025 р.	
4.	Збір, вивчення і обробка інформації, необхідної для виконання роботи	16.06.2025 р. – 20.06.2025 р.	
5.	Виконання теоретико-методологічного розділу роботи	08.09.2025 р. – 01.11.2025 р.	
6.	Виконання дослідницько-аналітичного розділу роботи	19.01.2026 р. – 14.02.2026 р.	
7.	Виконання проектно-рекомендаційного розділу роботи	16.02.2026 р. – 31.03.2026 р.	
8.	Оцінювання економічної ефективності результатів дослідження	01.04.2026 р. – 29.05.2026 р.	
9.	Оформлення тексту роботи	01.06.2026 р. – 06.06.2026 р.	
10.	Попередній захист роботи на кафедрі	08.06.2026 р.	
11.	Доопрацювання роботи з урахуванням зауважень і пропозицій	09.06.2026 р. – 10.06.2026 р.	
12.	Нормоконтроль	11.06.2026 р. – 15.06.2026 р.	
13.	Захист кваліфікаційної роботи	16.06.2026 р. - 18.06.2026 р.	

Здобувач вищої освіти

Віталіна ГОРДА

Керівник роботи

Леонід ФЛЕГАНТОВ

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК	6
ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗОВАНОГО СТВОРЕННЯ СИЛАБУСІВ.....	11
1.1 Силабус освітнього компонента як інформаційний документ у межах освітньо-професійної програми.....	11
1.2 Освітньо-професійна програма як джерело вхідних даних.....	12
1.3 Формалізація вимог методичних рекомендацій до розроблення силабусів ..	15
1.4 Аналіз підходів до автоматизації створення силабусів	18
1.5. Обґрунтування задачі розроблення системи інтелектуальної підтримки створення силабусів.....	20
РОЗДІЛ 2 ПРОЄКТУВАННЯ СИСТЕМИ ІНТЕЛЕКТУАЛЬНОЇ ПІДТРИМКИ СТВОРЕННЯ СИЛАБУСІВ	24
2.1 Формалізація вимог до інформаційної системи.....	24
2.2 Загальна архітектура системи.....	25
2.3 Модель оброблення вхідних документів.....	29
2.4 Агрегація та узгодження даних з різних джерел	32
2.5 Інтеграція API штучного інтелекту.....	34
2.6 Алгоритм автоматизованого формування силабусу	36
РОЗДІЛ 3 РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ РОБОТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	38
3.1 Обґрунтування вибору програмних засобів.....	38
3.2 Реалізація модуля завантаження та аналізу PDF-документів.....	40
3.3 Реалізація модуля взаємодії з API штучного інтелекту	42
3.4 Формування та експорт силабусу освітнього компонента	45
3.5 Тестування та аналіз результатів роботи системи.....	47
3.6 Оцінка економічної ефективності впровадження системи.....	49
ВИСНОВКИ.....	56
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	58

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

- API – Application Programming Interface (програмний інтерфейс застосунку)
- HTML – HyperText Markup Language (мова розмітки гіпертексту)
- HTTP – HyperText Transfer Protocol (протокол передавання гіпертексту)
- JSON – JavaScript Object Notation (формат обміну даними)
- LLM – Large Language Model (велика мовна модель)
- LMS – Learning Management System (система управління навчанням)
- MIME – Multipurpose Internet Mail Extensions (багатоцільові розширення інтернет-пошти)
- NLP – Natural Language Processing (обробка природної мови)
- PDF – Portable Document Format (формат портативних документів)
- ROI – Return on Investment (індекс рентабельності інвестицій)
- UMS – University Management System (система управління університетом)
- URL – Uniform Resource Locator (уніфікований вказівник ресурсу)
- UTF-8 – Unicode Transformation Format (формат перетворення Юнікоду)
- XML – Extensible Markup Language (розширювана мова розмітки)
- ДСТУ – Державний стандарт України
- ЄКТС – Європейська кредитна трансферно-накопичувальна система
- ЗВО – Заклад вищої освіти
- ЗК – Загальна компетентність
- КС – Спеціальна (фахова) компетентність
- МОНУ – Міністерство освіти і науки України
- НАЗЯВО – Національне агентство із забезпечення якості вищої освіти
- ОК – Освітній компонент
- ОПП – Освітньо-професійна програма
- ПДАУ – Полтавський державний аграрний університет
- ПР – Програмний результат навчання
- РПНД – Робоча програма навчальної дисципліни
- ШІ – Штучний інтелект

ВСТУП

Актуальність теми. У контексті цифровізації вищої освіти особлива увага приділяється якості навчально-методичного забезпечення, базовим елементом якого є силабус освітнього компонента (ОК).

Відповідно до термінологічного словника Верховної Ради України, силабус визначається як документ, що містить основну інформацію про ОК у стислій формі (зокрема тематику, види завдань, критерії оцінювання тощо), а його наявність, форма та зміст регламентуються безпосередньо закладом вищої освіти [1].

На практиці, користуючись цією автономією, заклади вищої освіти формують силабуси як комплексні нормативно-інформаційні масиви, що не лише закріплюють компетентності й програмні результати навчання, а й деталізують тематичні плани, погодинний розподіл, багатокритеріальні матриці оцінювання (лабораторні роботи, тести, самостійну й контрольну діяльність), регуляторні політики курсу, зокрема, щодо академічної доброчесності, термінів виконання завдань та ін.

Традиційне проектування таких багатокомпонентних документів здійснюється науково-педагогічними працівниками переважно вручну, що зумовлює значні часові витрати, ускладнює уніфікацію та підвищує ризик виникнення неузгодженостей із освітніми програмами. Наявна суперечність зумовлює актуальність автоматизації процесу формування силабусів ОК шляхом розроблення рішень на основі великих мовних моделей (LLM) та їхніх прикладних програмних інтерфейсів (API), які здатні забезпечити інтелектуальну обробку, агрегацію даних та швидку генерацію структурованої документації.

Аналіз стану дослідження проблеми. На сьогодні існує значна кількість наукових праць і практичних розробок, присвячених цифровізації освіти, автоматизації обігу освітньої документації та інтеграції штучного інтелекту (ШІ) в інформаційні системи. У сучасних дослідженнях активно розглядаються питання впровадження систем управління навчанням (LMS), генерації дидактичного

контенту, інтелектуального оброблення текстів та застосування великих мовних моделей для підтримки викладацької діяльності.

Водночас, питання автоматизованого проектування силабусів ОК із забезпеченням їх наскрізної відповідності вимогам освітніх програм, локальних методичних рекомендацій ЗВО та національних стандартів вищої освіти залишається недостатньо дослідженим. Наявні програмні рішення переважно не забезпечують комплексної агрегації даних із різномірних джерел і не адаптовані до динамічних локальних нормативних вимог. Це обґрунтовує доцільність розроблення спеціалізованої інформаційної системи інтелектуальної підтримки та автоматизованого формування силабусів навчальних дисциплін.

Мета роботи: розроблення та дослідження інформаційної системи інтелектуальної підтримки формування силабусів ОК на основі агрегації даних із різномірних джерел та використання API моделей ШІ.

Завдання роботи:

- проаналізувати нормативні вимоги до структури та змісту силабусів ОК;
- дослідити освітні програми та методичні рекомендації як джерела вхідних даних для формування силабусів;
- проаналізувати існуючі підходи до автоматизації створення документів та використання API моделей штучного інтелекту;
- сформулювати технічні та функціональні вимоги до інформаційної системи;
- спроектувати та реалізувати програмний прототип системи мовою Python із інтеграцією API великих мовних моделей;
- оцінити ефективність розробленого рішення та рівень оптимізації часових витрат.

Об'єкт дослідження: процес автоматизованого формування навчально-методичної документації у закладах вищої освіти.

Предмет дослідження: архітектура та програмні засоби інтелектуальної підтримки створення силабусів із використанням API великих мовних моделей.

Методи наукових досліджень: аналіз наукових і нормативних джерел – для дослідження теоретичних основ створення силабусів; системний аналіз – для формалізації вимог і проєктування архітектури системи; методи оброблення текстових даних – для аналізу PDF-документів; програмування – для реалізації програмного прототипу; експериментальні дослідження – для тестування розробленої системи; методи економічного аналізу – для оцінки економічної ефективності.

Інформаційна база дослідження: нормативно-правові акти України у сфері вищої освіти, освітньо-професійні програми (ОПП) та методичні рекомендації закладів вищої освіти, офіційна технічна документація прикладних програмних інтерфейсів (API) великих мовних моделей, специфікації та документація технологічного стеку розробки, наукові публікації вітчизняних і зарубіжних вчених у галузі штучного інтелекту та автоматизації текстових процесів.

Апробація результатів дослідження. За результатами роботи опубліковано тези доповідей: «Інформаційні системи як інструмент конкурентної переваги в епоху цифрової трансформації. Матеріали науково-практичної конференції за підсумками проходження виробничих практик здобувачів вищої освіти спеціальності 126 Інформаційні системи та технології, кафедра інформаційних систем та технологій Полтавського державного аграрного університету, 22 жовтня 2025 р., м. Полтава; «Багатомодельний підхід до автоматичного генерування сценаріїв та прийняття рішень засобами штучного інтелекту». Студентські роботи за науковою тематикою кафедри інформаційних систем та технологій: матеріали XXII щорічного міждисциплінарного семінару, 25 листопада 2025 р., м. Полтава; «Використання LLM-моделей для інтелектуальної підтримки розроблення навчально-методичної документації». Матеріали XXI щорічної студентської наукової конференції «Сучасні інформаційні технології та інноваційні методики в економіці, менеджменті та бізнесі», 22 квітня 2026 р., м. Полтава; «Формалізація методичних вимог як етап проєктування системи автоматизованого створення силабусів». Менеджмент XXI століття : глобалізаційні виклики : матеріали X Міжнародної науково-практичної конференції, 23 квітня 2026 р., м. Полтава;

«Автоматизація формування силабусів освітніх компонентів на основі даних ОПП у форматі PDF». Матеріали Міжнародної науково-практичної конференції пам'яті професора Володимира Юрженка «Інтернаціоналізація технологічної, математичної та професійної освіти: досвід та перспективи», 14-15 квітня 2026 р., м. Переяслав.

Практична значущість роботи. Результати роботи можуть бути використані викладачами та навчально-методичними підрозділами закладів вищої освіти для автоматизації процесу створення силабусів ОК. Запропонована система дозволяє зменшити витрати часу на підготовку навчально-методичної документації та підвищити її відповідність встановленим вимогам.

Структура роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. Основний текст викладений на 57 сторінках, містить 11 таблиць, 10 рисунків. Список використаних джерел складається з 40 найменувань.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗОВАНОГО СТВОРЕННЯ СИЛАБУСІВ

1.1 Силабус освітнього компонента як інформаційний документ у межах освітньо-професійної програми

Силабус – це стислий опис освітнього компонента (ОК) , створений на основі робочої програми навчальної дисципліни (РПНД), і орієнтований насамперед на здобувача освіти. Силабуси впроваджені Міністерством освіти і науки України (МОНУ) спільно з Національним агентством із забезпечення якості вищої освіти (НАЗЯВО) для забезпечення прозорості та відповідності європейським стандартам вищої освіти [2, 3].

Силабус надає повну, структуровану та доступну інформацію про ОК у межах освітньо-професійної програми (ОПП) [2]. Він готується для здобувачів вищої освіти з метою пояснення змісту навчальної дисципліни, її політики, оцінювання результатів навчання тощо. Силабус навчальної дисципліни розробляється з урахуванням студентоцентрованого підходу [4]. Під час акредитації освітніх програм експерти НАЗЯВО приділяють особливу увагу аналізу силабусів.

Роль силабусу можна розглядати по-різному:

– для здобувача освіти це головний навігатор курсу. Він дає чітке розуміння того, чому варто вивчати цю дисципліну (мета), що саме буде вивчено (зміст), які навички будуть отримані (результати) та як ці досягнення будуть виміряні (оцінювання), що дозволяє студенту свідомо планувати свій навчальний час та зусилля;

– для викладача це інструмент педагогічного дизайну та самоорганізації. Розробка силабусу спонукає викладача структурувати курс, переосмислити методики викладання та узгодити їх із сучасними вимогами ринку праці та стандартами освіти;

– для забезпечення прозорості – силабус усуває можливість суб’єктивізму та непорозумінь. Чітко прописані правила гри (дедлайни, критерії оцінювання, політика перескладання) мінімізують конфліктні ситуації та корупційні ризики.

Загалом, силабус навчальної дисципліни сприяє забезпеченню інформаційних умов для формування цілісного уявлення про якість освітнього процесу під час вивчення навчальної дисципліни, формуванню індивідуальної освітньої траєкторії та укладається на принципах прозорості, об’єктивності, академічної доброчесності [4].

Структура силабусу затверджується кожним закладом освіти окремо. Рекомендовані складники силабусу у Полтавському державному аграрному університеті (ПДАУ) описують Методичні рекомендації щодо розроблення силабусу навчальної дисципліни [4].

Головною вимогою є відповідність силабусу ОПП. Зокрема, загальні компетентності (ЗК), спеціальні (фахові) компетентності (СК) та програмні результати навчання (ПР), закладені в ОПП, повинні бути впроваджені в силабус навчальної дисципліни без спотворень.

1.2 Освітньо-професійна програма як джерело вхідних даних

Освітньо-професійна програма (ОПП) – це фундаментальний нормативний документ, що регламентує цілі, зміст та організацію підготовки фахівців. Вона визначає освітню траєкторію та гарантує відповідність навчання стандартам вищої освіти, фіксує унікальний профіль фахівця, набір компетентностей та ПР, які має опанувати здобувач [5].

ОК (дисципліни, практики, курсові роботи) є інструментами реалізації цілей ОПП. ОПП слугує базою, на основі якої проєктується кожен окремий ОК. Це забезпечує цілісність освітнього процесу, де кожна дисципліна працює на загальний результат, а не існує сама по собі [6].

При розробленні силабусу викладач насамперед звертається до ОПП. Обсяг дисципліни у кредитах ЄКТС, перелік загальних і фахових компетентностей, коди

та формулювання ПР – усе це береться безпосередньо з ОПП і не може бути змінено чи переформульовано на розсуд викладача. Будь-яке відхилення від оригінального тексту ОПП означає розрив між задекларованими цілями програми та реальним змістом курсу, що неминуче виявляється під час акредитаційної експертизи. Точність перенесення даних є змістовою необхідністю, що впливає на оцінку якості освітньої програми в цілому.

ОПП будується так, щоб надати вичерпну інформацію про підготовку здобувача від моменту вступу до отримання кваліфікації. Взаємозв'язок між елементами програми та їх відображенням у силабусі наведено в таблиці 1.1.

