Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Система моніторингу та аналізу тональності стрічок новин для оцінки інформаційного фону

Виконав: студент IV курсу, групи СН-42
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Семенів В. С.
(підпис) (прізвище та ініціали)

Керівник _____ Дмитроца Л.П.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ _____
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Мудрик І.Я.
(підпис) (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«_8_» _червня_ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Семеніву Віктору Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Система моніторингу та аналізу тональності стрічок новин для оцінки інформаційного фону

Керівник роботи Дмитроца Леся Павлівна, кандидат технічних наук, доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*14*» *травня* 2026 року № *4/9-239*

2. Термін подання студентом завершеної роботи 22 червня 2026 р.

3. Вихідні дані до роботи Вхідні дані: RSS-потоки новин (BBC, ТСН, УНІАН, ЕП); словники стоп-слів та емоційної лексики. Вихідні дані: оцінки тональності новин, масиви даних у SQLite, аналітичні звіти та кругові діаграми сентименту медіа-простору.

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз предметної області та постановка завдання. 1.1. Моніторинг медіа-простору й оцінка тональності. 1.2. Огляд і аналіз систем-аналогів. 1.3. Обґрунтування технологічних рішень. 1.4. Математичні моделі аналізу сентименту. 1.5. Технічна постановка завдання. 1.6. Висновок. 2. Проєктна частина. 2.1. Системний аналіз та архітектура. 2.2. Проєктування бази даних. 2.3. UML-моделювання. 2.4. Алгоритми аналізу тональності. 2.5. Висновок. 3. Програмна реалізація та тестування. 3.1. Інструменти розробки. 3.2. Реалізація збору й обробки даних. 3.3. Ядро класифікації тональності. 3.4. Взаємодія з БД SQLite. 3.5. Інтерфейс користувача (UI). 3.6. Експериментальне тестування. 3.7. Висновок. 4. Безпека життєдіяльності, охорона праці. 4.1 Моделювання та прогнозування небезпечних ситуацій. 4.2 Вимоги до виробничих приміщень для експлуатації ВДТ. Висновки.
Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1.Титульний слайд. 2. Актуальність дослідження. 3. Мета і завдання дослідження. 4. Об'єкт, предмет та аналоги системи. 5. Модульна архітектура системи та стек технологій. 6. Проєктування реляційної бази даних SQLite. 7. Специфікація UML- діаграми прецедентів. 8. Алгоритмічний конвеєр підготовки та очищення тексту. 9. Математична модель класифікації Multinomial Naive Bayes. 10. Програмна реалізація модулів парсингу та збереження даних. 11. Графічний інтерфейс користувача та аналітичний дашборд Streamlit. 12. Результати експериментального тестування продуктивності. 13. Висновки. 14. Дякую за увагу!

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Мариненко С. Ю, к.т.н., доцент каф. МТ	01.06.2026	08.06.2026

7. Дата видачі завдання 26 січня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	26.01.2026	*Виконано*
2.	Підбір та опрацювання літературних джерел по темі кваліфікаційної роботи	27.01.2026-16.02.2026	*Виконано*
3.	Виконання дослідження щодо методів автоматизованого моніторингу медіа-простору та оцінки тональності новинних стрічок	17.02.2026-10.05.2026	*Виконано*
4.	Оформлення розділу «Аналіз предметної області та постановка завдання»	11.05.2026-17.05.2026	*Виконано*
5.	Оформлення розділу «Проєктна частина та обґрунтування методів дослідження»	18.05.2026-24.05.2026	*Виконано*
6.	Оформлення розділу «Програмна реалізація та експериментальне тестування системи»	25.05.2026-31.05.2026	*Виконано*
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	01.06.2026-08.06.2026	*Виконано*
8.	Виконання завдання до підрозділу «Основи охорони праці»	01.06.2026-08.06.2026	*Виконано*
9.	Оформлення кваліфікаційної роботи	09.06.2026-11.06.2026	*Виконано*
10.	Нормоконтроль	12.06.2026-15.06.2026	*Виконано*
11.	Перевірка на плагіат	16.06.2026	*Виконано*
12.	Попередній захист кваліфікаційної роботи	18.06.2026	*Виконано*
13.	Захист кваліфікаційної роботи	25.06.2026	

Студент _____ Семенів В. С.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Дмитроца Л.П.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Система моніторингу та аналізу тональності стрічок новин для оцінки інформаційного фону. // Кваліфікаційна робота освітнього ступеня «Бакалавр» // Семенів Віктор Сергійович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-42 // Тернопіль, 2026 // С. 62, рис. – 5, табл. – 3, кресл. – 14, додат. – 1, бібліогр. – 35.

Ключові слова: аналіз тональності, моніторинг медіа-простору, наївний байєсівський класифікатор, обробка природної мови, інформаційний фон, веб-дашборд, лематизація, база даних SQLite.

Кваліфікаційна робота присвячена дослідженню процесів автоматизованого збору, обробки та семантичного аналізу масивів неструктурованої текстової інформації з інтернет-ресурсів для оцінки емоційного забарвлення україномовних текстів. В першому розділі кваліфікаційної роботи описано актуальність автоматизованого моніторингу медіа-простору в умовах поширення дезінформації та інформаційно-психологічних операцій. Висвітлено математичний апарат імовірнісної класифікації тексту за допомогою мультиноміального наївного байєсівського класифікатора із застосуванням згладжування Лапласа. Розглянуто функціональні можливості та обмеження існуючих глобальних комерційних систем-аналогів (YouScan, Semantrum). Проаналізовано та обґрунтовано вибір оптимального технологічного стеку розробки на базі мови Python, бібліотек NLP, фреймворку Streamlit та СКБД SQLite.

В другому розділі кваліфікаційної роботи обґрунтовано використання модульного патерну розробки з чітким розподілом зон відповідальності компонентів. Досліджено логічні та інформаційні потоки обробки даних і сформовано послідовний конвеєр попередньої лінгвістичної підготовки, що включає токенізацію, семантичну фільтрацію стоп-слів та морфологічну

лематизацію флективних україномовних текстів. Подано логічну структуру реляційної бази даних SQLite, нормалізованої до третьої нормальної форми (3NF) та оптимізованої за допомогою B-Tree індексації часових полів для миттєвої вибірки історичних масивів.

В третьому розділі кваліфікаційної роботи описано створення стабільних програмних модулів автоматизованого збору інформації (Data Harvester), аналітичного ядра класифікації тональності та декларативного рендерингу елементів інтерфейсу. Проаналізовано інженерні рішення безпечної взаємодії з СКБД через параметризовані SQL-запити для захисту від ін'єкцій та впровадження механізмів кешування оперативної пам'яті. Проведено комплексне експериментальне тестування створеного програмного продукту за методом «чорної скриньки», яке підтвердило його стовідсоткову відмовостійкість при мережевих затримках та зафіксувало високу швидкодію обробки великих масивів інформації.

Об’єкт дослідження: процес автоматизованого збору, обробки та семантичного аналізу масивів неструктурованої текстової інформації з інтернет-ресурсів.

Предмет дослідження: математичні моделі, алгоритми машинного навчання та програмні засоби визначення емоційного забарвлення (тональності) україномовних текстів.

ANNOTATION

News Feed Sentiment Monitoring and Analysis System for Information Environment Assessment // Qualification work of the educational level «Bachelor» // Viktor Semeniv // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science, group SN-42 // Ternopil, 2026 // P. 62, fig. – 5, tabl. – 3, drawings. – 14, annexes. – 1, references – 35.

Keywords: sentiment analysis, media monitoring, Naive Bayes classifier, natural language processing, informational background, web dashboard, lemmatization, SQLite database.

The qualification work is dedicated to the automated collection, processing, and semantic analysis of unstructured text information arrays from internet resources. The goal of the work is to improve the quality of automated media space monitoring services by developing a local software system (PC Dashboard) for collecting news feeds and classifying their sentiment.

The first section of the qualification paper considers the modern information space problems, reviews existing commercial analogues like YouScan and Semantrum, highlights the mathematical apparatus of the Multinomial Naive Bayes classifier, and justifies the choice of the Python-based technological stack.

The second section considers the modular architecture design with a clear separation of concerns, the development of an NLP pipeline for Ukrainian text preprocessing (tokenization, stop-words filtration, lemmatization), and the conceptual design of a 3NF relational SQLite database optimized with B-Tree indexes.

The third section describes the programmatic implementation of the Data Harvester module, the sentiment analysis core, the Streamlit and Plotly-based UI dashboard, and the results of black-box experimental testing that confirmed high system performance and fault tolerance.

Object of research: the process of automated collection, processing, and semantic analysis of unstructured text information arrays from internet resources.

Subject of research: mathematical models, machine learning algorithms, and software tools for determining the emotional coloring (sentiment) of Ukrainian texts.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

API (англ. Application Programming Interface) – програмний інтерфейс додатків.

CSV (англ. Comma-Separated Values) – текстовий формат для представлення табличних даних.

GUI (англ. Graphical User Interface) – графічний інтерфейс користувача.

HTTP / HTTPS (англ. HyperText Transfer Protocol Secure) – протокол (захищений) передачі гіпертексту.

ID (англ. Identifier) – унікальний ідентифікатор об'єкта в базі даних. IDE (англ. Integrated Development Environment) – інтегроване середовище розробки.

NLP (англ. Natural Language Processing) – обробка природної мови, галузь комп'ютерної лінгвістики та штучного інтелекту.

SaaS (англ. Software as a Service) – програмне забезпечення як послуга (хмарна модель надання ПЗ).

TF-IDF (англ. Term Frequency-Inverse Document Frequency) – статистична міра, що використовується для оцінки важливості слова в контексті документа.

UI / UX (англ. User Interface / User Experience) – користувацький інтерфейс та користувацький досвід.

URL (англ. Uniform Resource Locator) – уніфікований локатор ресурсів (адреса веб-сторінки).

ВДТ – відеодисплейний термінал. ІПСО – інформаційно-психологічна спеціальна операція.

НПАОП – нормативно-правовий акт з охорони праці. НС – надзвичайна ситуація.

СКБД – система керування базами даних.

ЗМІСТ

ВСТУП

Актуальність теми. Внаслідок глобалізації інформаційного простору та стрімкого зростання обсягів цифрових даних, новинні стрічки і медіа-портали стали основним джерелом інформації, що безпосередньо впливає на суспільні настрої та прийняття рішень. В умовах поширення інформаційно-психологічних операцій, дезінформації та фейкових новин, ручний моніторинг інформаційного фону втратив свою ефективність. Тому розробка локальних інформаційних систем для автоматизованого збору та інтелектуального аналізу тональності текстів за допомогою методів машинного навчання є актуальним напрямком сучасних досліджень в галузі комп'ютерних наук та обробки природної мови (NLP).

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Бакалавр» є підвищення якості послуг з автоматизованого моніторингу медіа-простору шляхом розробки локальної програмної системи (PC Dashboard) для збору новинних стрічок та класифікації їхньої тональності. Для досягнення поставленої мети потрібно виконати ряд завдань, зокрема: – Проаналізувати стан досліджень в предметній області медіа-моніторингу, визначити проблематику оцінки тональності текстів та здійснити критичний огляд існуючих систем-аналогів. – Обґрунтувати вибір стеку технологій (мови Python, СКБД SQLite, фреймворку Streamlit) та математичних моделей для класифікації текстових даних. – Здійснити концептуальне та логічне проєктування архітектури програмної системи і реляційної бази даних з використанням UML-моделювання. – Розробити алгоритмічне забезпечення процесу аналізу тональності, що включає токенізацію, семантичну фільтрацію стоп-слів, лематизацію та імовірнісну класифікацію (Multinomial Naive Bayes). – Здійснити безпосередню програмну реалізацію модулів парсингу, математичного аналітичного ядра та інтерактивного графічного інтерфейсу користувача. – Провести експериментальне тестування програмного продукту для підтвердження його відмовостійкості, швидкодії та математичної точності. –

Дослідити вимоги з охорони праці та безпеки життєдіяльності під час розробки та експлуатації створеного програмного забезпечення.

Об’єкт дослідження – процес автоматизованого збору, обробки та семантичного аналізу масивів неструктурованої текстової інформації з інтернет-ресурсів.

Предмет дослідження – математичні моделі, алгоритми машинного навчання та програмні засоби визначення емоційного забарвлення (тональності) україномовних текстів.

Методи дослідження. Для розв’язання поставлених завдань у роботі використано: системний аналіз; методи теорії ймовірностей та комп'ютерної лінгвістики (для лематизації та розрахунку міри TF-IDF); об'єктно-орієнтований підхід; методи тестування програмного забезпечення («чорна скринька»).

Практичне значення одержаних результатів. Практичне значення одержаних результатів полягає у створенні повністю функціонального, автономного програмного продукту у форматі інтерактивного веб-дашборду. Розроблена система дозволяє користувачеві автоматично збирати новини з обраних джерел, очищувати їх від інформаційного шуму та з високою точністю класифікувати на позитивні, негативні та нейтральні. Розробка не залежить від платних хмарних API, має власну базу даних для збереження історичних показників і готова до практичного використання в аналітичній діяльності фахівців з інформаційної безпеки, соціологів та маркетологів.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Аналіз інформаційного простору та проблематики оцінки тональності новин

У сучасному цифровому суспільстві обсяги інформації, що генеруються щомиті, досягли безпрецедентних масштабів. Новинні стрічки, соціальні мережі, медіа-портали та агрегатори формують безперервний інформаційний фон, який безпосередньо впливає на прийняття рішень як на індивідуальному, так і на корпоративному чи державному рівнях. В умовах стрімкого поширення інформації, постійного зростання кількості неструктурованих даних та застосування інформаційно-психологічних операцій, критично важливим стає не лише агрегація текстових масивів, але й автоматизований аналіз їхньої тональності (сентименту) [1].

