

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## **ПОЯСНЮВАЛЬНА ЗАПИСКА**

до кваліфікаційної роботи освітнього ступеня «бакалавр»  
за спеціальністю 121 «Інженерія програмного забезпечення»  
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Вебзастосунок для діагностики несправностей  
обладнання за допомогою прогнозної аналітики»**

Виконав студент групи ВТ-22-2  
ФІЩУК Євгеній Олегович

Керівник роботи:  
ВЛАСЕНКО Олег Васильович

Рецензент:  
ФАНТ Микола Олександрович

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»  
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ  
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

**«ЗАТВЕРДЖУЮ»**

Зав. кафедри інженерії  
програмного забезпечення

Тетяна Вакалюк

**«28» лютого 2026 р.**

**ЗАВДАННЯ**  
**на кваліфікаційну роботу**

Здобувач вищої освіти: **ФІЩУК Євгеній Олегович**

Керівник роботи: **ВЛАСЕНКО Олег Васильович**

Тема роботи: **«Вебзастосунок для діагностики несправностей обладнання за допомогою прогнозної аналітики»**,

затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., №92/с**

Вихідні дані для роботи: **розробка та впровадження системи для діагностики несправностей обладнання за допомогою прогнозної аналітики**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

| <b>Розділ</b> | <b>Консультант</b> | <b>Завдання видав</b> | <b>Завдання прийняв</b> |
|---------------|--------------------|-----------------------|-------------------------|
| 1             | Власенко О.В.      | 17.02.2026            | 17.02.2026              |
| 2             | Власенко О.В.      | 19.02.2026            | 19.02.2026              |
| 3             | Власенко О.В.      | 21.02.2026            | 21.02.2026              |

## РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра містить 109 сторінок, 32 ілюстрацій, 8 таблиць, 6 додатків та 32 використані джерела.

Метою роботи є розробка вебзастосунок для діагностики несправностей промислового обладнання на основі методів прогнозної аналітики.

У процесі дослідження виконано теоретичний аналіз предметної області, огляд існуючих аналогів, обґрунтовано вибір архітектури та інструментарію. Було спроектовано об'єктно-орієнтовану модель системи, структури даних та математичні алгоритми на базі методів машинного навчання (Isolation Forest для виявлення аномалій, Random Forest для класифікації відмов, Gradient Boosting для прогнозування залишкового ресурсу). Реалізовано повнофункціональний вебзастосунок CASS із сучасним адаптивним інтерфейсом, модулями реального часу моніторингу, візуалізації та журналювання подій. Проведено комплексне тестування, яке підтвердило високу точність моделей (ROC-AUC понад 0,89, MAE прогнозу RUL у межах 40–71 години) та стабільність роботи під навантаженням.

Розроблений застосунок рекомендується до впровадження на промислових підприємствах для переходу до прогнозного технічного обслуговування, що дозволить зменшити позапланові простой обладнання, оптимізувати витрати на ремонти та підвищити загальну надійність виробничих процесів.

**КЛЮЧОВІ СЛОВА:** АНАЛІТИКА, ДІАГНОСТИКА, МАШИННЕ НАВЧАННЯ, ОБЛАДНАННЯ, ПРОГНОЗУВАННЯ

|           |      |               |        |      |                                                                                          |  |  |                  |      |         |     |  |
|-----------|------|---------------|--------|------|------------------------------------------------------------------------------------------|--|--|------------------|------|---------|-----|--|
|           |      |               |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ                                                                 |  |  |                  |      |         |     |  |
|           |      |               |        |      |                                                                                          |  |  |                  |      |         |     |  |
| Змн.      | Арк. | № докум.      | Підпис | Дата |                                                                                          |  |  |                  |      |         |     |  |
| Розроб.   |      | Фіщук Є.О.    |        |      | Вебзастосунок для діагностики несправностей обладнання за допомогою прогнозної аналітики |  |  | Літ.             | Арк. | Аркушів |     |  |
| Керівник  |      | Власенко О.В. |        |      |                                                                                          |  |  |                  |      | 3       | 109 |  |
|           |      |               |        |      |                                                                                          |  |  |                  |      |         |     |  |
| Н. контр. |      |               |        |      |                                                                                          |  |  | ФІКТ Гр. ВТ-22-2 |      |         |     |  |
| Зав. каф. |      | Вакалюк Т.А.  |        |      |                                                                                          |  |  |                  |      |         |     |  |

## ABSTRACT

The bachelor's final qualifying work contains of 109 pages, 32 illustrations, 8 tables, 6 appendices and 32 references.

The aim of the work is to develop a web application for diagnosing equipment faults using predictive analytics methods.

The research includes theoretical analysis of the domain, review of existing solutions, justification of architecture and tools. An object-oriented system model, data structures and mathematical algorithms based on machine learning methods (Isolation Forest for anomaly detection, Random Forest for failure classification, Gradient Boosting for remaining useful life prediction) were designed. The full-featured CASS web application with a modern adaptive interface, real-time monitoring, visualization and event logging modules was implemented. Comprehensive testing confirmed high model accuracy (ROC-AUC over 0.89, RUL prediction MAE within 40–71 hours) and operational stability under load.

The developed application is recommended for implementation at industrial enterprises to enable predictive maintenance strategies, which will reduce unplanned equipment downtime, optimize repair costs and increase overall production reliability.

**KEYWORDS:** ANALYTICS, DIAGNOSTICS, EQUIPMENT, MACHINE LEARNING, PREDICTION

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 4    |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

## ЗМІСТ

|                                                                                                       |    |
|-------------------------------------------------------------------------------------------------------|----|
| РЕФЕРАТ .....                                                                                         | 3  |
| ABSTRACT .....                                                                                        | 4  |
| ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ .....                                                                       | 7  |
| ВСТУП .....                                                                                           | 8  |
| РОЗДІЛ 1. ТЕОРЕТИЧНИЙ АНАЛІЗ ВИМОГ ДО ВЕБЗАСТОСУНКУ .....                                             | 11 |
| 1.1 Визначення задач що підлягають автоматизації у предметній області<br>діагностики обладнання ..... | 11 |
| 1.2 Аналіз існуючих аналогів систем прогностної діагностики несправностей                             | 14 |
| 1.3 Вибір архітектури та структури вебзастосунок для діагностики<br>несправностей .....               | 20 |
| 1.4 Обґрунтування вибору інструментальних засобів та вимог до апаратних<br>ресурсів .....             | 24 |
| Висновки до першого розділу .....                                                                     | 28 |
| РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ ДЛЯ ДІАГНОСТИКИ<br>НЕСПРАВНОСТЕЙ ОБЛАДНАННЯ .....                | 29 |
| 2.1 Визначення варіантів використання та об'єктно орієнтованої структури<br>системи .....             | 29 |
| 2.2 Розроблення структур даних та математичних алгоритмів прогностної<br>аналітики .....              | 38 |
| 2.3 Проєктування та опис компонентної будови вебзастосунок .....                                      | 45 |
| Висновки до другого розділу .....                                                                     | 52 |
| РОЗДІЛ 3. ОПИС ІНТЕРФЕЙСУ ТА ТЕСТУВАННЯ ВЕБЗАСТОСУНКУ .....                                           | 54 |
| 3.1 Порядок роботи системи та демонстрація сценаріїв діагностики .....                                | 54 |
| 3.2 Тестування реалізованих алгоритмів та оцінювання коректності роботи<br>вебзастосунок .....        | 60 |

|                                    |    |
|------------------------------------|----|
| Висновки до третього розділу ..... | 64 |
| ВИСНОВКИ.....                      | 65 |
| СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....    | 67 |
| ДОДАТКИ.....                       | 71 |

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 6    |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

CASS – Condition-based Anomaly & Signature System

RUL – Remaining Useful Life

IoT – Internet of Things

AI – Artificial Intelligence

API – Application Programming Interface

JSON – JavaScript Object Notation

WAL – Write-Ahead Logging

MAE – Mean Absolute Error

ROC-AUC – Receiver Operating Characteristic – Area Under Curve

REST – Representational State Transfer

ERP – Enterprise Resource Planning

MES – Manufacturing Execution System

SCADA – Supervisory Control and Data Acquisition

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 7    |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## ВСТУП

У сучасних умовах стрімкого розвитку промисловості, енергетики, транспорту та цифровізації виробництва питання забезпечення надійності та безперебійної роботи технічного обладнання набуває особливого стратегічного значення. Сучасні підприємства стикаються з жорсткою конкуренцією, високими вимогами до якості продукції та необхідністю мінімізації витрат. Традиційні підходи до технічного обслуговування, засновані переважно на планових оглядах та реактивному реагуванні на вже виниклі несправності, все частіше демонструють свою обмеженість. Такі методи призводять до невиправданих простоїв обладнання, зростання експлуатаційних витрат, зниження продуктивності та підвищення ризиків аварійних ситуацій, що може мати серйозні економічні та безпекові наслідки.

У цьому контексті перехід від планово-попереджувального обслуговування до прогнозної аналітики (predictive analytics) та condition-based maintenance (обслуговування за фактичним технічним станом) стає не лише перспективним напрямом розвитку, але й об'єктивною необхідністю для підвищення конкурентоспроможності підприємств. Автоматизовані системи, здатні в реальному часі збирати, обробляти та аналізувати великі обсяги телеметричних даних від датчиків, виявляти приховані аномалії, класифікувати можливі режими відмов та прогнозувати залишковий корисний ресурс (Remaining Useful Life – RUL) обладнання, дозволяють суттєво оптимізувати процеси утримання виробничих активів, зменшувати економічні втрати та підвищувати загальну безпеку виробництва.

Саме тому розробка сучасних вебзастосунків для діагностики несправностей обладнання за допомогою методів прогнозної аналітики є актуальною науково-практичною задачею, що відповідає пріоритетам цифрової трансформації промисловості України та світу в цілому, а також стратегії Industry 4.0.

Мета роботи полягає в розробці та впровадженні повнофункціонального вебзастосунку для діагностики несправностей промислового обладнання на

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 8    |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

основі методів прогнозної аналітики, який забезпечує оперативний моніторинг технічного стану, автоматичне виявлення аномалій, класифікацію можливих режимів відмов та точне прогнозування залишкового корисного ресурсу (RUL) обладнання.

Об'єктом дослідження є процеси діагностики технічного стану промислового обладнання в умовах сучасного виробництва.

Предметом дослідження виступають методи та засоби прогнозної аналітики, архітектурні рішення та програмна реалізація вебзастосунку для автоматизованої діагностики несправностей.

Для досягнення поставленої мети в роботі визначено такі основні завдання:

- проаналізувати предметну область діагностики обладнання, визначити ключові задачі, що підлягають автоматизації, та дослідити існуючі аналоги систем прогнозної діагностики;
- обґрунтувати вибір архітектури, структури даних та інструментальних засобів для розробки вебзастосунку;
- розробити математичні алгоритми та моделі прогнозної аналітики, включаючи виявлення аномалій, класифікацію відмов і прогнозування RUL;
- спроектувати та реалізувати компонентну будову системи, сучасний користувацький інтерфейс та інтеграційні механізми;
- провести комплексне тестування алгоритмів і вебзастосунку в цілому, оцінити їхню коректність, ефективність та стабільність роботи.

Наукова новизна роботи полягає в розробці обладнання-агностичної архітектури вебзастосунку на основі декларативних паспортів моделей техніки, комплексній інтеграції ансамблевих методів машинного навчання (Isolation Forest для виявлення аномалій, Random Forest для класифікації режимів відмов, Gradient Boosting для прогнозування RUL з квантильною регресією) та реалізації повнофункціональної системи CASS з підтримкою реального часу моніторингу.

Практичне значення одержаних результатів полягає у створенні готового до впровадження вебзастосунку CASS (Condition-based Anomaly & Signature System), який може бути ефективно використаний на промислових

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 9    |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

підприємствах для переходу до прогнозного технічного обслуговування. Система забезпечує обладнання-агностичний підхід завдяки декларативним паспортам моделей техніки, що дозволяє легко адаптувати її під різні типи агрегатів – від серверних стійок і насосів до кріогенних установок. Реалізовані алгоритми демонструють високу точність: ROC-AUC для виявлення аномалій перевищує 0,89, а середня абсолютна похибка (MAE) прогнозу RUL знаходиться в прийнятних межах (40–71 година) для практичного застосування. Адаптивний та інтуїтивно зрозумілий інтерфейс зручний для операторів і інженерів, що сприяє швидкому прийняттю обґрунтованих рішень. Впровадження розробки дозволить суттєво зменшити позапланові простой обладнання, оптимізувати витрати на ремонти та підвищити загальну надійність виробничих процесів. Крім того, отримані результати можуть слугувати науковою основою для подальших досліджень у сфері цифрових двійників обладнання та інтеграції з корпоративними системами управління (ERP, MES, SCADA).

У роботі використано комплекс сучасних методів наукового дослідження. Теоретичний аналіз базується на вивченні вітчизняних та зарубіжних літературних джерел, міжнародних стандартів і наукових публікацій з питань condition-based maintenance, predictive maintenance, машинного навчання та промислового Інтернету речей (IoT). При проєктуванні системи застосовувалися об’єктно-орієнтований підхід, методи моделювання, принципи мікросервісної та багатошарової архітектури. Розробка алгоритмів ґрунтується на методах інженерії ознак, ансамблевих моделях машинного навчання, квантильній регресії та оцінці невизначеності. Практична реалізація та тестування проводилися з використанням сучасного інструментарію: Node.js, Python, Vue.js, SQLite та бібліотек для візуалізації даних.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 10   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## РОЗДІЛ 1. ТЕОРЕТИЧНИЙ АНАЛІЗ ВИМОГ ДО ВЕБЗАСТОСУНКУ

### 1.1 Визначення задач що підлягають автоматизації у предметній області діагностики обладнання

У сучасних умовах розвитку промисловості та енергетики діагностика технічного стану обладнання відіграє ключову роль у забезпеченні надійності виробничих процесів, зменшенні простоїв та оптимізації витрат на утримання. Традиційно діагностика базувалася на планових оглядах, періодичному технічному обслуговуванні та візуальному контролю фахівцями, що часто призводило до неефективного використання ресурсів. Такі підходи не завжди вчасно виявляли приховані дефекти, оскільки ґрунтувалися переважно на суб'єктивній оцінці та статичних графіках робіт.

У результаті підприємства стикалися з позаплановими зупинками обладнання, підвищенням аварійності та значними економічними втратами. Автоматизація процесів діагностики дозволяє перейти до більш прогресивних стратегій, орієнтованих на реальний стан техніки, що відповідає вимогам міжнародних стандартів з моніторингу та прогнозування [1].

Однією з головних задач, що підлягає автоматизації, є безперервний збір та обробка даних про параметри роботи обладнання. У промисловості обладнання генерує великі обсяги інформації від різних датчиків, таких як температура, вібрація, тиск, електричні параметри та інші характеристики. Ручна обробка цих даних є трудомісткою та схильною до помилок. Автоматизовані системи забезпечують реальний часовий моніторинг, що дає змогу оперативно фіксувати відхилення від нормальних режимів експлуатації. Це особливо важливо для складних агрегатів, де навіть незначні зміни можуть сигналізувати про початок деградації. Завдяки автоматизації збір даних стає систематичним і точним, що створює надійну основу для подальшого аналізу [2].

Наступною важливою задачею є виявлення аномалій у роботі обладнання. Традиційні методи часто обмежувалися порівнянням поточних показників з фіксованими пороговими значеннями, що не враховувало контекст роботи чи

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 11   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

історичні тенденції. Автоматизація дозволяє застосовувати складні алгоритми для аналізу багатовимірних даних, ідентифікуючи неочевидні відхилення, які можуть вказувати на потенційні проблеми. Такий підхід значно підвищує чутливість діагностики, зменшуючи кількість хибних спрацьовувань і пропущених загроз. У результаті підприємства отримують можливість попереджати розвиток несправностей на ранніх стадіях, що сприяє продовженню терміну служби техніки [3].

Важливим напрямом автоматизації є прогнозування залишкового ресурсу обладнання. Визначення того, скільки часу техніка ще може працювати без ризику виходу з ладу, є складним завданням, що вимагає врахування багатьох факторів, включаючи умови експлуатації, історію навантажень та статистичні моделі зношування. Автоматизовані рішення на основі статистичних методів і машинного навчання дозволяють будувати прогнози з урахуванням довірчих інтервалів, що допомагає планувати технічне обслуговування більш ефективно. Це особливо актуально для критичного обладнання, де несподіваний вихід з ладу може призвести до значних наслідків для всього виробничого ланцюга [4].

Автоматизація також охоплює класифікацію режимів можливих відмов. Обладнання різних типів може мати характерні сценарії деградації, пов'язані з механічним зносом, тепловими перевантаженнями, електричними проблемами чи іншими факторами. Сучасні системи здатні автоматично визначати ймовірний тип несправності на основі комбінації параметрів, що полегшує прийняття рішень щодо ремонту. Такий підхід зменшує залежність від досвіду окремих спеціалістів і стандартизує процес діагностики на підприємстві [5].

Для наочності процесу діагностики обладнання в автоматизованому середовищі доцільно розглянути загальну схему взаємодії основних етапів, що наведена на рисунку 1.1. На схемі відображено послідовність обробки даних, починаючи зі збору показників від обладнання та закінчуючи формуванням результатів діагностики і рекомендацій щодо технічного обслуговування. Такий підхід дає змогу простежити інформаційні потоки між окремими компонентами системи, визначити роль кожного етапу в процесі аналізу технічного стану

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 12   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

обладнання та демонструє принцип функціонування системи прогнозної діагностики в цілому.



