НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ КОРАБЛЕБУДУВАННЯ
імені адмірала Макарова

Навчально-науковий інститут автоматики та електротехніки
(повне найменування інституту, факультету)

Кафедра комп'ютеризованих систем управління
(повна назва кафедри)

«Допущений до захисту»
Завідувач кафедри
_____ О.О. Черно
« 20 » 12 2024 р.

Пояснювальна записка

до кваліфікаційної роботи

на здобуття другого (магістерського) рівня вищої освіти

за спеціальністю 174– «Автоматизація, комп'ютерно-інтегровані технології та робототехніка»
ОПП - «Інтелектуальні комп'ютеризовані системи управління»

на тему: ІНТЕЛЕКТУАЛЬНА СИСТЕМА КОНТРОЛЮ ТЕХНІЧНОГО СТАНУ ЕНЕРГЕТИЧНИХ СИСТЕМ МОРСЬКИХ АВТОНОМНИХ ОБ'ЄКТІВ

Виконав студент **6** курсу, **6342м** групи
Сауляк О.В.
(прізвище та ініціали)

Керівник
Кудін О.О.
(прізвище та ініціали)

Миколаїв – 2024

Національний університет кораблебудування імені адмірала Макарова

Інститут автоматики і електротехніки
Кафедра комп'ютеризованих систем управління
Освітньо-кваліфікаційний рівень **«магістр»**

Спеціальність **174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка»**
Освітня програма **«Інтелектуальні комп'ютерні системи управління»**

ЗАТВЕРДЖУЮ
Завідувач кафедри КСУ
_____ **Черно О.О.**
« 15 » 10 2024 р.

ЗАВДАННЯ

НА ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Сауляку Олексію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи: *Інтелектуальна система контролю технічного стану енергетичних систем автономних морських об'єктів*

Керівник роботи *Кудін Олег Олексійович, к.т.н.*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від " 14 " 10 2024р. № 1075-уч

2. Строк подання роботи 16 грудня 2024 р.
3. Вихідні дані до роботи *Система контролю технічного стану повинна використовувати універсальну систему управління базами даних (СУБД), яку можна інсталювати в операційних системах Microsoft Windows та Linux*

4. Зміст розрахунково-пояснювальної записки (основні задачі дослідження)
Аналіз існуючих методів контролю технічного стану енергетичних систем морських автономних об'єктів
Вибір системи управління базами даних. Концептуальне, логічне та фізичне проектування бази даних.
Розробка Web-інтерфейсу адміністратора системи контролю технічного стану енергетичних систем морських автономних об'єктів
Охорона праці

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Аналіз існуючих методів контролю технічного стану енергетичних систем морських автономних об'єктів
Концептуальне, логічне та фізичне проектування бази даних. ER-діаграма

Розробка Web-інтерфейсу адміністратора системи контролю технічного стану.
Охорона праці

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв
Охорона праці	*Кудін О.О., доц. каф. КСУ*	05.09.2024	21.10.2024

Дата видачі завдання: «_05_»___09____2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проектування	Строк виконання	Примітка
1	*Аналіз існуючих методів контролю технічного стану енергетичних систем морських автономних об'єктів*	*10.10.2024*	*Виконано*
2	*Вибір системи управління базами даних. Концептуальне, логічне та фізичне проектування бази даних.*	*04.11.2024*	*Виконано*
3	*Розробка Web-інтерфейсу адміністратора системи контролю технічного стану.*	*22.11.2024*	*Виконано*
4	*Охорона праці*	*09.12.2024*	*Виконано*

Студент  Сауляк О.В.
(підпис) (прізвище та ініціали)

Керівник роботи  Кудін О.О.
(підпис) (прізвище та ініціали)

Анотація

Дана робота присвячена аналізу сучасних методів контролю технічного стану суднових дизельних двигунів з акцентом на можливості застосування бездротових технологій для віддаленого моніторингу. Розглянуто переваги впровадження WiFi-мосту та супутникового зв’язку Starlink, які дозволяють знизити витрати на обслуговування, а також підвищити надійність та ефективність експлуатації двигунів. Описано структуру та функціонування системи моніторингу та управління дизельними генераторами на базі PLCnext, надано практичні рекомендації щодо модернізації систем управління судновими дизелями.

Ключові слова: судновий дизель, моніторинг, діагностика, бездротові технології, система управління.

Abstract

This paper focuses on the analysis of modern methods for monitoring the technical condition of marine diesel engines, with an emphasis on wireless technology for remote monitoring. The study examines the benefits of implementing WiFi bridge and Starlink satellite connection, which reduce maintenance costs and enhance the reliability and efficiency of engine operation. The structure and functioning of the monitoring and control system for diesel generators based on PLCnext are described, and practical recommendations for the modernization of marine diesel management systems are provided.

Keywords: marine diesel, monitoring, diagnostics, wireless technology, control system.

Перелік скорочень та символів

АДГ – Аварійний дизель-генератор

PLC – Програмно-логічний контролер (Programmable Logic Controller)

СЕУ – Суднова енергетична установка

WiFi – Бездротовий інтернет-зв’язок за стандартом IEEE 802.11

Starlink – Супутниковий інтернет-зв’язок

HMI – Людино-машинний інтерфейс (Human-Machine Interface)

GSM – Глобальна система мобільного зв’язку (Global System for Mobile Communications)

GPRS – Загальна пакетна радіослужба (General Packet Radio Service)

MOP – Панель керування на базі ПК (Main Operating Panel)

MPC – Контролер керування двигуном (Main Processor Controller)

ВМТ – Верхня мертва точка (максимальне положення поршня у циліндрі)

Δp/Δφ – Жорсткість робочого процесу, швидкість підвищення тиску при згорянні палива

pmax – Максимальний тиск у циліндрі двигуна

pc – Тиск в кінці стиснення в циліндрі двигуна

pi – Середній індикаторний тиск в циліндрі

VPN – Віртуальна приватна мережа (Virtual Private Network)

PMI – Індикатор продуктивності (Performance Measurement Indicator)

ENC – Енкодер, датчик положення або швидкості обертання

CAD – Кут повороту колінчастого вала (Crank Angle Degree)

Вступ

У сучасному морському транспорті забезпечення надійності та безпеки судноплавства є критично важливими факторами, від яких залежить ефективність експлуатації суднових енергетичних установок, зокрема дизельних двигунів. Суднові дизельні двигуни, як основні елементи енергетичної системи судна, забезпечують його рух та функціонування допоміжних систем, тому їхній технічний стан має безпосередній вплив на загальну експлуатаційну ефективність і безпеку плавання. Проте експлуатація таких двигунів в умовах великих навантажень, а також суворих погодних і експлуатаційних умов створює постійну необхідність у вдосконаленні методів контролю та діагностики їх технічного стану.

З переходом від механічних до електронних систем управління паливом і клапанами, значно розширюються можливості моніторингу технічного стану двигунів. Однак це вимагає від систем діагностики нового рівня технологічної зрілості для забезпечення віддаленого моніторингу параметрів роботи в реальному часі. Одним із перспективних напрямів є впровадження бездротових технологій, таких як WiFi та супутниковий зв’язок Starlink, які дозволяють зменшити витрати на обслуговування та модернізацію інфраструктури суднових енергетичних установок, забезпечуючи при цьому надійний зв’язок і передачу даних на великі відстані.

Метою цієї роботи є комплексний аналіз сучасних методів контролю технічного стану суднових дизельних двигунів, а також оцінка можливостей і переваг впровадження бездротових технологій для дистанційного моніторингу. У межах дослідження описані основні методи діагностики, що базуються на віброакустичному і амплітудному аналізах, а також можливості покращення систем моніторингу на основі програмно-логічного контролера PLCnext. Особлива увага приділена практичним аспектам модернізації систем

управління, що дозволяє досягти кращих показників надійності, знизити експлуатаційні витрати та підвищити безпеку судноплавства.

Розділ 1. Огляд існуючих методів контролю технічного стану енергетичних систем морських автономних об'єктів.

1.1 Методи контролю технічного стану двигунів засобів водного транспорту

Надійна та безпечна робота двигунів (як ходових, так і систем енергозабезпечення) на водному транспорті забезпечується шляхом впровадження систем контролю та діагностики їх технічного стану. До складу цих систем входять різні методи і засоби контролю та діагностування. Для ефективного технічного діагностування необхідно здійснювати регулярний моніторинг технічного стану двигунів, виявляти можливі дефекти та несправності, а також оцінювати ступінь їх небезпеки й залишковий ресурс обладнання [1, 2].

Мета дослідження полягає у визначенні методів контролю технічного стану двигунів на водному транспорті.

Одним з важливих етапів під час проведення технічного обслуговування та ремонту є моніторинг і оцінка реального стану обладнання для визначення необхідних обсягів і термінів проведення регламентних робіт. Точність і ефективність діагностики безпосередньо впливають на якість і вартість технічного обслуговування, а отже, і на економічні показники [1–3].

До основних вимог сучасних методів діагностики технічного стану двигунів водного транспорту належать:

− високий рівень точності та достовірності виявлення несправностей і пошкоджень;

− можливість вчасного виявлення основних електричних і механічних дефектів;

− здатність до дистанційного вимірювання необхідних параметрів;

− низька трудомісткість та простота технічного обслуговування;

− можливість оперативного аналізу результатів вимірювань завдяки використанню інформаційно-вимірювальних систем і комплексів.

Нижче наведено огляд основних методів контролю та діагностики технічного стану двигунів водного транспорту.

1.2 Віброакустичні методи діагностики технічного стану головних двигунів водного транспорту

Головні двигуни засобів водного транспорту є складними технічними системами, що складаються з великої кількості компонентів, вузлів і механізмів, які постійно взаємодіють між собою [2]. Враховуючи широкий діапазон частот коливань у таких установках, зміни у їх технічному стані швидко відображаються у віброакустичних сигналах, що є особливо важливим у випадку аварійних ситуацій [3]. Агрегати, від безперебійної роботи яких залежить безпека людей, потребують особливої уваги, і тому важливо виявляти дефекти на ранній стадії, щоб уникнути катастрофічних наслідків.

Сутність віброакустичної діагностики полягає у створенні алгоритмів, які дозволяють оцінювати стан технічних систем без потреби у їх розбиранні. Це відбувається шляхом аналізу віброакустичних процесів, що супроводжують роботу об’єкта [3]. Основними джерелами вібрації двигунів водного транспорту є: незбалансовані обертові маси, коливання, викликані зубчастими передачами, підшипникові вузли, власні коливання елементів та агрегатів, а також аеродинамічні коливання і коливання в газо-повітряному тракті [3].

Основну увагу приділяють низькочастотним (0,1–400 Гц) та середньочастотним (400–2000 Гц) коливанням, тоді як високочастотні (> 2000

Гц) коливання мають незначний енергетичний вплив і тому часто не враховуються. Вимушені та власні коливання несуть різний обсяг інформації: вимушені коливання інформують про якість збірки, ремонту і загальні зміни у технічному стані, тоді як власні коливання дозволяють виявити дефекти на ранніх стадіях [3, 4].

Вібраційний метод діагностики полягає у вимірюванні і аналізі вібраційних сигналів, таких як їх форма, амплітуда або спектральний склад. Порівнюючи результати з попередніми вимірюваннями, можна зробити висновки щодо подальшої експлуатації або необхідності ремонту [4, 5]. До недоліків цього методу належать обмежені можливості дистанційного контролю та складність проведення вимірювань параметрів вібрації.

Метод моделювання технічного стану передбачає створення імітаційної моделі двигуна з підключеними датчиками для вимірювання параметрів. Результати вимірювань порівнюються з даними, отриманими в процесі моделювання, що дозволяє виявляти несправності. Однак цей метод має недоліки, зокрема відсутність можливості дистанційного контролю, низьку точність та складність у розробці моделей, які б точно відображали роботу реального двигуна [6].

Метод спектрального аналізу передбачає контроль частотних характеристик двигуна протягом певного часу. Отриманий сигнал досліджується на характерних частотах, що дозволяє виявляти дефекти на ранніх стадіях та здійснювати дистанційний моніторинг [6, 7]. Однак цей метод має свої обмеження: складність оцінки результатів через подвійне врахування частот і неможливість збільшення кількості гармонік для точнішого аналізу.

Найбільш перспективним методом є спектральний аналіз моторного мастила, який дозволяє виявляти домішки і визначати експлуатаційні

характеристики масла, що свідчать про наявність дефектів у двигуні. Такий метод може своєчасно виявляти несправності, підвищувати надійність двигуна і скорочувати час простою [7, 8]. За допомогою цього аналізу можна визначати фізико-хімічні характеристики мастила, виявляти присутність металів і оцінювати в'язкість, що дає змогу прогнозувати стан поршнів, циліндрів та підшипників.

Проте спектральний аналіз моторного мастила, хоча й є ефективним, залишається дорогим методом через складність у технічній реалізації, що обмежує його широке використання.

1.3 Амплітудні методи контролю технічного стану двигунів системи енергозабезпечення водного транспорту

Для діагностики несправностей двигунів, що забезпечують енергозабезпечення на водному транспорті, використовуються характерні частоти струму або напруги. Ступінь та характер несправностей визначаються шляхом порівняння амплітудних значень на певних характерних частотах (Ai) з амплітудою при нульовій частоті (A0). Діагностика основних несправностей електродвигунів базується на таких характерних частотах:

- міжвиткові замикання в обмотках статора та проблеми з ротором виявляються на частоті мережі (50 Гц або 400 Гц);
- невідповідність співвісності валів електродвигунів та механізмів, що з ними пов'язані, діагностується на частотах, кратних частоті обертання двигуна;
- несправності ремінцевої передачі визначаються на частотах, кратних частоті биття ременя;

- проблеми з підшипниками ідентифікуються за частотами, кратними частоті обертання ротора;
- пошкодження механічних пристроїв, таких як насоси, вентилятори або компресори, виявляються на так званих лопаткових частотах.

Рішення про справний технічний стан електродвигуна приймається на основі порівняння амплітудних значень Ai на характерних частотах з амплітудою при нульовій частоті A0. Якщо амплітуди на характерних частотах є нижчими за задану величину різниці з A0, це вказує на справний стан двигуна та пов’язаних з ним механізмів. Якщо ж різниця перевищує допустимий рівень, робиться висновок про наявність несправності, що відповідає даній частоті.

Спектральний аналіз сигналу та порівняння амплітудних характеристик Ai зазвичай виконуються у частотному діапазоні від -100 дБ до 0 дБ. Характерні ознаки несправностей, такі як раптові стрибки амплітуди на певних частотах, допомагають визначити вид несправності. Для зниження ефекту розтікання спектра під час спектрального аналізу використовуються функції вікна, що забезпечують кращу точність при виконанні швидкого перетворення Фур’є [8, 9].

1.4 Засоби контролю технічного стану двигунів водного транспорту

Прилади для контролю та діагностики, що використовуються на морських і річкових суднах, зазвичай відповідають вимогам класифікаційних документів та стандартів виробників енергетичного обладнання на момент введення судна в експлуатацію. Як правило, ці системи реалізовані на основі відповідних технологій [9]. Проте, контрольно-вимірювальне обладнання, включаючи комп’ютерні інформаційно-вимірювальні компоненти, швидко

застаріває через довгий термін служби суден. Сучасні інформаційні технології дозволяють значно покращити якість контролю обладнання, підвищити надійність і створити кращі умови для роботи команди судна шляхом модернізації інформаційних систем судна. Така модернізація обходиться значно дешевше порівняно з повною заміною енергетичного обладнання [10].

Однією з ключових умов технічної безпеки під час плавання є постійний моніторинг параметрів головних і допоміжних дизельних двигунів під час експлуатації. Оперативна інформація про параметри робочих процесів двигунів дозволяє екіпажу підтримувати двигуни в належному технічному стані та запобігати аварійним ситуаціям. На більшості річкових суден цей моніторинг здебільшого зводиться до періодичної перевірки тиску та температури за допомогою максиметра, який визначає максимальний тиск газів у циліндрах (pmax) та тиск у кінці стиснення (pc) при відключеній подачі палива. Окрім pmax, pc, температур води і масла, існує ще ряд інших параметрів, моніторинг яких дозволяє здійснювати точніші налаштування дизельного двигуна та контролювати його робочі процеси. Наприклад, контроль середнього індикаторного тиску (pi) допомагає виявити перевантаження окремих циліндрів і рівномірно розподілити потужність між ними.

Контроль максимальної швидкості підвищення тиску при згорянні палива (жорсткість $\Delta p/\Delta \varphi$ робочого процесу) дозволяє зменшити ударні навантаження на підшипники циліндрів та виявити проблеми у роботі паливної апаратури (ПА). Крім того, за допомогою аналізу геометричних і реальних фаз подачі палива проводиться комплексна оцінка технічного стану ПА. Контроль фаз газорозподілу під час роботи двигуна дає змогу оперативно

оцінювати стан газорозподільного механізму та підтримувати відповідні кути відкриття й закриття клапанів [11].

Окрім цих параметрів, існує ще багато інших, моніторинг яких допомагає екіпажу підтримувати нормальний технічний стан дизеля під час його експлуатації [12].

До цього часу більшість систем моніторингу дизельних двигунів суден розроблені як єдиний програмно-апаратний комплекс, який записує параметри та частково розраховує робочий процес у реальному часі. Найбільш відомими системами є NK-5, NK-100, NK-200 від компанії Autronica A/S, а також системи від Terasaki Electric Co., Ltd, Kongsberg, JRCS, Hyundai, Samsung, Honeywell і Sulzer [11, 12]. Ці системи мають дві основні функції: збір даних у реальному часі та частковий розрахунок робочих процесів, що дозволяє виробникам створювати комплексні рішення для моніторингу двигунів та надавати технічному персоналу всі необхідні дані для забезпечення якісної експлуатації двигунів. Проте такі системи мають кілька недоліків:

- Недостатня повнота та точність моделювання робочого процесу.
- Обмежена кількість вимірюваних параметрів.
- Неможливість виявлення прогнозованих змін у технічному стані двигуна.
- Складність з’єднання вимірювальної і розрахункової частин систем, що зумовлює необхідність довгих кабельних ліній для передачі сигналів, а також використання додаткових підсилювачів та перетворювачів, що знижує надійність системи.
- Висока вартість таких систем, що обумовлена не тільки ціною датчиків і первинних перетворювачів, але й всього допоміжного

обладнання, включаючи комп’ютерні комплекси та програмне забезпечення. Більше того, комп’ютери задіяні лише для моніторингу двигуна [12].

На рисунках 1.1–1.7 представлені частотні характеристики для справного двигуна і для двигуна з різними типами несправностей. На цих графіках амплітуди сигналів показані на вертикальній осі, а частоти — на горизонтальній. Позначено також характерні частоти, які свідчать про наявність несправностей.

На рисунку 1.8 показаний приклад реальної напруги живлення електродвигуна водного транспортного засобу, де на горизонтальній осі відображено час.