Таблиця 1.1 – Основні структурні елементи ОПП та їх використання при створенні силабусу [4]

Елемент ОПП	Значення для формування силабусу
Загальна інформація (галузь, спеціальність, рівень)	Визначає контекст дисципліни, її статус (обов'язкова/вибіркова) та цільову аудиторію
Мета та профіль програми	Задає загальний вектор курсу, гарантуючи, що мета дисципліни не суперечить місії всієї програми
Перелік компетентностей (загальні та фахові)	Слугує основою для формулювання цілей курсу: які саме навички та здатності розвиває ця дисципліна
Програмні результати навчання	Є ключовим елементом, що переноситься в силабус. Дисципліна має забезпечувати досягнення конкретних ПРН із матриці відповідності
Обсяг освітніх компонентів (кредити ЄКТС)	Визначає загальне навантаження студента, кількість годин на аудиторну та самостійну роботу в силабусі
Структурно-логічна схема	Встановлює місце дисципліни в навчальному плані, визначає пререквізити (попередні курси) та постреквізити

У сучасній практиці українських університетів стандартом де-факто для оприлюднення та зберігання освітньо-професійних програм став формат PDF (Portable Document Format) [7]. Такий вибір зумовлений необхідністю уніфікації документації, зручністю архівування та забезпеченням незмінності вмісту офіційного документа, часто скріпленого кваліфікованим електронним підписом.

Однак, попри переваги для офіційного документообігу, такі як надійне відтворення форматування на будь-якому пристрої, захист від несанкціонованого редагування та широка сумісність із різними платформами, формат PDF має суттєві обмеження з погляду обробки даних. Це документ із фіксованою структурою,

призначений насамперед для візуального сприйняття людиною, а не для машинного аналізу. Відсутність структурованої розмітки та машиночитаності перетворює дані всередині ОПП на «статичний текст», що ускладнює автоматизований експорт інформації, змушуючи розробників силабусів вдаватися до ручного копіювання фрагментів тексту (рисунк 1.1).

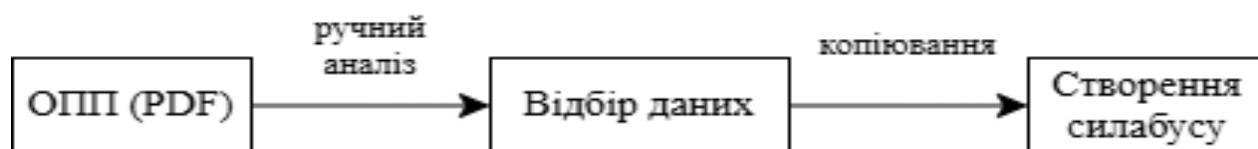


Рисунок 1.1 – Використання ОПП у форматі PDF як джерела вхідних даних

Необхідність «ручного» перенесення даних з PDF-файлів ОПП до шаблонів силабусів створює низку операційних та якісних ризиків. Цей процес не лише потребує значних витрат часу викладачів, які змушені виконувати механічну роботу замість творчої, але й провокує появу помилок [8].

Детальний аналіз можливих проблем ручного опрацювання ОПП та їх наслідків наведено у таблиці 1.2.

Таблиця 1.2 – Основні проблеми ручного опрацювання ОПП

Проблема	Наслідки
Ручне копіювання даних	Висока ймовірність механічних помилок (пропуски слів, неправильні коди компетентностей)
Великі часові витрати	Зниження ефективності роботи викладача, відволікання від методичного наповнення курсу на користь бюрократичних процедур
Неуніфіковані формулювання	Порушення узгодженості між ОПП та силабусами різних дисциплін, що ускладнює перевірку досягнення ПР
Складність оновлення	При внесенні змін до ОПП силабуси втрачають актуальність, а процес їхньої синхронізації стає тривалим і складним

Виявлені недоліки формату PDF спричиняють потребу у переосмисленні підходів до зберігання та обробки навчально-методичних матеріалів: зокрема, у переході від статичних, людиноорієнтованих документів до динамічних структур даних, придатних для машинного читання, міжсистемної інтеграції та повторного використання без втрати семантики.

1.3 Формалізація вимог методичних рекомендацій до розроблення силабусів

Формалізація вимог – необхідний крок до автоматизації процесів створення силабусів. Особливої актуальності формалізація вимог набуває в контексті цифровізації освіти [9]. Перехід до електронного документообігу та використання ІС управління навчанням вимагає, щоб правила створення документів були чіткими, однозначними та придатними для алгоритмічної обробки [10].

Методичні рекомендації закладів вищої освіти виступають нормативним базисом, що забезпечує уніфікацію підходів до викладання різнорідних дисциплін. В умовах академічної автономії внутрішні рекомендації університету гарантують, що незалежно від факультету чи кафедри, здобувач освіти отримає вичерпну та структуровану інформацію про кожен ОК.

У ПДАУ розроблено власні методичні рекомендації, які регламентують процес створення силабусів освітніх компонентів. Цей документ слугує дорожньою картою для науково-педагогічних працівників, визначаючи не лише зовнішній вигляд документа, але і його змістове наповнення та філософію. Сфера застосування рекомендацій охоплює всі освітні програми та рівні вищої освіти, що реалізуються в університеті, забезпечуючи стандартизацію навчально-методичного забезпечення [4]. Роль цих рекомендацій полягає у створенні чітких рамок, які, з одного боку, дисциплінують викладача, а з іншого – гарантують студенту отримання якісного освітнього продукту.

Вимоги, викладені в рекомендаціях ПДАУ, можна розділити на кілька груп, кожна з яких виконує свою функцію в системі забезпечення якості (таблиця 1.3).

Таблиця 1.3 – Основні групи вимог методичних рекомендацій ПДАУ [4, 12].

Група вимог	Характеристика та сфера регулювання
1	2
Структурні	Визначають обов’язковий «скелет» документа: наявність анотації, опису пререквізитів та постреквізитів, структури курсу, політики оцінювання та академічної доброчесності
Змістові	Регламентують сутність інформації: відповідність змісту курсу програмним результатам навчання, валідність методів контролю

Продовження таблиці 1.3

1	2
Організаційні	Встановлюють процедурні норми: терміни розроблення, порядок рецензування, затвердження на засіданні кафедри, періодичність перегляду та оновлення силабусу
Оформлювальні	Забезпечують візуальну стандартизацію: вимоги до шрифтів, відступів, стилів заголовків, використання корпоративної символіки університету, форматів файлів

Залежно від конкретної вимоги методичних рекомендацій, ступінь її придатності до автоматизованої перевірки може відрізнятися.

Формалізовані вимоги – це чітко визначені, однозначні критерії, виконання яких можна перевірити за принципом «так/ні» або шляхом порівняння числових значень. Вони не допускають подвійного тлумачення і є основою для побудови алгоритмів валідації в автоматизованій системі створення силабусів. У рекомендаціях ПДАУ до таких вимог належать: наявність усіх обов’язкових розділів, відповідність загальної кількості годин кредитам ЄКТС, наявність шкали оцінювання, що сумарно дає 100 балів тощо.

Неформалізовані вимоги мають описовий, рекомендаційний характер і стосуються переважно якісних аспектів контенту. Виконання таких вимог потребує експертної оцінки та розуміння контексту. До прикладів неформалізованих вимог можна віднести логічність та зрозумілість викладу матеріалу, відповідність тону комунікації принципам студентоцентрованості, актуальність рекомендованої літератури чи педагогічну доцільність обраних методів навчання.

Окремою проблемою є побудова засобів автоматизованого контролю відповідності силабусів. Ручна перевірка кожного документа на відповідність численним пунктам методичних рекомендацій є вразливою до помилок через людський фактор. Впровадження автоматизованих інструментів дозволить оптимізувати цей процес, змістивши акцент роботи з технічної вичитки на змістовий аналіз.

Можливості автоматизації насамперед стосуються формалізованих вимог. Сучасні ІС здатні миттєво перевірити структуру документа, наявність заповнених обов’язкових полів, коректність розподілу годин між лекціями та практичними

заняттями, а також відповідність кодів компетентностей даним освітньої програми. Використання такого механізму в ролі первинного фільтра дозволяє автоматично виявляти найдрібніші розбіжності та блокувати технічно некоректні документи ще на початкових етапах розробки, що суттєво пришвидшує загальний цикл погодження й звільняє фахівців від виснажливої технічної вичитки.

Натомість, існують об'єктивні обмеження автоматизації щодо неформалізованих вимог. Оцінка зрозумілості мови, етичності політики курсу чи інноваційності методів навчання залишається перевагою експертів. Перспективи розвитку в цьому напрямі пов'язані з використанням технологій обробки природної мови та штучного інтелекту, які в майбутньому зможуть надавати рекомендації щодо покращення стилістики чи виявляти логічні неузгодженості, проте остаточне рішення завжди прийматиметься людиною (рисунок 1.2).

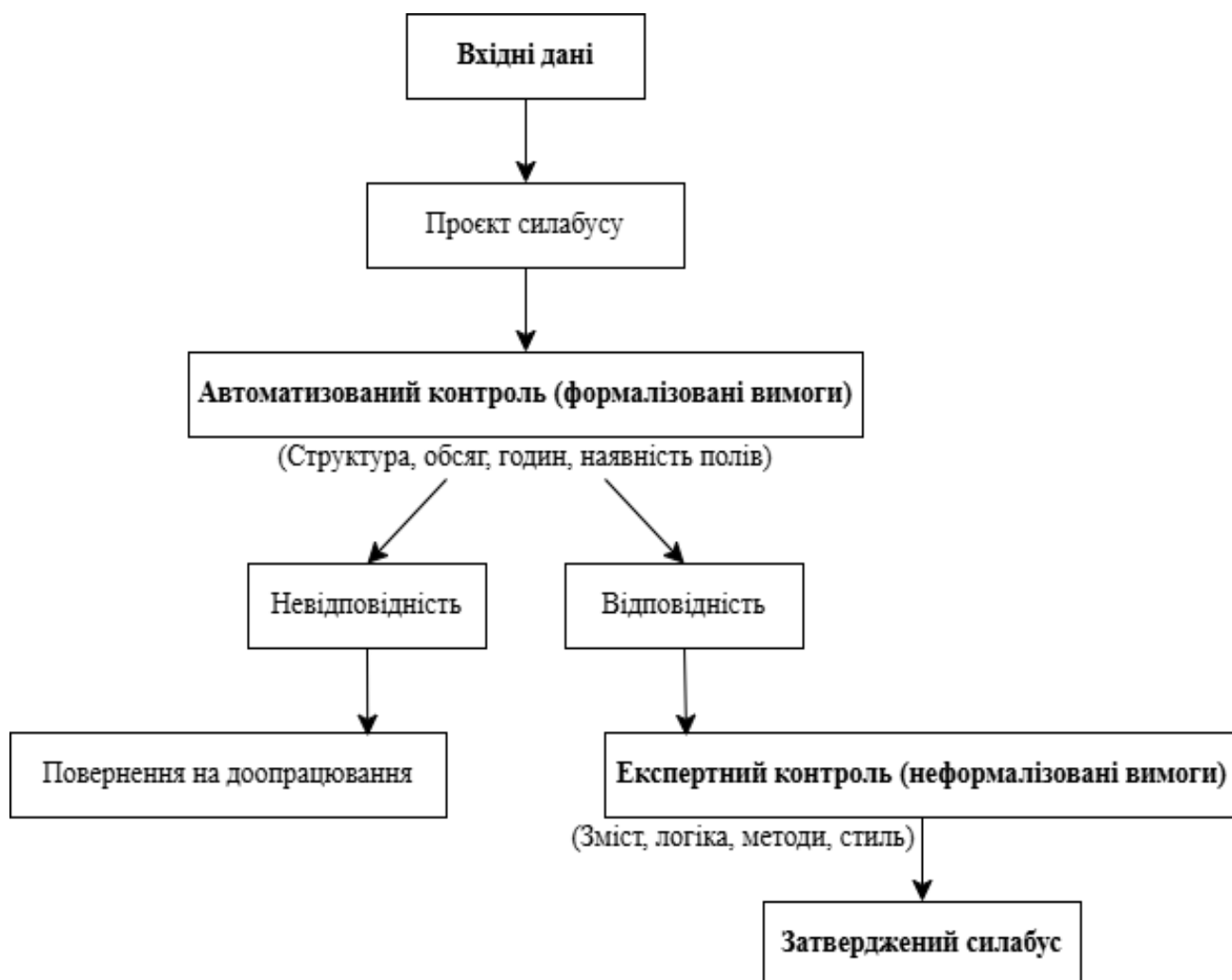


Рисунок 1.2 – Контроль відповідності силабусу методичним рекомендаціям

Складність автоматизованої обробки неформалізованих вимог полягає в тому, що програмні алгоритми поки що не здатні повноцінно оцінити семантику тексту чи методичну якість матеріалу. Тому, якщо формалізовані параметри легко перетворюються на правила валідації в ІС, то неформалізовані залишаються зоною відповідальності людини – викладача, методиста або гаранта освітньої програми.

1.4 Аналіз підходів до автоматизації створення силабусів

Для побудови моделі генерації силабусів необхідно дослідити, як сучасні ІС обробляють освітні дані та які інструменти генерації документів можуть бути адаптовані до потреб закладів вищої освіти. Під ІС підтримки освітнього процесу розуміють комплексні програмні рішення: системи управління університетом (UMS), як АСУ ПДАУ, та системи управління навчанням (LMS), як Moodle. Основні функції таких систем [11, 13]:

- зберігання та організація навчального контенту (лекції, презентації, завдання);
- керування розкладом та навчальними планами;
- комунікація між викладачами та студентами;
- прийом та перевірка завдань, оцінювання та ведення журналу успішності;
- тестування та контроль знань;
- відстеження активності та прогресу студентів;
- зберігання навчально-методичних документів (силабуси, робочі програми);
- управління доступом до матеріалів за роллю (студент, викладач);
- інтеграція із зовнішніми системами університету (деканат, бібліотека, відділ кадрів).

Найвні університетські системи виконують роль репозиторію готових документів або довідника вхідних даних. Автоматизація формування текстового змісту силабусів є обмеженою. Викладач завантажує готовий файл із зовнішнього

редактора або вручну заповнює форми. Системи рідко мають функціонал «конструктора», який би дозволяв зібрати готовий документ з наявних у базі даних фрагментів без участі користувача в рутинних операціях.

Засоби автоматичної генерації документів створюють документи на основі шаблонів і змінних даних. Підхід використовується в банківській сфері, юриспруденції та звітності і має потенціал для застосування в освіті. Їх принцип роботи базується на технології злиття даних, яка дозволяє автоматично поєднувати інформацію з різних джерел у єдину структуру [14]. Шаблонні рушії відповідають за відображення та форматування даних відповідно до заданого шаблону [15]. Процес створення силабусу за такого підходу складається з двох паралельних потоків: підготовки структурованих даних (наприклад, у форматі JSON, XML або вибірки з бази даних) та розробки шаблону документа з розміченими полями-плейсхолдерами (рисунок 1.3).

