НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

«На правах рукопису»
УДК 004.42

До захисту допущено:
Завідувач кафедри
_____ Михайло НОВОТАРСЬКИЙ
«___» _____2025 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інженерія програмного забезпечення комп'ютерних систем» спеціальності 121 «Інженерія програмного забезпечення» на тему: «Система виявлення, агрегації та семантичного аналізу інформаційних подій із використанням штучного інтелекту»

Виконав:
студент ІІ курсу, групи ІМ-42мп
Шевчук Петро Олександрович _____

Науковий керівник:
к.т.н., доц. Шимкович В.М. _____

Консультант:
доцент, Волокита Артем Миколайович _____

Рецензент:
проф. каф. ІПІ, д.т.н., проф. Стеценко Інна Вячеславівна
_____

Засвідчую, що у цьому дипломному проєкті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2025 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки
(повне найменування інституту, факультету)

Кафедра обчислювальної техніки
(повна назва кафедри)

Рівень вищої – освіти – другий (магістерський)

Спеціальність – 121 «Інженерія програмного забезпечення»

Освітньо-професійна програма – «Інженерія програмного забезпечення комп'ютерних систем»

До захисту допущено:
Завідувач кафедри

_____ Михайло НОВОТАРСЬКИЙ
(підпис) (ім'я, прізвище)

«_____»_____2025 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Шевчуку Петру Олександровичу

1. Тема проекту *"*Система виявлення, агрегації та семантичного аналізу інформаційних подій із використанням штучного інтелекту*"*, керівник дисертації *Шимкович Володимир Миколайович, к.т.н., доц.* , затверджені наказом по університету від *« 06 » 11* 2025р. No 4841

2. Термін подання студентом проекту *«10» 12 2025.*

3. Вихідні дані до проекту: аналіз вхідних даних із різних джерел, очистка, нормалізація даних, побудова системи класифікації семантичного аналізу текстів, побудова та збереження звітів аналізу

4. Зміст пояснювальної записки: *огляд існуючих аналогів, розгляд схем їх реалізації, а також принципів роботи, пошук та аналіз їх основних переваг*

та недоліків, проектування та розробка програмного забезпечення, тестування програмного модулю та аналіз отриманих результатів.

5. Перелік завдань, які потрібно розробити: *огляд існуючих рішень, розгляд схем їх реалізації, а також принципів роботи, проектування та розробка програмного забезпечення, тестування програмного модулю та аналіз отриманих результатів*

6. Консультанти розділів дисертації:

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
1.	*доц. Шимкович В.М*		
2.	*доц. Шимкович В.М*		
3.	*доц. Шимкович В.М*		
3.	*доц. Шимкович В.М*		

7. Дата видачі завдання 01.09.2025 року

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту(роботи)	Примітки
1	*Обговорення теми дисертації. Визначення предмету дослідження*	*1.09.25-7.09.25*	
2	*Дослідження існуючих рішень та проблем у досліджуваній області*	*22.09.25-28.09.25*	
3	*Побудова архітектурного рішення*	*29.09.25-05.10.25*	
4	*Програмна реалізація створеного алгоритму*	*06.10.25-10.10.25*	
5	*Програмування та загальне налагодження системи.*	*10.10.25-19.10.25*	
6	*Розробка стартам-проекту*	*14.11.25-18.11.25*	
7	*Оформлення магістерської дисертації.*	*19.11.25-24.11.25*	

Студент _____ Петро ШЕВЧУК
(підпис)

Керівник _____ Володимир ШИМКОВИЧ

РЕФЕРАТ

на магістерську дисертацію

виконану на тему:

“Система виявлення, агрегації та семантичного аналізу інформаційних подій із використанням штучного інтелекту”

Студентом групи ІМ-42мп Шевчук Петро

Робота складається із вступу та чотирьох розділів. Загальний обсяг роботи: 108 аркушів основного тексту, 32 ілюстрації, 34 таблиці. При підготовці використовувалася література з 35 різних джерел.

Актуальність

У сучасному світі обсяг інформації, що генерується користувачами, платформами новин та соціальними мережами, зростає експоненційно. У таких умовах здатність систем автоматично виявляти значущі інформаційні події та тренди у реальному часі набуває критичного значення. Це важливо як для оперативного реагування на надзвичайні події, так і для виявлення потенційно небезпечних або маніпулятивних інформаційних кампаній. Традиційні системи аналітики не здатні ефективно справлятися з потоковими, багатоджерельними даними без втрати релевантності або контексту.

Розвиток технологій штучного інтелекту, обробки природної мови та систем розподіленої обробки даних дозволяє створити інтелектуальну

платформу, здатну автоматично збирати, стандартизувати та аналізувати дані з відкритих джерел у режимі реального часу. Подібна система дозволяє не тільки агрегувати події, а й надавати їм семантичну оцінку та класифікацію, виявляючи значущі тренди до моменту їх пікового поширення.

Мета і завдання дослідження

Метою роботи є розробка інтелектуальної системи, що забезпечує виявлення, агрегацію та семантичний аналіз інформаційних подій і трендів на основі багатоджерельних даних. Система адапотована під потреби користувача та тем які йому важливо відслідковувати, які вони можуть обрати

самі. Дослідження спрямоване на створення інструменту, спрямованого на полегшення процесу відслідковування інформаційних сплесків на обрану категорію тем та систему сповіщення.Для досягнення цієї мети поставлено такі завдання:

● провести огляд існуючих підходів до виявлення інформаційних подій’і трендів;

● розробити архітектуру системи збору, попередньої обробки та уніфікації даних;

● реалізувати модулі AI-аналізу: семантична класифікація, кластерний аналіз, виявлення аномалій та трендів;

● створити API та інтерфейс для взаємодії з системою

● протестувати систему на реальних прикладах (наприклад, політичні події, інформаційні хвилі).

Об’єкт дослідження - процеси збору, агрегації та аналізу інформаційних повідомлень у реальному часі.

Предмет дослідження - методи та інструменти семантичного аналізу багатоджерельних потокових даних із використанням технологій штучного інтелекту.

Наукова новизна одержаних результатів роботи полягає у наступному:

● Розроблено підхід до автоматичного виявлення інформаційних подій та їх семантичної інтерпретації на основі уніфікованої моделі обробки

● Запропонована система поєднує AI-моделі, інфраструктуру для обробки даних у реальному часі, а також методи

● Візуалізації та API-доступу до результатів аналізу.

Практична цінність

Система може бути використана:

● для моніторингу інформаційних кампаній;

● у сфері фінансової аналітики (аналіз новин, пов’язаних із криптовалютами);

● для оперативного виявлення важливих подій у новинах та соцмережах;

● як інструмент для OSINT-розслідувань або ситуаційних центрів.

Розроблене рішення може слугувати основою для створення комерційного продукту або модульної платформи для інформаційної аналітики в різних доменах.

Ключові слова

Штучний інтелект, тренди, інформаційні події, семантичний аналіз, класифікація, Twitter, Telegram, NLP, Python, AWS, Airflow, NoSQL

ABSTRACT

of the master's thesis

on the topic:

"System for detection, aggregation, and semantic analysis

of information events using artificial intelligence"

by Petro Shevchuk, student of group IM-42mp

The work consists of an introduction and four chapters. Total volume of work: 108 pages of main text, 32 illustrations, 34 tables. Literature from 35 different sources was used in the preparation.

Relevance

In today's world, the amount of information generated by users, news platforms, and social networks is growing exponentially. In such conditions, the ability of systems to automatically detect significant information events and trends in real time becomes critical. This is important both for responding quickly to emergencies and for identifying potentially dangerous or manipulative information campaigns. Traditional analytics systems are unable to effectively handle streaming, multi-source data without losing relevance or context.

The development of artificial intelligence, natural language processing, and distributed data processing technologies allows us to create an intelligent platform capable of automatically collecting, standardizing, and analyzing data from open sources in real time. Such a system allows not only to aggregate events, but also to provide them with semantic evaluation and classification, identifying significant trends before they reach their peak.

Research goals and objectives

The goal of this work is to develop an intelligent system that provides detection, aggregation, and semantic analysis of information events and trends based on multi-source data. The system is adapted to the needs of the user and the topics that are important for them to track, which they can choose themselves. The research is aimed at creating a tool to facilitate the process of tracking information surges on a selected

category of topics and a notification system. To achieve this goal, the following tasks have been set:

- review existing approaches to identifying information events and trends;
- develop a system architecture for collecting, pre-processing, and unifying data;
- implement AI analysis modules: semantic classification, cluster analysis, anomaly and trend detection;
- create an API and interface for interacting with the system
- test the system on real-world examples (e.g., political events, information waves).

The object of study is the processes of collecting, aggregating, and analyzing information messages in real time.

The subject of the study is methods and tools for semantic analysis of multi-source streaming data using artificial intelligence technologies. The scientific novelty of the results obtained lies in the following:

The scientific novelty of the obtained results lies in the following:

- An approach has been developed for the automatic detection of information events and their semantic interpretation based on a unified processing model
- The proposed system combines AI models, infrastructure for real-time data processing,
- Methods of visualization and API access to the analysis results.

Practical value

The system can be used:

- to monitor information campaigns;
- in the field of financial analytics (analysis of news related to cryptocurrencies);
- for the rapid detection of important events in the news and social networks;
- as a tool for OSINT investigations or situation centers.

The developed solution can serve as a basis for creating a commercial product or a modular platform for information analytics in various domains.

Keywords

artificial intelligence, trends, information events, semantic analysis, classification, Twitter, Telegram, NLP, Python, AWS, Airflow, NoSQL

ЗМІСТ

ВСТУП

У сучасному інформаційному суспільстві динаміка появи нових подій, тем та інформаційних хвиль зростає в геометричній прогресії. Щодня користувачі соціальних мереж, новинних платформ та месенджерів генерують мільярди повідомлень, що відображають події локального та глобального масштабу — від стихійних лих і протестів до запусків нових технологій чи криптовалют. У таких умовах здатність виявляти інформаційно значущі тренди у режимі реального часу перетворюється з конкурентної переваги на необхідність для організацій, державних структур та суспільства загалом.

Водночас обробка подібних даних стикається з низкою викликів: багатоманітністю джерел, відсутністю уніфікованих структур, високою швидкістю надходження нових подій, а також інформаційним шумом, фейками та активністю ботів. Традиційні підходи до обробки даних — офлайн-аналітика, ручне сортування чи централізовані системи збору — виявляються малоефективними у таких умовах.

З одного боку, затримка в реагуванні на критичні події може коштувати людських життів або величезних фінансових втрат. З іншого — раннє виявлення трендів дає унікальну конкурентну перевагу бізнесу та аналітикам, дозволяючи приймати зважені рішення до піку популярності теми. Сучасні рішення, такі як Google Trends чи Bloomberg Terminal, забезпечують певну аналітику, але зазвичай не працюють у справжньому «стрімінгу» та не поєднують гнучку обробку різнорідних форматів даних із глибоким AI-аналізом.

Ключовою особливістю цієї роботи є також архітектурний підхід, що дозволяє масштабувати систему, розподіляти обчислення, обробляти дані потоково та інтегрувати її з реальними джерелами через webhook-и, API, брокери повідомлень тощо. Це дозволяє не тільки створити ефективну систему виявлення подій, а й забезпечити її застосування у виробничих умовах або як платформу для подальших досліджень.

Таким чином, дана магістерська дисертація присвячена розробці та впровадженню інтелектуальної системи виявлення та семантичного аналізу

інформаційних подій і трендів у реальному часі. Її завданням є об’єднати сучасні інженерні рішення з досягненнями в сфері штучного інтелекту, створивши інструмент, який не лише збирає та аналізує дані, але й робить їх семантично зрозумілими, інтерпретованими і корисними для ухвалення рішень.

РОЗДІЛ 1

АНАЛІЗ СИСТЕМ ТА МЕТОДІВ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ІНФОРМАЦІЙНИХ ПОДІЙ

1.1 Огляд задачі виявлення та аналізу інформаційних подій

У світі, де обсяг інформації зростає у геометричній прогресії, а швидкість її поширення перевищує можливості людини на осмислення, здатність вчасно виявляти інформаційні події, тренди та ризики є критично важливою як для державних структур, так і для бізнесу, журналістики та громадських організацій. Потоки даних із соцмереж (Twitter, Telegram, Reddit) [1], новинних стрічок, коментарів, відео та аудіо є прикладом так званих багатоджерельних потокових даних, що потребують спеціалізованих систем для їх обробки та аналізу.

У традиційному підході моніторинг новин або трендів часто здійснюється вручну або через спрощені агрегатори новин. Проте така модель є повільною, схильною до людських помилок та не дозволяє виявляти важливі події на ранньому етапі їх поширення, що є надзвичайно важливим у ситуаціях соціальних криз, фальсифікацій, інформаційних атак або навіть масових протестів.

Ще складнішою ситуація стає у випадку дезінформації або інформаційного тероризму — коли події створюються навмисне, з використанням ботоферм або скоординованих Telegram-каналів, і поширюються лавиноподібно до моменту офіційного спростування. Класичні системи або виявляють такі події із запізненням, або зовсім не помічають, що тягне за собою значні репутаційні, соціальні чи фінансові наслідки.

Одним із перспективних напрямів розвитку є використання штучного інтелекту (ШІ) для автоматичного збору, класифікації та семантичної інтерпретації таких подій. Системи, що базуються на ШІ-моделях, здатні аналізувати як зміст повідомлень, так і їхню емоційну насиченість, частоту повторень, крос-платформену присутність тощо. У поєднанні з потоковими обчисленнями та відповідною архітектурою (Kafka, Lambda, API Gateway)[2] це відкриває можливості для створення реальних інтелектуальних агентів моніторингу.

Крім оперативності, важливим є і аспект стандартизації та агрегації: інформація з різних джерел має різну структуру, мову, часову зону, що ускладнює її подальший аналіз. Успішні системи мають включати шар нормалізації даних, видалення шуму, дублювання, виявлення фейків, а також генерації узагальненої інформації, наприклад у вигляді embedding-структур.[3]

Особливу актуальність це питання набуло з початком повномасштабної війни в Україні, де інформаційна безпека стала не менш важливою за фізичну. Саме з цією метою аналітики, журналісти, OSINT-експерти та громадські ініціативи [4] шукають інструменти, що дозволяють отримувати об’єктивну інформацію раніше, ніж її опублікують офіційні медіа або урядові органи.

У бізнес-середовищі така система може бути використана для виявлення ринкових сигналів (наприклад, анонс запуску нової технології, активність навколо певної криптовалюти , або хвиля негативу щодо продукту), що відкриває можливості превентивного реагування або інвестиційних рішень.

Таким чином, задача по побудові даної системи охоплює перетин кількох напрямів:

- технології потокової обробки даних;
- машинне навчання й обробка природної мови (NLP)[5];
- інженерія ETL/ELT-пайплайнів[6];
- виявлення подій, трендів і аномалій;
- семантичне групування та embedding-аналіз;
- інтерфейсна й API-доступність результатів.

Далі у цьому розділі буде проаналізовано існуючі програмні рішення,

академічні системи та дослідження, які стосуються моніторингу подій у реальному часі з використанням ШІ, та визначено, чим саме пропонована система вирізняється з-поміж них.

1.2. Порівняльний аналіз існуючих рішень

У цьому підрозділі здійснюється аналіз наявних програмних систем і сервісів, які виконують завдання виявлення інформаційних подій, трендів або аномалій на основі даних з відкритих джерел. Основна мета аналізу — визначити сильні сторони, обмеження, архітектурні підходи та особливості реалізації, які можуть бути враховані при проєктуванні власної інтелектуальної системи.

1.2.1. Огляд системи Dataminr

Dataminr[7] є однією з найвідоміших комерційних платформ для моніторингу інформаційного простору в реальному часі. Система була створена з акцентом на надшвидке виявлення подій, про які ще не повідомили традиційні ЗМІ. Вона в реальному часі обробляє стрімінгові дані з соцмереж (історично — X/Twitter), новинних сайтів, блогів, окремих Telegram-каналів, форумів та інших відкритих джерел, перетворюючи «сирий» потік повідомлень на алерти про події для державних структур, корпорацій і фінансових організацій.

Рис.1.1. Dataminr користувацький інтерфейс

Архітектурно це SaaS-рішення на основі хмарної інфраструктури: у центрі — багаторівнева ШІ-платформа, яка поєднує NLP, аналіз зображень/відео та виявлення аномалій, поверх якої розгорнуті кілька продуктів. Користувачі працюють через веб-інтерфейс та мобільні застосунки, а також можуть інтегрувати сповіщення у внутрішні процеси за допомогою email, Slack та REST API.

Таблиця 1.1

Переваги системи Dataminr:

№	Аспект	Суть
1	Раннє виявлення подій	Сповіщення формуються за лічені секунди на основі слабких сигналів з багатьох джерел.
2	Широкий функціонал	Тематичні фільтри, геофільтри, готові сценарії для безпеки, кризового реагування тощо.
3	Зручна інтеграція	Хмарний доступ, REST API, email/Slack-оповіщення, мобільні додатки для оперативної роботи.
4	Розвинений аналітичний модуль	Класифікує події за тематикою (державні події, конфлікти, надзвичайні ситуації, транспорт, фінанси тощо

Таблиця 1.2

Недоліки системи Dataminr:

№	Аспект	Суть
1	Закритість	Алгоритми, моделі та внутрішні структури даних недоступні для дослідників і відтворення.
2	Орієнтація на «enterprise»	Висока вартість, фокус на державі й великих корпораціях, фактично недосяжна для невеликих команд.
3	Обмежена прозорість	Неможливо незалежно перевірити якість виявлення подій чи повторити підходи в open-source рішенні.

Технічна реалізація: хмарна архітектура з API-доступом, потокова обробка даних, власні ШІ-моделі.

1.2.2. Огляд системи Event Registry

Event Registry [8] — платформа, орієнтована на обробку новинних стрічок з тисяч джерел у різних країнах. Основний фокус платформи — структурування новин у вигляді подій: кожна новинна стаття прив’язується до теми, місця, часу та сутностей (персон, організацій, локацій), що дозволяє будувати запити за географією, тематикою, часовими інтервалами й пов’язаними об’єктами.

Рис.1.2. Event Registry користувацький інтерфейс

Таблиця 1.2

Переваги та недоліки системи Event Registry:

№	Тип	Аспект	Опис
1	Перевага	Широке покриття новин	Агрегація великої кількості RSS-стрічок і новинних сайтів з різних країн та мов, що дозволяє будувати глобальну картину подій.

Таблиця 1.2(продовження)

Переваги та недоліки системи Event Registry:

№	Тип	Аспект	Опис
2	Перевага	Структурування подій	Підтримка пошуку не лише окремих статей, а й «подій» з прив’язкою до тем, місць, часу та сутностей, що спрощує аналітику новинного простору.
3	Перевага	Зручна інтеграція	RESTful API, JSON/GeoJSON-відповіді та готові SDK (Python, Node.js) дають змогу легко вбудовувати сервіс у сторонні системи.
4	Недолік	Фокус на офіційних медіа	Основний акцент робиться на онлайн-ЗМІ, тоді як потоки з соцмереж та неформальних каналів охоплюються значно слабше.
5	Недолік	Обмежена «хвильова» семантика	Події добре структуровані на рівні новин, але немає явної підтримки аналізу «інформаційних хвиль» у сенсі агрегації багатьох слабких сигналів із соцмереж.
6	Недолік	Закритий сервіс	Платформа працює як комерційний сервіс: алгоритми та внутрішня модель даних недоступні для модифікації або відтворення у власній інфраструктурі.

Технічна реалізація: RESTful API, підтримка JSON/GeoJSON, інтеграція через SDK.

1.2.3. Огляд системи GDELT

GDELT [9] (Global Database of Events, Language and Tone) — це відкритий дослідницький проєкт, який автоматично агрегує новинний контент із глобальних медіа, кодує події у стандартизованому форматі й зберігає їх із прив'язкою до часу, місця, акторів та тональності. Дані доступні починаючи з 1979 року й оновлюються майже в реальному часі. Доступ до наборів надається через публічні CSV/TSV-файли, Google BigQuery та спеціалізовані API й утиліти, що робить GDELT одним із наймасштабніших відкритих джерел для аналізу глобальних подій.

Таблиця 1.3

Переваги та недоліки системи GDELT

№	Тип	Аспект	Опис
1	Перевага	Відкритість і доступ до даних	Повні набори подій і пов'язаних ознак доступні безкоштовно у форматах CSV/TSV, через BigQuery та API, що дозволяє відтворювати дослідження й будувати власні аналітичні пайплайни.
2	Перевага	Просторово-часова та подієва структура	Події мають координати, часові мітки та атрибути акторів/цілей, що дає змогу будувати карти, часові ряди та моделі взаємодії між суб'єктами.
3	Перевага	Довга історія спостережень	База охоплює кілька десятиліть (із кінця 1970-х років), що дозволяє досліджувати довгострокові тренди та історичну динаміку подій.

Таблиця 1.3(продовження)

Переваги та недоліки системи GDELT

№	Тип	Аспект	Опис
4	Недолік	Обмеженість джерел	Основний акцент робиться на новинних медіа; потоки із соціальних мереж або месенджерів у стандартних наборах практично не представлені, що знижує чутливість до «низових» інформаційних сигналів.
5	Недолік	Складність структури даних	Множина таблиць, кодів подій та технічних змін у версіях GDELT створює високий поріг входу: новим користувачам потрібен час, щоб розібратися у схемах і змінності формату.
6	Недолік	Відсутність «готового» інтерфейсу для кінцевого користувача	Система орієнтована на дослідників і розробників: немає повноцінного інтерактивного GUI для широкої аудиторії, більшість роботи виконується через SQL (BigQuery), скрипти або власні візуалізації.

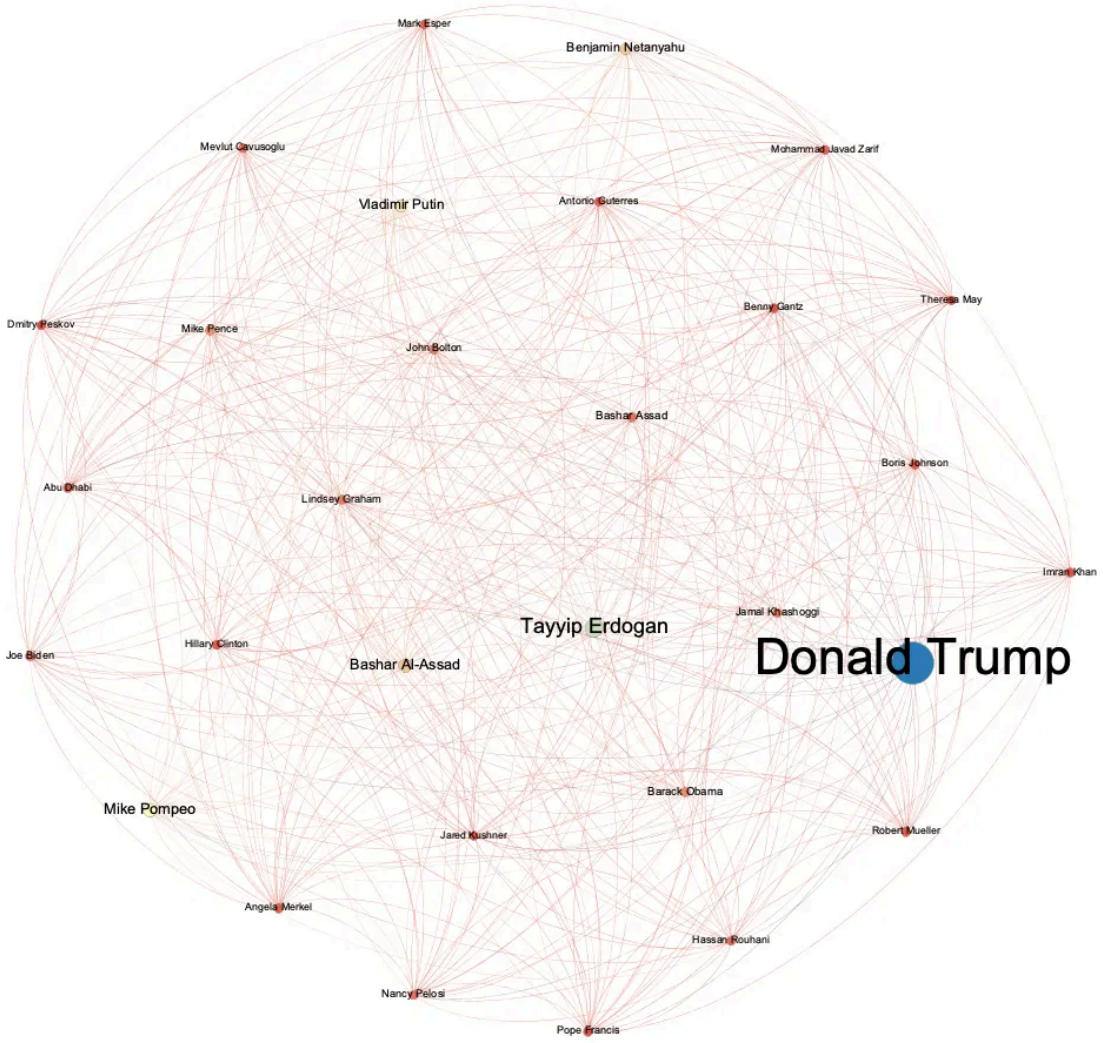

Рис.1.3. хмара слів GDELT

Технічні особливості: потокова обробка новин через crawler + NLP на Hadoop, інтеграція з Google Cloud.