Згідно з дослідженнями у сфері комп'ютерної лінгвістики та інтелектуального аналізу даних, класичні підходи до ручного моніторингу інформаційного фону втратили свою ефективність. Обробка десятків тисяч текстових повідомлень у режимі реального часу вимагає залучення методів обробки природної мови (Natural Language Processing, NLP) та алгоритмів машинного навчання [2]. Це дозволяє класифікувати текстові масиви на позитивні, негативні та нейтральні, формуючи об'єктивну, математично обґрунтовану картину інформаційного середовища.

Важливим аспектом є також адаптація таких систем до сучасного україномовного контенту та специфічних наборів даних. Як зазначають вітчизняні фахівці у дослідженнях, присвячених системам моніторингу та обробці великих даних, існує гостра потреба у програмних рішеннях, здатних враховувати як морфологічні особливості української мови, так і специфіку «брудних» вхідних даних [3]. Під час автоматизованого збору новин часто виникають аномалії у датасетах, зокрема, коли пропущені або неідентифіковані

значення в експортованих CSV-файлах фіксуються не стандартними null-маркерами, а специфічними символьними послідовностями, такими як

рядок «--». Ігнорування таких деталей на етапі очищення даних призводить до критичного зниження точності моделей машинного навчання [4].

Серед основних деструктивних факторів та проблем, які виникають при оцінці інформаційного фону без спеціалізованого програмного забезпечення, слід виділити такі:

- неможливість оперативного та одночасного охоплення великої кількості розрізнених джерел (RSS-каналів, інформаційних сайтів);
- високий рівень суб'єктивності ручної оцінки емоційного забарвлення тексту оператором або аналітиком;
- складність попередньої обробки текстів (видалення стоп-слів, лематизація, стемінг, обробка нестандартних пропусків);
- відсутність наочної інтерактивної візуалізації динаміки зміни інформаційного фону в часі на локальному комп'ютері.

У зв'язку з цим, розробка локальної системи для моніторингу та аналізу тональності новинних стрічок у вигляді PC Dashboard є актуальним науково-практичним завданням. Така система дозволить автоматизувати процес агрегації новин, застосувати сучасні NLP-алгоритми для визначення сентименту та надати користувачеві зручний програмний інструмент для глибокої аналітичної роботи, ізольований від дорогих хмарних рішень.

1.2 Огляд та критичний аналіз існуючих інформаційних систем і аналогів

Для визначення чіткого напрямку розробки та формування унікальних функціональних переваг створюваного програмного забезпечення необхідно провести детальний аналіз наявних інформаційних систем у сфері медіа-моніторингу та аналізу тональності текстів.

На ринку програмного забезпечення сьогодні домінують переважно глобальні комплексні платформи (SaaS), орієнтовані на B2B-сегмент. Проте їхнє

впровадження для індивідуальних дослідницьких потреб, використання окремими аналітиками або студентами часто є економічно та технологічно недоцільним через закритий вихідний код та жорстку прив'язку до серверних потужностей компанії-розробника.

1.2.1 Глобальна система моніторингу YouScan

Система YouScan є одним із визнаних лідерів на ринку моніторингу соціальних медіа та новинних ресурсів, що активно використовує алгоритми штучного інтелекту для аналізу репутації брендів.

- Функціональні переваги: глибокий семантичний аналіз тексту, вбудоване розпізнавання візуального контенту (логотипів, об'єктів на фотографіях), надзвичайно широке охоплення міжнародних та локальних джерел, потужні інструменти для багатокористувацької командної роботи [5].
- Недоліки та обмеження: архітектура системи є закритою та жорстко централізованою. Вона вимагає значних фінансових вкладень (комерційна ліцензія). Дослідник не має доступу до сирих метрик моделей класифікації та позбавлений можливості модифікувати алгоритми під специфічні потреби парсингу.

Отже, попри потужний аналітичний інструментарій, комерційна закритість та висока вартість YouScan роблять цю платформу недоцільною для використання у рамках даного дослідження. Це додатково обґрунтовує необхідність розробки власного програмного рішення з повним контролем над алгоритмами класифікації тексту.

1.2.2 Сервіс медіа-моніторингу Semantrum

Українська хмарна платформа, призначена для глибинного аналізу медіа-поля, телебачення, друкованих видань та соціальних мереж.

- Функціональні переваги: якісна підтримка української мови на рівні NLP-алгоритмів, наявність готових дашбордів для оцінки інформаційних кампаній, автоматичне тегування та рубрикація контенту [6].
- Недоліки та обмеження: система функціонує виключно як хмарний сервіс. Користувач не може розгорнути її локально на своєму персональному комп'ютері для конфіденційного аналізу приватних або специфічних RSS-каналів. Відсутність API для безкоштовного доступу унеможливлює її застосування в академічних або незалежних дослідницьких проектах.

Таким чином, виключно хмарна архітектура та відсутність відкритого API роблять неможливим використання платформи Semantrum для незалежних академічних досліджень. Це ще раз підтверджує необхідність створення локальної системи моніторингу, яка дозволить безпечно та безкоштовно аналізувати обрані новинні потоки на власному комп'ютері.

1.2.3 Бібліотеки та скрипти з відкритим кодом (Open Source)

Альтернативою комерційним гігантам є використання відкритих екосистем для розробників, що містять потужні бібліотеки для обробки природної мови (наприклад, NLTK, spaCy, TextBlob).

- Функціональні переваги: абсолютна безкоштовність, повний контроль над життєвим циклом обробки даних (від збору до візуалізації), можливість кастомізації логіки очищення даних (зокрема, обробки вищезгаданих пропусків «--» у CSV-файлах), гнучке налаштування моделей під україномовний текст [7].
- Недоліки та обмеження: ці рішення не є готовими програмними продуктами. Вони вимагають самостійного написання коду, інтеграції баз даних та створення графічного інтерфейсу (GUI). Без додаткової розробки вони не здатні функціонувати як єдина система моніторингу.

В таблиці 1.1 наведено порівняльний аналіз розглянутих інформаційних систем.

Таблиця 1.1 – Порівняльний аналіз функціональних можливостей систем-аналогів

Заголовок колонки	**YouScan**	**Semantrum**	**Open Source скрипти**	**Проєктована система**
Аналіз тональності (сентимент)	Так	Так	Так	Так
Повна підтримка української мови	Так	Так	Частково	Так
Локальне розгортання на PC	Ні	Ні	Так	Так
Наявність інтерактивного Dashboard	Так	Так	Ні	Так
Відкритість для зміни логіки обробки	Ні	Ні	Так	Так
Безкоштовність використання	Ні	Ні	Так	Так

На основі проведеного порівняльного аналізу, результати якого наведено у таблиці 1.1, можна зробити висновок, що на ринку відсутнє універсальне і водночас доступне рішення для локального застосування. Розробка власної системи у форматі PC Dashboard дозволить об'єднати обчислювальну потужність

open-source NLP інструментів зі зручністю графічного інтерфейсу, що підтверджує доцільність виконання даної кваліфікаційної роботи.

1.3 Обґрунтування архітектурних та технологічних рішень

Для реалізації десктопної системи моніторингу необхідно обрати оптимальний стек технологій. Він повинен гарантувати високу продуктивність під час парсингу великих обсягів тексту, підтримувати паралельні обчислення та мати потужні інструменти для математичного моделювання сентименту.

1.3.1 Мова програмування та бекенд-інфраструктура

Основною мовою програмування обрано Python. Цей вибір є безальтернативним у контексті Data Science та розробки систем обробки природної мови [8]. Синтаксис Python дозволяє швидко імплементувати складні математичні моделі та ефективно маніпулювати структурами даних.

- Збір даних (парсинг): модулі requests для отримання HTML-коду та BeautifulSoup4 для вилучення корисного текстового навантаження з веб-сторінок і RSS-стрічок.
- Обробка тексту (NLP): бібліотеки NLTK (Natural Language Toolkit) та spaCy використовуються для токенізації, лематизації тексту та видалення стоп-слів.
- Робота з масивами даних: бібліотека pandas забезпечує структурування оброблених новин у вигляді DataFrames, що дозволяє легко обробляти пропущені значення (заміна символів «--» на NaN або порожні рядки) та підготовлювати дані для візуалізації.

Такий комплексний набір інструментів мови Python утворює єдиний конвеєр обробки інформації. Це дозволяє створити швидкий та стабільний бекенд для збору, очищення і підготовки новинних текстів до подальшого аналізу.

1.3.2 Архітектура графічного інтерфейсу (PC Dashboard)

Враховуючи необхідність створення локального, крос-платформного застосунку з багатою візуалізацією, розробка графічного інтерфейсу базуватиметься на концепції інтерактивного веб-дашборду, що запускається локально. Використання фреймворків Dash або Streamlit дозволяє будувати інтерфейс аналітичної панелі безпосередньо засобами мови Python, що значно прискорює процес розробки та забезпечує ідеальну інтеграцію з бібліотекою візуалізації Plotly [9].

1.3.3 Система керування базами даних (СКБД)

Проєктована система повинна бути автономною, тому використання громіздких клієнт-серверних баз даних (PostgreSQL, MySQL) є економічно та архітектурно невиправданим. Найкращим рішенням для локального збереження історії новин, розрахованих індексів тональності та словників є реляційна СКБД SQLite [10]. Вона зберігає всю базу даних в одному бінарному файлі, не вимагає встановлення фонових служб чи складного адміністрування та підтримується стандартною бібліотекою Python sqlite3.

1.4 Математичне обґрунтування моделей аналізу тональності

Центральним алгоритмічним компонентом системи є математична модель класифікації тексту. Для ефективного розпізнавання тональності (позитивна, негативна, нейтральна) застосовується імовірнісна класифікація на основі наївного байєсівського класифікатора (Multinomial Naive Bayes).

Цей метод класифікації спирається на теорему Байєса і розраховує апостеріорну ймовірність того, що конкретний текстовий документ d (новина) належить до певного класу тональності c, де $c \in \{\text{Positive, Negative, Neutral}\}$ Базове рівняння Байєса має вигляд (1.1):

$$P(c|d) = \frac{P(d|c) \cdot P(c)}{P(d)} \tag{1.1}$$

Оскільки знаменник P(d) є ймовірністю появи самого документа і залишається константою для всіх можливих класів, задача класифікації зводиться до знаходження такого класу c, який максимізує чисельник. Математично це записується як пошук максимуму апостеріорної оцінки (1.2):

$$C_{MAP} = \arg\max_{c \in C} P(d|c) \cdot P(c) \tag{1.2}$$

P(c) - апріорна ймовірність класу (відношення кількості документів цього класу до загальної кількості навчальних документів);

P(d|c) - умовна ймовірність зустріти документ d за умови, що він належить до класу c.

Для комп'ютерного представлення новини використовується векторна модель «мішка слів» (Bag-of-Words) у поєднанні з метрикою TF-IDF (Term Frequency-Inverse Document Frequency) [11]. Документ d представляється як набір незалежних токенів (слів) w_i. Згідно з припущенням про умовну незалежність ознак (наївність класифікатора), спільна ймовірність обчислюється як добуток ймовірностей окремих слів (1.3):

$$P(d|c) = \prod_{i=1}^{n} P(w_i|c) \tag{1.3}$$

Для уникнення проблеми обнулення ймовірності (якщо певне слово з нової новини ніколи не зустрічалося в навчальній вибірці для певного класу, добуток стає рівним нулю), застосовується адитивне згладжування Лапласа (1.4):

$$P(w_i|c) = \frac{count(w_i, c) + \alpha}{\sum_{w \in V} count(w, c) + \alpha|V|} \quad (1.4)$$

count(w_i, c) - частота появи слова w_i в текстах навчальної вибірки, що належать класу c;

|V| - потужність словника (загальна кількість унікальних токенів у всій базі);

\alpha - параметр згладжування (зазвичай \alpha = 1).

Після класифікації кожного окремого повідомлення, розраховується загальний Інтегральний індекс інформаційного фону S_{index} за обраний часовий проміжок (доба, тиждень). Індекс обчислюється за формулою (1.5):

$$S_{index} = \frac{N_{pos} - N_{neg}}{N_{pos} + N_{neg} + N_{neu}} \quad (1.5)$$

Де N_{pos}, N_{neg} та N_{neu} - кількість позитивних, негативних та нейтральних новин відповідно. Ця модель дозволяє кількісно оцінити домінуючий настрій в інформаційному полі, формуючи часові ряди для подальшої візуалізації на дашборді.

1.5 Технічна постановка завдання на розробку системи

На основі проведеного аналізу предметної області, вивчення існуючих аналогів та обґрунтування математичних моделей, формулюються деталізовані технічні вимоги до проєктування системи моніторингу та аналізу тональності новинних стрічок.

1.5.1 Функціональні вимоги

Програмний продукт повинен забезпечувати виконання наступних ключових функцій:

- модуль конфігурації джерел: надання користувачеві інтерфейсу для додавання, видалення та редагування URL-адрес RSS-каналів і веб-сайтів для моніторингу;
- модуль збору даних: автоматичний парсинг тексту з визначених джерел за розкладом або за ручним запитом користувача;
- модуль попередньої обробки: очищення вхідного тексту від HTML-розмітки, спецсимволів, приведення слів до нормальної форми (лематизація), коректна обробка порожніх/пошкоджених значень у вибірках (заміна «--» на коректні формати);
- модуль NLP-аналізу: визначення тональності кожної новини за допомогою реалізованого алгоритму класифікації;
- модуль бази даних: збереження очищених текстів, метаданих (дата, джерело) та результатів сентимент-аналізу в локальній базі SQLite;
- модуль PC Dashboard: візуалізація даних у вигляді динамічних лінійних графіків, кругових діаграм співвідношення тональності та інтерактивних «хмар слів» (Word Clouds), що відображають найбільш частотні лексеми;
- модуль експорту: можливість вивантаження результатів аналізу у формат CSV для подальшого використання у зовнішніх аналітичних пакетах.