Рисунок 1.1 – Загальна схема автоматизованого процесу діагностики технічного стану обладнання

Схема ілюструє послідовність етапів від первинного збору даних до формування рекомендацій, підкреслюючи циклічний характер моніторингу. Така структура забезпечує безперервність аналізу та оперативність реагування [6]. Таблиця 1.1 демонструє, як автоматизація перетворює процес діагностики, роблячи його більш ефективним і економічним [7].

Таблиця 1.1

Порівняння традиційного та автоматизованого підходів до діагностики обладнання

| Аспект                    | Традиційний підхід                         | Автоматизований підхід                        |
|---------------------------|--------------------------------------------|-----------------------------------------------|
| Частота моніторингу       | Періодична, за графіком                    | Безперервна, у реальному часі                 |
| Точність виявлення        | Залежить від суб'єктивної оцінки           | Висока, на основі алгоритмів аналізу          |
| Прогнозування             | Обмежене, на основі досвіду                | Науково обґрунтоване, з довірчими інтервалами |
| Реакція на зміни          | Запізніла                                  | Оперативна                                    |
| Витрати на обслуговування | Високі через надмірні чи недостатні роботи | Оптимізовані, за реальним станом              |

Окрім основних задач, автоматизація включає підтримку прийняття рішень через інтеграцію результатів аналізу в єдину інформаційну систему. Це

дозволяє фахівцям отримувати комплексні звіти, візуалізації тенденцій та рекомендації щодо пріоритетів ремонту. Такий підхід сприяє зменшенню людського фактора та підвищенню загальної культури технічного обслуговування на підприємствах. У контексті сучасних викликів, пов'язаних з енергетичною безпекою та цифровою трансформацією промисловості, автоматизована діагностика стає невід'ємною частиною стратегії підвищення надійності обладнання [8].

Особливу увагу слід приділити задачам, пов'язаним з обробкою даних від різнотипного обладнання. Промислові об'єкти часто включають агрегати різних класів – від насосів і компресорів до серверних систем та спеціалізованих установок. Автоматизація дозволяє створювати уніфіковані моделі опису техніки, що полегшує масштабування рішень. Це відкриває можливості для прогностичного технічного обслуговування, орієнтованого на умови, а не на календар.

Узагальнюючи, визначення задач автоматизації у сфері діагностики обладнання охоплює широкий спектр процесів, починаючи від збору даних і закінчуючи прогнозуванням та рекомендаціями. Перехід до автоматизованих систем дозволяє не лише вирішувати поточні проблеми, але й формувати основу для стійкого розвитку промисловості, зменшуючи ризики та оптимізуючи ресурси. Такий розвиток відповідає сучасним тенденціям інтеграції інформаційних технологій у виробничу сферу та сприяє підвищенню конкурентоспроможності підприємств.

## **1.2 Аналіз існуючих аналогів систем прогностичної діагностики несправностей**

Сучасні промислові підприємства функціонують в умовах високої інтенсивності виробничих процесів, де навіть короткочасні простої обладнання можуть призводити до значних економічних втрат. У зв'язку з цим традиційні підходи до технічного обслуговування, що базуються на регламентних перевірках або реагуванні на вже виниклі несправності, поступово поступаються

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 14   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

місцем інтелектуальним системам прогнозової діагностики. Такі системи використовують дані, отримані від датчиків та засобів моніторингу, для оцінювання технічного стану обладнання в режимі реального часу, що дозволяє завчасно виявляти потенційні відмови та планувати необхідні профілактичні заходи.

Застосування прогнозової діагностики сприяє зменшенню кількості аварійних зупинок, оптимізації витрат на технічне обслуговування, підвищенню рівня безпеки виробництва та збільшенню загальної ефективності використання виробничих ресурсів. Розвиток технологій Інтернету речей (IoT), хмарних обчислень, великих даних та методів машинного навчання став основою для створення сучасних платформ моніторингу й прогнозування технічного стану промислового обладнання.

Одним із ключових напрямів розвитку таких рішень є інтеграція систем збору телеметричних даних із засобами інтелектуального аналізу інформації. Більшість сучасних платформ використовують масштабовані архітектури, які поєднують локальну обробку даних безпосередньо на виробничих майданчиках із можливостями хмарної аналітики. Для оцінювання існуючих підходів до реалізації прогнозової діагностики доцільно розглянути найбільш поширені програмні комплекси, представлені на ринку.

Першим аналогом є програмна платформа IBM Maximo Application Suite (рисунк 1.2), призначена для комплексного управління виробничими активами підприємства. Система забезпечує безперервний моніторинг технічного стану обладнання на основі даних, що надходять від різноманітних датчиків та систем автоматизації. Використовуючи алгоритми машинного навчання та засоби штучного інтелекту, платформа виконує аналіз телеметричної інформації, виявляє аномальні режими роботи, визначає потенційні причини несправностей і прогнозує залишковий корисний ресурс обладнання (Remaining Useful Life, RUL) із урахуванням довірчих інтервалів.

Крім функцій прогнозування, IBM Maximo підтримує автоматичне формування рекомендацій щодо проведення технічного обслуговування,

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 15   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

планування ремонтних робіт та управління життєвим циклом активів. Важливою перевагою системи є можливість інтеграції з корпоративними ERP-, MES- та SCADA-системами, що забезпечує централізоване управління виробничими процесами. Завдяки комплексному підходу до моніторингу та аналізу стану обладнання платформа дозволяє суттєво скоротити кількість позапланових простоїв, підвищити надійність виробництва та оптимізувати експлуатаційні витрати підприємства [9].

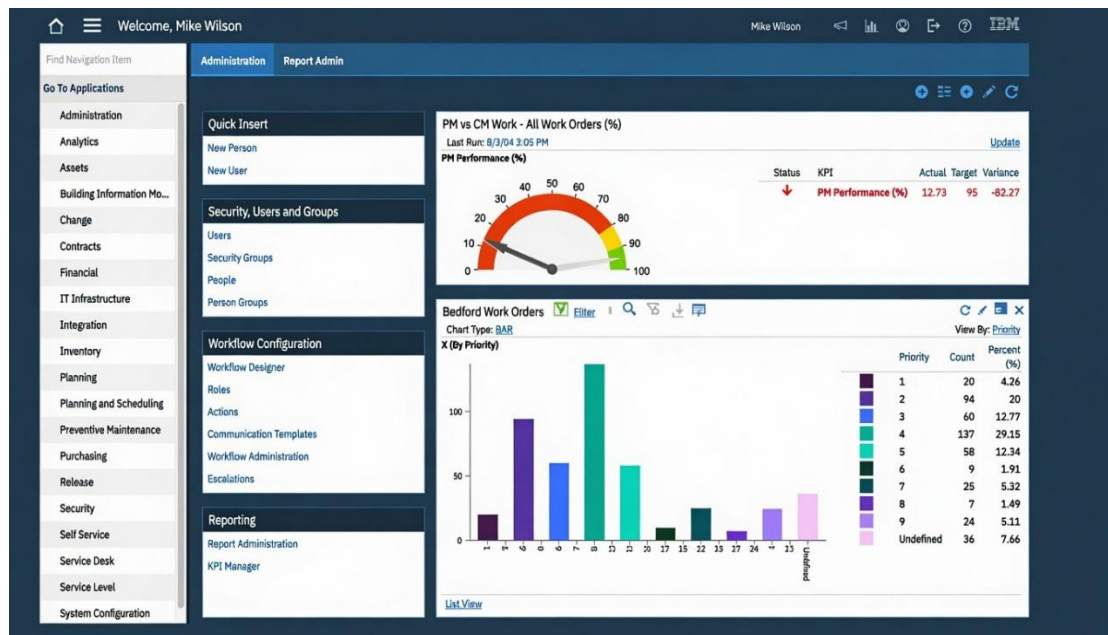


Рисунок 1.2 – Інтерфейс IBM Maximo Application Suite

Другим важливим прикладом є Siemens MindSphere (рисунок 1.3). Це хмарна IoT-платформа, орієнтована на промислове застосування. Основні функціональні можливості включають збір і попередню обробку даних на рівні edge-обчислень, аналіз вібраційних, температурних та інших параметрів, прогнозування деградації обладнання та візуалізацію стану в зручних інформаційних панелях (дашбордах). Платформа особливо ефективна для енергетичного та виробничого обладнання, підтримує створення цифрових двійників і дозволяє оперативно реагувати на відхилення від нормальних режимів роботи. Її сильною стороною є висока масштабованість та інтеграція з різноманітним промисловим обладнанням [10].

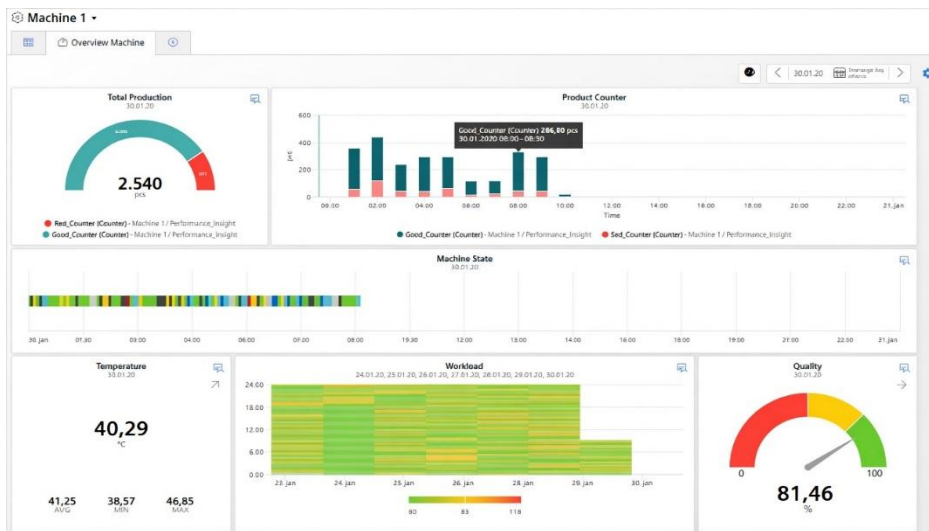


Рисунок 1.3 – Інтерфейс Siemens MindSphere

Третім розглянутим аналогом є Augury (рисунок 1.4) – спеціалізоване рішення для моніторингу обертового обладнання. Система фокусується на акустичному та вібраційному аналізі, автоматично виявляє ранні ознаки зносу підшипників, дисбалансу ротора та інших типових несправностей. Функціонал включає класифікацію типів дефектів, розрахунок ймовірності відмови в найближчій перспективі та надання конкретних рекомендацій обслуговуючому персоналу. Augury вирізняється високою точністю в умовах реального виробництва та відносно простою інтеграцією з існуючими сенсорними системами [11].

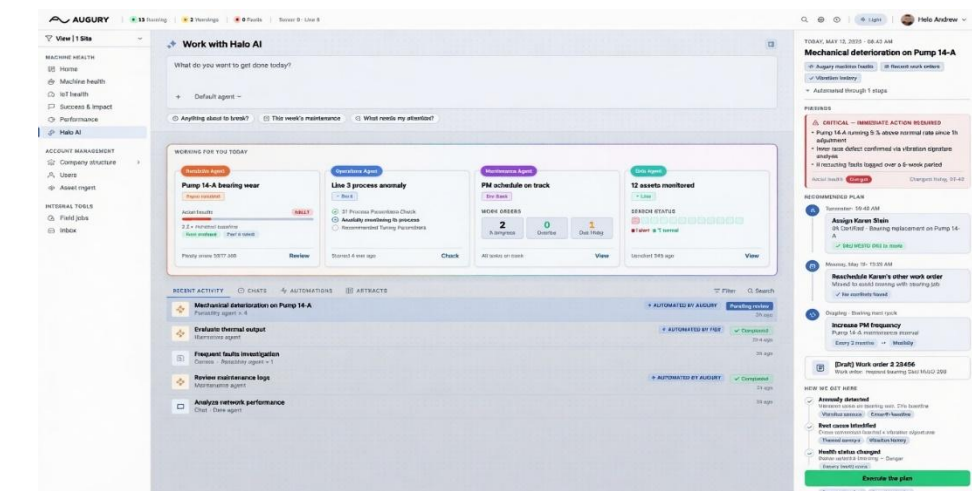


Рисунок 1.4 – Інтерфейс Augury

Порівнюючи функціонал розглянутих аналогів з перспективами розвитку подібних вебзастосунків, можна відзначити, що майбутні рішення, такі як ті, що відповідають сучасним вимогам, повинні поєднувати в собі сильні сторони

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 17   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

існуючих платформ. Зокрема, вони мають забезпечувати обладнання – агностичний підхід (можливість опису будь-якого типу техніки через декларативні моделі), моніторинг у реальному часі, виявлення аномалій, класифікацію режимів відмов, прогноз залишкового ресурсу з довірчими інтервалами та зручну візуалізацію для користувачів.

Такий комплексний функціонал дозволить не лише повторювати можливості провідних аналогів, але й адаптувати їх під специфічні потреби українських підприємств, забезпечуючи вищу гнучкість та незалежність від зовнішніх хмарних сервісів.

Аналіз ринку показує, що більшість сучасних платформ орієнтована на використання методів машинного навчання для класифікації станів техніки та оцінки ймовірності відмов. У літературі підкреслюється, що ефективність таких систем значною мірою залежить від якості вхідних даних, зокрема від точності сенсорів і частоти їх опитування.

Дослідження 2022–2025 років свідчать про зростання популярності гібридних підходів, які поєднують статистичні моделі з нейронними мережами для підвищення точності прогнозів у складних динамічних середовищах [12].

Для наочності порівняння ключових характеристик провідних аналогів доцільно звернутися до узагальненої таблиці 1.2, яка відображає основні аспекти функціональності.

Таблиця 1.2

Порівняльна характеристика комерційних платформ прогновної діагностики несправностей

| Платформа                   | IBM Maximo                     | Siemens MindSphere            | Augury                           | CASS                                               |
|-----------------------------|--------------------------------|-------------------------------|----------------------------------|----------------------------------------------------|
| 1                           | 2                              | 3                             | 4                                | 5                                                  |
| Основні технології аналізу  | Машинне навчання, AI-аналітика | Edge computing, IoT-платформа | Акустичний та вібраційний аналіз | Isolation Forest, Random Forest, Gradient Boosting |
| Підтримка реального часу    | Висока                         | Висока                        | Висока                           | Висока                                             |
| Прогноз залишкового ресурсу | Так, з довірчими інтервалами   | Так                           | Обмежена                         | Так, з 90% довірчим інтервалом                     |

| 1                    | 2            | 3                       | 4                   | 5                                          |
|----------------------|--------------|-------------------------|---------------------|--------------------------------------------|
| Гнучкість інтеграції | Висока       | Середня-висока          | Середня             | Висока (обладнання-агностична архітектура) |
| Галузева специфіка   | Універсальна | Виробництво, енергетика | Обертове обладнання | Універсальна (будь-яке обладнання)         |

Ця таблиця ілюструє, що більшість систем забезпечує високий рівень моніторингу в реальному часі, але відрізняється за глибиною прогнозних можливостей і легкістю впровадження. Системи з акцентом на edge-обчислення, як у Siemens, краще підходять для об'єктів з високими вимогами до швидкості реакції, тоді як універсальні рішення на кшталт IBM пропонують ширші можливості кастомізації.

Архітектурно більшість аналогів базується на багатоваршівній структурі, що включає рівні збору даних, їх попередньої обробки, аналітики та візуалізації. Для кращого розуміння типової структури таких систем можна розглянути узагальнену схему, наведену на рисунку 1.5.



Рисунок 1.5 – Узагальнена архітектура систем прогнозної діагностики несправностей

Як видно з рисунку 1.5, ключовим елементом є рівень аналітики, де відбувається основна обробка інформації за допомогою статистичних та інтелектуальних методів. Це дозволяє системам не лише реагувати на поточні відхилення, але й прогнозувати майбутній стан обладнання, що є суттєвою перевагою порівняно з традиційними підходами.

Окрім комерційних платформ, у науковій літературі описано дослідницькі та відкрито доступні рішення, які часто слугують основою для подальшої розробки. Такі системи, як правило, пропонують модульну структуру, що полегшує адаптацію під конкретні типи обладнання. Дослідження 2023–2025 років підкреслюють важливість стандартизації протоколів обміну даними для забезпечення сумісності різних компонентів. Це особливо актуально для підприємств, які експлуатують обладнання різного віку та виробників [12].

Аналіз аналогів свідчить про те, що успішні реалізації характеризуються поєднанням надійної апаратної інфраструктури з потужними програмними алгоритмами. Водночас залишаються виклики, пов'язані з якістю даних, інтеграцією в існуючі виробничі процеси та кваліфікацією персоналу. Подальший розвиток цих систем йтиме шляхом посилення штучного інтелекту, розширення можливостей цифрових двійників і підвищення рівня автоматизації рекомендацій. Таким чином, вивчення існуючих платформ дозволяє виділити найкращі практики для створення ефективних рішень у сфері прогностної діагностики, що відповідають реальним потребам промисловості.

### 1.3 Вибір архітектури та структури вебзастосунку для діагностики несправностей

Вибір архітектури вебзастосунку для діагностики несправностей обладнання на основі прогностної аналітики є визначальним етапом, оскільки така система повинна забезпечувати безперервний збір даних з датчиків, оперативну обробку великих обсягів інформації, точне прогнозування залишкового ресурсу та своєчасне попередження про можливі відмови.

