Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)
Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розробка програмної системи аналізу просторово-часових даних морського трафіку з урахуванням факторів зовнішнього впливу
(тема)

Виконав:
здобувач четвертого року навчання,
групи ІТШІ-22-3
Лев ЧУДАЙКІН
(власне ім'я, прізвище)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Штучний інтелект
(повна назва освітньої програми)
Керівник ст. викл. Олександр СТЬОПІН
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ШІ _____ Лариса ЧАЛА
(підпис) (власне ім'я, прізвище)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук

Кафедра Штучного інтелекту

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Штучний інтелект
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Чудайкіну Леву Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка програмної системи аналізу просторово-часових даних морського трафіку з урахуванням факторів зовнішнього впливу

затверджена наказом університету від 5 червня 20 26 р. № 488Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 14 липня 20 26 р.
3. Вихідні дані до роботи потоки AIS-повідомлень про координати, швидкість, курс та ідентифікатори суден; метеорологічні дані Open-Meteo; новинно-подієві дані GDELT; вимоги до програмної системи; технології Python, FastAPI, PostgreSQL/PostGIS та scikit-learn

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та постановка задачі
2) Модель
3) Розробка
4) Тестування

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / термін виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.06.2026	виконано
2	Аналіз предметної галузі	10.06.2026	виконано
3	Огляд існуючих рішень	14.06.2026	виконано
4	Постановка задачі	15.06.2026	виконано
5	Розробка концептуальної архітектури та моделі	17.06.2026	виконано
6	Реалізація модулів збору та зберігання даних	21.06.2026	виконано
7	Розробка алгоритмів просторово-часового	24.06.2026	виконано
8	Реалізація моделей машинного навчання	28.06.2026	виконано
9	Розробка користувацького інтерфейсу та	30.06.2026	виконано
	десктопного застосунку		
10	Тестування програмної системи	02.07.2026	виконано
11	Оформлення пояснювальної записки	11.07.2026	виконано
12	Підготовка презентаційних матеріалів	12.07.2026	виконано
13	Захист перед ЕК	14.07.2026	

Дата видачі завдання 8 червня 20 26 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст, викл. Олександр СТЬОПІН
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 51 с., 10 рис., 6 табл., 1 дод., 23 джерела.

АВТОМАТИЧНА ІДЕНТИФІКАЦІЙНА СИСТЕМА, АНАЛІЗ ПРОСТОРОВО-ЧАСОВИХ ДАНИХ, БАЗА ДАНИХ, ГЕОІНФОРМАЦІЙНА СИСТЕМА, ІНТЕГРАЦІЯ ДАНИХ, МАШИННЕ НАВЧАННЯ, МЕТЕОРОЛОГІЧНІ УМОВИ, МОРСЬКИЙ ТРАФІК, ПРОГРАМНА СИСТЕМА, ФАКТОРИ ЗОВНІШНЬОГО ВПЛИВУ.

Об’єкт дослідження – процес аналізу просторово-часових даних морського трафіку та впливу зовнішнього середовища на рух суден.

Предмет дослідження – методи збору, зберігання, просторово-часового зв’язування й аналізу різнорідних даних морського трафіку із застосуванням машинного навчання.

Мета роботи – розробити програмну систему, що об’єднує дані AIS, метеорологічні умови та новинно-подієву інформацію, автоматично виявляє вплив зовнішніх факторів на швидкість і поведінку суден та відображає результати на інтерактивній карті, графіках і в аналітичних попередженнях.

Методи дослідження – аналіз предметної області, проєктування архітектури й бази даних, розробка алгоритмів просторово-часового зв’язування, експериментальне навчання моделей та функціональне і модульне тестування. Використано Python, FastAPI, PostgreSQL/PostGIS і scikit-learn.

Розроблено прототип, який збирає AIS-дані, погоду Open-Meteo та новини GDELT, зберігає їх у PostgreSQL/PostGIS і застосовує Isolation Forest та Random Forest Regressor для виявлення аномалій та оцінювання впливу погоди. Результати подано у Windows-застосунку. Систему перевірено на реальних потокових даних і рекомендовано для логістичних, портових та освітніх задач.

ABSTRACT

Bachelor’s thesis contains: 51 pp., 10 fig., 6 tabl., 1 ann., 23 references.

AUTOMATIC IDENTIFICATION SYSTEM, DATABASE, DATA INTEGRATION, EXTERNAL IMPACT FACTORS, GEOGRAPHIC INFORMATION SYSTEM, MACHINE LEARNING, MARITIME TRAFFIC, METEOROLOGICAL CONDITIONS, SOFTWARE SYSTEM, SPATIOTEMPORAL DATA ANALYSIS.

The object of research is the analysis of spatiotemporal maritime traffic data and the influence of the external environment on vessel movement.

The subject of research is methods for collecting, storing, spatiotemporally linking, and analyzing heterogeneous maritime traffic data using machine learning.

The objective is to develop a software system that integrates AIS data, weather conditions, and maritime news, detects the influence of external factors on vessel speed and behaviour, and presents the results through an interactive map, charts, and analytical alerts.

The methods include domain analysis, system and database design, development of spatiotemporal-linking algorithms, experimental model training, and functional and unit testing. The system uses Python, FastAPI, PostgreSQL/PostGIS, and scikit-learn.

A working prototype was developed to collect real-time AIS data from AISStream.io, weather data from Open-Meteo, and news from GDELT, store them in PostgreSQL/PostGIS, and apply Isolation Forest and Random Forest Regressor to detect atypical behaviour and estimate the impact of weather on vessel speed. Results are displayed in a Windows desktop application. The prototype was verified on real streaming data and can be used by logistics companies, port authorities, analytical centres, and educational institutions.

ЗМІСТ

Перелік умовних позначень, символів, одиниць, скорочень і термінів 8
Вступ 9
1 Аналіз предметної галузі та постановка задачі 11
1.1 Опис предметної галузі 11
1.1.1 Опис проблем 11
1.1.2 Огляд існуючих застосунків – систем або рішень 12
1.1.3 Інноваційність та переваги розроблюваної системи 16
1.1.4 Постановка задачі 17
2 Модель 19
2.1 Концептуальна архітектура системи 19
2.2 Модель даних та схема бази даних 21
2.3 Модель просторово-часового зв’язування 22
2.4 Моделі машинного навчання 24
2.4.1 Модель виявлення аномалій – Isolation Forest 25
2.4.2 Модель кількісної оцінки впливу погоди – регресія 26
2.4.3 Класифікація попереджень 28
3 Розробка 29
3.1 Вибір технологічного стеку 29
3.2 Реалізація модулів збору даних 30
3.3 Реалізація аналітичного та ML-конвеєра 32
3.4 Реалізація користувацького інтерфейсу 33
3.5 Реалізація десктопного застосунку 35
4 Тестування 38
4.1 Методика тестування 38
4.2 Модульне тестування 38
4.3 Функціональне тестування наскрізного конвеєра 40
4.4 Тестування десктопного застосунку 43
4.5 Аналіз результатів і обмежень системи 44

Висновки 47
Перелік джерел посилання 49
Додаток А Відомість кваліфікаційної роботи 51

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

СУБД – система управління базами даних;

ШІ – штучний інтелект;

AI – Artificial Intelligence – штучний інтелект;

AIS – Automatic Identification System – автоматична ідентифікаційна система;

API – Application Programming Interface – програмний інтерфейс застосунку;

ETA – Estimated Time of Arrival – розрахунковий час прибуття;

GIS – Geographic Information System – геоінформаційна система;

HTTP – HyperText Transfer Protocol – протокол передавання гіпертексту;

JSON – JavaScript Object Notation – текстовий формат обміну даними;

ML – Machine Learning – машинне навчання;

MMSI – Maritime Mobile Service Identity – ідентифікатор морської рухомої служби;

ORM – Object-Relational Mapping – об’єктно-реляційне відображення;

REST – Representational State Transfer – архітектурний стиль програмного інтерфейсу;

SQL – Structured Query Language – структурована мова запитів;

SRID – Spatial Reference System Identifier – ідентифікатор системи просторових координат;

UI – User Interface – користувацький інтерфейс.

ВСТУП

Сучасна морська логістика забезпечує понад 80% обсягу міжнародної торгівлі [5], а основним інструментом моніторингу трафіку є автоматична ідентифікаційна система (AIS) що передає координати, швидкість і курс суден у реальному часі. Однак самі по собі телеметричні дані не пояснюють причин затримок чи нетипової поведінки судна: на його рух впливають гідрометеорологічні умови та зовнішні події (страйки, блокування проток, санкції), яких AIS не фіксує. Наявні комерційні рішення, як показано в підрозділі 1.1.2, переважно обмежуються візуалізацією координат і не поєднують ці джерела в одному аналітичному середовищі, залишаючи пошук причин на розсуд оператора [3], [4].

Актуальність роботи зумовлена відсутністю на ринку доступних комплексних рішень, які б в одному середовищі поєднували телеметрію суден, метеорологічні умови та новинно-подієвий контекст. У межах даної роботи розроблено, реалізовано та перевірено на реальних потокових даних програмну систему, що агрегує ці гетерогенні потоки, виконує їхню просторово-часову прив’язку та застосовує моделі машинного навчання для виявлення впливу зовнішніх факторів на характеристики руху суден.

Метою роботи є розробка та практична реалізація програмної системи, яка надає можливість ефективно об’єднувати гетерогенні потоки даних у єдиному аналітичному середовищі, автоматично ідентифікувати причини зміни швидкості, маршруту чи нетипової поведінки суден, а також наочно відображати результати аналізу у вигляді інтерактивної карти, графіків та аналітичних попереджень у власному десктопному застосунку.

Практична цінність розробленої системи полягає в тому, що вона побудована виключно на безкоштовних відкритих джерелах даних (AISStream.io, Open-Meteo, GDELT) і не потребує платних підписок чи серверної інфраструктури для розгортання, що робить її доступною для

наукових досліджень, освітніх установ та невеликих логістичних операторів [6], [7], [8], [9].

Робота складається зі вступу, чотирьох розділів, висновків, переліку джерел посилання (23 найменувань) та одного додатка. У розділі 1 проведено аналіз предметної області, огляд наявних комерційних рішень і сформульовано постановку задачі. У розділі 2 описано модель розробленої системи: концептуальну архітектуру, схему бази даних, алгоритми просторово-часового зв’язування та моделі машинного навчання. У розділі 3 висвітлено практичну реалізацію системи – від вибору технологічного стеку до десктопного застосунку. У розділі 4 наведено методику й результати тестування на реальних даних, а також аналіз виявлених обмежень.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Опис предметної галузі

Морський транспорт забезпечує понад 80% обсягів світової торгівлі, а основним засобом моніторингу трафіку є автоматична ідентифікаційна система (AIS): судна, обладнані транспондерами, безперервно транслюють координати, швидкість, курс та ідентифікатор (MMSI – Maritime Mobile Service Identity). У результаті формуються масиви просторово-часових даних, проте самі по собі вони фіксують лише факт переміщення чи зупинки судна, не пояснюючи причин його навігаційної поведінки.

Причина в тому, що судноплавство відбувається в мінливому середовищі, де на маршрут і графік руху впливають зовнішні фактори двох категорій. Природні (гідрометеорологічні) – вітер, висота хвиль, шторми, льодова обстановка – змушують судно знижувати швидкість або відхилятися від курсу, що в даних AIS виглядає як аномалія (Speed Over Ground, Course Over Ground). Соціально-економічні (подієві) – страйки в портах, блокування проток (Суецький, Панамський канали), піратство, санкції чи геополітичні конфлікти – здатні миттєво паралізувати рух на цілих ділянках маршруту.