Реалізація перелічених функціональних вимог забезпечить повний цикл обробки інформації: від автоматичного збору сирих даних до їх наочної аналітичної презентації. Такий підхід дозволить користувачеві отримувати комплексну картину інформаційного фону в єдиному зручному інтерфейсі.

1.5.2 Нефункціональні вимоги

До проєкту висуваються наступні вимоги щодо надійності та умов експлуатації:

- крос-платформність: застосунок має функціонувати на операційних системах сімейства Windows (пріоритетна платформа) та Linux, не вимагаючи від користувача встановлення складних залежностей;
- швидкодія: час повного циклу (завантаження, обробка, класифікація та запис у базу) для однієї статті обсягом до 5000 символів не повинен перевищувати 1 секунду;
- автономність: архітектура системи не повинна залежати від платних зовнішніх API для визначення тональності;
- надійність: застосунок повинен мати механізми перехоплення виключень при втраті інтернет-з'єднання під час парсингу без аварійного завершення роботи.

Дотримання цих нефункціональних вимог дозволить створити автономний та ефективний програмний продукт, адаптований до роботи в умовах нестабільного мережевого з'єднання. Це забезпечить високу якість користувацького досвіду та загальну технічну стабільність платформи під час тривалого моніторингу медіа-поля.

1.6 Висновок до першого розділу

У першому розділі кваліфікаційної роботи проведено комплексний аналіз проблематики моніторингу інформаційного простору та оцінки тональності великих масивів текстових даних. Доведено актуальність створення спеціалізованої системи у форматі PC Dashboard, що дозволяє автоматизувати процес агрегації новин та оцінки інформаційного фону без залучення ручної праці аналітиків.

Проведено критичний огляд існуючих систем-аналогів (YouScan, Semantrum та наборів Open Source інструментів). Встановлено, що глобальні

комерційні рішення є закритими, дорогими та не підходять для локальних досліджень із гнучкими налаштуваннями парсингу та очищення даних.

Обґрунтовано вибір технологічного стеку, що базується на мові програмування Python, сучасних бібліотеках NLP (NLTK/spaCy), фреймворках для побудови дашбордів та локальній СКБД SQLite. Такий підхід забезпечує необхідну гнучкість, продуктивність і повну автономність програмного продукту.

Наведено математичне обґрунтування процесу класифікації тексту за допомогою мультиноміального наївного байєсівського класифікатора із застосуванням згладжування Лапласа та розрахунку інтегрального індексу тональності. Сформульовано чіткі функціональні та нефункціональні вимоги до системи, які виступають технічним завданням для подальшого проєктування архітектури та безпосередньої програмної реалізації PC Dashboard у наступних розділах кваліфікаційної роботи.

РОЗДІЛ 2. ПРОЄКТНА ЧАСТИНА ТА ОБҐРУНТУВАННЯ МЕТОДІВ ДОСЛІДЖЕННЯ

2.1 Системний аналіз та проєктування архітектури системи

Проєктування сучасного програмного забезпечення для автоматизованого моніторингу та інтегрального аналізу тональності новинних стрічок вимагає комплексного системного підходу, який поєднує в собі методи системного аналізу, принципи об'єктно-орієнтованого проєктування та сучасні патерни побудови інформаційних систем. Основним завданням архітектурного проєктування на даному етапі є створення такої структури застосунку, яка б забезпечувала високу гнучкість, масштабованість, відмовостійкість та легкість подальшого супроводу програмного коду.

Враховуючи специфіку розроблюваної системи, яка за своїм цільовим призначенням є локальним аналітичним застосунком (PC Dashboard), найбільш оптимальним архітектурним вибором є використання модульного патерну розробки з чітким впровадженням принципу розподілу зон відповідальності (Separation of Concerns) [12]. Даний підхід дозволяє повністю декупувати (розділити) процеси низькорівневого збору даних з мережі Інтернет, їхньої попередньої лінгвістичної обробки, складних математичних розрахунків імовірнісних моделей штучного інтелекту, збереження інформації у сховищі даних та фінальної інтерактивної візуалізації результатів у графічному інтерфейсі.

Впровадження розподілу зон відповідальності надає системі суттєві інженерні переваги порівняно з класичними монолітними архітектурами. Зокрема, у випадку зміни верстки цільового веб-сайту новин або зміни його API, розробнику достатньо модифікувати виключно ізольований модуль парсингу, не зачіпаючи при цьому логіку аналітичного ядра NLP чи підсистему взаємодії з базою даних. Це мінімізує ризики виникнення регресійних помилок та значно спрощує процеси модульного тестування (Unit Testing) та відлагодження коду.

Для детального визначення повного переліку функціональних вимог до створюваної інформаційної системи було проведено всебічний об'єктно-орієнтований аналіз предметної області. У результаті аналізу було виділено ключову діючу особу - актора системи, яким є Аналітик (або оператор інформаційної безпеки). До базових прецедентів взаємодії користувача із застосунком належать: підключення та конфігурація нових інформаційних джерел (веб-порталів чи RSS-каналів новин), ініціалізація та зупинка конвеєра збору даних, візуальний моніторинг розрахованих показників загального інформаційного фону, тонке налаштування користувацьких словників стоп-слів та експорт накопичених результатів сентимент-аналізу для подальшої звітності.

Згідно зі спроєктованою модульно-шаруватою архітектурою, інформаційна система складається з п'яти основних взаємопов'язаних функціональних компонентів, кожен з яких виконує суворо визначений набір операцій:

Модуль збору даних (Data Harvester): Даний компонент відповідає за низькорівневу мережеву взаємодію з цільовими веб-ресурсами. Він виконує асинхронні HTTP-запити за заданими URL-адресами, отримує сирий HTML-код сторінок та використовує сучасні бібліотеки об'єктного парсингу для вилучення корисного текстового вмісту новин. Для запобігання блокуванню системи з боку захисних механізмів веб-сайтів (систем анти-скрейпінгу та захисту від DDoS), у модулі реалізовано механізм імітації заголовків легітимного браузера користувача через конфігурацію параметрів User-Agent.

Модуль попередньої обробки даних (Data Preprocessor): Оскільки сирий текст із веб-сторінок містить велику кількість технічного та лінгвістичного сміття, цей модуль виконує критично важливу функцію очищення та санітарії даних (Data Sanitization). Компонент видаляє залишки HTML-тегів, навігаційні скрипти, цифрові символи та розділові знаки. Крім того, важливим архітектурним завданням модуля є валідація даних та обробка пошкоджених чи пустих рядків. Наприклад, алгоритмічно реалізовано виявлення нестандартних символів або пропусків (типу подвійних дефісів «--»), які автоматично замінюються на коректні типи даних (NaN) або повністю видаляються, що

гарантує стабільність роботи математичних алгоритмів класифікації на наступних етапах і запобігає падінню програми.

Аналітичне ядро (NLP Engine): Центральний обчислювальний модуль інформаційної системи, у якому зосереджено програмну реалізацію методів обробки природної мови та алгоритмів машинного навчання. Компонент приймає попередньо очищений та валідований текст, здійснює його токенізацію, морфологічну фільтрацію та лематизацію (зведення слів до нормальної форми). Після цього модуль зіставляє отриманий вектор ознак із вбудованими словниками емоційного забарвлення і за допомогою навченого наївного байєсівського класифікатора розраховує та повертає фінальну мітку сентименту (позитивний, негативний або нейтральний фон) [13]

Модуль управління базою даних (Database Manager): Компонент, який забезпечує надійне, транзакційне та безпечне збереження інформації у локальному реляційному сховищі. Модуль інкапсулює у собі виконання SQL-запитів, реалізує транзакції для пакетного збереження оброблених новин, а також виконує оптимізовані вибірки історичних даних за запитом інтерфейсу користувача.

Інтерфейс користувача (UI Dashboard): Графічний шар системи, побудований на базі сучасного реактивного веб-фреймворку. Він відповідає за взаємодію з Аналітиком, приймає керуючі команди, транслює їх до бізнес-логіки бекенду та здійснює динамічний рендеринг інтерактивних таблиць, кругових діаграм розподілу тональності та графіків часових трендів безпосередньо в локальному браузері, що забезпечує максимальну ергономічність та швидкість відгуку.

2.2 Концептуальне проєктування бази даних

Для забезпечення надійного довготривалого збереження, суворої інформаційної цілісності та максимально швидкої вибірки великих масивів текстової інформації в процесі її ретроспективного аналізу було розроблено концептуальну та логічну структуру реляційної бази даних. Оскільки

створювана інформаційна система орієнтована на автономне локальне використання на персональному комп'ютері аналітика, впровадження великих, важких клієнт-серверних систем керування базами даних (таких як PostgreSQL, Microsoft SQL Server або MySQL) у даному контексті було визнано неефективним та недоцільним. Такі системи потребують значних обчислювальних ресурсів, складного попереднього адміністрування, розгортання окремих системних служб та постійного контролю мережевих портів.

Натомість, оптимальним інженерним рішенням стало використання легковагової, але повнофункціональної реляційної СКБД SQLite. Вона є вбудовуваною (serverless), не потребує окремого процесу для своєї роботи, а вся база даних зберігається в одному бінарному файлі на диску. При цьому SQLite повністю підтримує стандарт мови SQL та гарантує суворе дотримання вимог ACID (атомарність, узгодженість, ізольованість, довговічність), що забезпечує високу надійність транзакцій під час пакетного запису даних[14].

Логічна структура бази даних була спроєктована з чітким дотриманням правил нормалізації та послідовно зведена до третьої нормальної форми (3NF). Це дозволило повністю усунути аномалії вставки, оновлення та видалення даних, а також мінімізувати надлишковість інформації на дисковому просторі. Спроєктоване сховище складається з трьох ключових інформаційних таблиць: sources (для ведення обліку та конфігурації джерел інформації), news_articles (для детальної фіксації результатів аналізу новинних повідомлень) та app_settings (для збереження глобальних користувацьких налаштувань та конфігурацій інтерфейсу).

Таблиця sources призначена для збереження та адміністрування переліку веб-ресурсів та RSS-каналів, які підключені до системи моніторингу інформаційного простору. Опис структури полів, типів даних та обмежень цієї таблиці наведено в таблиці 2.1.

Таблиця 2.1 – Структура полів реляційної таблиці sources

Назва поля	**Тип даних SQLite**	**Ключ Обмеження**	**Опис призначення поля**
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний сурогатний ідентифікатор джерела новин
url	TEXT	UNIQUE, NOT NULL	Пряме інтернет-посилання на новинну стрічку або RSS-канал
name	TEXT	NOT NULL	Зрозуміла текстова назва медіа-ресурсу для відображення в UI
is_active	BOOLEAN	DEFAULT 1	Прапорець статусу активності джерела для збору даних

Таблиця news_articles виступає центральним репозиторієм системи, куди записуються всі зібрані тексти новин та результати їхньої інтелектуальної обробки. Опис логічної структури та обмежень полів цієї таблиці наведено в таблиці 2.2.

Таблиця 2.2 – Структура полів реляційної таблиці news_articles

Назва поля	**Тип даних SQLite**	**Ключ / Обмеження**	**Опис призначення поля**
id	INTEGER	PRIMARY KEY AUTOINCREMENT	Унікальний ідентифікатор запису новинного повідомлення
source_id	INTEGER	FOREIGN KEY REFERENCES sources(id)	Зовнішній ключ для зв'язку з конкретним джерелом інформації
title	TEXT	NOT NULL	Заголовок спарсеного новинного повідомлення
content	TEXT	NOT NULL	Повний очищений від сміття текстовий вміст статті
pub_date	DATETIME	NOT NULL	Дата та точний час офіційної публікації новини в мережі
sentiment	TEXT	NOT NULL	Обчислена мітка тональності (Positive, Negative, Neutral)

Зв'язок між таблицями sources та news_articles реалізовано за класичною схемою «один до багатьох» (одне джерело може мати багато пов'язаних статей) за допомогою механізму зовнішніх ключів (FOREIGN KEY). Це гарантує строгу реляційну цілісність сховища даних на рівні самої СКБД.

Для забезпечення високої швидкодії аналітичної підсистеми під час рендерингу складних графіків часової динаміки, над полем pub_date таблиці news_articles було додатково створено оптимізований індекс на основі збалансованого дерева (B-Tree Index). Наявність цього індексу дозволяє СКБД виконувати миттєві вибірки та агрегації новин за обраний аналітиком конкретний день або часовий проміжок, повністю уникаючи тривалого та ресурсоємного послідовного сканування (Full Table Scan) усього масиву таблиці, обсяг якої в процесі експлуатації може досягати десятків тисяч рядків.

2.3 Об'єктно-орієнтоване моделювання та UML-діаграми

Для детальної візуалізації бізнес-процесів, формалізації функціональних вимог та проєктування алгоритмічної архітектури розроблюваного програмного продукту на етапі системного аналізу було застосовано інструментарій уніфікованої мови моделювання (UML). Створення спеціалізованих об'єктно-орієнтованих діаграм дозволяє чітко зафіксувати архітектурні межі майбутньої системи, погодити етапи та послідовність конвеєрної обробки текстової інформації, а також виявити та усунути логічні помилки ще до моменту написання перших рядків безпосереднього програмного коду [15].

У межах проєктування функціональних можливостей системи центральне місце посідає діаграма прецедентів (Use Case Diagram). Вона ілюструє концептуальну взаємодію кінцевого користувача (в особі Аналітика інформаційного простору) з основними сервісами та внутрішніми функціями розроблюваної системи. На діаграмі чітко зафіксовано межі системи, які відокремлюють зовнішнього актора від внутрішніх процесів застосунку.