Архітектура має поєднувати гнучкість, масштабованість і надійність, адже обладнання працює в реальному часі, а затримки в аналізі можуть призвести до значних економічних втрат або навіть аварій. У сучасних умовах, коли промислові системи генерують терабайти телеметрії щодня, пріоритетними стають модульність і можливість незалежного розвитку окремих компонентів [13].

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 20   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Основним підходом до побудови таких систем є клієнт-серверна архітектура з елементами мікросервісного стилю. Вона дозволяє розділити обов'язки між інтерфейсом користувача, серверною логікою обробки даних і спеціалізованими сервісами машинного навчання. Клієнтська частина забезпечує візуалізацію стану обладнання, графіків тенденцій і сповіщень для операторів, тоді як серверна сторона відповідає за збереження даних, виконання складних обчислень і інтеграцію з зовнішніми джерелами.

Такий розподіл сприяє незалежному масштабуванню компонентів: наприклад, при зростанні кількості підключених пристроїв можна збільшити ресурси лише для частини, відповідальної за прийом телеметрії, без впливу на інші модулі [14].

Важливим аспектом є вибір між монолітною та мікросервісною архітектурою. Монолітна структура, де всі компоненти перебувають в одному додатку, спрощує початкову розробку та розгортання, але з часом ускладнює підтримку та оновлення, особливо коли потрібно вдосконалювати алгоритми прогнозування незалежно від інтерфейсу.

Мікросервісний підхід, навпаки, передбачає розбиття системи на невеликі, самостійно функціонуючі сервіси, кожен з яких вирішує конкретне завдання: прийом даних, зберігання, аналіз аномалій чи генерація прогнозів. Це забезпечує технологічну незалежність – один сервіс може бути реалізований на одній мові програмування, інший – на іншій, залежно від оптимальності для завдання. Для систем діагностики несправностей мікросервіси дозволяють швидко оновлювати моделі прогнозування залишкового ресурсу, не зупиняючи роботу всього застосунку [15].

Узагальнена структура вебзастосунку зазвичай включає кілька основних шарів. Шар представлення (клієнтський інтерфейс) забезпечує інтерактивну взаємодію з користувачем через сучасні веб-технології, дозволяючи переглядати дашборди, детальні графіки сигналів і звіти про стан обладнання. Бізнес-шар містить логіку обробки даних, правила класифікації режимів роботи та

механізми прийняття рішень. Шар даних відповідає за надійне зберігання телеметрії, паспортів обладнання та історичних подій.

Окремо виділяється аналітичний шар, де відбуваються обчислення аномалій, класифікація можливих відмов і прогнозування за допомогою методів машинного навчання. Така багатошарова організація сприяє чіткому розмежуванню відповідальності та полегшує тестування [16].

Для забезпечення реального часу в системах діагностики важливо передбачити асинхронну обробку даних і використання черг повідомлень. Це дозволяє приймати потік телеметрії без блокування основних процесів і негайно передавати критичні сповіщення. Крім того, архітектура повинна підтримувати контейнеризацію, що спрощує розгортання в хмарних або локальних середовищах і забезпечує відтворюваність налаштувань. Узагальнена схема архітектури вебзастосунку для діагностики несправностей обладнання представлена на рисунку 1.6.

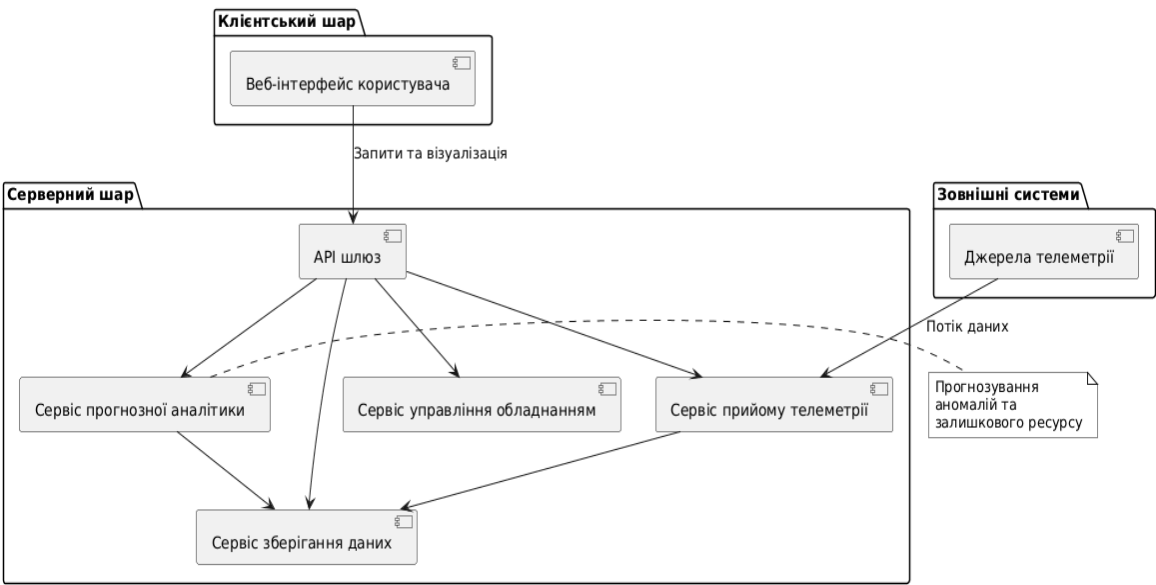


Рисунок 1.6 – Схема архітектури вебзастосунку для діагностики несправностей обладнання

Як видно з наведеної схеми, потік даних починається від джерел телеметрії, проходить через сервіс прийому, зберігається та аналізується в спеціалізованому модулі, після чого результати стають доступними через інтерфейс користувача. Така структура забезпечує високу відмовостійкість: вихід з ладу одного сервісу не паралізує всю систему [17].

При виборі конкретної архітектури важливо враховувати вимоги до продуктивності, безпеки та інтеграції. Для промислових застосунків рекомендують використовувати багат шарову архітектуру з чітким розділенням на presentation layer, application layer, domain layer та infrastructure layer. Це дозволяє ізолювати бізнес-логіку від технічних деталей реалізації, що особливо корисно при роботі з різними типами обладнання, кожен з яких має власні параметри та порогові значення.

Порівняльна характеристика архітектурних підходів для систем прогнозної діагностики наведена в таблиці 1.3.

Таблиця 1.3

Порівняльна характеристика архітектурних підходів для систем прогнозної діагностики

| Архітектурний підхід | Переваги                                                     | Недоліки                                         | Застосування в діагностиці                                                    |
|----------------------|--------------------------------------------------------------|--------------------------------------------------|-------------------------------------------------------------------------------|
| Монолітна            | Простота розробки, швидке розгортання                        | Складність масштабування, залежність компонентів | Маленькі системи з обмеженою кількістю обладнання                             |
| Мікросервісна        | Масштабованість, незалежне оновлення, технологічна гнучкість | Складність оркестрації, overhead на комунікацію  | Великі системи з високим навантаженням і потребою в частому оновленні моделей |
| Багат шарова         | Чітке розмежування, легкість тестування                      | Додаткові витрати на інтеграцію шарів            | Системи з складною бізнес-логікою прогнозування                               |

Вибір на користь мікросервісної архітектури з елементами багат шаровості дозволяє досягти оптимального балансу між швидкістю розробки та довгостроковою підтримкою. Особливо це актуально для систем, де прогнозна аналітика постійно вдосконалюється: нові алгоритми можна впроваджувати в ізольованому сервісі без ризику для стабільності всього застосунку. Крім того, така структура полегшує забезпечення безпеки даних, оскільки критичні операції аналізу можна проводити в захищеному середовищі з обмеженим доступом [18].

Не менш важливим є питання оркестрації та моніторингу. Сучасні системи передбачають використання інструментів автоматичного розгортання, які дозволяють динамічно керувати ресурсами залежно від навантаження. Це особливо цінно для підприємств, де кількість підключеного обладнання може

значно коливатися. Структура повинна також включати механізми резервного копіювання та відновлення даних, адже втрата історичної телеметрії унеможливить якісне навчання моделей прогнозування.

Таким чином, оптимальна архітектура вебзастосунку для діагностики несправностей обладнання ґрунтується на принципах модульності, масштабованості та чіткого розподілу обов'язків між шарами. Вона забезпечує не лише технічну ефективність, але й зручність для інженерів та менеджерів, які отримують достовірну інформацію для прийняття обґрунтованих рішень щодо технічного обслуговування. Правильно обрана структура стає фундаментом для створення надійної, гнучкої та перспективної системи, здатної адаптуватися до змін у промисловому середовищі та нових досягнень у галузі аналізу даних [19].

#### **1.4 Обґрунтування вибору інструментальних засобів та вимог до апаратних ресурсів**

Вибір інструментальних засобів для створення вебзастосунку, призначеного для діагностики несправностей обладнання на основі прогнозної аналітики, має ґрунтуватися на комплексному врахуванні функціональних потреб системи, таких як обробка великих обсягів даних у реальному часі, застосування методів машинного навчання для виявлення аномалій та прогнозування залишкового ресурсу, а також забезпечення зручного доступу через вебінтерфейс [20]. Цей вибір визначається необхідністю балансу між продуктивністю, масштабованістю, безпекою та зручністю підтримки. Сучасні системи такого типу вимагають розподіленої архітектури, де окремі компоненти відповідають за різні аспекти функціонування: прийом та зберігання телеметрії, аналітичну обробку та візуалізацію результатів.

Одним із ключових аспектів є використання середовищ програмування, які дозволяють ефективно реалізовувати як серверну логіку, так і клієнтську частину. Для серверної сторони доцільно обирати технології, що підтримують асинхронну обробку запитів і забезпечують високу швидкість роботи з базами

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 24   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

даних. Це особливо важливо для систем, які постійно отримують потоки даних від сенсорів обладнання.

З іншого боку, для аналітичних модулів, пов'язаних з побудовою моделей прогнозування, необхідні інструменти, що мають розвинені бібліотеки для статистичного аналізу та навчання моделей. Клієнтська частина повинна забезпечувати динамічну візуалізацію графіків, діаграм та інтерактивних панелей моніторингу, що вимагає сучасних фреймворків для створення користувацьких інтерфейсів [21].

Схема типової архітектури вебзастосунку для прогнозної діагностики обладнання, що ілюструє взаємодію основних компонентів: серверної частини, аналітичного модуля та клієнтського інтерфейсу, представлена на рисунку 1.7.



Рисунок 1.7 – Схема типової архітектури вебзастосунку для прогнозної діагностики обладнання

Така архітектура забезпечує чітке розмежування обов’язків між компонентами, що полегшує масштабування та оновлення системи.

При виборі бази даних перевага надається рішенням, які підтримують швидке записування та читання часових рядів даних, оскільки телеметрія від обладнання надходить з високою частотою. Реляційні системи з підтримкою механізмів журналізації, таких як WAL, добре підходять для забезпечення цілісності даних навіть за умов інтенсивного навантаження. Для аналітичної обробки доцільно використовувати мови програмування з потужними екосистемами для наукових обчислень, що дозволяють інтегрувати алгоритми

виявлення аномалій, класифікації режимів роботи та регресійного прогнозування [22].

Порівняння основних характеристик інструментальних засобів, що застосовуються при створенні вебзастосунків для промислової діагностики наведене в таблиці 1.4.

Таблиця 1.4

Порівняння характеристик інструментальних засобів для вебзастосунків  
прогнозної діагностики

| Інструмент   | Основне призначення             | Переваги                                 | Недоліки                                                    | Рекомендації щодо застосування  |
|--------------|---------------------------------|------------------------------------------|-------------------------------------------------------------|---------------------------------|
| Node.js      | Серверна обробка запитів        | Висока швидкість, асинхронність          | Обмежені можливості для складних обчислень                  | Основне серверне середовище     |
| Python       | Аналітичні розрахунки та моделі | Багаті бібліотеки для машинного навчання | Нижча швидкість виконання порівняно з компільованими мовами | Модуль прогнозування            |
| Vue.js       | Клієнтський інтерфейс           | Реактивність, компонентна архітектура    | Потреба в додаткових бібліотеках для візуалізації           | Інтерактивні панелі моніторингу |
| SQLite з WAL | Зберігання телеметрії           | Легка вбудованість, надійність           | Обмеження при дуже великих обсягах                          | Локальне зберігання даних       |

Ця таблиця демонструє, як різні інструменти доповнюють один одного, забезпечуючи повноцінне функціонування системи. Вибір саме таких засобів зумовлений їхньою перевіркою в промислових умовах і можливістю інтеграції в єдиний технологічний стек.

Вимоги до апаратних ресурсів визначаються обсягом оброблюваних даних, складністю моделей та кількістю одночасно підключених користувачів. Для серверної частини достатньо обладнання з багатоядерним процесором, обсягом оперативної пам'яті не менше 16 ГБ та швидкісним накопичувачем для бази даних. Аналітичний модуль, який виконує навчання та інференс моделей, потребує ресурсів, здатних ефективно обробляти вектори ознак, тому бажано використовувати системи з підтримкою паралельних обчислень. У разі інтенсивного використання доцільно передбачити можливість розгортання на

віртуальних серверах або в хмарних середовищах, що дозволяє динамічно масштабувати ресурси залежно від навантаження [23].

На (рисунку 1.8) наведено схематичне представлення розподілу апаратних ресурсів за компонентами системи.

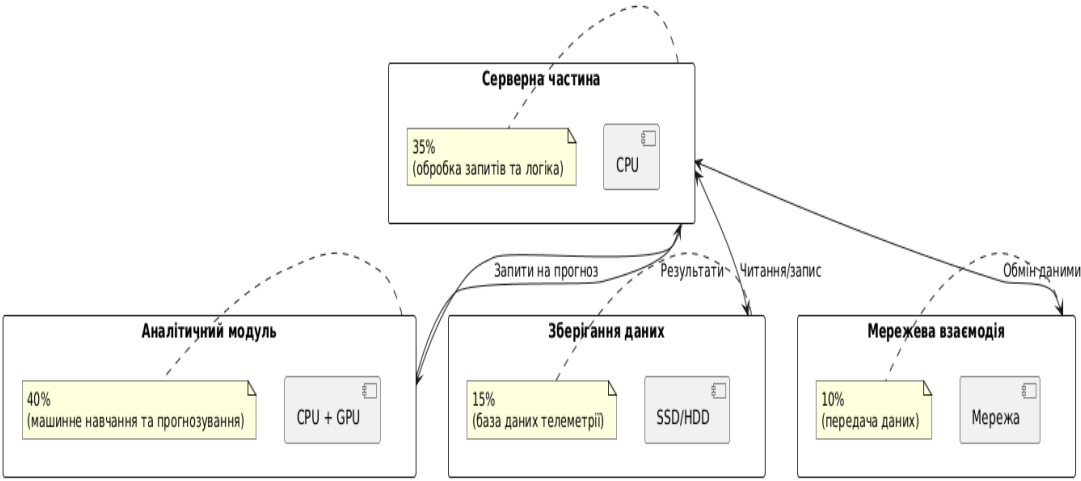


Рисунок 1.8 – Схема розподілу апаратних ресурсів за компонентами системи

Такий розподіл дозволяє оптимізувати витрати на обладнання, концентруючи обчислювальні потужності саме там, де вони найбільше потрібні – у модулях прогнозування.

Вимоги до апаратних ресурсів також залежать від очікуваної кількості обладнання, що підключається. Для систем середнього масштабу (десятки-сотні одиниць) достатньо стандартних серверних конфігурацій, тоді як для великих підприємств може знадобитися кластерна архітектура з розподіленим зберіганням даних. У будь-якому випадку, мінімальні вимоги повинні включати резервування ресурсів для пікових навантажень, пов'язаних з одночасним аналізом даних від багатьох джерел.

Таким чином, обґрунтований вибір інструментальних засобів і апаратних ресурсів забезпечує створення надійного, масштабованого та ефективного вебзастосунку, здатного вирішувати складні завдання діагностики несправностей обладнання на основі прогнозової аналітики. Цей підхід враховує як технічні можливості сучасних технологій, так і практичні потреби промислових підприємств, сприяючи підвищенню загальної ефективності технічного обслуговування.

## Висновки до першого розділу

Автоматизація діагностики технічного стану обладнання є необхідною умовою підвищення надійності виробничих процесів, зниження позапланових простоїв та оптимізації витрат на технічне обслуговування. Основними задачами, що підлягають автоматизації, виступають безперервний збір і обробка телеметрії, виявлення аномалій, прогнозування залишкового ресурсу обладнання з довірчими інтервалами, а також класифікація можливих режимів відмов. Перехід від традиційних періодичних оглядів до прогнозової стратегії, орієнтованої на реальний стан техніки, забезпечує оперативне реагування на відхилення та суттєве підвищення ефективності експлуатації промислових активів.

Аналіз сучасних комерційних платформ, таких як IBM Maximo Application Suite, Siemens MindSphere та Augury, продемонстрував їхні сильні сторони в реальному часі моніторингу, застосуванні методів машинного навчання та інтеграції з виробничими системами. Водночас виявлено потребу в обладнанні-агностичній архітектурі з високою гнучкістю, що особливо важливо для адаптації під умови українських підприємств.

Оптимальною для вебзастосунку визнана мікросервісна архітектура з багатошаровою структурою, яка поєднує масштабованість, незалежне оновлення компонентів та надійну обробку потоків даних. Вибрані інструментальні засоби – Node.js для серверної частини, Python для аналітичних модулів, Vue.js для клієнтського інтерфейсу та SQLite з WAL для зберігання телеметрії – забезпечують необхідний баланс продуктивності, зручності розробки та підтримки. Вимоги до апаратних ресурсів визначаються обсягами даних і складністю моделей, що дозволяє ефективно розгорнути систему як у локальному, так і в хмарному середовищі.