1.2.4. Огляд системи Meltwater

Meltwater [10] — це хмарна (SaaS) платформа для медіа-та соціального моніторингу, орієнтована насамперед на потреби PR-, маркетингових і бренд-команд. Вона агрегує контент із онлайн-новин, блогів, форумів, соціальних мереж, подкастів, а також друкованих та ефірних медіа, забезпечуючи єдину точку доступу до згадок про бренд у різних каналах. Платформа надає інструменти для

відстеження обсягу згадок, частки голосу, ключових тем, географії згадок та впливових авторів, а також використовує засоби аналізу тональності та штучного інтелекту для формування аналітичних звітів і дашбордів. Таким чином, Meltwater позиціонується як комплексне рішення для бренд-моніторингу, репутаційного менеджменту та маркетингової аналітики, а не як спеціалізований інструмент для OSINT чи безпекових завдань.

Таблиця 1.4

Переваги та недоліки системи Meltwater

№	Тип	Аспект	Опис
1	Перевага	Широке покриття джерел	Платформа охоплює онлайн-новини, соціальні мережі, блоги, форуми, подкасти, ТВ і радіо, що дозволяє будувати цілісну картину медіапростору навколо бренду.
2	Перевага	Фокус на брендовому моніторингу	Функціонал орієнтований на PR і маркетинг: відстеження згадок, частки голосу, конкурентного поля, ефективності кампаній та репутаційних ризиків.
3	Перевага	Аналітика на базі ШІ	Використовуються алгоритми аналізу тональності та інші AI-модулі, включно з LLM-підходами для узагальнення та автоматизації звітів, що зменшує ручну аналітичну роботу.
4	Недолік	Закритість алгоритмів	Ядро системи є комерційним продуктом із непрозорими моделями; користувач не має можливості змінювати чи відтворювати алгоритми в власній інфраструктурі.

Таблиця 1.4(продовження)

Переваги та недоліки системи Meltwater

№	Тип	Аспект	Опис
5	Недолік	Маркетинговий, а не OSINT-фокус	Архітектура й сценарії використання заточені під бренди та ринки; платформа не спеціалізується на виявленні критичних безпекових подій або глибокому OSINT-аналізі.
6	Недолік	Мовні та цінові обмеження	Хоча Meltwater підтримує багато ринків, найвищий рівень опрацювання зазвичай орієнтований на великі, переважно англомовні бренди; вартість і модель ліцензування роблять рішення малодоступним для академічних чи невеликих дослідницьких проєктів.

Рис.1.4 Meltwater користувацький інтерфейс

Технологічні властивості: AWS-based платформа, потокові pipeline-и, NLP з підтримкою ElasticSearch.

1.2.5. Огляд системи OSINT-систем

У контексті цієї роботи під OSINT-системами[11] розглядаються не стільки монолітні платформи на кшталт Dataminr чи Meltwater, скільки набір інструментів і практик, які використовують дослідники для глибокого аналізу відкритих джерел.

До таких інструментів належать:

- Bellingcat та подібні OSINT-спільноти — не стільки окрема система, скільки методологія й екосистема інструментів. Bellingcat публікує розслідування, засновані на відкритих даних (супутникові знімки, соцмережі, відкриті реєстри), та підтримує власний онлайн-«toolkit» із добіркою сервісів для пошуку, верифікації мультимедіа, аналізу соцмереж тощо.
- Maltego[12] — настільний інструмент для графового аналізу зв’язків: дозволяє будувати графи між доменами, IP-адресами, акаунтами соцмереж, e-mail-ами та іншими сутностями, підключаючи численні зовнішні джерела (у тому числі соцмережі й мессенджери через сторонні «трансформи»).
- Telegram-орієнтовані інструменти (TGStat, Telepathy та ін.) — сервіси й бібліотеки, що спеціалізуються на аналізі Telegram-каналів та чатів. Наприклад, TGStat надає веб-аналітику каналів (динаміка підписників, охоплення, цитування), а Telepathy — це open-source-toolkit для завантаження та аналізу повідомлень із Telegram-груп і каналів (експорт чатів, збір списків учасників, аналіз активності тощо).

Спільною рисою цих рішень є те, що вони орієнтовані на ручне або напівручне дослідження: аналітик сам формує запити, збирає шматки даних із різних джерел, пов’язує їх між собою та робить висновки. Інструменти при цьому забезпечують доступ до даних, зручну візуалізацію графів зв’язків або аналітику окремих каналів, але не виконують повністю автоматичне виявлення інформаційних подій у потоковому режимі.

Це відкриває нішу для створення інтелектуальної платформи з вищезазначеним функціоналом, яку буде розроблено в межах цієї дисертації.

1.3. Виявлення недоліків аналогів та формування вимог до системи

1.3.1. Аналіз обмежень поточних рішень

На основі порівняльного аналізу (див. п. 1.3) виявлено низку спільних недоліків, що обмежують застосування наявних рішень у завданнях реального часу та глибокого аналізу мультиджерельної інформації:

- Закритість та комерційна недоступність: більшість потужних систем (наприклад, Dataminr, Meltwater) функціонують як комерційні SaaS-продукти без можливості кастомізації чи локального розгортання.
- Фрагментарність збору джерел: системи або орієнтовані виключно на соціальні медіа, або лише на новинні API, не підтримуючи одночасне потокове спостереження за кількома категоріями джерел.
- Відсутність повноцінного семантичного аналізу: сучасні рішення здебільшого обмежуються ключовими словами, метаданими та простими правилами фільтрації, без глибинного embedding-аналізу чи кластеризації подій за сенсом.
- Неможливість локального використання: з міркувань безпеки або відповідності нормам (наприклад, в урядових структурах), локальне розгортання часто є обов'язковим, однак чинні системи не підтримують такого сценарію.
- Обмежена адаптивність до контексту: системи слабо підтримують контекстну фільтрацію за країною, мовою, часовим горизонтом або подієвою значущістю, що робить їх непридатними для ситуаційного реагування (наприклад, кризового моніторингу).

1.3.2. Системні можливості для поліпшення

Виходячи з вищезазначених обмежень, окреслюється потреба у розробці нової платформи, яка б включала:

- Підтримку потокових джерел даних у реальному часі: Telegram, Twitter/X, RSS-агрегатори, новинні API (GNews[13], Newsdata[14], EventRegistry)
- Семантичну обробку: кластеризацію подій за embedding-моделями (наприклад, SBERT, GTE) [15], об'єднання дублікатів, виявлення схожих повідомлень.
- Виявлення та ранжування інформаційних трендів: інтеграція моделей трендовості (наприклад, spike detection, moving average growth)[16], що дозволить відслідковувати нові інфохвилі до їхньої масової появи.
- Адаптивну, відкриту архітектуру: підтримка локального запуску на сервері або в хмарному середовищі з можливістю розширення (модулі, плагіни, API).
- Інтеграцію аналітики та інтерфейсів візуалізації: побудова графів подій, часових ліній, heatmap з трендами, що дозволить легко інтерпретувати динаміку подій.

1.3.3. Формування загальних вимог до системи

Таблиця 1.5

Формування загальних вимог до системи

Категорія	**Вимоги**
Джерела даних	Telegram API, Twitter/X API, RSS, RESTful новинні API

Таблиця 1.5(продовження)

Формування загальних вимог до системи

Категорія	Вимоги
Обробка	NLP-пайплайн, семантичне групування, скоринг важливості подій
Інфраструктура	Docker-орієнтована, масштабована, з можливістю кастомізації
Безпека	Мінімальне збереження персональних даних, локальний деплой
Інтерфейс	Панель візуалізації, графи подій, вивантаження у CSV/JSON
Модульність	Можливість додавати нові джерела та моделі без зміни ядра

Таблиця 1.6

Формування функціональних вимог до системи

№	Назва	Пріоритет	Ризики
1	Система збору текстових повідомлень із зовнішніх джерел	Високий	Високий
1.1	Періодичне отримання даних з Telegram, Reddit, X/Twitter	Високий	Високий

Таблиця 1.6(продовження)

Формування функціональних вимог до системи

№	**Назва**	**Пріоритет**	**Ризики**
1.2	Збереження сирих повідомлень у сховищі даних	Високий	Середній
2	Підсистема попередньої обробки текстів	Високий	Середній
2.1	Очищення тексту від технічних елементів та нормалізація	Високий	Низький
3	Підсистема класифікації повідомлень	Високий	Високий
3.1	Віднесення повідомлень до однієї з фіксованих категорій	Високий	Високий
4	Підсистема лексичного аналізу	Високий	Середній
4.1	Обчислення частот слів і словосполучень за періодами	Високий	Середній
4.2	Побудова хмар слів у розрізі дат, джерел та категорій	Середній	Низький
5	Підсистема семантичного аналізу	Високий	Високий
5.1	Обчислення ембеддингів текстових повідомлень	Високий	Високий

Таблиця 1.6(продовження)

Формування функціональних вимог до системи

№	**Назва**	**Пріоритет**	**Ризики**
5.3	Формування інформаційних подій з кластерів повідомлень	Високий	Високий
6	Підсистема аналітики та візуалізації подій	Високий	Середній
6.1	Перегляд переліку подій із базовими атрибутами	Високий	Середній
6.2	Перегляд детального профілю події (динаміка, ключові слова)	Середній	Середній

Таблиця 1.7

Формування не функціональних вимог до системи

№	Назва	Пріоритет	Ризики
1	Продуктивність обробки потоків повідомлень	Високий	Високий
1.1	Оновлення даних та подій у межах припустимого часу затримки	Високий	Високий
2	Масштабованість системи	Високий	Середній
3	Надійність та відмовостійкість	Високий	Високий
3.1	Обробка збоїв джерел та повторний запуск процесів	Середній	Високий
4	Модульність та розширюваність архітектури	Високий	Середній
5	Відтворюваність результатів аналізу	Середній	Середній
6	Зручність інтерпретації результатів для користувача	Середній	Низький

1.4. Висновки до розділу 1

У першому розділі було здійснено комплексний огляд теоретичних засад, практичних підходів і існуючих рішень у сфері виявлення та аналізу інформаційних подій у реальному часі. Проведене дослідження дозволило зробити низку важливих висновків:

- Сучасний ринок рішень для моніторингу інформаційного простору є фрагментованим. Більшість доступних інструментів орієнтовані або на аналітику соціальних мереж, або на агрегування новин, рідко поєднуючи обидва джерела в єдиній системі.
- Існуючі системи мають значні обмеження у контексті відкритості, локального використання, масштабованості та рівня автоматизації. Часто вони функціонують як SaaS-продукти без можливості кастомізації або гнучкої інтеграції з іншими сервісами.
- Відсутні комплексні рішення, здатні уніфіковано обробляти, агрегувати та аналізувати інформаційні події з різних потокових джерел, включаючи Telegram, Twitter/X, RSS та новинні API, із глибоким семантичним розумінням змісту.
- Семантичні технології аналізу ще не стали стандартом у цій сфері. Більшість систем використовують ключові слова або базові фільтри, що обмежує якість виявлення подієвих кластерів, дублікатів та трендів.

Таким чином, сформульовано концептуальне обґрунтування доцільності розробки нової інтелектуальної системи, що дозволить покрити виявлену функціональну та технологічну нішу. Подальші розділи дисертації будуть присвячені постановці задачі, розробці архітектури, реалізації програмного рішення та його тестуванню.

РОЗДІЛ 2

ПРОЄКТУВАННЯ ІНТЕЛЕКТУАЛЬНОЇ СИСТЕМИ ТА ЇЇ АРХІТЕКТУРНИХ КОМПОНЕНТІВ

2.1 Концептуальна модель інформаційних подій та вимоги до системи

Ключова ідея проєкту полягає в поєднанні лексико-класифікаційних методів (класифікація повідомлень, підрахунок частот слів, побудова хмар слів) із семантичним аналізом на основі векторних подань текстів. Такий підхід дозволяє не лише описувати окремі повідомлення, а й виявляти стійкі інформаційні події, що розгортаються у часі. У межах системи інформаційна подія розглядається як узагальнена сутність, що об’єднує семантично близькі повідомлення про один і той самий факт, тему або процес у межах обмеженого часового інтервалу.

На лексико-класифікаційному рівні потік повідомлень розглядається як сукупність окремих текстів, кожному з яких призначається певний тип інформаційної активності (наприклад, політична, економічна тощо). На основі текстів будується агрегована лексична картина: частотні списки слів, хмари слів, розподіли за категоріями й джерелами. Такий опис дозволяє окреслити загальний фон інформаційного простору та зафіксувати зміни, що можуть свідчити про зародження або посилення певних інформаційних процесів.

Семантичний рівень аналізу орієнтований не на окремі повідомлення, а на їхні групи, які описують один і той самий факт чи тему. На цьому рівні система працює з поняттям «інформаційна подія» як узагальненою сутністю, що поєднує семантично близькі повідомлення в межах обмеженого часового інтервалу. Для події фіксуються її часові характеристики, домінувальні теми та ключові слова, а також зв’язок із джерелами й типами повідомлень. Алгоритмічні аспекти реалізації обох рівнів аналізу розкриваються у розділах 2.3–2.4.

2.2 Архітектура системи та її основні компоненти

2.2.1 Загальна архітектурна схема системи

Для розроблюваної системи обрано мікросервісну архітектуру. Таке рішення обумовлене потребою у високій масштабованості, гнучкому управлінні окремими компонентами, а також спрощенні процесів розробки та тестування за рахунок декомпозиції функціоналу на незалежні сервіси. Кожен модуль відповідає за чітко визначену підзадачу, а взаємодія між ними здійснюється через формалізовані інтерфейси. Високорівнево архітектуру вебзастосунку подано у вигляді діаграми контексту на рисунку 2.1.

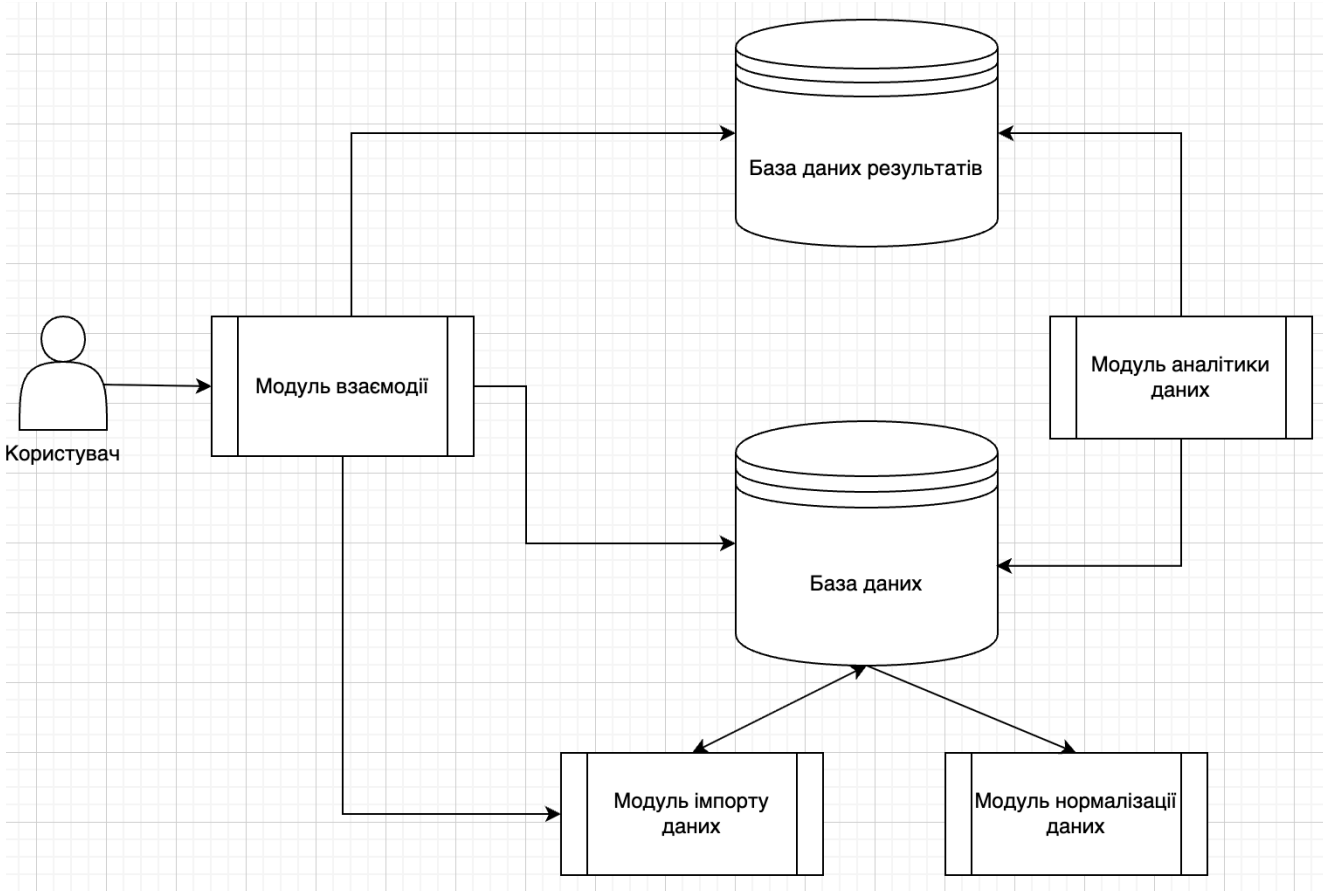

Рис.2.1 Високорівнево архітектуру вебзастосунку

Система побудована з кількох окремих модулів, що утворюють конвеєр обробки даних.

Логіку роботи системи зручно описати у вигляді послідовного конвеєра, який складається з кількох основних етапів:

Ingestion → Classification and/or Semantic Analysis → Storage → Reports.

Ця послідовність відображається на структурній схемі системи (рисунок 2.2) та визначає напрямок руху даних між підсистемами.

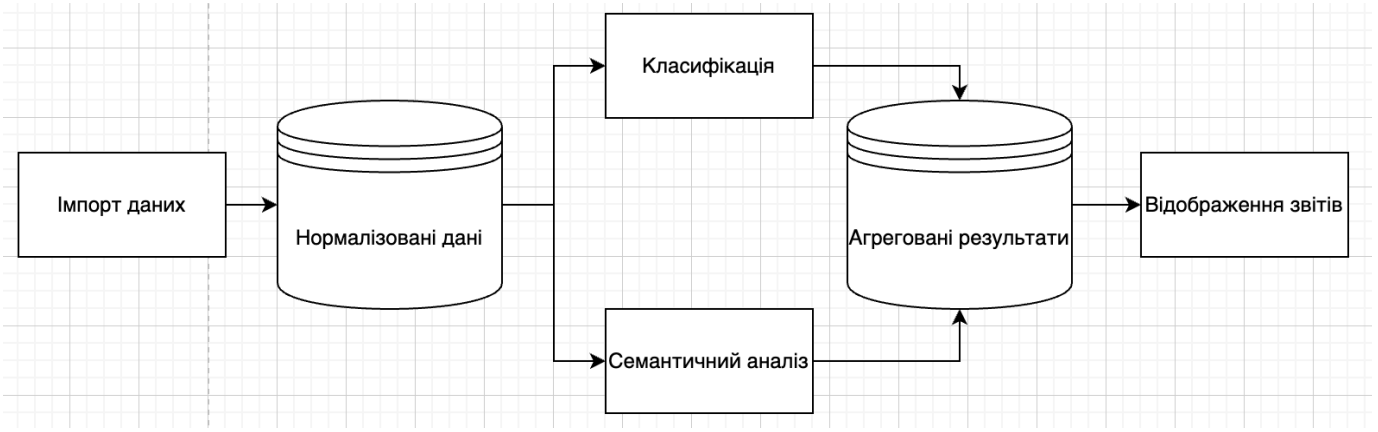

Рис 2.2 Структуран схема системи

На етапі забору та нормалізації даних працює підсистема імпорту та обробки даних. Вона включає конектори до Telegram, Reddit та X/Twitter, що взаємодіють із відповідними API або іншими механізмами доступу до контенту. Повідомлення, отримані з різних джерел, одразу приводяться до єдиного внутрішнього формату завдяки цьому подальші модулі можуть працювати з уніфікованою структурою, незалежно від специфіки конкретної платформи.

Далі реалізується етап класифікації. На цьому кроці тексти проходять попередню лінгвістичну обробку та подаються на вхід класифікаційній моделі. Модель відносить повідомлення до однієї з фіксованих категорій інформаційних подій. Отриманий клас додається як атрибут до кожного повідомлення та зберігається разом із іншими метаданими. Класифікація слугує основою для подальшої тематичної агрегації й відбору релевантних повідомлень під час формування подій.

На етапі семантичного аналізу підсистема використовує дані зі сховища для побудови ембеддингів, оцінки семантичної близькості між повідомленнями та кластеризації текстів у межах заданих часових вікон. У результаті утворюються кластери, які інтерпретуються як інформаційні події. Для кожної події агрегуються ключові слова, формується хмара слів, визначаються часові характеристики та

розподіл за джерелами й категоріями. Саме на цьому рівні відбувається перехід від розрізнених повідомлень до структурованих об’єктів «інформаційна подія».

Етап зберігання даних реалізується в межах центрального сховища даних. Тут накопичуються сирі повідомлення, результати класифікації, частотні характеристики лексики, а також семантичні подання текстів. Сховище виступає єдиним джерелом правди (single source of truth) для всіх підсистем, що звертаються до даних: семантичного аналізу, аналітики, візуалізації. Завдяки цьому забезпечується консистентність результатів і можливість повторного аналізу історичних даних.

Завершальний етап побудови та відображення звітів реалізується підсистемою аналітики та представлення результатів. Вона звертається до сховища й результатів семантичного модуля, формує списки актуальних та архівних подій, будує часові графіки, хмари слів та інші візуальні елементи. Веб-застосунок, що входить до цієї підсистеми, надає користувачеві можливість переглядати зведену інформацію про події, деталізувати їх до рівня окремих повідомлень, аналізувати динаміку ключових слів та структуру інформаційних хвиль у різні періоди.

Інтеграція різних джерел даних у такій архітектурі є частиною підсистеми імпорт та нормалізації даних і повністю ізолюється від решти логіки. Модулі семантичного аналізу та аналітики працюють уже з абстрактним поняттям «повідомлення», не залежачи від того, з якої саме платформи воно отримане. Це спрощує підтримку системи: зміни в API окремих платформ або додавання нових джерел не потребують модифікації класифікаційних і аналітичних модулів, а обмежуються переробкою відповідного конектора в підсистемі збору даних.

2.3 Підсистема збору та нормалізації даних

Підсистема збору та нормалізації даних відповідає за перетворення неоднорідного потоку повідомлень із різних платформ (Telegram, Reddit, X/Twitter) на уніфікований набір текстів, придатний для подальшої класифікації та семантичного аналізу. На цьому рівні система працює безпосередньо з зовнішніми

веб-сервісами, використовуючи HTTP-протокол та офіційні або документовані API відповідних платформ. Для Telegram передбачається використання Telegram Bot API або клієнтської API-бібліотеки, яка дозволяє отримувати повідомлення з каналів і чатів; для Reddit та X/Twitter — REST-інтерфейси, що повертають дані у форматі JSON. Отримані відповіді перетворюються у внутрішні транспортні об’єкти, де фіксуються текст, час публікації, ідентифікатори автора чи каналу, а також службова інформація про джерело.