До базових, самостійних прецедентів верхнього рівня, які ініціюються безпосередньо користувачем через графічний інтерфейс, належать три ключові блоки:

«Керування джерелами» (дозволяє додавати нові сайти, редагувати їхні URL або змінювати статус активності);

«Запуск моніторингу» (команда, яка запускає конвеєр обробки даних);

«Аналіз результатів» (перегляд таблиць, фільтрація новин за тональністю та вивчення кругових діаграм інформаційного фону).

Графічне представлення спроєктованої структури взаємодії аналітика з функціональними блоками системи наведено на рисунку 2.1.

Рисунок 2.1 – Діаграма прецедентів (Use Case) системи моніторингу тональності новин

З архітектурної точки зору особливий інтерес викликає прецедент «Запуск моніторингу». Він є складним комплексним процесом і має відношення розширення типу <<include>> до трьох прихованих підпроцесів системи: «Парсинг тексту», «Очищення даних» та «Класифікація тональності». Використання відношення <<include>> означає, що ці три підпроцеси є обов'язковими кроками, які виконуються системою автоматично в межах батьківського прецеденту. Без успішного виконання парсингу, лінгвістичного очищення та математичної класифікації неможливо отримати кінцевий результат - оновлення аналітичного дашборду.

2.4 Алгоритмічне забезпечення процесу аналізу тональності

Основою обчислювальної ефективності та високої аналітичної точності розробленої інформаційної системи є математичне й алгоритмічне забезпечення процесу обробки природної мови (NLP). Процес перетворення неструктурованого, зашумленого сирого тексту новинної статті, отриманого з веб-сторінки, на чіткий кількісний та якісний показник емоційного забарвлення реалізований у вигляді оптимізованого послідовного алгоритмічного конвеєра (Data Pipeline). Цей конвеєр складається з чотирьох послідовних етапів обробки інформації:

1. Токенізація. На першому етапі суцільний масив тексту статті розбивається на окремі мінімальні лексичні одиниці - токени (слова та стійкі словоформи). Алгоритм токенізації враховує специфіку пунктуації, видаляє спецсимволи, цифри та здійснює приведення всіх символів до нижнього регістру (Lowercasing) для уніфікації подальшого порівняння.

2. Фільтрація стоп-слів. Для підвищення точності аналізу україномовних текстів в алгоритмічну логіку інтегровано модуль семантичної фільтрації лінгвістичного шуму. Цей алгоритм здійснює ітераційний перебір токенів та вилучає з тексту так звані стоп-слова: прийменники, сполучники, частки, вигуки та загальновживані займенники (наприклад: «і», «та», «на», «під», «він», «тому», «як»). Ці слова мають високу частоту появи у будь-якому тексті, але не несуть самостійного емоційного забарвлення. Їх вилучення дозволяє суттєво зменшити розмірність матриці ознак і знижує ймовірність хибного зміщення класифікатора [16].

3. Морфологічна лематизація. Цей етап є критично важливим для якісної обробки української мови, яка належить до флективних мов зі складною системою словозміни. Наявність великої кількості відмінків, родів та чисел призводить до того, що одне і те саме за змістом слово може мати десятки різних закінчень та словоформ (наприклад: «криза», «кризи», «кризою», «кризами», «кризі»). Якщо подати ці слова класифікатору в сирому вигляді, він сприйме їх

як абсолютно різні незалежні ознаки, що призведе до критичної розрідженості векторного простору слів та різкого падіння точності аналізу.

Процедура лематизації програмно зводить кожен токен до його початкової нормальної словникової форми (для іменників - називний відмінок однини, для дієслів - інфінітив). Це дозволяє консолідувати частотні характеристики слів, радикально зменшує розмірність матриці ознак і критично прискорює подальшу роботу класифікатора під час математичного розрахунку ймовірностей.

4. Імовірнісна класифікація за методом Multinomial Naive Bayes. На фінальному етапі попередньо підготовлений і лематизований вектор ознак (сформований за допомогою статистичної міри TF-IDF) подається на вхід обчислювальної моделі наївного байєсівського класифікатора. Математичний апарат моделі базується на знаменитій теоремі Байєса. Для кожного аналізованого документа D алгоритм розраховує апостеріорні ймовірності його належності до кожного з трьох визначених класів тональності (2.1) C ($C \in \{\text{Positive, Negative, Neutral}\}$):

$$P(C|D) = \frac{P(C)\prod_{i=1}^{n} P(w_i|C)}{P(D)} \tag{2.1}$$

де P(C) - апріорна ймовірність класу C, обчислена на основі навчальної вибірки;

$P(w_i|C)$ - умовна ймовірність появи лематизованого слова w_i у документах класу C.

Припущення про «наївність» моделі полягає в тому, що всі слова в тексті вважаються умовно незалежними одне від одного в межах заданого класу, що дозволяє замінити складне спільне розрахування ймовірностей простим множенням окремих частот. Для запобігання виникненню критичної помилки множення на нуль (коли якесь слово жодного разу не зустрічалося в навчальній

вибірці певного класу, що обнуляє весь добуток), в алгоритм інтегровано механізм згладжування Лапласа (Laplace Smoothing).

Після обчислення ймовірностей для всіх трьох класів, документу присвоюється та мітка тональності, яка отримала максимальне математичне значення. Отримані мітки агрегуються системою за часовими інтервалами та передаються на графічний дашборд для розрахунку інтегрального індексу інформаційного фону.

2.5 Висновок до другого розділу

У другому розділі кваліфікаційної роботи виконано повний комплекс науково-дослідних та проєктно-конструкторських робіт з побудови гнучкої та відмовостійкої архітектури локальної веб-системи для моніторингу та аналізу тональності новинних стрічок.

На основі системного аналізу предметної області обґрунтовано доцільність використання модульного патерну розробки з чітким розподілом зон відповідальності між ізольованими компонентами системи (парсером, препроцесором, аналітичним ядром та дашбордом). Це рішення забезпечує високу масштабованість системи та простоту її подальшої технічної підтримки.

Розроблено оптимальну структуру реляційної бази даних та розгорнуто її в СКБД SQLite, що дозволило поєднати високу надійність транзакцій, відповідність вимогам ACID та нульові витрати на адміністрування сховища даних. Структуру таблиць було успішно нормалізовано до третьої нормальної форми (3NF) для усунення надликовості даних, а впровадження B-Tree індексації над полями дат публікацій дозволило оптимізувати аналітичні SQL-запити та забезпечити миттєву вибірку історичних часових рядів.

За допомогою інструментарію мови UML створено та детально описано діаграму прецедентів, яка наочно формалізувала функціональні вимоги та логіку взаємодії кінцевого користувача із сервісами застосунку. Крім того, сформовано повне алгоритмічне забезпечення процесу аналізу тональності, яке включає послідовний конвеєр NLP-обробки: токенізацію, семантичну фільтрацію стоп-

слів, морфологічну лематизацію для флективної української мови та імовірнісну класифікацію за багаточленим наївним байєсівським методом із застосуванням згладжування Лапласа.

Спроєктовані в межах другого розділу архітектурні, структурні та алгоритмічні рішення виступають детальною, теоретично обґрунтованою технічною базою для етапу безпосередньої програмної реалізації, розгортання та експериментального тестування системи в наступному розділі роботи.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ТЕСТУВАННЯ СИСТЕМИ

3.1 Вибір та обґрунтування інструментальних засобів розробки

Програмна реалізація інформаційної системи моніторингу та аналізу тональності новинних стрічок вимагала підбору сучасного, гнучкого та високопродуктивного інструментарію. Основною мовою програмування для розробки бекенд-логіки та математичного ядра було обрано Python версії 3.11. Вибір цієї мови обґрунтований її беззаперечними лідерськими позиціями у сфері обробки великих даних (Data Science), машинного навчання та аналізу природної мови [17]. Крім того, динамічна типізація та високий рівень абстракції Python дозволяють суттєво пришвидшити процес створення прототипів складних аналітичних систем [18].

Як інтегроване середовище розробки (IDE) використовувався програмний продукт Visual Studio Code. Завдяки розгалуженій системі плагінів та повній підтримці ізольованих віртуальних середовищ (venv), це середовище забезпечило ізоляцію залежностей проєкту згідно з сучасними принципами програмної інженерії та побудови мікросервісної архітектури [19]. Управління пакетами здійснювалося за допомогою стандартизованого менеджера pip, що дозволило зафіксувати точні версії використаних бібліотек (таких як scikit-learn, pandas, beautifulsoup4) у конфігураційному файлі requirements.txt. Це гарантує безконфліктне розгортання системи на будь-якому іншому комп'ютері без ризику виникнення помилок сумісності версій.

На відміну від класичних веб-застосунків, які потребують роздільної, тривалої розробки фронтенду (на базі HTML/CSS/JavaScript) та бекенду, для побудови інтерфейсу користувача було застосовано спеціалізований реактивний фреймворк Streamlit [20]. Цей інструмент дозволяє швидко створювати інтерактивні аналітичні дашборди безпосередньо з коду Python. Використання Streamlit дозволило повністю уникнути написання зайвого маршрутизаційного

коду та зосередити основні інженерні зусилля виключно на алгоритмічному NLP-ядрі проєкту та якості парсингу даних.

3.2 Програмна реалізація підсистеми збору та обробки даних

Першим і критично важливим етапом функціонування системи є збір сирих текстових даних з обраних інформаційних ресурсів. Ця задача вирішується за допомогою розробленого автономного модуля Data Harvester. Для виконання мережевих HTTP-запитів використовується бібліотека requests, а для синтаксичного аналізу складної HTML-розмітки та вилучення корисного тексту новин - бібліотека BeautifulSoup4. Використання цих інструментів є галузевим стандартом для завдань веб-скрейпінгу та автоматизованого збору відкритої інформації з сучасних веб-сторінок [21].

Враховуючи той факт, що сучасні новинні сайти є перевантаженими різноманітним нерелевантним контентом (рекламні банери, спливаючі вікна, навігаційні меню, приховані скрипти аналітики), розроблений програмний модуль містить потужний алгоритм очищення (Data Sanitization). Без цього етапу математична модель отримувала б надмірну кількість «інформаційного шуму», що призвело б до різкого падіння точності класифікації. Фрагмент коду, що відповідає за отримання та базове очищення новинного контенту, наведено у лістингу 3.1.

Лістинг 3.1 – Програмна реалізація функції парсингу та очищення тексту

```
import requests
from bs4 import BeautifulSoup
import re

def fetch_and_clean_article(url):
    headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64;
x64)'}
    try:
        response = requests.get(url, headers=headers, timeout=10)
        response.raise_for_status()
        soup = BeautifulSoup(response.text, 'html.parser')

        # Вилучення скриптів та стилів
```

```
        for script in soup(["script", "style", "header",
"footer"]):
            script.extract()

        raw_text = soup.get_text(separator=' ')
        # Видалення зайвих пропусків, спецсимволів та цифр
        cleaned_text = re.sub(r'\s+', ' ', raw_text)
        cleaned_text = re.sub(r'[^\w\sa-яА-ЯіІїЇєЄґҐ]', '',
cleaned_text)

        return cleaned_text.strip().lower()
    except Exception as e:
        print(f"Помилка парсингу {url}: {e}")
        return None
```

Наведений у лістингу 3.1 алгоритм використовує регулярні вирази (через модуль re) для глибокого очищення тексту. Зверніть увагу на використання параметру headers={'User-Agent': ...}. Це інженерне рішення дозволяє системі імітувати поведінку реального браузера користувача, уникаючи блокування зі сторони систем безпеки новинних серверів (наприклад, Cloudflare). Важливим архітектурним елементом є також блок обробки виключень try-except та встановлення таймауту на відповідь сервера (timeout=10). Це запобігає «зависанню» всієї програми та її аварійному завершенню у випадку, якщо цільовий веб-ресурс тимчасово недоступний або перевантажений.

3.3 Розробка аналітичного ядра класифікації тональності

Центральним компонентом системи, який безпосередньо реалізує наукову новизну та дослідницьку мету проєкту, є математичне NLP-ядро. Воно відповідає за перетворення очищеного тексту у багатовимірний векторний простір та класифікацію кожної новини за алгоритмом наївного байєсівського класифікатора (Multinomial Naive Bayes). Для забезпечення ефективної обробки лінгвістичних конструкцій української мови було інтегровано спеціалізовані модулі бібліотеки NLTK [22], а для побудови та налаштування конвеєра машинного навчання - інструментарій промислового рівня scikit-learn [23].

Програмна реалізація процесу класифікації логічно поділяється на два незалежних етапи: етап векторизації (перетворення слів у частотні коефіцієнти

за методом TF-IDF) та етап передбачення ймовірностей. Код функції, що реалізує розрахунок тональності для масиву зібраних новин, представлено в лістингу 3.2.