Проблема в тому, що AIS, метеорологічні API та новини існують в ізольованих інформаційних середовищах: диспетчери змушені вручну зіставляти координати судна з погодними радарами та новинними стрічками, щоб зрозуміти причину затримки. Це формує потребу в інструменті, який автоматично зв’яже ці джерела за єдиною системою координат і часовою шкалою.

1.1.1 Опис проблем

Реалізація системи для комплексного аналізу просторово-часових даних із урахуванням зовнішніх факторів пов’язана з низкою інженерних

викликів, зумовлених специфікою вхідних даних та складністю їхньої взаємодії.

По-перше, це гетерогенність джерел: телеметрія AIS надходить структурованими потоками з точними координатами й мітками часу; метеодані – точковими прогнозами, що потребують просторової прив'язки; новини – неструктурованим текстом, з якого потрібно виокремлювати географічні назви й типи подій.

По-друге, просторово-часова синхронізація: судно, погода і новини одночасно змінюються в просторі й часі, тож алгоритми мають безперервно зіставляти позицію судна з найближчими за часом і простором спостереженнями погоди та часовим вікном дії новини.

По-третє, обсяг і продуктивність: моніторинг навіть обмеженої акваторії генерує тисячі записів на годину, що вимагає буферизованого запису, просторового індексування бази даних та ефективних запитів найближчого сусіда.

По-четверте, якість вхідних даних: судна можуть тимчасово втрачати зв'язок або вимикати транспондери, метеопрогнози мають похибку, а новини – проблему релевантності, оскільки не всі події, що впливають на трафік, оперативно висвітлюються.

Нарешті, багатокритеріальна інтерпретація аномалій: падіння швидкості судна може бути спричинене погодою, чергою на вхід до каналу або комерційним рішенням судновласника, тож система має спиратися на надійні алгоритми ранжування цих факторів, а не на одне жорстко задане правило.

1.1.2 Огляд існуючих застосунків – систем або рішень

Ринок пропонує низку систем моніторингу морського судноплавства, більшість з яких базується на обробці даних AIS. Нижче розглянуто найпоширеніші рішення в контексті поставленої задачі.

MarineTraffic – одна з найбільших глобальних платформ відстеження суден у реальному часі через мережу наземних AIS-станцій і супутники. Дозволяє переглядати історію маршрутів, розраховувати ETA та накладати погодні шари на карту руху.

До переваг MarineTraffic можна віднести:

– глобальне покриття та величезна база даних суден (включаючи їхні статичні характеристики: розміри, тип, прапор);

– зручний користувацький інтерфейс та наявність API для розробників;

– можливість розрахунку щільності портів (Port Congestion) на основі агрегованих даних.

До недоліків системи можна віднести:

– обмеженість роботи з погодними даними: метеорологічні умови представлені переважно у вигляді візуальних шарів на карті, без глибокого автоматизованого зіставлення впливу конкретного шторму чи швидкості вітру на падіння швидкості конкретного судна;

– повна відсутність інтеграції з новинно-подієвою інформацією (геополітичні події, страйки, інциденти в портах не відображаються і не аналізуються системою);

– висока вартість підписки для доступу до історичних даних та розширеної аналітики, що ускладнює проведення незалежних наукових досліджень.

Spire Maritime збирає AIS через власні наносупутники й надає сервіс Weather Routing – оптимізацію маршрутів на основі ML-прогнозу погоди для економії палива.

До переваг рішень Spire Maritime належать:

– висока точність супутникових даних AIS, що дозволяє відстежувати судна у відкритому океані;

– глибока математична інтеграція телеметрії з гідрометеорологічними моделями;

– предиктивна аналітика, що дозволяє генерувати рекомендації для зміни курсу.

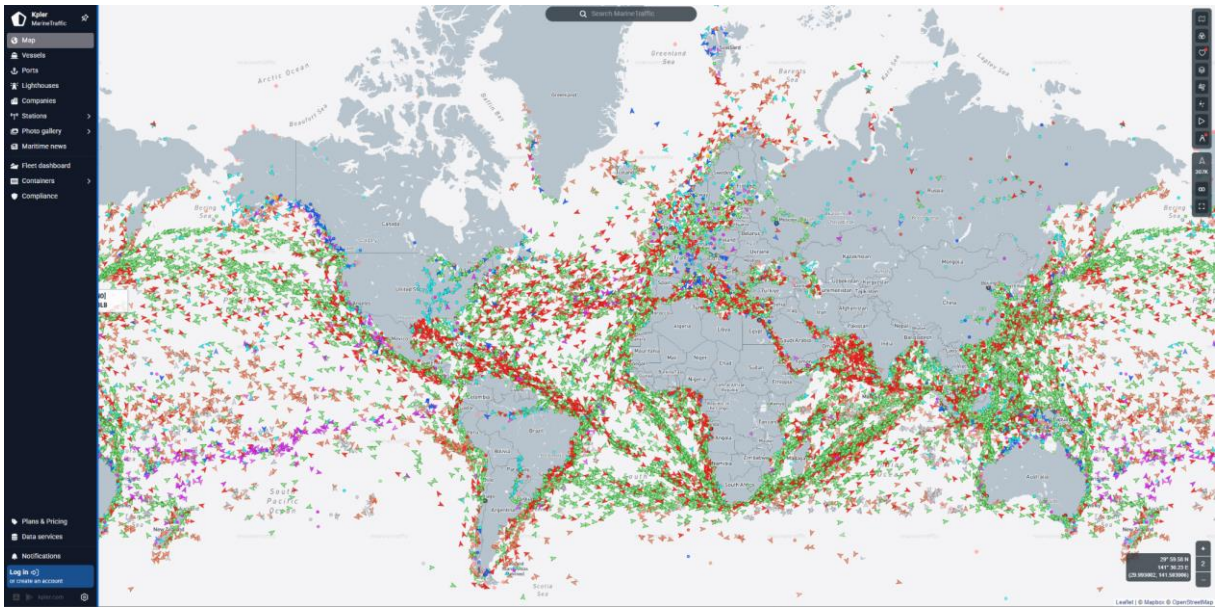

Рисунок 1.1 – Інтерфейс MarineTraffic

До недоліків такої системи можна віднести:

– це закрита B2B-екосистема, орієнтована виключно на великі судноплавні компанії, без відкритого інтерфейсу для гнучкого користувацького аналізу;

– система сфокусована суто на фізичних параметрах (маршрут, паливо, погода) і ігнорує зовнішній новинний контекст, що часто є реальною причиною зміни маршруту;

– відсутність інструментарію для ретроспективного виявлення аномалій, спричинених нефізичними факторами.

Платформи управління ланцюгами поставок (Project44, SeaRates) використовують AIS для відстеження контейнерних вантажів і прогнозування часу прибуття в порт.

Перевагами таких платформ є:

– зручність для логістичного бізнесу (облік номерів контейнерів, коносаментів);

– високоточні моделі прогнозування часу прибуття в порт на основі статистики затримок у конкретних терміналах.

Недоліками цих застосунків є:

– фокус на «вантажі», а не на «судні» та його просторово-часовому середовищі;

– відсутність інструментів для просторового аналізу траєкторій;

– неможливість користувачу самостійно налаштовувати кореляцію між зовнішніми подіями (новинами) та рухом флоту.

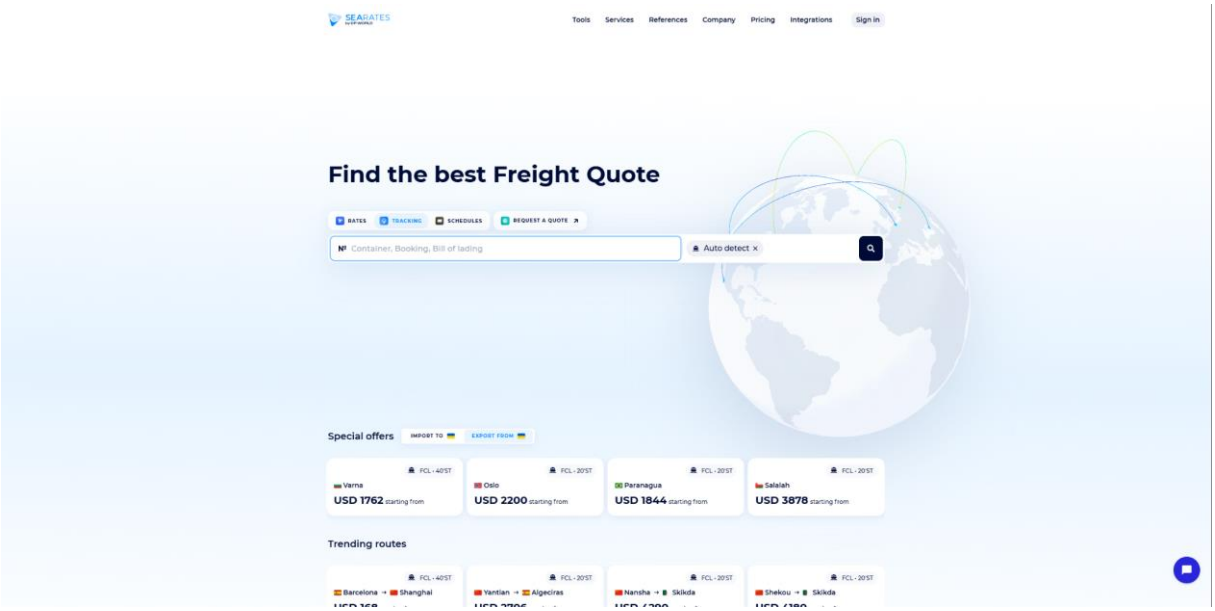

Рисунок 1.2 – Інтерфейс SeaRates

Огляд показує: потужні інструменти для трекінгу (MarineTraffic) чи погодної маршрутизації (Spire) існують, але жодне з розглянутих рішень не поєднує AIS, погоду й новинний контекст в одному аналітичному просторі – цей висновок безпосередньо обґрунтовує перевагу рішення, запропонованого в підрозділі 1.1.3.

1.1.3 Інноваційність та переваги розроблюваної системи

На відміну від рішень, розглянутих у підрозділі 1.1.2, жодне з яких не поєднує телеметрію, погоду та новини в одному аналітичному контурі, розроблена система відповідає на конкретну прогалину кожного з них.

– MarineTraffic накладає погодні дані на карту лише як візуальний шар, без автоматичного зіставлення конкретного шторму з падінням швидкості конкретного судна. Розроблена система обчислює для кожної позиції судна найближче за часом і простором спостереження погоди (підрозділ 2.3) і кількісно оцінює вплив цієї погоди на швидкість моделлю регресії (підрозділ 2.4) – результат виражається в конкретних вузлах, а не в накладеній картинці;

– жодна з розглянутих платформ не інтегрує новинно-подієвий контекст – страйки, блокування проток чи санкції залишаються поза полем зору їхніх алгоритмів. Розроблена система зіставляє позицію судна з геозіставленою новиною в межах часового вікна (підрозділ 2.3) і формує композитне попередження, коли аномалія збігається з подією;

– аналітика Spire Maritime закрита в межах B2B-контракту й не пояснює користувачу, чому конкретну поведінку визнано аномальною. Розроблена система використовує прозору евристику (z-оцінка відхилення ознак, підрозділ 2.4), яка дає змогу автоматично формувати людинозрозуміле пояснення прямо в інтерфейсі, без додаткової моделі генерації тексту;

– розгортання й використання всіх трьох комерційних платформ вимагає платної підписки; розроблена система побудована виключно на безкоштовних відкритих API (AISStream.io, Open-Meteo, GDELT), що робить її доступною для наукових досліджень і невеликих операторів без бюджету на комерційні рішення.