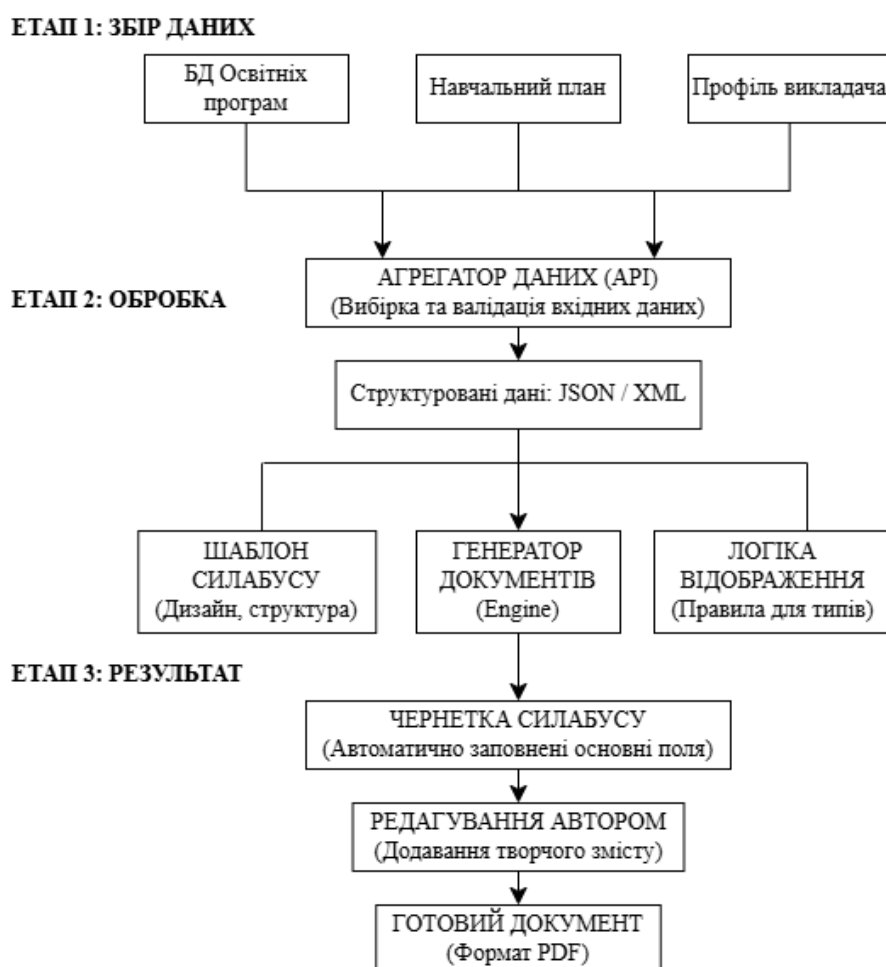


Рисунок 1.3 – Концептуальна модель автоматичної генерації силабусу

Можливості засобів генерації для створення силабусів включають:

- формування жорсткої структури – шаблон гарантує, що кожен згенерований силабус матиме ідентичний набір розділів, заголовків, шрифтів та відступів, що забезпечує повну відповідність оформлювальним вимогам;
- автоматичне наповнення розділів – система самостійно підставляє назву дисципліни, кількість кредитів, прізвище викладача, перелік компетентностей та ПР, витягуючи їх із бази даних ОПП;
- можливість включати або виключати певні блоки тексту залежно від типу дисципліни.

1.5. Обґрунтування задачі розроблення системи інтелектуальної підтримки створення силабусів

Повна автоматизація створення силабусів має певні обмеження:

- відсутність єдиних стандартів структури силабусів унеможливорює створення універсального програмного продукту та вимагає адаптації системи під конкретний університет;
- результати роботи автоматизованої системи залежать від якості вхідних даних – помилки або неуніфіковані дані в базі освітньої програми відтворюються у згенерованому документі;
- існуючі алгоритми не здатні автоматично сформувати неформалізований зміст, що потребує участі людини.

Розглянуті вище засоби вирішують лише частину завдань автоматизації – переважно зберігання файлів або шаблонну генерацію за фіксованими правилами. Традиційні засоби не здатні працювати з неструктурованими текстовими даними, зокрема PDF-файлами ОПП, та не забезпечують змістову валідацію контенту. Це часто призводить до ситуації, коли документ формально створений, але його наповнення містить логічні помилки або застарілі дані. Отже, виникає потреба у переході від простої автоматизації до створення системи інтелектуальної

підтримки, що має працювати як повноцінний інструмент підтримки викладача: самостійно аналізувати вхідні документи, пропонувати варіанти текстового наповнення та перевіряти відповідність вимогам.

Процес розроблення силабусів характеризується фрагментацією даних та домінуванням ручної праці: викладач вручну переносить інформацію з PDF-файлів освітніх програм, адаптує її до вимог методичних рекомендацій та структурує у форматі силабусу. Кожен етап цього процесу є вразливим до людського фактору, що призводить до системних проблем. Найбільш життєздатною є модель поєднання автоматизації для структури та формальних даних із ручним авторським наповненням змісту.

Основні проблеми процесу створення силабусів узагальнені у таблиці 1.4.

Таблиця 1.4 – Основні проблеми процесу створення силабусів

Проблема	Сутність та прояв	Наслідки для освітнього процесу
Дезінтеграція даних	Вхідні дані (ОПП) знаходяться у неструктурованих форматах (PDF), що унеможливує їх прямий імпорт та вимагає ручного копіювання	Ризик механічних помилок, розбіжність між ОПП та силабусом
Семантична неузгодженість	Труднощі у формулюванні методів навчання та оцінювання, які б реально забезпечували досягнення задекларованих програмних результатів	Формальний підхід до навчання, складність підтвердження якості освіти під час акредитації
Ресурсна неефективність	Високі часові витрати на технічне оформлення документа (форматування, вивірка) замість роботи над змістом	Перевантаження викладачів, зменшення часу на підготовку до занять та наукову діяльність
Суб'єктивізм контролю	Відсутність інструментів автоматичної перевірки призводить до різного тлумачення вимог методичних рекомендацій	Різнобій у якості та структурі силабусів у межах однієї кафедри або факультету

Вирішення даних проблем неможливе лише засобами традиційного програмування, оскільки значна частина роботи із силабусом стосується обробки природної мови (NLP) та генерації нового контенту [16]. Для цього доцільним є використання моделей штучного інтелекту (зокрема, великих мовних моделей – LLM).

Сучасні інтелектуальні моделі здатні [17]:

- виконувати семантичний аналіз тексту ОПП, виокремлюючи сутність компетентностей навіть із неструктурованого тексту;
- виконувати функцію генеративного асистента, пропонуючи викладачу варіанти опису пререквізитів, анотацій курсу чи політики оцінювання на основі заданих параметрів;
- здійснювати контекстний контроль, виявляючи не лише відсутність полів, а й змістові суперечності (наприклад, якщо для досягнення практичної навички заплановано лише лекційні години).

Для реалізації такого підходу необхідно сформулювати чіткі вимоги до програмного продукту. Оскільки система розробляється як інструментальний засіб для підтримки викладача, головним пріоритетом є баланс між функціональною потужністю та простотою використання. Система не повинна бути перевантажена зайвими адміністративними функціями, а має фокусуватися на вирішенні конкретної прикладної задачі – швидкому створенні якісного силабусу (таблиця 1.5).

Таблиця 1.5 – Вимоги до системи інтелектуальної підтримки створення силабусів

Вимога	Характеристика
Функціональні вимоги	
Парсинг вхідних даних	Здатність зчитувати текст із PDF-файлів ОПП, ідентифікувати та виокремлювати структурні блоки
Генерація контенту	Автоматичне створення чернетки силабусу із заповненням полів на основі вилучених даних та генерація текстових описів за допомогою API мовних моделей
Валідація структури	Перевірка створеного документа на наявність усіх обов'язкових елементів згідно з методичними рекомендаціями ЗВО
Експорт результату	Можливість збереження готового силабусу у редагованому форматі (DOCX) або фінальному (PDF)
Нефункціональні вимоги	
Зручність (Usability)	Інтуїтивно зрозумілий інтерфейс, що не потребує спеціальних технічних навичок від користувача-викладача
Адаптивність	Можливість легкого налаштування промптів (інструкцій для ШІ) для зміни тональності тексту або структури без переписування коду програми

Головною перевагою інтелектуального підходу є його гнучкість. На відміну від статичних шаблонів та регулярних виразів, що застосовуються у традиційному програмуванні, система на базі ІШІ адаптується до контексту конкретної дисципліни, надаючи рекомендації, релевантні саме для цього курсу. Це дозволяє залишити за викладачем роль експерта, який приймає фінальні рішення, делегувавши рутинну генерацію та первинну перевірку машині [18].

Основний акцент робиться на здатності системи «розуміти» вхідні файли ОПП та взаємодіяти з користувачем через зрозумілий інтерфейс. Реалізація сформульованих вимог у програмному продукті дозволить автоматизувати рутинні процеси, підвищити якість методичного забезпечення та забезпечити відповідність силабусів нормативним вимогам при мінімальних витратах часу викладачів.

РОЗДІЛ 2

ПРОЄКТУВАННЯ СИСТЕМИ ІНТЕЛЕКТУАЛЬНОЇ ПІДТРИМКИ СТВОРЕННЯ СИЛАБУСІВ

2.1 Формалізація вимог до інформаційної системи

Загальна характеристика функціоналу системи передбачає її роботу в режимі інтелектуальної підтримки, де користувач не просто отримує статичний шаблон, а взаємодіє з системою, що аналізує контекст.

Основні функціональні вимоги згруповані за чотирма напрямками:

- робота з документами – система має забезпечувати механізми імпорту даних, включаючи завантаження документів за прямим URL-посиланням або локальне завантаження файлів. Обов'язковою є підтримка PDF-формату, що є стандартним для оприлюднення ОПП;
- аналітична обробка тексту, що включає семантичне розпізнавання структурних елементів програми (мета, компетентності, результати навчання) та їхню класифікацію;
- генерація силабусу – на основі зібраних даних система повинна автоматично формувати логічну структуру силабусу, заповнюючи нормативні поля та пропонуючи варіанти текстового наповнення для методичних розділів;
- взаємодія з користувачем – функціонал системи має включати інструменти для ручного редагування згенерованого контенту та експорту результату у формат, зручний для подальшого друку або оприлюднення (наприклад, DOCX або PDF) [19].

Нефункціональні вимоги визначають якісні характеристики системи та умови її стабільного функціонування.

Важливим аспектом є зручність використання (usability). Оскільки цільовою аудиторією є викладачі з різним рівнем ІТ-компетентності, інтерфейс системи (UI) має бути інтуїтивно зрозумілим, а користувацький досвід (UX) – спрямованим на мінімізацію кількості кліків для досягнення результату.

Продуктивність та масштабованість системи мають дозволяти обробляти об'ємні PDF-документи без значних затримок у часі. Надійність передбачає коректну обробку виняткових ситуацій (наприклад, пошкоджених файлів або відсутності зв'язку з API), а безпека даних гарантує, що конфіденційна інформація про навантаження або внутрішні розробки кафедри не потрапить до третіх осіб.

При розробці системи необхідно враховувати низку технічних та операційних обмежень, які визначають рамки проекту.

По-перше, використання сторонніх API штучного інтелекту (наприклад, OpenAI або Anthropic) накладає обмеження на кількість запитів (quotas) та вимагає стабільного інтернет-з'єднання. По-друге, система критично залежить від якості вхідних документів: PDF-файли з низькою якістю сканування або відсутнім текстовим шаром можуть бути розпізнані некоректно, що потребуватиме ручного втручання.

Також існують мовні обмеження: система повинна коректно працювати зі специфічною термінологією українського законодавства у сфері вищої освіти. Крім того, критичним фактором є обмеження обчислювальних ресурсів на боці клієнта (браузерне виконання) або сервера (ліміти хмарного хостингу), що вимагають необхідності оптимізації алгоритмів обробки великих об'ємів тексту.

2.2 Загальна архітектура системи

Для реалізації системи інтелектуальної підтримки створення силабусів обрано сервісну архітектуру з чітким поділом на функціональні шари, що дозволяє ізолювати складні процеси аналізу тексту від інтерфейсної логіки.

Кожен програмний модуль спроектований як відносно незалежна одиниця, що взаємодіє з іншими через чітко визначені інтерфейси, що забезпечує модульність та спрощує подальше тестування окремих функцій. На рисунку 2.1 відображено організацію системи, де простежується механізми взаємодії компонентів.

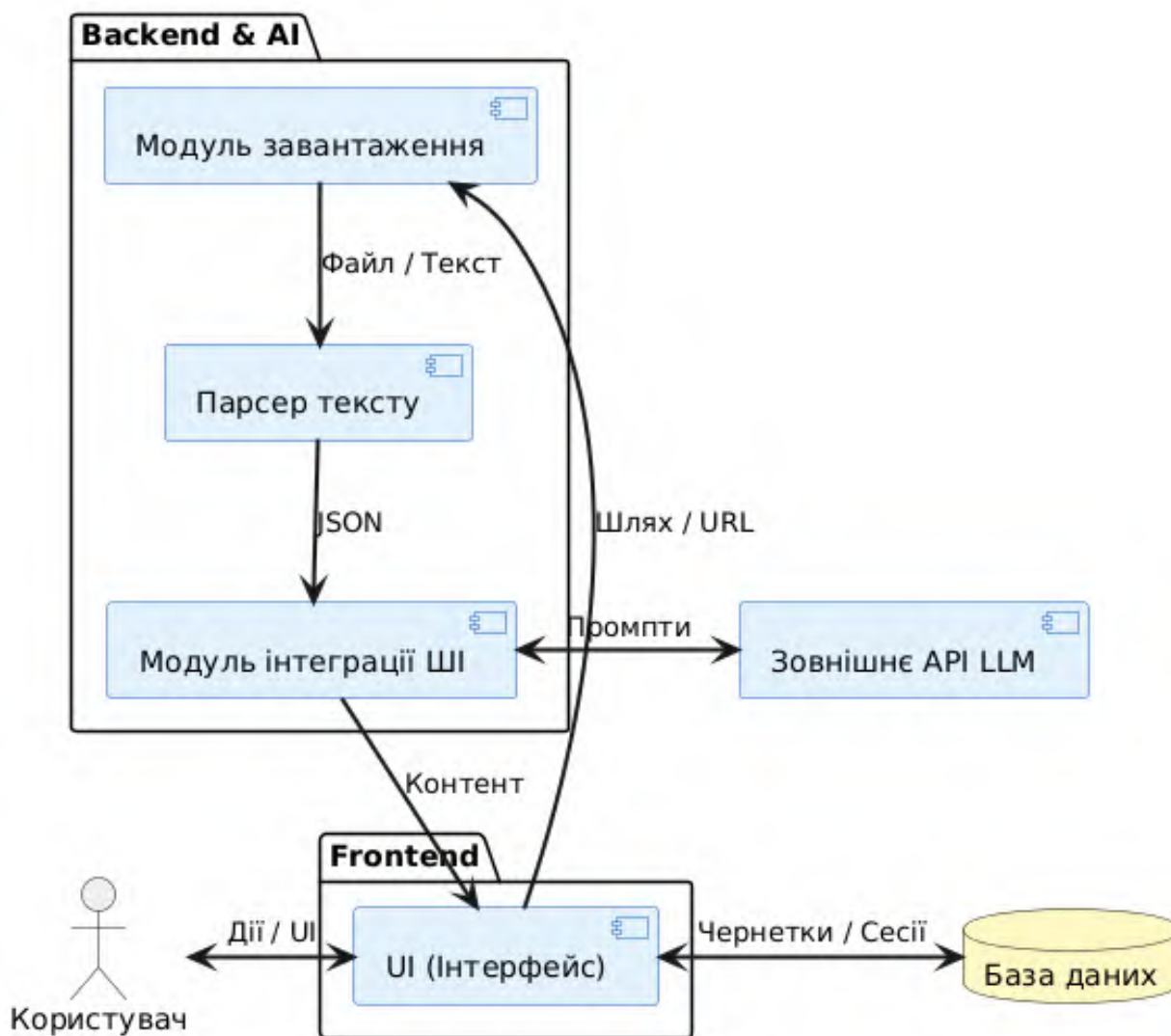


Рисунок 2.1 – Загальна архітектура системи інтелектуальної підтримки створення силабусів

Інтерфейс користувача (frontend) є точкою входу в систему, реалізованою на основі сучасних вебтехнологій. Він призначений для візуалізації процесу створення силабусу, забезпечення зручного завантаження файлів та надання інструментів для редагування згенерованого контенту. Вхідними даними для цього компоненту є дії певні користувача (кліки, введення тексту, вибір файлів), а вихідними – структуровані запити до серверної частини системи (backend) та відображення отриманих результатів у зручному для читання вигляді.