Таким чином, проведений теоретичний аналіз формує концептуальну основу для створення сучасного вебзастосунку прогнозової діагностики, здатного вирішувати актуальні промислові завдання з високою точністю та надійністю.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 28   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ ДЛЯ ДІАГНОСТИКИ НЕСПРАВНОСТЕЙ ОБЛАДНАННЯ

### 2.1 Визначення варіантів використання та об'єктно орієнтованої структури системи

Проектування вебзастосунку для діагностики несправностей обладнання на основі прогнозової аналітики вимагає чіткого визначення потреб усіх учасників процесу технічного обслуговування. Сучасні системи такого типу орієнтовані на перехід від традиційного планового обслуговування до підходу, що ґрунтується на реальному стані техніки [24]. Це дозволяє своєчасно виявляти відхилення в роботі обладнання, прогнозувати його залишковий ресурс і попереджати критичні відмови.

Такий підхід значно підвищує надійність виробничих процесів, знижує витрати на ремонти та оптимізує використання ресурсів підприємства.

У центрі уваги перебуває визначення варіантів використання, які відображають основні сценарії взаємодії користувачів із системою. Головними дійовими особами (акторами) виступають оператори виробничого обладнання, інженери з технічного обслуговування, керівники підрозділів та адміністратори системи. Кожен з них має специфічні завдання, що вимагають відповідного функціоналу. Наприклад, оператор потребує доступу до поточного стану обладнання в реальному часі, щоб оперативно реагувати на сигнали.

Інженер з обслуговування використовує інструменти для глибокого аналізу даних, класифікації можливих режимів відмов і формування рекомендацій щодо профілактичних заходів. Керівництво отримує узагальнені звіти про стан парку техніки, ризики та прогнозовані показники ефективності.

Адміністратор системи відповідає за управління моделями обладнання, налаштування доступу та забезпечення безперебійної роботи платформи. Варіанти використання охоплюють увесь цикл моніторингу та прогнозування. Центральним є процес безперервного збору та візуалізації телеметричних даних з різних каналів сигналів обладнання. Це включає відстеження параметрів, таких

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 29   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

як температура, вібрація, тиск, навантаження та інші характеристики, що дозволяють формувати повну картину стану техніки.

Наступним важливим варіантом є виявлення аномалій, яке відбувається через аналіз накопичених даних і порівняння з нормальними режимами роботи. Система повинна автоматично сигналізувати про відхилення, що перевищують встановлені порогові значення або демонструють нехарактерну динаміку.

Особливе місце посідає класифікація режимів відмов. Тут застосовуються методи, що дозволяють не лише констатувати проблему, але й визначати її тип на основі характерних ознак. Це допомагає інженерам швидко зорієнтуватися в ситуації та обрати оптимальний спосіб втручання.

Прогнозування залишкового ресурсу (RUL) є одним із найбільш цінних варіантів використання, оскільки надає кількісну оцінку часу до ймовірної відмови з урахуванням довірчих інтервалів. Такий прогноз базується на комплексному аналізі історичних даних і поточних тенденцій, що дає змогу планувати обслуговування з випередженням.

Крім того, система повинна підтримувати управління моделями обладнання, включаючи створення, редагування та видалення описів різних типів техніки. Це забезпечує гнучкість і можливість адаптації до нових видів обладнання без значних змін у коді.

Важливим є також ведення журналу подій, де фіксуються всі значущі зміни стану, дії користувачів та рекомендації системи. Це сприяє прозорості процесів і полегшує подальший аудит.

Для наочного представлення взаємодії акторів із системою доцільно використовувати діаграму варіантів використання, що представлена на рисунку 2.1, яка відображає основні сценарії та їхні зв'язки.

Окрім визначення сценаріїв використання, важливим етапом проектування є формування об'єктно-орієнтованої структури системи. Такий підхід дозволяє розділити функціональність на окремі сутності, кожна з яких відповідає за певний аспект роботи програмного забезпечення.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 30   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Основними об'єктами системи є обладнання, датчики, телеметричні показники, події, прогнози, моделі машинного навчання та користувачі.

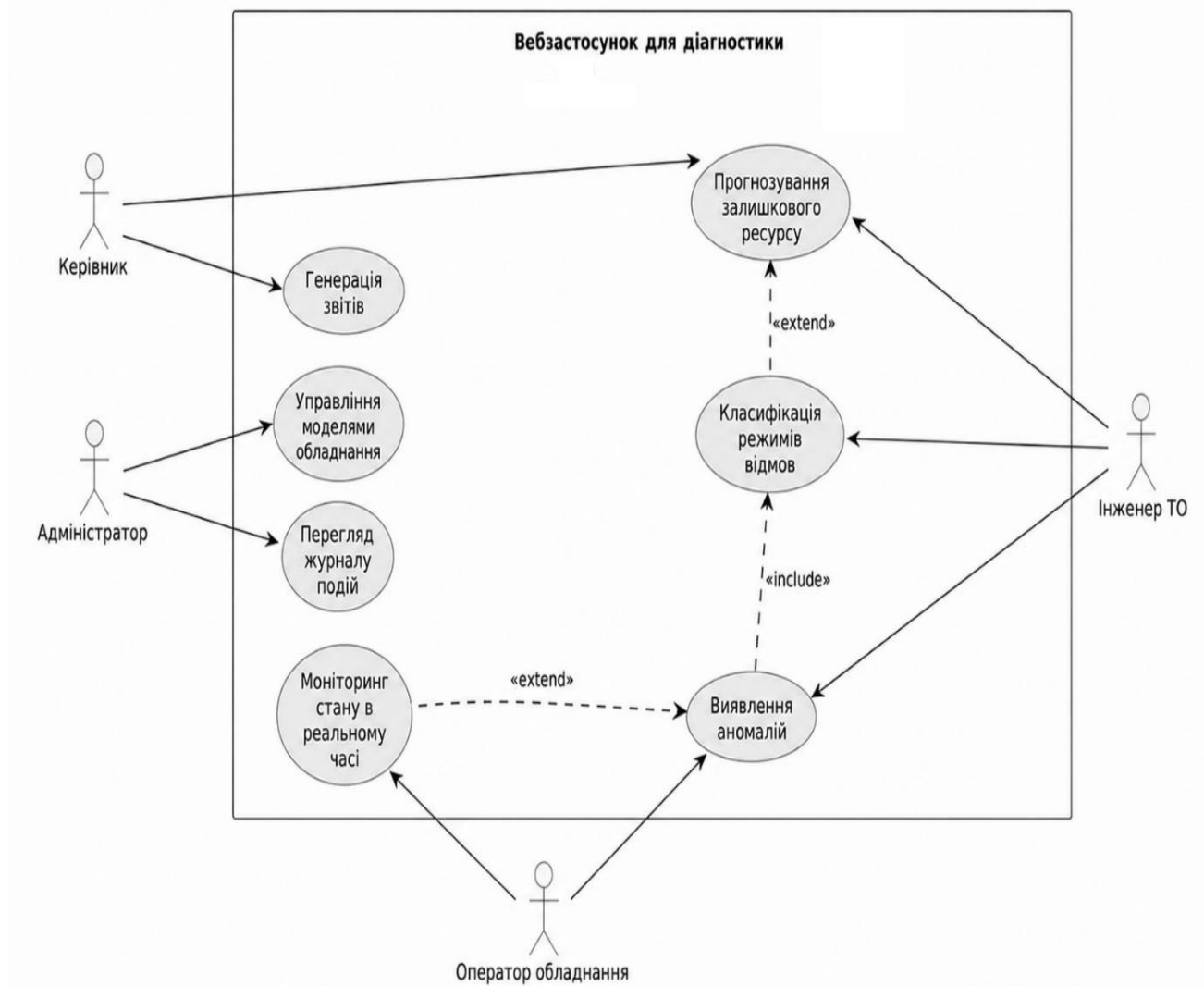


Рисунок 2.1 – Діаграма варіантів використання системи діагностики обладнання

Рисунок 2.1 ілюструє ключові взаємодії між користувачами та функціональними модулями вебзастосунку, підкреслюючи залежності між різними сценаріями роботи. Така структура забезпечує логічну послідовність процесів і дозволяє уникнути дублювання функцій.

Переходячи до об'єктно орієнтованої структури системи, слід зазначити, що вона базується на принципах інкапсуляції, спадкування та поліморфізму, що сприяє гнучкості та масштабовності рішення [25]. Основними класами є сутності, які моделюють реальні об'єкти виробничого середовища.

Центральне місце посідає клас «Модель обладнання», який містить декларативний опис техніки, включаючи перелік каналів сигналів, порогові значення, діапазони нормальної та критичної роботи. Цей клас служить основою для створення конкретних екземплярів обладнання.

Клас «Одиниця обладнання» успадковує властивості моделі та додає специфічні атрибути, такі як серійний номер, місце розташування, дата введення в експлуатацію та накопичений ресурс. Він підтримує зв'язок із телеметричними даними, дозволяючи відстежувати історію роботи. Важливим елементом є клас «Телеметричний запис», який фіксує значення сигналів у певний момент часу з урахуванням якості даних.

Для обробки аналітики передбачено класи, пов'язані з прогнозуванням. Клас «Оцінка стану» агрегує результати аналізу аномалій, класифікації режимів і прогнозу ресурсу, надаючи комплексну характеристику поточного здоров'я техніки. Клас «Подія» реєструє всі значущі інциденти, включаючи автоматичні сповіщення системи та дії користувачів. Це забезпечує повну історію для подальшого аналізу.

Об'єктно орієнтована модель також включає сервісні класи для взаємодії з даними. Наприклад, клас «Збір даних» відповідає за прийом телеметрії, а клас «Аналізатор» інкапсулює логіку виявлення відхилень і прогнозування. Використання інтерфейсів дозволяє легко інтегрувати різні алгоритми аналізу без зміни основної структури.

Для візуального представлення статичної структури системи застосовується діаграма класів, що наведена на рисунку 2.2, яка демонструє основні сутності та їхні взаємозв'язки.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 32   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

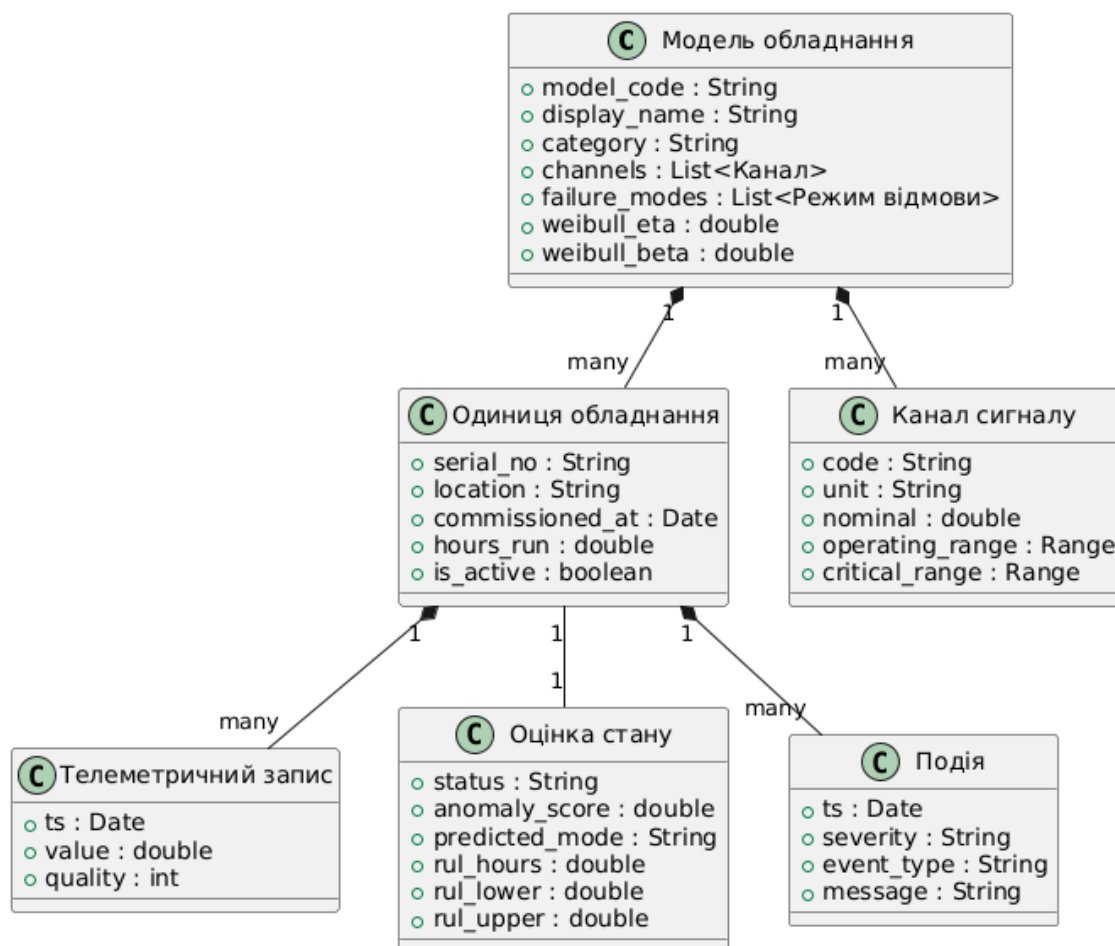


Рисунок 2.2 – Діаграма класів об’єктно орієнтованої структури системи

Така модель забезпечує чітке розділення відповідальності між компонентами та полегшує подальшу розробку та підтримку.

Основні класи об’єктно орієнтованої структури системи включають клас «Модель обладнання», який інкапсулює декларативний опис типової техніки з переліком каналів сигналів, порогових значень та можливих режимів відмов. Клас «Одиниця обладнання» успадковує властивості моделі та розширює їх атрибутами конкретного екземпляра, такими як серійний номер, місце розташування, накопичений ресурс роботи та статус активності. Клас «Телеметричний запис» відповідає за фіксацію значень сигналів у часі з урахуванням їхньої якості.

Клас «Оцінка стану» агрегує результати аналізу аномалій, класифікації режимів відмов та прогнозування залишкового ресурсу, формуючи комплексну характеристику техніки. Нарешті, клас «Подія» забезпечує системне журналювання всіх значущих інцидентів, сповіщень та дій користувачів.

Така структура сприяє цілісності даних, логічній організації функціональності та полегшує розуміння розподілу відповідальності між компонентами вебзастосунку.

Архітектура об'єктно орієнтованого проєктування системи побудована на ключових принципах інкапсуляції, спадкування та поліморфізму, що забезпечує високу гнучкість і масштабованість рішення. Інкапсуляція дозволяє приховати внутрішню реалізацію класів, таких як обробка телеметрії чи алгоритми прогнозування, та надавати лише необхідний інтерфейс. Спадкування застосовується для створення ієрархії між загальними моделями обладнання та їхніми конкретними екземплярами, що зменшує дублювання коду та спрощує розширення системи новими типами техніки.

Поліморфізм забезпечує можливість використання різних алгоритмів аналізу через єдиний інтерфейс, що особливо важливо для інтеграції методів машинного навчання. Використання патерну «Фабрика» для створення екземплярів обладнання та патерну «Спостерігач» для оновлення стану в реальному часі дозволяє ефективно організувати взаємодію компонентів, забезпечити слабку залежність між модулями та спростити подальше технічне обслуговування системи.

Такий вибір архітектури обґрунтований необхідністю обробки різномірних типів обладнання, динамічної роботи з великими обсягами телеметричних даних та забезпеченням зручності розширення функціональності без суттєвих змін у базовій структурі.

Для забезпечення ефективної роботи високонавантаженої системи реального часу, особливо з урахуванням обробки телеметрії з інтервалом у 10 секунд та застосування складних моделей машинного навчання, таких як Isolation Forest для виявлення аномалій, Random Forest для класифікації режимів відмов і Gradient Boosting для прогнозування залишкового ресурсу, важливо детально описати динамічну взаємодію компонентів.

Архітектура системи передбачає асинхронний обмін даними між основними модулями, що дозволяє уникнути блокувань і забезпечити

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 34   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

масштабованість. Центральним елементом є потік обробки телеметрії від джерел даних до аналітичних сервісів. Такий підхід відповідає сучасним вимогам до промислових систем моніторингу, де затримки в обробці інформації можуть призвести до пропуску критичних відхилень у роботі обладнання. Асинхронна архітектура дає змогу серверу обробки телеметрії продовжувати приймати нові дані, навіть коли аналітичний модуль виконує обчислення, що особливо важливо при великій кількості одиниць обладнання, підключених одночасно.

Рисунок 2.3 демонструє послідовність взаємодії компонентів при надходженні телеметрії, що забезпечує безперервну роботу системи в реальному часі з асинхронною обробкою в аналітичному модулі.