Таким чином, перевага розробленої системи не в окремій технології, а в тому, що вона закриває водночас кілька задокументованих у підрозділі 1.1.2 прогалин існуючих рішень, а не лише одну з них.

1.1.4 Постановка задачі

Морський транспорт домінує серед видів вантажних перевезень за обсягом (рисунок 1.3), що зумовлює масштаб задачі моніторингу.

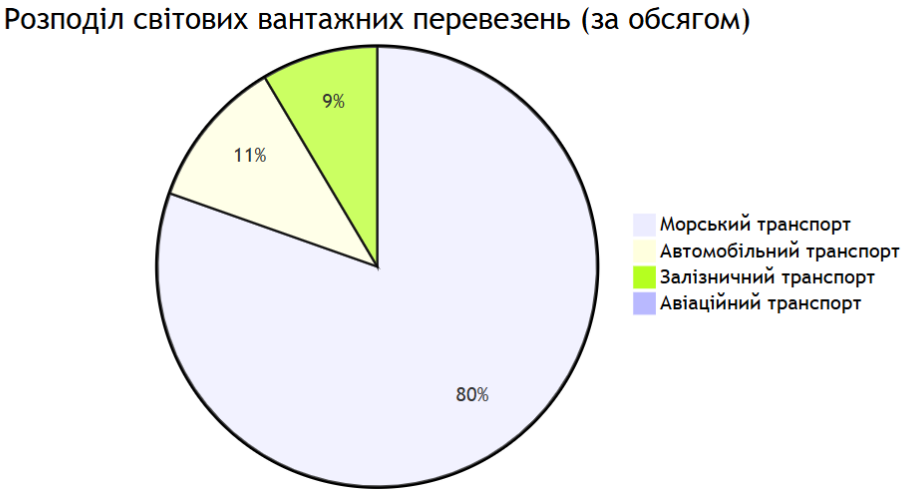

Рисунок 1.3 – Розподіл світових вантажних перевезень за обсягом

Проведений аналіз предметної галузі, переліку проблем і наявних рішень дозволяє сформулювати задачу кваліфікаційної роботи: розробити комплексну програмну систему для автоматизованого збору, просторово-часової прив’язки та аналізу даних морського трафіку з урахуванням зовнішніх факторів впливу (погоди та новинно-подієвої інформації).

Для досягнення поставленої мети в роботі виконано такі кроки:

– проведено аналіз доступних відкритих джерел та API для отримання телеметричних даних суден (AIS), актуальних метеорологічних зведень та новин, пов’язаних із морськими перевезеннями;

– розроблено архітектуру бази даних з використанням просторового розширення PostGIS, здатну ефективно зберігати та обробляти просторово-часові послідовності координат суден і спостережень погоди;

– реалізовано алгоритми просторово-часового зв’язування зібраних даних для співставлення координат судна з погодними умовами та зонами впливу новинних подій у конкретний момент часу;

– розроблено аналітичний модуль на основі моделей машинного навчання для виявлення кореляцій між зовнішніми факторами та змінами в характеристиках руху суден (затримки, відхилення від курсу);

– створено користувацький інтерфейс, що включає інтерактивну карту, графічні панелі та систему аналітичних повідомлень, а також окремий десктопний застосунок для Windows.

2 МОДЕЛЬ

2.1 Концептуальна архітектура системи

Розроблена програмна система реалізована як єдиний Python-застосунок, у межах якого одночасно виконуються фонові задачі збору даних (WebSocket-клієнт AIS, періодичні задачі опитування погодного та новинного API) та вебсервер FastAPI, що надає REST API й обслуговує інтерфейс користувача. Усі асинхронні задачі працюють у спільному циклі подій (event loop) бібліотеки asyncio, що дозволяє уникнути розгортання окремих черг повідомлень (Celery, Redis) або мікросервісної інфраструктури. Такий вибір є свідомим спрощенням, виправданим одноосібною розробкою кваліфікаційної роботи з обмеженим часовим горизонтом; винесення компонентів збору даних в окремі процеси або воркери є задокументованим напрямком подальшого розвитку системи (підрозділ 4.5) [12].

Архітектуру системи побудовано за принципом розділення відповідальності на рівні, кожен з яких відповідає за один етап обробки даних: рівень джерел даних (зовнішні API), рівень збору (ingestion), рівень зберігання (база даних PostgreSQL з розширенням PostGIS), рівень аналізу (просторово-часове зв’язування та машинне навчання) та рівень представлення (REST API, інтерактивна вебсторінка, десктопна оболонка). Концептуальну схему архітектури наведено на рисунку 2.1 [10], [11].

Рівень збору даних складається з трьох незалежних модулів. Модуль ais_client.py підтримує постійне WebSocket-з’єднання з сервісом AISStream.io, буферизує вхідні повідомлення про позиції суден та скидає їх у базу даних пакетами раз на кілька секунд, що суттєво знижує навантаження на СУБД порівняно зі вставкою кожного повідомлення окремо; при розриві з’єднання модуль автоматично перепідключається зі зростаючою затримкою (backoff). Модулі weather_client.py та

news_client.py, на відміну від ais_client.py, працюють не в режимі постійного потоку, а як періодичні задачі, що керуються планувальником APScheduler і виконуються з інтервалом, заданим у конфігурації системи.

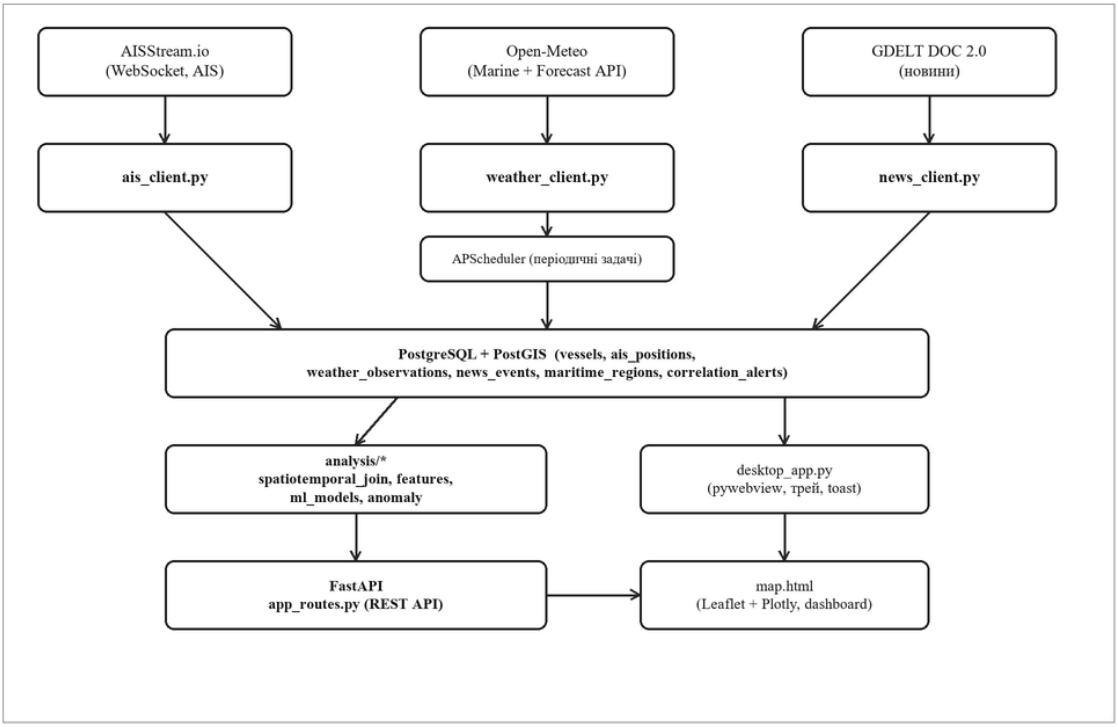

Рисунок 2.1 – Концептуальна архітектура системи

Рівень зберігання реалізовано на основі PostgreSQL з просторовим розширенням PostGIS, що надає нативну підтримку географічних типів даних і просторових операторів, необхідних для просторово-часового зв’язування (підрозділ 2.3). Рівень аналізу інкапсульовано в пакеті analysis та виконується як періодична фонова задача, що будує ознаки, перенавчає моделі машинного навчання та генерує аналітичні попередження. Рівень представлення надає два незалежні входи до системи: REST API на базі FastAPI, що використовується вебсторінкою з інтерактивною картою, та десктопна оболонка на базі pywebview, яка відображає той самий вебінтерфейс у нативному вікні Windows, додатково забезпечуючи системний трей і спливаючі сповіщення.

2.2 Модель даних та схема бази даних

Схему бази даних спроєктовано навколо шести основних сутностей, що відображають три джерела даних, які надходять у систему, та результат їх аналітичної обробки.

Просторові атрибути всіх таблиць збережено в типі PostGIS GEOGRAPHY з системою координат WGS 84 (SRID 4326) [10], [11], [24], що на відміну від типу GEOMETRY автоматично враховує кривизну земної поверхні під час обчислення відстаней (функція ST_Distance) та належності точки заданій області (ST_DWithin, ST_Covers) – це критично важливо для морських координат, що охоплюють значні географічні площі. Загальну структуру основних таблиць наведено в таблиці 2.1.

Таблиця 2.1 – Основні таблиці бази даних системи

Таблиця	Ключові поля	Призначення
vessels	mmsi (PK), imo_number, name, ship_type, dimensions, last_seen_at	Довідник суден: статичні характеристики та час останнього спостереження.
ais_positions	id, mmsi (FK), ts, geom (POINT), sog, cog, true_heading, nav_status	Телеметричні позиції AIS; складений індекс (mmsi, ts) для вибірки треку судна.
weather_ observations	id, ts, geom (POINT), wind_speed_ms, wave_height_m, swell_wave_height_m, visibility_m	Погодні спостереження, прив’язані до координат і часу конкретного судна.
news_events	id, source, external_id (унікальний), title, published_at, matched_region, geom	Новинні події з дедуплікацією за external_id та (за наявності) прив’язкою до регіону.
maritime_regions	id, name (унікальний), region_type, geom (POLYGON)	Довідник морських проток, каналів і портів для зіставлення новин із географією.
correlation_alerts	id, mmsi (FK), ts, geom, alert_type, severity, anomaly_score, residual_sog, description	Результат аналітичного модуля: попередження з поясненням природною мовою.

Для запобігання накопиченню дублікатів аналітичних попереджень на таблицю correlation_alerts накладено складений унікальний індекс за комбінацією (mmsi, ts, alert_type): якщо задача повторного скорингу повторно обробить те саме спостереження, вставка дубліката буде відхилена на рівні бази даних, а не логіки застосунку. Еволюцію схеми бази даних керовано через міграції Alembic, кожна з яких відповідає одному логічному етапу розробки (створення vessels/ais_positions, додавання weather/news/regions, додавання correlation_alerts, додавання унікального обмеження).

2.3 Модель просторово-часового зв’язування

Центральним науково-технічним елементом моделі системи є алгоритм просторово-часового зв’язування, що встановлює відповідність між позицією судна в конкретний момент часу та найбільш релевантними для неї погодними спостереженнями й новинними подіями. Загальну схему зв’язування наведено на рисунку 2.2.