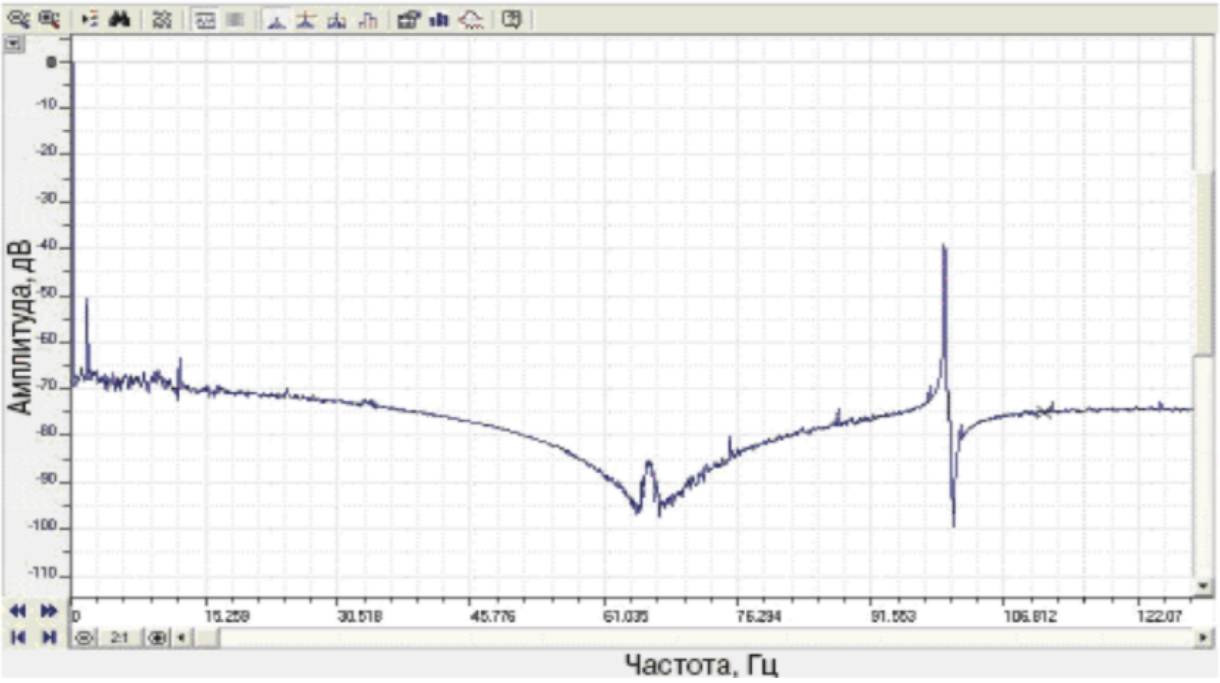


Рисунок 1.1 - Частотна характеристика модулю вектору струму справного електродвигуна (навантаження - насос)

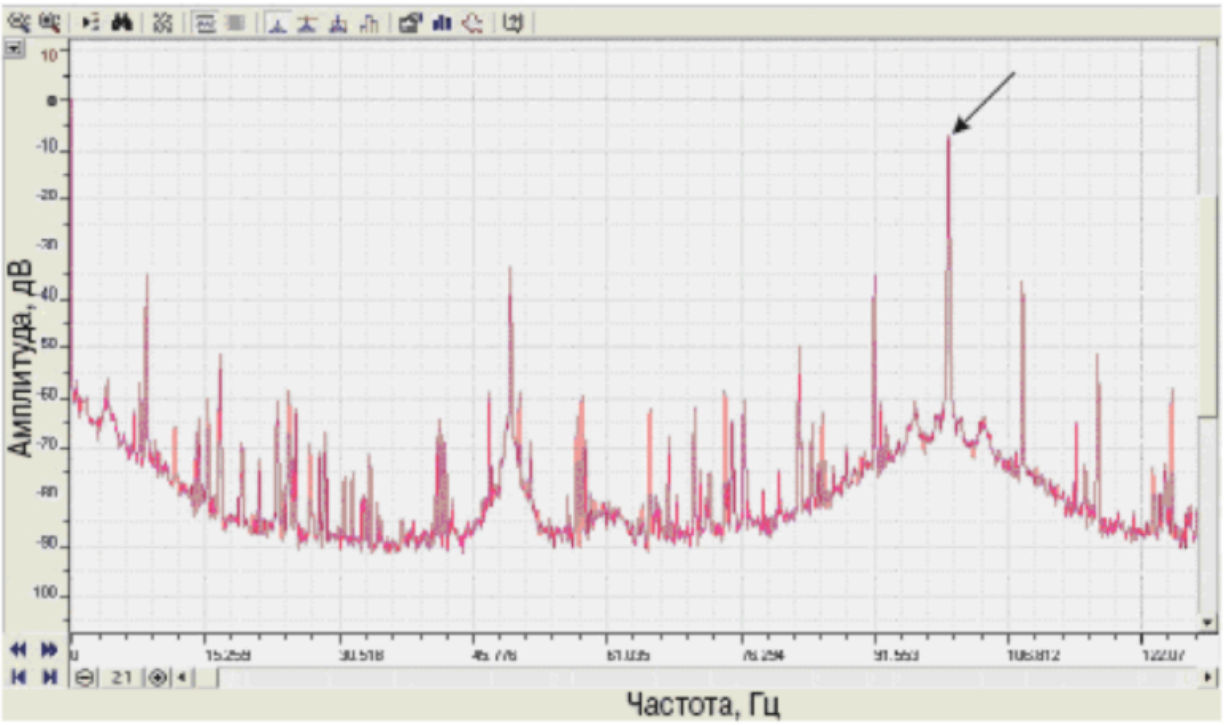


Рисунок 1.2 - Частотна характеристика модулю вектору струму електродвигуна при короткому замкненні обмотки статора

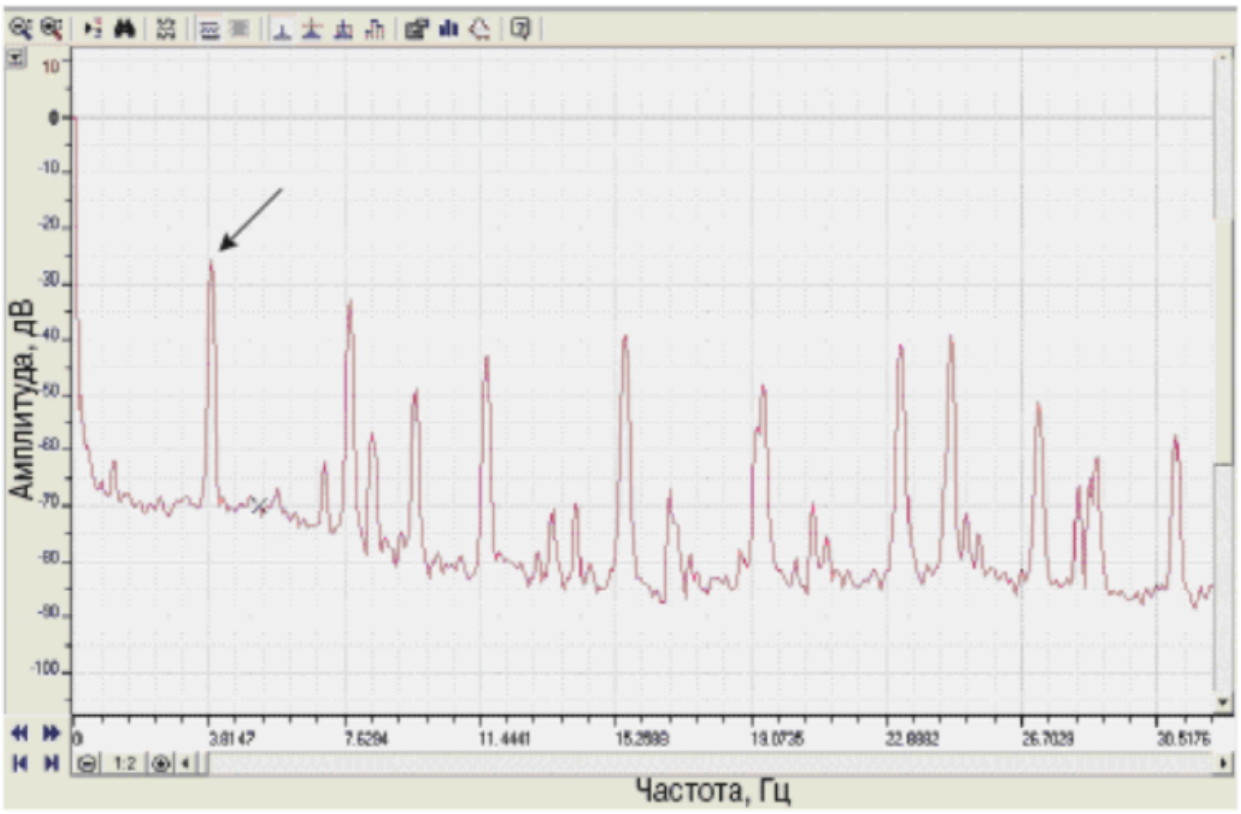


Рисунок 1.3 - Частотна характеристика модулю вектору струму електродвигуна при несправності ротора

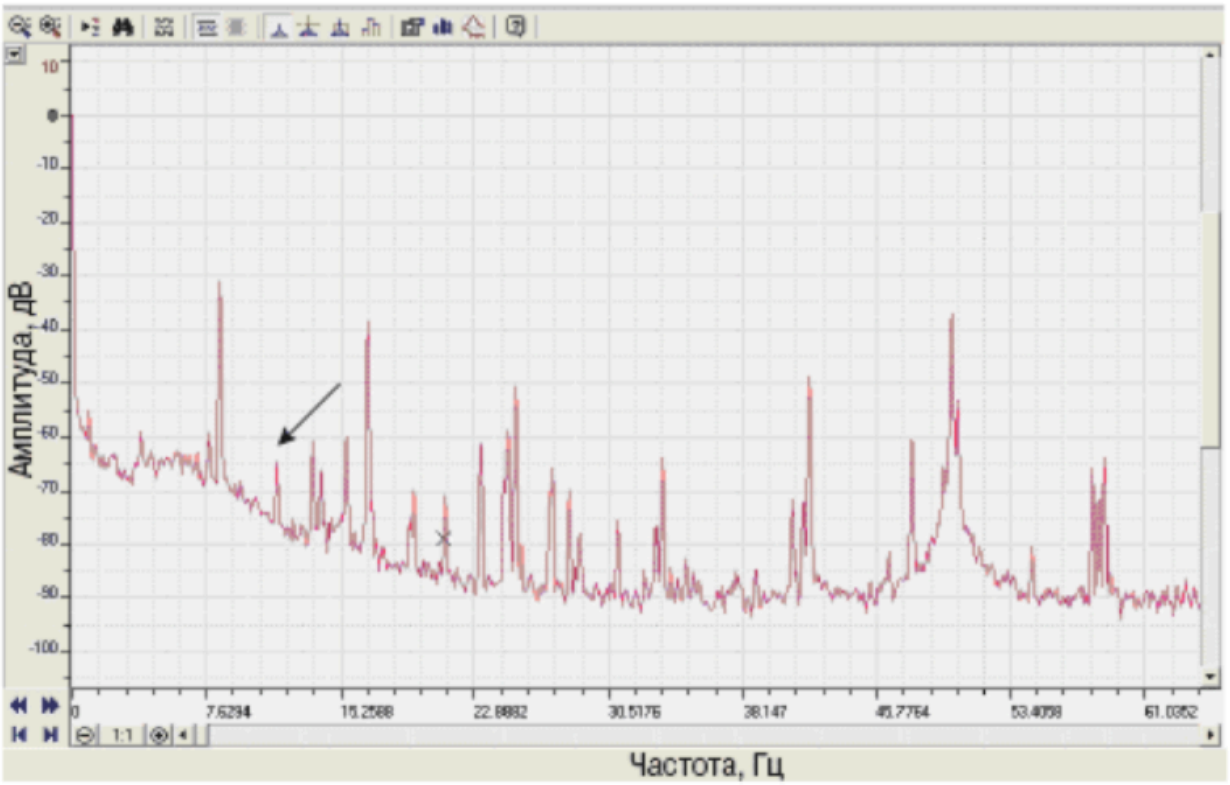


Рисунок 1.4 - Частотна характеристика модулю вектору струму електродвигуна при терті ротора та статора

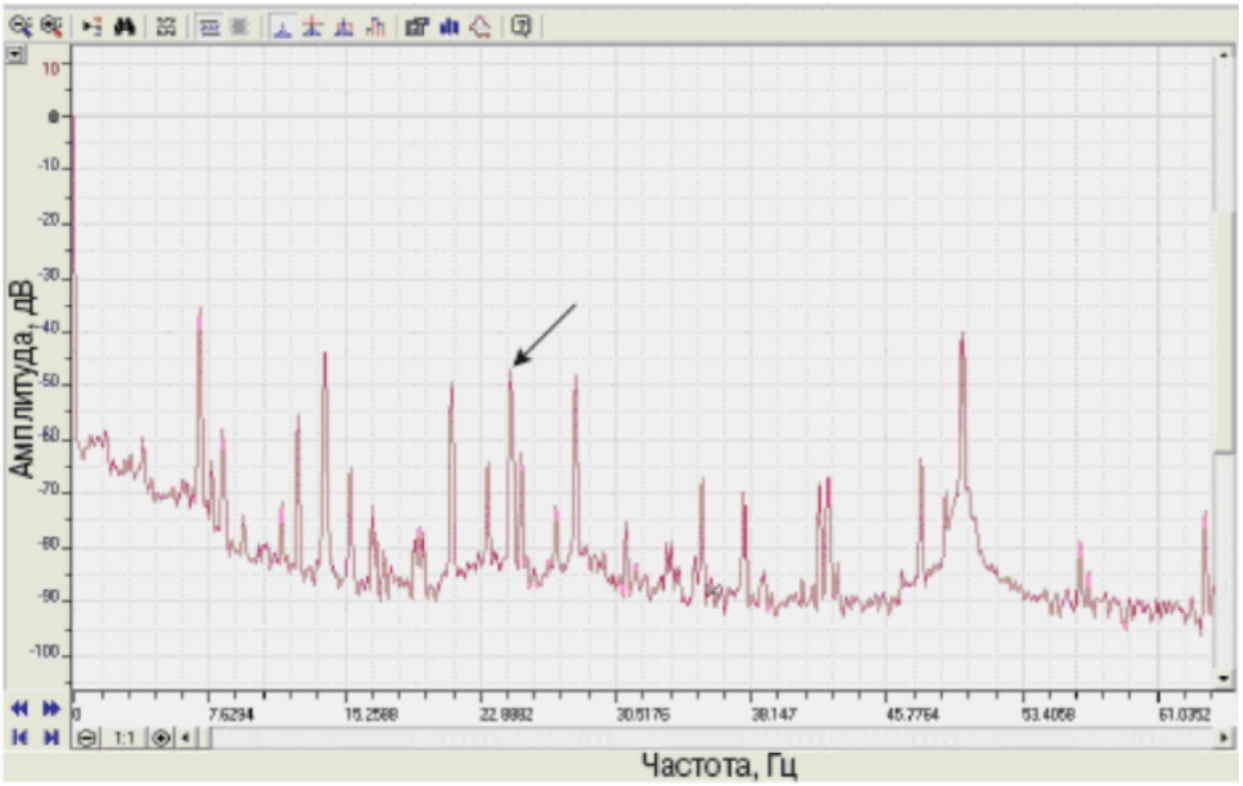


Рисунок 1.5 - Частотна характеристика модулю вектору струму електродвигуна при неузгоджені із навантаженням

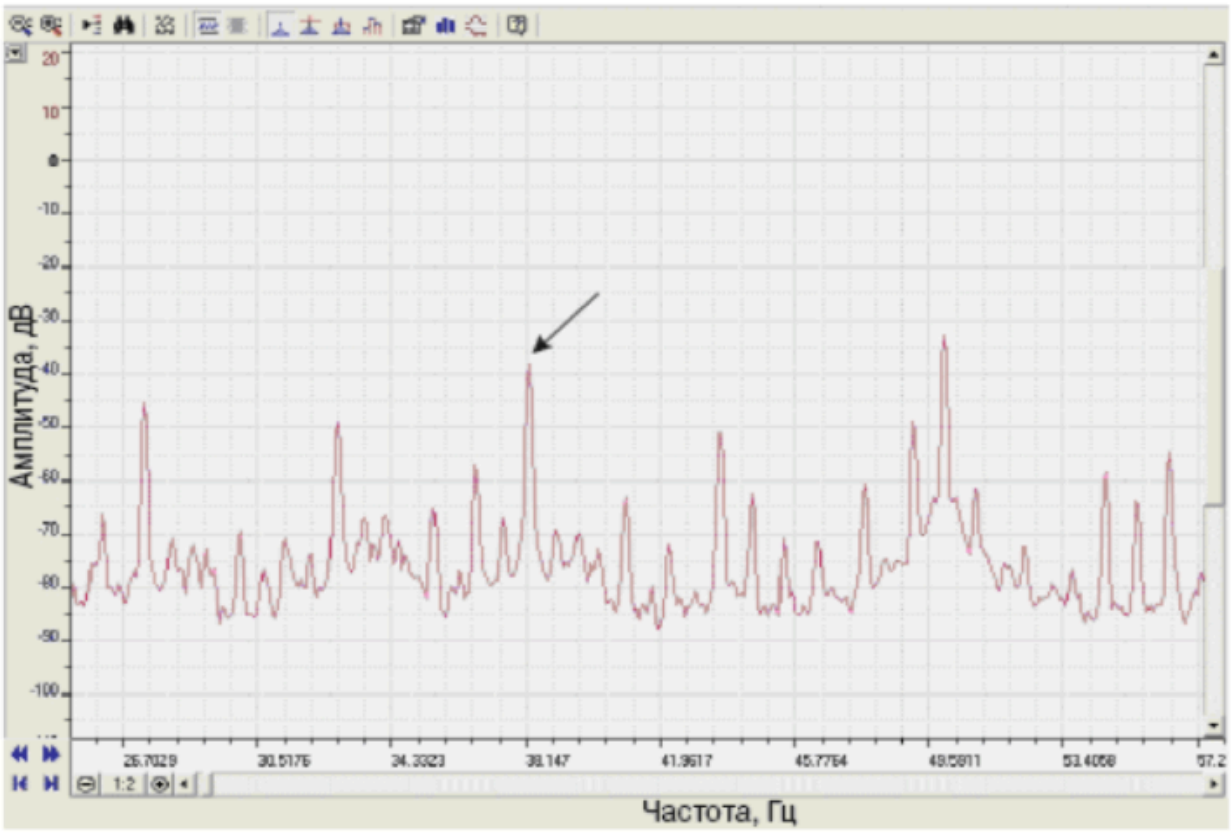


Рисунок 1.6 - Частотна характеристика модулю вектору струму електродвигуна при несправності підшипника