Організація обробки даних у межах підсистеми поєднує елементи підходів ETL та ELT. На першому кроці (*Extract/Load*) конектори періодично виконують HTTP-запити до API зовнішніх платформ, отримують дані у вигляді JSON та з мінімальними перетвореннями зберігають їх у так званій “сирій” зоні сховища. Це відповідає ELT-підходу, коли пріоритет надається швидкому та повному завантаженню вихідних даних. На другому кроці (*Transform*) виконується побудова уніфікованої моделі «повідомлення» та нормалізація текстових і часових полів. Такий розподіл дозволяє за потреби повторно програти етапи трансформації або застосувати альтернативні алгоритми обробки без необхідності повторно звертатися до зовнішніх API.

Нормалізація тексту спрямована на усунення технічних артефактів і приведення повідомлень до єдиного стандарту обробки, що є критично важливим для роботи подальших NLP-модулів. Для кожного отриманого тексту послідовно виконуються такі дії:

– видалення службових елементів (URL-адреси, зайва розмітка, спеціальні символи, повторювані пробіли);

– приведення тексту до єдиного регістру (зазвичай нижнього) для зменшення варіативності написання;

– уніфікація та нормалізація часових міток до спільного часового поясу та єдиного формату;

– базова фільтрація тривіальних або надто коротких повідомлень, що не несуть змістового навантаження;

– збереження очищеного тексту окремо від сирого варіанту для можливості повторного аналізу й тестування інших сценаріїв обробки.

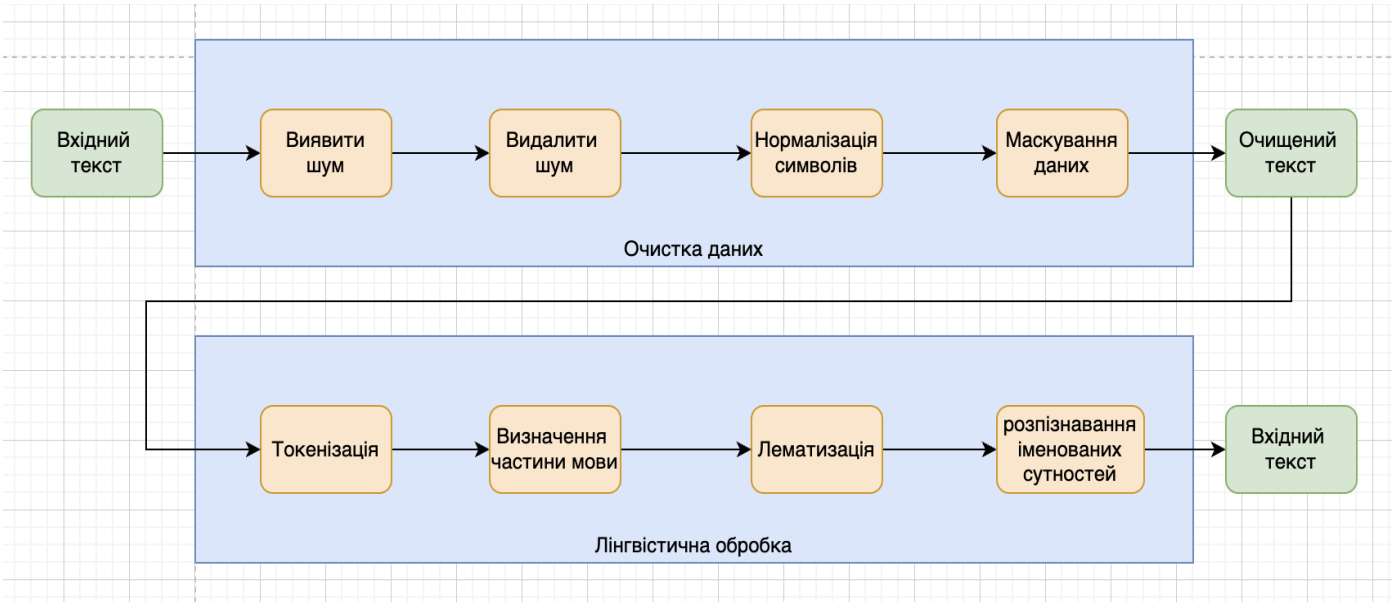

Рис.2.3 Процес очистки та нормалізації тексту

Комунікація між підсистемою збору та подальшими компонентами системи здійснюється через чітко визначений інтерфейс даних. На виході підсистеми формуються записи у внутрішній таблиці або колекції «повідомлень» із фіксованим набором полів (ідентифікатор, джерело, час, текст у сирому та нормалізованому вигляді, службові атрибути). Доступ до цих даних може надаватися як безпосередньо через базу даних, так і через внутрішні REST-ендпоїнти або черги повідомлень, залежно від конкретної реалізації мікросервісної архітектури. Результатом роботи підсистеми збору та нормалізації є впорядкований масив повідомлень із чітко визначеною структурою та очищеною текстовою складовою, який у подальшому використовується підсистемами класифікації та семантичного аналізу (розділ 2.4).

2.4 Підсистеми аналізу текстових повідомлень

Підсистеми аналізу текстових повідомлень утворюють логічне продовження підсистеми збору та нормалізації даних. Їх завдання полягає в тому, щоб із потоку вже підготовлених, очищених текстів сформувати більш високорівневі сутності — тематично класифіковані повідомлення та інформаційні події. У межах системи

виділяються два тісно пов’язані рівні аналізу: підсистема класифікації, що працює на рівні окремих повідомлень, та підсистема семантичного аналізу, яка оперує сукупностями повідомлень і формує з них події.

2.4.1 Підсистема класифікації повідомлень

Підсистема класифікації повідомлень використовує нормалізовані тексти, сформовані підсистемою збору та нормалізації, та доповнює їх тематичною ознакою. Для кожного повідомлення необхідно визначити, до якого типу інформаційної активності воно належить. У межах системи прийнято фіксований набір із п’яти класів, що відображають основні напрями новинного потоку:

- **crisis (криза)** — повідомлення про бойові дії, теракти, техногенні чи природні катастрофи, інші події, що призводять до значних людських втрат або суттєвих руйнувань;
- **economical (економіка)** — новини про фінансові ринки, санкційну політику, промислове виробництво, оборонно-промисловий комплекс та інші аспекти економічної діяльності;
- **corruption (корупція)** — повідомлення про корупційні практики, зловживання владою, незаконне привласнення ресурсів на державному чи побутовому рівні;
- **political (політика)** — політичні заяви, рішення та події, які не потрапляють до попередніх категорій, але мають виражений політичний характер;
- **other (інше)** — усі інші типи новин (спорт, кримінал, технології, культурні події тощо), що не належать до базових чотирьох категорій, але становлять загальний фон інформаційного простору.

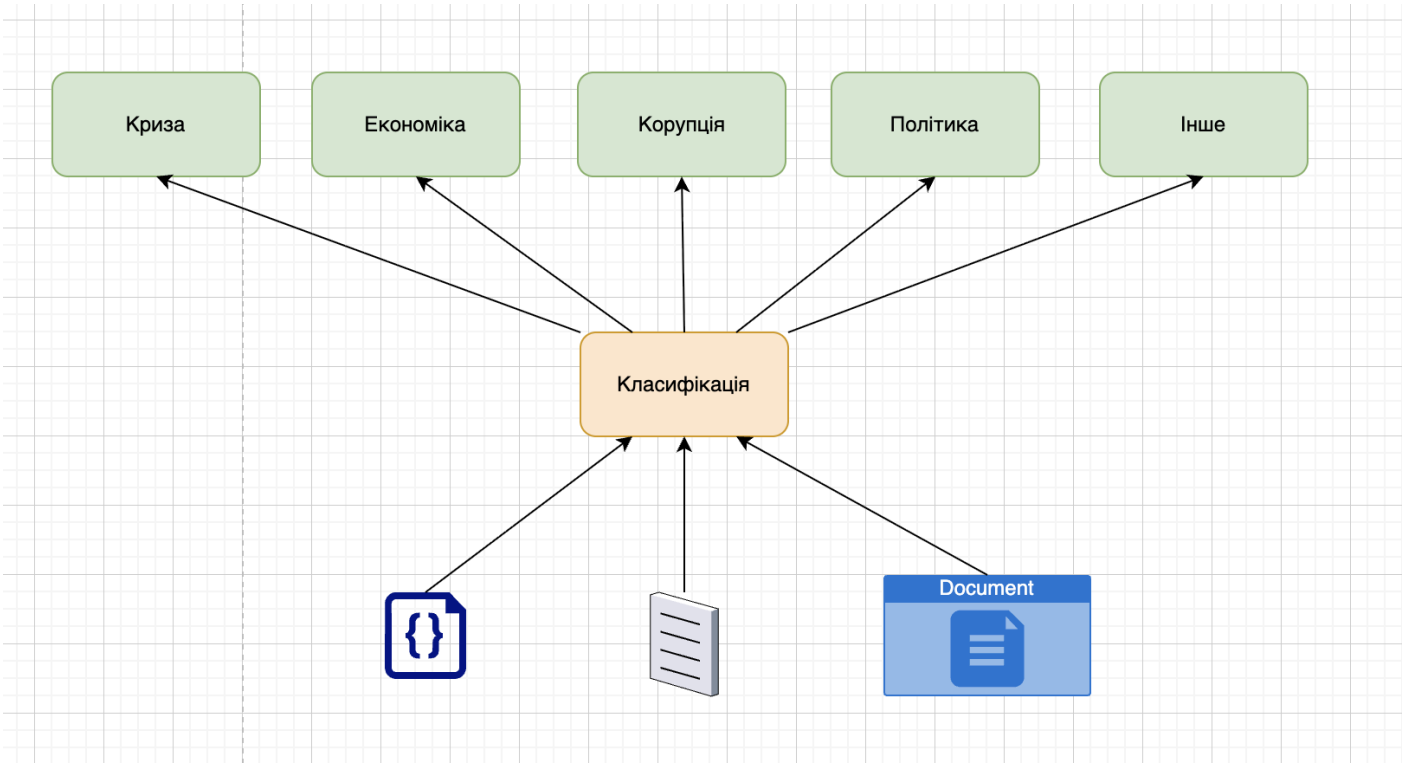

Рис.2.4 Загальна схема роботи класифікації

На вхід класифікаційної підсистеми надходить очищений текст повідомлення разом із базовими метаданими (джерело, час публікації, мова тощо). Далі виконується перетворення тексту у форму, придатну для роботи моделі: застосовується єдиний для всієї системи конвеєр попередньої лінгвістичної обробки (нормалізація, токенізація, після чого текст представляється у вигляді векторних ознак. Це можуть бути як класичні представлення (на кшталт TF–IDF[17]), так і ембеддинги, що безпосередньо використовують розподілені подання слів або речень.

Класифікаційна модель, навчена на розміченій вибірці, для кожного повідомлення повертає ймовірнісний розподіл по п’яти класах, а також базовий прогноз — клас із максимальною ймовірністю. Обраний клас зберігається як атрибут повідомлення поряд з іншими метаданими, а додаткова інформація (наприклад, значення ймовірності або альтернативний клас) може використовуватися для аналізу неоднозначних випадків. Результати класифікації застосовуються у двох напрямах: для побудови тематичних статистик (динаміка категорій у часі, відмінності між

джерелами) та як додаткова ознака в підсистемі семантичного аналізу під час формування інформаційних подій.

Таким чином, підсистема класифікації забезпечує перший рівень семантичного узагальнення: кожне повідомлення отримує свій тип інформаційної події, що дозволяє структурувати загальний потік текстів і виступає основою для подальшої агрегованої аналітики.

2.4.2 Підсистема семантичного аналізу інформаційних подій

Підсистема семантичного аналізу розширює результати класифікації, переходячи від окремих повідомлень до рівня інформаційних подій. Її мета — виявити групи семантично пов’язаних повідомлень, що описують один і той самий факт, процес або тему в межах певного часового інтервалу, та надати їм узгоджене представлення.

Вихідними даними для семантичного модуля є нормалізовані повідомлення з атрибутами, включно з результатами класифікації. Для кожного тексту будується векторне подання (ембеддинг), яке відображає зміст повідомлення у багатовимірному семантичному просторі. Для цього використовується модель глибинного навчання, здатна відображати схожі за змістом тексти у близькі точки простору незалежно від конкретної лексичної форми. Обчислені ембеддинги зберігаються у сховищі та стають основою для подальших алгоритмів групування.

Аналіз виконується в межах обраних часових вікон. Для кожного інтервалу формується множина повідомлень (за потреби — відфільтрованих за категорією, джерелом або іншими ознаками), і їхні ембеддинги подаються на вхід алгоритму кластеризації. Отримані кластери інтерпретуються як кандидати в інформаційні події. Кластер визнається подією, якщо він задовольняє визначені критерії: містить достатню кількість повідомлень, демонструє характерний профіль у часі (наприклад, сплеск активності) та має змістову цілісність. На цьому етапі можуть використовуватися результати класифікації — наприклад, для обмеження аналізу певною тематичною категорією або для аналізу подій на стику кількох типів.

Для кластера, визнаного інформаційною подією, формується окремий об’єкт «подія». Для нього визначаються часові межі, кількість і розподіл повідомлень за джерелами та категоріями, а також лексичний профіль. Лексичний профіль включає частотні списки слів і словосполучень, а також хмару слів, що дозволяє візуально оцінити домінуючу лексику в межах події.

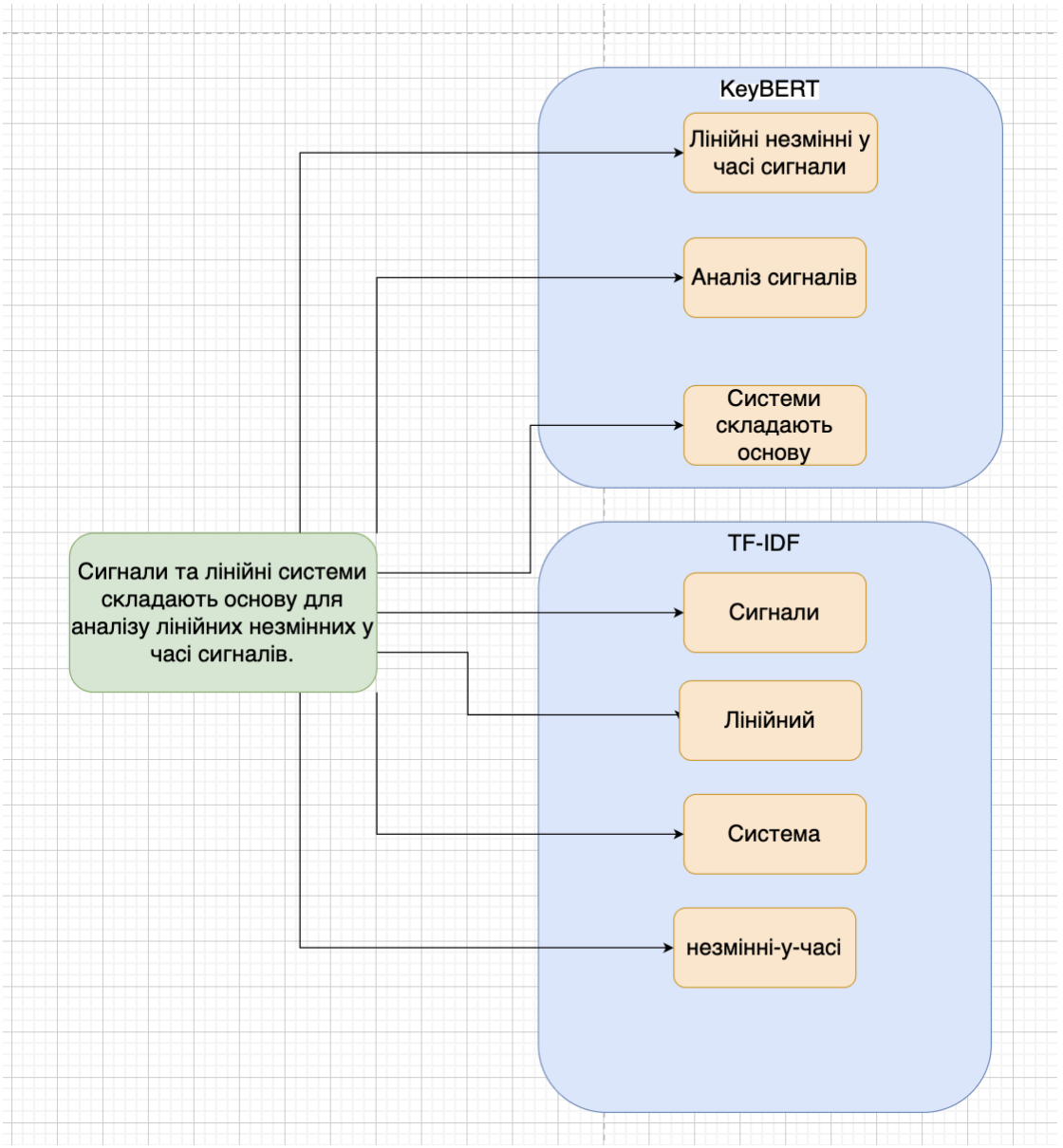

Рис.2.5. Варіанти виокремлення семантичних значень

2.5 Обґрунтування вибору засобів розробки

Під час розроблення веб-застосунків важливу роль відіграє вибір бібліотек і фреймворків, які забезпечують реалізацію як клієнтської, так і серверної логіки — від побудови інтерфейсу користувача до обробки HTTP-запитів на бекенді. Конкретний стек технологій визначається з урахуванням функціональних можливостей інструментів, активності спільноти розробників, наявності супровідної документації та успішного досвіду застосування в подібних проєктах.

Для створення клієнтської частини застосунку доцільно розглядати фреймворки React [18] та Angular [19]. React орієнтований на побудову інтерактивних інтерфейсів за допомогою компонентного підходу, має розвинену екосистему додаткових бібліотек і активно підтримується спільнотою. Angular, своєю чергою, є більш «важкою» та всеосяжною платформою, що надає вбудовані засоби для організації структури проєкту, тестування та побудови великих корпоративних рішень.

У межах даної роботи для фронтенду було обрано React, оскільки він поєднує високу гнучкість, достатню продуктивність та широке поширення серед розробників. Як бібліотека, спочатку розроблена компанією Facebook, React підтримує компонентну архітектуру, що спрощує поділ інтерфейсу на логічні частини, полегшує їх повторне використання та обслуговування в довгостроковій перспективі.

Під час вибору технологій для серверної частини насамперед визначаються з мовою програмування, оскільки саме вона задає екосистему інструментів, принципи розробки та особливості масштабування. Найпоширенішими серед сучасних бекенд-рішень є JavaScript (у середовищі Node.js), Python, .NET та Java. Кожна з цих платформ має власні сильні сторони й типові сфери застосування, що впливають на доцільність вибору у конкретному проєкті.

Node.js використовується там, де потрібна висока швидкість обробки великої кількості паралельних запитів, наприклад у веб-сервісах з інтенсивною мережею взаємодій. Його неблокуюча модель виконання забезпечує низькі затримки, а

спільність мови з фронтендом спрощує єдину кодову базу. Python, навпаки, найчастіше застосовується в системах, де ключовим фактором є наявність потужних бібліотек для штучного інтелекту, аналітики та машинного навчання. Він забезпечує швидку розробку та високу читабельність коду, що робить його зручним для прототипування та інтеграції моделей.

Платформа .NET орієнтована на побудову продуктивних і надійних серверних застосунків, часто у корпоративних середовищах. Вона пропонує розвинуту інфраструктуру, інструменти та підтримку великомасштабних рішень. Java традиційно використовується для складних, масштабованих і довготривалих у підтримці систем завдяки стабільності, широкій екосистемі фреймворків та можливості запуску на різних операційних системах.

У підсумку вибір технологій залежить від специфіки задач: інтенсивні веб-інтерфейси краще реалізовувати за допомогою Node.js, аналітичні системи — на Python, великі корпоративні сервіси — на .NET або Java. Кожна платформа є зрілою та здатною забезпечити надійний серверний фундамент, важливо лише співвіднести її можливості з потребами проєкту.

З урахуванням специфіки проєкту було прийнято рішення використовувати Python як основну мову для серверної частини. Це обґрунтовується насамперед наявністю зрілої екосистеми для обробки текстів, аналітики та машинного навчання, що критично важливо для реалізації завдань, пов’язаних із семантичним аналізом та класифікацією. Додатковою перевагою є зрозумілий синтаксис, який спрощує підтримку коду та знижує поріг входу для нових учасників проєкту.

У процесі проєктування серверної частини важливим етапом є визначення веб-фреймворку, який відповідатиме за організацію маршрутизації, структурування логіки застосунку та обробку HTTP-запитів. Python має кілька популярних рішень для побудови бекенду, і кожне з них формує різну архітектурну модель. Django пропонує повноцінний високорівневий набір інструментів «із коробки», що дозволяє швидко створювати комплексні вебзастосунки без необхідності підбирати додаткові компоненти. Flask, навпаки, дає лише мінімальну основу, забезпечуючи максимальну

гнучкість, однак потребує від команди більше рішень щодо структури та вибору сторонніх бібліотек.

FastAPI вирізняється орієнтацією на сучасні підходи до побудови API та поєднує декларативне описання даних, вбудовану асинхронність і автоматичну генерацію документації. Його продуктивність та зручність роботи з типізованими структурами роблять його особливо привабливим для систем, де критичними є швидкість обробки запитів і коректність передавання даних між компонентами. Саме тому FastAPI було обрано як основний фреймворк для бекенду: він дозволяє реалізувати модульну архітектуру, ефективно працює з великими обсягами текстових даних та забезпечує високу якість API без додаткових накладних витрат.

Для розробки було використано інтегровані середовища PyCharm (для серверної частини на Python) та WebStorm (для клієнтської частини на основі React). Обидва середовища, надають розширені можливості для навігації по коду, рефакторингу, відлагодження та роботи з системами керування залежностями й версіями. PyCharm оптимізований для роботи з Python-проєктами та підтримує типові сценарії розробки бекенду, а WebStorm орієнтований на сучасні JavaScript/TypeScript-стеки та фреймворки. Використання цих IDE підвищує продуктивність розробки та забезпечує комфортні умови для реалізації функціональності клієнтської й серверної частин застосунку.

У процесі розроблення серверної частини програмного забезпечення було використано багаторівневий підхід, який спрощує загальну структуру системи, підвищує її модульність та полегшує подальший супровід і розвиток. У межах такого підходу кожен мікросервіс логічно поділяється на чотири основні рівні: рівень представлень, сервісний рівень, доменний рівень та рівень репозиторіїв.

Рівень представлень (Presentation Layer) є верхнім шаром, що відповідає за приймання зовнішніх запитів та їх початкову перевірку. У межах розроблюваної системи цей рівень реалізовано у вигляді обробників HTTP-запитів, а також компонентів, які приймають та опрацьовують повідомлення, що надходять через RabbitMQ.

Сервісний рівень (Service Layer) містить основну бізнес-логіку застосунку та правила опрацювання даних. Саме на цьому рівні сервіси координують виконання операцій, використовуючи доменні моделі та звертаючись до репозиторіїв для читання й запису інформації.

Доменний рівень (Domain Layer) виступає ядром архітектури, оскільки описує ключові бізнес-сутності та моделі, з якими працює система. Тут задаються структура даних, їхні властивості та взаємозв'язки між ними. Відповідні доменні сутності та їхню взаємодію відображено на діаграмі (рисунок 3.6).

Рівень репозиторіїв (Repository Layer) забезпечує доступ до бази даних і реалізує шаблон репозиторію, який інкапсулює деталі виконання запитів. Завдяки цьому джерело даних або спосіб його зберігання можна змінити без необхідності модифікувати бізнес-логіку чи компоненти рівня представлень.

Висновки до розділу 2

У другому розділі дисертації було виконано комплексні заходи щодо конструювання програмного забезпечення. Задля досягнення високого рівня якості та функціональності продукту, спочатку було ретельно розроблено архітектуру програмного забезпечення, вибудовуючи її на багаторівневому підході, який дозволяє чітко відділяти різні компоненти системи та забезпечувати їх високу модульність та легкість у підтримці.