Модуль завантаження документів відповідає за первинну роботу з джерелами вхідних даних. Його головним призначенням є отримання файлів ОПП у форматі PDF, незалежно від способу їх надання: через локальне завантаження або

за посиланням (URL). Вхідними даними для модуля є шлях до файлу або мережева адреса документа, а вихідними – «сирий» двійковий потік даних або тимчасовий об'єкт файлу, готовий до передачі в модуль оброблення.

Модуль оброблення тексту (Parser) виконує роль технічного фільтра та структуроутворювача. Його завданням є розпізнавання текстового шару PDF-документа, очищення його від «сміттєвих» символів (номери сторінок, колонтитули) та сегментація на логічні блоки, такі як компетентності, результати навчання та назви дисциплін. Вхідними даними тут виступає неструктурований потік тексту з PDF, а вихідними – структурований JSON-об'єкт, де кожен елемент ОПП має свою мітку, що важливо для подальшої передачі в нейронну мережу.

Модуль інтеграції ШІ є інтелектуальним ядром системи, яке взаємодіє з API великих мовних моделей. Його призначенням є перетворення структурованих даних ОПП на методично грамотні описи для силабусу шляхом формування складних промптів (інструкцій). Початковими даними є сегментований текст програми та метадані дисципліни, а вихідними – згенеровані варіанти анотацій, політик курсу та описів результатів навчання, адаптовані під стиль конкретного ЗВО.

База даних/Сховище забезпечує тривале зберігання інформації в системі. Вона використовується для збереження поточних чернеток силабусів, налаштувань користувача та шаблонів методичних рекомендацій. Система отримує на вхід результати роботи всіх попередніх модулів (текстові блоки, метадані, результати генерації ШІ), а вихідними – вибірки даних для відновлення сесії користувача або фінального експорту документа.

При виборі архітектурного підходу розглядалися три варіанти: монолітна, модульна та мікросервісна архітектури. Для даної розробки обрано модульну архітектуру (Modular Monolith), що поєднує в собі простоту розгортання з перевагами розділення функцій [20, 22]:

- при зміні методичних рекомендацій ЗВО або оновленні версії AI API простіше модифікувати лише один відповідний модуль, не зачіпаючи логіку інтерфейсу чи бази даних;

– модуль аналізу PDF або інтеграції з ШІ може бути повторно використаний в інших університетських сервісах без глобальних змін коду;

– сервісна орієнтованість дозволяє легко інтегрувати систему з іншими інформаційними ресурсами університету (наприклад, порталом розкладу або LMS Moodle).

Логіка взаємодії елементів системи побудована на послідовному перетворенні неструктурованого документа на формалізований результат (рисунок 2.2).

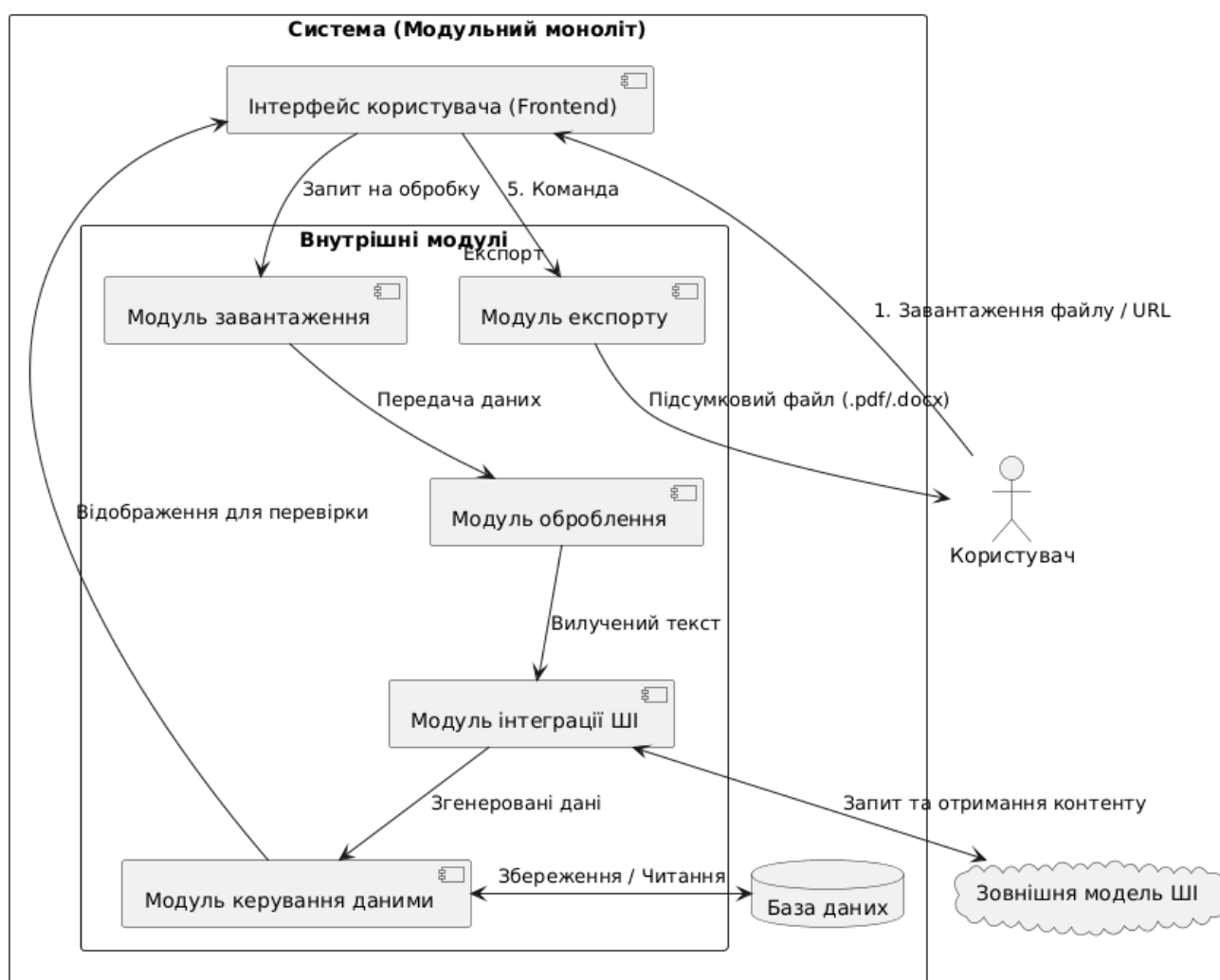


Рисунок 2.2 – Логіка взаємодії елементів системи

Послідовність обміну даними в системі виглядає наступним чином:

1. Користувач через браузер завантажує файл або вказує URL, що ініціює запит від Frontend до Backend;

2. Модуль завантаження захоплює файл і передає його в Модуль оброблення, де відбувається вилучення тексту;
3. Отриманий структурований текст передається в Модуль інтеграції ШІ, який формує запит до зовнішньої моделі та отримує згенерований контент;
4. Згенеровані дані повертаються на Frontend для перевірки користувачем та одночасно фіксуються у Базі даних;
5. Після фінального редагування користувач ініціює команду експорту, і система формує підсумковий файл силабусу.

2.3 Модель оброблення вхідних документів

Модель оброблення документів забезпечує перехід від візуального представлення документа до його логічної структури, придатної для аналізу алгоритмами штучного інтелекту. Оскільки ОПП найчастіше зберігаються у форматі PDF, процес їхнього перетворення на машиночитаний вигляд є складним багатоетапним завданням.

Для забезпечення максимальної зручності користувача, система реалізує механізм доступу до документів за прямим посиланням (URL), що дозволяє викладачеві не завантажувати файл локально на свій пристрій, а просто скопіювати адресу документа з офіційного сайту закладу вищої освіти або репозиторію.

Процес завантаження починається з ініціації HTTP-запиту до віддаленого сервера. На цьому етапі система виконує первинну перевірку доступності ресурсу. Якщо сервер повертає статус, відмінний від «200 OK» (наприклад, помилку 404 або 403), система інформує користувача про неможливість отримання документа. Особливу увагу приділено безпеці: механізм завантаження обмежує час очікування (timeout), щоб уникнути зависання потоків при роботі з повільними серверами.

Важливим кроком є валідація даних на етапі отримання заголовків (headers). Система перевіряє MIME-тип файлу (Content-Type), щоб переконатися, що за посиланням дійсно знаходиться документ формату application/pdf. Такий

превентивний контроль формату дозволяє уникнути спроб обробки некоректних даних (наприклад, HTML-сторінок або зображень), що могли б призвести до критичних помилок у роботі парсера.

Формат PDF за своєю природою не є текстовим у класичному розумінні. Через це прямий витяг тексту часто супроводжується проблемами порушення логічної послідовності рядків, особливо у документах із таблицями або двоколонковим верстанням [22-23]. Основні проблеми форматування під час екстракції включають «розриви» слів через переноси, втрату пробілів між символами та некоректне розпізнавання табличних структур, які в ОПП використовуються для опису компетентностей. Окремим викликом є кодування тексту. Старіші версії ОПП можуть використовувати специфічні кодування (наприклад, Windows-1251), що при некоректній обробці призводить до появи нечитабельних символів. Система використовує сучасні бібліотеки розпізнавання, що підтримують перекодування в універсальний стандарт UTF-8.

На рисунку 2.3 показано, як система аналізує макет сторінки, виокремлює текстові блоки та намагається відновити їхню логічну послідовність.



Рисунок 2.3 – Процес витягу тексту з PDF

Після отримання «сирого» тексту настає етап попередньої обробки, мета якого – очистити дані від шумів. Процес починається з видалення технічних артефактів: номерів сторінок, колонтитулів із назвами університету, зайвих

табуляцій та множинних пробілів. Наступним кроком є нормалізація тексту, що включає приведення символів до єдиного реєстру (де це необхідно) та виправлення помилок кодування. Важливу роль відіграє токенизація – поділ тексту на окремі речення та смислові одиниці, що дозволяє алгоритму точніше ідентифікувати межі розділів.

Фінальним етапом є сегментація даних. Система шукає задані ключові слова («Загальні компетентності», «Програмні результати навчання», «Обсяг курсу») для виділення відповідних розділів ОПП, і перетворює лінійний текст на деревоподібну структуру даних, де кожному гілці (розділу) відповідає свій масив інформації. Технологічна послідовність, що дозволяє усунути специфічні дефекти формату PDF та підготувати контент до інтелектуального аналізу, представлена у таблиці 2.1.

Таблиця 2.1 – Етапи попередньої обробки тексту

Назва етапу	Технологічне завдання	Очікуваний результат
Вилучення метаданих	Пошук та вилучення ідентифікаторів документа: шифр спеціальності, назва ОПП, рік затвердження	Сформований профіль документа для контекстної прив'язки ІІІ
Детекція логічних списків	Відновлення цілісності маркованих та нумерованих списків, які часто розриваються при витягу з PDF	Об'єднання розрізнених рядків у логічно завершені пункти компетентностей
Фільтрація рекурентних патернів	Видалення повторюваних елементів: назви університету, підписів відповідальних осіб	Мінімізація «шуму», що не несе змістового навантаження для силабусу
Семантичне мапування	Асоціація витягнутих текстових блоків із конкретними розділами майбутнього силабусу за допомогою регулярних виразів	Розподіл тексту за категоріями
Контроль цілісності кодування	Виявлення та корекція символів, що не належать до стандартного набору UTF-8 (артефакти вбудованих шрифтів PDF)	Текст без некоректних гліфів та нечитабельних символів
Підготовка контекстного вікна	Оптимізація довжини отриманого тексту для відповідності лімітам токенів обраної моделі ІІІ	Компактний, але інформативно насичений масив даних для передачі в API

Така багаторівнева підготовка гарантує, що модель штучного інтелекту отримає на вхід не хаотичний набір літер, а чітко структуровану інформацію, що є важливим для генерації методично коректного силабусу.

2.4 Агрегація та узгодження даних з різних джерел

Процес інтелектуального створення силабусу не обмежується витягом тексту з одного документа, а потребує використання трьох фундаментальних джерел: ОПП, що визначає змістовий стандарт, методичних рекомендацій закладу, що встановлюють правила оформлення та зразків (шаблонів) силабусів, які задають стилістичний тон. Агрегація цих даних – це не просто механічне поєднання, а процес логічного синтезу, що потребує вирішення конфліктів та усунення надмірності.

Першим етапом агрегації є встановлення зв'язків між вимогами різних джерел. Кожне з них виконує свою роль у формуванні майбутнього документа:

- ОПП слугує джерелом фіксованих даних для формування силабусу. Попри незмінність цих показників, викладач адаптує їх до специфіки конкретної дисципліни та умов її викладання;

- методичні рекомендації регламентують структуру силабусу, вимоги до критеріїв оцінювання, включення обов'язкових політик (зокрема академічної доброчесності та використання студентами інструментів штучного інтелекту);

- зразки силабусів слугують стилістичним орієнтиром, оскільки аналіз успішних прикладів допомагає сформувати розуміння тональності викладу.

Об'єднання відбувається через створення проміжних об'єктів даних. Наприклад, якщо ОПП вимагає формування «здатності до аналізу», а методичні рекомендації вимагають описувати результати через дієслова таксономії Блума, система інтегрує ці вимоги, генеруючи формулювання типу «Студент зможе класифікувати та порівнювати дані...».

Поширеною проблемою агрегації є суперечність даних. Ситуація, коли в ОПП вказано один обсяг годин, у навчальному плані інший, а в методичних рекомендаціях міститься застаріла назва кафедри, є типовою для академічного середовища.

Процес усунення суперечностей у системі базується на принципах ієрархії пріоритетів [24]:

- пріоритет нормативності (дані з офіційної ОПП та навчального плану мають вищий пріоритет над текстовими шаблонами);
- пріоритет актуальності (якщо методичні рекомендації університету оновилися пізніше, ніж був створений зразок силабусу, система автоматично ігнорує структуру зразка на користь чинних рекомендацій);
- семантичне дедублювання (компетентності в ОПП часто перетинаються за змістом, тому система групує схожі вимоги, уникаючи дублювання в силабусі).