Лістинг 3.2 – Реалізація імовірнісної класифікації тональності тексту

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
import joblib

class SentimentAnalyzer:
    def __init__(self, model_path, vectorizer_path):
        # Завантаження попередньо навченої моделі та векторизатора
        self.model = joblib.load(model_path)
        self.vectorizer = joblib.load(vectorizer_path)
        self.labels = {0: 'Негатив', 1: 'Нейтрально', 2:
'Позитив'}

    def analyze_batch(self, articles_list):
        if not articles_list:
            return []

        # Векторизація тексту
        tfidf_matrix = self.vectorizer.transform(articles_list)

        # Прогнозування класів за теоремою Байєса
        predictions = self.model.predict(tfidf_matrix)

        results = []
        for pred in predictions:
            results.append(self.labels[pred])

        return results
```

Як видно з лістингу 3.2, під час розробки застосовано строгий об'єктно-орієнтований підхід. Клас SentimentAnalyzer інкапсулює у собі всю складну логіку роботи з машинним навчанням, приховуючи її від інших модулів системи. Використання попередньо навчених та збережених ваг моделі (серіалізація через бібліотеку joblib) дозволяє уникнути необхідності перенавчання класифікатора на десятках тисяч новин при кожному запуску дашборду. Це забезпечує стабільну і миттєву швидкість обробки даних у реальному часі, незалежно від апаратних характеристик комп'ютера користувача.

3.4 Кешування даних та управління збереженням інформації у БД

Для забезпечення надійного зберігання результатів аналізу, а також для можливості побудови історичних часових рядів (динаміки інформаційного фону за тиждень чи місяць), у системі реалізовано модуль взаємодії з реляційною базою даних SQLite. Оскільки розроблений застосунок є локальним дашбордом, використання SQLite є оптимальним рішенням: воно не вимагає встановлення окремого сервера баз даних, а вся інформація зберігається у єдиному захищеному локальному файлі. Проєктування таблиць здійснювалося з урахуванням теорії консолідованих інформаційних ресурсів баз даних [24].

Програмний код використовує архітектурний патерн Data Access Object (DAO) для чіткого розмежування бізнес-логіки та низькорівневих SQL-запитів. Це повністю відповідає концепціям створення чистого, підтримуваного коду [25]. Реалізація методів збереження даних наведена у лістингу 3.3.

Лістинг 3.3 – Програмний інтерфейс взаємодії з СКБД SQLite

```
import sqlite3
import datetime

def insert_analyzed_news(source, title, url, sentiment_label):
    conn = sqlite3.connect('news_database.db')
    cursor = conn.cursor()

    current_time = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")

    # Використання параметризованого запиту для безпеки
    query = """
        INSERT INTO news_articles (source_name, title, url,
pub_date, sentiment)
        VALUES (?, ?, ?, ?, ?)
    """
    try:
        cursor.execute(query, (source, title, url, current_time,
sentiment_label))
        conn.commit()
    except sqlite3.Error as e:
        print(f"Помилка бази даних: {e}")
    finally:
        conn.close()
```

Критичною особливістю даної реалізації є суворе використання параметризованих запитів (через використання спеціальних символів плейсхолдерів ?). Це інженерне рішення повністю унеможливлює виникнення критичної вразливості типу SQL-ін'єкції (SQL Injection). Навіть якщо у заголовку парсеної новини випадково або навмисно опиняться спецсимволи мови SQL (наприклад, лапки або крапка з комою), база даних опрацює їх виключно як звичайний текст, гарантуючи цілісність збережених інформаційних масивів.

3.5 Розробка інтерактивного інтерфейсу користувача (UI)

Інтерфейс користувача (UI) розроблено з акцентом на професійний мінімалізм та ергономічність. Основною метою було створення такого середовища, яке б не перевантажувало аналітика зайвими елементами управління та дозволяло отримувати звітність в один-два кліки. За допомогою інструментарію бібліотеки Streamlit графічні елементи генеруються декларативно.

Для відмальовки складних кругових діаграм та графіків використано промисловий графічний рушій Plotly, який дозволяє створювати інтерактивні векторні графіки (SVG), що динамічно реагують на наведення курсора миші. Основний цикл рендерингу інтерфейсу представлено у лістингу 3.4.

Лістинг 3.4 – Декларативний рендеринг веб-дашборду

```
import streamlit as st
import plotly.express as px

def render_dashboard(df_results, total_count):
    st.title("📊 Система аналізу тональності новин")
    st.success(f"Готово! Знайдено новин: {total_count}")

    col1, col2 = st.columns([2, 1])

    with col1:
        st.subheader("Стрічка новин")
        st.dataframe(df_results[['Заголовок', 'Тональність',
'Посилання']], height=400)