Вибір засобів розробки був обґрунтований на основі аналізу наявних інструментів та специфіки задач, що стоять при побудові даного проєкту. Зокрема, було визначено використання мови програмування Python з використанням фреймворку FastAPI для бекенду та React для фронтенду, забезпечуючи таким чином ефективність та сучасність вебзастосунку.

Ключову роль у процесі конструювання відіграло створення моделі класифікації новин, заснованої на архітектурі BERT. Було проведено детальну

розробку та тренування моделі, забезпечивши її здатність ефективно розподіляти новинні тексти за відповідними категоріями з високою точністю.

Розробка бази даних була зосереджена на використанні сучасних технологій, таких як PostgreSQL, та прийомів, які сприяють гнучкості обробки та зберігання даних. Спеціальна увага була приділена застосуванню структур, які підтримують JSONB формат, забезпечуючи необхідну динаміку та розширюваність системи.

Останнім, але не менш важливим етапом було проведення аналізу безпеки даних. Виявлення потенційних вразливостей та впровадження заходів щодо їх запобігання.

У сукупності цей розділ підтвердив, що розроблене програмне забезпечення відповідає сучасним стандартам якості та безпеки, надаючи надійний фундамент для його подальшого розширення та використання в різноманітних дослідницьких та аналітичних цілях.

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура системи агрегації

Незважаючи на наявність єдиної бази даних та єдиної доменної моделі, структура кодової бази застосунку чітко розділена на 4 основні частини, відповідно до раніше зазначених:

- система авторизації та управління звітами
- система управління моделями та шаблонами візуалізації
- система генерації звітів
- система аналізу даних

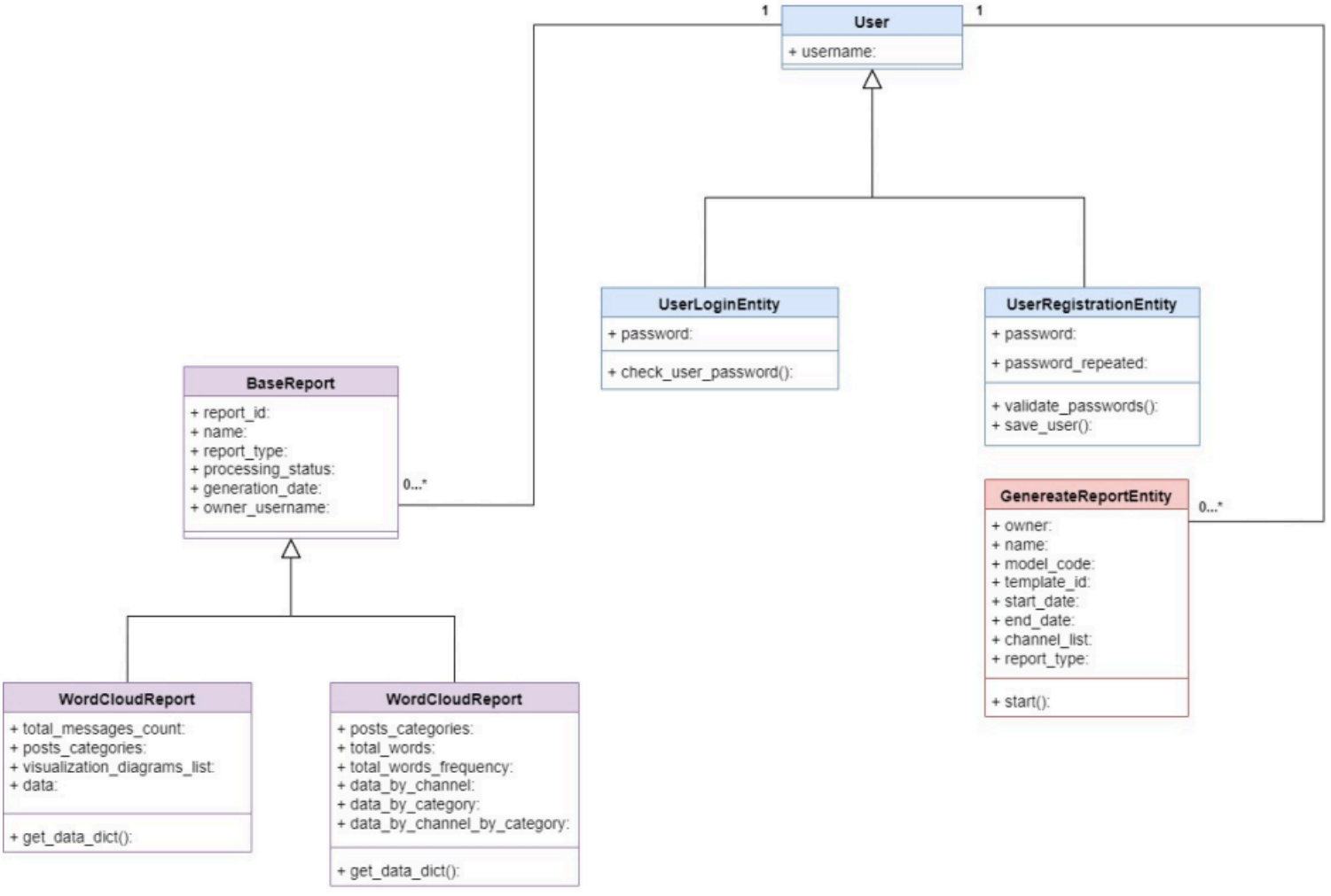

Рис. 3.1 Схема структури класів системи управління звітами та авторизації

Таблиця 2.1

Опис класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
User	Представляє користувача системи та пов’язує його зі звітами.	Містить інформацію про обліковий запис користувача (ідентифікатор, ім’я, хеш пароля) та виконує роль сутності, яка зв’язує користувача з його звітами. Забезпечує доступ до переліку звітів, створених цим користувачем.
UserLoginEntity	Обробляє дані під час авторизації користувача.	Використовується для приймання даних, які користувач вводить при вході: логін і пароль. Перевіряє коректність введених даних і передає їх далі для механізму аутентифікації. Не зберігається в БД — це службовий клас для логіну.
EntityUserRegistration	Обробляє дані під час реєстрації.	Приймає інформацію для створення нового користувача: ім’я, пароль, повтор пароля та інші параметри. Виконує валідацію, а також ініціює створення нового запису у таблиці користувачів. Також може містити логіку перевірки унікальності імені користувача.

Таблиця 2.1(продовження)

Опис класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
Generate Report Entity	Приймає параметри для створення нового звіту.	Використовується для передачі даних від клієнта до сервера, коли формується запит на генерацію звіту. Може містити діапазон дат, вибрані джерела, тип звіту (статистика чи хмара слів), ідентифікатор моделі тощо.
Base Report	Базовий клас, що містить спільні поля для різних типів звітів.	Описує параметри, які є спільними для всіх звітів: ідентифікатор, дата створення, власник, часовий діапазон, метадані. Забезпечує можливість уніфікованої обробки різних типів звітів у системі.
Statistical Report	Містить логіку для статистичних звітів.	Розширює BaseReport та додає поля і методи, потрібні для побудови статистичних діаграм: кількість постів за класами, розподіл за джерелами, часові ряди, графіки активності. Використовується модулем аналітики.
Word Cloud Report	Описує звіти у форматі хмари слів.	Також наслідує BaseReport. Зберігає інформацію про ключові слова, їхні ваги, джерела текстів, з яких побудована хмара слів. Використовується під час роботи семантичного модуля та для генерації відповідної візуалізації.

У структурі застосунку класи поділяються на три групи, кожна з яких виконує свою роль у загальній логіці роботи системи. Перша група стосується управління користувачами. Сутність User є основною моделлю, що зберігається у базі даних і описує інформацію про користувача, включно з тими звітами, які він створює. Поряд із цим існують допоміжні класи UserLoginEntity та UserRegistrationEntity. Це не моделі, а DTO-об’єкти, які використовуються для передачі даних між клієнтом і сервером під час логіну та реєстрації. Вони містять лише ті поля, які необхідні для виконання відповідних операцій, і не зберігаються в БД.

Друга група класів відповідає за роботу зі звітами. Для уніфікації їх структури використовується базовий клас BaseReport, який містить спільні параметри, притаманні будь-якому типу звіту. На його основі створюються два різні види звітів: StatisticalReport, що реалізує логіку формування статистичних даних і графіків, та WordCloudReport, який управляє звітами у форматі хмари слів. Запит на створення будь-якого з цих звітів надходить через спеціальний DTO-клас GenerateReportEntity, який збирає параметри, указані користувачем (тип звіту, часовий діапазон, джерела даних тощо), і передає їх у відповідний сервіс для генерації.

Третя група описує взаємозв’язки між вказаними класами. Один користувач може створювати необмежену кількість звітів різних типів, тому кожен об’єкт StatisticalReport або WordCloudReport містить посилання на користувача, який його сформував. Обидва ці класи наслідують BaseReport, що дозволяє обробляти їх уніфіковано там, де це необхідно. Процес створення звіту починається з формування об’єкта GenerateReportEntity, який визначає, який саме підтип звіту буде створено надалі та які параметри потрібно застосувати. Таким чином, усі класи працюють узгоджено: користувачі виконують операції реєстрації та авторизації, DTO[28] передають дані між клієнтською та серверною частинами, а звіти створюються та прив’язуються до користувача у спосіб, що забезпечує чисту логіку та розподіл відповідальностей між компонентами системи.

Представлення моделі класів системи управління моделями та шаблонами візуалізації зображене на рисунку 3.2.

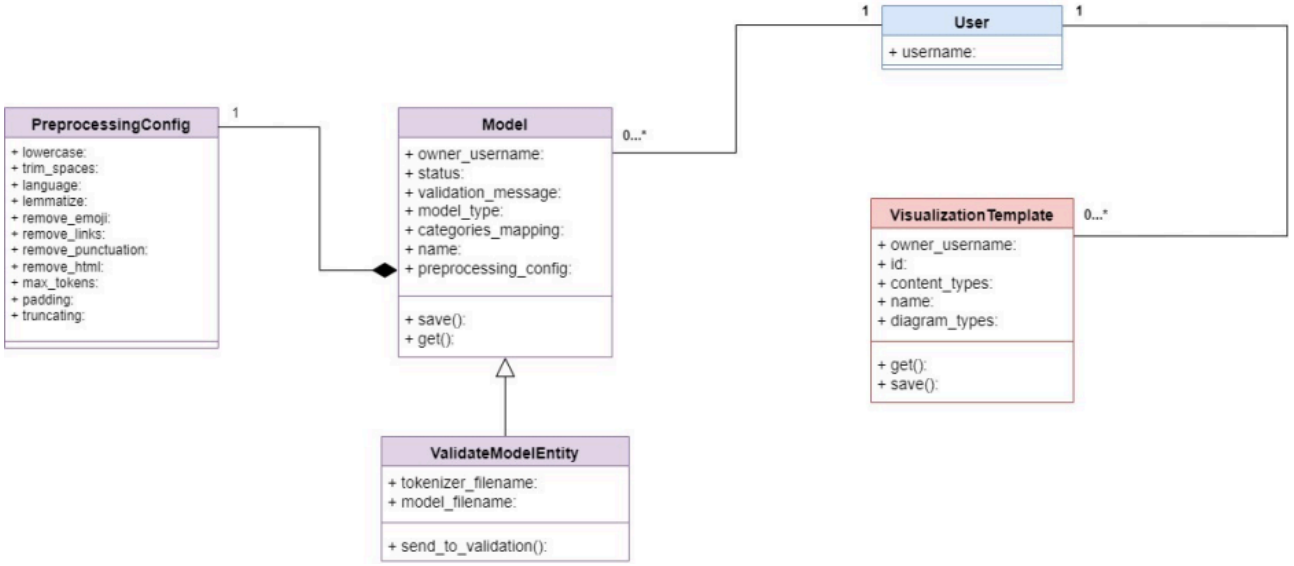

Рис. 3.2 Схема структури класів системи управління моделями та шаблонами візуалізації

Таблиця 2.2

Опис класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
User	Представляє користувача та зв’язує його з моделями й шаблонами.	Описує інформацію про користувача та містить зв’язки з об’єктами, які він створив: моделями ML та шаблонами візуалізації. Клас забезпечує ізоляцію ресурсів між користувачами та дозволяє кожному мати власний набір моделей і конфігурацій для аналізу.

Таблиця 2.2(продовження)

Опис класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
Model	Управляє машинними моделями, що використовуються в системі.	Містить метадані про завантажені або вбудовані моделі: тип, назву, версію, шлях до файлів, параметри моделі та налаштування інференсу. Дозволяє кожному користувачеві зберігати та керувати власними ML-моделями, а також визначати, яку модель застосувати під час генерації звітів.
ValidationModelEntity	DTO, що передає дані для валідації моделі користувача.	Використовується при завантаженні користувацької ML-моделі. Містить інформацію про файл моделі, формат, очікуваний інтерфейс та параметри. Передається в сервіс валідації, який перевіряє сумісність моделі з системою, її структуру та здатність до інференсу.
PreprocessingTextConfig	Налаштовує правила попередньої обробки текстів для конкретної моделі.	Містить конфігурації токенізації, очистки тексту, нормалізації, видалення стоп-слів, лематизації та інших операцій. Дозволяє використовувати різні pipelines препроцесингу для різних моделей, що забезпечує гнучкість у роботі з різними формами текстових даних.

Таблиця 2.2(продовження)

Опис класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
VisualizationStatsTemplate	Описує шаблони візуалізації, доступні конкретному користувачу.	Містить інформацію про тип візуалізації (графіки, word cloud, таблиці), доступні параметри й спосіб відображення даних у звітах. Шаблони можуть налаштовуватися користувачем, зберігатися в системі та використовуватися під час формування звітів.

Уся логіка роботи системи, що стосується моделей і візуалізацій, базується на узгодженій взаємодії цих класів.

Перш за все, User виступає центральною сутністю, яка володіє своїми моделями машинного навчання та власними шаблонами візуалізації. Коли користувач завантажує нову модель, дані про неї надходять через DTO-клас ValidateModelEntity, який передає необхідну інформацію до модуля перевірки. Після успішної валідації модель реєструється в системі у вигляді об’єкта Model, і відтоді вона може застосовуватись при генерації різних звітів.

Будь-яка модель працює на основі певних правил обробки текстів. Ці правила описуються в класі PreprocessingConfig, що дозволяє поєднувати одну модель з індивідуальним pipeline препроцесингу. Таким чином, різні моделі одного користувача можуть мати різні вимоги щодо очищення та підготовки вхідних даних.

Після того як модель створена та налаштована, користувач може створювати візуальні звіти. За те, як саме дані будуть подані у вигляді графіків чи хмари слів, відповідає клас VisualizationTemplate. Він визначає структуру, типи елементів, стиль та конфігурацію візуалізації. Шаблони належать конкретному користувачу, тож кожен може мати власні варіанти відображення.

У підсумку всі ці класи разом формують повний цикл роботи з моделями: від завантаження та валідації → через налаштування препроцесингу → до створення та використання візуальних шаблонів у звітах.

Уся система організована так, щоб кожен користувач міг мати власний набір моделей і конфігурацій, не перетинаючись з іншими, що забезпечує ізоляцію, кастомізацію та масштабованість.

Представлення моделі класів системи валідації моделей та генерації звітів зображене на рисунку 3.3.

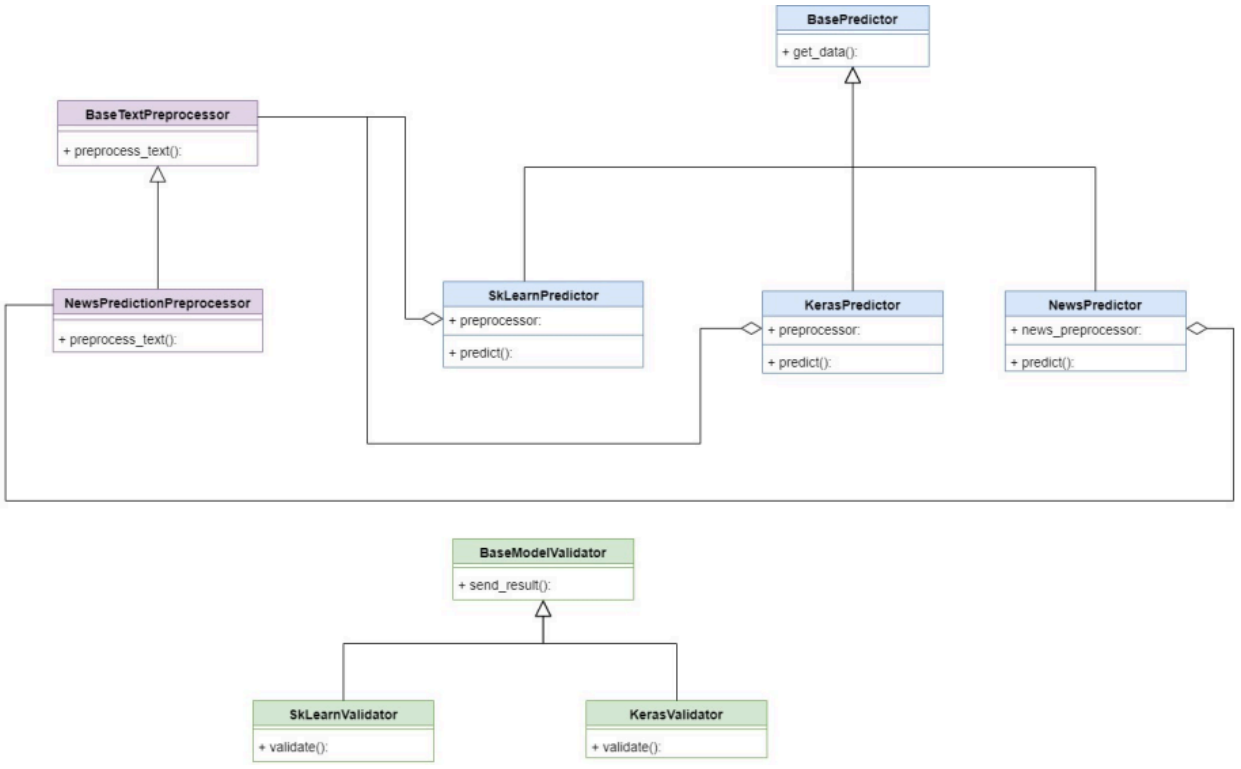

Рис. 3.3 Представлення моделі класів системи валідації моделей та генерації звітів

Таблиця 2.3

Представлення моделі класів системи валідації моделей та генерації звітів

Клас	Призначення (коротко)	Розширений опис
BasePredictor	Базовий клас для обробки даних та отримання передбачень.	Містить загальні методи для завантаження текстів, їх обробки, запуску моделі та формування результату. Є спільною основою для всіх типів предикторів, забезпечуючи єдиний інтерфейс роботи з даними.
SkLearnPredictor	Використовує завантажені користувачем sklearn-моделі для обробки текстів.	Реалізує специфічні механізми інференсу для моделей sklearn. Завантажує модель, застосовує відповідний препроцесинг і виконує передбачення для вхідних текстів.

Таблиця 2.3(продовження)

Структура класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
KerasPredictor	Використовує завантажені користувачем Keras-моделі.	Реалізує логіку роботи з keras-моделями: завантаження, обробка вхідних даних у форматах, сумісних з keras, та виконання інференсу. Підтримує як послідовні, так і функціональні моделі.
NewsPredictor	Використовує вбудовану модель класифікації новин.	Реалізує той самий інтерфейс передбачення, що й інші предиктори, але працює з уже вбудованою в систему моделлю. Використовується як дефолтне рішення для класифікації новинних текстів без необхідності завантажувати власні моделі.
BaseTextPreprocessor	Базовий клас для попередньої обробки текстів.	Містить загальні методи нормалізації, очистки, токенізації та трансформації текстів. Є фундаментом для створення більш спеціалізованих препроцесорів.

Таблиця 2.3(продовження)

Структура класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
NewsPredictionPreprocessor	Виконує спеціалізовану передобробку новинних текстів.	Реалізує правила очистки, токенізації, нормалізації, характерні саме для новинної моделі: видалення зайвих символів, обробка пунктуації, можливе перетворення у векторне подання тощо.
BaseModelValidator	Базовий клас для валідації завантажених моделей.	Містить спільні методи перевірки формату, структури та доступності моделі для інференсу. Забезпечує уніфіковану логіку валідації для різних типів моделей.

Таблиця 2.3(продовження)

Структура класів системи управління звітами та авторизації

Клас	Призначення (коротко)	Розширений опис
KerasValidator	Перевіряє коректність користувацьких Keras-моделей.	Виконує перевірку файлу моделі, структури шарів, можливості завантаження, відповідності очікуваному інтерфейсу та тестовий запуск для переконання, що модель працездатна в системі.
SkLearnValidator	Перевіряє коректність користувацьких sklearn-моделей.	Валідує формат Pickle-файлу, перевіряє наявність методу predict, сумісність із препроцесором і можливість виконання тестового передбачення.

Уся логіка роботи з машинним навчанням у системі поділяється на три рівні: передобробка текстів, валидація моделей, та виконання інференсу (predict).

На першому рівні працюють класи BaseTextPreprocessor та його спеціалізований нащадок NewsPredictionPreprocessor. Вони відповідають за приведення тексту до формату, з яким може працювати модель: очищення, токенізацію, нормалізацію чи перетворення у числове подання.

Другий рівень — це перевірка якості моделей, що завантажуються користувачами. Тут базовий клас BaseModelValidator задає загальні правила перевірки, а класи KerasValidator і SkLearnValidator реалізують конкретні механізми перевірки відповідно до типу моделі. Вони гарантують, що модель можна коректно

завантажити, що вона має потрібні методи й що здатна виконати тестове передбачення.

Третій рівень — це безпосередня робота моделей. Базовий клас BasePredictor визначає загальний інтерфейс для виконання передбачень: Від нього успадковуються конкретні реалізації, такі як SkLearnPredictor, KerasPredictor, NewsPredictor. SkLearnPredictor обробляє дані за допомогою sklearn-моделей. KerasPredictor — працює з keras-моделями. NewsPredictor — використовує вбудовану модель класифікації новин:

```python
class NewsClassificationBertPredictor(IPredictor):
    MAX_LENGTH = 360

    model_code: str = "bert_news_classification"
    mapping: dict[str, str] = {
        "0": "Corruption",
        "1": "Crisis",
        "2": "Economical",
        "3": "Other",
        "4": "Political",
    }

    def __init__(
        self,
        models_storage_path: str,
        report_type: ReportTypes,
        preprocessor: NewsClassificationPreprocessor,
    ) -> None:
        self._logger = logging.getLogger(self.__class__.__name__)
        self._predicted_items_counter = 0

        self._report_type = report_type
        self.preprocessor = preprocessor

        self.tokenizer = BertTokenizer.from_pretrained("bert-base-multilingual-uncased")

        news_classification_model_path = os.path.join(models_storage_path, "news-classification-model")
        self.model = load_model(news_classification_model_path, custom_objects={"TFBertModel": TFBertModel})

    def predict_post(self, entity: ClassificationEntity) -> str:
        if self._predicted_items_counter % 250 == 0:
            self._logger.info(
                f"[NewsClassificationBertPredictor] Predicted {self._predicted_items_counter} items so far..."
            )

        preprocessed_text = self.preprocessor.preprocess_text(
            entity.text,
            lemmatize=False,
        )

        input_ids, attention_mask = self._text_to_tokens(preprocessed_text)

        prediction_index = self.model.predict(
            {
                "input_ids": tf.expand_dims(input_ids, 0),
                "attention_mask": tf.expand_dims(attention_mask, 0),
            },
            verbose=False,
        ).argmax()

        self._predicted_items_counter += 1

        return self.mapping[str(prediction_index)]
```

Рис 3.4 Загальний код реалізації передбачення класу інформаційного поста

- SkLearnPredictor обробляє дані за допомогою sklearn-моделей;
- KerasPredictor — працює з keras-моделями;
- NewsPredictor — використовує вбудовану модель класифікації новин.

Таким чином, структура побудована так, щоб кожен тип моделі мав свій валідатор і свій предиктор, але працював через одну уніфіковану архітектуру. Це спрощує інтеграцію нових моделей, робить систему розширюваною та дозволяє зберігати однаковий workflow для всіх сценаріїв передбачення.