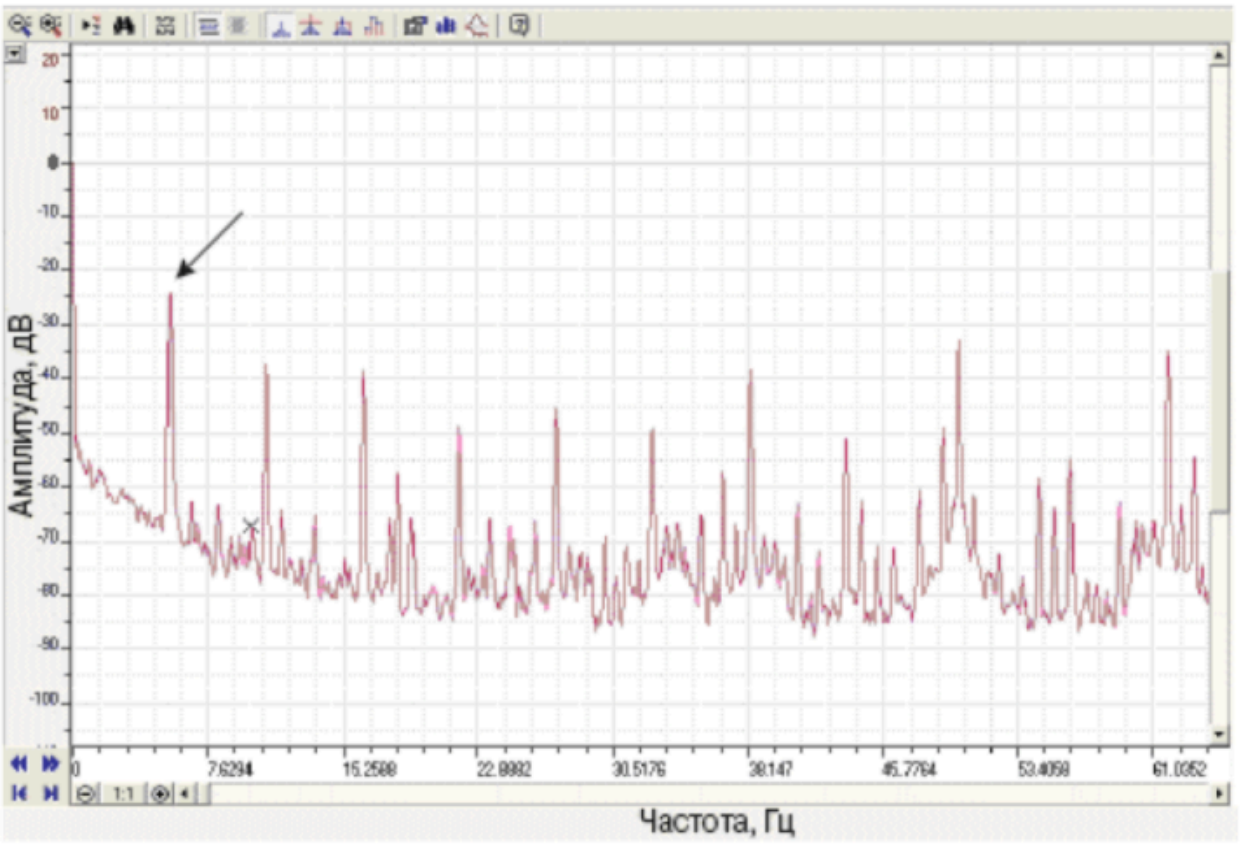


Рисунок 1.7 - Частотна характеристика модулю вектору струму електродвигуна при несправності передатного механізму

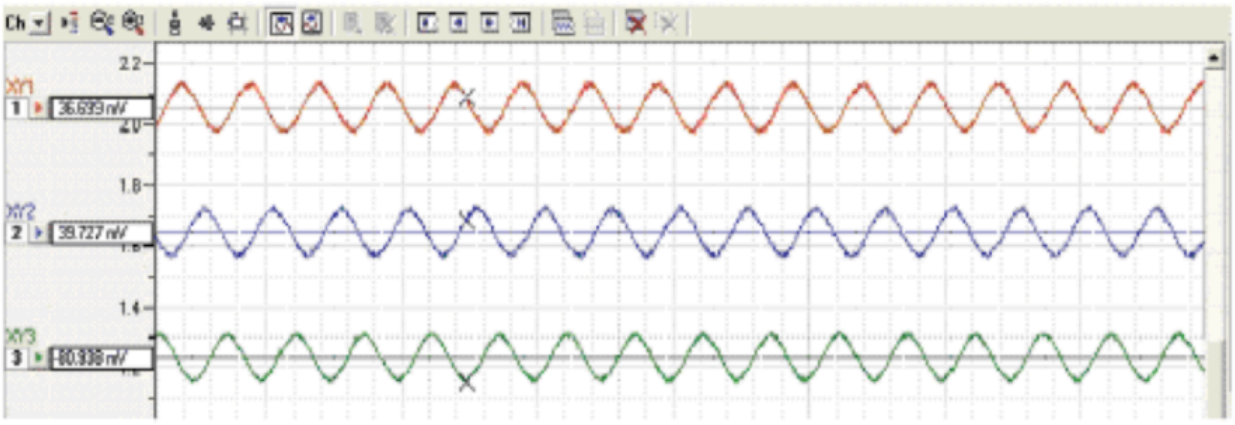


Рисунок 1.8 - Приклад напруги електродвигуна засобу водного транспорту

1.5 Системи діагностики середньо- і малооборотних двигунів

Сучасні системи діагностики середньо- та малооборотних двигунів активно використовуються на різних суднах. Наприклад, система CEDC компанії «Зульцер» (Швейцарія) призначена для діагностики циліндро-поршневої групи (ЦПГ), паливної апаратури (ПА), турбокомпресора та охолоджувача наддувочного повітря (ОНП). Ця система встановлена на дизелях типу 6RND-90 теплохода «Віллі де Страсбург» (Франція). Вона використовує міні-ЕОМ для аналізу поточних параметрів дизеля і його технічного стану. Під час роботи система аналізує зміни параметрів у часі та визначає необхідні терміни обслуговування. При досягненні граничних значень параметрів видається попередження про необхідність заміни або ремонту відповідних компонентів, що дозволяє своєчасно проводити необхідні роботи для підтримки оптимальної роботи двигуна. Основні компоненти цієї системи включають датчики, центральний оброблювальний пристрій і засоби зв’язку для взаємодії оператора з системою [13, 15].

Термічне навантаження циліндрів вимірюється спеціальними термодатчиками, що встановлені на різних частинах циліндра. Ці датчики дозволяють виявляти відхилення в роботі поршневих кілець, включаючи їх знос або втрату рухливості, що може призвести до серйозних поломок. Інформація збирається за допомогою індуктивних датчиків, що встановлюються на поверхні втулки циліндра. На основі зібраних даних обчислювальний пристрій визначає загальний технічний стан двигуна і залишковий ресурс деталей, а також видає рекомендації щодо часу проведення профілактичного обслуговування. Система також може взаємодіяти з системами автоматичного управління двигуном, регулювати циркуляцію мастила та автоматично зупиняти двигун у разі небезпечного відхилення параметрів від норми [15].

Інша система, DETS фірми «Норконтрол» (Норвегія), забезпечує діагностику процесу впорскування палива і згоряння в дизелях. Вона використовує п’єзоелектричні датчики для вимірювання тиску впорскування палива та тиску в циліндрах, а також магнітні датчики для визначення положення клапанів і частоти обертання. Додаткові датчики реєструють тиск повітря та інші ключові параметри, що відображаються у вигляді графіків та діаграм для візуального контролю [15, 16].

Система PED від компанії «Пилстик» (Франція) застосовується для діагностики середньооборотних дизелів. Вона відстежує стан підшипників колінчастого вала, верхнього поршневого кільця і турбокомпресора. Система використовує датчики для вимірювання температури та тиску, що допомагає виявити знос підшипників або інші несправності двигуна [16].

Система «Віброметр» (Швейцарія) спеціалізується на діагностиці поршневих кілець, систем упорскування палива і турбокомпресорів. Вона використовує п’єзоелектричні датчики для збору акустичних сигналів, які

потім аналізуються для виявлення дефектів у вузлах двигуна [17, 18]. Інша система, МЕКОМ (Норвегія), призначена для діагностики дизелів, турбін і котлів. Вона реєструє параметри, такі як вібрації механізмів, температура підшипників і тиск у газо-повітряному тракті [13].

Сучасні діагностичні системи дозволяють контролювати зміни тиску у циліндрах дизелів в залежності від часу або кута повороту колінчастого вала. Такі системи можуть відображати діаграми та зберігати інформацію для подальшого аналізу, що дає можливість оптимізувати роботу двигунів і вчасно проводити обслуговування [17, 18].

З аналізу існуючих систем діагностики можна зробити висновок, що на сьогодні виробники концентруються на діагностиці лише власних моделей двигунів, і універсальні системи діагностики для всіх типів двигунів не розробляються. Більшість сучасних систем застосовуються на великих морських суднах, тоді як на річкових судах їх використання поки що обмежене [19].

1.6 Вдосконалення систем віддаленого моніторингу параметрів суднових дизелів з електронним управлінням

З початком впровадження електронних систем управління паливом та клапанами у суднові двигуни, провідні виробники постійно вдосконалюють ці системи. Це відбувається не тільки шляхом покращення електронного обладнання, але й завдяки розвитку програмного забезпечення, що підвищує надійність їх функціонування.

Окрім цього, з появою технологій, таких як Starlink, з’явилася можливість отримувати доступ до систем моніторингу та управління судновими двигунами в режимі реального часу не лише для суднової команди, але й для офісного персоналу. Універсальні програмно-логічні

контролери з розширеними функціями дозволяють проводити швидку модернізацію системи через заміну модуля пам’яті або контролера в цілому.

Судна, побудовані на початку 2000-х років, часто оснащені системами моніторингу без можливості подвоєного резервування модулів управління та з використанням застарілих операційних систем, підтримка яких уже припинена. Через це актуальним є вдосконалення систем моніторингу суднових дизелів для підвищення їх надійності та розширення функціональних можливостей, що має як наукове, так і практичне значення.

Аналіз літератури та електронних джерел, присвячених системам моніторингу параметрів суднового дизеля MAN-B&W типу ME, показав, що з початку 2000-х років електронні системи управління паливом і клапанами стали невід’ємною частиною суднових дизелів, що дозволило знизити експлуатаційні витрати та забезпечити більшу гнучкість у режимах роботи двигуна.

Основними цілями вдосконалення таких систем є:

1. Підвищення надійності двигуна через моніторинг параметрів у режимі реального часу, що забезпечує рівномірний розподіл навантаження між циліндрами.
2. Підвищення можливостей контролю викидів.
3. Зниження витрат палива та мастила.

Електронна система управління двигуном складається з контролерів (MPC), які пов’язані між собою двома мережевими лініями. Ці контролери мають однакову конструкцію і різняться лише програмним забезпеченням. До мережі підключені дві панелі керування (MOP) на базі ПК під управлінням WindowsXP. Програмне забезпечення для всіх контролерів зберігається на жорсткому диску MOP, що дозволяє легко проводити модернізацію за

допомогою новітніх технологій, які розширюють функціональні можливості системи.

Аналіз процесу вдосконалення системи моніторингу показав, що його можна розділити на кілька етапів: розробка первинних перетворювачів для вимірювання параметрів, вибір способів передачі інформації до системи диспетчеризації та інтеграція диспетчеризації із судновими системами менеджменту.

На рис. 1.9 представлена функціональна схема моніторингу параметрів суднового дизеля MAN-B&W типу ME. Вона включає первинні перетворювачі, систему портативного збору інформації, блок калібрування, систему PMI-online (Performance Measurement Indicator), тахогенератор, підсилювач сигналів та VPN роутер [2, 3].

Система PMI-online, яка постійно відстежує робочі параметри двигуна, дозволяє швидко налаштовувати його роботу та забезпечує ефективний контроль.

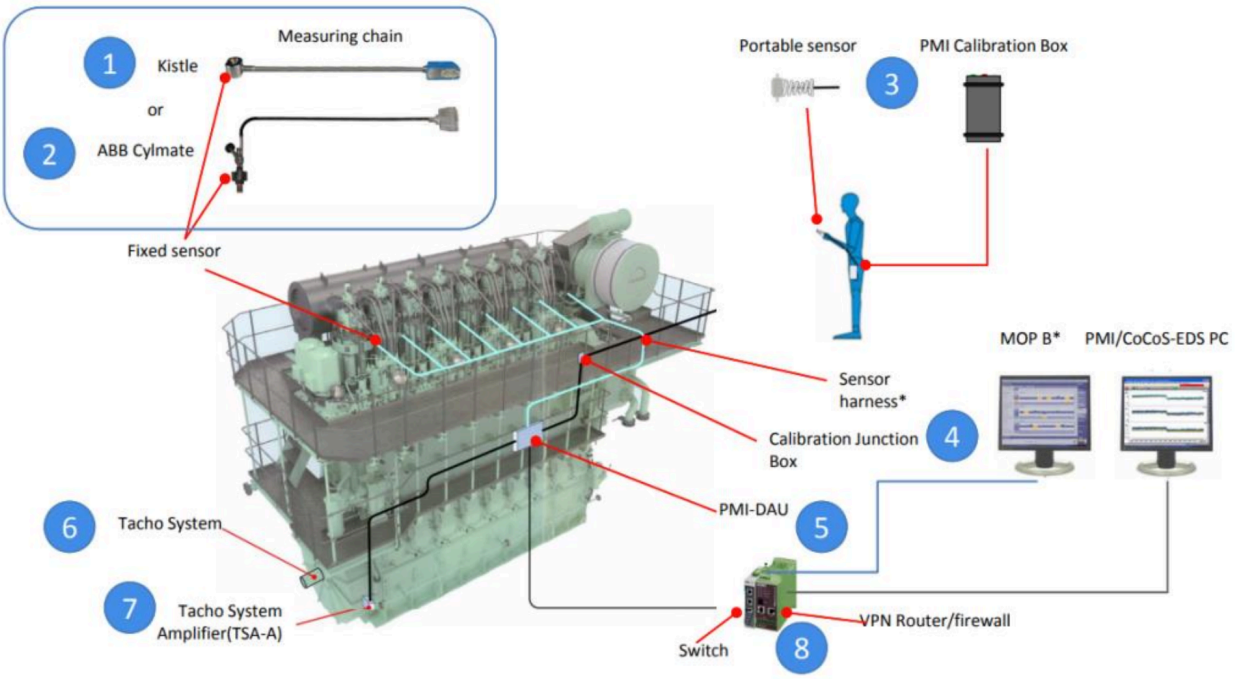


Рис. 1.9 Функціональна схема моніторингу параметрів суднового дизеля

MAN-B&W типу ME [2, 3]

Для моніторингу стану двигуна ми можемо використовувати наступні параметри:

1. Основні значення тиску – Відображає основні значення тиску з кожного циліндра, а також вихідну потужність, наведену в таблиці.
2. Відхилення тиску – Показує відхилення тиску для кожного циліндра від “середнього” значення.
3. Тиск по куту ВМТ – Наносить значення тиску в циліндрах на графік відносно кута верхньої мертвої точки (ВМТ) кожного циліндра.
4. Тиск по куту колінчастого вала – Відображає значення тиску в циліндрі в залежності від кута колінчастого вала.
5. Тиск по об’єму циліндра – Відображає значення тиску в циліндрі на графіку відносного об’єму циліндра.
6. Графіки кривих тиску – Будує ряди кривих тиску в циліндрах через 10 обертів колінчастого вала. Кожна крива представляє фактичний сигнал, виміряний датчиком.

1.7 Запровадження бездротових технологій та їх переваги

Сучасний розвиток технологій дистанційного моніторингу та управління сприяє широкому впровадженню бездротових методів, які мають низку значних переваг над традиційними дротовими системами. Однією з основних переваг є можливість уникнення необхідності в дротових з’єднаннях, що значно спрощує інфраструктуру та процес монтажу. Це також дозволяє зменшити витрати на обладнання та його встановлення.

Бездротові системи управління надають більше можливостей для гнучкого розташування та переміщення обладнання, що є особливо важливим під час проектування систем дистанційного управління. Завдяки швидкій інсталяції бездротових рішень такі системи стали дуже популярними, оскільки вони дозволяють суттєво скоротити час впровадження.

Крім того, бездротові системи значно легше розширювати, модернізувати та адаптувати до нових вимог і умов. Це надає змогу користувачам ефективно налаштовувати систему під свої потреби без складних інфраструктурних змін.

Запровадження бездротових технологій у системах дистанційного управління має й інші важливі переваги. Зокрема, це сприяє зниженню вартості обслуговування та покращенню надійності з’єднань. Крім того, бездротові системи забезпечують кращу доступність та знижують витрати на модернізацію енергетичної інфраструктури. Завдяки таким перевагам, ці технології дозволяють запроваджувати інноваційні рішення та зменшувати негативний вплив на навколишнє середовище.

Актуальність застосування бездротових технологій для передачі інформаційних потоків у системах моніторингу й управління енергетичними об’єктами підтверджена у ряді досліджень [3, 5, 6]. З 2010 року особливо поширеними стали програмно-логічні контролери з вбудованими Web-серверами, що відкриває нові можливості для впровадження гнучких систем дистанційного управління та моніторингу параметрів технологічних процесів.

На рис. 1.10 наведена концепція трирівневої ієрархічної системи обміну інформацією та моніторингу параметрів від компанії Phoenix_Contact. Ця ієрархічна система дозволяє не тільки впроваджувати бездротові технології

типу WiFi і Bluetooth, але й забезпечувати дворівневе резервування, що значно підвищує надійність системи моніторингу та управління.

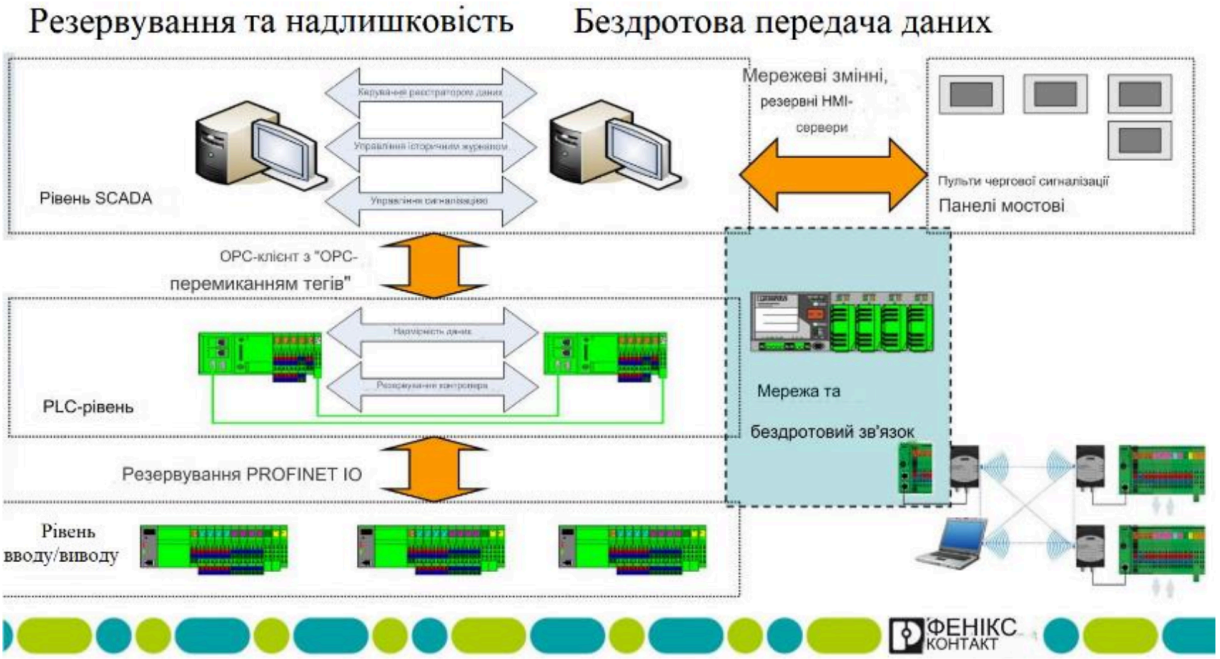


Рис. 1.10. Концепція суднової ієрархічної системи управління

У рамках діагностики аварійного дизель-генератора (АДГ) компанії Kohler, модель 50EOZD з потужністю 50 кВт, що встановлений в навчальному машино-котельному відділенні кафедри, були визначені умови для впровадження бездротової схеми дистанційного управління. Ця система була розроблена та інтегрована з автоматичною системою управління АДГ без втручання в роботу штатної системи.