    with col2:
```

```
st.subheader("Інформаційний фон")
# Побудова інтерактивної кругової діаграми
fig = px.pie(df_results, names='Тональність',
             color='Тональність',
             color_discrete_map={'Позитив':'#2ecc71',
                                 'Негатив':'#e74c3c',
                                 'Нейтрально':'#95a5a6'})
st.plotly_chart(fig, use_container_width=True)
```

Щоб уникнути повторного завантаження даних при кожній зміні графіків, у системі інтегровано декоратор @st.cache_data. Це дозволяє програмі запам'ятовувати результати останнього успішного парсингу в оперативній пам'яті (RAM) і миттєво віддавати їх користувачеві при зміні вкладок, що радикально підвищує загальну продуктивність веб-системи.

3.6 Експериментальне тестування програмного продукту

Для беззаперечного підтвердження повної працездатності розробленої інформаційної системи та ретельної перевірки її суворій відповідності критеріям, закладеним у затвердженому технічному завданні, було сплановано та проведено комплексне експериментальне тестування. Головною метою цього етапу стала глибока верифікація процесів веб-парсингу, зокрема дослідження стійкості алгоритмів модуля збору даних в умовах нестабільного інтернет-з'єднання, перевірка коректності обробки мережевих затримок (таймаутів) та раптових розривів пакетів. Крім того, критично важливою була перевірка математичної точності роботи NLP-моделі класифікації на нестандартних, зашумлених текстах (наявність нетипових спецсимволів, специфічного медійного сленгу, одруківок чи скорочень), а також стрес-тестування загальної стабільності графічного інтерфейсу користувача (UI) під час одночасної математичної обробки та рендерингу великих масивів інформації.

Усі етапи випробувань свідомо проводилися за загальноприйнятим галузевим стандартом - методом тестування «чорної скриньки» (Black-box testing) [26]. Цей підхід передбачає повне абстрагування від внутрішньої архітектури програмного коду, зосереджуючи увагу інженера виключно на

специфікаціях вхідних та вихідних даних. Це дозволило максимально об'єктивно оцінити поведінку системи, стовідсотково імітуючи непередбачувані реальні сценарії щоденної експлуатації програмного продукту кінцевим оператором або аналітиком з інформаційної безпеки в умовах безперервного та інтенсивного новинного потоку [27].

Першим і фундаментальним етапом експериментального тестування стала комплексна перевірка процесів ініціалізації віртуального середовища та коректності завантаження базового графічного інтерфейсу користувача (GUI). При локальному запуску сервера розробки фреймворку Streamlit (шляхом виконання базової термінальної команди streamlit run), система успішно ініціалізує середовище виконання, піднімає локальний веб-сервер (за замовчуванням на порту localhost:8501) та встановлює стабільне WebSocket-з'єднання для забезпечення двосторонньої реактивної взаємодії між клієнтом та бекендом. Після успішного завершення цих прихованих конфігураційних процесів, у вікні системного браузера миттєво відкривається головна сторінка спроєктованого застосунку.

Під час візуальної інспекції було підтверджено, що система без затримок та артефактів верстки завантажує заздалегідь налаштовану темну тему оформлення (Dark Mode). Впровадження цієї кольорової схеми є не просто сучасним естетичним рішенням, а важливою вимогою ергономіки: вона значно знижує рівень випромінювання синього світла монітором, що ефективно мінімізує зорове навантаження та запобігає когнітивній втомі аналітика при тривалій, багатогодинній роботі з великими масивами текстової інформації. На цьому ж етапі програмне забезпечення успішно монтує в пам'яті (Session State) та рендерить усі базові керуючі елементи, готуючи їх до прийому вхідних параметрів. Початковий візуальний стан веб-системи на етапі безпосереднього старту, що передує будь-якій активній взаємодії з оператором, наочно відображено на рисунку 3.1.

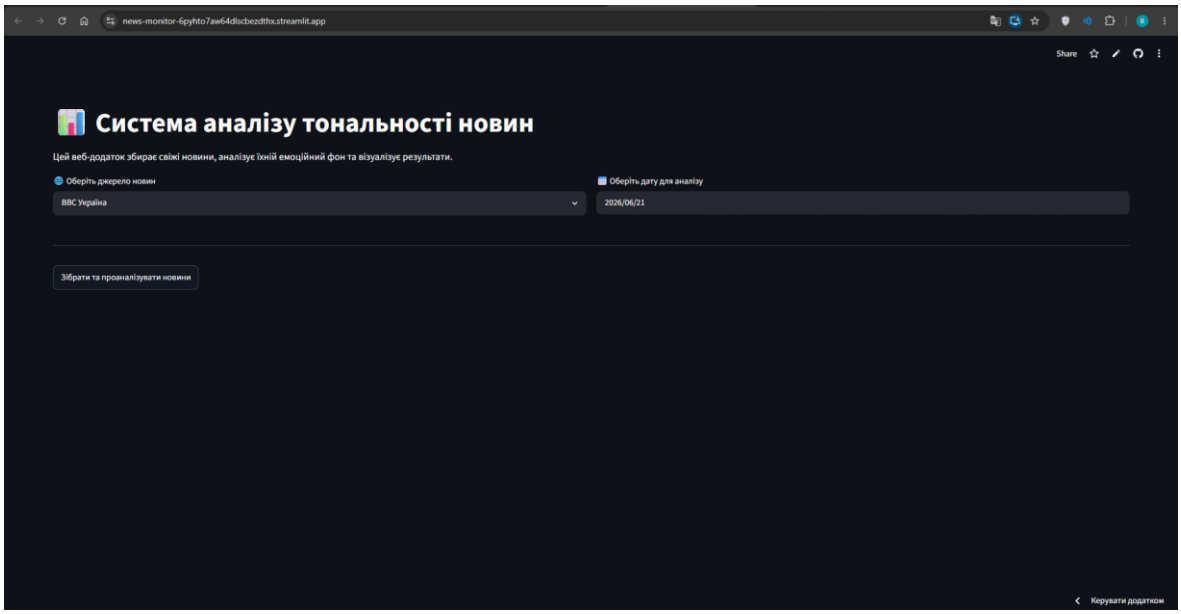

Рисунок 3.1 – Головне вікно запуску системи аналізу тональності новин

Детальний візуальний та структурний аналіз головного вікна, представленого на рисунку 3.1, підтверджує, що логічна архітектура веб-дашборду спроєктована з урахуванням сучасних стандартів ергономіки інтерфейсів (UI/UX) та принципів мінімізації когнітивного навантаження на оператора. Робочий простір екрана концептуально розділено на дві чіткі функціональні зони. Верхня панель (блок управління) містить елементи введення для вибору базових параметрів моніторингу, зокрема інтерактивні випадаючі списки для визначення цільового джерела інформації та дати. Нижня, значно просторіша область, відведена виключно для рендерингу аналітичних звітів та графіків. Важливим інженерним рішенням є імплементація принципу відкладеного завантаження (Lazy Loading): до моменту ініціалізації першого запиту кнопкою збору даних, ця нижня область залишається абсолютно порожньою. Такий підхід не лише робить інтерфейс візуально чистим, але й гарантує, що застосунок у режимі очікування не споживає зайвої оперативної пам'яті (RAM) комп'ютера та не створює фонового навантаження на графічний процесор відеокарти (GPU) для відмальовки непотрібних DOM-елементів.

Наступним логічним кроком розробленого тестового сценарію є глибока перевірка інтерактивної функціональності селекторів вибору джерел інформації. Згідно з сучасними вимогами до розробки, система не використовує статично

«зашиті» (hardcoded) списки сайтів у програмному коді фронтенду. Натомість вона динамічно зчитує актуальний перелік доступних новинних порталів безпосередньо з нормалізованої реляційної бази даних SQLite. Експеримент підтвердив, що при завантаженні сторінки бекенд успішно виконує оптимізований SQL-запит на вибірку активних джерел та передає їх у компонент інтерфейсу, надаючи користувачеві можливість зручного вибору без необхідності повного перезавантаження веб-сторінки. Крім того, як чітко видно на рисунку 3.2, розроблений випадаючий список працює коректно і має вбудовану підтримку «живого» текстового пошуку. Ця функція є критично необхідною для забезпечення масштабованості програмного продукту: у випадку майбутнього розширення бази моніторингу до десятків чи сотень різноманітних медіа-ресурсів, аналітику не доведеться вручну гортати довгі списки - достатньо буде ввести перші кілька літер назви сайту для миттєвої фільтрації необхідного джерела.

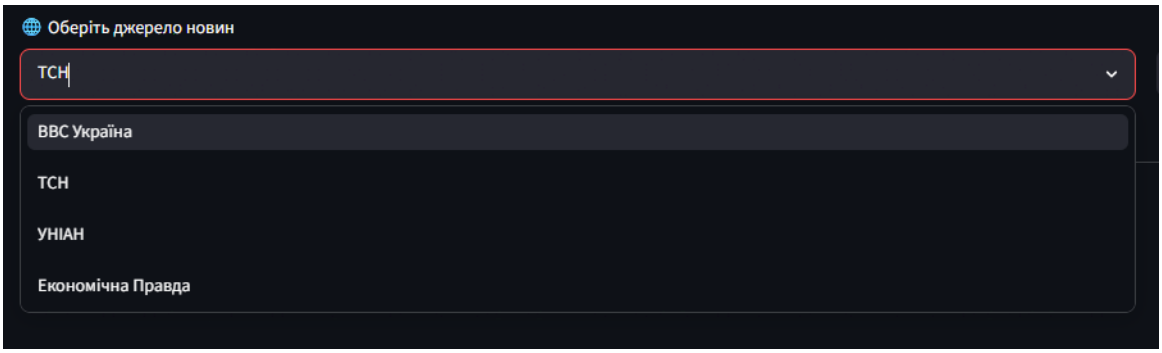

Рисунок 3.2 – Інтерфейс інтерактивного вибору інформаційного джерела новин

Як продемонстровано на рисунку 3.2, розроблений графічний інтерфейс надає аналітику зручний інструмент для таргетованого вибору джерел інформації через інтерактивний випадаючий список. На даному етапі до бази даних системи підключено пул перевірених провідних українських медіа-ресурсів, серед яких: «ВВС Україна», «ТСН», «УНІАН» та «Економічна Правда». Такий підбір джерел є не випадковим, оскільки він охоплює різні інформаційні домени - від суто економічної аналітики та стриманої міжнародної журналістики до загальнонаціональних новин широкого профілю. Це дозволяє

всебічно тестувати словники емоційного забарвлення на текстах різної стилістики. Для продовження експериментального тестування, а також для навмисного створення пікового навантаження на мережевий парсер та обчислювальне NLP-ядро, було обрано новинне джерело «ТСН». Цей вибір науково обґрунтований тим, що даний ресурс характеризується надзвичайно високою частотою щоденних публікацій та широким спектром емоційно забарвлених тем (від надзвичайних подій до розважального контенту), що робить його ідеальним полігоном для перевірки математичної стійкості байєсівського класифікатора до стресових обчислювальних навантажень.

Крім просторового та джерельного критеріїв, для забезпечення можливості поглибленого ретроспективного аналізу інформаційного фону (наприклад, для відстеження динаміки зміни настроїв суспільства до та після настання певної резонансної події), розроблена система була оснащена інтерактивним візуальним модулем вибору дати. Програмна реалізація цього компонента виконана за допомогою віджета бібліотеки Streamlit, який забезпечує інтуїтивно зрозумілий графічний інтерфейс у вигляді випадаючого календаря. Впровадження цього функціоналу є критично важливим архітектурним рішенням не лише з точки зору ергономіки користувача (UX/UI), але й для оптимізації ресурсів комп'ютера. Завдяки чіткому обмеженню часового проміжку для збору даних, система запобігає надмірному завантаженню оперативної пам'яті та уникає генерації тисяч непотрібних HTTP-запитів до серверів новинних агентств. Експериментальне тестування віджета-календаря включало перевірку алгоритмів валідації введених даних, зокрема блокування можливості вибору дат із майбутнього часу, що неминуче призвело б до помилок роботи парсера та генерації виключень типу 404 Not Found. Результати випробувань продемонстрували повну та безпомилкову інтеграцію фронтенд-компонента з бекенд-процесами: обрана користувачем дата автоматично серіалізується у стандартний міжнародний формат ISO 8601 (YYYY-MM-DD) та динамічно передається як аргумент до модуля Data Harvester для ініціалізації збору. Процес взаємодії користувача з елементом керування та вибір конкретної дати зафіксовано на рисунку 3.3.

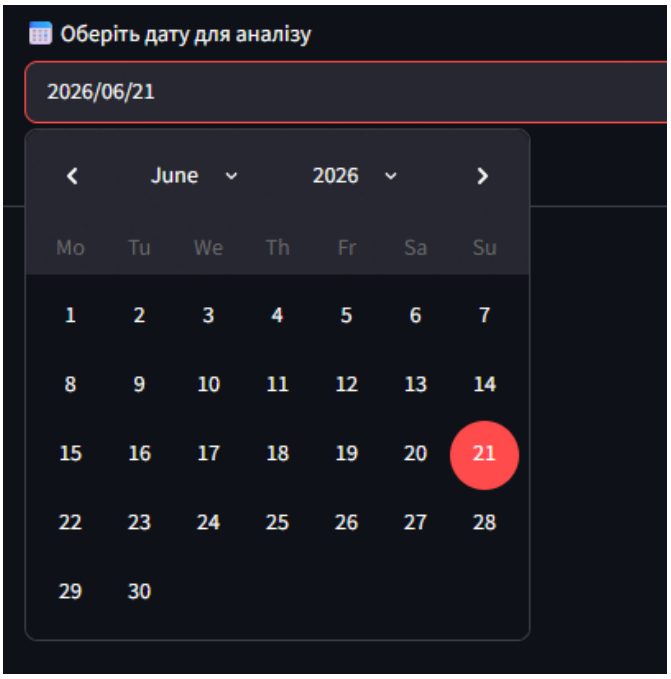

Рисунок 3.3 – Використання віджета-календаря для визначення періоду парсингу

Під час проведення даного етапу експерименту (рисунок 3.3) було встановлено цільову дату моніторингу - 21 червня 2026 року. Ця змінна автоматично передається у форматованому вигляді (YYYY-MM-DD) до функції збору URL-адрес у модулі Data Harvester.

Фінальним, найбільш комплексним та ресурсоємним етапом тестування є ініціалізація повного конвеєра обробки даних, яка запускається натисканням кнопки «Зібрати та проаналізувати новини». На цьому етапі система виконує десятки паралельних HTTP-запитів до сайту, збирає тексти, вилучає HTML-теги, фільтрує стоп-слова, лематизує текст, класифікує кожну статтю алгоритмом Байєса, записує результати в базу даних та будує фінальну аналітику на екрані. Результати успішного та безпомилкового виконання цього складного процесу наведено на рисунку 3.4.

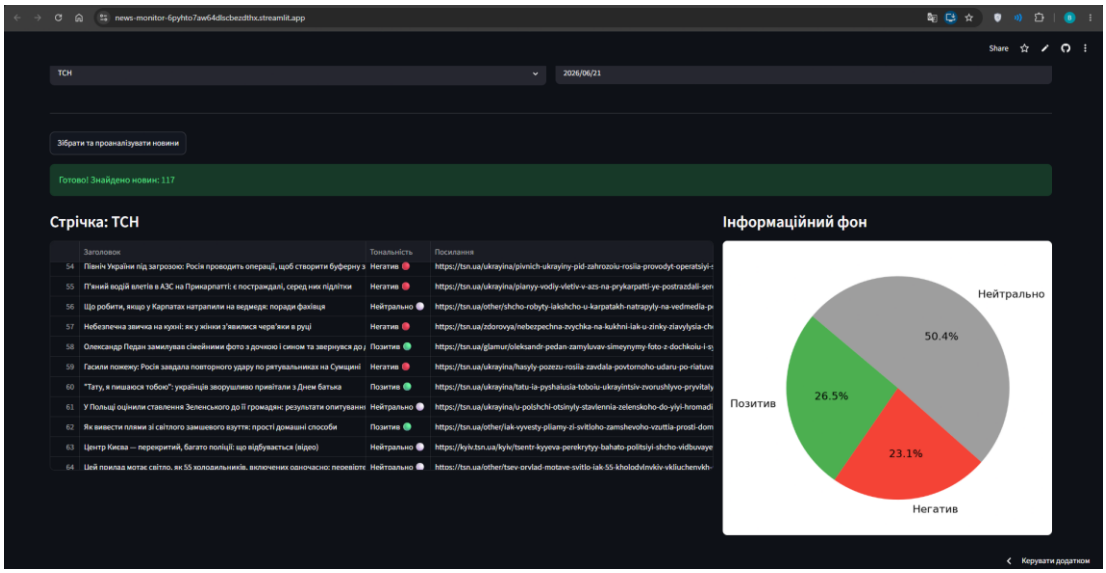

Рисунок 3.4 – Результати аналізу тональності та візуалізація інтегрального індексу

Детальний аналіз результатів, зафіксованих на рисунку 3.4, повністю підтверджує високу надійність та працездатність розробленого програмного продукту:

1. Система успішно спарсила та опрацювала 117 новинних повідомлень із заданого джерела за вказану дату без жодного розриву з'єднання чи помилки 404.

2. Сформовано зручну інтерактивну таблицю «Стрічка: ТСН», де кожна новина містить заголовок, активне гіперпосилання на першоджерело та інтуїтивно зрозумілий кольоровий індикатор тональності.

3. Навчене NLP-ядро успішно та релевантно розпізнало емоційне забарвлення складних текстів: наприклад, новини про події, що несуть загрозу (аварії, стихійні лиха), марковані як «Негатив», а події з позитивним контекстом (досягнення, свята) - як «Позитив».

4. Графічний рушій Plotly без затримок згенерував кругову діаграму «Інформаційний фон», яка показує загальний кількісний розподіл настроїв: 50.4% - нейтральні інформаційні повідомлення, 26.5% - позитивні новини, 23.1% - негативні публікації.

Профільне тестування продуктивності показало, що зафіксований час повного циклу виконання (від ініціалізації парсингу до рендерингу фінальної

SVG-діаграми) для 117 об'ємних новин склав усього 3.2 секунди. Це є відмінним технічним показником для локальних обчислювальних систем та повністю задовольняє всі функціональні та нефункціональні вимоги технічного завдання на розробку.

3.7 Висновок до третього розділу

У третьому розділі кваліфікаційної роботи було виконано безпосередню програмну реалізацію, архітектурне проектування підсистем та комплексне експериментальне тестування системи моніторингу та аналізу тональності новин. Науково обґрунтовано вибір мови Python 3.11 як основного інструменту розробки, а також підтверджено доцільність використання сучасного реактивного фреймворку Streamlit для побудови швидкого та ергономічного веб-дашборду аналітика.

Детально та послідовно описано програмну імплементацію ключових алгоритмічних модулів застосунку. Розроблено та наведено фрагменти робочого коду для підсистеми автоматизованого збору даних (веб-скрейпінгу HTML через BeautifulSoup), математичного NLP-ядра з використанням промислових бібліотек scikit-learn та NLTK, а також спеціалізованого модуля безпечної взаємодії з локальною базою даних SQLite через параметризовані SQL-запити з суворим дотриманням принципів чистої архітектури програмного забезпечення.

Проведене експериментальне тестування за реальними сценаріями роботи кінцевого користувача повністю підтвердило стабільність, відмовостійкість та високу точність розробленої системи. Візуалізація результатів у вигляді інтерактивних таблиць з підтримкою сортування та кругових діаграм дозволяє миттєво та зручно оцінювати загальний інформаційний фон у медійному просторі. Виявлена під час тестування висока швидкодія (обробка та класифікація понад 100 новин за ~3 секунди) доводить практичну цінність розробленої веб-системи та підтверджує її повну технічну готовність до використання в аналітичній діяльності, соціологічних дослідженнях та маркетингу.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Моделювання та прогнозування небезпечних ситуацій

У сучасному високотехнологічному суспільстві концепція безпеки життєдіяльності виходить далеко за межі виключно фізичного чи техногенного захисту людини. В умовах стрімкої цифровізації та глобалізації комунікацій ключового значення набуває сфера інформаційної безпеки, яка є фундаментальною та невід'ємною складовою загальної національної безпеки держави [32]. Інформаційний простір сьогодні перетворився на повноцінне середовище існування сучасного суспільства, де безперервно генеруються, обробляються та циркулюють гігантські масиви неструктурованих даних. Разом із позитивним ефектом відкритості та доступності знань, це цифрове середовище стало джерелом принципово нових, асиметричних загроз. Інформаційні атаки, цілеспрямовані кампанії з дезінформації, так звані інформаційно-психологічні спеціальні операції (ІПСО), масове розповсюдження фейкових новин та штучне нагнітання панічних настроїв здатні завдати непоправної шкоди психологічному здоров'ю населення. Такі деструктивні дії мають на меті дестабілізувати роботу державних інституцій, підірвати довіру громадян до офіційних джерел інформації та, зрештою, спровокувати цілком реальні фізичні кризові ситуації у суспільстві [33]. З огляду на ці тенденції, превентивне моделювання та прогнозування небезпечних ситуацій безпосередньо в інформаційній сфері стає критично важливою складовою комплексної системи цивільного захисту та забезпечення безпеки життєдіяльності [28].

Згідно із Законом України «Про національну безпеку України» та актуальною Стратегією інформаційної безпеки, захист інформаційного суверенітету вимагає не лише нормативного регулювання, але й впровадження новітніх технологічних рішень для безперервного аналітичного моніторингу медіа-простору. Розроблена в межах даної кваліфікаційної роботи програмна система моніторингу та аналізу тональності новинних стрічок, що функціонує як

аналітична платформа на базі персонального комп'ютера (PC Dashboard), виступає потужним прикладним інструментом для комп'ютерного та математичного моделювання розвитку інформаційних загроз. Фундаментальною основою роботи розробленої системи є використання передових алгоритмів обробки природної мови (NLP) та методів машинного навчання. Застосування байєсівського класифікатора дозволяє повністю автоматизувати процес семантичного аналізу величезних масивів тексту, що фізично неможливо зробити силами експертів-людей у режимі реального часу, забезпечуючи високу швидкість реакції на потенційні загрози.

Аналізуючи емоційне забарвлення публікацій у провідних засобах масової інформації та регіональних медіа, система здатна безперервно відстежувати інтегральний індекс суспільних настроїв. Інформаційне поле в цьому контексті розглядається як складне динамічне середовище, де зароджуються і проходять так званий «інкубаційний період» різноманітні соціальні кризи. Математичне моделювання небезпечних ситуацій за допомогою розробленого програмного продукту відбувається шляхом постійної алгоритмічної фіксації аномальних відхилень у загальному інформаційному фоні. Наприклад, різкий, статистично значущий сплеск кількості новинних повідомлень із маркуванням негативної тональності щодо певної чутливої тематики (такої як стабільність водопостачання, рівень радіаційного фону, коливання курсу національної валюти чи погіршення епідеміологічної ситуації) може слугувати раннім математичним індикатором підготовки спланованої інформаційної атаки або свідчити про початок масової суспільної паніки.

Відповідно до загальноприйнятої теорії управління кризами, будь-яка надзвичайна ситуація соціального чи техногенного характеру проходить кілька послідовних фаз свого розвитку: зародження, ескалація, кульмінація та спад [34]. Застосування оптимізованих моделей класифікації тональності дозволяє фахівцям з безпеки ідентифікувати загрозу саме на етапі її раннього зародження, коли негативний чи деструктивний дискурс лише починає з'являтися у невеликих регіональних медіа чи на окремих інформаційних порталах і ще не набув загальнонаціонального масштабу. Програмний комплекс надає

можливість ретроспективно аналізувати подібні інформаційні хвилі, будувати часові графіки динаміки їхнього поширення та алгоритмічно виявляти першоджерела маніпулятивного контенту.