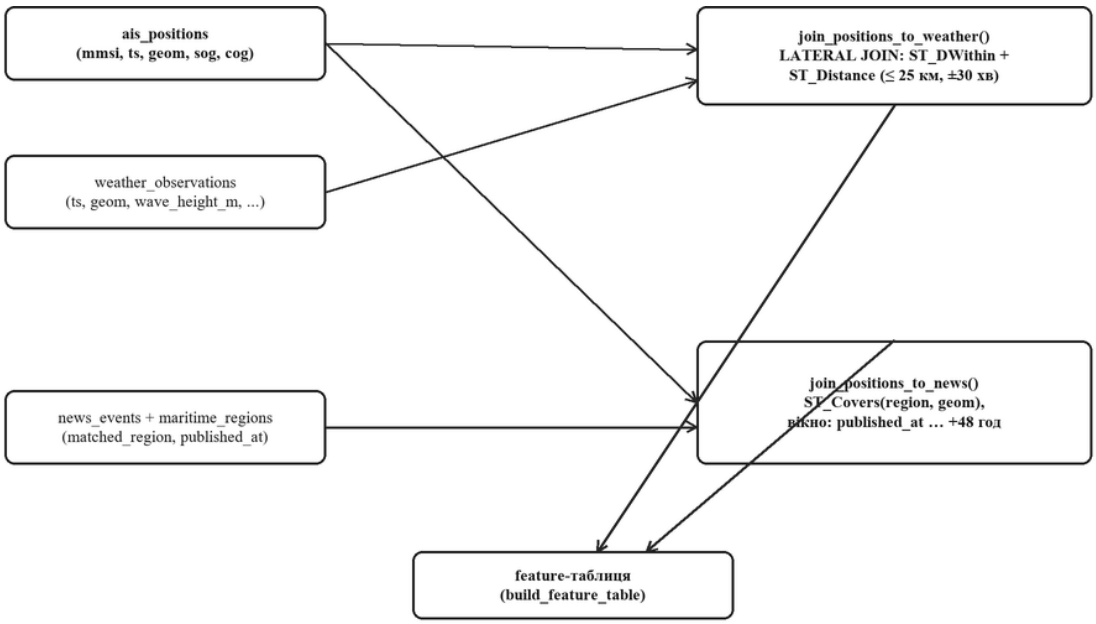

Рисунок 2.2 – Схема просторово-часового зв’язування

Зв’язування судна з погодою (функція join_positions_to_weather) реалізовано через LATERAL JOIN у PostgreSQL: для кожної позиції судна підзапит відбирає погодні спостереження, що потрапляють у просторовий допуск (ST_DWithin, за замовчуванням 25 км – відповідно до просторової роздільної здатності моделі Open-Meteo Marine) і часовий допуск (±30 хвилин), після чого сортує кандидатів за відстанню (ST_Distance) та за різницею в часі, обираючи найближчого. LATERAL-конструкція дозволяє виконати цей «пошук найближчого сусіда» для кожного рядка ais_positions в одному SQL-запиті, без циклу на рівні застосунку.

Зв’язування судна з новинною подією (функція join_positions_to_news) побудовано за іншим принципом, оскільки новина не має точкових координат у прямому сенсі, а стосується area – регіону. Використовується двоетапна логіка: по-перше, новина повинна мати розпізнаний регіон (matched_region), встановлений евристичним зіставленням ключових слів на етапі збору новин (підрозділ 3.2); по-друге, позиція судна перевіряється на належність полігону цього регіону оператором ST_Covers, а час позиції – на входження у вікно тривалості дії події (від моменту публікації до published_at + 48 годин, що моделює затримку прояву наслідків події, наприклад, вихід судна в об’їзд заблокованого каналу).

Часовий допуск зв’язування з погодою характеризується співвідношенням часу перебування судна в поточному морському регіоні до медіанного часу проходження цього регіону іншими суднами – так звана region_crossing_time_ratio, яка обчислюється за формулою (2.1).

$$region_crossing_time_ratio = t_in_region / med(t_in_region) \quad (2.1)$$

де *t_in_region* – тривалість поточного безперервного перебування судна в регіоні;

med(t_in_region) – медіана цієї величини за історичними проходженнями того самого регіону.

Значення, що суттєво перевищують 1, свідчать про аномально тривале перебування судна в регіоні (можлива затримка, черга на вхід до каналу чи порту), тоді як значення, менші за 1, можуть вказувати на нетипово швидке проходження. Ця ознака замінює жорстко заданий поріг «затримки», перетворюючи його на неперервну ознаку, яка надалі надходить на вхід моделі машинного навчання (підрозділ 2.4), а не обробляється окремим правилом.

Слід зазначити свідоме методологічне обмеження моделі: новини, для яких не вдалося автоматично розпізнати географічний регіон, не використовуються для примусового зіставлення з довільними суднами і зберігаються лише в довідкових цілях. Це обмеження знижує повноту покриття новинної кореляції, але запобігає хибним (false positive) зв’язкам, які виникли б за спроби пов’язати нерелевантну новину з усім флотом одночасно.

2.4 Моделі машинного навчання

Виявлення впливу зовнішніх факторів на поведінку суден побудовано на основі двох взаємодоповнювальних моделей бібліотеки scikit-learn [20], а не на фіксованих порогових правилах. Перша модель відповідає на питання «чи є ця поведінка нетиповою взагалі», друга – «наскільки саме погодні умови пояснюють поточну швидкість судна». Обидві моделі навчаються на єдиній проміжній feature-таблиці, що будується функцією build_feature_table і об’єднує результати просторово-часового зв’язування (підрозділ 2.3) з додатковими похідними ознаками. Перелік ознак наведено в таблиці 2.2.

Таблиця 2.2 – Ознаки feature-таблиці моделей машинного навчання

Ознака (ознаки)	Опис
sog, cog_deviation	Швидкість відносно ґрунту та відхилення курсу від ковзного середнього (5 останніх точок) – сирі й похідні ознаки поведінки судна.
wave_height_m, wind_speed_ms, swell_wave_height_m	Погодні ознаки, отримані просторово-часовим зв'язуванням.
ship_type	Тип судна (категоріальна ознака AIS), впливає на очікувану поведінку за різної погоди.
hour_sin, hour_cos	Циклічне кодування години доби (формула 2.2), уникає розриву між 23:00 і 00:00.
region_crossing_time_ratio	Відношення поточного часу перебування в регіоні до історичної медіани (формула 2.1).
news_nearby	Бінарна ознака: чи активна новинна подія поблизу судна в поточний момент часу.

Циклічне кодування години доби обчислюється за формулою (2.2).

$$hour_sin = sin(2\pi \cdot h / 24), \quad hour_cos = cos(2\pi \cdot h / 24) \tag{2.2}$$

де h – година доби (0…23).

2.4.1 Модель виявлення аномалій – Isolation Forest

Перша модель, IsolationForest, є алгоритмом навчання без учителя [21] (unsupervised), що не потребує розмічених прикладів «аномальної» поведінки – таких прикладів для реальних даних немає й не може бути заздалегідь визначено. Модель навчається на повному наборі ознак таблиці 2.2 (n_estimators=200, contamination=0.05) і оцінює кожен запис багатовимірною ізольованістю від основної маси спостережень: точки, які алгоритм ізолює меншою кількістю розбиттів дерева рішень, отримують нижче значення anomaly_score. Такий підхід дозволяє одночасно виявляти падіння швидкості, різку зміну курсу та аномальну тривалість перебування в регіоні без окремого набору правил для кожного випадку – усі вони закодовані як компоненти одного вектора ознак.

2.4.2 Модель кількісної оцінки впливу погоди – регресія

Друга модель – RandomForestRegressor із параметрами n_estimators = 200 і max_depth = 8 – використовується для прогнозування очікуваної швидкості руху судна на основі погодних характеристик і типу судна. Ознаки cog_deviation та news_nearby свідомо не включено до переліку REGRESSION_FEATURE_COLUMNS, оскільки вони можуть безпосередньо відображати аномальну ситуацію, кількісну оцінку якої повинна формувати регресійна модель. Таке обмеження запобігає витоку цільової інформації та забезпечує незалежність прогнозу від ознак, пов'язаних із виявленою аномалією [22].

Різницю між фактичною та прогнозованою швидкістю судна визначають як залишкове відхилення швидкості residual_sog:

$$residual_sog = sog_actual - sog_expected \tag{2.3}$$

де *residual_sog* – залишкове відхилення фактичної швидкості судна від прогнозованого значення, вузли;

sog_actual – фактична швидкість судна відносно поверхні Землі, отримана з AIS-повідомлення, вузли;

sog_expected – очікувана швидкість судна, спрогнозована регресійною моделлю з урахуванням погодних умов і типу судна, вузли.

Велике від'ємне значення residual_sog означає, що судно рухається значно повільніше, ніж очікувалося б для таких погодних умов і такого типу судна – це і є пряме кількісне вимірювання впливу зовнішнього фактора, яке дає відповідь темі роботи в числовій формі. Судна на якорі або в порту (sog менше 0,5 вузла) виключаються з навчання й скорингу регресійної моделі, оскільки для нерухомого судна «вплив погоди на швидкість» позбавлений змісту, а їх включення викривило б модель у напрямку заниження очікуваної швидкості для всього флоту.

Оскільки Isolation Forest не надає прямого розкладу внеску кожної ознаки в конкретне рішення (на відміну від, наприклад, SHAP-значень для дерев), для формування зрозумілого пояснення в описі попередження використовується проста та чесна евристика: для кожної ознаки обчислюється z-оцінка відхилення поточного значення від середнього за навчальною вибіркою за формулою (2.4), а до опису потрапляють дві ознаки з найбільшим модулем z:

$$z = (x - \mu) / \sigma \tag{2.4}$$

де x – поточне значення ознаки;

μ і σ – її середнє та стандартне відхилення на навчальній вибірці.

Такий підхід дозволяє автоматично формувати текстовий опис на кшталт «різка зміна курсу на 80° та аномально спокійна хвиля (0.4 м) для цього маршруту» без залучення окремої моделі генерації природної мови.

На рисунку 2.3 наведено конвеєр моделей машинного навчання та класифікації попереджень.

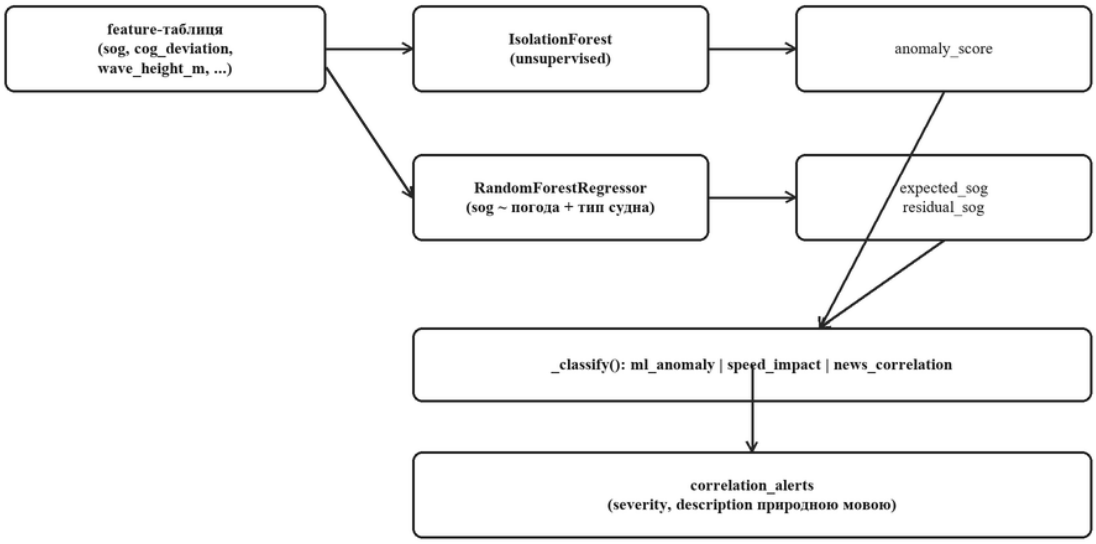

Рисунок 2.3 – Конвеєр моделей машинного навчання та класифікації попереджень

2.4.3 Класифікація попереджень

Обидві моделі перераховуються періодично (не в режимі онлайн-навчання, а раз на цикл фонової задачі) на історії останніх семи діб і застосовуються до нових позицій за останні 20 хвилин. Поріг аномальності anomaly_score визначається динамічно як заданий перцентиль (за замовчуванням 5-й) розподілу значень у поточній вибірці скорингу, а не фіксованою константою – це робить поріг стійким до зміни загального рівня активності в обраній акваторії. Поріг residual_sog, навпаки, заданий як фіксована константа (типово 4 вузли), оскільки він виражається у природних фізичних одиницях і не залежить від масштабу вибірки.