Розробка інформаційної моделі бездротової системи передачі даних для аварійного дизель-генератора була представлена у вигляді структурної схеми (рис. 1.11).

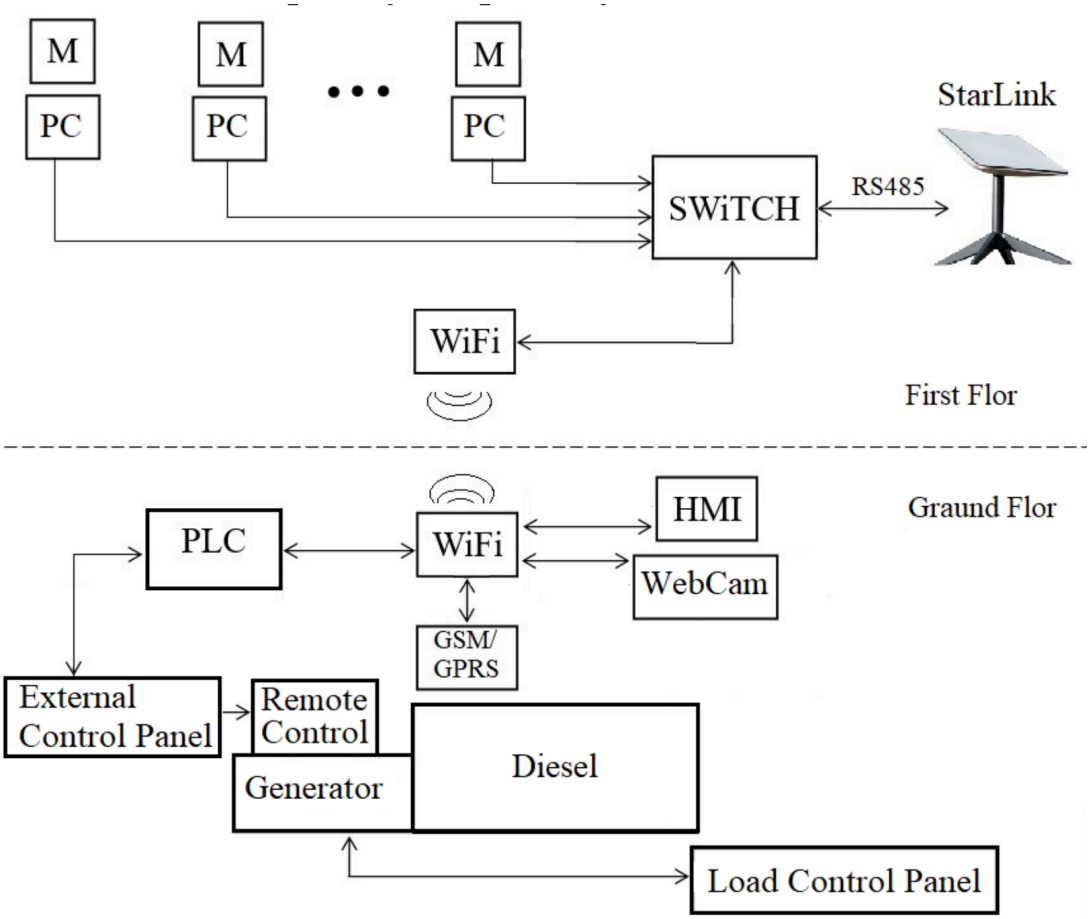


Рис. 1.11. Інформаційна модель бездротової системи передачі інформаційного потоку контролю параметрів аварійного дизель-генератора [5]

Опис схеми:

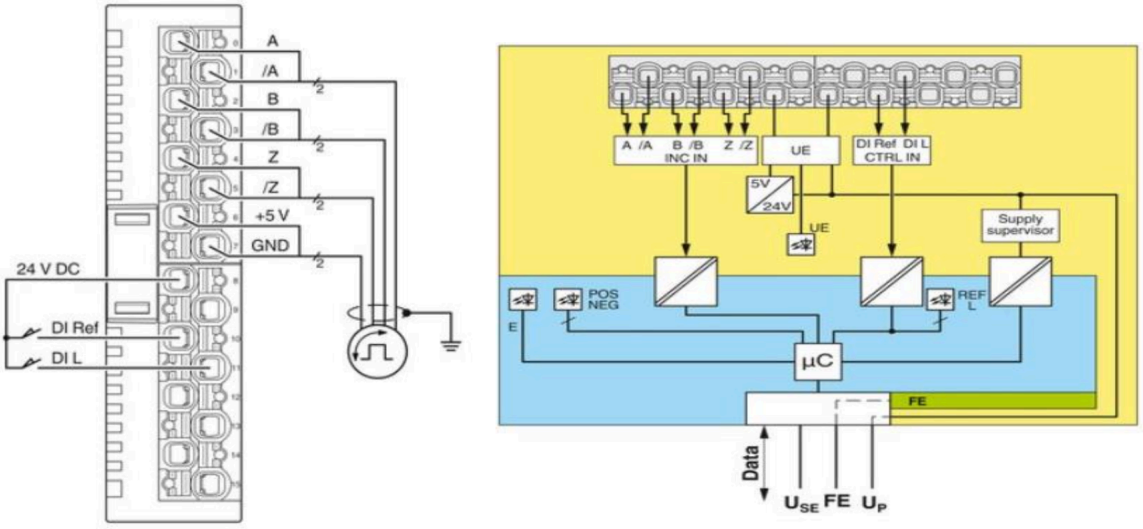
- M – Монітор (у контексті схеми – комп’ютери, які, ймовірно, отримують або передають інформацію).
- PC – Персональний комп’ютер (пристрій для управління або моніторингу процесу).
- SWITCH – Мережевий комутатор (пристрій для об’єднання комп’ютерів у локальній мережі).
- PLC – Програмований логічний контролер (пристрій для автоматизації, керування генератором).

- WiFi – Бездротовий зв’язок (технологія для передачі даних між пристроями).
- GSM/GPRS – Глобальна система мобільного зв’язку/загальна пакетна радіослужба (стандарти мобільного зв’язку для передачі даних).
- HMI – Людино-машинний інтерфейс (інтерфейс для взаємодії оператора з системою).
- WebCam – Вебкамера (пристрій для відеонагляду або моніторингу).
- StarLink – Супутниковий інтернет (технологія для передачі даних через супутники).
- Load Control Panel – Панель керування навантаженням (пристрій для керування потужністю або навантаженням генератора).
- Remote Control – Дистанційне керування (пристрій або система для віддаленого керування).
- External Control Panel – Зовнішня панель керування (пристрій для управління генератором ззовні).

Такий підхід дозволяє чітко відобразити складові елементи та їх взаємозв’язки, що спрощує розуміння функціонування системи та її структури. Для забезпечення з’єднання між приміщеннями на першому та другому поверхах була обрана технологія WiFi-мосту. Це дозволило організувати бездротовий зв’язок між системою управління АДГ та комп’ютерним класом, забезпечуючи швидкий і зручний обмін даними.

Для моніторингу кількості обертів колінчастого валу використовувався датчик на базі геркону, встановлений на шків колінчастого валу. Однак це рішення не дозволяло визначати кут повороту валу. Тому, на відміну від обладнання, запропонованого в дослідженні [6], контролер PLCnext Control

AXC F 2152 було додатково обладнано модулем для визначення положення інкрементальних енкодерів AXL SE INC1 SYM (номер виробу 1088130).



а) *б*)

Рис. 1.12. Модуль визначення положення інкрементальніх енкодерів AXL SE INC1 SYM:

а – приклад підключення; б – схема внутрішніх з’єднань

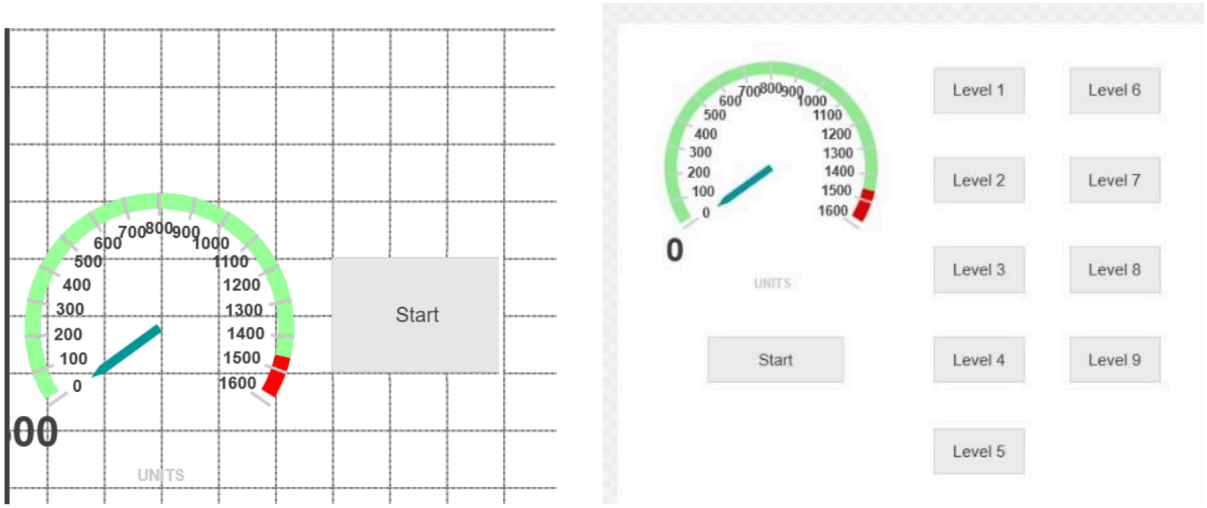
При виборі програмного забезпечення було знову використано PLCNEXT ENGINEER [4], яке надає можливість створювати, конфігурувати та налаштовувати функціональні блоки і зв’язки між ними. Це значно спрощує процес програмування та інтеграції компонентів системи, що дозволяє швидко модернізувати апаратну частину.

PLCNEXT ENGINEER також дає змогу одночасно налаштовувати як програмно-логічну частину, так і Web-сервер. Для порівняння, при програмуванні контролерів попереднього покоління потрібно було використовувати два різні програмні інструменти.

Існує можливість використання програмного середовища CoDeSys, проте його функціональність часто вимагає значних ресурсів комп’ютера, на

якому воно встановлене. Це може бути неприйнятним для суднових умов, де операційна система залишається незмінною протягом усього життєвого циклу судна.

На рис. 6а представлений фрагмент інтерфейсу управління запуском АДГ і відображення кількості обертів колінчастого валу. На відміну від інтерфейсу, що був запропонований у роботі [6], тут відсутня кнопка зупинки АДГ, оскільки при підключенні до штатного пульту дистанційного управління зупинка дизель-генератора здійснюється автоматично через 5 хвилин після зняття навантаження.



а) *б*)

Рис. 1.13. Інтерфейс проекту в програмному середовищі PLCNEXT ENGINEER:

а – з можливістю запуска та моніторингу обертів; б – з додатковою функцією керування навантаженням [5]

На рис. 1.13(б) представлено детальний фрагмент інтерфейсу, який використовується для керування режимом роботи каналу управління. Ця форма дозволяє запускати АДГ за допомогою кнопки Start. Крім того, інтерфейс відображає показники обертів колінчастого валу АДГ. Кнопки

Level 1,…, Level 9 відповідають за підключення електричних ланцюгів. Зупинка генератора виконується автоматично через 5 хвилин після зняття навантаження.

Додатково, зона розташування АДГ оснащена веб-камерою, що дозволяє здійснювати відеомоніторинг навколо аварійного дизель-генератора. Для цього використовувалося програмне забезпечення DMSS App [0], яке доступне як для мобільних пристроїв, так і для персональних комп’ютерів.

На рис. 1.14 зображено зовнішній вигляд модернізованої системи віддаленого управління аварійним дизель-генератором, що включає такі компоненти: 1 – персональний комп’ютер; 2 – монітор; 3 – роутер; 4 – дротовий пульт дистанційного управління; 5 – пульт керування навантаженням генератора; 6 – система управління та моніторингу параметрів на базі програмованого контролера AXC F 2152 (PLC Next); 7 – веб-камера.

Рис. 1.14. Зовнішній вигляд системи моніторингу та управління

Експериментальні дослідження розробленої системи бездротового доступу до управління АДГ і моніторингу його параметрів підтвердили ефективність запропонованого підходу. У зв’язку з цим пропонується впровадження бездротових технологій у систему моніторингу параметрів суднового дизеля MAN-B&W типу ME.

На рис. 1.15 показана вдосконалена система дистанційного моніторингу параметрів суднового дизеля, яка дозволяє швидко проводити модернізацію систем управління та моніторингу.

Таким чином, модернізація АДГ підтвердила технічну можливість віддаленого управління з одночасним відображенням необхідних параметрів на мобільних пристроях і персональних комп’ютерах. Запропонована схема управління дозволяє використовувати модернізовану систему до четвертого рівня автоматизації.

Застосоване апаратне забезпечення надає можливості для подальшого розширення функціоналу в залежності від майбутніх задач, які можуть виникнути.

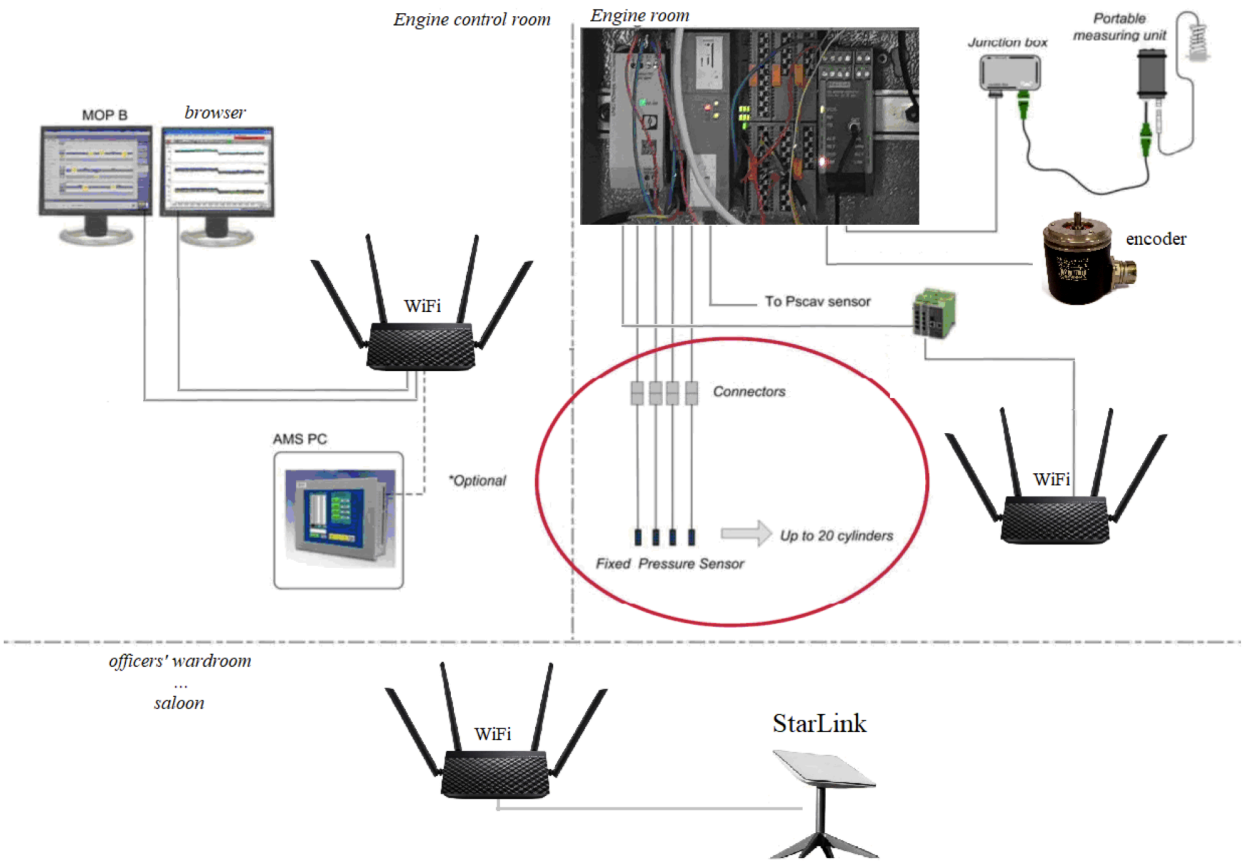


Рис. 1.15. Вдосконалена система дистанційного моніторингу параметрів суднового дизеля

Опис схеми:

- MOP B – Монітор операційної панелі B (екран для відображення параметрів роботи суднового дизеля).
- browser – Браузер (інтерфейс для віддаленого доступу до даних моніторингу).
- WiFi – Бездротовий зв’язок (забезпечує зв’язок між пристроями через локальну мережу).
- AMS PC – Персональний комп’ютер автоматичної системи моніторингу (пристрій для обробки та відображення даних).

- Engine control room – Кімната управління двигуном (місце, де оператор контролює роботу двигуна).
- Engine room – Машинне відділення (місце розташування основних компонентів двигуна).
- Junction box – З’єднувальна коробка (пристрій для підключення і розгалуження електричних з’єднань).
- Portable measuring unit – Портативний вимірювальний блок (пристрій для збору даних, який може бути переміщений).
- Encoder – Енкодер (пристрій для перетворення фізичних параметрів у цифровий сигнал).
- Fixed Pressure Sensor – Стаціонарний датчик тиску (вимірює тиск у циліндрах двигуна).
- StarLink – Супутниковий інтернет (технологія для передачі даних через супутники).

Завдяки обладнанню компанії Phoenix Contact (Німеччина) забезпечується висока надійність системи бездротового управління, а також завадостійкість. Запроваджене програмне забезпечення значно полегшує процес модернізації та вдосконалення системи моніторингу параметрів суднових дизелів.

1.8 Висновки за Розділом 1

Аналіз розглянутих та інших, не описаних у роботі, систем дозволяє визначити перелік діагностичних параметрів двигунів, необхідних для впровадження алгоритмів безрозбірної автоматичної технічної діагностики. Результати аналізу свідчать про те, що сучасні електронні системи управління дозволяють здійснювати безперервний моніторинг технічних параметрів, збір і обробку даних, отриманих від датчиків. Це дає змогу виконувати діагностику не лише двигунів, але й інших, менш складних систем суднових енергетичних установок (СЕУ) річкових суден.

Таким чином, розробка спеціалізованих діагностичних комплексів набуває актуальності, оскільки інформація від електронних систем управління двигунами та іншими об’єктами СЕУ може оброблятися центральним комп’ютером управління або комп’ютером машинного відділення для діагностики та управління всією СЕУ, а також окремими об’єктами, якщо це необхідно [14].