Для вирішення конфліктів використовується алгоритм порівняння «ключ-значення». Якщо для одного ключа (наприклад, «форма підсумкового контролю») знайдено два різні значення (іспит/залік), система не приймає рішення самостійно, а маркує цей блок як «конфліктний» та виносить його на розгляд користувача (викладача), забезпечуючи тим самим контроль над точністю документа.

Фінальним етапом агрегації є створення єдиної інформаційної моделі – це цифрова структура, яка містить у собі всі необхідні складники для генерації фінального документа. У цій моделі дані перестають бути просто текстом і стають взаємопов'язаними об'єктами. Завдяки її формуванню досягається головна мета – цілісність даних. Система гарантує, що жодна компетентність з ОПП не буде втрачена, а кожне завдання, описане в силабусі, буде прямо працювати на досягнення результатів навчання, закладених у програмі (таблиця 2.2).

Таблиця 2.2 – Структура єдиної інформаційної моделі силабусу

Компонент моделі	Функція в моделі
Профіль компонента	Формує паспорт дисципліни (код, назва, статус)
Матриця зв'язків	Пов'язує конкретні теми лекцій із компетентностями
Фільтрація рекурентних патернів	Задає послідовність розділів та правила форматування
Регуляторна рамка	Розподіл тексту за категоріями
Методичний блок	Описує стратегії викладання, методи оцінювання та критерії
Етичний кодекс	Містить уніфіковані політики щодо плагіату та поведінки

Таким чином, агрегація даних трансформує розрізнені масиви академічної інформації у цілісну цифрову екосистему, де нормативна точність ОПП поєднується зі структурною логікою методичних стандартів.

2.5 Інтеграція API штучного інтелекту

Інтелектуальна складова системи реалізована через API-доступ до великих мовних моделей, яким делегується обробка природної мови та генерація методичного контенту. Процес інтеграції охоплює вибір оптимального технологічного рішення, розробку стратегії взаємодії з моделлю та побудову надійних механізмів перевірки згенерованих даних.

Вибір конкретної моделі для генерації силабусів визначається сукупністю технічних та якісних характеристик, що впливають на точність результату. Здатність моделі до складного логічного виведення є головним параметром, оскільки створення силабусу вимагає не просто відтворення тексту, а встановлення семантичних зв'язків між освітніми цілями та змістом тем [25]. Важливим критерієм є також обсяг контекстного вікна моделі, що дозволяє завантажувати повний текст освітньо-професійної програми без ризику втрати інформації на фінальних етапах обробки.

Модель має коректно оперувати українською академічною термінологією, включаючи поняття національної рамки кваліфікацій, дотримуючись офіційного стилю викладу. Порівняльний аналіз провідних рішень наведено в таблиці 2.3.

Таблиця 2.3 – Порівняльна характеристика моделей ШІ для інтеграції в систему

Модель	Постачальник	Розмір контексту (токенів)	Специфіка використання
Безкоштовно			
Gemini 2.5 Flash	Google AI Studio	1000000+ токенів	Оптимальна для обробки надвеликих пакетів документів одночасно
Llama 4 Scout	Groq Cloud	128000 токенів	Максимальна швидкість генерації; ідеально для швидких правок
Mistral NeMo	Hugging Face	32000 токенів	Висока точність у логічних завданнях та відкрита архітектура
Qwen 2.5	OpenRouter	64000 токенів	Гнучке рішення для генерації технічного контенту
Платно			
GPT-4o mini	OpenAI API	128000 токенів	Універсальне рішення з високою точністю дотримання інструкцій

Проектування промптів у цій системі розглядається як створення структурованого технічного завдання для нейронної мережі. Стратегія базується на поєднанні системних інструкцій та динамічних контекстних даних [26]. Системний промпт визначає рольову модель ШІ як досвідченого експерта з розробки освітніх програм та методиста закладу вищої освіти.

Процес формування запиту передбачає інтеграцію попередньо підготовлених даних із бази ОПП та методичних рекомендацій. Використовується метод ланцюжка думок, що змушує модель спочатку проаналізувати логічну послідовність результатів навчання, а вже потім переходити до написання тексту розділів силабусу. Важливою складовою є надання чітких обмежень щодо формату відповіді, оскільки система очікує структуровані дані для подальшої обробки в інтерфейсі.

Динамічний промпт – це запит до ШІ, який містить змінні, шаблони або логічні умови, що змінюються залежно від контексту, вхідних даних або випадкового вибору [27]. Промпт будується за модульним принципом, де кожен блок відповідає за окремий розділ силабусу, що зменшує навантаження на модель і підвищує точність формування критеріїв оцінювання та політик курсу. Використання прикладів (Few-shot prompting) у тілі запиту допомагає моделі зрозуміти специфічний стиль оформлення, прийнятий у ЗВО, що робить згенерований текст максимально наближеним до оригінальних зразків.

Оскільки відповіді LLM є недискретними [28], API налаштовується на повернення результатів у форматі JSON, що забезпечує сумісність з іншими модулями системи та дозволяє автоматично розподіляти згенеровані блоки за відповідними полями форми й проводити їх технічну перевірку.

Валідація результатів здійснюється на двох рівнях: синтаксичному та семантичному. На синтаксичному рівні система перевіряє цілісність отриманого файлу та відповідність типів даних. На семантичному рівні застосовуються детерміновані алгоритми для контролю відповідності ключових параметрів.

Для мінімізації ризиків «галюцинацій» система порівнює згенеровані власні назви, коди дисциплін та перелік літератури з еталонними даними, що зберігаються

в базі. У разі виявлення значних відхилень або нелогічних тверджень система маркує такий блок як «потребує уваги» та пропонує викладачеві кілька варіантів виправлення.

2.6 Алгоритм автоматизованого формування силабусу

На відміну від звичайного заповнення шаблону документу, автоматизований алгоритм реалізує динамічну модель побудови документа, де кожен наступний крок залежить від змісту та повноти даних, отриманих на попередніх етапах. Основна мета цього алгоритму полягає в перетворенні формальних вимог ОПП на структурований, зрозумілий для студента документ, що відповідає всім критеріям академічної якості.

Логічну послідовність операцій алгоритму автоматизованого формування силабусу ОК та механізми їх реалізації представлено в таблиці 2.4.

Таблиця 2.4 – Логічні кроки алгоритму формування силабусу

Фаза алгоритму	Зміст операції	Механізм реалізації
Аналітична	Ідентифікація структури ОПП та вилучення атрибутів навчального компонента	Регулярні вирази та семантичні фільтри
Координаційна	Мапування вилучених даних на структуру обраного шаблону силабусу	Логічні умови (If-Then) та шаблонізатор
Креативна	Генерація унікальних описів розділів (анотація, методи навчання, політики)	Запити до великої мовної моделі (LLM API)
Контрольна	Крос-валідація згенерованого тексту на відповідність вхідним параметрам	Порівняльний аналіз ключових токенів

Діаграма процесу представлена у додатку А.

Процес починається з етапу ініціалізації та збору первинних даних, де система ідентифікує вхідний файл та визначає його приналежність до конкретної спеціальності. Після успішного завантаження запускається процедура декомпозиції тексту, під час якої алгоритм виокремлює важливі сутності, такі як ПР та ОК. Цей етап важливий для подальшої роботи, оскільки будь-яка втрата даних на рівні парсингу призведе до логічних розривів у фінальному

документі [29]. Визначення розділів, що потребують обробки LLM, відбувається шляхом зіставлення витягнутих даних зі структурою шаблону силабусу. Наступним кроком є етап контекстуалізації, де витягнуті дані порівнюються з активним шаблоном методичних рекомендацій. Алгоритм визначає, які розділи силабусу можуть бути заповнені автоматично шляхом прямого переносу (наприклад, код і найменування спеціальності або рівень вищої освіти), а які потребують інтелектуальної обробки LLM. Завершується основний цикл етапом синтезу, на якому відбувається звернення до API штучного інтелекту для генерації описів стратегій викладання, політик курсу та критеріїв оцінювання.

Генерація розділів відбувається за ієрархічним принципом, де спочатку формується «паспорт» дисципліни, а потім її змістовне наповнення. Нормативний контроль перевіряє не лише наявність розділів, а й коректність термінології та відповідність формальним вимогам університету: якщо методичні рекомендації вимагають конкретних формулювань щодо академічної доброчесності, алгоритм замінює згенеровані варіанти на еталонний текст із бази знань. До того ж, алгоритм перевіряє, чи кожен задекларований результат навчання має відповідний інструмент вимірювання в розділі контрольних заходів.

РОЗДІЛ 3

РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ РОБОТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Обґрунтування вибору програмних засобів

Для реалізації програмного забезпечення з автоматизованого формування силабусів було обрано мову програмування Python, яка на сьогодні є однією з найпоширеніших у сфері розробки інтелектуальних систем та автоматизації [30]. Її синтаксис відрізняється лаконічністю і читабельністю, що суттєво скорочує час розробки та спрощує подальшу підтримку коду. Важливою перевагою Python є розвинена екосистема бібліотек, які охоплюють широкий спектр задач (наприклад, роботи з файлами різних форматів та взаємодії із зовнішніми API). Саме ця універсальність зробила цю мову оптимальним вибором для проєкту, де потрібно було поєднати роботу з PDF-документами, інтерактивним введенням даних та генеративним штучним інтелектом.

Середовищем виконання було обрано Google Colaboratory [31] – хмарну платформу від компанії Google, що надає доступ до інтерактивних Jupyter-блокнотів у браузері без необхідності локального встановлення будь-якого програмного забезпечення. Додатковою перевагою є вбудована підтримка апаратного прискорення у вигляді GPU та TPU. Ці особливості є важливими з точки зору доступності розробленого інструменту для кінцевого користувача – викладача, який може не мати технічного досвіду налаштування середовища розробки. Google Colab забезпечує можливість зберігання конфіденційних даних, а саме API-ключів, через вбудований менеджер секретів, що дозволяє уникнути їх відкритого зберігання у коді. Блокнотна структура середовища також природно відповідає покроковому характеру розробленої програми, де кожен етап формування силабусу виконується в окремій комірці.

Для наочного представлення загальної логіки функціонування розробленої системи на рисунку 3.1 наведено блок-схему алгоритму її роботи.

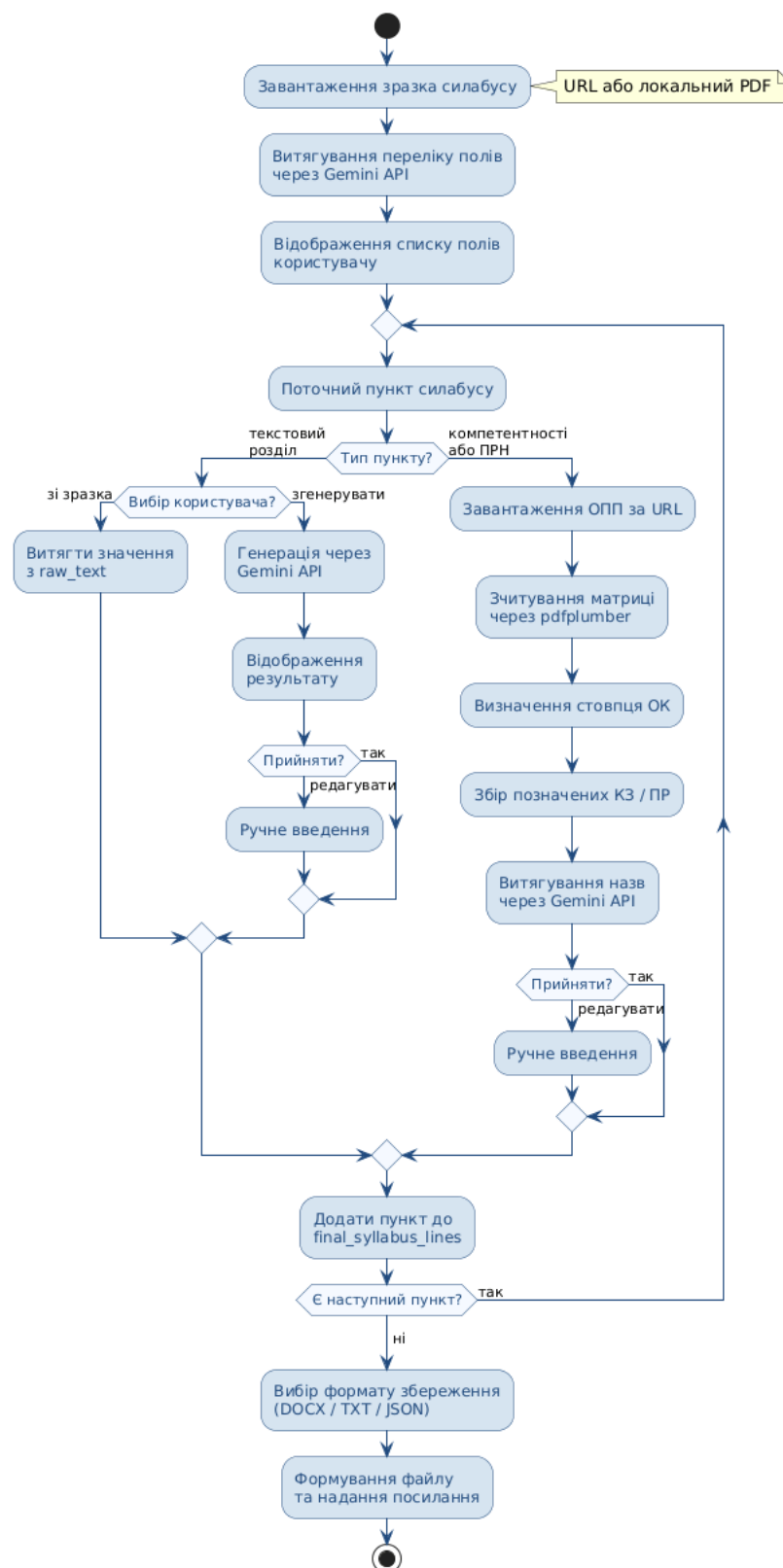


Рисунок 3.1 – Загальна блок-схема алгоритму роботи системи

Схема відображає послідовність основних етапів та точки прийняття рішень, у яких система звертається до зовнішнього API або передає керування користувачу для підтвердження чи редагування результату.

Для роботи з PDF-документами було використано бібліотеки з різним призначенням. Бібліотека PyPDF застосовувалась для швидкого витягування тексту з перших сторінок документів у випадках, коли вміст не містить складних табличних структур [32]. Для роботи з таблицями було задіяно бібліотеку pdfplumber, яка забезпечує значно точніше розпізнавання структури таблиць завдяки аналізу координат елементів на сторінці [33]. Бібліотека requests використовувалась для завантаження PDF-файлів та веб-сторінок за URL-адресами, а BeautifulSoup4 – для парсингу HTML-вмісту у випадках, коли джерелом даних слугувала веб-сторінка, а не документ [34-35]. Для формування вихідного документа у форматі Word застосовувалась бібліотека python-docx [36], що дозволила реалізувати структуроване табличне представлення силабусу з форматуванням, придатним до безпосереднього використання у навчальному процесі.