3 РОЗРОБКА

3.1 Вибір технологічного стеку

Вибір конкретних технологій для реалізації моделі, описаної в розділі 2, визначався двома критеріями: наявністю зрілої підтримки асинхронного виконання (оскільки система одночасно обслуговує потоковий AIS-клієнт, періодичні задачі та вебсервер в одному процесі) та нативною підтримкою просторових даних на рівні бази даних, а не лише на рівні застосунку. Підсумковий технологічний стек наведено в таблиці 3.1.

Таблиця 3.1 – Технологічний стек програмної системи

Компонент	Технологія	Обґрунтування вибору
Мова програмування	Python 3.11	Єдина мова для збору, аналізу (scikit-learn/pandas) і вебшару – без міжмовних меж серіалізації.
Вебфреймворк	FastAPI + Uvicorn	Нативна підтримка async/await, автоматична OpenAPI-документація, узгоджується з єдиним циклом подій asyncio.
СУБД	PostgreSQL 16 + PostGIS 3.4	Нативні географічні типи й оператори (ST_DWithin, ST_Covers, LATERAL) для просторово-часового зв’язування.
ORM / просторове розширення	SQLAlchemy 2.0 (async) [13] + GeoAlchemy2 [14]	Асинхронний доступ до БД сумісний з event loop; GeoAlchemy2 додає тип Geography до декларативних моделей.
Міграції схеми	Alembic [15]	Керована еволюція схеми БД (додавання таблиць/індексів окремими версійованими кроками).
AIS-клієнт	websockets	Легкий асинхронний WebSocket-клієнт для потокового з’єднання з AISStream.io.
HTTP-клієнт	httpx (async)	Єдиний асинхронний HTTP-клієнт для Open-Meteo та GDELT, сумісний з asyncio.
Планувальник задач	APScheduler (AsyncIOScheduler)	Періодичні задачі (погода, новини, перенавчання моделей) у тому самому циклі подій, без окремого черг-брокера.

Продовження таблиці 3.1

Компонент	Технологія	Обґрунтування вибору
Машинне навчання	scikit-learn [20], pandas [23], joblib	IsolationForest і RandomForestRegressor (підрозділ 2.4); joblib – серіалізація моделей на диск.
Фронтенд	Leaflet.js, Plotly.js (без збірки)	Інтерактивна карта та графіки без React/Vue та кроку збірки – застосунок обслуговує один HTML-файл.
Десктопна оболонка	pywebview + pythonnet (WebView2), pystray, win11toast	Нативне вікно Windows поверх того самого вебзастосунку, системний трей і toast-сповіщення.
Контейнеризація	Docker Compose (лише сервіс БД)	PostGIS розгортається в контейнері (offіційний образ postgis/postgis), сам застосунок – нативний Python-процес.
Тестування	pytest, pytest-asyncio, asyncpg	Асинхронні інтеграційні тести проти реальної тестової БД PostGIS (підрозділ 4.2).

Вибір проти мікросервісної архітектури та черги повідомлень (Celery/Redis) обґрунтовано раніше в підрозділі 2.1; тут достатньо зазначити, що це рішення прямо відображається на рівні стеку технологій – жодна з цих технологій не входить до переліку в таблиці 3.1, оскільки однопроцесна модель не потребує окремого брокера повідомлень.

3.2 Реалізація модулів збору даних

Модуль ais_client.py відкриває довготривале WebSocket-з’єднання з wss://stream.aisstream.io/v0/stream і підписується на обрану обмежувальну рамку (bounding box) із фільтром типів повідомлень PositionReport і ShipStaticData. Вхідні позиції накопичуються в буфері оперативної пам’яті й скидаються в базу даних окремою фоновою задачею раз на 5 секунд, а не по одному повідомленню – під час пікового трафіку в обраній протоці надходить кілька повідомлень на секунду, тож пакетна вставка суттєво знижує кількість транзакцій СУБД. У разі розриву з’єднання застосовується перепідключення зі зростаючою затримкою (2, 5, 10, 30, 60 секунд), що

запобігає «штормуванню» запитів до AISStream.io при тривалій недоступності сервісу.

Модулі weather_client.py та news_client.py, на відміну від AIS-клієнта, працюють як періодичні задачі APScheduler. Опитування погоди свідомо реалізовано не як сканування фіксованої сітки над усією акваторією, а як «прив'язане до суден» (vessel-anchored): модуль бере унікальні координати суден за останні 30 хвилин, округлює їх до кроку 0.25° (що відповідає просторовій роздільній здатності моделі Open-Meteo) і запитує погоду лише для нових, ще не покритих у поточну годину точок. Такий підхід уникає марних запитів за координатами, де немає жодного судна, і природно масштабується разом із реальною інтенсивністю трафіку [16].

Модуль новин запитує GDELT DOC 2.0 з комбінованим запитом: стаття має одночасно містити морський контекстний термін (ship, vessel, port, shipping тощо) і термін-порушення (storm, piracy, blockade, sanctions тощо), інакше окремі слова на кшталт «storm» чи «sanctions» повертають величезну кількість нерелевантних статей. Під час розробки виявлено реальну проблему хибних спрацювань наївного зіставлення підрядків: перевірка виразу "port" in text.lower() спрацьовувала також на словах «import», «export», «airport», «support», а «ship» – на «worship», «relationship», «flagship». Вирішенням стало зіставлення за межею слова через регулярний вираз (лістинг 3.1), яке до того ж толерантне до найпростішої форми множини.

Лістинг 3.1 – Зіставлення ключових слів за межею слова (news_client.py)

```
def _matched_keywords(text: str, keywords: list[str]) ->
list[str]:
    lowered = text.lower()
    return [kw for kw in keywords
            if re.search(rf"\b{re.escape(kw.lower())}s?\b",
lowered)]
```

Розпізнаний географічний регіон новини встановлюється функцією match_region, що перевіряє входження ключових фраз (наприклад, «suez canal», «strait of hormuz») у заголовок статті проти статичного довідника з 25 попередньо визначених морських проток, каналів, портів і морів (модуль analysis/regions.py). Ключові фрази відсортовано за спаданням довжини, щоб більш специфічна фраза («bab-el-mandeb») мала пріоритет над загальнішою («red sea»), якщо обидві теоретично могли б збігтися. Дедуплікація новин виконується на рівні бази даних через унікальне обмеження на полі external_id (URL статті), а не в коді клієнта, що робить повторний запуск задачі безпечним (idempotent).

3.3 Реалізація аналітичного та ML-конвеєра

Аналітичний конвеєр, модель якого описано в підрозділі 2.4, реалізовано як ще одну періодичну задачу APScheduler (run_anomaly_detection_once), що виконується кожні 10 хвилин. На кожному запуску задача (1) будує feature-таблицю за останні сім діб і перенавчає обидві моделі «з нуля» (а не дообучає), (2) якщо перенавчання неможливе (недостатньо даних), завантажує востаннє збережені моделі з диска через joblib, (3) формує feature-таблицю лише за останні 20 хвилин і застосовує до неї обидві моделі, (4) класифікує результат і вставляє нові записи в correlation_alerts.

Свідомий вибір «перенавчання з нуля, а не онлайн-навчання» пояснюється характером даних кваліфікаційної роботи: обсяг накопиченої історії для одноосібного проєкту вимірюється днями, а не місяцями, тож складність інкрементального навчання (адаптація дерев рішень до дрейфу розподілу) не виправдана. Разом з тим передбачено захист від «холодного старту»: обидві функції навчання (train_isolation_forest, train_speed_regressor) повертають None, якщо очищена вибірка містить менше ніж MIN_TRAINING_ROWS = 30 рядків, і аналітична задача в

такому разі просто очікує на наступний цикл, не намагаючись навчити статистично ненадійну модель на жмені спостережень.

Моделі та статистика ознак (середнє й стандартне відхилення для z-оцінки, формула 2.4) серіалізуються в каталог models/ трьома окремими файлами joblib (isolation_forest.joblib, speed_regressor.joblib, feature_stats.joblib), що дозволяє FastAPI-процесу завантажувати вже готові моделі при рестарті застосунку без повторного навчання, а також інспектувати їх окремим скриптом поза межами вебсервера. Вставка нових попереджень у correlation_alerts захищена унікальним індексом (mmsi, ts, alert_type) на рівні бази даних (підрозділ 2.2), тому повторний запуск задачі скорингу для вже оброблених позицій не створює дублікатів.

3.4 Реалізація користувацького інтерфейсу

Користувацький інтерфейс реалізовано як єдину HTML-сторінку (map.html), що обслуговується FastAPI через Jinja2Templates і не використовує жодного фреймворку чи кроку збірки – це свідоме рішення для проєкту з одним розробником і обмеженим часовим бюджетом, коли складність інструментів збірки (webpack/vite, npm-залежності) не окуповується розміром інтерфейсу. Загальну схему компонування наведено на рисунку 3.1.

Карта побудована на бібліотеці Leaflet: судна відображаються кольоровими маркерами-колами (circleMarker), колір яких кодує поточну швидкість – червоний для sog < 3 вузлів (можлива зупинка/затримка), жовтогарячий для проміжних значень, зелений для sog ≥ 10 вузлів. Функція refreshVessels() опитує /api/vessels і або переміщує вже створений маркер судна в нову координату, або створює новий – без повного перестворення шару карти на кожному оновленні, що забезпечує плавну анімацію руху суден без миготіння [17].