Аналіз методів і засобів контролю технічного стану двигунів водного транспорту показав, що спектральний аналіз сигналів вібродіагностики або складових спектру робочої рідини (наприклад, моторного мастила) є актуальним підходом. Для цього слід використовувати методи дослідження оцінок спектральної щільності потужності під час аналізу випадкових сигналів, що може застосовуватися при контролі технічного стану двигунів.

Вдосконалення системи контролю суднових дизелів було досягнуто завдяки кільком ключовим нововведенням. Перехід від традиційних механічних систем управління паливом і клапанами до електронного управління суттєво підвищив надійність та ефективність роботи двигунів. Електронні системи дозволили знизити експлуатаційні витрати, забезпечили

більшу гнучкість у режимах роботи, а також дозволили точніше контролювати параметри роботи, що стало важливим кроком у модернізації.

Інтеграція новітніх технологій, таких як Starlink, відкрила нові можливості для віддаленого моніторингу та управління судновими двигунами в реальному часі, що суттєво підвищило ефективність роботи як на борту, так і в берегових офісах. Це зробило контроль за процесами більш оперативним та надійним.

Крім того, впровадження універсальних програмно-логічних контролерів (PLC) із розширеними функціями значно спростило процес модернізації. Такі контролери дозволили модернізувати системи шляхом простої заміни модулів пам’яті або контролера, що зменшило витрати часу та ресурсів на оновлення.

Важливим етапом вдосконалення стало використання бездротових технологій, що спростило інфраструктуру та суттєво знизило витрати на обслуговування. Вони також надали гнучкість у розміщенні обладнання, прискорили процес інсталяції та полегшили адаптацію системи до змінних вимог.

Таким чином, вдосконалення системи контролю суднових дизелів стало можливим завдяки використанню сучасних електронних систем управління, бездротових технологій і універсальних програмно-логічних контролерів, що значно підвищило надійність, ефективність та гнучкість управління.

Розділ 2. Проектування бази даних системи мониторингу

У сучасному морському транспорті важливу роль відіграє безперервний моніторинг технічного стану суднових двигунів, оскільки від їхньої надійності залежить як ефективність експлуатації судна, так і безпека судноплавства. Постійний збір і аналіз даних про роботу двигунів дозволяють завчасно виявляти потенційні відхилення в їхній роботі та своєчасно реагувати на них, запобігаючи аварійним ситуаціям. Для забезпечення централізованого збору таких даних та їх подальшого аналізу, особливо у випадку, коли йдеться про флотилію з кількох суден, доцільним є створення бази даних, яка здатна обробляти великі обсяги інформації в реальному часі.

Цей розділ присвячений проєкту бази даних, яка буде використовуватися для зберігання даних про робочі параметри двигунів суден. Проєкт бази даних враховує специфічні вимоги до моніторингу технічних параметрів, зокрема потребу у швидкому доступі до даних, забезпеченні надійності зберігання та можливості інтеграції з іншими системами управління та аналізу даних. У рамках цього розділу будуть розглянуті концептуальна, логічна та фізична моделі бази даних, а також структура таблиць, зв’язки між сутностями, основні операції для роботи з даними та заходи для забезпечення їхньої безпеки.

Мета розробки бази даних полягає у створенні надійної системи зберігання та обробки даних, яка дозволить ефективно відстежувати технічний стан двигунів у реальному часі, аналізувати зібрані дані та швидко реагувати на зміни в їх роботі. Це, в свою чергу, сприятиме підвищенню надійності та довговічності суднових двигунів, а також оптимізації процесів їх обслуговування і ремонту.

2.1 Аналіз вимог

Для забезпечення ефективного моніторингу технічного стану суднових дизельних двигунів база даних повинна відповідати ряду вимог. У цьому пункті розглянемо основні функціональні та нефункціональні вимоги, які допоможуть побудувати систему, здатну надійно зберігати й обробляти дані.

2.1.1 Функціональні вимоги

- Збір та зберігання даних: Система повинна забезпечувати можливість збирання даних від різних датчиків, встановлених на суднових двигунах. Дані повинні автоматично записуватися у базу у визначені інтервали часу або за подією.
- Агрегація даних для кількох двигунів: Оскільки база даних розробляється для збору даних із двигунів декількох суден, система повинна забезпечувати окремий облік даних для кожного двигуна з урахуванням його унікального ідентифікатора.
- Фільтрація та сортування даних: Для забезпечення гнучкості роботи з даними необхідно підтримувати можливість фільтрації даних за датою, типом параметра (температура, тиск тощо), а також сортування даних за різними критеріями.
- Аналітика та статистичний аналіз: База даних повинна дозволяти отримувати агреговані дані для аналізу стану двигунів, наприклад, середню температуру, тиск, оберти за певний період. Це може включати генерацію базових статистичних звітів.
- Інтеграція з іншими системами: Система повинна мати можливість інтеграції з іншими програмними рішеннями, які забезпечують моніторинг та управління судном. Це може бути досягнуто за допомогою API, що дозволить стороннім системам зчитувати та записувати дані.

- Керування доступом: Система повинна забезпечувати контроль доступу користувачів із різними рівнями прав (наприклад, адміністратор, оператор), щоб захистити дані від несанкціонованого доступу та редагування.

2.1.2. Нефункціональні вимоги

- Продуктивність: Система повинна обробляти великі обсяги даних від численних датчиків в реальному часі, щоб запобігти затримкам у записі та доступі до даних.
- Масштабованість: База даних має бути масштабованою, що дозволить у майбутньому додавати більше суден і двигунів без зниження продуктивності системи.
- Надійність: Система повинна забезпечувати надійне збереження даних, включно з резервним копіюванням, для запобігання втратам даних у разі збою.
- Доступність: База даних повинна бути доступною 24/7, щоб забезпечити постійний моніторинг стану двигунів і можливість віддаленого доступу до даних.
- Безпека: Захист даних є важливим, оскільки інформація про технічний стан суден має бути захищена від несанкціонованого доступу. Це включає шифрування даних та аутентифікацію користувачів.
- Легкість обслуговування: Система повинна бути легкою у налаштуванні та підтримці, що дозволить швидко адаптувати її до змінних вимог або додаткових параметрів для моніторингу.

2.1.3 Користувацькі сценарії

- Оператор моніторингу переглядає в реальному часі показники температури, тиску, обертів двигуна і визначає потенційні відхилення.

- Технік з обслуговування використовує історичні дані для аналізу стану двигуна, наприклад, переглядає середні показники температури або тиску за минулий місяць.
- Адміністратор додає нові судна чи двигуни до бази даних, налаштовує рівні доступу користувачів та виконує резервне копіювання даних.

2.2 Концептуальна модель бази даних

Судно (Ship)

Кожен запис у цій сутності представляє окреме судно. Включає атрибути, які дозволяють ідентифікувати судно, такі як:

- Ship_ID (Ідентифікатор судна) – первинний ключ.
- Name (Назва судна).
- Type (Тип судна) – наприклад, вантажне, пасажирське тощо.
- Year_Built (Рік побудови).

Двигун (Engine)

Представляє двигуни, встановлені на суднах. Можливо, кілька двигунів можуть належати одному судну. Включає атрибути:

- Engine_ID (Ідентифікатор двигуна) – первинний ключ.
- Ship_ID (Ідентифікатор судна) – зовнішній ключ, який зв’язує двигун із судном.
- Type (Тип двигуна) – наприклад, дизельний, турбінний тощо.
- Power (Потужність).

- Model (Модель двигуна).

Зчитування (Reading)

Зберігає дані, отримані від датчиків двигуна в певний момент часу. Кожне зчитування містить показники роботи двигуна і прив’язане до конкретного двигуна через зовнішній ключ. Включає атрибути:

- Reading_ID (Ідентифікатор зчитування) – первинний ключ.
- Engine_ID (Ідентифікатор двигуна) – зовнішній ключ.
- Timestamp (Час зчитування) – дата і час, коли було отримано дані.
- Temperature (Температура двигуна).
- Cylinder_Pressure (Тиск у циліндрах).
- Oil_Pressure (Тиск масла).
- Boost_Pressure (Тиск наддуву).
- RPM (Оберти двигуна).
- Exhaust_Temperature (Температура вихлопних газів).
- Fuel_Consumption (Витрата палива).
- Coolant_Temperature (Температура охолоджувальної рідини).
- Vibration (Вібрація двигуна).

Тип події (EventType)

Містить типи подій, що можуть фіксуватися в системі, наприклад, аварія, планове обслуговування, діагностика тощо. Включає атрибути:

- EventType_ID (Ідентифікатор типу події) – первинний ключ.

- Name (Назва типу події) – опис типу події.

Подія (Event)

Зберігає дані про певні події, які відбуваються з двигуном. Наприклад, аварії чи планове обслуговування, що можуть мати вплив на технічний стан. Включає атрибути:

- Event_ID (Ідентифікатор події) – первинний ключ.
- Engine_ID (Ідентифікатор двигуна) – зовнішній ключ.
- EventType_ID (Ідентифікатор типу події) – зовнішній ключ.
- Timestamp (Час події).
- Description (Опис події) – додаткові примітки щодо події.

Зв’язки між сутностями

Судно (Ship) – Двигун (Engine): Одне судно може мати декілька двигунів, але кожен двигун прив’язаний лише до одного судна. Це зв’язок один-до-багатьох.

Двигун (Engine) – Зчитування (Reading): Один двигун може мати багато зчитувань за певний період, але кожне зчитування відноситься тільки до одного двигуна. Це також зв’язок один-до-багатьох.

Тип події (EventType) – Подія (Event): Один тип події може мати багато записів у таблиці Подія, але кожна конкретна подія належить тільки одному типу. Це зв’язок один-до-багатьох.

Двигун (Engine) – Подія (Event): Один двигун може мати багато подій, але кожна подія стосується конкретного двигуна. Це також зв’язок один-до-багатьох.

2.3 Логічна модель бази даних

ER-діаграма бази даних

ER-діаграма (Entity-Relationship Diagram) є одним із ключових інструментів для проєктування бази даних. Вона візуально відображає структуру даних, які будуть зберігатися в системі, та їх взаємозв'язки. ER-діаграма дозволяє проєктувальникам, розробникам і аналітикам зрозуміти логічну організацію даних ще до початку створення фізичної бази даних.

Основна мета ER-діаграми — чітко визначити **сутності** (об'єкти, які потребують збереження у базі даних), **атрибути** (характеристики сутностей) і **зв'язки** між ними. Це дає можливість оптимізувати структуру даних, уникнути дублювання та забезпечити цілісність інформації.

Структура ER-діаграми для моніторингу стану суднових двигунів

ER-діаграма розробленої бази даних для моніторингу технічного стану суднових дизельних двигунів включає наступні основні компоненти:

1. **Сутності (Entities):**

- **Судно (Ship):**

Ця сутність відповідає за збереження інформації про судна, на яких встановлені двигуни. Атрибути:

  - Ship_ID — унікальний ідентифікатор судна (ключ).
  - Name — назва судна.
  - Type — тип судна (наприклад, вантажне, пасажирське).
  - Year_Built — рік побудови судна.

- **Двигун (Engine):**

Сутність зберігає дані про двигуни, встановлені на суднах. Атрибути:

- Engine_ID — унікальний ідентифікатор двигуна (ключ).
- Ship_ID — зовнішній ключ для зв’язку з судном.
- Type — тип двигуна (дизельний, газотурбінний тощо).
- Power — потужність двигуна (у кВт).
- Model — модель двигуна.

- **Зчитування (Reading):**

Ця сутність відповідає за збереження технічних параметрів роботи двигунів у різний час. Атрибути:

- Reading_ID — унікальний ідентифікатор зчитування (ключ).
- Engine_ID — зовнішній ключ для зв’язку з двигуном.
- Timestamp — дата та час зчитування параметрів.
- Temperature — температура двигуна.
- Oil_Pressure — тиск масла.
- RPM — кількість обертів на хвилину.
- Exhaust_Temperature — температура вихлопних газів.

- **Тип події (EventType):**

Зберігає довідкову інформацію про типи подій, пов’язаних із двигунами (наприклад, поломка, обслуговування). Атрибути:

- EventType_ID — унікальний ідентифікатор типу події (ключ).

- Name — назва типу події.

- **Подія (Event):**

Ця сутність описує конкретні події, пов’язані з двигунами, наприклад, обслуговування чи аварію. Атрибути:

- Event_ID — унікальний ідентифікатор події (ключ).
- Engine_ID — зовнішній ключ для зв’язку з двигуном.
- EventType_ID — зовнішній ключ для зв’язку з типом події.
- Timestamp — дата та час події.
- Description — опис події.

Зв’язки між сутностями (Relationships):

- **“Судно” та “Двигун”** мають зв’язок **“один-до-багатьох”** (1:N). Одне судно може мати декілька двигунів, але кожен двигун прив’язаний лише до одного судна.
- **“Двигун” та “Зчитування”** мають зв’язок **“один-до-багатьох”** (1:N). Один двигун може мати багато зчитувань технічних параметрів у різний час.
- **“Двигун” та “Подія”** мають зв’язок **“один-до-багатьох”** (1:N). Кожен двигун може мати декілька пов’язаних подій.
- **“Подія” та “Тип події”** мають зв’язок **“багато-до-одного”** (N:1). Кілька подій можуть належати одному типу події.

Значення ER-діаграми для проєкту

1. **Візуалізація структури даних:**

ER-діаграма дозволяє легко зрозуміти, як організована база даних, які сутності в ній присутні та як вони взаємодіють одна з одною.

2. **Уникнення дублювання даних:**

Завдяки правильно спроєктованим зв’язкам між сутностями забезпечується нормалізація даних, що усуває дублювання інформації.

3. **Логічна організація даних:**

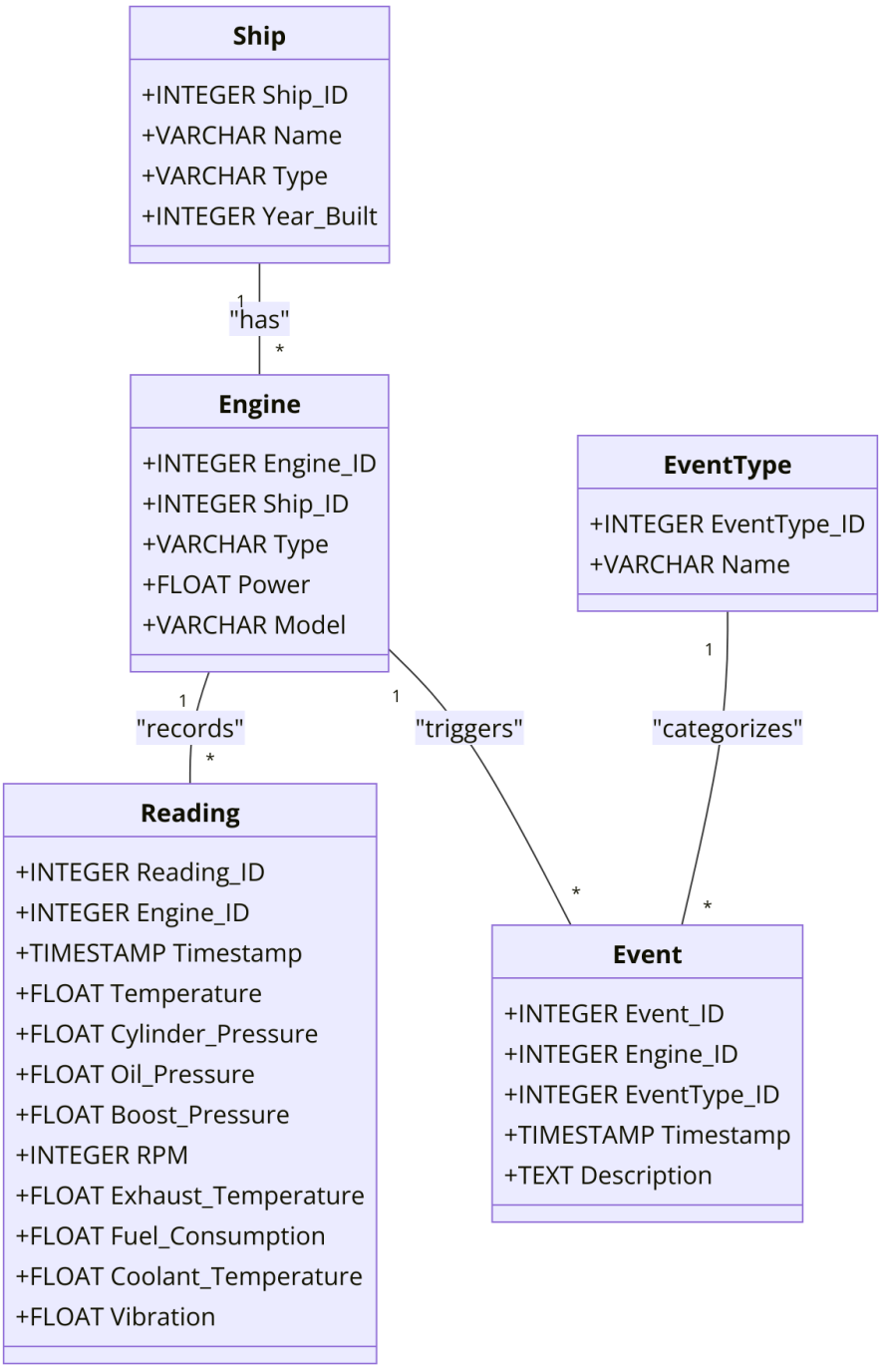
Зв’язки типу “один-до-багатьох” забезпечують цілісність даних та зручність їх збереження й обробки.

4. **Можливість подальшої масштабованості:**

ER-діаграма полегшує розширення структури бази даних, оскільки додаючи нові сутності або зв’язки, можна зберегти існуючу логіку.

5. **Основу для реалізації запитів:**

Чітке розуміння структури дозволяє створювати ефективні SQL-запити для отримання необхідної інформації.



Логічна модель бази даних деталізує концептуальну модель, надаючи структуру таблиць із визначенням полів, типів даних, ключів і зв’язків. Вона описує, як інформація буде організована в базі даних на рівні таблиць і

стосунків між ними. Нижче наведена логічна модель для бази даних, яка зберігатиме дані про технічний стан суднових двигунів.

Таблиця Ship (Судно)

Ця таблиця зберігає інформацію про судна.

Поле	Тип даних	Опис
Ship_ID	INTEGER	Первинний ключ
Name	VARCHAR	Назва судна
Type	VARCHAR	Тип судна (вантажне
Year_Built	INTEGER	Рік побудови судна

Зв'язки: Один-до-багатьох із таблицею Engine.

Таблиця Engine (Двигун)

Ця таблиця зберігає інформацію про двигуни, встановлені на суднах.

Поле	Тип даних	Опис
Engine_ID	INTEGER	Первинний ключ
Ship_ID	INTEGER	Зовнішній ключ
Type	VARCHAR	Тип двигуна (дизельний
Power	FLOAT	Потужність двигуна
Model	VARCHAR	Модель двигуна

Зв'язки:

- Багато-до-одного із таблицею Ship (одне судно може мати кілька двигунів).

- Один-до-багатьох із таблицею Reading.
- Один-до-багатьох із таблицею Event.

Таблиця Reading (Зчитування)

Ця таблиця зберігає показники, отримані від датчиків на двигунах у різні моменти часу.