3.2 Розробка БД

Як основну систему керування базами даних застосовано PostgreSQL. Серверна база використовується для зберігання інформації про користувачів, сформовані звіти, шаблони візуалізації та параметри моделей. Структура та зміст кожної таблиці подано у наступних розділах.

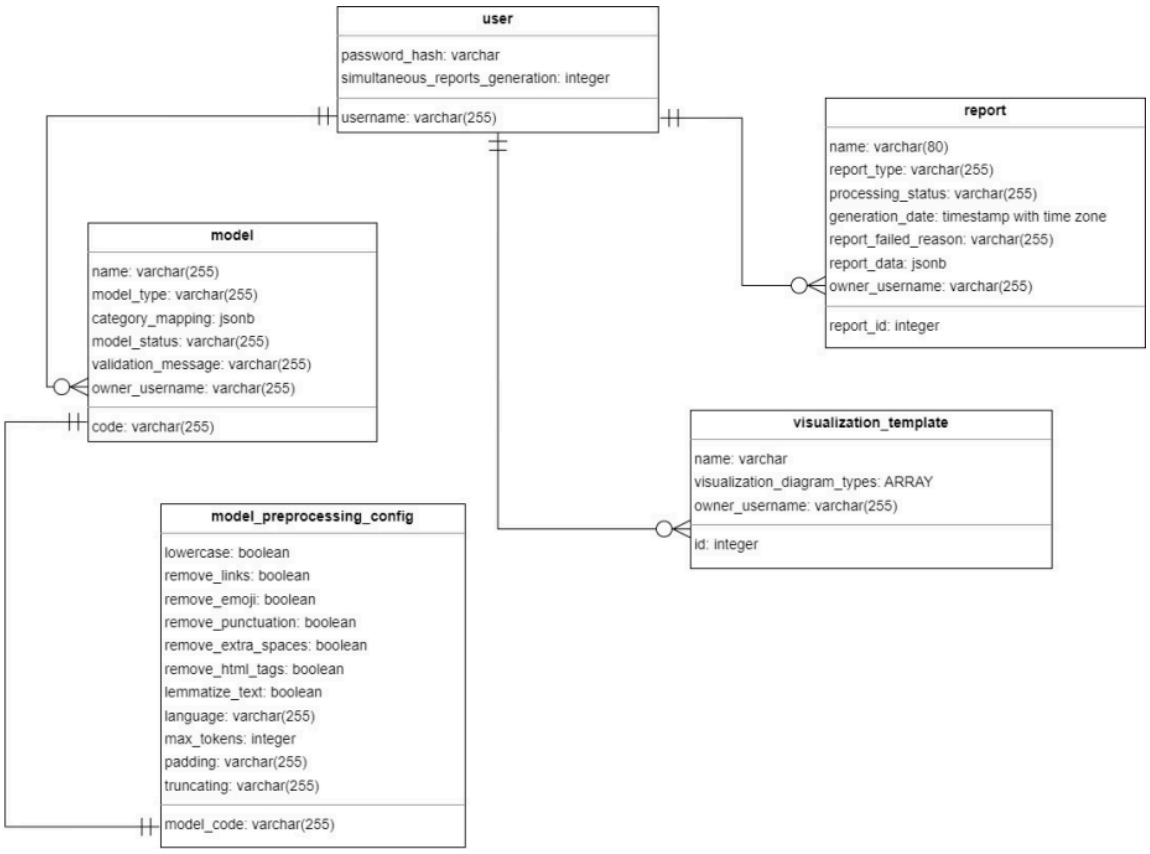

Рис. 3.5 Модель бази даних

Таблиця 3.1

Структура таблиці user

Назва поля	Опис
user_nickname	Власний нікнейм
user_password_hash	Хеш від паролю користувача

Таблиця 3.2

Структура таблиці report

Назва поля	Опис
report_id	Ідентифікаційний номер звіту
report_name	Назва звіту
report_type	Тип звіту
status	Статус генерації звіту
creation_date	Дата створення звіту

Таблиця 3.2(продовження)

Структура таблиці report

Назва поля	Опис
failed_reason	Опціональне поле, показує можливу помилку при генерації звіту, якщо така була
data	Агреговані та підраховані дані звіту
owner_username	Ім’я власника звіту

Таблиця 3.3

Структура таблиці model

Назва поля	Опис
code	Унікальний код моделі
name	Назва моделі
model_type	Вибір типу моделі
category_mapping	Назви категорій на які модель класифікує тексти
status	Статус валідності моделі
owner_username	Ім’я власника моделі

Таблиця 3.4

Структура таблиці visualization_template

Назва поля	Опис
id	ID шаблону візуалізації
diagrams_types	Список типів графіків шаблону
owner_username	Власник шаблону, ім`я

Таблиця 3.5

Структура таблиці model_preprocessing_config

Назва поля	Опис
model_code	Код моделі до якої відносяться налаштування
bool_lowercase	Марка чи треба приводити текст до нижнього регістру
bool_ remove_links	Марка чи треба видаляти посилання
remove_emoji_bool	Марка чи треба видаляти емоджі
remove_ punctuation_bool	Марка чи треба видаляти пунктуацію

Таблиця 3.5(продовження)

Структура таблиці model_preprocessing_config

Назва поля	Опис
extra_spaces_remove	Марка, чи зайві пробіли потрібно видалити
html_tags_remove	Марка чи потрібно видаляти HTML теги
lemmatization_text	Марка чи треба проводити лематизацію тексту
language	Оригінальна мова моделі
max_tokens_int	Максимальна кількість токенів, яку може отримати модель на вхід

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.6.

Таблиця 3.6

Опис утиліт

№ п/п	Назва утиліти	Опис застосування
1	PyCharm IDE	Головне середовище розробки програмного забезпечення серверної частини дипломної роботи.
2	WebStorm IDE	Головне середовище розробки програмного забезпечення клієнтської частини дипломної роботи.
3	Postman	Платформа для тестування API, що використовувалась для створення, надсилання та перегляду запитів на серверну частину дипломної роботи.

3.1 Конструювання модуля аналізу тектсу

3.3.1 Конструювання моделі класифікації новин

У якості ядра класифікаційного модуля в роботі використано модель BERT (Bidirectional Encoder Representations from Transformers[29]), яка побудована на багатошаровому енкодері трансформерного типу. Головна ідея полягає в тому, що вхідний текст послідовно проходить через низку енкодерних шарів, у результаті чого формується його компактне векторне подання фіксованої розмірності. Це векторне

представлення надалі використовується як вхід для надбудованого класифікаційного шару, що розв’язує вже конкретну задачу — віднесення повідомлення до однієї з наперед визначених категорій. Схематичну структуру такої моделі наведено на рисунку 3.6.

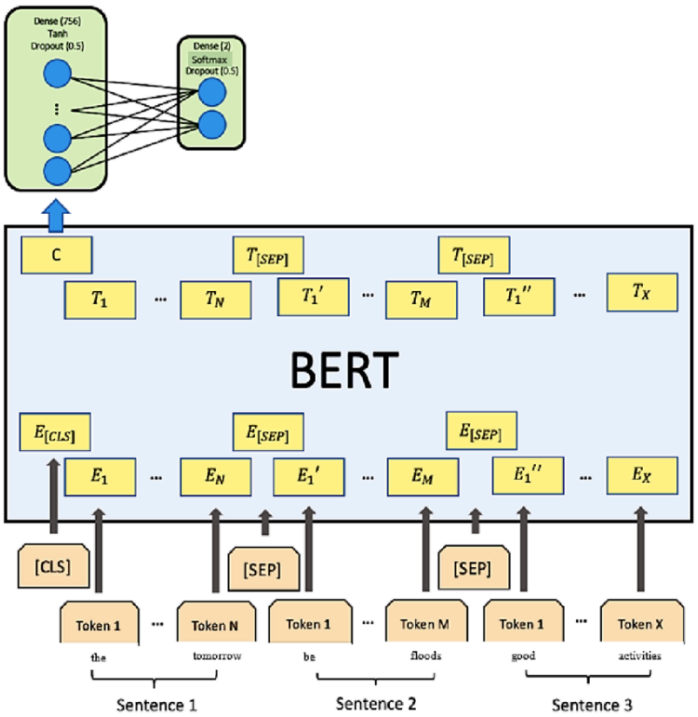

Рис 3.6 Представлення роботи моделі на основі BERT

Кожен енкодерний шар BERT містить механізм self-attention, який дозволяє моделі враховувати контекст усієї послідовності слів при обробці кожного окремого токена. Для цього для кожного слова обчислюються три типи векторів: запити (query), ключі (key) та значення (value). На основі порівняння запитів із ключами модель визначає ваги уваги, що показують, наскільки важливими є інші слова в реченні для поточного токена. Ці ваги надалі застосовуються до векторів value, завдяки чому формується контекстуалізоване представлення кожного слова з урахуванням усієї послідовності.

Такий підхід дозволяє BERT не обмежуватися локальним оточенням слова, а інтегрувати інформацію з будь-якої позиції в тексті, формуючи узгоджене

семантичне подання на рівні всього речення або документа. Після проходження через усі шари енкодера спеціальний токен [CLS][30], який вставляється на початок послідовності, набуває векторного представлення розмірністю 768 елементів. Цей вектор розглядається як узагальнена характеристика змісту всього тексту і використовується як база для подальшої класифікації.

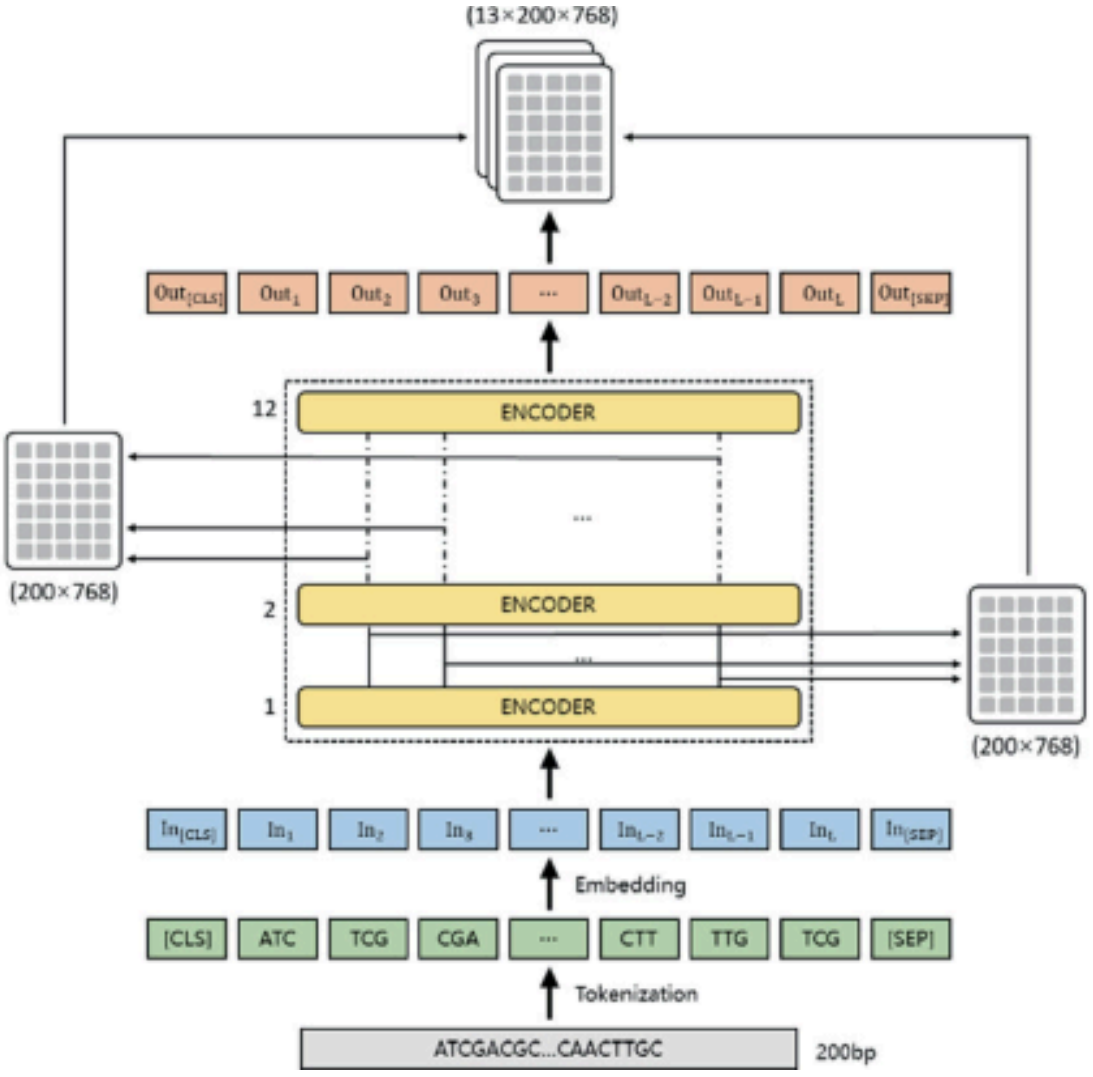

Рис. 3.7 Архітектура BERT

Як базову модель у даній роботі використано Multilingual BERT Base (mBERT), яка адаптована для роботи з багатьма мовами та містить 12 послідовних енкодерних шарів із загальною кількістю параметрів близько 110 мільйонів. Це

дозволяє застосовувати єдиний підхід до текстів різними мовами без потреби в окремому тренуванні моделей під кожну з них.

Поверх mBERT побудовано просту нейронну «голову» для класифікації новин. Безпосередньо до вектора [CLS] під'єднано перший додатковий повнозв'язний шар із 128 нейронів, який відповідає за виокремлення більш спеціалізованих ознак, корисних саме для розрізнення обраних типів інформаційних подій. Після нього розташовано фінальний шар із 5 вихідними нейронами, що відповідають п'яти цільовим класам: Crisis, Economical, Political, Corruption, Other. На виході цього шару застосовується функція активації softmax [31], яка перетворює набори вихідних значень на ймовірнісний розподіл по класах. Клас із найбільшою ймовірністю вважається прогнозованою категорією для відповідного текстового повідомлення.

Для формування класифікаційної моделі було підготовлено корпус приблизно з 2500 новинних повідомлень, що охоплюють різні тематичні категорії. Кожен текстовий елемент був уважно опрацьований та вручну розмічений, що забезпечило високу якість навчальних даних і мінімізувало ризик помилкових анотацій.

Після завершення розмітки набір даних було розділено на дві частини: 80% використано для навчання моделі, а 20% — для підсумкового тестування. Навчання здійснювалося протягом п'яти епох, тобто вся тренувальна вибірка п'ять разів була пропущена через модель для поступового покращення її параметрів. Як алгоритм оптимізації було застосовано Adam [32], який добре зарекомендував себе в задачах глибинного навчання завдяки здатності адаптивно підбирати швидкість навчання окремих параметрів, прискорюючи збіжність моделі.

Після завершення тренування якість класифікації було оцінено на тестовій вибірці, що не використовувалася під час навчання. За результатами експерименту модель досягла точності 96.8%, що свідчить про її здатність узагальнювати отримані знання та надійно віднесувати новинні повідомлення до відповідних категорій. Це підтверджує ефективність розробленого підходу та його придатність для використання в межах даної дисертаційної роботи.

3.3.2 Модуль семантичного виділення ключових фраз

Окремим компонентом системи є модуль семантичного виділення ключових слів і фраз, реалізований із використанням підходу KeyBERT[33]. Основна ідея цього методу полягає в тому, що спочатку для всього тексту (окремого документа або сукупності повідомлень, які розглядаються як одна інформаційна подія) обчислюється узагальнене векторне подання, а потім формуються кандидати ключових фраз, для яких також будуються векторні представлення. Далі кожен кандидат порівнюється з вектором документа, і в результаті відбираються ті фрази, які є семантично найближчими до загального змісту тексту.

На першому етапі нормалізований текст (або агрегований текст кількох пов’язаних повідомлень) подається на вхід трансформерній моделі, сумісній із BERT. Для документа формується єдиний вектор — наприклад, шляхом використання спеціального токена [CLS] або усереднення ембеддингів окремих токенів. Отриманий вектор розглядається як компактне семантичне подання всього тексту, яке узагальнює його основний зміст.

Наступний крок полягає у формуванні набору кандидатів ключових фраз. Для цього з тексту виділяються n-грамні послідовності слів (як правило, біґрами та триґрами), які відповідають простим лінгвістичним та частотним критеріям. На цьому етапі можуть відсіюватися фрази, що складаються переважно зі стоп-слів або не несуть самостійного змістового навантаження. Для кожного такого кандидата за допомогою тієї ж самої моделі обчислюється векторне подання, що забезпечує взаємну порівнюваність документа та фраз у єдиному семантичному просторі.

```python
# Function to apply KeyBERT on the 'Keywords' column
def apply_keybert(df):
    if 'Keywords' not in df.columns:
        print("Error: The dataframe must contain a column named 'Keywords'.")
        return None

    # Create new columns for unigrams and bigrams
    def extract_ngram(text, ngram_range):
        # Extract keywords with specified ngram range, handle the case where no keywords are found
        keywords = kw_model.extract_keywords(text, keyphrase_ngram_range=ngram_range, stop_words='english')
        return keywords[0][0] if keywords else ""  # Return the keyword or an empty string if none found

    # Apply to the 'Keywords' column
    df['Core (1-gram)'] = df['Keywords'].apply(lambda x: extract_ngram(x, (1, 1)) if len(x) > 0 else "")
    df['Core (2-gram)'] = df['Keywords'].apply(lambda x: extract_ngram(x, (2, 2)) if len(x) > 0 else "")

    return df

# Main function to upload the file and apply the transformations
def main():
    df = load_dataframe()

    if df is not None:
        # Apply KeyBERT to extract keywords
        df_with_keybert = apply_keybert(df)

        if df_with_keybert is not None:
            # Show the modified dataframe
            print(df_with_keybert.head())
            # Save the modified dataframe to a new CSV
            df_with_keybert.to_csv('keywords_with_keybert.csv', index=False)
            print("File saved as 'keywords_with_keybert.csv'.")
            files.download('keywords_with_keybert.csv')
```

Рис 3.8 Побудова семантичних n-грам із використанням KeyBERT

Після цього обчислюється міра схожості між вектором документа і векторами ключових фраз (як правило, використовується косинусна подібність). Кандидати ранжуються за значенням цієї подібності: чим ближче вектор фрази до вектору документа, тим більш репрезентативною вона вважається для даного тексту. Для уникнення надмірної повторюваності серед найкращих кандидатів може застосовуватися додатковий крок фільтрації або диверсифікації, коли фрази з майже ідентичним змістом не включаються одночасно до фінального списку.

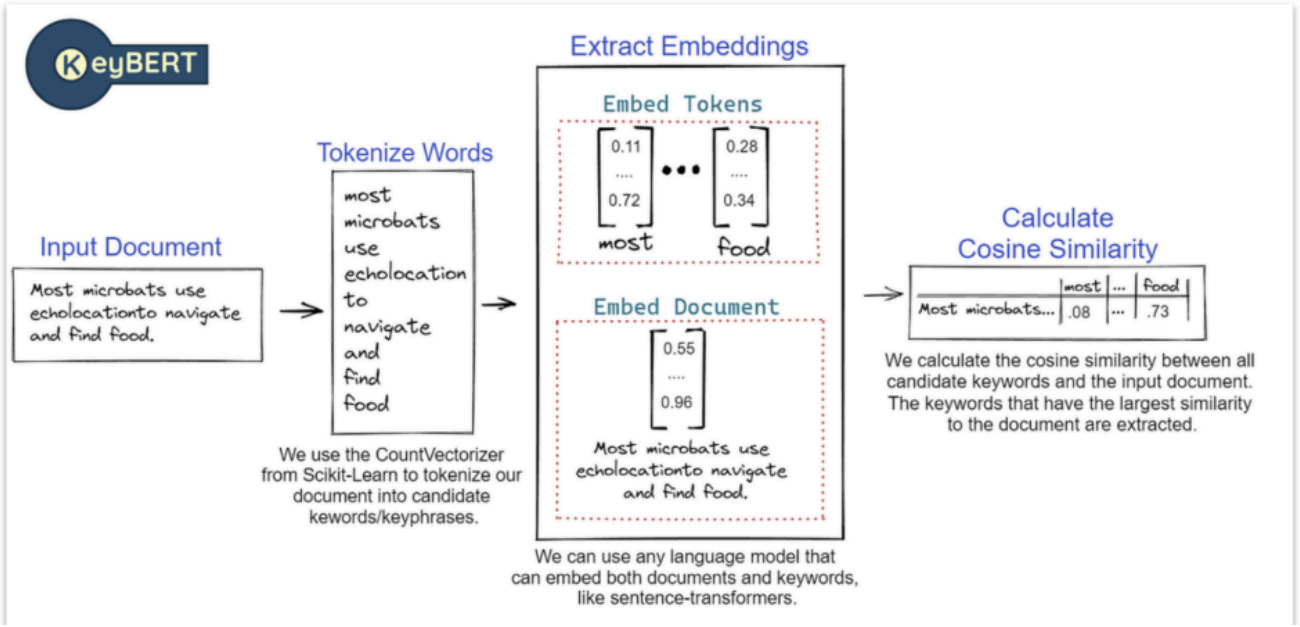

Рис 3.9 Схема семантичної послідновності слів у KeyBERT на основі BERT[34]

Результатом роботи модуля KeyBERT є впорядкований перелік ключових слів та фраз, які найточніше передають зміст аналізованого тексту чи групи текстів. У контексті розроблюваної системи ці ключові фрази використовуються для побудови хмар слів, формування коротких описів інформаційних подій, а також для підвищення інтерпретованості результатів семантичної кластеризації. Таким чином, модуль KeyBERT доповнює класифікаційну модель, забезпечуючи більш зрозуміле текстове представлення виявлених подій для кінцевого користувача.

3.4 Тестування системи

При першому відкритті застосунку користувачу буде відображено форму авторизації (рисунок 3.10), якщо користувач не має облікового запису, він може перейти на форму реєстрації (рисунок 3.11).