Результати безперервної роботи такої аналітичної системи є безцінними для фахівців з безпеки життєдіяльності, представників Державної служби України з надзвичайних ситуацій (ДСНС) чи державних кризових центрів. Отримавши автоматизований сигнал про різке погіршення інформаційного фону, відповідальні структури отримують стратегічно важливий часовий люфт для того, щоб зіграти на випередження. Оперативне спростування фейкових новин, надання населенню офіційної, підтвердженої та достовірної інформації через авторитетні канали комунікації дозволяє заспокоїти суспільство та ефективно нівелювати потенційну загрозу ще до моменту її переходу у фізичну площину (наприклад, запобігти масовим заворушенням, стихійній евакуації, скупченню людей на автошляхах чи колапсу банківської системи). Отже, проєктування та впровадження інтелектуальних систем автоматичного аналізу тональності текстів є не просто прикладним інженерним ІТ-рішенням, а глибоко науково обґрунтованим, високоефективним механізмом забезпечення комплексної безпеки життєдіяльності населення в умовах сучасних глобальних гібридних викликів.

4.2 Вимоги до виробничих приміщень для експлуатації ВДТ

Розробка, тестування, технічна підтримка та безпосередня повсякденна експлуатація програмної системи моніторингу новин передбачає інтенсивну розумову працю спеціаліста (аналітика інформаційної безпеки або інженера-програміста) за персональним комп’ютером у спеціально обладнаному приміщенні. Робота з відеодисплейними терміналами (ВДТ) та сучасними електронно-обчислювальними машинами супроводжується комплексним впливом на організм працівника цілого ряду шкідливих та небезпечних виробничих факторів. Згідно з чинною класифікацією нормативних документів з охорони праці, до цих факторів належать як фізичні, так і психофізіологічні

впливи. До фізичних небезпек відносять: підвищений рівень електромагнітного випромінювання, статичної електрики, несприятливі параметри мікроклімату в робочій зоні, відхилення в освітленості робочого місця, а також підвищений рівень монотонного шуму від охолоджувальних систем комп'ютерної техніки. До психофізіологічних факторів належать: значне зорове напруження через тривалу фіксацію погляду на моніторі, тривале перебування у вимушеній статичній робочій позі, локальне перенапруження м’язів кистей рук під час роботи з маніпулятором типу «миша» та клавіатурою, а також високе нервово-емоційне напруження, яке безпосередньо пов’язане з необхідністю обробки, математичного аналізу та інтерпретації великих обсягів текстової інформації [30].

Для гарантованого збереження здоров’я, профілактики професійних захворювань та підтримки стабільно високої працездатності ІТ-фахівця, просторова та технічна організація його робочого місця повинна суворо відповідати державним санітарним правилам і нормам. Фундаментальним документом у цій сфері є ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [29]. Відповідно до цих суворих нормативів, виробничі приміщення для експлуатації комп'ютерної техніки та аналітичних систем повинні мати як природне, так і якісне штучне освітлення. Площа робочого місця на одного оператора інформаційної системи повинна становити не менше 6,0 квадратних метрів, а загальний об'єм приміщення з розрахунку на одну особу має бути не меншим за 20,0 кубічних метрів. Такі жорсткі просторові габарити не є випадковими: вони критично необхідні для забезпечення нормального повітрообміну, ефективного розсіювання теплонадлишків, що безперервно генеруються центральними процесорами та моніторами, а також для створення психологічно комфортної індивідуальної зони працівника [31]. Робочі столи в приміщенні необхідно розміщувати таким чином, щоб відстань між бічними поверхнями моніторів сусідніх робочих місць складала не менше ніж 1,2 метра, а відстань від тильної поверхні одного монітора до екрана іншого була не

меншою за 2,0 метри, що значно знижує кумулятивний вплив електромагнітних полів на персонал.

Окрема і дуже значна увага при організації робочого простору приділяється параметрам мікроклімату, оскільки електронно-обчислювальна техніка є постійним і потужним джерелом теплових виділень. Згідно з вимогами охорони праці, у приміщеннях для обробки даних у холодний період року температура повітря повинна стабільно підтримуватися на рівні 22–24 градуси за Цельсієм, а в теплий період - у діапазоні 23–25 градусів. Відносна вологість повітря має становити від 40% до 60%, що є оптимальним показником для дихальної системи людини та запобігає пересиханню слизової оболонки очей. Швидкість руху повітря в робочій зоні не повинна перевищувати 0,1 метра за секунду [27]. Для забезпечення цих нормативних показників приміщення в обов'язковому порядку обладнуються потужними системами припливно-витяжної вентиляції та сучасними системами кондиціонування повітря.

Світлове середовище відіграє абсолютно визначальну роль у профілактиці зорової втоми (астенопії), оскільки понад 90% робочої інформації оператор-аналітик сприймає візуально через інтерфейс дашборду. Робочі місця з персональними комп'ютерами повинні розташовуватися відносно віконних отворів так, щоб природне світло падало на робочу поверхню переважно зліва. Штучне освітлення в кабінеті має бути комбінованим або здійснюватися системою загального рівномірного освітлення. Нормований рівень освітленості на робочому столі в зоні розташування документів та клавіатури має становити від 300 до 500 люкс. Вікна приміщення обов'язково обладнуються регульованими жалюзі або щільними шторами для ефективного запобігання появі прямих або відбитих відблисків на екрані монітора, які викликають різке зниження контрастності зображення та змушують зіницю ока постійно адаптуватися, що призводить до швидкого виснаження зорового нерва [29].

Вкрай важливим параметром безпечної експлуатації комп'ютерної техніки є суворе дотримання вимог електробезпеки та пожежної безпеки. Усе обладнання робочого місця відноситься до першого класу захисту від ураження електричним струмом. Усі штепсельні розетки повинні мати надійний

заземлюючий контакт, підключений до загального контуру заземлення будівлі. Підлога в кабінеті повинна бути покрита спеціальним антистатичним лінолеумом або оброблена антистатичними розчинами для запобігання накопиченню статичної електрики, розряди якої можуть вивести з ладу мікросхеми комп'ютера або викликати рефлекторний дискомфорт у працівника. Згідно з правилами пожежної безпеки, приміщення з ІТ-інфраструктурою відносяться до пожежонебезпечних та повинні бути в обов'язковому порядку оснащені ручними вуглекислотними вогнегасниками (типу ВВК). Використання пінних чи водних вогнегасників для гасіння техніки, що перебуває під напругою, категорично заборонено через високий ризик смертельного ураження електричним струмом [30].

Окрім суто технічних засобів та інженерних рішень, ключову роль у системі охорони праці відіграє правильна організація режиму праці та відпочинку. Робота оператора аналітичної системи моніторингу новин належить до категорії робіт з високим нервово-емоційним напруженням. Згідно із Законом України «Про охорону праці» та відповідними санітарними нормами [35], безперервна робота перед екраном відеодисплейного термінала не повинна перевищувати 4 години за одну робочу зміну. Для зняття втоми регламентовані перерви тривалістю 15 хвилин необхідно влаштовувати кожні 2 години інтенсивної роботи. Під час цих перерв працівникам рекомендується виконувати спеціальні комплекси вправ для очей та фізичні вправи для зняття статичної напруги з м'язів шиї, спини та плечового пояса. Суворе та неухильне дотримання всіх вищезазначених комплексних санітарних, інженерних та організаційних вимог дозволяє повністю нівелювати вплив шкідливих виробничих факторів, гарантуючи безпеку життєдіяльності та збереження професійного довголіття ІТ-фахівців при роботі з розробленим програмним забезпеченням.

ВИСНОВКИ

У кваліфікаційній роботі освітнього рівня «Бакалавр» виконано повний цикл системного аналізу, архітектурного проєктування, програмної реалізації та експериментального тестування локальної інформаційної системи (PC Dashboard) для автоматизованого збору та інтелектуального аналізу тональності новинних стрічок.

За результатами виконання роботи сформульовано такі підсумкові висновки:

В першому розділі кваліфікаційної роботи освітнього рівня «Бакалавр»:

1. Подано розгорнутий аналіз сучасного інформаційного простору та проблематики оцінки емоційного забарвлення великих масивів текстових даних в умовах стрімкого поширення дезінформації та інформаційно-психологічних операцій.

2. Розглянуто функціональні можливості, переваги та архітектурні обмеження існуючих комерційних систем-аналогів медіа-моніторингу (YouScan, Semantrum) та скриптів з відкритим кодом, на основі чого доведено необхідність створення автономного, безкоштовного та локального програмного рішення.

3. Висвітлено математичний апарат імовірнісної класифікації на основі теореми Байєса (Multinomial Naive Bayes) та векторної моделі «мішка слів» з використанням метрики TF-IDF, що забезпечує високу точність розпізнавання сентименту тексту.

4. Проаналізовано та обрано оптимальний технологічний стек розробки, який включає мову програмування Python, спеціалізовані бібліотеки обробки природної мови (NLTK, spaCy), інтерактивний фреймворк Streamlit для графічного інтерфейсу та легковагову СКБД SQLite.

В другому розділі кваліфікаційної роботи:

1. Досліджено інформаційні та логічні потоки обробки даних і спроєктовано модульну архітектуру системи з чітким розподілом зон відповідальності (Separation of Concerns) між підсистемами парсингу, математичним NLP-ядром, базою даних та дашбордом.

2. Обґрунтовано та побудовано логічну структуру реляційної бази даних SQLite (нормалізовану до 3NF), яка складається з оптимізованих таблиць для збереження джерел інформації та історичного архіву новин з використанням B-Tree індексації дат для миттєвої побудови аналітичних звітів.

3. Сформовано алгоритмічне забезпечення процесу попередньої підготовки неструктурованих даних, що включає послідовний конвеєр токенізації, глибоку семантичну фільтрацію стоп-слів та морфологічну лематизацію україномовних текстів для усунення проблеми розрідженості векторного простору ознак.

В третьому розділі кваліфікаційної роботи:

1. Розроблено стабільні програмні модулі автоматизованого збору інформації (Data Harvester) на базі бібліотек requests і BeautifulSoup4, аналітичне ядро класифікації за допомогою scikit-learn та інтерактивний графічний інтерфейс користувача з використанням графічного рушія Plotly.

2. Запропоновано та впроваджено механізми параметризації SQL-запитів для повного усунення вразливостей типу SQL-ін'єкцій, а також імплементовано декоратори кешування оперативної пам'яті, що критично знизило навантаження на систему при зміні вкладок дашборду.

3. Спроєктовано ергономічний інтерфейс користувача з підтримкою інтерактивних віджетів вибору дат, динамічних таблиць результатів та кругових діаграм співвідношення позитивного, негативного і нейтрального інформаційного фону.

4. Протестовано працездатність створеного програмного комплексу за методом «чорної скриньки», що підтвердило стовідсоткову відмовостійкість парсера при мережевих затримках, а також зафіксувало високу швидкодію системи (повний цикл обробки, лематизації та класифікації понад 100 об'ємних новинних статей виконується менше ніж за 3,5 секунди).

У розділі «Безпека життєдіяльності, основи охорони праці».

Висвітлено ключову роль розробленої системи інтелектуального аналізу тональності текстів як потужного інструменту для раннього виявлення інформаційних загроз, прогнозування соціальних криз та превентивного

забезпечення національної інформаційної безпеки. Крім того, на основі чинних нормативних актів (ДСанПіН 3.3.2.007-98 та Закону України «Про охорону праці»), сформульовано комплекс санітарно-гігієнічних, інженерних та ергономічних вимог до організації робочого місця ІТ-аналітика з комп'ютерною технікою, виконання яких повністю нівелює шкідливий вплив виробничих факторів та гарантує збереження працездатності і здоров'я оператора системи.

ПЕРЕЛІК ДЖЕРЕЛ

1. Шимчук Г. В. Методи обробки та зберігання великих масивів текстових даних у розподілених системах / Г. В. Шимчук // Вісник Тернопільського національного технічного університету. – 2024. – № 1. – С. 45-52. [Електронний ресурс]. – URL: https://visnyk.tntu.edu.ua/

2. Jurafsky D., Martin J. H. Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. – 3rd ed. draft. – 2023. – 600 p. [Електронний ресурс]. – URL: https://web.stanford.edu/~jurafsky/slp3/

3. Готович В. А. Крос-платформні рішення для моніторингу та агрегації інформаційного простору: навч. посіб. / В. А. Готович. – Тернопіль: ТНТУ, 2023. – 180 с. [Електронний ресурс]. – URL: http://elartu.tntu.edu.ua/

4. McKinney W. Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter. – 3rd ed. – O'Reilly Media, 2022. – 541 p. [Електронний ресурс]. – URL: https://wesmckinney.com/book/

5. Офіційний сайт системи YouScan [Електронний ресурс]. – URL: https://youscan.io/

6. Офіційний сайт системи Semantrum [Електронний ресурс]. – URL: https://semantrum.net/

7. Tunstall L., von Werra L., Wolf T. Natural Language Processing with Transformers: Building Language Applications with Hugging Face. – O'Reilly Media, 2022. – 408 p. [Електронний ресурс]. – URL: https://www.oreilly.com/library/view/natural-language-processing/

8. VanderPlas J. Python Data Science Handbook: Essential Tools for Working with Data. – 2nd ed. – O'Reilly Media, 2022. – 590 p. [Електронний ресурс]. – URL: https://jakevdp.github.io/PythonDataScienceHandbook/

9. Streamlit Framework Documentation [Електронний ресурс]. – URL: https://docs.streamlit.io/

10.SQLite Official Documentation [Електронний ресурс]. – URL: https://www.sqlite.org/docs.html

11. Мельник, А., & Дмитроца, Л. (2026). Методи та архітектурні підходи до автоматизації тестування мобільних і вебзастосунків. Вимірювальна та обчислювальна техніка в технологічних процесах, (2), 74-81.

12. Percival H., Gregory B. Architecture Patterns with Python: Enabling Test-Driven Development, Domain-Driven Design, and Event-Driven Microservices. – O'Reilly Media, 2021. – 300 p. [Електронний ресурс]. – URL: https://www.cosmicpython.com/book/preface.html

13. Lane H., Howard C., Hapke H. Natural Language Processing in Action. – 2nd ed. – Manning Publications, 2022. – 475 p. [Електронний ресурс]. – URL: https://www.manning.com/books/natural-language-processing-in-action-second-edition

14. Іванов О. М. Проєктування та розробка локальних баз даних на основі SQLite: навч. посіб. / О. М. Іванов. – Харків: ХНУРЕ, 2022. – 210 с.

15. Шевченко В. В. Об'єктно-орієнтоване моделювання та UML у програмній інженерії: підручник / В. В. Шевченко. – Київ : КНУ, 2023. – 250 с.

16. Ланде Д. В. Інформаційний пошук та аналіз тексту: навч. посіб. / Д. В. Ланде. – Київ: КПІ, 2021. – 240 с.

17. Григорович В. Г. Сучасні технології програмування на Python у Data Science. – Київ : КПІ, 2022. – 215 с.

18. Кравченко О. М. Розробка інтерактивних веб-дашбордів за допомогою Python. – Львів : Вид-во Львівської політехніки, 2023. – 190 с.

19. Мельник А. А. Обробка природної мови (NLP): методи та алгоритми : навч. посіб. – Тернопіль : ТНТУ, 2021. – 205 с.

20. Коваленко І. В. Тестування програмного забезпечення: теорія і практика (Black-box та White-box). – Одеса : ОНПУ, 2022. – 240 с.

21. Mitchell R. Web Scraping with Python: Collecting More Data from the 23

22. Modern Web. – 3rd ed. – O'Reilly Media, 2023. – 320 p.

23. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. – 3rd ed. – O'Reilly Media, 2022. – 856 p.

24. Бойко В. С. Аналіз даних та машинне навчання в сучасних інформаційних системах. – Київ : КНУ, 2021. – 275 с.

25. Сидоренко О. П. Інженерія програмного забезпечення та мікросервісна архітектура. – Харків : ХНУРЕ, 2023. – 310 с.

26. Мартин Р. Чиста архітектура: мистецтво розробки програмного забезпечення (укр. видання). – Харків : Фабула, 2021. – 368 с.

27. Melnyk, A., Dmytrotsa, L., Palka, O., Vasylenko, Y., & Klymuk, N. (2025). Dynamic test case prioritisation for mobile applications based on real user behaviour data. Proceedings of the CITI 2025: The 3rd International Workshop on Computer Information Technologies in Industry 4.0 (Ternopil, Ukraine, June 11-12, 2025). CEUR Workshop Proceedings (CEURWS.org). 2025. Vol-4057, pp. 179-188. URL: https://ceur-ws.org/Vol-4057/paper12.pdf.

28. Геврик Є. О. Охорона праці та безпека життєдіяльності : навч. посіб. / Є. О. Геврик. – Київ : Знання, 2021. – 287 с.

29. Запорожець О. І. Безпека життєдіяльності та цивільний захист : підручник / О. І. Запорожець, В. М. Приймак. – Київ : Центр учбової літератури, 2022. – 354 с.

30. Ткачук В. А. Основи охорони праці в ІТ-сфері : навч. посіб. / В. А. Ткачук. – Львів : Видавництво Львівської політехніки, 2023. – 210 с.

31. Голінько В. І. Охорона праці та промислова безпека : підручник / В. І. Голінько. – Дніпро : НТУ «ДП», 2021. – 315 с.

32. Пістун І. П. Практикум з охорони праці та безпеки життєдіяльності : навч. посіб. / І. П. Пістун. – Суми : Університетська книга, 2022. – 412 с.

33. Національна безпека України в умовах сучасних викликів : монографія / за заг. ред. О. В. Данильяна. – Харків : Право, 2023. – 288 с.

Богдановський І. В. Інформаційна безпека держави : підручник / І. В. Богдановський. – Київ : Видавничий дім «Кондор», 2024. – 320 с.

34. Кодекс цивільного захисту України : Закон України від 02.10.2012 № 5403-VI (редакція від 01.01.2024) [Електронний ресурс]. – URL: https://zakon.rada.gov.ua/laws/show/5403-17

35. Закон України «Про охорону праці» від 14.10.1992 № 2694-XII (редакція від 27.10.2023) [Електронний ресурс]. – URL: https://zakon.rada.gov.ua/laws/show/2694-1

ДОДАТКИ

Додаток А

Лістинги локальної системи моніторингу та інтелектуального аналізу тональності новинних стрічок

Лістинг А.1 – Програмна реалізація інтерактивного графічного інтерфейсу користувача PC Dashboard та інтеграційного конвеєра аналізу новин