Поле	Тип даних	Опис
Reading_ID	INTEGER	Первинний ключ
Engine_ID	INTEGER	Зовнішній ключ
Timestamp	TIMESTAMP	Час зчитування даних
Temperature	FLOAT	Температура двигуна
Cylinder_Pressure	FLOAT	Тиск у циліндрах
Oil_Pressure	FLOAT	Тиск масла
Boost_Pressure	FLOAT	Тиск наддуву
RPM	INTEGER	Оберти двигуна за хвилину
Exhaust_Temperature	FLOAT	Температура вихлопних газів
Fuel_Consumption	FLOAT	Витрата палива
Coolant_Temperature	FLOAT	Температура охолоджувальної рідини
Vibration	FLOAT	Вібрація двигуна

Зв’язки: Багато-до-одного із таблицею Engine.

Таблиця EventType (Тип події)

Ця таблиця містить типи подій, які можуть відбуватися з двигуном (наприклад, аварія, планове обслуговування).

Поле	Тип даних	Опис
EventType_ID	INTEGER	Первинний ключ
Name	VARCHAR	Назва типу події

Зв’язки: Один-до-багатьох із таблицею Event.

Таблиця Event (Подія)

Ця таблиця зберігає інформацію про події, що відбулися з двигуном.

Поле	Тип даних	Опис
Event_ID	INTEGER	Первинний ключ
Engine_ID	INTEGER	Зовнішній ключ
EventType_ID	INTEGER	Зовнішній ключ
Timestamp	TIMESTAMP	Час події
Description	TEXT	Опис події

Зв’язки:

- Багато-до-одного із таблицею Engine.
- Багато-до-одного із таблицею EventType.

Опис зв’язків між таблицями

- **Ship – Engine**: Зв’язок один-до-багатьох. Одне судно може мати кілька двигунів, але кожен двигун прив’язаний лише до одного судна.
- **Engine – Reading**: Зв’язок один-до-багатьох. Один двигун може мати багато записів про зчитування параметрів, але кожне зчитування стосується тільки одного двигуна.

- **EventType – Event**: Зв’язок один-до-багатьох. Один тип події може мати кілька записів у таблиці подій, але кожна подія належить до одного типу.

- **Engine – Event**: Зв’язок один-до-багатьох. Один двигун може мати багато подій, але кожна подія стосується конкретного двигуна.

2.4 Фізична модель бази даних

Для реалізації фізичної моделі бази даних у проєкті обрано систему керування базами даних (СКБД) PostgreSQL, яка забезпечує надійність, масштабованість і продуктивність у роботі з великими обсягами даних, що надходять у реальному часі. Нижче описані основні аспекти фізичної моделі, які враховують вибір типів даних, індексацію, оптимізацію продуктивності, масштабованість, резервне копіювання та безпеку даних.

Вибір типів даних

Для забезпечення ефективного використання пам’яті й оптимізації швидкості доступу до даних у PostgreSQL обрано відповідні типи даних для кожного поля:

- Цілі числа (INTEGER, BIGINT) використовуються для ідентифікаторів (Ship_ID, Engine_ID), які є унікальними ідентифікаторами у таблицях. BIGINT може застосовуватися для великих обсягів даних, щоб забезпечити масштабованість.

- Текстові поля (VARCHAR, TEXT) призначені для текстових значень змінної довжини, таких як назви та типи. VARCHAR(n) використовують для контрольованих текстових полів із зазначенням максимальної довжини, тоді як TEXT підходить для полів із довшими описами, як-от Description у таблиці Event.

- Числові поля (FLOAT, NUMERIC) застосовуються для полів, де зберігаються показники двигуна з плаваючою комою (наприклад, Power, Temperature). FLOAT забезпечує швидший доступ, а NUMERIC — точніші обчислення.
- Часові позначки (TIMESTAMP WITH TIME ZONE) використовуються для зберігання даних про час подій і зчитувань, що дозволяє враховувати різні часові зони і відстежувати дані у реальному часі.

Індексація

Індекси є ключовими для оптимізації швидкості запитів у PostgreSQL:

- Первинні та зовнішні ключі: PostgreSQL автоматично індексує первинні ключі, такі як Ship_ID, Engine_ID, Reading_ID, забезпечуючи швидкий доступ до даних. Зовнішні ключі (наприклад, Ship_ID у Engine і Engine_ID у Reading) індексуються для пришвидшення JOIN-запитів.
- Часові індекси: Індекси на полі Timestamp у таблиці Reading дозволяють швидко вибирати останні записи та отримувати актуальні дані про стан двигуна.
- Складені індекси (Composite Indexes): Для оптимізації складних запитів, які часто виконуються за кількома полями (наприклад, Engine_ID і Timestamp), створюються складені індекси.
- Часткові індекси (Partial Indexes): Часткові індекси застосовуються для фільтрації записів за певними умовами, що допомагає прискорити вибірку даних за заданими критеріями, наприклад, для активних двигунів.

Оптимізація продуктивності

- Розбиття таблиць (Partitioning): У PostgreSQL таблиці можна розбивати за діапазонами або списками. Наприклад, таблиця Reading, яка зберігає дані з високою частотою зчитувань, розбивається за Timestamp, щоб

зберігати дані за місяцями чи роками. Це зменшує обсяг даних в одній таблиці, що підвищує швидкість обробки.

- Foreign Data Wrappers (FDW): PostgreSQL підтримує підключення до зовнішніх джерел даних через FDW. Це дозволяє інтегрувати дані з інших систем моніторингу або аналітичних джерел у реальному часі.
- Матеріалізовані уявлення (Materialized Views): Для зберігання результатів складних запитів, які потребують агрегованих даних, використовуються матеріалізовані уявлення. Це дозволяє зберегти запити у фізичному вигляді та оновлювати їх за розкладом, зменшуючи навантаження на базу даних під час запитів.

Масштабованість та резервне копіювання

- Реплікація: PostgreSQL дозволяє налаштувати реплікацію для високої доступності та розподілу навантаження. Основний сервер відповідає за запис, тоді як додаткові сервери обслуговують запити на читання, що корисно для забезпечення відмовостійкості.
- Резервне копіювання: Регулярне резервне копіювання реалізується за допомогою інструментів pg_dump для збереження структури й даних бази та pg_basebackup для повного резервного копіювання на рівні файлової системи.
- Point-in-Time Recovery (PITR): Для забезпечення можливості відновлення бази даних до певного моменту використовується архівування журналів транзакцій (WAL). Це дозволяє відновити інформацію до конкретного часу у випадку збою.

Безпека даних

- Аутентифікація та авторизація: PostgreSQL підтримує різні методи аутентифікації, включно з паролями, Kerberos, SSL-сертифікатами.

SSL-з’єднання захищає підключення до бази даних від несанкціонованого доступу.

- Ролі та привілеї: Для управління доступом створюються окремі ролі для адміністраторів, операторів і технічних працівників, з налаштуванням відповідних прав на доступ до таблиць та операцій.
- Шифрування: У разі зберігання конфіденційної інформації в базі даних можна налаштувати шифрування на рівні диска за допомогою інструментів, таких як LUKS, або шифрувати конкретні дані в базі даних.

Інфраструктура для зберігання даних

1. Апаратне забезпечення: Сервери з SSD-дисками забезпечують високу швидкість доступу до даних і зменшують затримки, особливо для таблиць з інтенсивними операціями читання і запису.
2. Розширюваність та масштабованість: Система підтримує горизонтальне масштабування, що дозволяє додавати сервери для розподілу навантаження на читання, або вертикальне (збільшення потужності серверів), щоб задовольнити зростаючі вимоги до даних.

Фізична модель у PostgreSQL забезпечує ефективне зберігання та обробку даних для моніторингу технічного стану суднових двигунів, використовуючи всі можливості цієї СУБД. Реплікація, розбиття таблиць, індексація та матеріалізовані уявлення дозволяють досягти високої продуктивності та забезпечують гнучкість у розширенні функціоналу. Це робить систему надійною та придатною для тривалої експлуатації в умовах морського транспорту, де потрібні безперервний моніторинг і високий рівень доступності даних.

Таблиця Ship (Судно)

Ця таблиця зберігає інформацію про судна.

Поле	Тип даних	Опис
Ship_ID	INTEGER	Первинний ключ
Name	VARCHAR	Назва судна
Type	VARCHAR	Тип судна (вантажне
Year_Built	INTEGER	Рік побудови судна

Зв’язки: Один-до-багатьох із таблицею Engine.

Таблиця Engine (Двигун)

Ця таблиця зберігає інформацію про двигуни, встановлені на суднах.

Поле	Тип даних	Опис
Engine_ID	INTEGER	Первинний ключ
Ship_ID	INTEGER	Зовнішній ключ
Type	VARCHAR	Тип двигуна (дизельний
Power	FLOAT	Потужність двигуна
Model	VARCHAR	Модель двигуна

Зв’язки:

- Багато-до-одного із таблицею Ship (одне судно може мати кілька двигунів).
- Один-до-багатьох із таблицею Reading.
- Один-до-багатьох із таблицею Event.

Таблиця Reading (Зчитування)

Ця таблиця зберігає показники, отримані від датчиків на двигунах у різні моменти часу.

Поле	Тип даних	Опис
Reading_ID	INTEGER	Первинний ключ
Engine_ID	INTEGER	Зовнішній ключ
Timestamp	TIMESTAMP	Час зчитування даних
Temperature	FLOAT	Температура двигуна
Cylinder_Pressure	FLOAT	Тиск у циліндрах
Oil_Pressure	FLOAT	Тиск масла
Boost_Pressure	FLOAT	Тиск наддуву
RPM	INTEGER	Оберти двигуна за хвилину
Exhaust_Temperature	FLOAT	Температура вихлопних газів
Fuel_Consumption	FLOAT	Витрата палива
Coolant_Temperature	FLOAT	Температура охолоджувальної рідини
Vibration	FLOAT	Вібрація двигуна

Зв’язки: Багато-до-одного із таблицею Engine.

Таблиця EventType (Тип події)

Ця таблиця містить типи подій, які можуть відбуватися з двигуном (наприклад, аварія, планове обслуговування).

Поле	Тип даних	Опис
EventType_ID	INTEGER	Первинний ключ
Name	VARCHAR	Назва типу події

Зв'язки: Один-до-багатьох із таблицею Event.

Таблиця Event (Подія)

Ця таблиця зберігає інформацію про події, що відбулися з двигуном.

Поле	Тип даних	Опис
Event_ID	INTEGER	Первинний ключ
Engine_ID	INTEGER	Зовнішній ключ
EventType_ID	INTEGER	Зовнішній ключ
Timestamp	TIMESTAMP	Час події
Description	TEXT	Опис події

Зв'язки:

- Багато-до-одного із таблицею Engine.
- Багато-до-одного із таблицею EventType.

2.5 Опис основних операцій

Основні операції, які виконуватимуться над базою даних для моніторингу технічного стану суднових дизельних двигунів, включають додавання,

оновлення, видалення та вибірку даних. Ці операції забезпечують зберігання, коригування, очищення та отримання інформації для аналізу. Наведені приклади SQL-запитів допоможуть зрозуміти, як реалізуються ці операції в базі даних.

1. Додавання даних

Додавання нового судна до таблиці Ship:

```
INSERT INTO Ship (Ship_ID, Name, Type, Year_Built)
VALUES (1, 'Aegis', 'Cargo', 2015);
```

Додавання нового двигуна до таблиці Engine, прив’язаного до певного судна:

```
INSERT INTO Engine (Engine_ID, Ship_ID, Type, Power, Model)
VALUES (1, 1, 'Diesel', 2500, 'MTU-4000');
```

Додавання нового зчитування до таблиці Reading:

```
INSERT INTO Reading (
    Reading_ID, Engine_ID, Timestamp, Temperature, Cylinder_Pressure, Oil_Pressure,
    Boost_Pressure, RPM, Exhaust_Temperature, Fuel_Consumption, Coolant_Temperature, Vibration
) VALUES (
    1, 1, '2024-11-10 14:35:00', 85.2, 210.5, 5.3, 2.0, 1500, 500, 12.5, 65.0, 0.02
);
```

2. Оновлення даних

Оновлення інформації про судно в таблиці Ship (наприклад, зміна типу судна):

```
UPDATE Ship
SET Type = 'Passenger'
WHERE Ship_ID = 1;
```

Оновлення поточних значень зчитування двигуна в таблиці Reading:

```
UPDATE Reading
SET Temperature = 90.0, RPM = 1550
WHERE Reading_ID = 1;
```

Оновлення опису події у таблиці Event:

```
UPDATE Event
SET Description = 'Планове обслуговування системи охолодження'
WHERE Event_ID = 1;
```

3. Видалення даних

Видалення двигуна з таблиці Engine:

```
DELETE FROM Engine
WHERE Engine_ID = 1;
```

Видалення записів зчитувань, що старші за один рік:

```
DELETE FROM Reading
WHERE Timestamp < NOW() - INTERVAL '1 year';
```

Видалення події з таблиці Event:

```
DELETE FROM Event
WHERE Event_ID = 1;
```

4. Вибірка даних

Отримання всієї інформації про конкретне судно:

```
SELECT * FROM Ship
WHERE Ship_ID = 1;
```

Отримання останніх 5 зчитувань для конкретного двигуна:

```
SELECT * FROM Reading
WHERE Engine_ID = 1
ORDER BY Timestamp DESC
LIMIT 5;
```

Отримання середніх значень температури та тиску для конкретного двигуна за останній місяць:

```
SELECT AVG(Temperature) AS Avg_Temperature, AVG(Cylinder_Pressure) AS Avg_Cylinder_Pressure
FROM Reading
WHERE Engine_ID = 1 AND Timestamp >= NOW() - INTERVAL '1 month';
```

Вибірка всіх подій типу “Аварія” для конкретного двигуна:

```
SELECT e.Event_ID, e.Timestamp, e.Description
FROM Event e
JOIN EventType et ON e.EventType_ID = et.EventType_ID
WHERE e.Engine_ID = 1 AND et.Name = 'Аварія';
```

Отримання списку всіх суден разом з кількістю встановлених двигунів:

```
SELECT s.Name, COUNT(e.Engine_ID) AS Engine_Count
FROM Ship s
LEFT JOIN Engine e ON s.Ship_ID = e.Ship_ID
GROUP BY s.Name;
```

- Додавання дозволяє вставляти нові записи в таблиці для суден, двигунів, зчитувань та подій.
- Оновлення забезпечує зміну існуючих даних, наприклад, оновлення параметрів двигуна або опису події.
- Видалення дозволяє очищати дані, наприклад, видалення старих зчитувань або непотрібних записів про події.

- Вибірка забезпечує отримання даних для аналізу, зокрема середніх значень, останніх зчитувань, подій певного типу тощо.

Ці SQL-запити представляють основні операції, які забезпечують управління даними у системі моніторингу, допомагаючи ефективно працювати з базою та отримувати необхідну інформацію для аналізу технічного стану суднових двигунів.

2.6 Висновки до Розділу 2

Розроблена база даних для моніторингу технічного стану суднових дизельних двигунів забезпечує надійне та ефективне зберігання, обробку і доступ до великого обсягу даних, які збираються у реальному часі. Основні переваги бази даних включають:

1. Надійне зберігання даних: Структура бази даних дозволяє зберігати повну історію показників роботи двигунів, включно з температурою, тиском, обертами та іншими критично важливими параметрами. Це надає можливість для детального аналізу та діагностики, що підвищує безпеку і надійність роботи суден.

2. Оптимізація продуктивності: Використання індексації, розбиття таблиць та оптимізованих SQL-запитів забезпечує високу швидкість доступу до даних навіть при великому обсязі записів. Це особливо важливо для отримання оперативної інформації у реальному часі.

3. Гнучка інтеграція: База даних підтримує інтеграцію з іншими системами моніторингу та управління завдяки можливості підключення зовнішніх даних (Foreign Data Wrappers) та доступу через API. Це дозволяє розширювати можливості моніторингу та з’єднувати систему з іншими інформаційними системами компанії.

4. Забезпечення безпеки: Використання ролей, привілеїв, шифрування та безпечної аутентифікації дозволяє захистити конфіденційні дані від несанкціонованого доступу, що є важливим при управлінні технічними даними суден.

5. Висока доступність та відмовостійкість: Реплікація даних та резервне копіювання забезпечують безперервний доступ до даних, навіть у випадку збою основного сервера. Це знижує ризик втрати даних та сприяє надійності системи.

Подальші перспективи розширення та вдосконалення системи

1. Розширення показників моніторингу: Базу даних можна доповнити додатковими параметрами, наприклад, моніторингом вібрації корпусу або температури навколишнього середовища. Це дозволить отримувати більш комплексні дані про стан двигунів.

2. Аналіз та прогнозування: Інтеграція з інструментами аналітики, такими як машинне навчання, дозволить прогнозувати можливі несправності на основі історичних даних. Це сприятиме більш ефективному плануванню технічного обслуговування та зниженню витрат на ремонт.

3. Розширення функціоналу для віддаленого моніторингу: Додаткові інтерфейси для мобільних додатків чи веб-платформ дозволять технічним працівникам віддалено контролювати стан суден та отримувати сповіщення про відхилення в режимах роботи.

4. Оптимізація для великих даних: У майбутньому можливо інтегрувати рішення для обробки великих даних, що дозволить ефективніше управляти даними, зібраними від флотилії суден, та поліпшити швидкість обробки.

5. Підтримка стандартизації даних: З впровадженням міжнародних стандартів для моніторингу та управління судновими двигунами система

може бути розширена для підтримки стандартів (наприклад, ISO або IMO), що спростить її інтеграцію з зовнішніми системами.

Створена база даних є надійною та масштабованою основою для управління технічними параметрами суднових двигунів і має потенціал для подальшого розвитку, що забезпечить підтримку інноваційних технологій і нових потреб у сфері морського транспорту.

Розділ 3. Розробка серверного програмного забезпечення для системи моніторингу

У цьому розділі описано процес розробки системи моніторингу технічного стану суднових двигунів, яка включає проєктування бази даних та реалізацію REST API для роботи з нею. Основна мета полягає у створенні ефективного інструменту для зберігання, обробки та доступу до даних про роботу суднових двигунів, що дозволить забезпечити їх надійний моніторинг та аналіз.

Розробка сервісу передбачає кілька ключових етапів. Спочатку обґрунтовується вибір бази даних PostgreSQL, яка стане основою для зберігання інформації. Далі виконується проєктування структури бази даних, що враховує зв’язки між таблицями, зокрема суднами, двигунами, зчитуваннями технічних параметрів та подіями.

Для реалізації REST API використовується фреймворк NestJS, який дозволяє створити гнучкий, масштабований і легко підтримуваний серверний додаток. REST API забезпечить доступ до бази даних, підтримуючи CRUD-операції (створення, читання, оновлення, видалення) для всіх ключових сутностей. Окрім того, описано етапи інтеграції всіх компонентів системи та тестування API для забезпечення його коректної роботи.

Таким чином, у цьому розділі детально розглянуто всі аспекти проєктування та реалізації сервісу, що дозволить створити ефективну та надійну систему для моніторингу технічного стану суднових двигунів.