Головним компонентом системи став Gemini API від компанії Google, який забезпечує доступ до LLM для виконання задач генерації та аналізу тексту. Для взаємодії з ним використовувалась офіційна бібліотека google-generativeai [37]. З міркувань економії ресурсів та мінімізації ризику перевищення лімітів безкоштовного рівня використання було обрано модель gemini-2.5-flash-lite, яка забезпечує достатню якість генерації при значно менших витратах токенів порівняно з потужнішими версіями. Для підвищення стійкості системи до обмежень API було реалізовано механізм ротації між кількома ключами та автоматичного повторного надсилання запиту після витримки паузи у разі отримання помилки перевищення квоти.

3.2 Реалізація модуля завантаження та аналізу PDF-документів

Одним із головних завдань при розробці системи було забезпечення можливості отримання вхідних даних з різних джерел – як з локально збережених файлів, так і з документів, розміщених у відкритому доступі в мережі. Це

зумовлено реальними умовами використання програми: частина навчально-методичних матеріалів університету зберігається на офіційних сайтах у вигляді PDF-файлів, тоді як інші документи викладач може мати лише локально. Тому на початку роботи система пропонує користувачу самостійно обрати спосіб надання документа, що робить її гнучкою і не обмежує сценарії застосування.

Завантажений вміст не зберігається на диск, а одразу передається до об'єкта BytesIO зі стандартної бібліотеки io, який забезпечує роботу з даними безпосередньо в оперативній пам'яті [38]. Такий підхід дозволяє уникнути зайвих операцій запису та читання файлів і прискорює обробку документа. Для запитів встановлено тайм-аут у 10–30 секунд на випадок недоступності ресурсу. Після отримання відповіді від сервера викликається метод `raise_for_status`, який автоматично ініціює виняток при отриманні HTTP-кодів помилок.

Витяг тексту з PDF реалізовано з використанням двох різних бібліотек залежно від характеру вмісту документа. Для отримання тексту із звичайних сторінок силабусу-зразка використовується PyPDF, яка послідовно обходить сторінки документа і витягує текстовий шар методом `extract_text`. З метою зменшення кількості токенів, що передаються до LLM, обробка обмежується лише першими трьома сторінками документа, оскільки загальна інформація про дисципліну, як правило, зосереджена саме на початку силабусу. Для роботи з табличними структурами освітньо-професійної програми застосовується бібліотека `pdfplumber`, яка аналізує геометрію елементів на сторінці та дозволяє коректно відновити структуру рядків і стовпців таблиці. Це особливо важливо при роботі з матрицями відповідності компетентностей та програмних результатів навчання, де кожна комірка таблиці має чітке позиційне значення і її неправильне зчитування призводить до хибного результату. Завантажений байтовий потік зберігається в пам'яті протягом усього сеансу роботи, а метод `seek(0)` використовується для його перемотування на початок перед кожним повторним зверненням, що дозволяє уникнути повторного завантаження файлу з мережі.

Обробка виняткових ситуацій реалізована так: на рівні мережевих запитів перехоплюються винятки класу `requests.exceptions.RequestException`, який охоплює

помилки з'єднання, тайм-аути та некоректні відповіді сервера, а на рівні роботи з файловою системою перевіряється існування локального файлу перед спробою його відкрити, а для всіх операцій читання PDF передбачено загальний блок експерт для перехоплення непередбачуваних помилок формату або пошкодження документа.

У всіх випадках функції повертають значення `None` замість того щоб переривати виконання програми, а основна логіка перевіряє це значення перед подальшою обробкою. Такий підхід забезпечує стійкість системи до некоректних вхідних даних і дозволяє інформативно повідомити користувача про проблему без аварійного завершення роботи всього блокнота.

3.3 Реалізація модуля взаємодії з API штучного інтелекту

Взаємодія з LLM є центральним елементом розробленої системи, оскільки саме на неї покладається виконання інтелектуальних задач: аналіз структури документів, витягування структурованої інформації, генерація текстових фрагментів силабусу та формування списків джерел. Якість і надійність цієї взаємодії визначають практичну цінність усього програмного забезпечення, тому реалізації відповідного модуля приділялась особлива увага як з точки зору коректності результатів, так і з точки зору стійкості до збоїв.

Підготовка вхідних даних зводиться до формування текстового промпту, який передається моделі разом із фрагментом оброблюваного документа. Оскільки мовні моделі сприймають увесь контекст як єдиний текстовий рядок, структура промпту відіграє вирішальну роль у якості отриманої відповіді [39].

У розробленій системі кожен промпт складається з двох частин: інструктивної, де чітко формулюється завдання та вимоги до формату відповіді, і контекстної, де розміщується витягнутий фрагмент документа. Інструктивна частина містить явні вказівки щодо того, що слід включати до відповіді, а що ігнорувати (наприклад, при витягуванні полів силабусу модель отримує пряму

вказівку не додавати коментарів і повернути виключно нумерований список назв). Для розмежування двох секцій промпту використовуються візуальні роздільники у вигляді рядка з великими літерами, що допомагає моделі чітко відокремити інструкцію від даних. Обсяг переданого тексту документа свідомо обмежується (як правило, до 3000-4000 символів) щоб уникнути зайвих витрат токенів і не перевантажувати контекстне вікно інформацією, яка не є релевантною для конкретної задачі.

Ініціалізація клієнта відбувається через виклик `genai.configure` з передачею відповідного ключа, після чого створюється об'єкт моделі і викликається метод `generate_content` з підготовленим промптом. Для підвищення стійкості системи до обмежень безкоштовного рівня використання API реалізовано функцію `generate_with_retry`, яка інкапсулює логіку повторних спроб і ротації ключів. При отриманні помилки з кодом 429, що сигналізує про перевищення квоти [40], функція послідовно перемикається між доступними ключами зі списку, намагаючись виконати запит через інший акаунт. Якщо всі наявні ключі вичерпані в межах поточного циклу, система автоматично призупиняє виконання на 30 секунд, після чого відновлює спроби. Після кожного успішного запиту витримується мінімальна пауза в одну секунду, що знижує інтенсивність звернень до API і зменшує ймовірність повторного перевищення ліміту. Ротація ключів реалізована через глобальний лічильник `key_index`, який інкрементується при кожному виклику функції `get_model` і береться за модулем від кількості доступних ключів, що забезпечує рівномірне розподілення навантаження між ними (Додаток Б).

На рисунку 3.2 наведено блок-схему алгоритму функції `generate_with_retry`, яка забезпечує автоматичну ротацію API-ключів та повторне надсилання запитів у разі перевищення квоти. Логіка розгалуження триступенева – спочатку перевіряється успішність запиту, потім тип помилки, і лише після цього вирішується чи є ще невикористані ключі.

Обробка відповідей моделі залежить від характеру конкретної задачі. У випадках, коли від моделі очікується структурована відповідь спочатку розбивається по цих роздільниках, після чого кожна секція обробляється окремо.

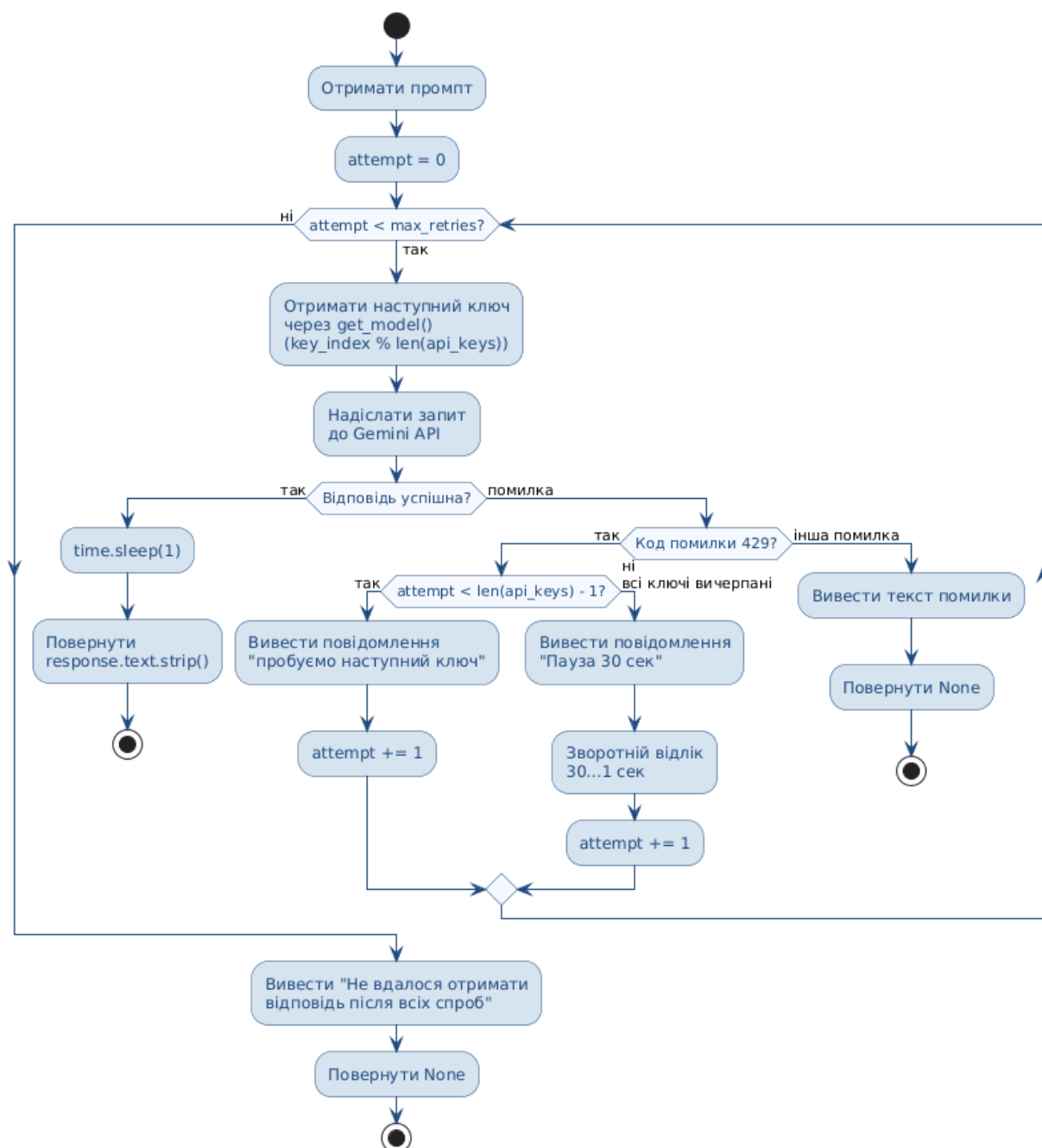


Рисунок 3.2 – Схема роботи generate_with_retry

Для задач витягування інформації перевіряється мінімальна довжина отриманого тексту: якщо модель повернула порожній рядок або службове слово, система інтерпретує це як сигнал про відсутність потрібної інформації у переданому фрагменті і переходить до альтернативного сценарію де пропонує користувачу ввести дані вручну. Для задач генерації тексту відповідь приймається як є після виклику методу strip для видалення зайвих пробілів і переносів рядків на початку та в кінці. В усіх блоках взаємодія з моделлю обгорнута в конструкцію try-

ехсерт, що дозволяє перехопити будь-який непередбачений виняток, вивести інформативне повідомлення про помилку і продовжити роботу програми у ручному режимі введення даних.

3.4 Формування та експорт силабусу освітнього компонента

Завершальним етапом роботи системи є збирання всіх сформованих фрагментів у цілісний документ та його збереження у форматі, придатному для подальшого використання викладачем. Цей етап охоплює як технічну сторону (структурування даних і генерацію файлу), так і змістову, що полягає у забезпеченні відповідності результату встановленим вимогам до оформлення силабусу.

Структурування згенерованого матеріалу відбувається поступово протягом усього сеансу роботи з програмою. Кожен підтверджений користувачем фрагмент додається до списку `final_syllabus_lines` у вигляді рядка, де назва пункту відокремлена від його змісту символом переносу рядка. Такий формат зберігання є навмисно простим, бо він не нав'язує жодної складної структури даних на проміжному етапі, що робить список стійким до непередбачуваного порядку заповнення пунктів і спрощує додавання нових розділів без змін у решті коду. Перед збереженням список перетворюється на словник, де кожен елемент розбивається на ключ і значення по першому символу переносу рядка. Саме символ перенесення рядка, а не двокрапка, використовується як роздільник з тієї причини, що значення багатьох пунктів самі містять двокрапки. Використання двокрапки як роздільника в такому разі призводило б до некоректного розбиття тексту, тоді як перенос рядка між назвою пункту і його вмістом є однозначним маркером межі.

Формування фінального документа реалізовано з підтримкою трьох форматів збереження на вибір користувача. Текстовий формат є найпростішим і містить послідовний перелік усіх пунктів силабусу, розділених порожніми рядками, з заголовком і міткою дати генерації. JSON-формат зберігає силабус у вигляді

структурованого об'єкта, де кожен пункт представлений парою ключ-значення, що робить файл придатним для подальшої програмної обробки або імпорту в інші системи. Основним форматом є документ Word. Вміст силабусу розміщується у двостовпчастій таблиці: ліва колонка містить назву пункту на кольоровому фоні з жирним форматуванням, права – відповідний текст. Рядки таблиці чергуються між двома відтінками фону, що покращує сприйняття документа при читанні. Ім'я файлу автоматично включає мітку часу у форматі рік-місяць-день-година-хвилина, що унеможливорює випадкове перезаписування попередніх версій документа при повторному запуску програми. Після збереження система відображає клікабельне посилання для завантаження файлу безпосередньо з середовища Google Colab (Додаток Б).

Перевірка відповідності зразку силабусу закладена в логіку системи не як окремий автоматизований крок, а як наскрізний принцип, що реалізується на кількох рівнях протягом усього процесу формування документа.

На першому рівні система зчитує зразок силабусу, наданий користувачем, і витягує з нього перелік фактичних полів, після чого саме цей перелік стає основою для подальшого заповнення. Таким чином структура результуючого документа повторює структуру зразка, а не є наперед заданою у коді (Додаток Б).

На другому рівні для тих пунктів, зміст яких доцільно перенести зі зразка без змін, система звертається до LLM для витягування відповідного фрагмента оригінального тексту, а не генерує його заново. Це стосується, зокрема, політики оцінювання та інших розділів, які є уніфікованими в межах кафедри чи університету (Додаток Б).

На третьому рівні кожен згенерований або витягнутий фрагмент проходить через підтвердження користувача перед додаванням до фінального списку. Викладач має можливість відхилити запропонований варіант, запросити повторну генерацію або ввести власний текст, що залишає за людиною остаточне рішення щодо відповідності кожного пункту реальним вимогам конкретної освітньої програми. Історія підтверджених і відхилених варіантів зберігається в межах сесії, що дозволяє користувачу повернутися до попередніх рішень і за потреби переглянути їх.