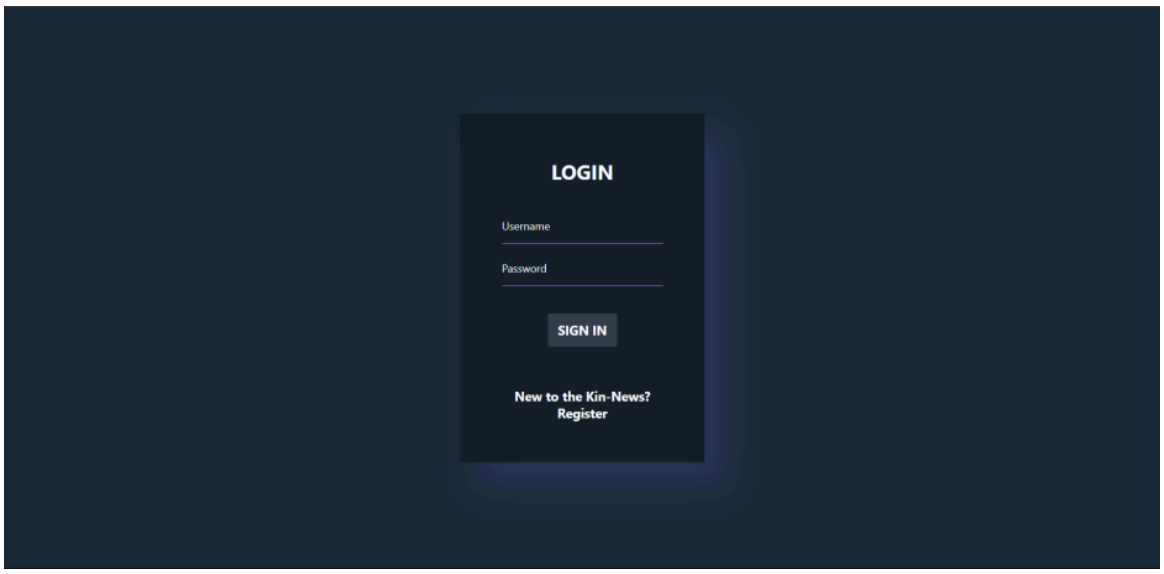

Рис. 3.10 Форма авторизації

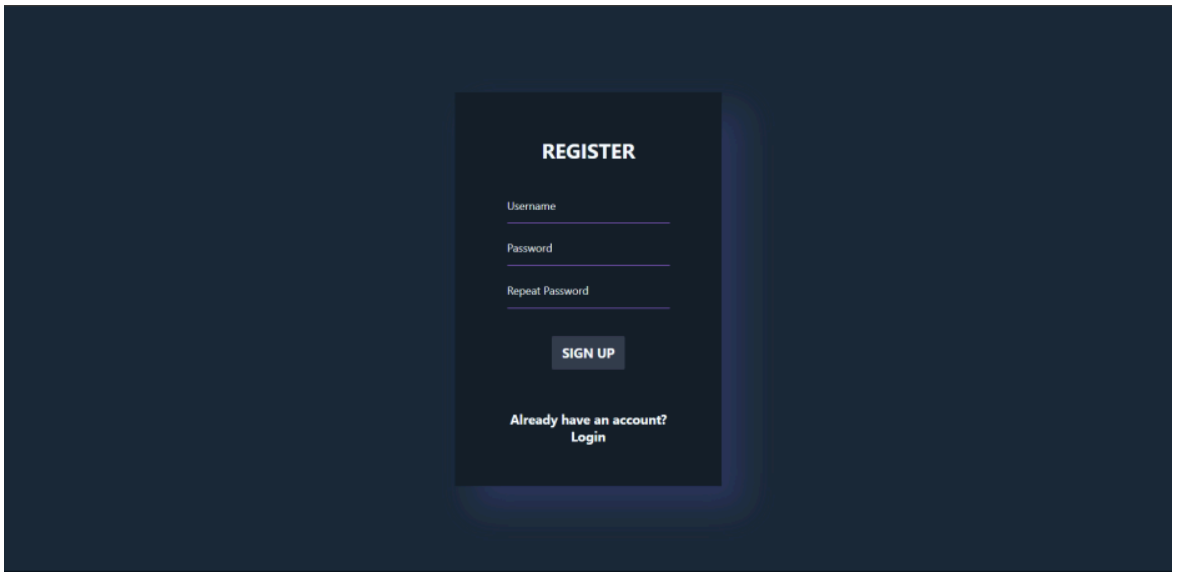

Рис 3.11 Форма реєстрації

Після авторизації користувач потрапляє на сторінку звітів, де можна побачити їх список та відповідні статуси(рисунок 3.12).

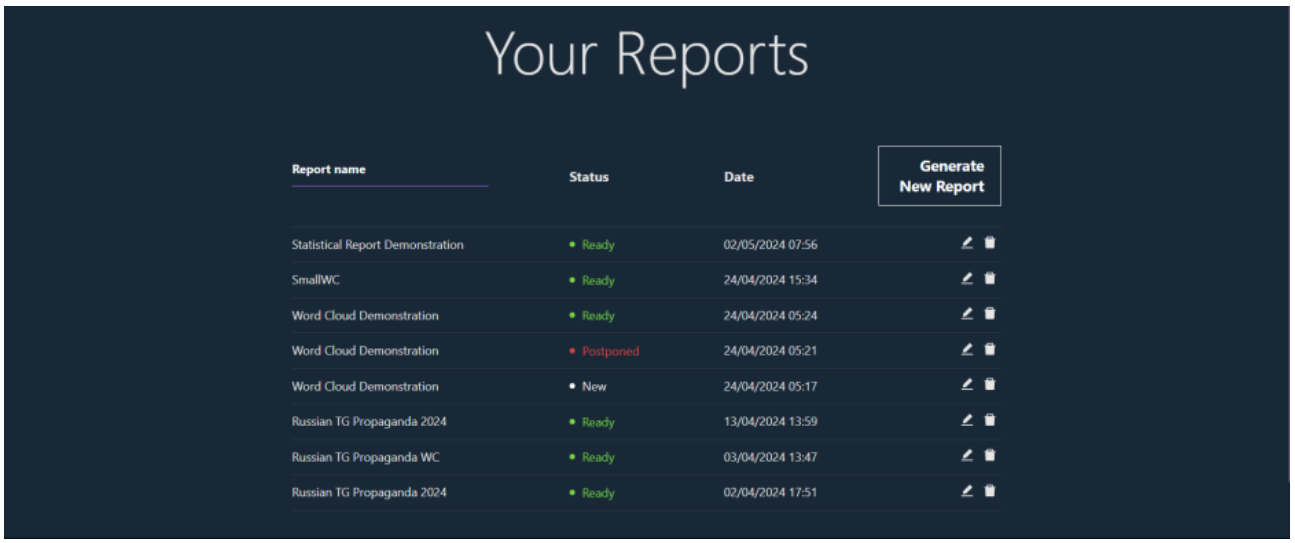

Рис. 3.12 Сторінка звітів

Використовуючи поле пошуку користувач може відфільтрувати звіти за відповідною назвою як показано на рисунку 3.13.

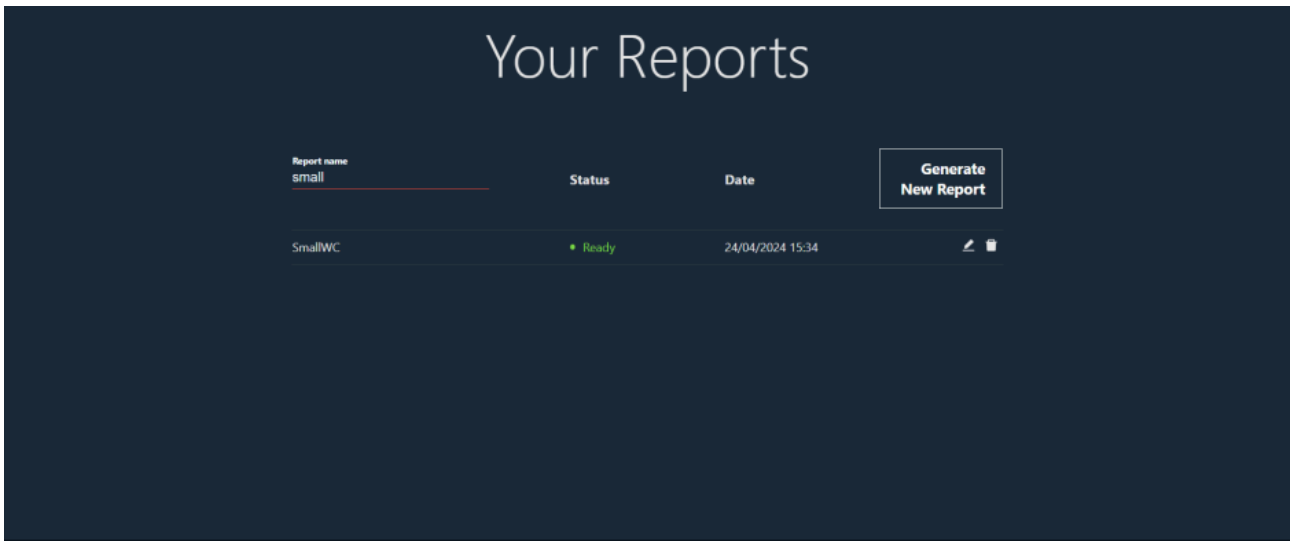

Рис. 3.13 Пошук звітів за назвою

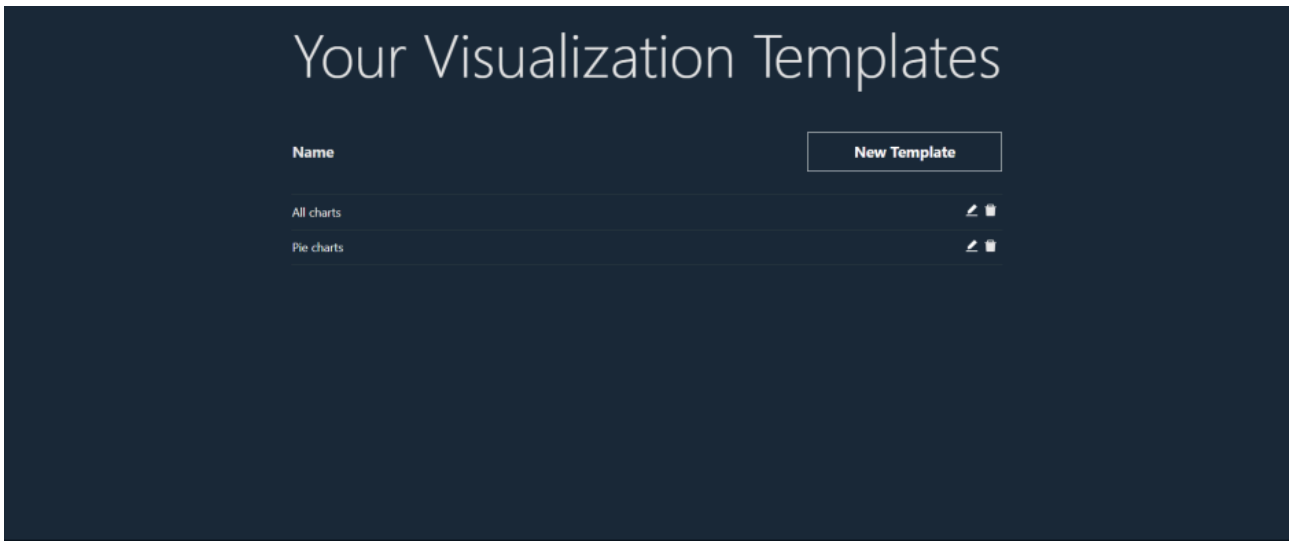

Рис. 3.14 Список орієнтовних шаблонів візуалізації

Користувач може створити шаблон натиснувши кнопку «New Template». Буде висвітлена відповідна форма, де користувач може ввести назву звіту та обрати необхідні візуалізації як показано на рисунку 3.15.

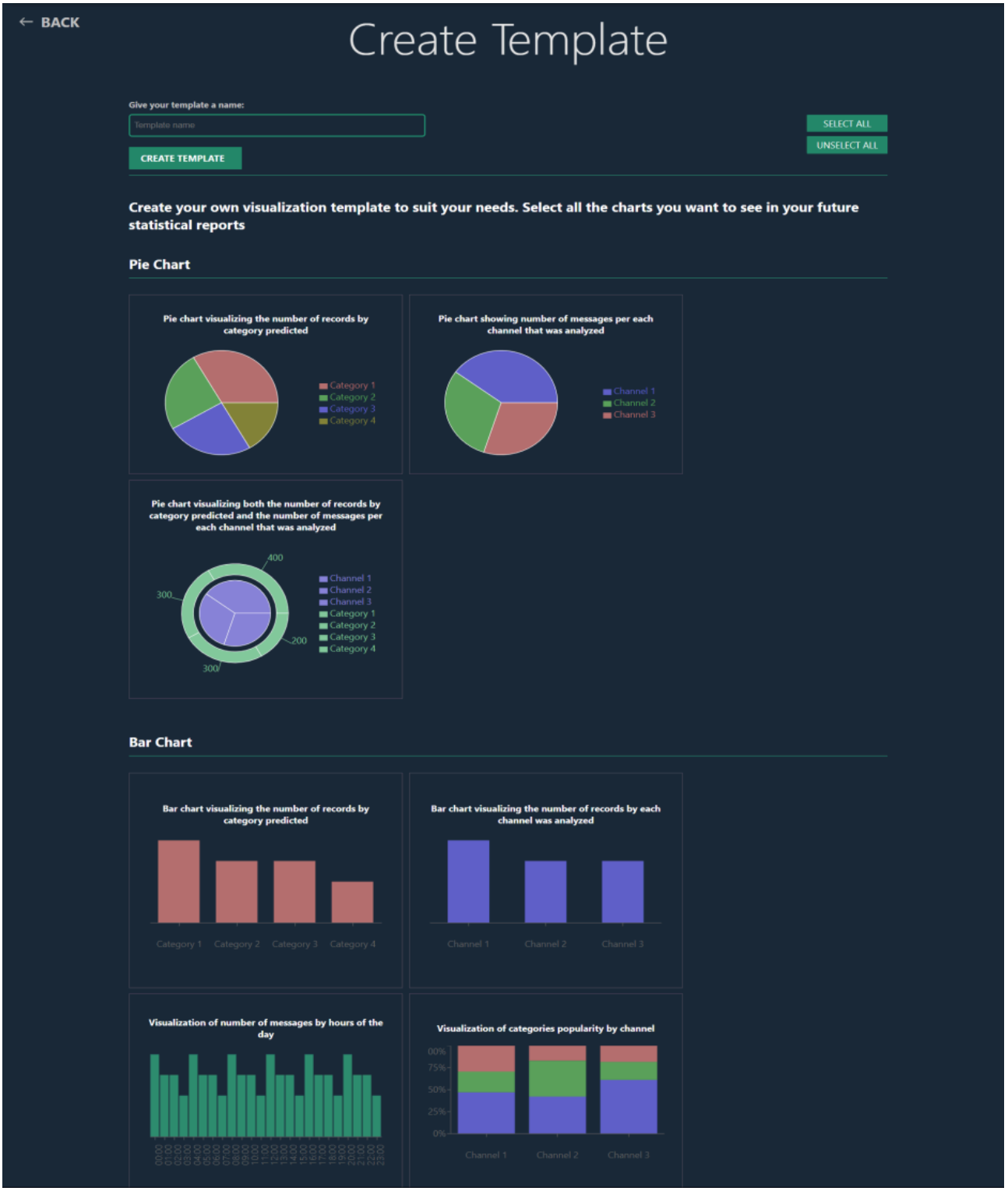

Рис. 3.15 Створення шаблону візуалізації

Для створення та управління моделями машинного навчання можна перейти на відповідну сторінку (рисунок 3.16).

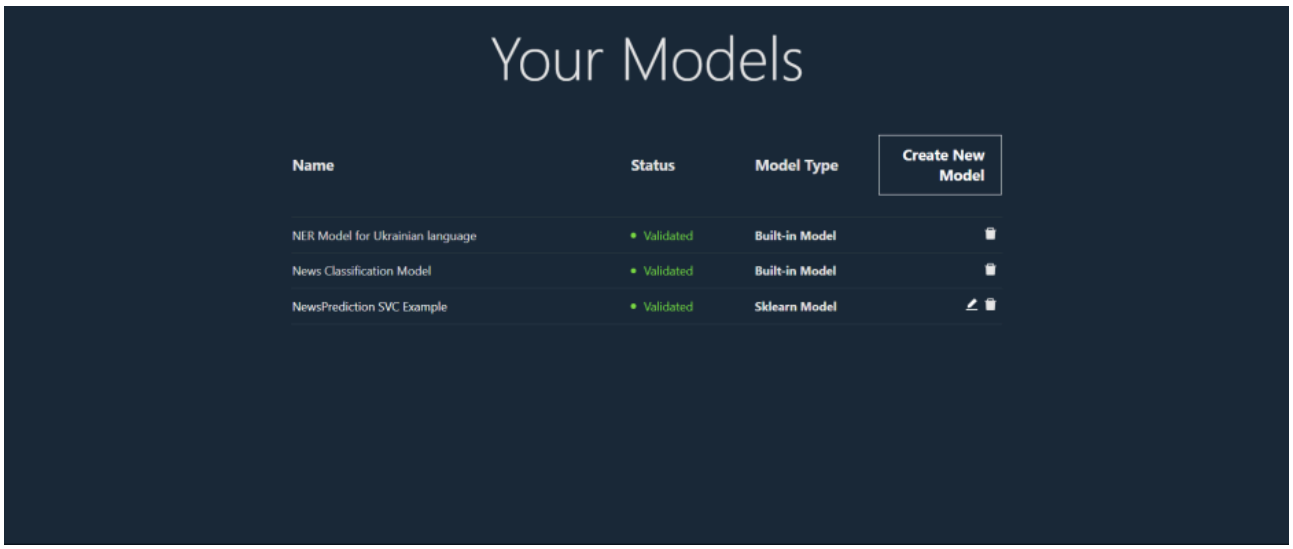

Рис. 3.16 Список моделей машинного навчання

Для завантаження нової моделі можна натиснути «Create New Model», форма створення моделі зображена на рисунку 3.17. Додаткові налаштування перед обробки тексту для класифікації зображені на рисунку 3.18.

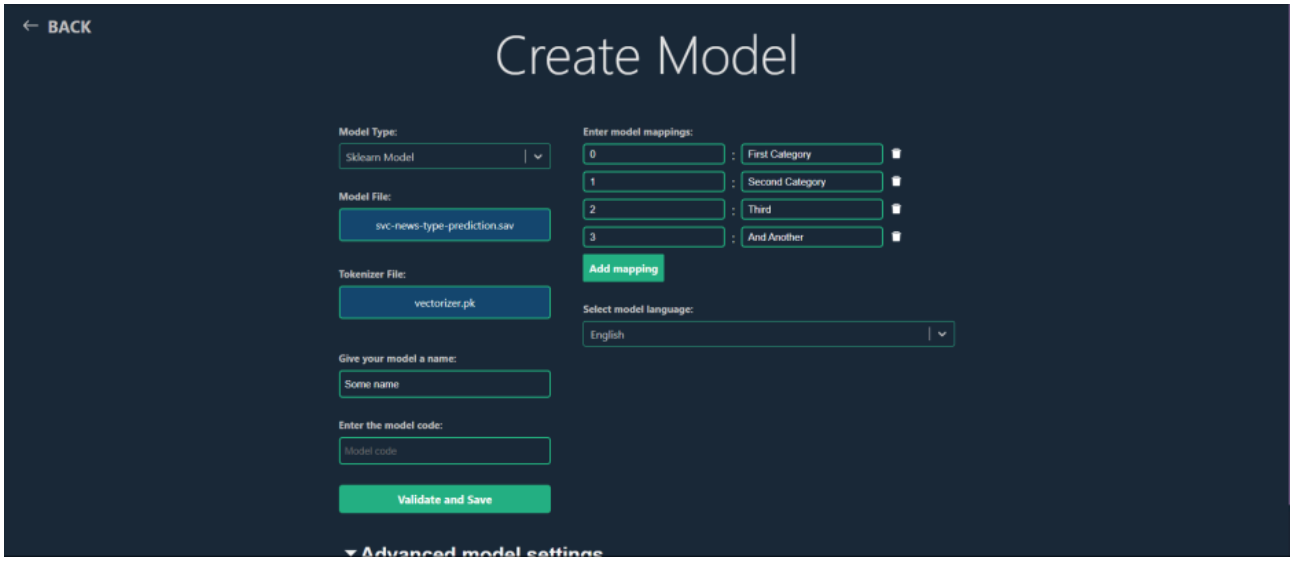

Рисунок 3.17 – Форма створення моделі

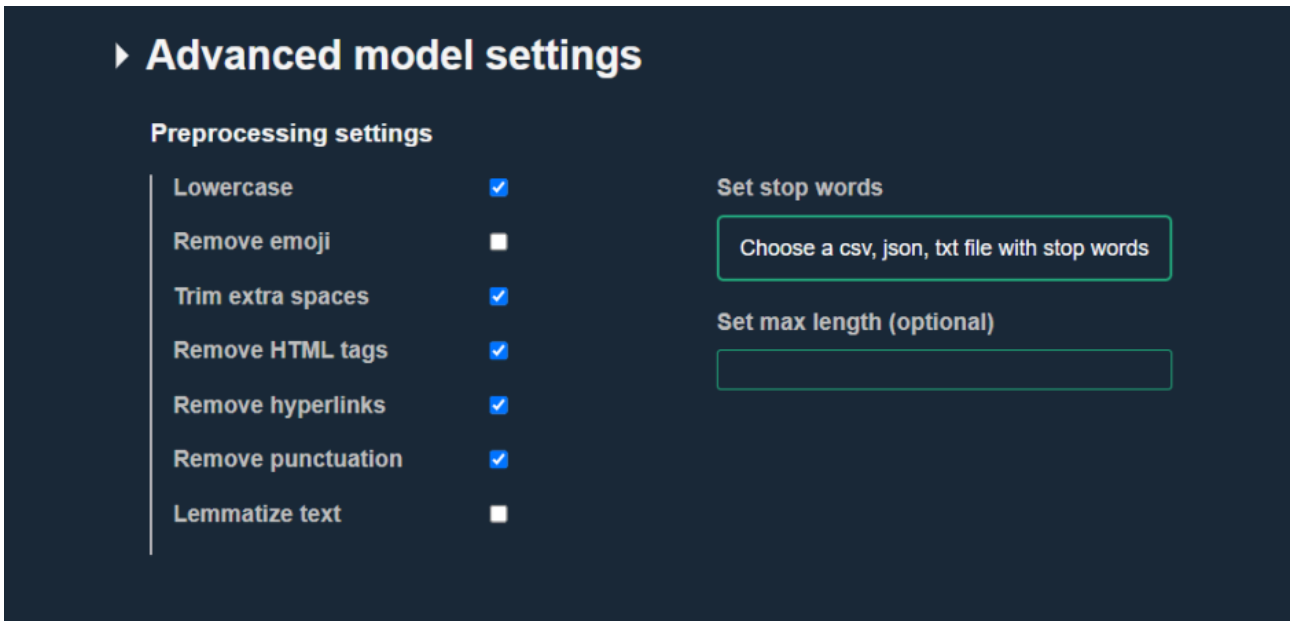

Рис. 3.18 – Налаштування моделі перед обробкою тексту

Маючи модель та шаблон візуалізації користувач може згенерувати звіт. Форма запиту на генерацію звіту зображена на рисунку 3.19.

Рис. 3.19 Форма запиту для генерацію звіту

Результат генерації статистичного звіту зображений на рисунку 3.20.

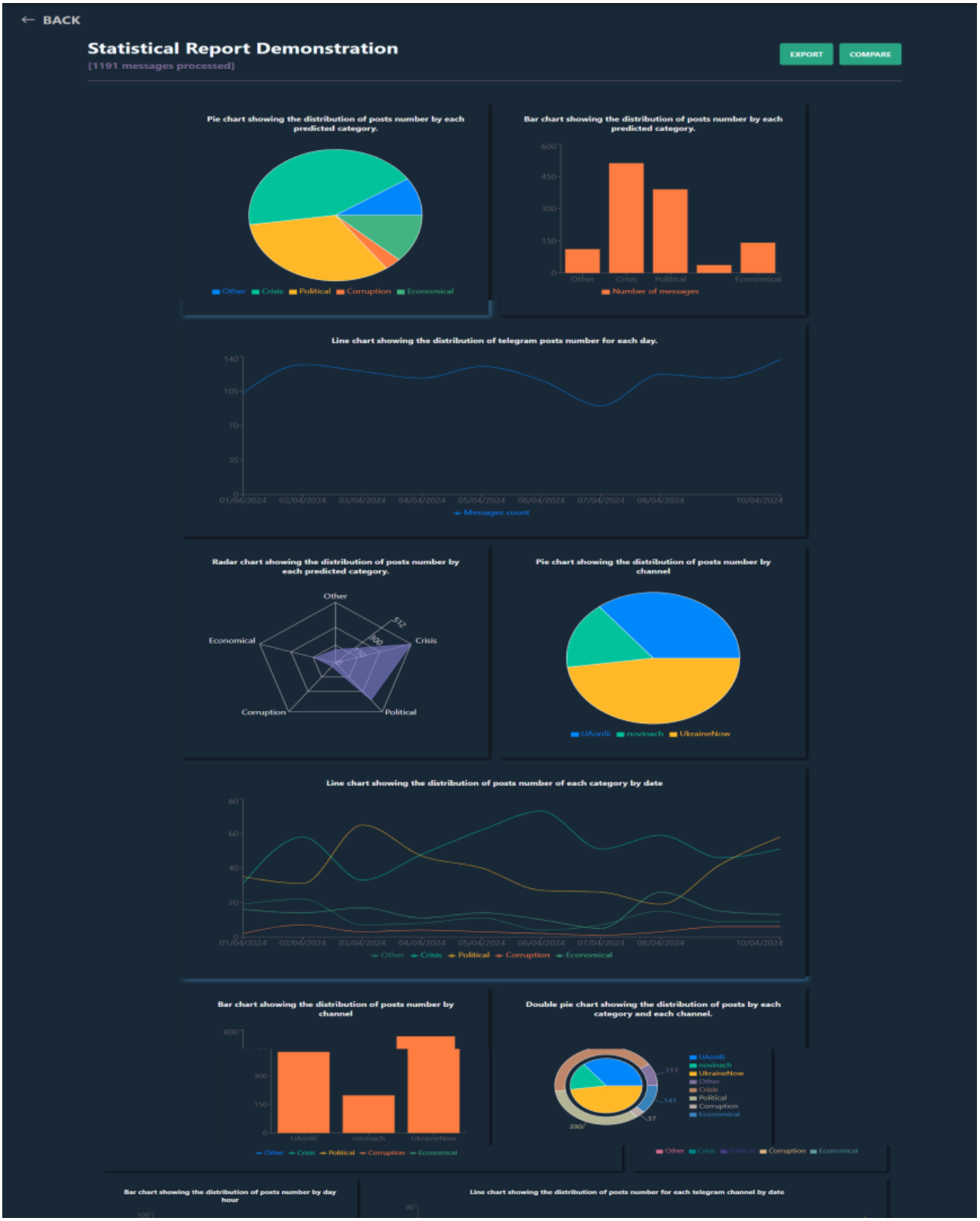

Рис. 3.20 Загальний статистичний звіт

Готовий звіт «хмара слів» зображений на рисунку 3.21.