3.1 Процес ініціалізації нової бази даних і створення нового користувача в PostgreSQL

Вибір системи керування базами даних (СКБД) є ключовим етапом під час проєктування системи моніторингу технічних параметрів суднових двигунів. PostgreSQL обрано як базу даних для цього проєкту з кількох причин. Ця СКБД є однією з найпотужніших реляційних баз даних з відкритим кодом, що забезпечує високу надійність, масштабованість і продуктивність. PostgreSQL

підтримує складні типи даних, включаючи реляційні зв’язки та JSON, а також забезпечує функціональність, необхідну для зберігання великого обсягу даних і виконання складних запитів. Крім того, її інтеграція з сучасними ORM, такими як TypeORM, спрощує роботу з базою даних у серверному середовищі.

Для створення нового користувача та бази даних у PostgreSQL виконуються наступні дії:

1. Підключення до PostgreSQL як суперкористувач

Підключення до PostgreSQL здійснюється під обліковим записом суперкористувача (наприклад, postgres), який має всі необхідні права для створення користувачів і баз даних. Команда для підключення:

```
psql -U postgres
```

2. Створення нового користувача

Для створення нового користувача, який буде використовуватись для доступу до бази даних, використовується команда:

```
CREATE USER dispatcher WITH PASSWORD 'password';
```

- **dispatcher** – ім’я нового користувача.
- **‘password’** – пароль, який буде використовуватись для автентифікації користувача.

3. Створення нової бази даних

Далі створюється нова база даних і задається її власник – користувач, створений на попередньому кроці:

```
CREATE DATABASE ships_monitoring OWNER dispatcher;
```

- **ships_monitoring** – назва нової бази даних.

- **OWNER dispatcher** – власником бази даних призначається користувач dispatcher.

4. Перевірка створення користувача і бази даних

Щоб переконатися, що користувача і базу даних створено успішно, виконуються наступні команди:

- Перегляд усіх користувачів (ролей):

```
\du
```

- Перегляд усіх баз даних:

```
\l
```

5. Підключення до нової бази даних

Після створення бази даних і користувача можна підключитися до бази даних під обліковим записом нового користувача. Для цього завершується поточна сесія і виконується команда:

```
psql -U dispatcher -d ships_monitoring
```

- **-U dispatcher** – підключення від імені користувача dispatcher.
- **-d ships_monitoring** – підключення до бази даних ships_monitoring.

6. Надання додаткових прав

Якщо необхідно надати користувачеві додаткові права доступу до бази даних, виконується команда:

```
GRANT ALL PRIVILEGES ON DATABASE ships_monitoring TO dispatcher;
```

Ці кроки дозволили створити нового користувача та базу даних, які можна використовувати для подальшої розробки сервісу.

3.2 Створення RESTful API

NestJS обрано для реалізації REST API завдяки його модульній архітектурі, що дозволяє легко масштабувати і підтримувати систему. Цей фреймворк

забезпечує високу продуктивність завдяки використанню Node.js і підтримці асинхронної роботи. NestJS надає розробникам інтуїтивний синтаксис та інтегрується з TypeORM, що значно полегшує роботу з базами даних, такими як PostgreSQL. Вибір NestJS також обумовлений його сумісністю з сучасними стандартами TypeScript, що підвищує безпеку коду та спрощує його підтримку. Завдяки вбудованій підтримці валідаторів, гнучкості в налаштуванні маршрутизації й зручності тестування, NestJS стає оптимальним рішенням для створення REST API будь-якої складності.

1. Ініціалізація проєкту NestJS

1. За допомогою команди nest new my-project створено новий проєкт NestJS.
2. Встановлено необхідні залежності:
   - @nestjs/typeorm для інтеграції з базою даних.
   - typeorm та pg для роботи з PostgreSQL.
   - class-validator і class-transformer для перевірки даних.
3. У файлі app.module.ts налаштовано підключення до бази даних PostgreSQL через TypeOrmModule.

2. Реалізація сутностей

Для кожної сутності створено:

1. **Модуль** — команда nest generate module <entity-name>.
2. **Сервіс** — команда nest generate service <entity-name> --no-spec.
3. **Контролер** — команда nest generate controller <entity-name> --no-spec.

Під час створення сутностей використано наступні підходи:

• **Опис структури сутності**: у файлі <entity-name>.entity.ts описано структуру таблиці, включаючи зовнішні ключі для зв’язків між таблицями.

• **Зв’язки** між сутностями (OneToMany, ManyToOne) налаштовано за допомогою декораторів TypeORM.

3. Реалізація функціоналу для кожної сутності

Сутність Ship

1. Створено модуль ShipModule.

2. У файлі ship.entity.ts описано структуру таблиці Ship, яка містить поля Ship_ID, Name, Type, Year_Built.

3. У файлі ship.service.ts реалізовано методи:

   • Отримання всіх суден (findAll).

   • Отримання судна за ID (findOne).

   • Додавання нового судна (create).

   • Оновлення даних про судно (update).

   • Видалення судна (remove).

4. У файлі ship.controller.ts реалізовано маршрути:

   • GET /ships — отримання всіх суден.

   • GET /ships/:id — отримання конкретного судна.

   • POST /ships — додавання нового судна.

   • PUT /ships/:id — оновлення судна.

   • DELETE /ships/:id — видалення судна.

Сутність Engine

1. Створено модуль EngineModule.

2. У файлі engine.entity.ts описано таблицю Engine, яка містить зв’язок із таблицею Ship.

3. Реалізовано методи в engine.service.ts для роботи з двигунами.

4. У контролері engine.controller.ts налаштовано маршрути для роботи з двигунами, аналогічно до сутності Ship.

Сутність Reading

1. Створено модуль ReadingModule.

2. У файлі reading.entity.ts описано таблицю Reading, яка містить технічні параметри двигуна та зв’язок із таблицею Engine.

3. Реалізовано методи в reading.service.ts для:

- Отримання всіх зчитувань (findAll).
- Отримання конкретного зчитування (findOne).
- Додавання нового зчитування (create).
- Видалення зчитування (remove).

4. У файлі reading.controller.ts додано маршрути для відповідних CRUD-операцій.

Сутність EventType

1. Створено модуль EventTypeModule.

2. У файлі eventtype.entity.ts описано таблицю EventType, яка містить поля EventType_IDта Name.

3. У файлі eventtype.service.ts реалізовано методи для створення та отримання списку типів подій.

4. У контролері eventtype.controller.ts додано маршрути для:

- GET /eventtypes — отримання списку типів подій.
- POST /eventtypes — створення нового типу події.

Сутність Event

1. Створено модуль EventModule.

2. У файлі event.entity.ts описано таблицю Event, яка містить зв’язки з таблицями Enginета EventType.

3. Реалізовано методи в event.service.ts для:

- Отримання списку подій.
- Додавання нової події.

4. У файлі event.controller.ts налаштовано маршрути:

- GET /events — отримання списку подій.
- POST /events — створення нової події.

4. Інтеграція модулів

Кожен модуль (ShipModule, EngineModule, ReadingModule, EventTypeModule, EventModule) імпортовано до AppModule через файл app.module.ts. Це забезпечує доступність API через HTTP-запити.

5. Тестування API

1. Для тестування кожного ендпоінта використано інструмент Postman.

2. Перевірено коректність CRUD-операцій для кожної сутності:

- Додавання нових записів (POST-запити).
- Отримання даних (GET-запити).
- Оновлення даних (PUT-запити).
- Видалення записів (DELETE-запити).

У рамках даної роботи частина інтерфейсу для користувачів (frontend) не була реалізована, оскільки основна увага зосереджена на розробці серверної частини та бази даних. Проте за необхідності інтерфейс може бути доданий у майбутньому для полегшення взаємодії з системою.

Для створення інтерфейсу можна використати сучасні інструменти, такі як React або Angular, які забезпечують динамічний і зручний користувацький досвід. Інтерфейс дозволить користувачам виконувати такі дії, як перегляд даних, додавання нових записів, оновлення існуючих та аналіз інформації через графічні звіти або таблиці.

Завдяки тому, що REST API вже реалізований, інтеграція інтерфейсу з серверною частиною не потребуватиме значних зусиль і дозволить розширити функціонал системи без зміни її основної архітектури.

3.3 Висновки до Розділу 3

У третьому розділі було детально розглянуто процес розробки системи моніторингу технічного стану суднових двигунів, який включав проєктування бази даних, вибір інструментів для реалізації та створення REST API для роботи з даними.

Розробка системи почалася з обґрунтування вибору бази даних PostgreSQL. Ця система керування базами даних була обрана через її надійність, продуктивність, підтримку складних типів даних і реляційних зв’язків, що є критично важливими для зберігання структурованої інформації про судна, двигуни, технічні параметри та події. PostgreSQL також забезпечує високу масштабованість, що дозволяє системі зростати разом зі збільшенням обсягу даних.

Другим важливим етапом стало проєктування структури бази даних. Розроблено схему, що включає ключові сутності, такі як Ship (судна), Engine (двигуни), Reading (зчитування параметрів), EventType (типи подій) та Event (події). Усі сутності було пов’язано між собою реляційними зв’язками, що дозволяє забезпечити цілісність і зручність роботи з даними. Фізична модель

бази даних враховує особливості PostgreSQL, такі як індексація для прискорення запитів та автоматичне створення зовнішніх ключів.

Для реалізації серверної частини було використано фреймворк NestJS, який забезпечив гнучкість і структурованість коду. Реалізовано REST API, що включає маршрути для виконання CRUD-операцій для кожної сутності. Це дозволяє взаємодіяти з базою даних для додавання, оновлення, видалення та отримання даних. API побудовано з використанням сучасних підходів, таких як модульна архітектура та асинхронна обробка запитів, що забезпечує масштабованість та зручність підтримки.

Хоча інтерфейс користувача (frontend) не був реалізований у межах цієї роботи, REST API готовий до інтеграції з будь-якою клієнтською платформою. У майбутньому це дозволить створити зручний веб- або мобільний інтерфейс для візуалізації даних, аналізу та взаємодії з системою.

Під час розробки проведено тестування роботи API за допомогою інструментів, таких як Postman, для перевірки коректності виконання запитів. Тестування підтвердило правильність роботи всіх CRUD-операцій, відповідність структур даних вимогам та ефективність роботи бази даних.

Таким чином, у третьому розділі створено повноцінну серверну частину системи, яка готова до розгортання та подальшої інтеграції з іншими компонентами. Реалізована архітектура є масштабованою, гнучкою та відповідає сучасним вимогам до програмного забезпечення. Розроблена система забезпечує зберігання, обробку та доступ до даних, що дозволяє ефективно здійснювати моніторинг технічного стану суднових двигунів.

Розділ 4. Охорона праці

4.1 Основні положення охорони праці

Охорона праці — це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних та профілактичних заходів, спрямованих на збереження життя, здоров'я і працездатності людини в процесі її трудової діяльності. Основні положення щодо охорони праці регулюються Законом України «Про охорону праці» від 14.10.1992 року, який визначає права та обов'язки працівників і роботодавців у питаннях безпеки, гігієни праці та виробничого середовища.

Державна політика в галузі охорони праці базується на таких принципах:

- Пріоритету життя і здоров'я працівників над результатами виробничої діяльності.
- Відповідальності роботодавця за створення безпечних умов праці.
- Соціального захисту працівників, які постраждали від нещасних випадків на виробництві.
- Забезпечення координації між органами державної влади та роботодавцями у вирішенні питань охорони праці.

4.2 Аналіз небезпечних і шкідливих факторів під час роботи з програмно-апаратними комплексами

Розробка і експлуатація програмно-апаратних систем для моніторингу технічного стану суднових двигунів передбачає використання комп'ютерної техніки, що може створювати небезпечні і шкідливі виробничі фактори. До основних небезпечних і шкідливих факторів, які можуть впливати на працівників, належать:

1. Фізичні фактори:

• Електромагнітне випромінювання від моніторів і серверів.

Рівень випромінювання сучасних моніторів зазвичай не перевищує встановлених норм, проте тривала робота з комп’ютером може призвести до кумулятивного впливу на організм.

• Електричний струм. Робота з комп’ютерним обладнанням передбачає ризик ураження струмом через можливі пошкодження ізоляції чи помилки в електромережі.

• Шум від серверів і охолоджувальних систем. Постійний шум на рівні понад 85 дБ може викликати роздратування, головний біль та зниження працездатності.

2. Психофізіологічні фактори:

• Розумова перенапруга через тривалу концентрацію уваги під час роботи над проєктами з моніторингу технічного стану.

• Монотонність праці, яка може викликати втому та зниження продуктивності.

• Напруга зорового апарату через недостатнє освітлення або неправильно налаштоване робоче місце.

3. Санітарно-гігієнічні умови:

• Недостатня вентиляція приміщення, де розміщено сервери, що може спричинити підвищення температури повітря та негативно вплинути на здоров’я персоналу.

4.3 Шляхи покращення умов праці

Для мінімізації впливу шкідливих і небезпечних факторів необхідно вживати такі заходи:

• Оптимізація робочого середовища:

- Оснащення приміщення сучасними системами вентиляції та кондиціювання для підтримання оптимального мікроклімату.
- Забезпечення робочого місця відповідним рівнем освітлення, з урахуванням норм ДСТУ для роботи з комп’ютерною технікою.
- Технічні заходи:
- Використання моніторів із низьким рівнем випромінювання.
- Регулярний огляд і технічне обслуговування електрообладнання для зниження ризику ураження струмом.
- Організаційні заходи:
- Встановлення перерв для відпочинку працівників, які працюють за комп’ютером.
- Організація навчань для персоналу з питань електробезпеки та правил роботи з комп’ютерною технікою.
- Використання шумопоглинаючих матеріалів для зниження рівня шуму у приміщеннях із серверами.

4.4 Особливості безпеки при роботі з серверним обладнанням

Серверне обладнання, що використовується для обробки даних у системі моніторингу, має високі вимоги до безпечної експлуатації. Основні заходи безпеки:

- Захист від перегріву: використання ефективних систем охолодження.
- Захист від перенапруг: встановлення джерел безперебійного живлення (UPS) для захисту обладнання та персоналу.
- Безпека кабельних систем: організація прокладки кабелів у захищених каналах для уникнення механічних пошкоджень.

Висновки до розділу 4

У розділі 4 розглянуто питання забезпечення безпечних умов праці для розробників та операторів програмно-апаратного комплексу моніторингу. Проаналізовано шкідливі та небезпечні фактори, пов’язані з роботою комп’ютерного та серверного обладнання, та запропоновано шляхи їх мінімізації. Реалізація запропонованих заходів дозволить покращити умови праці, знизити ризики професійних захворювань і підвищити продуктивність роботи персоналу.

Перелік літератури

1. DMSS HD - Apps on Google Play [Internet]. Available from: URL: https://play.google.com/store/apps/details?id=com.mm.android.DMSSHD&hl=en_US (Date: 20/08/2023).
2. MAN B&W PMI Introduction. MAN ME-C ENGINE OPERATION COURSE. – Копенгаген: MAN Diesel & Turbo, 2015. 31 p.
3. Diploma Thesis by KONSTANTINA BERDEMPE Electronic Marine Diesel Engine operating data management An approach to components' failures forecasting, 2020 [Internet]. Available from: URL: https://dspace.lib.ntua.gr/xmlui/bitstream/handle/123456789/51558/%CE%94%CE%99%CE%A0%CE%9B%CE%A9%CE%9C%CE%91%CE%A4%CE%99%CE%9A%CE%97%20%CE%9C%CE%A0%CE%95%CE%A1%CE%94%CE%95%CE%9C%CE%A0%CE%95.pdf?sequence=1. (Date: 02/12/2023).
4. Programming software - PLCNEXT ENGINEER [Internet]. Available from: URL: https://www.phoenixcontact.com/uk-ua/produkcija/programming-software-plcnext-engineer-1046008 (Date: 20/08/2023).
5. Vitalii Nikolskyi, Mykola Slobodianiuk, Maksym Levinskyi, Mark Nikolskyi Wireless Remote Control Systems Of Marine Diesel Engine, Helsinki, 12 p. (in print).
6. Голіков В.А., Нікольський В.В., Левінський М.В., Нікольський М.В., Слободянюк М.В. Модернізація системи віддаленого управління та контролю аварійного дизель-генератора навчального машино-котельного відділення // Суднові енергетичні установки:

науково-технічний збірник. Вип. 44. - Одеса: НУ «ОМА», 2022. – С. 64 – 70.

7. Миюсов М. В. Электронные системы управления главными судовыми двигателями: учебное пособие / М. В. Миюсов, В. И. Ланчуковский, Е. М. Оженко. – Одесса: ОНМА, 2013. – 98 с.

8. Робочий цикл корабельних дизелів та його індикаторні та ефективні показники: навчальний посібник / М.Г. Єрмошкін, Ю.В. Заблоцький, С.В. Сагін. – Одеса: НУ «ОМА», 2019. – 116 с.

9. Система моніторінгу завантаження контейнеровоза: звіт з НДР: ДР № 0117 U 000317 / кер. роботи В.В. Нікольський, виконавець Ю.О Накул. К.: УКРНТЕІ, 2018 – 79 с.

10. Сучасні суднові дизелі: особливості конструкції, експлуатації та автоматизованого управління / І. І. Черниш, С

11. Суворов П.С. Динаміка дизеля в судновому пропульсивному комплексі. Одеса: ОНМА, 2004. 304 с.

12. Суворов П.С. Управління режимами роботи головних суднових дизелів. Одеса: ЛАТСТАР, 2000. 238 с.

13. Тихонов І.В. Прогнозування параметрів траєкторії руху об’єктів на внутрішніх водних шляхах. «Практичні проблеми розвитку радіозв’язку та радіонавігації в ГМЗЛБ, у системах АІС, СУКС і РІС». Матеріали VIII науково-практичної конференції (Одеса, 6–7 листопада 2007 р.). Одеса: ОНМА, 2007. С. 8–9.

14. Тихонов І.В. Посібник судноводія малотоннажного судна / Тихонов І.В. та ін. Одеса: Фенікс, 2007. 302 с.

15. Lisowski J. Game control methods in navigator decision support system. The Archives of Transport. 2005. No 3–4, Vol. XVII. C. 133–147.

16. Reform in the inland water transport: China's experience URL: https://www.unescap.org/our-work/transport (дата звернення 12.12.2018).
17. Sustainable development of inland waterway transport in China (2009). Theme I of a World Bank Project: Comprehensive Transport System Analysis in China. URL: https://documents1.worldbank.org/curated/en/860411468024540285/pdf/549620WP0P109910printing1En109jul09.pdf (дата звернення 12.12.2018).

Додаток до курсової роботи: Проєкт REST API та UI

У межах курсової роботи розроблено проєкт REST API, який забезпечує взаємодію з базою даних для управління інформацією про судна, двигуни, технічні зчитування та події. Цей проєкт реалізовано з використанням фреймворку NestJS та бази даних PostgreSQL.

Інструкція з запуску REST API

1. Встановіть необхідні інструменти:
   - Node.js (версія 16 або новіша) – завантажити тут.
   - PostgreSQL – завантажити тут.
2. Клонування проєкту: завантажте архів проєкту за посиланням https://drive.google.com/file/d/1-A0xllfyfqeO87AeTyXt4Ly3EXePe7jL/view?usp=sharing та розпакуйте
3. Встановлення залежностей:

Виконайте команду для встановлення всіх необхідних бібліотек:

```
npm install
```

4. Налаштування бази даних:
   - Створіть базу даних у PostgreSQL:

```
CREATE DATABASE your_database_name;
```

   - У файлі .env або src/app.module.ts вкажіть параметри з’єднання з базою даних:

```
DATABASE_HOST=localhost
DATABASE_PORT=5432
DATABASE_USER=dispatcher
```

DATABASE_PASSWORD=password

DATABASE_NAME=ships_monitoring

5. Запуск сервера:

Для запуску проєкту використовуйте команду:

npm run start

Сервер буде доступний за адресою: http://localhost:3000.