3.5 Тестування та аналіз результатів роботи системи

Тестування розробленої системи проводилось в умовах, наближених до реального використання, із залученням навчально-методичних документів кафедри інформаційних систем та технологій ПДАУ. Метою тестування було не лише перевірити технічну працездатність окремих модулів, а й оцінити практичну придатність результату: наскільки сформований системою силабус відповідає вимогам, які ставляться до цього документа в реальному навчальному процесі.

Опис тестових сценаріїв охоплював кілька послідовних перевірок, кожна з яких була спрямована на окремий аспект функціонування системи [41]. У першому сценарії зразок силабусу подавався у вигляді прямого посилання на PDF-файл, розміщений на офіційному сайті університету, тоді як ОПП також завантажувалась за URL. Цей сценарій відповідає найбільш поширеній ситуації, коли викладач не має файлів локально і працює виключно з відкритими ресурсами. У другому сценарії перевірялась робота системи з локально збереженим PDF-файлом силабусу, що дозволило підтвердити коректність альтернативної гілки завантаження документа. Окремо тестувалась стійкість модуля роботи з матрицями відповідності. Для перевірки обирались різні коди ОК, щоб переконатись що система коректно знаходить відповідний стовпець таблиці незалежно від його позиції. Також перевірявся сценарій перевищення квоти API – штучно вичерпувались доступні запити для перевірки коректності роботи механізму ротації ключів та автоматичного повторного надсилання запиту після витримки паузи. Приклади згенерованих силабусів представлені у додатку В.

Аналіз якості згенерованих силабусів проводився за кількома критеріями. З точки зору структурної відповідності результуючий документ містив усі пункти, передбачені зразком, у тій самій послідовності, що підтвердило коректність роботи модуля витягування переліку полів. Текстові фрагменти, згенеровані мовною моделлю (зокрема розділи про роль дисципліни у формуванні soft skills, мету вивчення та методи навчання) відзначались змістовністю і відповідали академічному стилю викладу, характерному для навчально-методичної

документації. Список рекомендованих джерел у більшості випадків містив релевантні найменування, хоча окремі позиції потребували ручного коригування з огляду на те, що мовна модель може генерувати бібліографічні описи неіснуючих видань із правдоподібними, але вигаданими вихідними даними. Ця особливість є відомим обмеженням LLM і була врахована в інтерфейсі системи через надання користувачу можливості редагувати або повністю замінювати будь-який згенерований фрагмент. Витягування компетентностей і ПР з матриць ОПП показало високу точність при умові, що PDF-файл містить текстовий шар і таблиця не є відсканованим зображенням – у такому випадку бібліотека pdfplumber коректно відновлювала структуру рядків і стовпців і система правильно визначала наявність позначки у відповідній комірці.

Порівняння з ручним способом створення силабусу виявило суттєву різницю у витратах часу та зусиль. Традиційний процес підготовки силабусу передбачає послідовне звернення до кількох джерел, з ручним перенесенням інформації та її адаптацією до конкретної дисципліни. Згідно норм робочого часу на виконання методичної роботи у ПДАУ на 2025-2026 н.р., на розроблення силабусу навчальної дисципліни, що викладається вперше, відведено 3 години на 1 кредит. А для оновлення силабусу навчальної дисципліни, що викладається не вперше – 2 години на 1 кредит. Отже, оновлення силабусу займає щонайменше 6 годин робочого часу, а розроблення – 9 годин. Реальні витрати часу залежать від досвіду виконавця і наявності готових шаблонів.

Використання розробленої системи дозволяє скоротити цей час до 50-60 хвилин, більша частина з яких витрачається на перевірку та підтвердження запропонованих варіантів, а не на самостійний пошук і введення інформації. Водночас важливо підкреслити, що система не усуває участь викладача в процесі, а змінює її характер. Замість механічного перенесення даних фахівець зосереджується на змістовній перевірці та прийнятті рішень, що відповідає більш високому рівню залучення і підвищує якість кінцевого результату. Таким чином, розроблене програмне забезпечення виступає не заміном експертизи викладача,

а інструментом, що звільняє його від рутинної складової роботи і дозволяє більше уваги приділяти змістовній частині навчально-методичної діяльності.

3.6 Оцінка економічної ефективності впровадження системи

Для отримання об'єктивних показників економічної ефективності розробленого програмного забезпечення було проведено порівняльний аналіз трудових витрат при ручному та автоматизованому способах створення силабусу, на основі якого розраховано потенційну економію ресурсів у масштабах кафедри та університету [42].

Аналіз витрат часу при ручному створенні силабусу базується існуючих нормах навантаження науково-педагогічних працівників ПДАУ та оціночних даних щодо орієнтовного розподілу часу у процесі підготовки навчально-методичної документації (таблиця 3.1).

Таблиця 3.1 – Витрати часу на ручне створення силабусу

№ з/п	Етап	Мін. час (хв)	Макс. час (хв)
1	Пошук компетентностей і ПРН в ОПП	36	36
2	Формулювання мети, soft skills, програми навчальної дисципліни	36	80
3	Визначення методів навчання і викладання	18	54
4	Опис політики навчальної дисципліни та оцінювання	18	36
5	Формування актуального списку рекомендованих джерел інформації	90	126
6	Оформлення списку джерел за ДСТУ 8302:2015	36	36
7	Розроблення додатків до силабусу	90	130
8	Оформлення документа	36	42
	Разом *	360	540

* Відповідно до існуючих норм навантаження.

Розглянемо етапи 1-6 створення силабусу, що піддаються автоматизації з використанням розробленої системи. Згідно таблиці 3.1, мінімальний час на

виконання цих етапів становить 234 хв. (майже 4 год.), а максимальний час – 368 хв. (близько 6 год.).

Середнє значення витрат часу на ручне створення одного силабусу визначається як середнє арифметичне граничних значень:

$$T_{\text{руч}} = \frac{T_{\text{min}} + T_{\text{max}}}{2}, \quad (3.1)$$

де $T_{\text{руч}}$ – середній час на ручне створення одного силабусу;

T_{min} – мінімальний час на виконання етапів 1–6 = 234 хв;

T_{max} – максимальний час на виконання етапів 1–6 = 368 хв.

Підставивши вищезазначені значення у формулу (3.1), отримаємо:

$$T_{\text{руч}} = \frac{234 + 368}{2} = 301 \text{ хв} \approx 5 \text{ год.}$$

При використанні розробленої системи витрати часу суттєво скорочуються, оскільки викладач виконує переважно функцію перевірки та підтвердження, а не самостійного пошуку й введення інформації (таблиця 3.2).

Таблиця 3.2 – Витрати часу при використанні розробленої системи

№ з/п	Етап	Час (хв)
1	Завантаження документів та налаштування	5
2	Заповнення пунктів 1-10 (підтвердження/редагування)	20
3	Автоматичне витягування компетентностей і ПРН	5
4	Перевірка згенерованих текстових розділів	15
5	Перевірка та коригування списку джерел	15
	Разом	60

Розрахунок економії трудових ресурсів проводився на основі отриманих середніх показників. Економія часу на один силабус визначається як:

$$T = T_{\text{руч}} - T_{\text{авт}}, \quad (3.2)$$

де T – економія часу (різниця);

$T_{\text{авт}}$ – час на створення силабусу з використанням автоматизованої системи.

За формулою (3.2) отримаємо:

$$T = 301 - 60 = 241 \text{ хв} \approx 4 \text{ год} .$$

Щоб розрахувати коефіцієнт скорочення трудомісткості який показує, у скільки разів зменшуються витрати часу, скористаємося формулою:

$$K_{\text{скор}} = \frac{T_{\text{руч}}}{T_{\text{авт}}} , \quad (3.3)$$

де $K_{\text{скор}}$ – коефіцієнт скорочення трудомісткості.

Після підстановки вихідних величин у формулу (3.3) отримаємо:

$$K_{\text{скор}} = \frac{301}{60} \approx 5,0 .$$

Таким чином, використання системи забезпечує п'ятиразове скорочення часу на підготовку одного силабусу. Відносна економія часу у відсотках визначається за такою формулою:

$$E_{\%} = \frac{T}{T_{\text{руч}}} \cdot 100\% , \quad (3.4)$$

де $E_{\%}$ – коефіцієнт скорочення трудомісткості.

За формулою (3.4), підставивши у відповідні місця відомі вихідні дані, отримаємо:

$$E_{\%} = \frac{241}{301} \cdot 100\% \approx 80\% .$$

Для розрахунку економії у масштабах кафедри введемо такі позначення: $N_{\text{викл}}$ – кількість викладачів на кафедрі, $N_{\text{сил}}$ – середня кількість силабусів на одного викладача на рік. При типових значеннях $N_{\text{викл}} = 12$ осіб та $N_{\text{сил}} = 5$ документів загальна кількість силабусів на рік складає:

$$N_{\text{заг}} = N_{\text{викл}} \cdot N_{\text{сил}} , \quad (3.5)$$

де $N_{\text{заг}}$ – загальна кількість силабусів, які створюються на кафедрі за рік.

Після підстановки вихідних величин у формулу (3.5) отримаємо:

За формулою (3.5) отримаємо:

$$N_{\text{заг}} = 12 \cdot 5 = 60 \text{ документів} .$$

Сукупна економія трудових витрат у людино-годинах на рік для однієї кафедри:

$$E_{\text{кафедра}} = N_{\text{заг}} \cdot T , \quad (3.6)$$

де $E_{\text{кафедра}}$ – сукупна економія трудових витрат у людино-годинах на рік для всієї кафедри, тобто скільки годин загалом заощаджують усі викладачі кафедри завдяки використанню автоматизованої системи.

Розрахунок за формулою (3.6) після підстановки відповідних параметрів дає такий результат:

$$E_{\text{кафедра}} = 60 \cdot 4 = 240 \text{ людино – годин/рік} .$$

Перерахунок у грошовий еквівалент здійснюється на основі середньої погодинної ставки науково-педагогічного працівника. Відповідно до чинних тарифних ставок для посади доцента з науковим ступенем кандидата наук, середня погодинна ставка з урахуванням надбавок становить $C_{\text{год}} = 200$ грн/год:

$$E_{\text{грн}}^{\text{кафедра}} = E_{\text{кафедра}} \cdot C_{\text{год}}, \quad (3.7)$$

де $E_{\text{грн}}^{\text{кафедра}}$ – вартісний еквівалент зекономленого часу для однієї кафедри.

Замінивши змінні у формулі (3.7) їхніми фактичними значеннями, отримаємо:

$$E_{\text{грн}}^{\text{кафедра}} = 301 \cdot 200 = 60200 \text{ грн/рік}.$$

Для оцінки економічного ефекту в масштабах університету загальна економія трудових витрат розраховується як:

$$N_{\text{грн}}^{\text{унів}} = E_{\text{грн}}^{\text{кафедра}} \cdot N_{\text{каф}}, \quad (3.8)$$

де $N_{\text{грн}}^{\text{унів}}$ – загальна річна економія фонду оплати праці в гривнях для всього університету;

$N_{\text{каф}}$ – кількість кафедр в університеті.

У масштабах університету з $N_{\text{каф}} = 28$ кафедрами річна економія фонду оплати праці становить:

$$N_{\text{грн}}^{\text{унів}} = 60200 \cdot 28 = 1685600 \text{ грн/рік}.$$

Оцінка економічної доцільності використання розробленої системи передбачає зіставлення отриманої економії з витратами на впровадження. Оскільки система реалізована на базі Google Colaboratory і використовує безкоштовний рівень Gemini API, прямі фінансові витрати на інфраструктуру є відсутніми. Єдиною статтею витрат є час на розробку системи та час на ознайомлення викладачів з її інтерфейсом. Приймаючи $T_{\text{роз}} = 120$ год при погодинній ставці розробника $C_{\text{розр}} = 300$ грн/год та $T_{\text{навч}} = 0,75$ год на одного викладача при загальній кількості $N_{\text{викл}}^{\text{унів}} = 28 \cdot 12 = 336$ осіб:

$$V_{\text{впров}} = T_{\text{розр}} \cdot C_{\text{розр}} + T_{\text{навч}} \cdot N_{\text{викл}}^{\text{унів}} \cdot C_{\text{год}}, \quad (3.9)$$

де $V_{\text{впров}}$ – загальні одноразові витрати на впровадження системи в університеті;

$T_{\text{розр}}$ – час на розробку системи;

$C_{\text{розр}}$ – погодинна ставка розробника;

$T_{\text{навч}}$ – час на ознайомлення одного викладача з інтерфейсом системи;

$N_{\text{викл}}^{\text{унів}}$ – загальна кількість викладачів в університеті.

Після підстановки чисельних значень параметрів у відповідні змінні формули (3.9), загальні одноразові витрати на впровадження становитимуть:

$$V_{\text{впров}} = 120 \cdot 300 + 0,75 \cdot 336 \cdot 200 = 36000 + 50400 = 86400 \text{ грн.}$$

Період окупності системи визначається як відношення витрат на впровадження до річної економії:

$$T_{\text{окуп}} = \frac{V_{\text{впров}}}{E_{\text{грн}}^{\text{унів}}}, \quad (3.10)$$

де $T_{\text{окуп}}$ – термін окупності системи.

За формулою (3.10) отримаємо:

$$T_{\text{окуп}} = \frac{86400}{1685600} \approx 0,05 \text{ року} \approx 0,6 \text{ місяця.}$$

Отриманий результат свідчить про те, що система повністю окупається протягом першого півроку після впровадження ще до завершення першого повного циклу формування силабусів.

Індекс рентабельності інвестицій за перший рік експлуатації визначається за формулою:

$$ROI = \frac{E_{\text{грн}}^{\text{унів}} - V_{\text{впров}}}{V_{\text{впров}}} \cdot 100\%, \quad (3.11)$$

де ROI – коефіцієнт окупності інвестицій.

Підставивши вищезазначені значення у формулу (3.11), отримаємо:

$$ROI = \frac{1685600 - 86400}{86400} \cdot 100\% \approx 1850,93\%$$

Значення ROI на рівні 1850,93% є переконливим свідченням високої економічної ефективності розробленого рішення. Сукупність прямих фінансових показників і непрямих ефектів (зниження ризику помилок при перенесенні компетентностей, уніфікація оформлення документів і вивільнення інтелектуального ресурсу викладача) дозволяє стверджувати, що впровадження системи є економічно обґрунтованим і доцільним як на рівні окремої кафедри, так і в масштабах університету в цілому.

ВИСНОВКИ

У кваліфікаційній роботі розв'язано актуальне науково-практичне завдання з автоматизації процесів формування навчально-методичної документації закладів вищої освіти шляхом проєктування та програмної реалізації системи інтелектуальної підтримки створення силабусів на основі агрегації даних та використання API моделей штучного інтелекту.