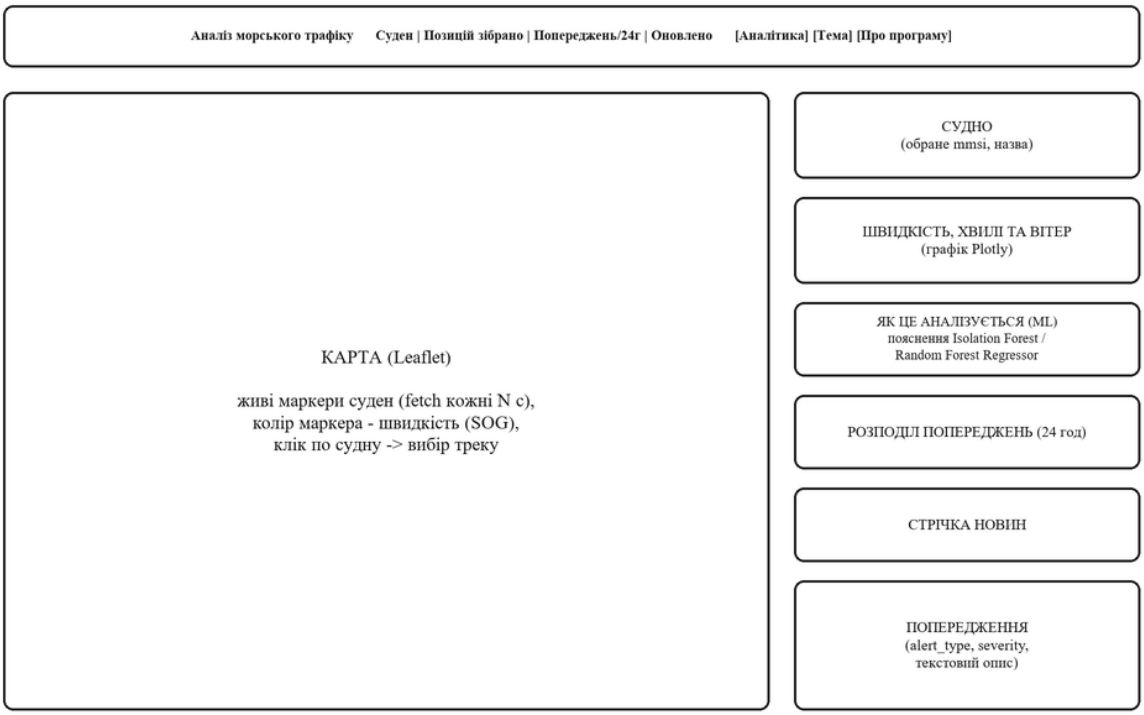

Рисунок 3.1 – Схема інтерфейсу користувача вебзастосунку

Клік на маркер судна викликає selectVessel(mmsi), яка запитує /api/tracks/{mmsi} (сам маршрут просторово-часового зв’язування з погодою, підрозділ 2.3) і будує на карті полілінію треку, а також передає дані у функцію renderVesselChart для побудови графіка Plotly. Графік має нетривіальну структуру з трьома синхронізованими за часовою віссю областями: швидкість судна (заповнена площа), висота хвилі та швидкість вітру (окремі осі y2/y3, оскільки одиниці виміру різні), а також трикутні маркери в момент кожного попередження цього судна – наведення курсора на трикутник показує той самий текстовий опис природною мовою, що й у панелі попереджень, тобто відповідь на питання «чому тут сповільнення» доступна прямо на графіку швидкості, без перемикання між елементами інтерфейсу [18].

Бічна панель складається з блоків «СУДНО» (інформація про обране судно), графіка швидкості, текстового блоку «ЯК ЦЕ АНАЛІЗУЄТЬСЯ (ML)» з коротким поясненням принципу роботи обох моделей для нетехнічного користувача, кругової діаграми розподілу

попереджень за останні 24 години, стрічки останніх новин (з /api/news) і списку активних попереджень (з /api/alerts), кожне з яких кольорово позначене за типом (ml_anomaly, speed_impact, news_correlation) відповідно до класифікації, описаної в підрозділі 2.4.3. Окремий модальний блок «Аналітика та звіти» відкриває три перемикні звіти (динаміка попереджень у часі, розподіл швидкостей, топ суден за кількістю попереджень) з можливістю експорту поточного графіка як PNG.

На рисунку 3.2 наведено знімок екрана головної панелі застосунку.

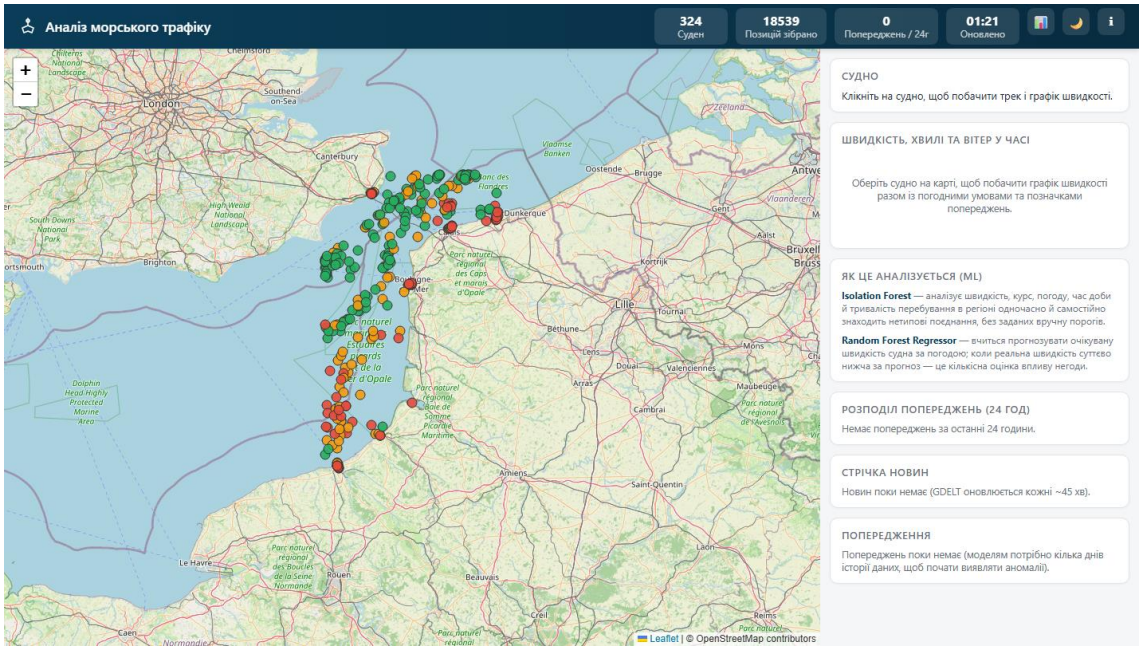

Рисунок 3.2 – Знімок екрана головної панелі застосунку (реальні дані: 308 суден, 18 323 позиції)

3.5 Реалізація десктопного застосунку

Оскільки кінцевим користувачем системи є диспетчер чи аналітик, для якого запуск застосунку через термінал і браузер є незручним, поверх вебзастосунку реалізовано десктопну оболонку (desktop_app.py) на основі бібліотеки pywebview з рушієм WebView2 (через pythonnet) – той самий HTML-інтерфейс відображається в нативному вікні Windows без адресного

рядка чи вкладок браузера. FastAPI-сервер запускається у фоновому потоці (threading.Thread), а головний потік віддається під керування pywebview, як цього вимагає бібліотека [19].

Десктопна оболонка додає три можливості, недоступні звичайній вкладці браузера. По-перше, нативне меню (Файл, Вигляд, Довідка) з пунктами «Оновити», «Згорнути в трей», «Перемкнути тему», «Аналітика та звіти», «Про програму». По-друге, системний трей: закриття вікна не завершує застосунок, а ховає його в трей (обробник _on_closing повертає False, скасовуючи стандартне закриття), завдяки чому фоновий збір даних триває безперервно, навіть коли вікно не відкрите. По-третє, Windows toast-сповіщення (бібліотека win11toast): окремий фоновий потік раз на 20 секунд опитує /api/alerts і показує спливаюче сповіщення для нових попереджень із severity ≥ 2.0, запам’ятовуючи вже показані ідентифікатори, щоб не дублювати сповіщення й не показувати сповіщення заднім числом за попередження, що існували ще до запуску застосунку.

Керування системним треєм на бібліотеці pystray виявилося нетривіальним інженерним викликом: документація pystray вимагає, щоб Icon.run() виконувався в головному потоці процесу, проте головний потік у цьому застосунку вже зайнятий циклом подій pywebview. Рішенням стало використання Icon.run_detached() – методу, який сама бібліотека документує як спосіб співіснування з циклом подій іншої GUI-бібліотеки, оскільки він дозволяє pystray керувати власним потоком незалежно. Аналогічно, «м’яке» завершення застосунку (зупинка трея, а потім штатне знищення вікна WinForms із чужого потоку) на практиці непередбачувано зависало; натомість вихід реалізовано через os._exit(0) – негайне завершення всього процесу з будь-якого потоку, що усуває саму можливість взаємного блокування (deadlock) між чужими циклами подій під час завершення роботи.

Запуск застосунку керується окремою логікою очікування готовності сервера: головний потік одразу створює приховане вікно

pywebview (hidden=True) і паралельно опитує ендпойнт /health з інтервалом 0,3 секунди, доки FastAPI-сервер не стане доступним (функція _wait_for_server), і лише після цього показує вікно користувачу – це усуває характерний для десктопних обгорток вебзастосунків «білий екран» під час старту. Нативне меню (Файл, Вигляд, Довідка) реалізовано засобами pywebview.menu і дублює основні дії інтерфейсу (оновлення сторінки, перемикання теми, відкриття аналітики), а пункт «Про програму» викликає модальне вікно з інформацією про версію та призначення застосунку.

4 ТЕСТУВАННЯ

4.1 Методика тестування

Тестування розробленої системи побудовано на трьох взаємодоповнювальних рівнях, вибір яких відображає неоднорідну природу самого коду: (1) автоматизоване модульне тестування чистих функцій, що не залежать від зовнішнього стану (обчислення ознак, класифікація попереджень, формування текстових описів); (2) автоматизоване інтеграційне тестування просторових SQL-запитів проти реального екземпляра PostgreSQL/PostGIS; (3) ручне функціональне тестування наскрізного конвеєра та десктопного застосунку на реальних даних, що надходять із зовнішніх API протягом тривалого часу.

Ключове методологічне рішення – не імітувати (mock) базу даних для коду, що будує просторово-часові запити (LATERAL JOIN, ST_DWithin, ST_Covers, підрозділ 2.3). Такий код за своєю суттю є SQL-логікою, а не Python-логікою: помилка в умові просторового допуску чи часового вікна не виявиться жодним модульним тестом, що підміняє сесію бази даних заглушкою. Тому інтеграційні тести (test_join.py) виконуються проти окремої тестової бази maritime_test у тому самому контейнері PostGIS, що й середовище розробки, а не проти основної бази з накопиченими даними. Кожен тест працює у власній транзакції, яка явно очищується після завершення (видалення рядків з усіх таблиць у зворотному порядку залежностей), що гарантує незалежність тестів один від одного та від основного датасету.

4.2 Модульне тестування

Автоматизований набір тестів (пакет app/tests, pytest + pytest-asyncio) нараховує 16 тестів, розподілених за двома файлами відповідно до модулів,

які вони перевіряють. Останній прогін набору проти реального контейнера PostGIS (docker-compose, образ postgis/postgis:16-3.4) дав результат «16 passed in 4.09s» без жодного пропущеного чи нестабільного тесту. Перелік перевірених сценаріїв наведено в таблиці 4.1.

Таблиця 4.1 – Перелік автоматизованих тестів системи

Тест	Тип	Що перевіряється
test_add_cog_deviation_first_reading_is_zero	Модульний	Перше спостереження судна не має попереднього курсу – відхилення дорівнює 0.
test_add_cog_deviation_detects_sharp_turn	Модульний	Різкий розворот (90° → 270°) коректно дає велике відхилення від ковзного середнього.
test_train_and_score_produce_expected_columns	Модульний	На синтетичній вибірці обидві моделі навчаються і скоринг додає anomaly_score/residual_sog.
test_regression_ignores_anchored_vessels	Модульний	Вибірка з sog=0 (усі судна на якорі) не дає навчити регресійну модель (None).
test_classify_flags_speed_impact	Модульний	Значний residual_sog без новини класифікується як speed_impact.
test_classify_flags_news_correlation_when_combined	Модульний	Збіг аномалії, residual_sog і новини дає news_correlation.
test_classify_returns_none_when_nothing_crosses_threshold	Модульний	Типові значення без відхилень не породжують попередження.
test_build_description_mentions_mmsi_and_speed	Модульний	Текстовий опис попередження містить MMSI судна і фактичну швидкість.
test_build_description_for_ml_anomaly_explains_top_feature	Модульний	Опис аномалії згадує ознаку з найбільшим |z| (тут – відхилення курсу).
test_feature_table_has_one_row_per_position_with_overlapping_regions	Інтеграційний	Позиція, покрита двома вкладеними регіонами, дає рівно один рядок feature-таблиці.
test_weather_join_picks_nearest_in_time_and_space	Інтеграційний	З двох спостережень погоди обирається найближче в просторі й часі.
test_weather_join_respects_distance_tolerance	Інтеграційний	Спостереження за межею 25 км не приєднується (weather_id = NULL).