Роути REST API

1. Ship (Судна)

```
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { Ship } from './ship.entity';

@Injectable()
export class ShipService {
  constructor(
    @InjectRepository(Ship)
    private shipRepository: Repository<Ship>,
  ) {}

  findAll(): Promise<Ship[]> {
    return this.shipRepository.find();
  }

  findOne(id: number): Promise<Ship> {
    return this.shipRepository.findOne({ where: { Ship_ID: id } });
  }

  create(ship: Partial<Ship>): Promise<Ship> {
    return this.shipRepository.save(ship);
  }

  async update(id: number, ship: Partial<Ship>): Promise<Ship> {
    await this.shipRepository.update(id, ship);
    return this.findOne(id);
  }

  async remove(id: number): Promise<void> {
    await this.shipRepository.delete(id);
  }
}
```

src > ship > ship.controller.ts > ...

```ts
import {
  Controller,
  Get,
  Post,
  Put,
  Delete,
  Param,
  Body,
} from '@nestjs/common';
import { ShipService } from './ship.service';
import { Ship } from './ship.entity';

@Controller('ships')
export class ShipController {
  constructor(private readonly shipService: ShipService) {}

  @Get()
  findAll(): Promise<Ship[]> {
    return this.shipService.findAll();
  }

  @Get(':id')
  findOne(@Param('id') id: number): Promise<Ship> {
    return this.shipService.findOne(id);
  }

  @Post()
  create(@Body() ship: Partial<Ship>): Promise<Ship> {
    return this.shipService.create(ship);
  }

  @Put(':id')
  update(@Param('id') id: number, @Body() ship: Partial<Ship>): Promise<Ship> {
    return this.shipService.update(id, ship);
  }

  @Delete(':id')
  remove(@Param('id') id: number): Promise<void> {
    return this.shipService.remove(id);
  }
}

```

- GET /ships

Отримання списку всіх суден.

- GET /ships/:id

Отримання інформації про конкретне судно.

- POST /ships

Додавання нового судна.

Тіло запиту:


```
{
  "Name": "Titanic",
  "Type": "Passenger",
  "Year_Built": 1912
}
```


- PUT /ships/:id

Оновлення інформації про судно.

Тіло запиту:


```
{
  "Name": "New Titanic",
  "Type": "Cargo",
  "Year_Built": 1920
}
```


- DELETE /ships/:id

Видалення судна.

2. Engine (Двигуни)

```
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { Engine } from './engine.entity';

@Injectable()
export class EngineService {
  constructor(
    @InjectRepository(Engine)
    private engineRepository: Repository<Engine>,
  ) {}

  findAll(): Promise<Engine[]> {
    return this.engineRepository.find({ relations: ['Ship_ID'] });
  }

  findOne(id: number): Promise<Engine> {
    return this.engineRepository.findOne({
      where: { Engine_ID: id },
      relations: ['Ship_ID'],
    });
  }

  create(engine: Partial<Engine>): Promise<Engine> {
    return this.engineRepository.save(engine);
  }

  async update(id: number, engine: Partial<Engine>): Promise<Engine> {
    await this.engineRepository.update(id, engine);
    return this.findOne(id);
  }

  async remove(id: number): Promise<void> {
    await this.engineRepository.delete(id);
  }
}
```

src > engine > engine.controller.ts > ...

```ts
import {
  Controller,
  Get,
  Post,
  Put,
  Delete,
  Param,
  Body,
} from '@nestjs/common';
import { EngineService } from './engine.service';
import { Engine } from './engine.entity';

@Controller('engines')
export class EngineController {
  constructor(private readonly engineService: EngineService) {}

  @Get()
  findAll(): Promise<Engine[]> {
    return this.engineService.findAll();
  }

  @Get(':id')
  findOne(@Param('id') id: number): Promise<Engine> {
    return this.engineService.findOne(id);
  }

  @Post()
  create(@Body() engine: Partial<Engine>): Promise<Engine> {
    return this.engineService.create(engine);
  }

  @Put(':id')
  update(
    @Param('id') id: number,
    @Body() engine: Partial<Engine>,
  ): Promise<Engine> {
    return this.engineService.update(id, engine);
  }

  @Delete(':id')
  remove(@Param('id') id: number): Promise<void> {
    return this.engineService.remove(id);
  }
}
```

- GET /engines

Отримання списку всіх двигунів.

- GET /engines/:id

Отримання інформації про конкретний двигун.

- POST /engines

Додавання нового двигуна.

Тіло запиту:


```
{
  "Ship_ID": 1,
  "Type": "Diesel",
  "Power": 2500,
  "Model": "MTU-4000"
}
```


- PUT /engines/:id

Оновлення інформації про двигун.

- DELETE /engines/:id

Видалення двигуна.

3. Reading (Зчитування параметрів)

```
import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { Reading } from './reading.entity';

@Injectable()
export class ReadingService {
  constructor(
    @InjectRepository(Reading)
    private readonly readingRepository: Repository<Reading>,
  ) {}

  findAll(): Promise<Reading[]> {
    return this.readingRepository.find({ relations: ['Engine'] });
  }

  findOne(id: number): Promise<Reading> {
    return this.readingRepository.findOne({
      where: { Reading_ID: id },
      relations: ['Engine'],
    });
  }

  create(reading: Partial<Reading>): Promise<Reading> {
    return this.readingRepository.save(reading);
  }

  async remove(id: number): Promise<void> {
    await this.readingRepository.delete(id);
  }
}
```

src > reading > reading.controller.ts > ...

```ts
import { Controller, Get, Post, Delete, Param, Body } from '@nestjs/common';
import { ReadingService } from './reading.service';
import { Reading } from './reading.entity';

@Controller('readings')
export class ReadingController {
  constructor(private readonly readingService: ReadingService) {}

  @Get()
  findAll(): Promise<Reading[]> {
    return this.readingService.findAll();
  }

  @Get(':id')
  findOne(@Param('id') id: number): Promise<Reading> {
    return this.readingService.findOne(id);
  }

  @Post()
  create(@Body() reading: Partial<Reading>): Promise<Reading> {
    return this.readingService.create(reading);
  }

  @Delete(':id')
  remove(@Param('id') id: number): Promise<void> {
    return this.readingService.remove(id);
  }
}

```

- GET /readings

Отримання списку всіх зчитувань.

- GET /readings/:id

Отримання інформації про конкретне зчитування.

- POST /readings

Додавання нового зчитування.

Тіло запиту:

{
"Engine_ID": 1,
"Temperature": 85.2,
"Cylinder_Pressure": 210.5,

```
  "Oil_Pressure": 5.3,
  "Boost_Pressure": 2.0,
  "RPM": 1500,
  "Exhaust_Temperature": 500,
  "Fuel_Consumption": 12.5,
  "Coolant_Temperature": 65.0,
  "Vibration": 0.02
}
```


- DELETE /readings/:id

Видалення зчитування.

4. EventType (Типи подій)

src > eventtype > eventtype.controller.ts > EventTypeController > create

```ts
import { Controller, Get, Post, Body } from '@nestjs/common';
import { EventTypeService } from './eventtype.service';
import { EventType } from './eventtype.entity';

@Controller('eventtypes')
export class EventTypeController {
  constructor(private readonly eventTypeService: EventTypeService) {}

  @Get()
  findAll(): Promise<EventType[]> {
    return this.eventTypeService.findAll();
  }

  @Post()
  create(@Body() eventType: Partial<EventType>): Promise<EventType> {
    return this.eventTypeService.create(eventType);
  }
}
```

- GET /eventtypes

Отримання списку типів подій.

- POST /eventtypes

Додавання нового типу події.

Тіло запиту:

```
{
  "Name": "Аварія"
}
```

5. Event (Події)

```
~/Work/ships/src/event • Contains emphasized items estjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { Event } from './event.entity';

@Injectable()
export class EventService {
  constructor(
    @InjectRepository(Event)
    private readonly eventRepository: Repository<Event>,
  ) {}

  findAll(): Promise<Event[]> {
    return this.eventRepository.find({ relations: ['Engine', 'EventType'] });
  }

  create(event: Partial<Event>): Promise<Event> {
    return this.eventRepository.save(event);
  }
}
```

```
src > event > event.controller.ts > EventController > findAll
1  import { Controller, Get, Post, Body } from '@nestjs/common';
2  import { EventService } from './event.service';
3  import { Event } from './event.entity';
4
5  @Controller('events')
6  export class EventController {
7    constructor(private readonly eventService: EventService) {}
8
9    @Get()
10   findAll(): Promise<Event[]> {
11     return this.eventService.findAll();
12   }
13
14   @Post()
15   create(@Body() event: Partial<Event>): Promise<Event> {
16     return this.eventService.create(event);
17   }
18 }
19
```

- GET /events

Отримання списку всіх подій.

- POST /events

Додавання нової події.

Тіло запиту:

```
{
  "Engine_ID": 1,
  "EventType_ID": 2,
  "Description": "Планове обслуговування",
  "Timestamp": "2024-11-09T14:00:00Z"
}
```


Особливості використання

Реалізований REST API дозволяє ефективно управляти даними, забезпечуючи можливість інтеграції з будь-яким клієнтським додатком (наприклад, веб-інтерфейсом чи мобільним додатком). Проєкт готовий до подальшого розширення функціоналу, включаючи додавання нових маршрутів або інтеграцію з іншими системами.

Цей API є важливим доповненням до курсової роботи, демонструючи практичну реалізацію серверної частини системи моніторингу технічного стану суднових двигунів.

UI для нашого сервісу виглядає так:

Опис HTML-сторінки зі списком івентів

Розроблена HTML-сторінка зі списком івентів є прикладом клієнтського інтерфейсу, який відображає події, що стосуються технічного стану суднових двигунів. Основна функція сторінки полягає у візуалізації даних про івенти в зручному для користувача вигляді.

Призначення сторінки

Сторінка дозволяє відобразити список івентів (подій), які були згенеровані в системі моніторингу суднових двигунів. Кожна подія містить основну інформацію про технічний стан двигуна або виконану дію (наприклад, обслуговування, попередження або аварійний стан).

Дані івентів подаються у структурованому вигляді, що спрощує їх аналіз і дає можливість користувачу швидко отримати уявлення про ситуацію.

Що відображає сторінка

HTML-сторінка показує такі ключові дані для кожного івенту:

1. Опис події (Description):

Короткий текст, що пояснює, що саме сталося (наприклад, “Engine maintenance completed” або “Temperature warning”).

2. Ідентифікатор двигуна (Engine ID):

Унікальний номер двигуна, до якого прив’язана подія. Це дозволяє ідентифікувати конкретний двигун на судні.

3. Тип події (Event Type ID):

Код типу події, що класифікує її (наприклад, попередження, обслуговування або несправність).

4. Час події (Timestamp):

Дата та час, коли подія була зафіксована у системі.

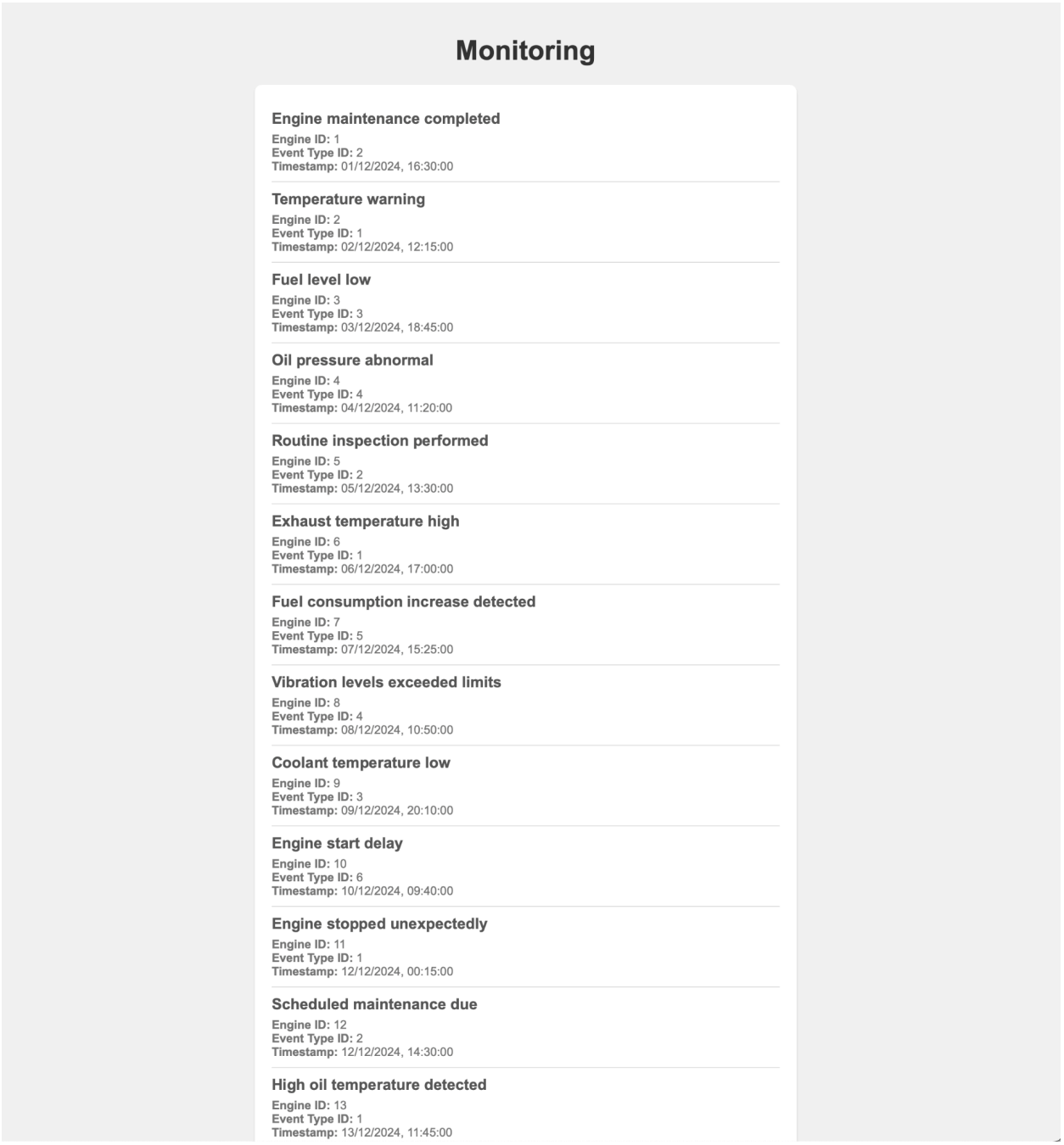
Особливості відображення

1. Структурований список:

Події відображаються одна за одною у вигляді окремих блоків. Кожен блок містить заголовок події (опис) і додаткові деталі.

2. Візуальне оформлення:

- Використано мінімалістичний дизайн із приємними кольорами, щоб зробити інтерфейс зрозумілим і не перевантаженим.
- Кожна подія відокремлена горизонтальною лінією для кращого візуального сприйняття.


Monitoring
Engine maintenance completed
Engine ID: 1
Event Type ID: 2
Timestamp: 01/12/2024, 16:30:00
Temperature warning
Engine ID: 2
Event Type ID: 1
Timestamp: 02/12/2024, 12:15:00
Fuel level low
Engine ID: 3
Event Type ID: 3
Timestamp: 03/12/2024, 18:45:00
Oil pressure abnormal
Engine ID: 4
Event Type ID: 4
Timestamp: 04/12/2024, 11:20:00
Routine inspection performed
Engine ID: 5
Event Type ID: 2
Timestamp: 05/12/2024, 13:30:00
Exhaust temperature high
Engine ID: 6
Event Type ID: 1
Timestamp: 06/12/2024, 17:00:00
Fuel consumption increase detected
Engine ID: 7
Event Type ID: 5
Timestamp: 07/12/2024, 15:25:00
Vibration levels exceeded limits
Engine ID: 8
Event Type ID: 4
Timestamp: 08/12/2024, 10:50:00
Coolant temperature low
Engine ID: 9
Event Type ID: 3
Timestamp: 09/12/2024, 20:10:00
Engine start delay
Engine ID: 10
Event Type ID: 6
Timestamp: 10/12/2024, 09:40:00
Engine stopped unexpectedly
Engine ID: 11
Event Type ID: 1
Timestamp: 12/12/2024, 00:15:00
Scheduled maintenance due
Engine ID: 12
Event Type ID: 2
Timestamp: 12/12/2024, 14:30:00
High oil temperature detected
Engine ID: 13
Event Type ID: 1
Timestamp: 13/12/2024, 11:45:00

Розроблена HTML-сторінка зі списком кораблів є прикладом клієнтського інтерфейсу, який відображає інформацію про судна, що зберігаються у системі. Основна функція сторінки полягає у візуалізації даних про судна у структурованому та зрозумілому для користувача вигляді.

Призначення сторінки

Сторінка дозволяє відобразити список суден із основними характеристиками. Це корисно для навігації по базі даних суден, їх аналізу та ідентифікації.

Що відображає сторінка

HTML-сторінка показує такі ключові дані для кожного судна:

1. Назва судна (Name):

Ім’я корабля, що є його основним ідентифікатором для користувача (наприклад, “Titanic”, “Queen Mary”).

2. Тип судна (Type):

Короткий опис типу судна (наприклад, “Passenger”, “Cruise”, “Cargo”, “Warship”).

3. Рік побудови (Year Built):

Рік, коли було збудовано судно. Це важлива характеристика для аналізу віку та стану корабля.

4. Ідентифікатор судна (Ship ID):

Унікальний номер судна, який використовується для ідентифікації в базі даних.

Особливості відображення

1. Структурований список:

Кожне судно відображається у вигляді окремого блоку з ключовими характеристиками. Блоки відділені один від одного для кращої читабельності.

2. Візуальне оформлення:

- Застосовано мінімалістичний та зрозумілий стиль для легкого сприйняття інформації.
- Кожен запис оформлено як окремий контейнер із заголовком (назвою судна) та деталями, такими як тип, рік побудови та ідентифікатор.
- Горизонтальні лінії використовуються для розділення записів.

Ships List

Titanic
Type: Passenger
Year Built: 1912
Ship ID: 1

Queen Mary
Type: Cruise
Year Built: 1936
Ship ID: 2

Bismarck
Type: Battleship
Year Built: 1940
Ship ID: 3

USS Enterprise
Type: Aircraft Carrier
Year Built: 1960
Ship ID: 4

Ever Given
Type: Cargo
Year Built: 2018
Ship ID: 5

Symphony of the Seas
Type: Cruise
Year Built: 2017
Ship ID: 6

HMS Victory
Type: Warship
Year Built: 1765
Ship ID: 7

USS Missouri
Type: Battleship
Year Built: 1944
Ship ID: 8

MS Costa Concordia
Type: Cruise
Year Built: 2005
Ship ID: 9

RMS Lusitania
Type: Passenger
Year Built: 1906
Ship ID: 10