Основні результати роботи:

- досліджено структуру силабусу ОК відповідно до стандартів вищої освіти України та рекомендацій НАЗЯВО. Визначено, що ОПП та методичні рекомендації ЗВО є основними джерелами вхідних даних для силабусів, проте їх публікація у слабкоструктурованому форматі PDF ускладнює прямий машинний аналіз та потребує автоматизованої агрегації інформації;

- проведено аналіз існуючих підходів до автоматизації документообігу в освіті. Доведено доцільність інтеграції прикладних програмних інтерфейсів (API) великих мовних моделей (LLM) для обробки неструктурованих текстових масивів (матриць компетентностей, описів дисциплін та політик курсів), що забезпечує гнучкість формування семантично зв'язаного контенту;

- на основі системного аналізу предметної області визначено технічні та функціональні вимоги до інформаційної системи. Спроектовано її модульну архітектуру та схеми інформаційних потоків, що реалізують багаторівневу підготовку контекстних вікон (вилучення метаданих, токенизація, детекція списків та регулярне мапування) для оптимізації запитів до API моделей ШІ;

- засобами мови програмування Python реалізовано працездатний програмний прототип системи. Розроблено гнучкі модулі структурного парсингу PDF-документів (із використанням бібліотек PyPDF та pdfplumber) для точного відновлення табличних матриць ОПП, асинхронний модуль взаємодії з API великих мовних моделей, а також інтерфейс фінального експорту сформованого силабусу у кросплатформові формати .docx, .json та .txt;

- дослідження функціонування системи на базі нормативних документів кафедри інформаційних систем та технологій ПДАУ підтвердили високу

лінгвістичну якість згенерованих розділів. Інтегрована функція покрокової валідації та ручного коригування блоків користувачем дозволила усунути системні ризики «галюцинування» нейромереж, забезпечивши роль викладача як експерта-валідатора;

- розрахунки операційної ефективності показали, що використання розробленої інтелектуальної системи дозволяє скоротити час на розроблення або оновлення одного силабусу з 6–9 годин ручної праці до 50–60 хвилин інтелектуальної взаємодії з програмою, що вивільняє суттєвий часовий ресурс викладача для наукової та профільної діяльності.

Наукова новизна одержаних результатів полягає у подальшому розвитку підходів до інтелектуальної агрегації слабкоструктурованих даних (PDF-файлів ОПП) та їх контекстно-залежного семантичного аналізу за допомогою комбінування інструментів структурного парсингу й інженерії промтів для прикладних програмних інтерфейсів (API) великих мовних моделей під час генерації нормативної текстової документації.

Перспективи подальших досліджень пов'язані з розширенням технологічного стеку розробленої системи, зокрема:

- розробленням повнофункціонального Web-інтерфейсу на базі архітектури клієнт-сервер;
- розширенням джерел вхідних даних шляхом інтеграції модулів автоматичного зчитування з баз даних ЄДЕБО та внутрішніх університетських LMS Moodle;
- дослідженням можливості розгортання та налаштування локальних open-source моделей штучного інтелекту на внутрішніх серверах закладу освіти для підвищення автономності системи та захисту даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Термін «Силабус». *Рада (Верховна Рада України. Законодавство України. Термінологія законодавства)*: вебсайт. URL: <https://zakon.rada.gov.ua/laws/term/en-sr/63717?lang=uk> (дата звернення: 23.05.2026).
2. Про затвердження Положення про акредитацію освітніх програм, за якими здійснюється підготовка здобувачів вищої освіти : наказ Міністерства освіти і науки України від 15.05.2024 р. № 686. Зареєстровано в Мін'юсті України 04.07.2024 р. № 1013/42358. *Офіційний вісник України*. URL: <https://mon.gov.ua/npa/pro-zatverdzhennia-polozhennia-pro-akredytatsiiu-osvitnikh-program-za-iakymy-zdiisniuietsia-pidhotovka-zdobuvachiv-vyshchoi-osvity-zareiestrovany-i-u-ministerstvi-iustytzii-ukrainy-04-lypnia-2> (дата звернення: 23.05.2026).
3. Щодо рекомендацій з навчально-методичного забезпечення : лист Міністерства освіти і науки України від 09.07.2018 р. № 1/9-434. *Законодавство України* : веб-сайт. URL: <https://zakon.rada.gov.ua/rada/show/v-434729-18> (дата звернення: 23.05.2026).
4. Галич О., Аранчій В., Костенко О. та ін. Методичні рекомендації щодо розроблення силабусу навчальної дисципліни у Полтавському державному аграрному університеті. Полтава : ПДАУ, 2024. 13 с. URL: <https://www.pdau.edu.ua/sites/default/files/node/5555/metodychnirekomendaciyisylabus24.pdf> (дата звернення: 23.05.2026).
5. Термін «Освітня (освітньо-професійна, освітньо-наукова чи освітньо-творча) програма». *Законодавство України* : веб-сайт. URL: <https://zakon.rada.gov.ua/laws/term/39732> (дата звернення: 24.05.2026).
6. Бахрушин В. Компетентності і результати навчання у нових стандартах вищої освіти. *Освітня політика* : веб-сайт. URL: <http://education-ua.org/ua/articles/702-kompetentnosti-i-rezultati-navchannya-u-novikh-standartakh-vishchoji-osviti> (дата звернення: 24.05.2026).
7. PDF Reference: sixth edition, version 1.7. *Adobe*: вебсайт. URL: <https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.7old.pdf> (дата звернення: 24.04.2026)

8. Горда В.В. Автоматизація формування силабусів освітніх компонентів на основі даних ОПП у форматі PDF. *Матеріали Міжнародної науково-практичної конференції пам'яті професора Володимира Юрженка «Інтернаціоналізація технологічної, математичної та професійної освіти: досвід та перспективи»*, 14-15 квітня 2026 року. Переяслав: Ун-т Григорія Сковороди в Переяславі, 2026.

9. Горда В. Інформаційні системи як інструмент конкурентної переваги в епоху цифрової трансформації. *Матеріали науково-практичної конференції за підсумками проходження виробничих практик здобувачів вищої освіти спеціальності 126 Інформаційні системи та технології, кафедра інформаційних систем та технологій Полтавського державного аграрного університету*, 22 жовтня 2025 р. Вип. XI. Полтава: ПДАУ, 2025. С. 8-9.

10. Цифрова трансформація освіти і науки. *Міністерство освіти і науки України*: веб-сайт. URL: <https://mon.gov.ua/tag/tsifrova-transformatsiya-osviti-i-nauki?&tag=tsifrova-transformatsiya-osviti-i-nauki> (дата звернення: 24.05.2026).

11. Ahsan A. What is a University Management System? [2026 Guide]. *GR Tech* : веб-сайт. 2026. URL: <https://www.grtech.com/blog/what-is-a-university-management-system-everything-colleges-need-to-know-in-2025> (дата звернення: 24.05.2026).

12. Флегантов Л.О., Горда В.В. Формалізація методичних вимог як етап проектування системи автоматизованого створення силабусів. *Менеджмент XXI століття : глобалізаційні виклики : матеріали X Міжнародної науково-практичної конференції*, 23 квітня 2026 р. Полтава: ПДАУ, 2026. С. 513-516.

13. Що таке LMS та як підібрати LMS-систему для навчання у 2025 році. *AcademyOcean* : веб-сайт. URL: <https://academyocean.com/ua/blog/post/shho-take-lms-sistema-povnii-gid-po-viboru-platformi-dlya-navcannya-u-2025-roci-1> (дата звернення: 25.05.2026).

14. Data Merging: Process, Challenges, and Best Practices for Combining Data from Multiple Sources. *DataLadder* : веб-сайт. URL: <https://dataladder.com/merging-data-from-multiple-sources/> (дата звернення: 25.05.2026).

15. Why are template engines important? *Full Stack Python* : веб-сайт. URL: <https://www.fullstackpython.com/template-engines.html> (дата звернення: 25.05.2026).

16. Що таке обробка природної мови? *SAP Ukraine* : веб-сайт. URL: <https://www.sap.com/ukraine/resources/what-is-natural-language-processing> (дата звернення: 26.05.2026).

17. Горда В. Використання LLM-моделей для інтелектуальної підтримки розроблення навчально-методичної документації. *Матеріали XXI щорічної студентської наукової конференції «Сучасні інформаційні технології та інноваційні методики в економіці, менеджменті та бізнесі»*. Полтава: ПДАУ, 22 квітня 2026 р. Полтава: ПДАУ, 2026. С. 27-28.

18. Chacon F. AI-based curriculum evaluation in higher education. *ICERI2024 Proceedings : 17th annual International Conference of Education, Research and Innovation*, м. Севілья, 11-13 листоп. 2024 р. IATED, 2024. С. 484-492. DOI: 10.21125/iceri.2024.0216.

19. Цибульник С., Накорик В., Півторак Д. Розроблення прототипу автоматизованої системи створення супровідних документів освітнього процесу. *Вісник Національного технічного університету «ХПІ»*. Серія: Нові рішення у сучасних технологіях, 2024. № 2(20). С. 30-37. DOI: 10.20998/2413-4295.2024.02.05.

20. Su R., Li X. Modular Monolith: Is This the Trend in Software Architecture? *arXiv*. 2024. arXiv:2401.11867 [cs.SE]. DOI: 10.48550/arXiv.2401.11867. URL: <https://arxiv.org/abs/2401.11867> (дата звернення: 26.05.2026).

21. Горда В. Багатомодельний підхід до автоматичного генерування сценаріїв та прийняття рішень засобами штучного інтелекту. *Студентські роботи за науковою тематикою кафедри інформаційних систем та технологій: матеріали XXII щорічного міждисциплінарного семінару*, 25 листопада 2025 р. Полтава: ПДАУ, 2025. С. 35-37.

22. Meuschke N., Jagdale A., Spinde T., Mitrović J., Gipp B. A Benchmark of PDF Information Extraction Tools using a Multi-Task and Multi-Domain Evaluation Framework for Academic Documents. *arXiv*. 2023. arXiv:2303.09957 [cs.IR]. DOI: 10.48550/arXiv.2303.09957. URL: <https://arxiv.org/abs/2303.09957> (дата звернення: 27.05.2026).

23. Atagong S. D., Tonnang H., Senagi K., Wamalwa M., Agboka K. M., Odindi J. A review on knowledge and information extraction from PDF documents and storage

approaches. *Frontiers in Artificial Intelligence*, 2025. Vol. 8. Art. 1466092. DOI: 10.3389/frai.2025.1466092.

24. Kasneci E., Sessler K., Küchemann S., Bannert M., Dementieva D., Fischer F. та ін. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and Individual Differences*. 2023. Vol. 103. Art. 102274. DOI: 10.1016/j.lindif.2023.102274. URL: <https://www.sciencedirect.com/science/article/abs/pii/S1041608023000195> (дата звернення: 28.05.2026).

25. Sahoo P., Singh A. K., Saha S., Jain V., Mondal S., Chadha A. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. *arXiv*. 2024. arXiv:2402.07927 [cs.AI]. DOI: 10.48550/arXiv.2402.07927. URL: <https://arxiv.org/abs/2402.07927> (дата звернення: 28.05.2026).

26. Prompt engineering: overview and guide. *Google Cloud* : веб-сайт. 2026. URL: <https://cloud.google.com/discover/what-is-prompt-engineering> (дата звернення: 28.05.2026).

27. Dynamic prompting. *GeeksforGeeks* : веб-сайт. 2025. URL: <https://www.geeksforgeeks.org/artificial-intelligence/dynamic-prompting/> (дата звернення: 29.05.2026).

28. Kanade U. LLMs Are Randomized Algorithms. *Towards Data Science* : веб-сайт. 2025. URL: <https://towardsdatascience.com/llms-are-randomized-algorithms/> (дата звернення: 29.05.2026).

29. Maity S., Deroy A. The Future of Learning in the Age of Generative AI: Automated Question Generation and Assessment with Large Language Models. *arXiv*. 2024. arXiv:2410.09576 [cs]. DOI: 10.48550/arXiv.2410.09576. URL: <https://arxiv.org/abs/2410.09576> (дата звернення: 30.05.2026).

30. Python in 2026: Why It Remains the Most In-Demand Language. *GoodSpace AI* : веб-сайт. 2025. URL: <https://goodspace.ai/blogs/167836/python-in-2025-why-it-remains-the-most-in-demand-language-1> (дата звернення: 31.05.2026).

31. Що таке Google Colab і чим він схожий на проект Jupyter? *Eitca*: веб-сайт. URL: <https://uk.eitca.org/штучний-інтелект/eitc-ai-tff-tensorflow-основи/tensorflow-в-співпраці-Google/Початок-роботи-з-Google-Colaboratory/екзаменаційний-огляд-початок-роботи-з-google-colaboratory/що-таке-google-colab-і-чим-він-схожий-на-проект-jupyter/> (дата звернення: 31.05.2026).

32. Thoma M. pypdf. *Python Package Index (PyPI)* : веб-сайт. URL: <https://pypi.org/project/pypdf/> (дата звернення: 01.06.2026).
33. Singer-Vine J., Liew C. F. pdfplumber-aemc. *Python Package Index (PyPI)* : веб-сайт. 2024. URL: <https://pypi.org/project/pdfplumber-aemc/0.7.6/> (дата звернення: 01.06.2026).
34. Reitz K. requests. *Python Package Index (PyPI)* : веб-сайт. URL: <https://pypi.org/project/requests/> (дата звернення: 01.06.2026).
35. Richardson L. beautifulsoup4. *Python Package Index (PyPI)* : веб-сайт. URL: <https://pypi.org/project/beautifulsoup4/> (дата звернення: 01.06.2026).
36. Canny S. python-docx : документація. *Read the Docs* : веб-сайт. 2025. URL: <https://python-docx.readthedocs.io/en/latest/> (дата звернення: 01.06.2026).
37. Google. google-generativeai. *Python Package Index (PyPI)* : веб-сайт. 2024. URL: <https://pypi.org/project/google-generativeai/> (дата звернення: 01.06.2026).
38. io – Core tools for working with streams. *Python 3 Documentation* : веб-сайт. URL: <https://docs.python.org/3/library/io.html> (дата звернення: 01.06.2026).
39. Raihan N., Siddiq M. L., Santos J. C. S., Zampieri M. Large Language Models in Computer Science Education: A Systematic Literature Review. *arXiv*. 2024. arXiv:2410.16349 [cs.LG]. DOI: 10.48550/arXiv.2410.16349. URL: <https://arxiv.org/abs/2410.16349> (дата звернення: 02.06.2026).
40. What is API Rate Limiting? Understanding Request Throttling and Best Practices. *Postman Blog* : веб-сайт. URL: <https://blog.postman.com/what-is-api-rate-limiting/> (дата звернення: 02.06.2026).
41. García-Méndez S., de Arriba-Pérez F., Somoza-López M. del C. A review on the use of large language models as virtual tutors. *Science & Education*. 2024. Vol. 1–16. arXiv:2405.11983. DOI: 10.1007/s11191-024-00530-2. URL: <https://arxiv.org/abs/2405.11983> (дата звернення: 02.06.2026).
42. Методичні рекомендації щодо виконання кваліфікаційної роботи для здобувачів вищої освіти Інформаційні управляючі системи спеціальності 126 Інформаційні системи та технології галузі знань 12 Інформаційні технології першого (бакалаврського) рівня вищої освіти (2022 рік набору) / О. П. Копішинська, Ю. В. Уткін, О. Д. Дивнич. Полтава: ПДАУ, 2021. 64 с.