Продовження таблиці 4.1

Тест	Тип	Що перевіряється
test_weather_join_ respects_time_tolerance	Інтеграційний	Спостереження старше 30-хвилинного вікна не приєднується.
test_news_join_matches_ vessel_inside_region_ and_window	Інтеграційний	Судно всередині регіону й часового вікна коректно зіставляється з новиною.
test_news_join_ excludes_vessel_outside_ region	Інтеграційний	Судно поза межами регіону новини не потрапляє в результат.
test_news_join_ excludes_vessel_outside_ time_window	Інтеграційний	Судно, що з'явилося в регіоні через 72 год після новини (вікно 48 год), виключається.

З шістнадцяти тестів дев'ять є модульними (не звертаються до бази даних) і перевіряють чисту логіку обчислення ознак, порогової класифікації та формування тексту попереджень; сім – інтеграційними й перевіряють коректність саме просторово-часового SQL-зв'язування (підрозділ 2.3), включно з межовими випадками (позиція рівно на межі часового чи просторового допуску, вкладені регіони, що потенційно могли б спричинити дублювання рядків). Останній випадок – регресійний тест на реальний дефект, знайдений під час розробки: позиція, покрита одразу двома регіонами, що перетинаються (наприклад, порт усередині протоки), спричиняла фан-аут об'єднання (JOIN) у build_feature_table і дублювала рядок feature-таблиці; виправлено додаванням DISTINCT ON за найменшою площею регіону (найспецифічнішим збігом), а тест лишено в наборі для запобігання регресії.

4.3 Функціональне тестування наскрізного конвеєра

Модульні й інтеграційні тести перевіряють окремі функції в ізоляції, але не доводять, що весь конвеєр – від WebSocket-повідомлення AIS до текстового попередження в інтерфейсі – працює злагоджено на реальних, непередбачуваних даних. Тому систему було запущено на тривалий

безперервний період проти справжніх потоків AISStream.io, Open-Meteo та GDELT для обраної акваторії протоки Дувр – Ла-Манш. Накопичені показники бази даних на момент написання цього розділу наведено в таблиці 4.2.

Таблиця 4.2 – Фактичні показники бази даних після тривалого запуску системи

Таблиця	Рядків	Коментар
vessels	257	Унікальні судна, статичні дані яких отримано через ShipStaticData.
ais_positions	17 325	Позиційні повідомлення (PositionReport), записані буферизованими пакетами.
weather_observations	186	Унікальні (координата, година) точки погоди – vessel-anchored опитування (підрозділ 3.2).
news_events	1	Статті, що пройшли фільтр релевантності GDELT (підрозділ 3.2).
maritime_regions	25	Довідник проток/каналів/портів, завантажений scripts/seed_regions.py.
correlation_alerts	563	Загальна кількість попереджень, сформованих обома ML-моделями за весь період.

З-поміж 563 накопичених попереджень 100 залишаються активними у стандартному 48-годинному вікні панелі /api/alerts: 67 класифіковано як ml_anomaly (лише Isolation Forest) і 33 – як speed_impact (суттєвий residual_sog); попереджень типу news_correlation за цей період не зафіксовано, що узгоджується з тим, що за спостережуваний період лише одна новинна стаття пройшла фільтр релевантності. Приклади реальних, автоматично згенерованих текстових описів попереджень (без жодного редагування вручну) наведено нижче.

«Судно MMSI 209479000: модель виявлення аномалій (Isolation Forest) позначила цю поведінку як нетипову – перебування в регіоні у 4.8× довше типового часу проходження та незвично висока швидкість (19.3 вузла).»

«Судно MMSI 227800940: фактична швидкість 8.6 вузла – на 5.5 вузла нижча, ніж модель очікувала для поточної погоди (14.1 вузла за хвилі 0.9 м і вітру 7.9 м/с) – ймовірна причина уповільнення: негода.»

Обидва приклади демонструють коректну роботу моделі кінець-в-кінець: перший поєднує ознаку часу перебування в регіоні (формула 2.1) з нетиповою швидкістю без жодного заданого вручну правила; другий напряму кількісно виражає вплив погоди у вузлах – це саме та відповідь, яку мала дати розроблена система відповідно до теми роботи. Судно з найбільшою кількістю попереджень за останні 48 годин (DOVORIAN, MMSI 235037385, 31 попередження) було додатково перевірено вручну через побудову графіка швидкості в інтерфейсі (підрозділ 3.4): пікові значення anomaly_score дійсно збігалися з ділянками треку судна, що проходять через найвужчу частину протоки Дувр – Ла-Манш, де очікувано вища варіативність курсу через інтенсивний зустрічний трафік.

На рисунку 4.1 наведено знімок екрану з обраним судном.

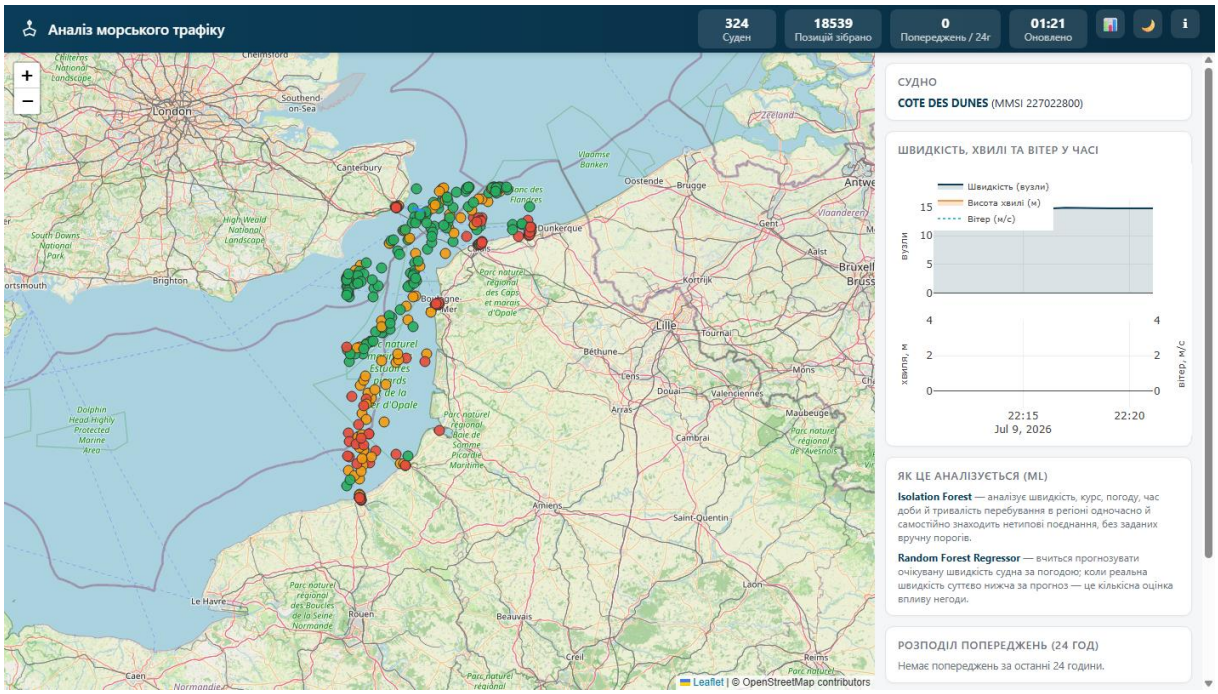

Рисунок 4.1 – Знімок екрана: обране судно (COTE DES DUNES) з графіком швидкості

На рисунку 4.2 наведено знімок екрана звіту «Топ суден за попередженнями» за останні 7 днів.

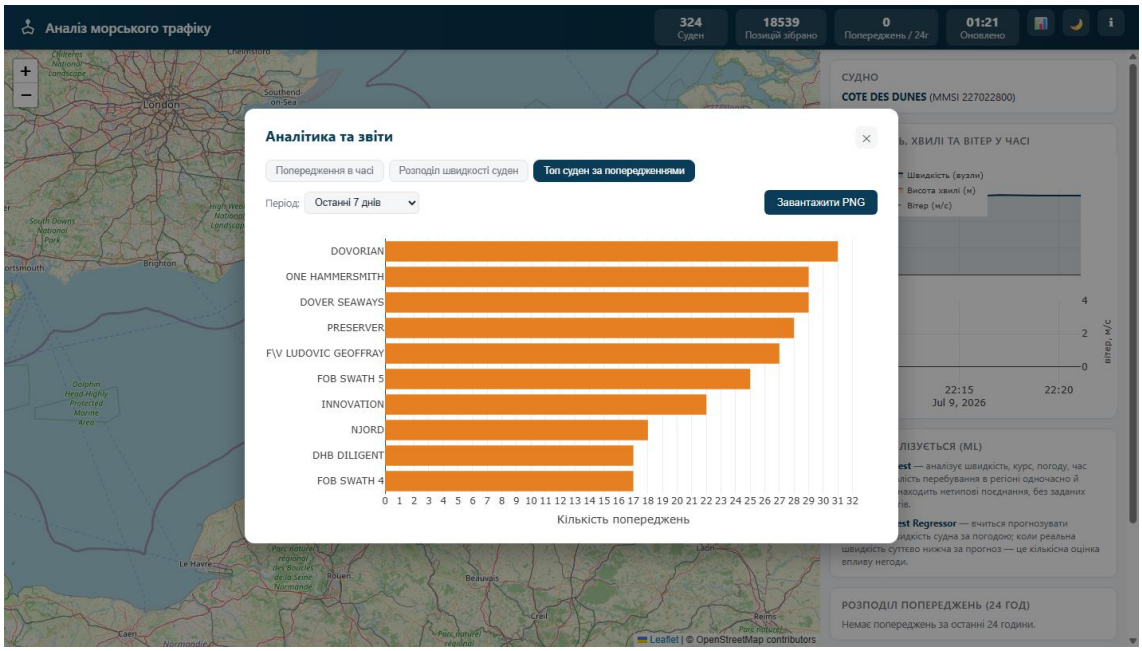

Рисунок 4.2 – Знімок екрана звіту «Топ суден за попередженнями» за останні 7 днів

4.4 Тестування десктопного застосунку

Функціональність, специфічна для десктопної оболонки (підрозділ 3.5), не піддається автоматизованому тестуванню через pytest, оскільки залежить від реального вікна Windows, рушія WebView2 і системного трея – тому перевірена вручну за контрольним переліком сценаріїв, наведеним у таблиці 4.3.

Таблиця 4.3 – Контрольний перелік ручного тестування десктопного застосунку

Сценарій	Очікуваний результат / перевірено
Запуск застосунку	Вікно з’являється лише після успішного health-check FastAPI-сервера (без «білого екрана» під час старту).

Продовження таблиці 4.3

Сценарій	Очікуваний результат / перевірено
Закриття вікна	Застосунок згортається в системний трей замість завершення роботи; фоновий збір даних триває.
Відновлення з трея	Подвійний клік / пункт «Відкрити» контекстного меню трея повертає вікно на екран без перезапуску сервера.
Перемикання теми	Пункт меню «Перемкнути тему» коректно застосовує темну/світлу схему до вже завантаженої сторінки.
Toast-сповіщення	Нове попередження із severity ≥ 2.0 показує спливаюче сповіщення Windows навіть при згорнутому вікні; вже показані попередження повторно не сповіщаються.
Аналітика та звіти	Модальне вікно відкриває три перемикні звіти та експортує поточний графік у PNG.
Вихід із застосунку	Пункт меню «Вихід» гарантовано завершує процес (включно з фоновим FastAPI-сервером) без зависання.