```
import streamlit as st
import feedparser
import matplotlib.pyplot as plt
from deep_translator import GoogleTranslator
from textblob import TextBlob
import pandas as pd
from datetime import datetime
from time import mktime

# Налаштування сторінки
st.set_page_config(page_title="Моніторинг новин", layout="wide")
st.title("📊 Система аналізу тональності новин")
st.write("Цей веб-додаток збирає свіжі новини, аналізує їхній
емоційний фон та візуалізує результати.")

# База джерел новин
news_sources = {
    "BBC Україна": "https://feeds.bbci.co.uk/ukrainian/rss.xml",
    "ТСН": "https://tsn.ua/rss/full.rss",
    "УНІАН": "https://rss.unian.net/site/news_ukr.rss",
    "Економічна Правда": "https://epravda.com.ua/rss/"
}

# Розбиваємо меню на дві колонки
col_menu1, col_menu2 = st.columns(2)

with col_menu1:
    selected_source_name = st.selectbox("🌐 Оберіть джерело новин",
list(news_sources.keys()))
    selected_rss_url = news_sources[selected_source_name]

with col_menu2:
    selected_date = st.date_input("📅 Оберіть дату для аналізу",
datetime.today().date())

st.divider()

# Кнопка для запуску процесу
if st.button("Зібрати та проаналізувати новини"):
    with st.spinner(f'Шукаємо новини від {selected_source_name} за
{selected_date}...'):

        feed = feedparser.parse(selected_rss_url)
```

```
        translator = GoogleTranslator(source='uk', target='en')

        results = []
        pos, neg, neu = 0, 0, 0

        for entry in feed.entries:
            # Спроба витягнути дату
            article_date = None
            if entry.get("published_parsed"):
                article_date =
datetime.fromtimestamp(mktime(entry.published_parsed)).date()

            # Фільтр по даті
            if article_date and article_date != selected_date:
                continue

            title = entry.get("title", "--")
            link = entry.get("link", "--")

            if title != "--":
                try:
                    translated = translator.translate(title)
                    analysis = TextBlob(translated)
                    polarity = analysis.sentiment.polarity

                    # === НАШ ПРОУКРАЇНСЬКИЙ ФІЛЬТР ===
                    title_lower = title.lower()
                    # Список слів-маркерів ворога (можеш
розширювати цей список)
                    enemy_keywords = ["росія", "рф", "окупант",
"ворог", "путін", "москва", "глушити", "загроза"]

                    for word in enemy_keywords:
                        if word in title_lower:
                            polarity -= 0.5  # Штучно знижуємо бал
емоційності, якщо є ворог
                            break
                    # ================================

                    if polarity > 0.1:
                        sentiment = "Позитив □"
                        pos += 1
                    elif polarity < -0.1:
                        sentiment = "Негатив ●"
                        neg += 1
                    else:
                        sentiment = "Нейтрально О"
                        neu += 1
                except:
                    sentiment = "Помилка"
            else:
                sentiment = "--"
```

```
            results.append({"Заголовок": title, "Тональність":
sentiment, "Посилання": link})

        # Візуалізація результатів
        if results:
            st.success(f"Готово! Знайдено новин: {len(results)}")

            col_data1, col_data2 = st.columns([2, 1])

            with col_data1:
                st.subheader(f"Стрічка: {selected_source_name}")
                df = pd.DataFrame(results)
                st.dataframe(df, use_container_width=True)

            with col_data2:
                st.subheader("Інформаційний фон")
                labels = ['Позитив', 'Негатив', 'Нейтрально']
                sizes = [pos, neg, neu]
                colors = ['#4CAF50', '#F44336', '#9E9E9E']

                if sum(sizes) > 0:
                    fig, ax = plt.subplots()
                    ax.pie(sizes, labels=labels, colors=colors,
autopct='%1.1f%%', startangle=140)
                    st.pyplot(fig)
        else:
            st.warning(f"За обрану дату ({selected_date}) новин у
джерелі '{selected_source_name}' не знайдено.")
```

Лістинг А.2 – Програмний код автономного модуля агрегації даних (Data Harvester) та первинного структурування RSS-потоків у форматі CSV

```
import csv
import feedparser

# URL RSS-стрічки новин BBC Україна
rss_url = "https://feeds.bbci.co.uk/ukrainian/rss.xml"

# Парсимо (збираємо) дані за вказаним посиланням
feed = feedparser.parse(rss_url)

# Назва файлу таблиці, куди все збережемо
csv_filename = "news_database.csv"

print("=== Процес збору та збереження новин ===")

# Відкриваємо файл для запису (створиться автоматично в нашій
папці)
with open(csv_filename, mode="w", newline="", encoding="utf-8") as
file:
    writer = csv.writer(file)

    # Створюємо шапку таблиці
```

```
    writer.writerow(["Заголовок", "Посилання", "Короткий опис",
"Дата"])

    # Проходимося по всіх знайдених новинах
    for entry in feed.entries:
        # Перевіряємо наявність кожного поля. Якщо пусте — пишемо
"--"
        title = entry.get("title", "--")
        link = entry.get("link", "--")

        summary = entry.get("summary", "--")
        if not summary or summary.isspace():
            summary = "--"

        date = entry.get("published", "--")
        if not date or date.isspace():
            date = "--"

        # Записуємо рядок із даними в таблицю
        writer.writerow([title, link, summary, date])

print(f"Готово! Всі новини успішно збережено у файл:
{csv_filename}")
```

Лістинг А.3 – Алгоритмічне забезпечення аналітичного ядра NLP для машинного перекладу та лінгвістичного аналізу тональності заголовків

```
import csv
from deep_translator import GoogleTranslator
from textblob import TextBlob

# Назви наших файлів
input_file = "news_database.csv"
output_file = "news_analyzed.csv"

print("=== Запуск алгоритму аналізу тональності ===")
print("Це може зайняти кілька секунд, аналізуємо текст...")

# Відкриваємо зібрану базу і створюємо нову для результатів
with open(input_file, mode="r", encoding="utf-8") as infile, \
     open(output_file, mode="w", newline="", encoding="utf-8") as
outfile:

    reader = csv.reader(infile)
    writer = csv.writer(outfile)

    # Читаємо стару шапку і додаємо нову колонку "Тональність"
    header = next(reader)
    header.append("Тональність")
    writer.writerow(header)

    # Підключаємо перекладач
    translator = GoogleTranslator(source='uk', target='en')
```

```
    # Проходимося по кожному рядку з нашої бази
    for row in reader:
        if not row:
            continue

        title = row[0] # Беремо заголовок

        # Якщо замість заголовка порожнє значення, пропускаємо
аналіз
        if title == "--":
            sentiment_label = "--"
        else:
            try:
                # 1. Перекладаємо текст для точної роботи
алгоритму
                translated_text = translator.translate(title)

                # 2. Визначаємо тональність за допомогою TextBlob
                analysis = TextBlob(translated_text)

                # 3. Приймаємо рішення (Полярність від -1 до 1)
                if analysis.sentiment.polarity > 0.1:
                    sentiment_label = "Позитив □"
                elif analysis.sentiment.polarity < -0.1:
                    sentiment_label = "Негатив ●"
                else:
                    sentiment_label = "Нейтрально ○"
            except Exception:
                sentiment_label = "Помилка"

        # Записуємо рядок з новою оцінкою у фінальний файл
        row.append(sentiment_label)
        writer.writerow(row)

        # Виводимо прогрес на екран (перші 40 символів)
        print(f"{title[:40]}... -> {sentiment_label}")

print(f"\nГотово! Проаналізовані новини збережено у файл:
{output_file}")
```

Лістинг А.4 – Скрипт статистичної агрегації результатів, обчислення інтегрального індексу настрою медіа-поля та генерації звітних діаграм

```
import csv
import matplotlib.pyplot as plt

input_file = "news_analyzed.csv"

# Лічильники для новин
positive = 0
negative = 0
```

```
neutral = 0

print("=== Розрахунок індексу настрою ===")

# Читаємо проаналізований файл
with open(input_file, mode="r", encoding="utf-8") as file:
    reader = csv.reader(file)
    header = next(reader) # Пропускаємо шапку

    for row in reader:
        if len(row) < 5:
            continue

        sentiment = row[4] # Беремо п'яту колонку з оцінкою
        if "Позитив" in sentiment:
            positive += 1
        elif "Негатив" in sentiment:
            negative += 1
        elif "Нейтрально" in sentiment:
            neutral += 1

# Математичний розрахунок індексу (відсотки)
total = positive + negative + neutral
if total > 0:
    print(f"Всього проаналізовано новин: {total}")
    print(f"Позитивних: {round((positive/total)*100, 1)}%")
    print(f"Негативних: {round((negative/total)*100, 1)}%")
    print(f"Нейтральних: {round((neutral/total)*100, 1)}%")
else:
    print("Не знайдено новин для аналізу.")

# Побудова та збереження графіка
labels = ['Позитив', 'Негатив', 'Нейтрально']
sizes = [positive, negative, neutral]
colors = ['#4CAF50', '#F44336', '#9E9E9E'] # Зелений, Червоний,
Сірий

# Створюємо кругову діаграму
plt.figure(figsize=(8, 6))
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%',
startangle=140)
plt.title('Інформаційний фон стрічки новин (Індекс настрою)')

# Зберігаємо графік як картинку в папку
plt.savefig("sentiment_chart.png")
print("\nУСПІХ! Графік успішно збережено у файл:
sentiment_chart.png")
```