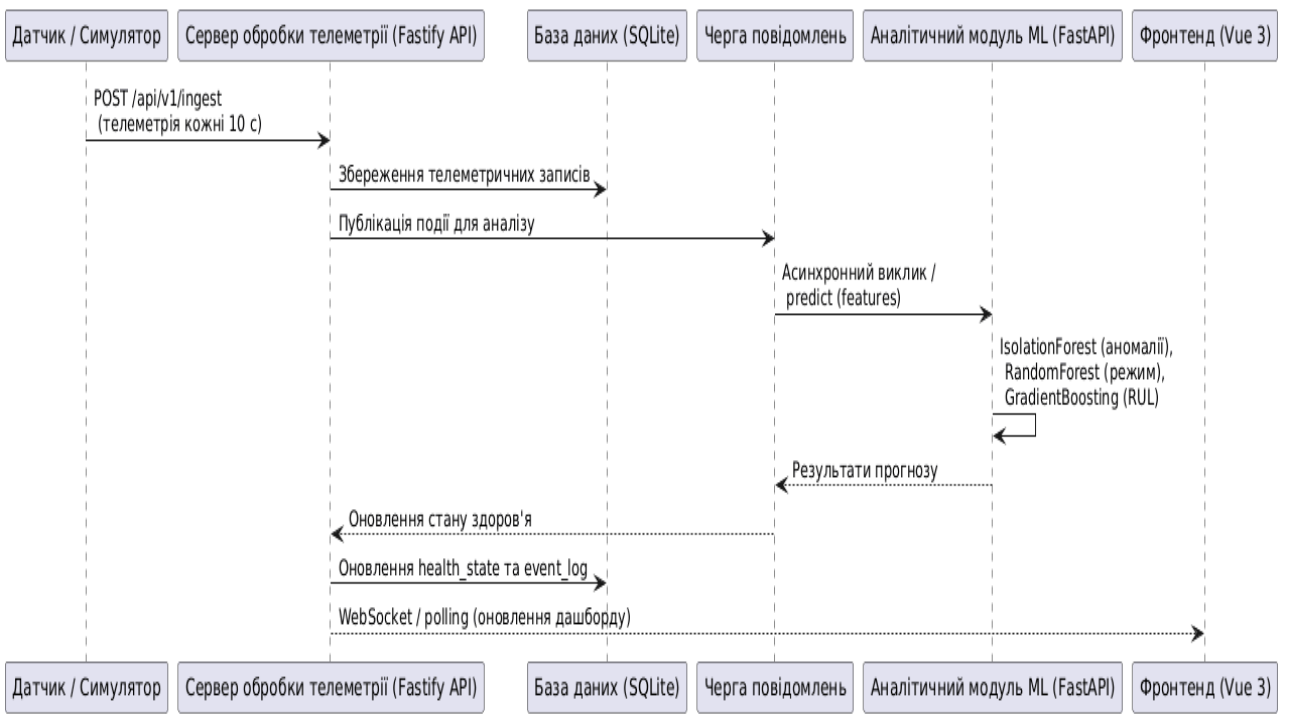


Рисунок 2.3 – Діаграма послідовності обробки телеметрії та прогнозової аналітики

Кожен крок у цій послідовності оптимізований для мінімальної затримки: сервер приймає дані, зберігає їх у базі з використанням режиму WAL для швидкого запису, а потім через чергу передає ознаки для аналізу. Це дозволяє ML-модулю незалежно виконувати обчислення, не блокуючи основний потік обробки вхідних сигналів.

Така організація особливо ефективна при одночасній роботі з кількома

типами обладнання, оскільки черга забезпечує балансування навантаження та можливість горизонтального масштабування аналітичних інстансів. Крім того, після отримання результатів прогнозу система оновлює стан здоров'я в базі даних і негайно повідомляє фронтенд через механізми реального часу, що дозволяє операторам оперативно реагувати на зміни.

Подібна динаміка взаємодії компонентів сприяє підвищенню загальної надійності системи. Вона мінімізує ризики втрати даних під час пікових навантажень і забезпечує високу точність прогнозів завдяки комплексному використанню різних алгоритмів машинного навчання на одному вікні ознак. У разі зростання обсягів телеметрії архітектура дозволяє легко додавати нові інстанси ML-сервісу без переривання роботи основного сервера.

Для демонстрації масштабованості системи на фізичному або хмарному рівні застосовується діаграма розгортання (рисунок 2.4), яка відображає розміщення компонентів і їхні взаємозв'язки.

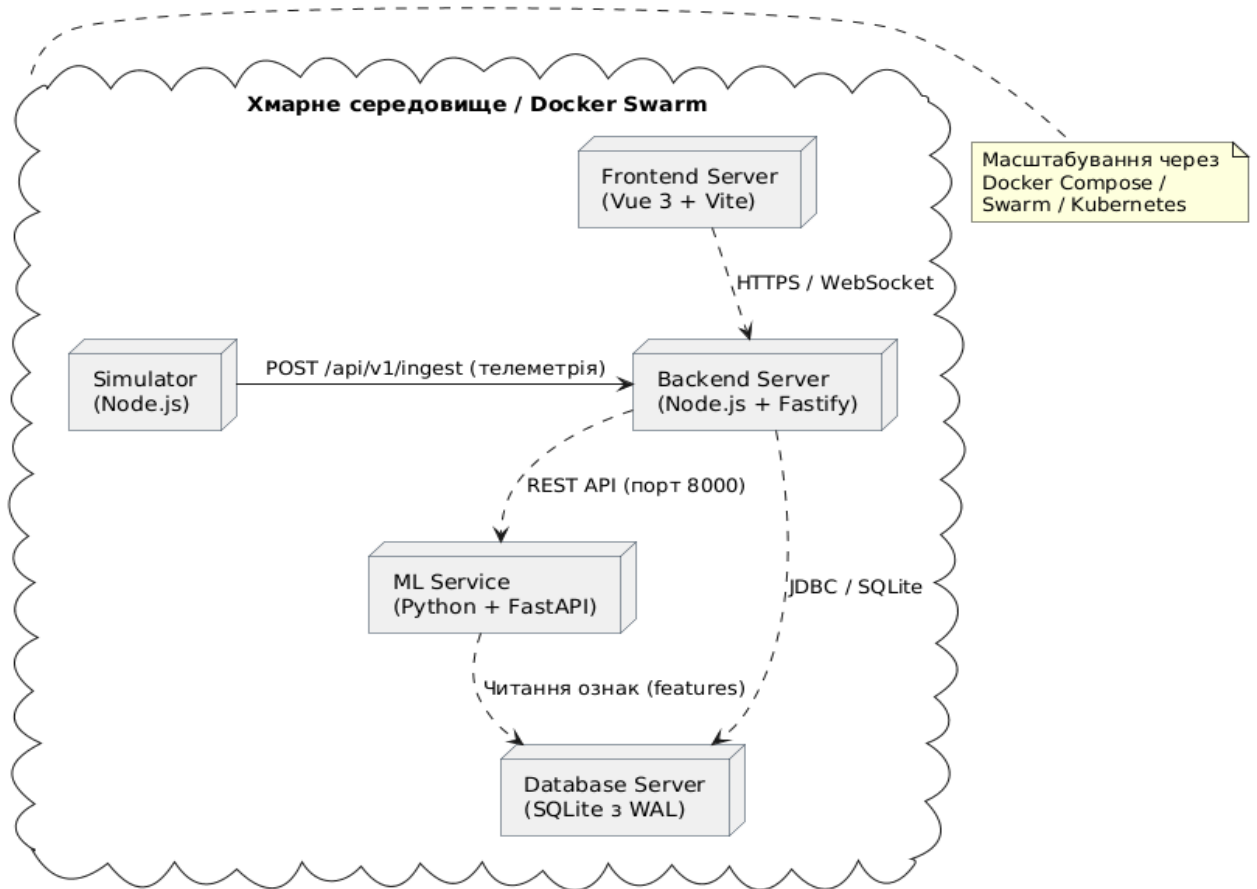


Рисунок 2.4 – Діаграма розгортання системи

Рисунок 2.4 ілюструє розподілену архітектуру, що підтримує контейнеризацію через Docker Compose, горизонтальне масштабування та інтеграцію з хмарними ресурсами, забезпечуючи високу доступність і продуктивність при обробці великих обсягів телеметрії.

На рівні розгортання система демонструє гнучкість: фронтенд може працювати на окремому сервері або в CDN для швидкого відображення інтерфейсу, бекенд-сервер відповідає за прийом даних і оркестрацію, а ML-сервіс може масштабуватися незалежно залежно від обчислювального навантаження.

База даних з режимом WAL забезпечує швидкий запис телеметрії навіть при високій частоті надходження даних. Така структура дозволяє розгортати систему як на локальних серверах промислового підприємства, так і в хмарному середовищі, де ресурси динамічно виділяються під поточне навантаження.

Детальна архітектурна модель з урахуванням динамічних і розгортальних аспектів суттєво посилює обґрунтування системи як рішення для реального промислового застосування. Вона враховує потреби високої пропускної здатності, низької затримки обробки та можливості подальшого розвитку без радикальних змін у структурі.

Завдяки чіткому розділенню обов’язків між компонентами забезпечується висока стійкість до відмов, легкість у супроводі та можливість інтеграції додаткових модулів аналізу в майбутньому.

Такий підхід повністю відповідає принципам побудови сучасних прогностичних систем технічного обслуговування, де поєднання реального часу та глибокої аналітики є ключовим фактором успіху.

Таким чином, представлені діаграми послідовності та розгортання доповнюють базову об’єктно орієнтовану модель, створюючи повну картину архітектури, готової до промислової експлуатації в умовах інтенсивного потоку даних.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 37   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Правильно спроектована архітектура дозволяє масштабувати рішення, інтегрувати нові методи аналізу та підтримувати високий рівень надійності в умовах постійного розвитку промислових технологій [26].

## **2.2 Розроблення структур даних та математичних алгоритмів прогнозної аналітики**

У сучасних системах технічного обслуговування обладнання, орієнтованих на прогнозування, ключову роль відіграє чітко організована система збору, зберігання та обробки інформації про стан пристроїв [27]. Ефективна прогнозна аналітика вимагає не лише накопичення сирих вимірювань, але й створення уніфікованих структур, які дозволяють описувати технічні характеристики обладнання незалежно від його типу, а також алгоритмів, здатних виявляти відхилення, класифікувати режими несправностей і оцінювати залишковий ресурс.

Таке поєднання забезпечує перехід від реактивного підходу до випереджального, зменшуючи ймовірність аварійних зупинок і оптимізуючи витрати на утримання.

Основою для опису будь-якого обладнання є уніфікований паспорт, який містить декларативний набір параметрів. Цей паспорт включає перелік каналів сигналів з їхніми технічними характеристиками, такими як номінальні значення, діапазони нормальної та критичної експлуатації, інерційність сенсорів і рівень шумів. Крім того, в ньому зазначаються можливі режими відмов з вказівкою на уражені канали та типові часові горизонти розвитку несправностей.

Така структура дозволяє системі працювати з різноманітним обладнанням – від обчислювальних серверів до кріогенних установок – без потреби в перепрограмуванні основної логіки. Паспорт зберігається в компактному текстовому форматі, що полегшує його передачу, перевірку цілісності через хешування та подальшу обробку.

Для фіксації поточного стану обладнання формується окрема структура даних, яка агрегує результати моніторингу. Вона охоплює поточний статус

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 38   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

(нормальний, ризиковий чи критичний), кількісну оцінку аномальності, прогнозований режим відмови з рівнем впевненості, а також оцінку залишкового корисного життя з довірчими інтервалами. Ця структура постійно оновлюється на основі надходження нових даних і служить фундаментом для візуалізації та прийняття рішень операторами.

Для формування таких комплексних оцінок стану застосовуються методи машинного навчання, реалізовані в модулі `ml_service` проєкту. Виявлення аномалій і класифікація режимів відмови здійснюються відповідно алгоритмами `IsolationForest` (навчається на здорових даних) та `RandomForestClassifier`, тренування яких реалізовано в `ml_service/training/train.py`; інженерія ознак описана в `ml_service/app/features.py`. Прогнозування залишкового ресурсу виконується трьома `GradientBoostingRegressor` з квантилями  $\alpha = 0.1, 0.5, 0.9$ , а повний пайплайн інференсу реалізований в `ml_service/app/inference.py`.

Параметри каналів сигналів у структурі паспорта обладнання представлені в таблиці 2.1.

Таблиця 2.1

Параметри каналів сигналів у структурі паспорта обладнання

| Назва параметра            | Опис                                      | Приклади значень                                            |
|----------------------------|-------------------------------------------|-------------------------------------------------------------|
| Код каналу                 | Унікальний ідентифікатор сигналу          | <code>cpu_temp_pkg</code> ,<br><code>vibration x rms</code> |
| Клас сигналу               | Тип фізичної величини                     | Т (температура), V (вібрація)                               |
| Номінальне значення        | Типове значення в здоровому стані         | 55 °C, 2.5 мм/с                                             |
| Діапазон нормальної роботи | Межі допустимих відхилень                 | [30, 75]                                                    |
| Критичний діапазон         | Межі, що сигналізують про небезпеку       | [85, 105]                                                   |
| Інерційність (секунди)     | Час згладжування сенсором                 | 30, 120                                                     |
| Рівень шуму                | Стандартне відхилення випадкових коливань | 0.6, 0.2                                                    |

Таблиця 2.1 ілюструє ключові елементи опису каналів, які забезпечують контекст для подальшої аналітики.

Важливим етапом обробки є інженерія ознак на основі часових вікон телеметрії. Для кожного каналу обчислюються статистичні характеристики: середнє значення, стандартне відхилення, мінімум, максимум, медіана та кут нахилу тренду.

Додатково враховуються контекстні показники, такі як відхилення від номіналу відносно робочого діапазону, частка часу перебування поза нормальними межами та в критичній зоні. Ці розрахунки виконуються над рухомим вікном останніх вимірювань, що дозволяє алгоритмам реагувати на динаміку процесів. Такий підхід перетворює сирі послідовності даних у багатовимірний вектор ознак, придатний для застосування математичних моделей.

Для оцінки часу до відмови (залишкового ресурсу) широко застосовується розподіл Вейбулла, який добре описує процеси зношування обладнання.

Формула 2.1 – формула розподілу щільності ймовірності Вейбулла:

$$f(t, \eta, \beta) = \frac{\beta}{\eta} \left(\frac{t}{\eta}\right)^{\beta-1} \exp\left(-\left(\frac{t}{\eta}\right)^{\beta}\right) \quad (2.1)$$

де  $t$  – час до відмови,  $\eta$  – масштабний параметр (характеристичний ресурс),  $\beta$  – параметр форми (визначає тип зношування).

Для виявлення аномалій застосовується ансамблевий метод, заснований на ізоляції викидів. Алгоритм будує випадкові дерева, які намагаються ізолювати кожну точку даних. Чим швидше точка ізолюється, тим вища ймовірність її аномальності. Цей метод ефективний для виявлення відхилень без потреби в попередньому навчанні на прикладах несправностей, оскільки спирається переважно на здорові режими роботи. Результуюча оцінка аномальності нормалізується до інтервалу від 0 до 1, де вищі значення вказують на потенційні проблеми.

Класифікація режиму відмови реалізується за допомогою ансамблю дерев рішень, який враховує збалансованість класів. Модель навчається на векторах ознак, розмічених відповідними режимами несправностей або позначкою нормального стану. Вона дозволяє не лише виявити факт відхилення, але й визначити найбільш імовірний тип проблеми, наприклад, знос підшипника чи порушення вакуумної ізоляції, з відповідним рівнем впевненості. Для прогнозування залишкового корисного життя здійснюється навчання регресійних моделей на основі градієнтного бустінгу [28].

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 40   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Для прогнозування залишкового корисного життя використовуються квантильні регресори. Основна формула прогнозу в градієнтному бустингу для квантилю  $\alpha$  має вигляд формули 2.2:

$$\hat{y}_{\alpha} = F_0(x) + \sum_{m=1}^M v \cdot \gamma_m \cdot h_m(x) \quad (2.2)$$

де  $F_0(x)$  – початкова функція (зазвичай середнє значення),  $v$  – коефіцієнт навчання,  $\gamma_m$  – оптимальне значення для  $m$ -го дерева,  $h_m(x)$  –  $m$ -те дерево рішень,  $M$  – кількість дерев.

Прогнозування залишкового корисного життя здійснюється за допомогою регресійних моделей, побудованих на основі градієнтного бустингу. Для підвищення надійності оцінок використовуються квантильні регресори, які надають не лише точкову оцінку, але й нижню та верхню межі довірчого інтервалу (зазвичай 90%). Це дає змогу враховувати невизначеність прогнозу, що особливо важливо при прийнятті управлінських рішень. Моделі навчаються на історичних даних, де цільовою змінною є час до переходу в критичний стан.

Для побудови ознак на основі часового вікна використовуються такі статистичні показники:

Середнє значення для каналу має вигляд формули 2.3:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i \quad (2.3)$$

де  $x_i$  – значення сигналу в  $i$ -й точці вікна,  $n$  – кількість точок у вікні.

Стандартне відхилення представлене формулою 2.4:

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2} \quad (2.4)$$

Кут нахилу (тренд) за методом найменших квадратів, визначається за формулою 2.5:

$$slope = \frac{\sum_{i=1}^n (i - \bar{i})(x_i - \bar{x})}{\sum_{i=1}^n (i - \bar{i})^2} \quad (2.5)$$

де  $\bar{i}$  – середнє значення індексів часу.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 41   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Рисунок 2.5 відображає загальну схему взаємодії структур даних та алгоритмів у процесі прогнозової аналітики.