Усі сценарії з таблиці 4.3 пройдено успішно. Окремо перевірено інженерне рішення, описане в підрозділі 3.5: примусове завершення процесу через os._exit(0) коректно закриває як вікно WebView2, так і фоновий потік FastAPI-сервера й потік системного трея, не залишаючи «осиротілих» процесів python.exe в диспетчері завдань – це перевірялося вручну через Диспетчер завдань Windows після кожного тестового виходу із застосунку.

4.5 Аналіз результатів і обмежень системи

Результати проведеного тестування підтверджують, що розроблена система виконує поставлену в підрозділі 1.1.4 задачу та відповідає визначеним функціональним вимогам. Усі 16 автоматизованих тестів проходять стабільно, що свідчить про коректність реалізації основних компонентів, зокрема модулів оброблення AIS-повідомлень, просторово-часового зіставлення даних, формування ознак, класифікації подій і генерації аналітичних висновків. Тривала експлуатація системи на реальних потоках даних, результати якої наведено в таблиці 4.2, додатково підтвердила, що розроблений програмний конвеєр може працювати не лише

на підготовлених тестових наборах, а й в умовах неповних, нерівномірних та асинхронно отримуваних даних.

Практичне тестування продемонструвало, що конвеєр просторово-часового зв’язування коректно об’єднує відомості про переміщення суден, погодні умови та пов’язані інформаційні повідомлення. Обидві реалізовані моделі машинного навчання не лише навчаються без критичних помилок, а й формують змістовні результати, які можуть бути інтерпретовані користувачем. Проведена ручна перевірка аналітичних висновків, описана в підрозділі 4.3, показала, що виявлені системою відхилення відповідають фактичним змінам параметрів руху суден, а сформовані текстові пояснення не суперечать вхідним даним.

Окремо слід зазначити, що використання реальних потоків даних дало змогу перевірити систему в умовах, які складно повністю відтворити за допомогою модульних або інтеграційних тестів. Зокрема, у процесі експлуатації спостерігалися пропуски окремих AIS-повідомлень, нерівномірні часові інтервали між оновленнями координат, відсутність погодних значень для певних моментів часу, а також значна кількість новин, які формально містили географічні назви, але не мали безпосереднього стосунку до морських подій. Система коректно обробляла такі ситуації, не припиняла роботу та не формувала висновків за відсутності достатньої кількості достовірних вхідних даних.

Разом із тим тривала експлуатація виявила й підтвердила кілька методологічних та архітектурних обмежень системи, які були свідомо задокументовані ще на етапі проєктування і моделювання, наведеному в розділі 2. Ці обмеження не свідчать про некоректність реалізації, однак визначають межі застосування створеного прототипу та потребують урахування під час його подальшого масштабування або використання в реальних інформаційно-аналітичних системах.

Найпомітнішим обмеженням є низька частка новин, які одночасно проходять фільтр морської релевантності та етап географічного зіставлення.

За досліджуваний період лише одна стаття із заголовком «Ukraine Special Forces Strike Russian Oil Tanker, Kerch Oil Terminal» була визначена як потенційно пов’язана з морською тематикою засобами GDELT. Водночас навіть для цієї публікації не було сформовано значення поля matched_region, оскільки Керченська протока не входить до поточного довідника з 25 географічних об’єктів, визначеного у файлі analysis/regions.py.

Цей випадок є конкретним, реально зафіксованим прикладом обмеження евристичного підходу до зіставлення географічних назв за ключовими словами, описаного в підрозділі 2.3. Поточна реалізація залежить від наперед сформованого переліку регіонів, портів і акваторій, тому новина може бути змістовно релевантною, але не пов’язаною із жодним об’єктом системи через відсутність відповідної назви, альтернативного написання або синоніма в довіднику. Крім того, одна й та сама морська акваторія може згадуватися в новинах різними мовами, у скороченій формі або через назву найближчого міста чи порту, що також знижує ефективність простого пошуку ключових слів.

Наслідком зазначеного обмеження стало те, що протягом спостережуваного періоду не було сформовано жодного попередження типу news_correlation. Водночас це не означає, що відповідна функціональна гілка системи є непрацездатною. Її програмний код покритий автоматизованим тестуванням і перевірений модульним тестом test_classify_flags_news_correlation_when_combined, у якому на синтетично сформованих даних відтворюється ситуація одночасної наявності аномальної поведінки судна та релевантної новини.

Таким чином, відсутність попереджень цього типу в реальному експерименті пояснюється не помилкою алгоритму класифікації, а недостатньою повнотою зовнішніх і довідкових даних. Для усунення цього обмеження в подальшому доцільно розширити перелік географічних об’єктів, додати альтернативні назви, транслітерації та багатомовні варіанти написання, а також застосувати методи розпізнавання іменованих сутностей. Додаткове використання координат портів, проток і морських районів дало б змогу перейти від простого текстового зіставлення до повноцінного географічного аналізу згаданих у новині об’єктів.

ВИСНОВКИ

У кваліфікаційній роботі розроблено та практично реалізовано програмну систему аналізу просторово-часових даних морського трафіку з урахуванням факторів зовнішнього впливу. Створено працездатний прототип, який безперервно збирає AIS-телеметрію, погодні дані та новинну інформацію, виконує їх просторово-часове зіставлення в базі даних PostgreSQL/PostGIS, а також за допомогою моделей машинного навчання Isolation Forest і RandomForestRegressor виявляє нетипову поведінку суден та кількісно оцінює вплив погодних умов на швидкість їх руху.

У межах роботи виконано всі поставлені завдання. Реалізовано збір даних із трьох безкоштовних джерел: AISStream.io, Open-Meteo та GDELT. Спроєктовано структуру бази даних із використанням PostGIS, реалізовано алгоритми просторово-часового зв’язування даних, розроблено аналітичний модуль на основі методів машинного навчання, а також створено вебінтерфейс і десктопний застосунок для взаємодії із системою. Коректність роботи основних компонентів підтверджено інтеграційним та автоматизованим тестуванням.

У процесі тривалої експлуатації система накопичила інформацію про 257 суден, 17 325 AIS-позицій і сформувала 563 автоматичні попередження. Розроблений набір із 16 автоматизованих тестів проходить стабільно, що підтверджує працездатність програмної системи та правильність реалізації її основних функцій.

Принциповою відмінністю розробленої системи від наявних комерційних рішень є комплексне поєднання телеметрії суден, погодних умов і новинного контексту в єдиному аналітичному середовищі. На відміну від таких сервісів, як MarineTraffic, Spire Maritime та SeaRates, система не обмежується візуальним представленням даних, а надає кількісну оцінку впливу погоди на рух суден і формує зрозуміле пояснення причин кожного виявленого попередження природною мовою.

Під час експлуатації було виявлено окремі обмеження системи, зокрема низьку частку новин, які можуть бути точно прив’язані до географічних координат, потребу в накопиченні достатнього обсягу даних для початкового навчання моделей машинного навчання та обмеження однопроцесної архітектури. Зазначені особливості не впливають на досягнення основної мети роботи та визначають напрями подальшого вдосконалення системи.

Практична цінність роботи полягає в тому, що створена система є не лише дослідницьким прототипом, а й придатним до розгортання програмним рішенням. Вона використовує безкоштовні джерела даних та доступну програмну інфраструктуру, пройшла перевірку на реальних даних і може бути впроваджена логістичними компаніями, аналітичними центрами та портовими адміністраціями без значних початкових витрат. Крім того, систему можна використовувати в навчальному процесі як приклад інтеграції просторових баз даних, потокової обробки інформації та машинного навчання в межах єдиного програмного комплексу.

Таким чином, мету кваліфікаційної роботи досягнуто: розроблено, реалізовано та перевірено на реальних даних програмну систему, яка автоматизує аналіз морського трафіку, виявляє аномальну поведінку суден і дає змогу оцінювати вплив зовнішніх факторів на їх рух.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. Київ : ДП «УкрНДНЦ», 2016. 26 с.

2. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. Київ : ДП «УкрНДНЦ», 2016. 17 с.

3. Recommendation ITU-R M.1371-5. Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile band. Geneva : International Telecommunication Union, 2014. 129 p.

4. International Convention for the Safety of Life at Sea (SOLAS). Chapter V. Regulation 19: Carriage requirements for shipborne navigational systems and equipment. London : International Maritime Organization, 1974, as amended.

5. Review of Maritime Transport 2023. Geneva : United Nations Conference on Trade and Development, 2023. 178 p.

6. AISStream.io documentation. *AISStream.io*. URL: https://aisstream.io/documentation (date of access: 15.06.2026).

7. Marine weather API. *Open-Meteo*. URL: https://open-meteo.com/en/docs/marine-weather-api (date of access: 15.06.2026).

8. Weather forecast API. *Open-Meteo*. URL: https://open-meteo.com/en/docs (date of access: 15.06.2026).

9. GDELT 2.0 Event Database and DOC 2.0 API. *The GDELT Project*. URL: https://blog.gdeltproject.org/gdelt-doc-2-0-api-debuts/ (date of access: 15.06.2026).

10. PostGIS 3.4 Manual. *PostGIS*. URL: https://postgis.net/docs/ (date of access: 18.06.2026).

11. PostgreSQL 16 documentation. *PostgreSQL*. URL: https://www.postgresql.org/docs/ (date of access: 18.06.2026).

12. FastAPI documentation. *FastAPI*. URL: https://fastapi.tiangolo.com (date of access: 20.06.2026).

13. SQLAlchemy 2.0 documentation. *SQLAlchemy*. URL: https://docs.sqlalchemy.org/en/20/ (date of access: 20.06.2026).

14. GeoAlchemy 2 documentation. *GeoAlchemy*. URL: https://geoalchemy-2.readthedocs.io (date of access: 20.06.2026).

15. Alembic documentation. *Alembic*. URL: https://alembic.sqlalchemy.org (date of access: 20.06.2026).

16. APScheduler documentation. *APScheduler*. URL: https://apscheduler.readthedocs.io (date of access: 21.06.2026).

17. Leaflet: an open-source JavaScript library for interactive maps. *Leaflet*. URL: https://leafletjs.com (date of access: 22.06.2026).

18. Plotly JavaScript open source graphing library. *Plotly*. URL: https://plotly.com/javascript/ (date of access: 22.06.2026).

19. Pywebview documentation. *Pywebview*. URL: https://pywebview.flowrl.com (date of access: 23.06.2026).

20. Scikit-learn: machine learning in python / F. Pedregosa et al. *The journal of machine learning research*. 2011. Vol. 12, no. 85. P. 2825–2830. URL: https://jmlr.org/papers/v12/pedregosa11a.html (date of access: 11.07.2026).

21. Liu F. T., Ting K. M., Zhou Z.-H. Isolation forest. *2008 eighth IEEE international conference on data mining (ICDM)*, Pisa, Italy, 15–19 December 2008. 2008. URL: https://doi.org/10.1109/icdm.2008.17 (date of access: 11.07.2026).

22. Breiman L. *Machine learning*. 2001. Vol. 45, no. 1. P. 5–32. URL: https://doi.org/10.1023/a:1010933404324 (date of access: 11.07.2026).

23. McKinney W. Data structures for statistical computing in python. *Python in science conference*, Austin, Texas. 2010. URL: https://doi.org/10.25080/majora-92bf1922-00a (date of access: 11.07.2026).