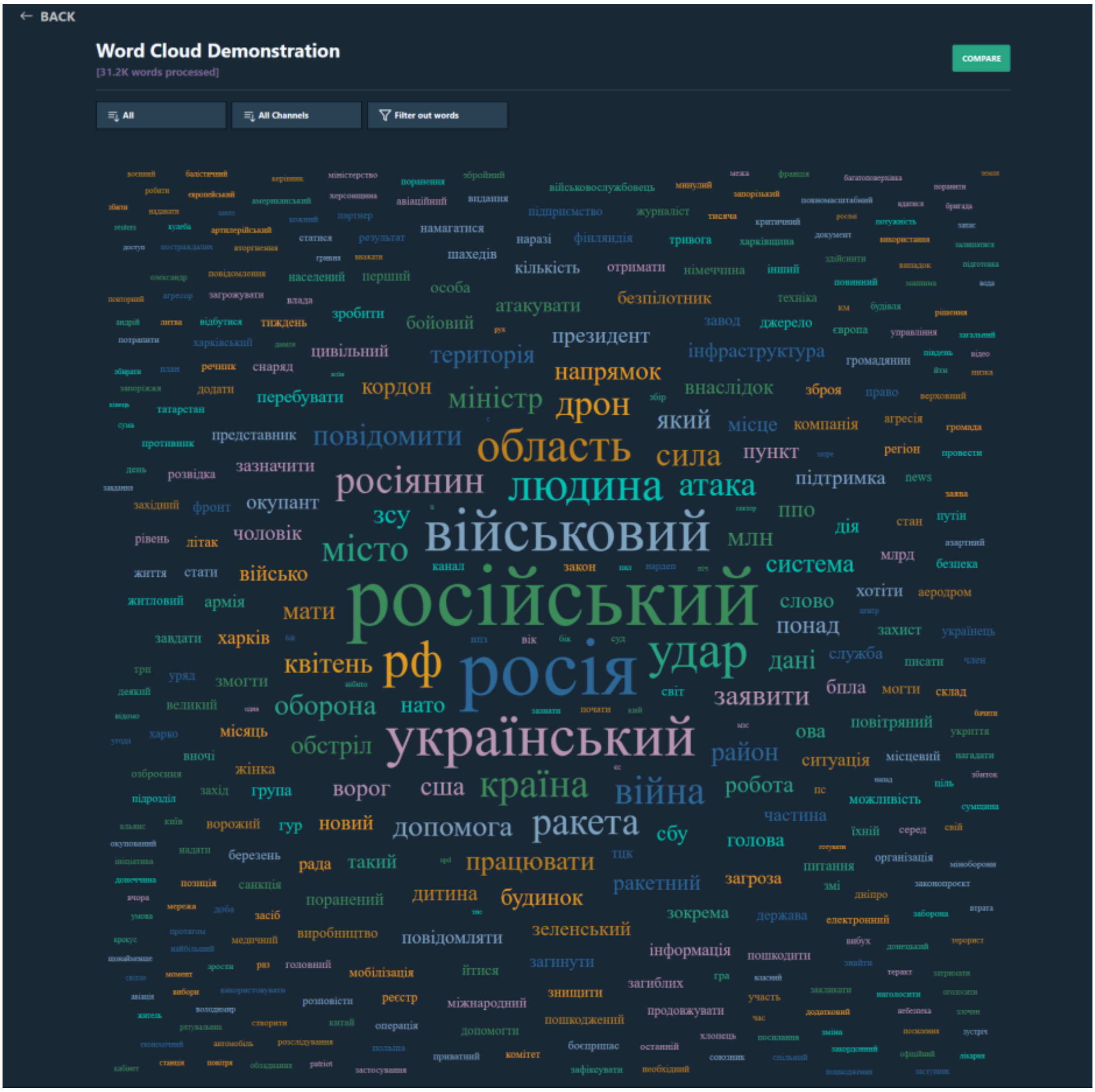

Рис. 3.21 Звіт типу «хмара слів»

Звіти можна порівнювати, для цього треба натиснути на кнопку «Compare» на сторінці будь-якого звіту та обрати зі списку інший звіт з яким необхідно провести порівняння. В результаті екран буде розділено на дві частини, на першій буде відображений один звіт, на другій інший, як показано на рисунку 3.22.

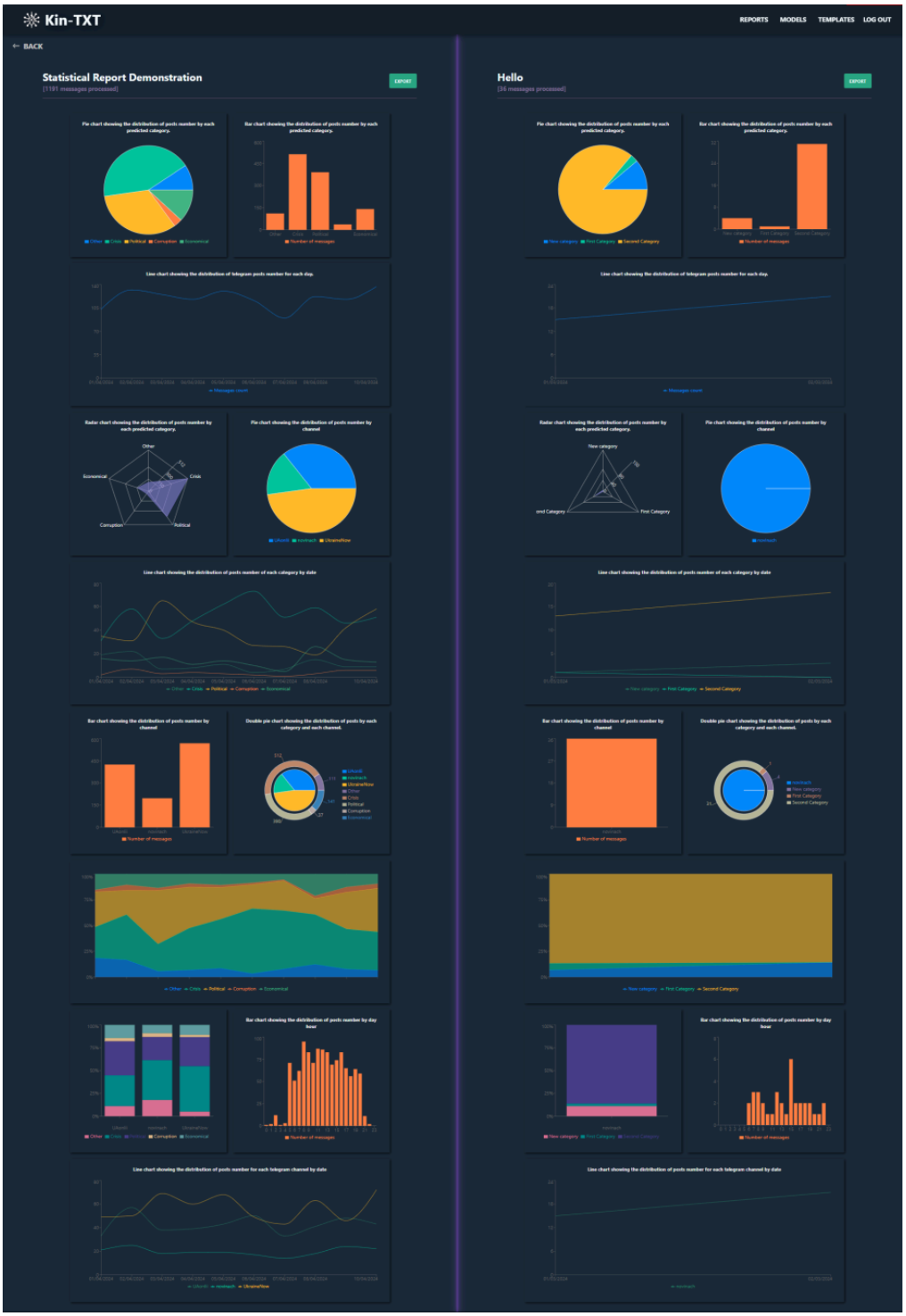

Рис. 3.22 Порівняння статистичних звітів

Хмари слів можна фільтрувати за категорією, приклад хмари слів для категорії «Crisis» зображений на рисунку 3.23.

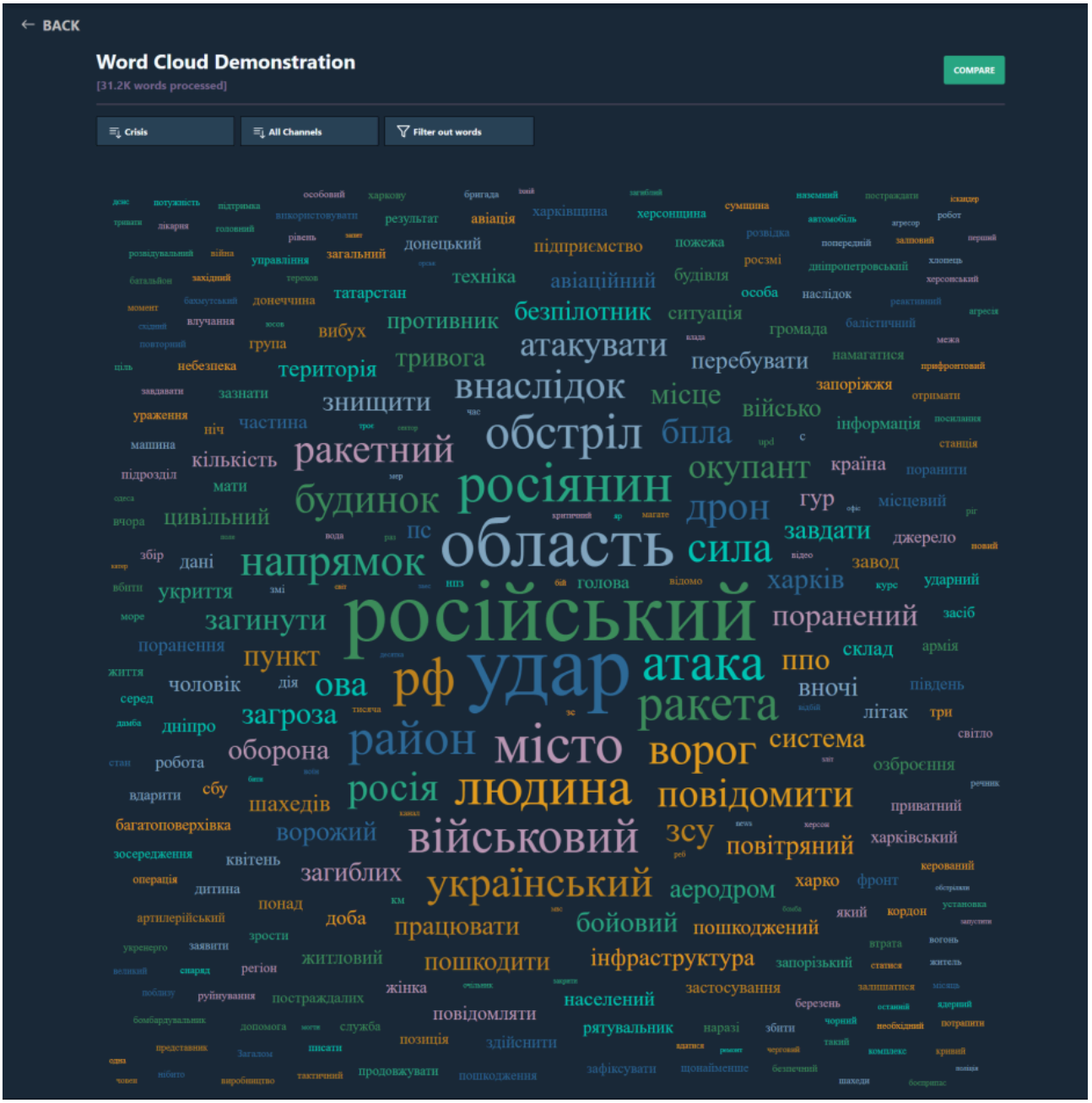

Рис 3.23 Приклад хмари слів відфільтрованої за категорією «Crisis»

Висновок до 3 розділу:

У третьому розділі було реалізовано ключові компоненти програмної системи, призначеної для виявлення, агрегування та семантичного опрацювання інформаційних подій. Структура системи сформована за модульним принципом:

окремо функціонують блоки збору та нормалізації текстових даних, класифікаційний модуль, семантичний аналізатор і підсистема зберігання результатів. Такий підхід забезпечує гнучкість архітектури, зручність підтримки та можливість подальшого масштабування.

Центральним елементом роботи стала розробка моделі тематичної класифікації на основі архітектури BERT. Після формування та ручної анотації навчального корпусу була побудована та навчена модель, адаптована для багатокласового поділу повідомлень. Оцінювання на тестових даних засвідчило її високу точність та здатність надійно структурувати інформаційний потік відповідно до визначених категорій.

У межах семантичного аналізу було реалізовано механізм виділення ключових слів на основі векторних подань тексту. Це дало змогу формувати цілісні лексичні описи подій, визначати схожість між повідомленнями та переходити від аналізу окремих елементів до роботи з їх смислово пов’язаними групами.

Крім того, у розділі описано створення інфраструктури для зберігання й доступу до опрацьованих даних, що забезпечує узгоджену роботу всіх компонентів системи та дає змогу підключати зовнішні сервіси через програмні інтерфейси.

У підсумку реалізовано повний набір функцій, необхідних для роботи інтелектуальної системи аналізу інформаційних потоків. Досягнуті результати підтверджують доцільність обраних методів і створюють підґрунтя для подальшого розвитку платформи та впровадження розширених алгоритмів штучного інтелекту.

РОЗДІЛ 4
РОЗРОБЛЕННЯ СТАРТАПУ-ПРОЕКТУ

4.1 Опис ідеї проекту

Проєкт «Система виявлення, агрегації та семантичного аналізу інформаційних подій із використанням штучного інтелекту» пропонується як відповідь на проблему перевантаження інформаційного простору та складності своєчасного виявлення важливих інформаційних подій у потоках повідомлень із соціальних мереж і медіаплатформ. Система орієнтована на аналітиків, дослідників, журналістів та організації, яким необхідно не просто читати окремі пости, а бачити події, тренди та інформаційні хвилі.

Відмінними якостями проєкту є:

- агрегування повідомлень із різних джерел в єдину уніфіковану модель даних;
- автоматична класифікація постів за типами інформаційних подій (криза, економіка, політика, корупція, інше);
- побудова семантичних подань текстів, виявлення кластерів повідомлень і формування на їх основі інформаційних подій;
- можливість перегляду аналітичних візуалізацій (хмари слів, часові графіки активності, розподіл подій за джерелами та категоріями).

Унікальність технології полягає в тому, що результатом роботи системи є не лише «сирий» масив постів, а структурований шар знань у вигляді інформаційних подій, для яких збережено повний контекст: вихідні повідомлення, семантичні ключові слова, часові характеристики та джерела. Це створює основу для подальшого кількісного та якісного аналізу інформаційних процесів.

Опис ідеї стартап-проєкту наведено в таблиці 4.1.

Таблиця 4.1

Опис ідеї стартап-проєкту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Веб-сервіс для виявлення, агрегації та семантичного аналізу інформаційних подій із потоків соціальних мереж та онлайн-медіа	Моніторинг інформаційного простору у режимі, наближеному до реального часу	Аналітики, дослідники та журналісти отримують узагальнену картину подій замість розрізнених постів у стрічках соцмереж.
	Підтримка OSINT-досліджень та аналітичних розслідувань	Зменшується обсяг ручної роботи: система автоматично виявляє кластери повідомлень, формує події та ключові слова.
	Використання в навчальних та науково-дослідницьких цілях	Студенти й викладачі отримують практичний стенд для вивчення NLP, семантичного аналізу та візуалізації інформаційних трендів.
	Підтримка прийняття рішень у організаціях (державні структури, НГО, бізнес)	Користувачі можуть раніше виявляти кризові, економічні чи політичні інформаційні події та оперативно реагувати на ризики.

Аналіз потенційних техніко-економічних переваг ідеї порівняно із пропозиціями конкурентів наведено у таблиці 4.2.

Таблиця 4.2

Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	Мій проект	Dataminr	Event Registry	GDELT	W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
1	Вартість використання	Безкоштовний	Висока абонентська плата	Комерційний сервіс, платний доступ до API	Дані безкоштовні, але потрібні власні ресурси	+		
2	Прозорість алгоритмів та можливість модифікації	Відкрита архітектура, можливість змінювати моделі та конвеєри	Закриті алгоритми та моделі	Частково задокументовані API, ядро закрите	Відкриті схеми даних, але складні для освоєння		+	

Таблиця 4.2(продовження)

Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ п/п	Техніко-економічні характеристики ідеї	Мій проект	Dataminr	Event Registry	GDELT	W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
3	Рівень готовності до масового корпоративного використання	Дослідницький/стартап-рівень, прототип	Зрілий корпоративний продукт	Стабільний комерційний сервіс	Орієнтовано на дослідників		+	
4	Гнучкість розгортання та інтеграції	Можливість локального або хмарного розгортання,	REST API	Можливість повністю локальної обробки даних	Хмарний сервіс, залежність від постачальника			+

4.2 Технологічний аудит ідеї проекту

Розробка такої системи потребує аналізу наявних аналогів та проблем з якими можна зіткнутися, у випадку відсутності засобів.

Розглянемо технологічний аудит в таблиці 4.3.

Таблиця 4.3

Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Реалізація модулю збору даних за допомогою ELT підходу	Існуючі приклади реалізації підходу	Наявна	Засоби є загальнодоступними
2	Реалізація модуля токенізації текстового документу	Відкриті бібліотеки та модулі, приклади реалізації	Необхідно розробити	Засоби є загальнодоступними
3	Реалізація модулю категоризації вакансій	Відкриті бібліотеки та модулі, приклади реалізації	Необхідно розробити	Засоби є загальнодоступними
4	Реалізація модуля семантичного аналізу та виділення ключових слів	Відкриті бібліотеки та модулі, приклади реалізації	Необхідно розробити	Засоби є загальнодоступними
5	Реалізація модулю аналітики	Існуючі приклади реалізації підходу	Наявна	Засоби є загальнодоступними

Таблиця 4.3(продовження)

Технологічна здійсненність ідеї проекту

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
6	Програмний продукт, що включає в себе всі розроблені модулі	Програмні засоби для розробки модуля та описані вище підходи	Необхідно розробити	Засоби є загальнодоступними
Обрані технології реалізації ідеї проекту: Python,React, PostgreSQL				

Можна зробити висновок, що реалізація цієї ідеї можливо і включає в себе розробку нових модулів та використання відкритих бібліотек.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Визначення сприятливих обставин, які можна використати для отримання переваг ринкових можливостей є важливою частиною при запуску стартап-проекту. Це дозволяє спланувати напрями розвитку проекту, визначити потреби потенційних клієнтів та оцінити ризики.

У таблиці 4.4 відображено попередню характеристику потенційного ринку стартап-проекту

Таблиця 4.4

Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	5
2	Загальний обсяг продаж, грн/ум.од	-
3	Динаміка ринку (якісна оцінка)	Стабільна
4	Наявність обмежень для входу (вказати характер обмежень)	Відсутні

Таблиця 4.4(продовження)

Попередня характеристика потенційного ринку стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	25%

Можна зробити висновок, що ринок є привабливим для входження за попереднім оцінюванням.

У таблиці 4.5 наводяться визначені потенційні клієнти, а також їх характеристика.

Таблиця 4.5

Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Потреба в агрегації даних серед її великої кількості	Дослід ники або корист увачі для власної вигоди	Поведінка залежить глибини потірбного аналізу даних	Надійна робота системи. Простота використання Можливість своєчасно отримувати статистичні звіти

Таблиця 4.5(продовження)

Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
2	Потреба в поверхнево му аналізі цих даних	Дослід ники або корист увачі для власної вигоди	Поведінка залежить від інформації, яка необхідна, та її кількості	Надійність роботи системи Швидкодія Можливість своєчасно отримувати статистичні звіти

Для оцінювання ринкових ризиків і бар'єрів для впровадження інтелектуальної системи моніторингу інформаційного простору доцільно враховувати низку зовнішніх чинників. Зокрема, слід аналізувати:

– появу нових технологічних рішень, здатних витіснити або дублювати функціональність системи;

– регуляторні зміни у сфері доступу до даних, приватності та API-політики платформ (Telegram, X/Twitter, Reddit);

– економічні та політичні фактори, які можуть вплинути на доступність джерел, стійкість ринку або потребу в аналітичних інструментах;

– конкурентні продукти, що пропонують ширший функціонал, інтеграції або кращі умови використання.

Таблиця 4.6.

Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Зміни у політиці доступу до API платформ	Проблеми в просуванні системи	Запуск системи в Україні та за її межами, вдала та продумана рекламна кампанія
2	Поява нових потужних конкурентів із ширшими функціями	Зниження конкурентоспроможності та необхідність пришвидшеного масштабування	Проаналізувати наявний функціонал та шляхи взаємодії з ними
3	Залежність від зовнішніх сервісів і хмарних платформ	Ризик недоступності даних або підвищення операційних витрат	Підключення альтернативних джерел даних
4	Нестача об’ємів сховища для збереження даних	Клієнти можуть мати обмежені локальні технічні ресурси, недостатні для повноцінної роботи системи	Винесення модуля обчислення на сервери компаній-партнерів

Таблиця 4.7.

Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Високий потенціал розвитку ринку інфоаналітики	Зростаючий інтерес до інструментів моніторингу інформаційного поля, соцмереж та новин. Збільшення потреби у швидкому виявленні інформаційних подій.	Забезпечення стабільної роботи сервісу, відповідність очікуваній якості, розширення функціоналу відповідно до запитів користувачів.
2	Можливість масштабування на світовий ринок	Продукт з якісними моделями та прозорою аналітикою може бути затребуваний у країнах із високою інтенсивністю медіа-трафіку та потребою в OSINT/alert-системах.	Адаптація системи під різні мови, часові пояси й локальні медіа; стратегія виходу на ринки з найбільшим попитом.
3	Інтерес інвесторів до AI/OSINT-проєктів	IT-платформи у сфері аналітики даних і ШІ активно купуються або інвестуються на стадії швидкого зростання.	Активне вдосконалення продукту, збір зворотного зв’язку, створення дорожньої карти розвитку та підготовка інвестиційного пакету.
4	Орієнтованість сервісу на кінцевого користувача	Зростання попиту на інструменти, що полегшують роботу аналітиків, журналістів, OSINT-дослідників, державних структур і приватних компаній.	Посилення маркетингу, демонстрація реальних кейсів, робота з ком’юніті, підвищення впізнаваності та довіри до сервісу.

Таблиця 4.7(продовження)

Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
5	Додаткові моделі монетизації	Окрім стандартної підписки можливі надходження від API-доступу, корпоративних пакетів, white-label-рішень та модулів аналітики преміум-рівня.	Розробка альтернативних тарифів, створення корпоративних пропозицій, інтеграція партнерських сервісів та консалтингові послуги.

У таблиці 4.8 наведено загальні риси конкуренції на ринку.

Таблиця 4.8.