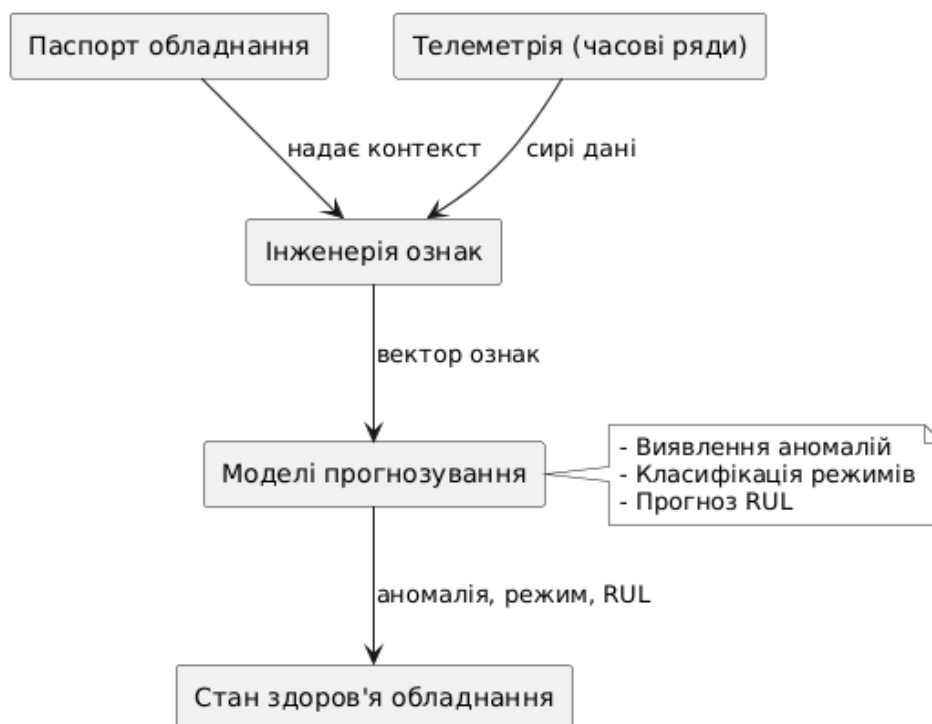


Рисунок 2.5 – Схема взаємодії структур даних та алгоритмів прогнозової аналітики

На схемі (Рисунок 2.5) показано, як паспорт і поточна телеметрія живлять процес інженерії ознак, який, у свою чергу, забезпечує вхідними даними математичні моделі для формування комплексної оцінки стану. Крім того, для класифікації поточного статусу обладнання застосовується правило з гістерезисом, що враховує попередній стан і запобігає нестабільним переходам між категоріями.

Для виявлення аномалій оцінка обчислюється на основі глибини ізоляції в ансамблі дерев, а для прогнозу режиму відмови – ймовірності класів з ансамблю. Усі ці компоненти інтегруються в єдину систему, де результати одного етапу стають вхідними для наступного, забезпечуючи комплексний аналіз.

Важливим елементом проєктування структур даних є створення реляційної моделі бази даних, яка забезпечує цілісність інформації, ефективне зберігання паспортів моделей обладнання, телеметрії, поточного стану та журналу подій. Для цього розроблено ER-діаграму (рисунок 2.6), що відображає основні

сутності, їхні атрибути, типи даних, первинні та зовнішні ключі, а також зв'язки між ними. Така модель дозволяє уникнути надмірної надлишковості даних, забезпечити швидкий доступ до інформації та підтримувати складні аналітичні запити.

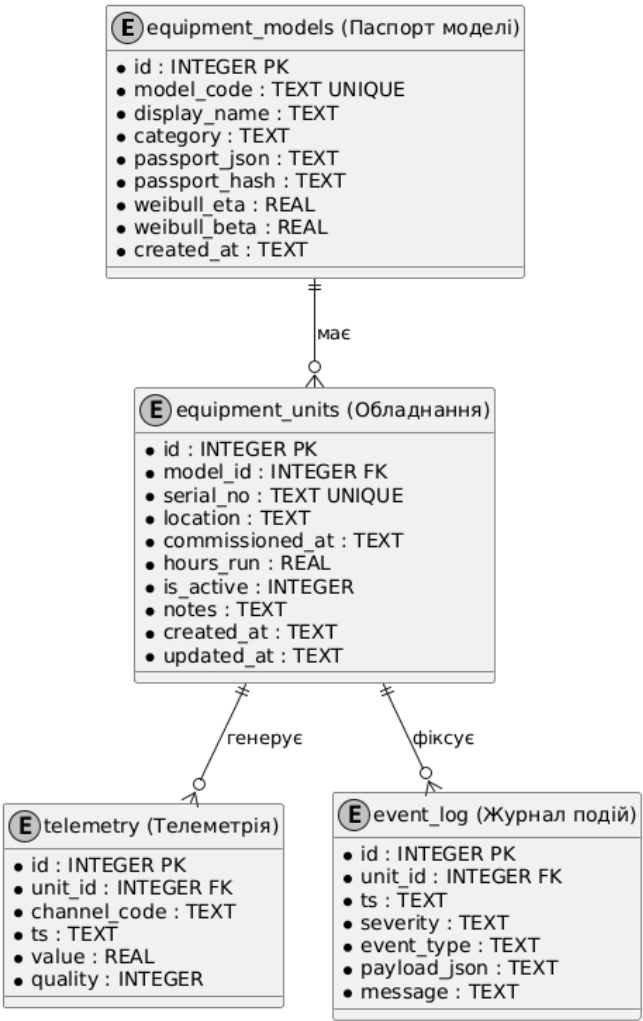


Рисунок 2.6 – ER-діаграма основних сутностей бази даних

У наведеній ER-діаграмі (Рисунок 2.6) сутність «Паспорт моделі» (equipment\_models) є центральною для декларативного опису обладнання. Вона містить первинний ключ id, унікальний код моделі model\_code, повне ім'я display\_name, категорію, повний JSON-паспорт з характеристиками каналів і режимів відмов, хеш для перевірки цілісності, параметри розподілу Вейбулла (weibull\_eta та weibull\_beta) для моделювання ресурсу, а також мітку часу створення.

Сутність «Обладнання» (equipment\_units) пов'язана з паспортом моделі через зовнішній ключ model\_id, що забезпечує посилову цілісність. Вона включає первинний ключ id, унікальний серійний номер serial\_no, місце розташування, дату введення в експлуатацію, накопичені години роботи, статус активності, нотатки та мітки часу оновлення.

Сутність «Телеметрія» (telemetry) зберігає всі вхідні сигнали і пов'язана з обладнанням через зовнішній ключ unit\_id. Кожний запис містить ідентифікатор каналу, часову мітку ts, числове значення та показник якості. Індеси на комбінаціях unit\_id + channel\_code + ts забезпечують швидкий пошук за часом і обладнанням.

Сутність «Журнал подій» (event\_log) фіксує всі значущі зміни стану, повідомлення та дії системи. Вона також пов'язана з обладнанням через unit\_id і містить рівень серйозності, тип події, JSON-навантаження та текстове повідомлення.

Додаткова сутність «Стан здоров'я» (health\_state) пов'язана один-до-одного з обладнанням і зберігає поточну оцінку аномальності, прогнозований режим відмови, залишковий ресурс з довірчими інтервалами та версію моделі, що використовувалася для розрахунку.

Така структура бази даних відповідає принципам нормалізації (принаймні до третьої нормальної форми), мінімізує дублювання даних і забезпечує високий рівень цілісності завдяки обмеженням FOREIGN KEY, UNIQUE, NOT NULL та DEFAULT. Використання індесів на часто використовуваних полях (наприклад, ts у телеметрії та unit\_id у подіях) дозволяє ефективно обробляти великі обсяги даних у реальному часі. Нормалізована модель полегшує підтримку, масштабування та інтеграцію з алгоритмами прогнозової аналітики, оскільки кожна сутність має чітко визначене призначення та зв'язки.

Для підтримки прийняття рішень розроблено правила класифікації статусу обладнання з урахуванням гістерезису, що запобігає частим коливанням між категоріями. Якщо оцінка аномальності перевищує певний поріг або прогнозований ресурс стає критично малим, статус підвищується до ризикового

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 44   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

чи невідкладного. Додатково реалізовано механізм автоматичного підвищення статусу для застарілих ризикових станів, що гарантує своєчасну увагу до потенційно небезпечних об'єктів.

Таким чином, поєднання чітко визначених структур даних із сучасними математичними алгоритмами машинного навчання створює надійну основу для прогнозної діагностики. Цей підхід забезпечує універсальність застосування, високу інтерпретованість результатів завдяки поясненням внеску окремих ознак, а також можливість постійного вдосконалення моделей на нових даних. У результаті формується система, яка не лише реєструє поточні відхилення, але й передбачає розвиток подій, сприяючи підвищенню надійності та ефективності експлуатації різноманітного технічного обладнання [29].

### 2.3 Проектування та опис компонентної будови вебзастосунку

Проектування компонентної будови вебзастосунку для діагностики несправностей обладнання на основі прогнозної аналітики вимагає створення гнучкої, масштабовної та надійної архітектури, яка забезпечує безперервний збір даних, їх обробку в реальному часі та надання користувачам зрозумілих результатів прогнозування [30]. Така система має поєднувати елементи клієнт-серверної взаємодії з елементами розподілених обчислень, що дозволяє ефективно обробляти великі обсяги телеметричної інформації від різноманітного промислового обладнання.

Основна мета полягає в забезпеченні незалежності від конкретного типу пристроїв, що досягається через декларативний опис характеристик обладнання та стандартизовані інтерфейси обміну даними.

Архітектура вебзастосунку будується за принципами багат шарової організації, де кожен шар виконує чітко визначені функції, мінімізуючи залежності між компонентами. Це сприяє легкості супроводження, розширенню функціональності та інтеграції з існуючими виробничими системами. Центральне місце посідає серверна частина, яка відповідає за прийом, зберігання

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 45   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

та первинну обробку даних, а також координацію взаємодії з модулями машинного навчання.

Клієнтська частина забезпечує інтерактивний інтерфейс для фахівців з технічного обслуговування, дозволяючи візуалізувати поточний стан обладнання, прогнози залишкового ресурсу та рекомендації щодо втручань.

Важливим аспектом є забезпечення реального часу моніторингу, що реалізується через регулярний обмін даними між джерелами телеметрії та основними сервісами. Для цього застосовуються асинхронні механізми передачі інформації, що дозволяють оперативно реагувати на відхилення параметрів. Крім того, система передбачає механізми автентифікації та авторизації запитів, що підвищує рівень безпеки при роботі з чутливими виробничими даними.

На рисунку 2.7 наведено схематичне представлення архітектури вебзастосунку для діагностики несправностей обладнання.

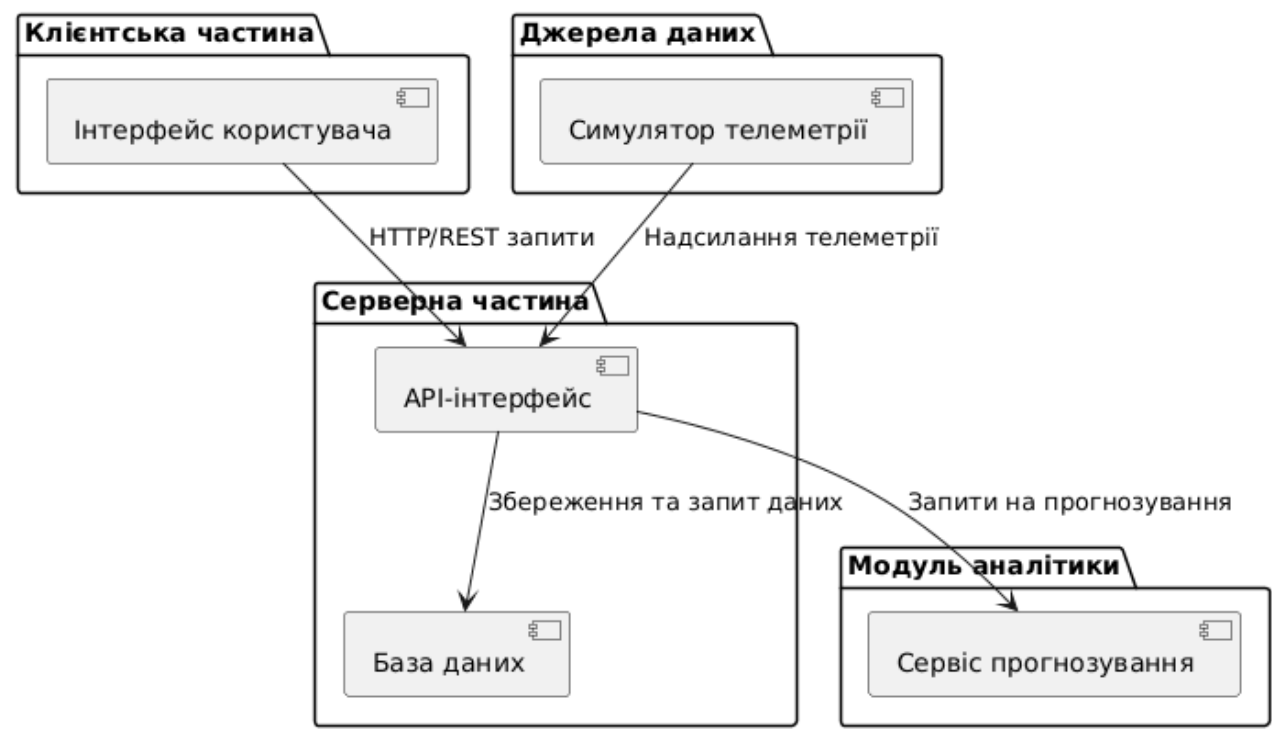


Рисунок 2.7 – Загальна компонентна архітектура вебзастосунку для діагностики несправностей обладнання

Як видно з рисунку 2.7, загальна компонентна архітектура передбачає чітке розділення відповідальності між основними блоками. Клієнтська частина взаємодіє з сервером через стандартизовані веб-протоколи, що забезпечує

кросплатформенність і доступність з різних пристроїв. Серверна частина виступає центральним хабом, який обробляє вхідні дані та делегує складні обчислення спеціалізованому модулю аналітики. Такий підхід відповідає сучасним тенденціям розвитку промислових інформаційних систем, де акцент робиться на модульності та можливості незалежного масштабування окремих компонентів [31].

Для ефективної роботи системи необхідне надійне сховище даних, здатне зберігати історичну телеметрію, описи обладнання та результати прогнозів. База даних має підтримувати швидкий доступ до інформації за різними критеріями, зокрема за часом та ідентифікаторами пристроїв. Це досягається завдяки індексації ключових полів і застосуванню механізмів оптимізації запитів.

Таблиця 2.2 ілюструє розподіл функцій між ключовими компонентами вебзастосунку. Кожен елемент архітектури спроектований таким чином, щоб забезпечити максимальну ефективність у контексті обробки даних від промислового обладнання.

Таблиця 2.2

#### Основні компоненти вебзастосунку та їх функції

| Компонент            | Основні функції                                                       | Технологічні особливості                      |
|----------------------|-----------------------------------------------------------------------|-----------------------------------------------|
| Клієнтська частина   | Візуалізація даних, взаємодія з користувачем, відображення прогнозів  | Сучасні фреймворки для динамічних інтерфейсів |
| API-інтерфейс        | Обробка запитів, автентифікація, маршрутизація                        | Швидкі веб-фреймворки з підтримкою REST       |
| База даних           | Зберігання телеметрії, моделей обладнання, журналів подій             | Реляційні системи з підтримкою транзакцій     |
| Сервіс прогнозування | Виявлення аномалій, класифікація режимів, прогноз залишкового ресурсу | Бібліотеки машинного навчання                 |
| Симулятор телеметрії | Генерація тестових даних для перевірки функціональності               | Автономний модуль для моделювання процесів    |

Більш детальний розгляд серверної частини показує, що вона складається з кількох взаємопов'язаних модулів. Модуль прийому даних відповідає за валідацію та збереження вхідної телеметрії, забезпечуючи цілісність інформації

навіть при великих обсягах надходжень. Модуль управління моделями обладнання дозволяє описувати різні типи пристроїв у стандартизованому форматі, що робить систему універсальною. Крім того, передбачено компонент планування регулярних оцінок стану, який періодично ініціює процеси прогнозування для активних одиниць обладнання.

На рисунку 2.8 представлено внутрішню структуру серверної частини, яка демонструє тісну взаємодію між модулями для забезпечення безперервного циклу моніторингу та діагностики.

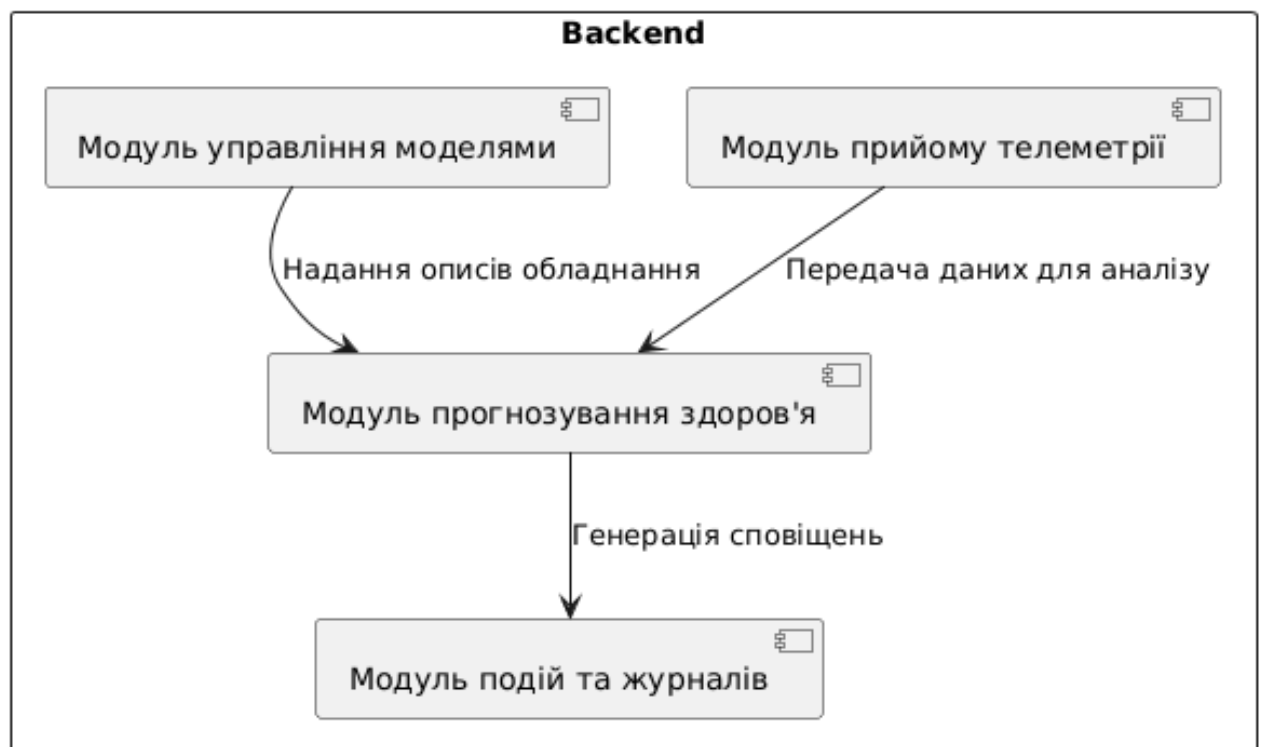


Рисунок 2.8 – Внутрішня структура серверної частини вебзастосунку

Така організація дозволяє оперативно реагувати на зміни стану обладнання та формувати своєчасні рекомендації для персоналу.

Модуль прогнозування, у свою чергу, є самостійним сервісом, що використовує методи машинного навчання для аналізу ознак сигналів. Він здатний виявляти відхилення, класифікувати можливі режими відмов та прогнозувати залишковий ресурс з урахуванням довірчих інтервалів. Такий підхід значно підвищує точність діагностики порівняно з традиційними методами, заснованими лише на порогових значеннях. Інтеграція цього модуля з

основним сервером відбувається через чітко визначені інтерфейси обміну, що забезпечує незалежність розробки та оновлення моделей.

Клієнтська частина вебзастосунку орієнтована на зручність використання фахівцями різних рівнів підготовки. Вона включає панелі моніторингу з візуальними індикаторами стану, детальні сторінки окремих одиниць обладнання та інструменти для аналізу історичних даних. Особлива увага приділяється інформативності графіків і діаграм, що дозволяють швидко оцінювати тенденції зміни параметрів. Крім того, передбачено систему сповіщень про критичні стани, що сприяє оперативному прийняттю рішень щодо технічного обслуговування.

Одним з ключових елементів архітектури є можливість розширення системи новими типами обладнання без суттєвих змін у коді основних компонентів. Це досягається завдяки декларативному підходу до опису характеристик пристроїв, включаючи набори каналів сигналів, порогові значення та типові режими відмов. Така гнучкість є важливою для промислових підприємств, де парк обладнання постійно оновлюється [32].

Механізм взаємодії між компонентами серверної частини забезпечує ефективну обробку даних без блокування основного потоку виконання. Основний сервер, реалізований на Node.js, виконує роль оркестратора запитів і координації, тоді як обчислювально інтенсивні задачі машинного навчання делегуються окремому мікросервісу на Python. Такий розподіл дозволяє оптимально використовувати переваги обох технологій: асинхронну модель обробки запитів Node.js та потужні математичні бібліотеки Python, зокрема scikit-learn для навчання та інференсу моделей.

Взаємодія між сервісами здійснюється через REST API за допомогою бібліотеки Axios у Node.js. Це забезпечує асинхронний обмін даними без блокування основного event loop сервера. Модуль клієнта ML-сервісу (ml/client.js) створює інстанс Axios з базовою URL-адресою Python-сервісу, заданим таймаутом у 30 секунд та необхідними заголовками. При кожній оцінці стану обладнання backend формує запит з набором інженерних ознак, передає

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 49   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Як показано на рисунку 2.9, процес взаємодії є чітко структурованим і включає резервний механізм евристичного прогнозування у разі відсутності навченої моделі для конкретного типу обладнання. Це підвищує стійкість системи.

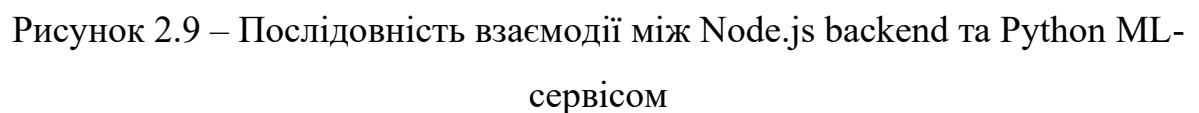


Таблица 2.3

| Параметр     | Використаний механізм (REST API + Axios) | Альтернативи (gRPC, RabbitMQ) | Переваги обраного рішення    |
|--------------|------------------------------------------|-------------------------------|------------------------------|
| 1            | 2                                        | 3                             | 4                            |
| Синхронність | Асинхронний (Promise-based)              | Синхронний / асинхронний      | Не блокує event loop Node.js |

Продовження таблиці 2.3

|                         |                           |                           |                                        |
|-------------------------|---------------------------|---------------------------|----------------------------------------|
| Формат даних            | JSON                      | Protocol Buffers / JSON   | Простота налагодження та читабельність |
| Складність впровадження | Низька                    | Середня / висока          | Швидка інтеграція з Fastify та FastAPI |
| Масштабованість         | Висока (HTTP/2 підтримка) | Дуже висока               | Достатня для промислового навантаження |
| Відмовостійкість        | Таймаути + fallback       | Вбудовані retry-механізми | Автоматичний перехід на евристику      |

У коді планувальника (ml/scheduler.js) функція `assessUnit` отримує вікно останніх 360 вимірювань з бази даних, формує ознаки за допомогою `buildFeatures` та передає їх у ML-сервіс через асинхронний виклик. У разі недоступності Python-модуля система продовжує роботу з попередніми оцінками, логуючи подію та використовуючи евристичний fallback. Такий дизайн відповідає вимогам промислових систем, де безперервність моніторингу є критичною.

Додатково, для локального сервера передбачено системний агент (sysagent.js), який самостійно реєструє обладнання, збирає метрики хосту (завантаження процесора, оперативної пам'яті, дискового простору) та надсилає телеметрію через той самий ingest-ендпоінт. Це демонструє гнучкість архітектури та можливість розширення на нові джерела даних без модифікації основного ядра. Агент працює в окремому інтервалі (кожні 30 секунд), забезпечуючи постійний моніторинг інфраструктури самого сервера.

Механізм взаємодії також підтримує пакетну обробку даних завдяки обмеженню розміру батчів у конфігурації (`ingest.batchLimit`). Це дозволяє ефективно обробляти великі обсяги телеметрії від кількох одиниць обладнання одночасно, мінімізуючи мережеве навантаження. У поєднанні з WAL-режимом SQLite та оптимізованими індексами база даних забезпечує швидке записування та читання даних навіть при високій частоті надходження сигналів.

Загалом, реалізований підхід до міжпроцесної взаємодії між Node.js та Python демонструє продуману архітектурну стратегію, орієнтовану на продуктивність, масштабованість та надійність. Він дозволяє системі ефективно справлятися з реальними виробничими навантаженнями, забезпечуючи

своєчасну діагностику несправностей та підтримку проактивного технічного обслуговування на підприємствах. Подальше вдосконалення може передбачати введення додаткових шарів кешування або перехід на більш спеціалізовані протоколи за умови зростання обсягів даних.

У цілому, спроектована компонентна будова вебзастосунку забезпечує баланс між функціональністю, продуктивністю та зручністю використання. Вона відповідає сучасним вимогам до систем прогностичного технічного обслуговування, дозволяючи підприємствам переходити від реактивних до проактивних стратегій управління обладнанням. Подальше вдосконалення архітектури може включати інтеграцію з хмарними сервісами та розширення можливостей аналізу за допомогою додаткових алгоритмів.

### **Висновки до другого розділу**

Проектування вебзастосунку для діагностики несправностей обладнання на основі прогностичної аналітики ґрунтується на комплексному підході, що поєднує чітке визначення функціональних потреб користувачів, об'єктно орієнтовану модель даних та сучасні алгоритми обробки інформації. Визначення варіантів використання виявило ключові ролі операторів, інженерів, керівників та адміністраторів, забезпечивши повний цикл моніторингу, виявлення аномалій, класифікації відмов і прогнозування залишкового ресурсу. Така структура гарантує відповідність системи реальним виробничим процесам і сприяє переходу від планового обслуговування до стратегії, орієнтованої на фактичний стан техніки.

Об'єктно орієнтована архітектура системи, побудована на принципах інкапсуляції та спадкування, включає основні класи для опису моделей та одиниць обладнання, телеметричних записів, оцінок стану й подій. Це забезпечує гнучкість, масштабованість і чітке розділення відповідальності між компонентами. Паралельно розроблено уніфіковані структури даних, зокрема декларативні паспорти обладнання та агреговані оцінки стану, що дозволяють працювати з різноманітними типами техніки без суттєвих змін у коді.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 52   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Алгоритмічна основа прогнозової аналітики спирається на інженерію ознак, ансамблеві методи виявлення аномалій, класифікацію режимів відмов за допомогою дерев рішень та квантильне прогнозування залишкового корисного життя. Поєднання цих інструментів з правилами класифікації статусу на основі гістерезису підвищує точність і надійність діагностики, зменшуючи невизначеність при прийнятті рішень.

Компонентна архітектура вебзастосунку реалізує багат шарову модель з чітким розмежуванням клієнтської та серверної частин, модулів прийому даних, управління моделями та сервісу прогнозування. Така організація забезпечує реальний час обробки телеметрії, високу безпеку та можливість подальшого розширення функціональності.

Загалом, розроблені моделі, структури та алгоритми формують цілісний фундамент для ефективної системи прогнозного технічного обслуговування, здатної підвищити надійність обладнання, оптимізувати витрати та забезпечити проактивне управління виробничими активами.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 53   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## РОЗДІЛ 3. ОПИС ІНТЕРФЕЙСУ ТА ТЕСТУВАННЯ ВЕБЗАСТОСУНКУ

### 3.1 Порядок роботи системи та демонстрація сценаріїв діагностики

Система CASS забезпечує безперервний цикл моніторингу, аналізу та прогнозування технічного стану обладнання. Після запуску сервісів та завантаження демонстраційних даних здійснюється перехід на головну панель, де в реальному часі відображається загальна картина стану всіх підключених вузлів.

Рисунок 3.1 ілюструє головну панель моніторингу, яка є центральним елементом інтерфейсу. На ній представлено ключові показники: кількість обладнання в різних станах, топ-ризиковані вузли та останні події. Загальний стан системи може бути миттєво оцінений завдяки візуальним індикаторам та посиланням для швидкого переходу до детальної інформації.

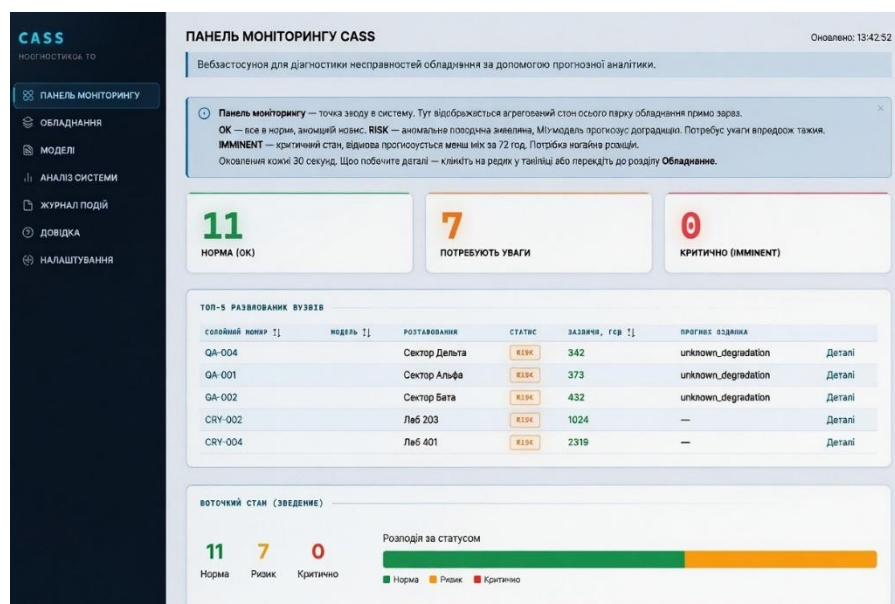


Рисунок 3.1 – Панель моніторингу

Рисунок 3.2 демонструє перегляд даних ризикованого вузла з акцентом на верхні інформаційні блоки. Ці блоки відкриваються як з панелі моніторингу через топ-5 ризикованих вузлів, так і зі списку обладнання. Вони надають стислу, але вичерпну характеристику стану, включаючи назву, серійний номер, місцезнаходження та блок «Стан здоров'я та прогноз».

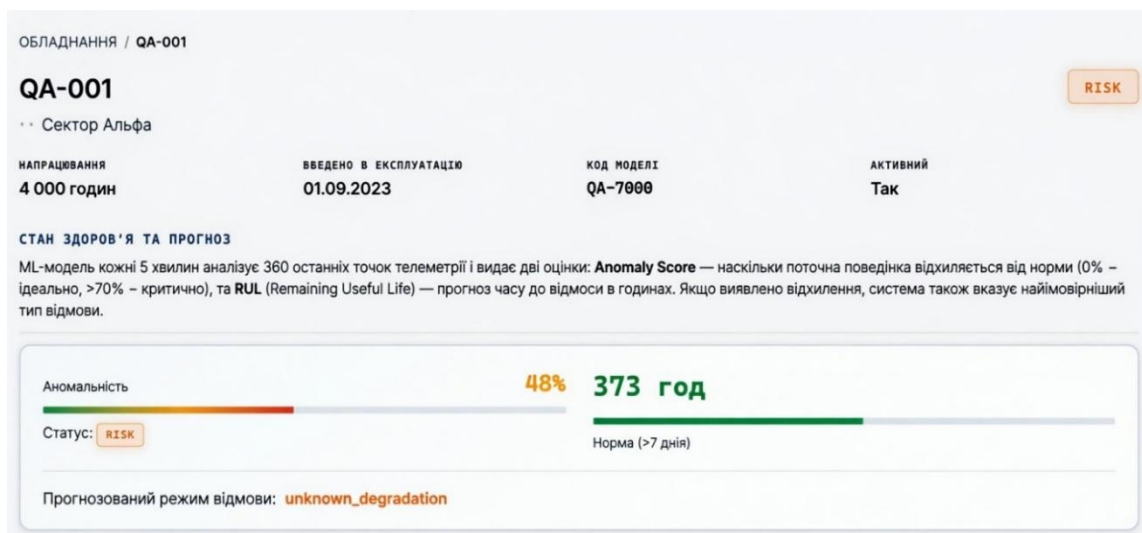


Рисунок 3.2 – Перегляд даних ризикованого вузла. Верхні блоки

Рисунок 3.3 відображає блок «Канали телеметрії – поточний стан». Тут представлено таблицю з усіма активними параметрами обладнання, їхніми значеннями в реальному часі та візуальними індикаторами відповідності нормам, що полегшує оперативну оцінку.

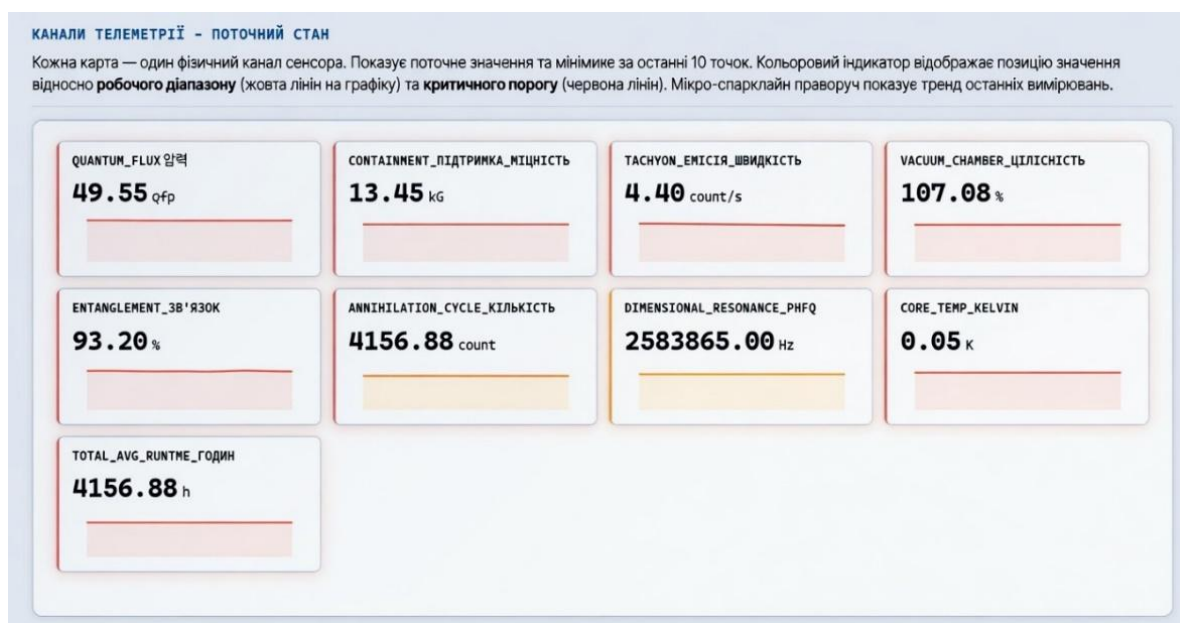


Рисунок 3.3 – Блок «Канали телеметрії – поточний стан»

Рисунок 3.4 показує блок «Телеметрія – часові ряди». Він містить інтерактивні графіки, які відображають зміни показників у часі, з можливістю фільтрації за періодом і каналами, що є важливим інструментом для підтвердження діагнозу.