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: чиста	На ринку присутні різні рішення для моніторингу інформаційного поля: Dataminr, Event Registry, GDELT, Meltwater, OSINT-платформи. Конкуренція полягає у швидкості, точності та зручності аналізу.	Запуск системи на території України та подальша вихід на міжнародні ринки; позиціонування як доступної альтернативи корпоративним платформам.

Таблиця 4.8(продовження)

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
2. За рівнем конкурентної боротьби: глобальний	Аналіз інформаційних подій є актуальним у будь-якій країні. Конкуренти працюють на світових ринках, незалежно від мови чи джерел даних.	Впровадження сучасних AI-технологій, підтримка мультимовності, робота з різними джерелами (Telegram, Reddit, X/Twitter), розширення набору аналітичних інструментів.
3. За галузевою ознакою: внутрішньогалузева	Усі продукти належать до галузі інформаційної аналітики та AI-рішень. Конкуренція ведеться між програмними системами з подібною структурою (збір → аналіз → візуалізація).	Формування унікальності через гібридний підхід (класична класифікація + семантичний аналіз), інтеграцію соцмереж та візуалізацію подій у зручному форматі.
4. Конкуренція за видами товарів: за бажанням	Користувачі очікують швидкого доступу до подій, гнучких фільтрів, автоматичного аналізу ключових слів та високої точності моделі. Конкуренти змагаються у тому, хто швидше задовольнить ці потреби.	Проведення регулярних опитувань користувачів, розширення функціоналу (алерти, дашборди, API), постійне підвищення якості алгоритмів і UX.

У таблиці 4.9 наведено більш детальні риси конкуренції на ринку за М. Портером.

Таблиця 4.9.

Аналіз конкуренції в галузі за М. Портером[34]

№	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
1	Dataminr, Event Registry, GDELT, Meltwater, OSINT-інструменти (Bellingcat, Maltego, TgStat, Telepathy)	Стартапи й дослідницькі проєкти, що працюють у сфері OSINT, соціальних медіа-аналітики та систем раннього виявлення інформаційних подій	Платформи-джерела даних (Telegram, X/Twitter, Reddit), постачальники хмарних сервісів, API-провайдери	Аналітичні центри, медіа, OSINT-дослідники, держустанови, приватні компанії, журналісти	Часткові аналоги: сервіси моніторингу соцмереж, окремі інструменти аналітики, агрегатори новин без семантичного аналізу

Можливість виходу на світовий ринок у сегменті систем інформаційно-аналітичного моніторингу оцінюється як **висока**. Хоч на ринку вже присутня значна кількість конкурентів, жоден із них не пропонує повністю відкритого або універсального рішення, яке б поєднувало:

– збір даних із соцмереж у режимі близькому до реального часу;

– автоматичну класифікацію повідомлень за тематичними категоріями;

– глибинний семантичний аналіз і кластеризацію подій;

– формування хмар слів та інші інструменти аналітики;

– можливість локального або гібридного розгортання.

Постачальники (Telegram, X/Twitter, Reddit API, хмарні сервіси) диктують технічні обмеження, але не визначають функціонал ринку.

Клієнти, навпаки, формують вимоги — від швидкості обробки до гнучкості аналітичних інструментів, — що створює можливість для зростання продукту.
Поточні конкуренти не пропонують універсального та доступного рішення, яке б поєднувало семантику, класифікацію та аналітичні модулі в одному продукті. Це дає змогу розроблюваній системі зайняти свою нішу та масштабуватися як інноваційна платформа для аналізу інформаційних подій.

На основі вище наведених таблиць виведено фактори конкурентоспроможності, та наведено у таблиці 4.10.

Таблиця 4.10.

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (чинники, що роблять фактор значущим у порівнянні з конкурентами)
1	Інноваційність технологій	Використання класифікації на основі моделей BERT, семантичного аналізу та векторних подань забезпечує вищу точність і швидше виявлення подій порівняно з класичними системами моніторингу.
2	Цінова політика	Наявність безкоштовного або гнучкого тарифу робить систему доступною ширшому колу користувачів, зокрема малому бізнесу, журналістам, OSINT-аналітикам і дослідникам.

Таблиця 4.10(продовження)

Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (чинники, що роблять фактор значущим у порівнянні з конкурентами)
3	Можливість використання системи для створення власних продуктів	Відкритий API та модульна архітектура дозволяють інтегрувати систему у сторонні рішення, будувати на її основі власні аналітичні сервіси або автоматизовані системи моніторингу.
4	Адаптація системи до різних ринків та мов	Підтримка мультимовних моделей (mBERT), можливість додавання нових джерел даних та гнучке масштабування роблять систему універсальною для різних країн і галузей.

Порівняльний аналіз сильних та слабких сторін системи збору та агрегації даних для аналізу вакансій відображено у таблиці 4.11.

Таблиця 4.11.

Порівняльний аналіз сильних та слабких сторін модуля аналітичного опрацювання текстової інформації

№ п/п	Фактор конкурентоспроможності	Бали (1–20)	Рейтинг товарів-конкурентів у порівнянні з системою (–3 … +3)
1	Інноваційність рішення (семантичний аналіз, кластеризація, AI-моделі)	12	+3
2	Цінова політика (гнучкі тарифи, доступність для малого бізнесу та аналітиків)	10	+1
3	Можливість використання системи для створення власних продуктів (API, модульність)	15	+2
4	Адаптація до різних ринків та мов (мультимовні моделі, масштабованість)	10	+1

Враховуючи вище наведені таблиці зробимо висновок ринкового аналізу, та сформуємо його у вигляді SWOT- аналізу (таблиця 4.12).

Таблиця 4.12.

SWOT- аналіз стартап-проекту

Сильні сторони (Strengths)	Слабкі сторони (Weaknesses)
Високий рівень інноваційності (семантичний аналіз, кластеризація, AI-моделі). Гнучка та конкурентна цінова політика. Можливість використання системи як платформи для створення власних продуктів (API, модульність). Адаптація до різних ринків та багатомовність.	Відсутність співпраці з інноваційними центрами та науковими інкубаторами. Відносно висока складність системи для непідготовлених користувачів. Початкова залежність від доступності зовнішніх джерел даних та API-платформ.
Можливості (Opportunities)	Загрози (Threats)
Висока зацікавленість користувачів у швидкому пошуку та аналізі актуальних інформаційних подій. Вихід на світовий ринок завдяки мультимовності та масштабованості системи.Можливість продажу бізнесу на ранній стадії росту зацікавленим інвесторам.Орієнтованість сервісу на користувача, що дозволяє швидко масштабувати аудиторію. Додаткові джерела монетизації (API-доступ, аналітичні звіти, корпоративні рішення).	Поява нових сильних конкурентів з більшими інвестиційними можливостями. Погіршення фінансового стану організацій-клієнтів, що знижує попит. Відсутність достатньої зацікавленості у частини потенційних користувачів. Нестача обсягів сховищ або обчислювальних ресурсів для обробки великих масивів даних.

На основі SWOT-аналізу визначено альтернативи ринкового впровадження стартап-проекту (таблиця 4.13).

Таблиця 4.13

Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Розробка системи з основним функціональним забезпеченням	Висока	4 місяці
2	Пошук замовників системи	Середня	3 місяця
3	Покращення основних можливостей системи на основі відгуків користувачів	Висока	5 місяців

Обрано першу альтернативу, тому що отримання ресурсів є більш простим та ймовірним, а строки реалізації – більш стислими.

4.4 Розроблення ринкової стратегії проекту

Для визначення ринкової стратегії потрібно спочатку визначити стратегії охоплення ринку (таблиця 4.14).

Таблиця 4.14

Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит у межах цільового сегмента	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Приватні компанії, що працюють з інформацією (аналітичні центри, медіа, OSINT-команди, консалтинг)	Середня	Високий	Середня	Середня складність залучення
2	Державні установи та органи безпеки, що потребують моніторингу інформаційного простору	Середня	Вище середнього	Низька на момент запуску	Висока складність залучення
3	Журналісти, аналітики, дослідники, що працюють з інформаційними подіями	Висока	Вище середнього	Середня	Низька складність залучення

Таблиця 4.14.(продовження)

Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит у межах цільового сегмента	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
4	Бізнеси, що потребують моніторингу репутації та кризових ситуацій (банки, логістика, енергетика)	Середньо-висока	Високий	Середня	Середня

При виборі одного сегменту використаємо стратегію диференційованого маркетингу та визначимо її базову стратегію розвитку (таблиця 4.15).

Таблиця 4.15

Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Розробка системи з основним функціональним забезпеченням	Диференційований маркетинг	Універсальність системи. Можливість використання системи для створення власних продуктів	Стратегія диференціації

У таблиці 4.16 описано стратегію базової конкурентної поведінки.

Таблиця 4.16.

Визначення базової стратегії конкурентної поведінки

№ п/п	Чи є проект «першопроходьцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні	Змішаний підхід – буде шукати нових споживачів та забирати існуючих у конкурентів.	Компанія буде досліджувати найкращі практики конкурентів та впроваджувати найсучасніші рішення швидше за конкурентів	Стратегія заняття конкурентної ніші на глобальному ринку і стати лідером у даному сегменті.

Враховуючи описані вище стратегій визначимо стратегію позиціонування продукту (таблиця 4.17).

Таблиця 4.17

Визначення стратегії позиціонування

№ п/п	Вимоги цільової аудиторії до продукту	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Асоціативний імідж, який має сформувати позицію проєкту
1	Універсальність системи та охоплення широкого спектра інформаційних джерел (Telegram, X/Twitter, Reddit, новинні сайти)	Стратегія диференціації через якість обробки даних та глибину аналітики	Використання сучасних AI-методів для збору, класифікації та семантичної агрегації інформаційних подій	Масштабованість, надійність, універсальність, можливість працювати з великими потоками даних

Таблиця 4.17(продовження)

Визначення стратегії позиціонування

№ п/п	Вимоги цільової аудиторії до продукту	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Асоціативний імідж, який має сформувати позицію проєкту
2	Можливість інтеграції через API та використання системи в існуючих робочих процесах	Стратегія технологічної диференціації через відкриті інтерфейси	Наявність API для доступу до агрегованих подій, ключових слів, трендів, а також можливість розширення під конкретні задачі	Відкритість, гнучкість, інноваційність — система як інструмент побудови власних продуктів
3	Глибокий семантичний аналіз та можливість виявлення трендів, інформаційних хвиль, ключових тем	Стратегія створення унікальної цінності (semantic-first approach)	Використання embedding-моделей, кластеризації та семантичного узагальнення для формування подій	Глибина аналізу, інтелектуальність, експертність

Таблиця 4.17(продовження)

Визначення стратегії позиціонування

№ п/п	Вимоги цільової аудиторії до продукту	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Асоціативний імідж, який має сформувати позицію проєкту
4	Інтуїтивна аналітика: графіки, хмари слів, часові профілі, категоріальні розрізи	Стратегія фокусування на користувацькому досвіді	Візуалізація складних інформаційних потоків у доступній формі, аналітична панель для різних типів користувачів	Зручність, прозорість, інформативність

4.5 Розроблення маркетингової програми стартап-проекту

Для формування маркетингової концепції продукту, спершу треба визначити результати аналізу конкурентоспроможності продукту (таблиця 4.18).

Таблиця 4.18.

Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує система	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
1	Універсальність системи та охоплення різних джерел (Telegram, Reddit, X/Twitter)	Доступ до актуальних інформаційних подій, агрегованих у єдиному вікні; можливість переглядати аналітику, хмари слів та часові профілі	Використання сучасних методів збору, нормалізації та семантичної агрегації даних; гнучка масштабована архітектура
2	Можливість інтегрувати систему у власні продукти або аналітичні процеси	Можливість використовувати API для витягу подій, трендів, кластерів і ключових слів у зовнішніх платформах	Наявність відкритого API; гнучкі формати передачі даних; можливість розширювати систему відповідно до потреб користувача

Вартість на рекламу у товарі визначається типом реклами кількістю її відображення.. У таблиці 4.19 наведено ціни відповідно до кількості реклами.

Таблиця 4.19

Визначення меж встановлення ціни

№ п/п	Рівень цін на товари-замінники, тис. грн / програмний продукт	Рівень цін на товари-аналоги, тис. грн / програмний продукт	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар / послугу
1	Відсутні прямі замінники (є часткові: Telegram-боти, прості OSINT-інструменти)	15–150 тис. грн (Dataminr, Meltwater, EventRegistry, OSINT-платформи із підпискою)	Середній – високий (аналітичні центри, приватні компанії, держустанови)	Нижня межа: 0 грн для базового доступу (freemium)Верхня межа: ціна за 1 показ повідомлення / події або пакетна передплата

У таблиці 4.20 наведена оптимальна система збуту.

Таблиця 4.20

Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Придбання доступу до аналітичної платформи та модулів семантичного аналізу; оплата підписки або пакетів подій	Надання демо-доступу; технічні консультації; налаштування інтеграцій через API; підтримка користувачів	Прямий, без посередників	Веб-сайт платформи, прямі продажі B2B, рекомендації між організаціями
2	Замовлення кастомної аналітики або автоматичних звітів	Консалтинг, підготовка аналітичних звітів, технічна адаптація модулів під клієнта	Прямий	Контрактні угоди з приватними компаніями та аналітичними центрами
3	Оплата корпоративної ліцензії або розширених можливостей	Персоналізоване налаштування, розгортання, навчання персоналу	Прямий	Продаж через сайт + персональні менеджери для корпоративних клієнтів

У таблиці 4.21 наведено концепцію маркетингових комунікацій.

Таблиця 4.21

Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Приватні аналітичні компанії, медіа-організації, консалтингові центри, державні установи, що потребують оперативного виявлення інформаційних подій; організації, що працюють з OSINT;	Професійні соцмережі (LinkedIn), галузеві форуми, офіційний веб-сайт системи, тематичні блоги, email-розсилки, онлайн-конференції, Telegram-канали	Унікальність аналітичного ядра: автоматичне виявлення подій, семантичний аналіз, ключові слова та інформаційні хвилі в реальному часі	Показати клієнтам, що система дає більш глибоке та швидке розуміння інформаційної ситуації, ніж ручний моніторинг або існуючі інструменти; підкреслити ефективність, точність і можливість інтеграції	“Інтелектуальна платформа для моніторингу інформаційного простору: швидко, точно, автоматично”

Таблиця 4.21(продовження)

Концепція маркетингових комунікацій

№ п/п	Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
2	Комунікаційні та PR-відділи компаній, які відстежують медіа-події та репутаційні ризики	Соціальні мережі (Facebook, X/Twitter), корпоративні презентації, вебінари, інформаційні дайджести	Здатність системи виявляти тренди та репутаційні ризики ще до появи у традиційних ЗМІ	Пояснити користувачам, що система допомагає уникати кризових ситуацій і приймати рішення на основі достовірних даних	“Будьте на крок попереду інформаційних ризиків”
3	Дослідники OSINT та журналісти, які потребують швидкого доступу до перевірених	Telegram, Reddit, спеціалізовані OSINT-спільноти, конференції та семінари	Позиціонування як інструмента, що значно прискорює перевірку великої кількості	Показати, що система економить час, агрегує джерела та виділяє ключові події і факти автоматично	“Від сирих даних — до зрозумілих подій за лічені секунди”

Висновки до розділу 4

У четвертому розділі було проведено комплексний аналіз ринкових передумов, конкурентного середовища та технологічних можливостей стартап-проєкту, який передбачає створення системи виявлення, агрегації та семантичного аналізу інформаційних подій на основі методів штучного інтелекту. Оцінено перспективи впровадження продукту, можливі бар'єри входження на ринок, ключові фактори конкурентоспроможності та потенційні стратегічні напрями розвитку.

Порівняльний аналіз існуючих рішень (Dataminr, Event Registry, GDELT, Meltwater, низка OSINT-інструментів) показав, що, попри їхню функціональність, жодна з платформ не пропонує комплексного поєднання потокового моніторингу, класифікації повідомлень, семантичної агрегації та автоматичного виділення інформаційних подій. Саме ця комбінація формує унікальну цінність запропонованої системи та створює нову нішу на ринку. Основними конкурентними перевагами проєкту є універсальність архітектури, модульність, можливість інтеграції з зовнішніми сервісами через API та висока адаптивність до різних типів даних.

У межах розділу було визначено ключові можливості (зростання попиту на системи інформаційної аналітики, перспективи виходу на міжнародний ринок, додаткові джерела монетизації) та загрози (поява нових конкурентів, економічні коливання, недостатня обізнаність користувачів). Результати аналізу підтверджують технологічну здійсненність, економічну доцільність та рентабельність майбутнього продукту.

Таким чином, системі доцільно розпочати вихід на ринок із мінімально життєздатної версії (MVP), яка включатиме основні функції збору даних, класифікації та семантичного аналізу. За умови правильно вибудуваної маркетингової стратегії та поетапного розвитку функціоналу, проєкт має потенціал зайняти конкурентну позицію як на українському, так і на міжнародному ринках.

ВИСНОВКИ

У магістерській дисертації вирішено науково-практичну задачу створення інтелектуальної системи для виявлення, агрегації та семантичного аналізу інформаційних подій на основі багатоджерельних даних. Розроблена система поєднує методи лінгвістичної обробки текстів, машинного навчання та семантичного моделювання, що забезпечує автоматизоване виявлення тематичних сплесків, формування інформаційних трендів та побудову релевантних сповіщень. Таким чином, реалізовано комплексний інструмент для підтримки інформаційної аналітики, здатний адаптуватися під потреби користувача та різні доменні області.

У межах поставлених завдань проведено огляд сучасних методів і технологій, що застосовуються у задачах виявлення інформаційних подій, аналізу трендів та обробки потокових даних. Розглянуто можливості комерційних і дослідницьких систем (Dataminr, Event Registry, GDELT, Meltwater, OSINT-інструменти), визначено їхні обмеження та окреслено нішу, в якій необхідною є більш гнучка, відкрита та семантично орієнтована система. Це дало змогу сформувати вимоги до архітектури та функціональності майбутнього рішення.

Було розроблено архітектуру системи збору, уніфікації та нормалізації текстових даних із різних платформ, включаючи Telegram, Reddit та X/Twitter. Архітектура вибудувана на принципах мікросервісності, що забезпечує масштабованість, незалежність модулів і можливість їхнього подальшого розширення. Реалізовано конвеєр обробки, який включає етапи отримання даних, очищення, стандартизації, збереження та формування єдиної структури повідомлення.

Також рамках роботи було створено модулі AI-аналізу, що охоплюють семантичну класифікацію повідомлень, кластерний аналіз, побудову векторних представлень (ембеддингів) та виявлення аномалій. Для класифікації застосовано модифіковану модель mBERT із додатковими шарами нейронної мережі, що забезпечує розподіл повідомлень за п’ятьма тематичними категоріями.

Семантичний модуль побудовано з використанням сучасних моделей embeddings та алгоритмів KeyBERT-подібного виділення ключових фраз, що дало змогу формувати структуру подій, визначати їхнє смислове ядро та аналізувати зв’язки між текстами в межах часових інтервалів.

Створено API та інтерфейс для взаємодії з системою, що дозволяє користувачеві переглядати інформаційні події, аналізувати семантичні характеристики, відстежувати динаміку трендів та налаштовувати сповіщення. Реалізовано модулі візуалізації, які забезпечують побудову словникових хмар, часових графіків та структурованих звітів.

Функціональність системи протестовано на реальних інформаційних потоках — політичних подіях, кризових повідомленнях, інформаційних хвилях у соціальних мережах. Отримані результати підтверджують здатність системи виявляти тематичні сплески, групувати повідомлення за семантичними ознаками та будувати узагальнені події з високим ступенем репрезентативності. Це демонструє практичну цінність розробленого інструменту для аналітичних служб, медіамоніторингу, безпекових структур та дослідницьких організацій.

Узагальнюючи, у роботі досягнуто поставленої мети, виконано всі дослідницькі та прикладні завдання, а отримані результати підтверджують ефективність запропонованої системи та її перспективність у контексті подальшого розвитку інтелектуальних платформ аналізу інформаційного простору.

Список використаних джерел

1. Twitter. Twitter API Documentation. 2024. URL: https://developer.twitter.com/en/docs

2. Amazon Web Services. Building serverless streaming data pipelines with AWS Lambda and Amazon Kinesis. 2024. URL: https://aws.amazon.com/blogs/big-data/

3. Reimers N., Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. 2019. URL: https://arxiv.org/abs/1908.10084

4. Bellingcat. Online Investigations Toolkit. 2023. URL: https://www.bellingcat.com/resources

5. Machine Learning Research Group. Machine learning and natural language processing (NLP) fundamentals. 2024. URL: https://machinelearningmastery.com/

6. Snowflake Inc. ETL and ELT pipeline engineering best practices. 2024. URL: https://www.snowflake.com/guides/etl-vs-elt/

7. Dataminr. Real-time AI platform for event detection. 2024. URL: https://www.dataminr.com/

8. Event Registry. News event detection and analysis platform. 2024. URL: https://eventregistry.org/

9. GDELT Project. Global Database of Events, Language, and Tone (GDELT). 2024. URL: https://www.gdeltproject.org/

10. Meltwater. Media Intelligence and Analytics Platform. 2024. URL: https://www.meltwater.com/

11. Bellingcat. Overview of modern OSINT systems and investigative methodologies. 2023. URL: https://www.bellingcat.com/

12. Paterva. Maltego OSINT Tool. 2023. URL: https://www.maltego.com/

13. GNews API. Global News Search and Aggregation Service. 2024. URL: https://gnews.io/

14. Newsdata.io. News API Documentation. 2024. URL: https://newsdata.io/documentation

15. Hugging Face. SBERT and GTE multilingual embedding models. 2024. URL: https://huggingface.co/

16. Hyndman R. Time Series Forecasting with R: Trend and Spike Detection Methods. 2021. URL: https://otexts.com/fpp3/

17. Manning Publications. TF–IDF: Term Frequency–Inverse Document Frequency Explained. 2020. URL: https://www.manning.com

18. React. React Official Documentation. 2024. URL: https://react.dev/

19. Angular. Angular Developer Guide. 2024. URL: https://angular.dev/

20. Node.js. Node.js Documentation. 2024. URL: https://nodejs.org/

21. Python Software Foundation. Python 3 Documentation. 2024. URL: https://docs.python.org/

22. Microsoft. .NET Developer Platform Overview. 2024. URL: https://dotnet.microsoft.com/

23. Oracle. Java Platform, Standard Edition Documentation. 2024. URL: https://docs.oracle.com/javase/

24. Google. TensorFlow Machine Learning Framework Documentation. 2024. URL: https://www.tensorflow.org/

25. Django Software Foundation. Django Framework Documentation. 2024. URL: https://docs.djangoproject.com/

26. Pallets Projects. Flask Documentation. 2024. URL: https://flask.palletsprojects.com/

27. FastAPI. FastAPI Framework Documentation. 2024. URL: https://fastapi.tiangolo.com/

28. Microsoft. Data Transfer Object (DTO) Design Pattern Overview. 2023. URL: https://learn.microsoft.com/dotnet/architecture/

29. Google AI Language. BERT: Bidirectional Encoder Representations from Transformers. 2018. URL: https://arxiv.org/abs/1810.04805

30. Google AI. Understanding the CLS token in Transformer-based language models. 2018. URL: https://arxiv.org/abs/1810.04805

31. Stanford University. Softmax Function Explained. 2021. URL: https://cs231n.github.io/linear-classify/#softmax

32. Kingma D. P., Ba J. Adam: A Method for Stochastic Optimization. 2015. URL: https://arxiv.org/abs/1412.6980

33. Grootendorst M. KeyBERT: Minimal Keyword Extraction with BERT. 2020. URL: https://github.com/MaartenGr/KeyBERT

34. mlforseo.com. How to do keyword clustering with KeyBERT: machine learning implementation guide. 2024. URL: https://www.mlforseo.com/machine-learning-implementation-guides/keyword-research/how-to-do-keyword-clustering-with-keybert/

35. Porter M. E. The Five Competitive Forces That Shape Strategy (модель аналізу конкуренції в галузі). 2008. URL: https://hbr.org/2008/01/the-five-competitive-forces-that-shape-strategy

ДОДАТОК А

Репозиторій користувацького інтерфейсу проекту

Репозиторій дял запуску проєкту доступний за посиланням: https://github.com/PSheva/text_semantic_classification