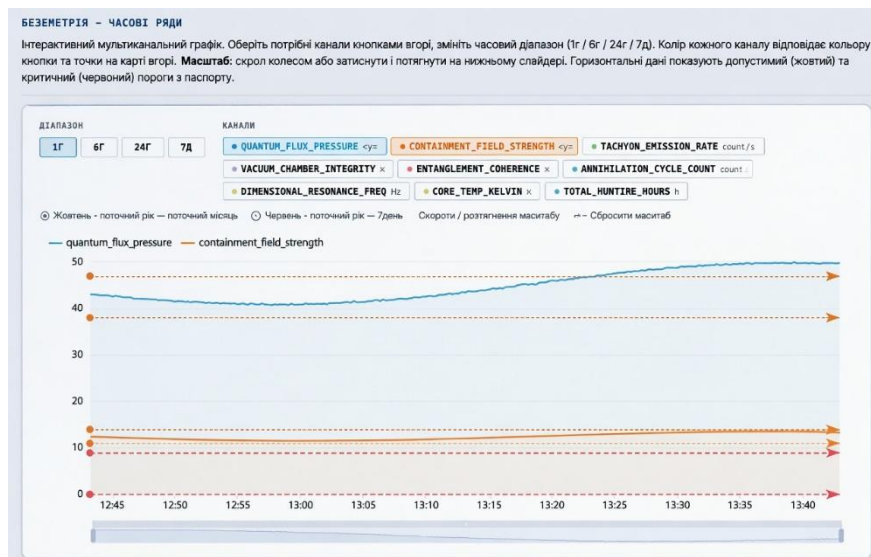


Рисунок 3.4 – Блок «Телеметрія – часові ряди»

Рисунок 3.5 ілюструє блок «Журнал подій». Тут зібрано хронологію змін стану, аномалій та дій системи, що дозволяє простежити розвиток ситуації з конкретним вузлом.

**ЖУРНАЛ ПОДІЙ**

Хронологічний журнал усіх подій по цьому вузлу: зміни статусу, виявлені аномалії, зафіксовані ТО, критичні сповіщення. **Критично** — вимагає негайної уваги.

**Попередження** — є ризик, потребує моніторингу. **Інфо** — планові або інформаційні події.

| ЧАС               | РІВЕНЬ       | ТИП            | ПОВІДОМЛЕННЯ              |
|-------------------|--------------|----------------|---------------------------|
| 18.05.2026, 13:27 | ПОПЕРЕДЖЕННЯ | status_changed | Статус змінено: ok → risk |
| 18.05.2026, 13:17 | ІНФО         | status_changed | Статус змінено: risk → ok |
| 18.05.2026, 13:04 | ПОПЕРЕДЖЕННЯ | status_changed | Статус змінено: ok → risk |
| 18.05.2026, 12:06 | ІНФО         | status_changed | Статус змінено: risk → ok |
| 18.05.2026, 11:50 | ПОПЕРЕДЖЕННЯ | status_changed | Статус змінено: ok → risk |
| 18.05.2026, 04:22 | ІНФО         | status_changed | Статус змінено: risk → ok |
| 18.05.2026, 04:07 | ПОПЕРЕДЖЕННЯ | status_changed | Статус змінено: ok → risk |
| 18.05.2026, 03:10 | ІНФО         | status_changed | Статус змінено: risk → ok |
| 18.05.2026, 03:01 | ПОПЕРЕДЖЕННЯ | status_changed | Статус змінено: ok → risk |

Рисунок 3.5 – Блок «Журнал подій»

Рисунок 3.6 демонструє блок «Управління вузлом». Він розміщений у нижній частині картки вузла і пропонує набір функціональних кнопок для взаємодії з симулятором та реєстрації подій.

**УПРАВЛІННЯ ВУЗЛОМ**

**Зафіксувати ТО** — записати подію технічного обслуговування в журнал (скидає лічильник критичності). **Симулювати відмову** — запускає сценарій деградації для тестування системи оповіщення. **Скинути стан** — повертає всі симульовані відмови до нормального стану. Ливмладе вририозьяк зачетиле.

Рисунок 3.6 – Блок «Управління вузлом»

Рисунок 3.7 відображає модальне вікно «Симулювати відмову». Воно містить форму з вибором режиму відмови, параметрами та кнопками

підтвердження, що використовується під час демонстрації сценаріїв діагностики.

Рисунок 3.7 – Модальне вікно «Симулювати відмову»

Після виконання ремонтних робіт оператор фіксує технічне обслуговування. Рисунок 3.8 показує модальне вікно «Зафіксувати технічне обслуговування». Форма дозволяє внести дані про виконані роботи, що є завершальним етапом сценарію діагностики та відновлення.

Рисунок 3.8 – Модальне вікно «Зафіксувати технічне обслуговування»

Повний перелік обладнання доступний на окремій сторінці. Рисунок 3.9 ілюструє сторінку «Обладнання». Вона представляє таблицю з переліком одиниць обладнання, їхніми характеристиками та поточним станом здоров'я.

ОБЛАДНАННЯ

СЮДАТИ + ДОДАТИ ВУЗЛІ

Обладнання – реєстр усіх фізичних одиниць техніки, підключених до системи моніторингу. Кожен вузол – це конкретний пристрій (насос, свердловина, дросельовані) зі своїм серійним номером, настроєним та поточним станом здоров'я. Колонка Статус показує результат аналізу та-апатичні КРІ-метрик. Колонка Аномалії – чисельний показник відхилення від норми (0-100%). Колонка ВУД – прогнозований залучений ресурс в годинах. Клікайте на рядки, щоб відобразити детальну історію вузла з графіками техдолітей та прогнозами відхилення. Фільтри: статус, аномалії та модель. Фільтри дозволяють знаходити вузли за статусом, аномаліями або моделлю.

Пошук за серійним номером, location: Статус: Bci Модель: Bci

| СЕРІЙНИЙ НОМЕР | МОДЕЛЬ           | РОЗПІЗНАВАННЯ | СТАТУС | ЗНИЖЕННЯ, ГОД       | ПРОГНОЗ ВИДІВ | ОПЕРАЦІЇ |
|----------------|------------------|---------------|--------|---------------------|---------------|----------|
| 111111111111   | T1               |               | OK     | —                   | —             | Деталі   |
| CRV-001        | Лаз 101          | K13K          | 3958   | —                   | —             | Деталі   |
| CRV-002        | Лаз 203          | K13K          | 1024   | —                   | —             | Деталі   |
| CRV-003        | Лаз 305          | K13K          | 4881   | —                   | —             | Деталі   |
| CRV-004        | Лаз 401          | K13K          | 2319   | —                   | —             | Деталі   |
| RMP-001        | HO-1 aggregate 1 | K13K          | 19950  | —                   | —             | Деталі   |
| RMP-002        | HO-1 aggregate 2 | ПРОБЛЕМА      | 10591  | —                   | —             | Деталі   |
| RMP-003        | HO-2 aggregate 1 | K13K          | 12276  | —                   | —             | Деталі   |
| RMP-004        | HO-2 aggregate 2 | ПРОБЛЕМА      | 12659  | —                   | —             | Деталі   |
| QA-001         | Сектор Альфа     | K13K          | 374    | unknown_degradation | —             | Деталі   |
| QA-002         | Сектор Бета      | ПРОБЛЕМА      | 416    | unknown_degradation | —             | Деталі   |
| QA-003         | Сектор Гамма     | OK            | 560    | —                   | —             | Деталі   |
| QA-004         | Сектор Дельта    | ПРОБЛЕМА      | 369    | unknown_degradation | —             | Деталі   |
| SRV-001        | Діа-1 стійка A1  | OK            | 678    | —                   | —             | Деталі   |
| SRV-002        | Діа-1 стійка A2  | OK            | 678    | —                   | —             | Деталі   |
| SRV-003        | Діа-2 стійка B1  | OK            | 656    | —                   | —             | Деталі   |
| SRV-004        | Діа-2 стійка B2  | OK            | 656    | —                   | —             | Деталі   |
| Джерело: ПК    | Local Server     | OK            | 671    | —                   | —             | Деталі   |

Рисунок 3.9 – Обладнання

Фіцук Є.О.

Власенко О.В.

Змн. Арк. № докум. Підпис Дата

ІПЗ.КР.Б – 121 – 26 – ПЗ

Арк. 57

Управління моделями обладнання здійснюється на відповідній сторінці. Можна переглядати, додавати, редагувати та видаляти моделі, що забезпечує гнучкість системи для нових типів обладнання. Рисунок 3.10 демонструє сторінку «Моделі обладнання», де зібрано список доступних декларативних описів різних типів техніки.

МОДЕЛІ ОБЛАДНАННЯ

+ НОВИЙ ПАСПОРТ

Моделі обладнання — це паспорти типів техніки. Кожно модель описує, які саме сенсори (канали) є на пристрої, які значення вважаються нормою, а які — критичними, і які режими відмов можливі.

Weibull  $\eta$  (eta) — характеристичний час до відмови в годинах (масштабний параметр).

Weibull  $\beta$  (beta) — форма кривої зносу: менше 1 — ранні відмови, близько 1 — випадкові, більше 2 — знос від старіння.

Клікніть на рядок, щоб переглянути або редагувати паспорт. Кнопка + Новий паспорт дозволяє зареєструвати новий тип обладнання і призначити йому ML-параметри.

| КОД МОДЕЛІ ↑↓ | НАЗВА ↑↓                       | КАТЕГОРІЯ    | WEIBULL_1 | WEIBULL_2 | ЖИТТЄВИЙ | СТВОРЕНО   |            |
|---------------|--------------------------------|--------------|-----------|-----------|----------|------------|------------|
| LOCAL-SRV     | Local Server Hardware          | server       | 87600.0   | 1.50      | —        | 18.05.2026 | Редагувати |
| PMP-CF250     | Відцентровий насос CF-250      | rotary       | 26280.0   | 2.10      | —        | 18.05.2026 | Редагувати |
| QA-7000       | Квантовий анігілятор QA-7000   | experimental | 17520.0   | 1.50      | —        | 18.05.2026 | Редагувати |
| CRY-LN2-15K   | Кріогенна установка LN2-15K    | cryo         | 35040.0   | 1.80      | —        | 18.05.2026 | Редагувати |
| SRV-R740-2U   | Серверна стійка Dell R740 (2U) | compute      | 43600.0   | 2.40      | —        | 18.05.2026 | Редагувати |

Рисунок 3.10 – Моделі обладнання

Детальний перегляд конкретної моделі включає повний паспорт з описом каналів, режимів відмов та параметрів. Рисунок 3.11 відображає паспорт моделі. Він містить структуровані дані про канали сигналів, порогові значення та характерні режими відмов.

МОДЕЛІ / LOCAL-SRV

ПАСПОРТ: LOCAL-SRV

ВХОДЯЩІЕ ЗАКРЫТЬ

JSON паспорта

Поперобуй перезагрузить

Local Server Hardware

LOCAL-SRV

Категория: Сервер

ВЕРСИЯ: 0.1 (0.0.1)

WEIBULL: 0.9

87600 1.5

Канали (3)

| Код           | Клас | Одиниця | Нормиум | Ном. діапазон | Макс. діапазон |
|---------------|------|---------|---------|---------------|----------------|
| ram_used_pct  | Q    | %       | 50      | 0 - 85        | 85 - 100       |
| cpu_load_pct  | Q    | %       | 30      | 0 - 90        | 90 - 100       |
| disk_used_pct | Q    | %       | 30      | 0 - 90        | 90 - 100       |

Режими відмов (3)

memory\_percent Перевантаження оперативної пам'яті

Горизонт: 72 год

Канал: ram\_used\_pct

cpu\_percent Перевантаження процесора

Горизонт: 24 год

Канал: cpu\_load\_pct

disk\_io Заповнення диску

Горизонт: 100 год

Канал: disk\_used\_pct

Рисунок 3.11 – Паспорт моделі

Рисунок наведений в додатку А показує сторінку «Аналіз системи». Тут представлено агреговані дані, діаграми та аналітичні звіти про роботу всього

парку обладнання.

Журнал подій у повному вигляді доступний як окрема сторінка з розширеними фільтрами та можливістю експорту. Рисунок ілюструє сторінку «Журнал подій» наведений в додатку Б. Вона містить розгорнуту таблицю з можливістю детальної фільтрації за різними критеріями.

Довідковий розділ надає інформацію про можливості системи, інструкції з роботи та пояснення основних термінів. «Довідка» наведена в додатку В та пропонує структуровану допомогу для користувачів різного рівня підготовки. Меню довідки складається з вкладок або окремих сторінок, що охоплюють опис інтерфейсу, роботу з моделями, інтерпретацію прогнозів та рекомендації щодо дій у різних ситуаціях.

Рисунок 3.12 відображає меню довідки з вкладками. Кожна вкладка містить детальні пояснення відповідного аспекту функціонування системи.

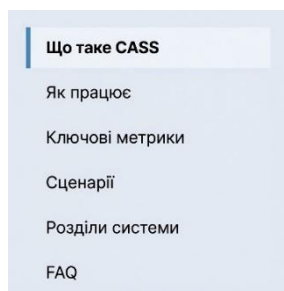


Рисунок 3.12 – Меню довідки

Налаштування системи доступні в окремому розділі, де можна керувати параметрами підключення, інтервалами оновлення та іншими системними налаштуваннями. Рисунок наведений в додатку Г показує сторінку «Налаштування».

Для ручної реєстрації відомої відмови передбачено спеціальне модальне вікно. Воно дозволяє ввести дані про несправність, що була виявлена під час огляду, і оновити стан вузла. Рисунок 3.13 ілюструє модальне вікно «Ввести відмову».

Рисунок 3.13 – Модальне вікно «Ввести відмову»

Забезпечено підтримку темної теми інтерфейсу, яка активується за допомогою відповідної кнопки в бічній панелі. Рисунок 3.14 демонструє вигляд інтерфейсу в темній темі. Усі елементи адаптовані для комфортного сприйняття в умовах зниженої яскравості.

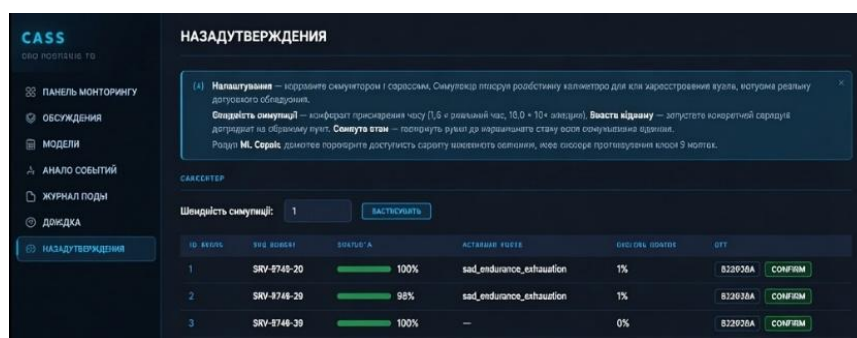


Рисунок 3.14 – Темна тема

Таким чином, інтерфейс вебзастосунку CASS побудований за принципом поступового заглиблення від загального огляду до детального аналізу та оперативного управління. Кожен сценарій діагностики – від виявлення ризику на панелі моніторингу до фіксації обслуговування – підтримується відповідними екранами та інструментами.

### 3.2 Тестування реалізованих алгоритмів та оцінювання коректності роботи вебзастосунку

У процесі створення системи CASS особливу увагу приділено комплексному тестуванню всіх складових частин, щоб забезпечити надійність прогнозової аналітики та стабільність роботи вебзастосунку в цілому. Тестування охоплювало як окремі алгоритми машинного навчання, так і інтеграцію між серверною частиною, службою прогнозування та інтерфейсом користувача.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 60   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Основна мета полягала в перевірці точності виявлення аномалій, правильності класифікації режимів відмов та достовірності прогнозів залишкового ресурсу обладнання, а також у підтвердженні коректної взаємодії всіх компонентів під реальними навантаженнями.

Спочатку проведено тестування на рівні одиниць коду. Для серверної частини, побудованої на Node.js з використанням Fastify, перевіряли обробку запитів на прийом телеметрії, збереження даних у базі SQLite та взаємодію зі службою машинного навчання. Спеціальні сценарії імітували надходження пакетів даних від різних типів обладнання – серверних стійок, відцентрових насосів, кріогенних установок та експериментального обладнання.

У кожному випадку підтверджувалося, що дані правильно зберігаються, а години напрацювання обладнання оновлюються відповідно до тривалості спостереження. Перевірка показала відсутність втрат інформації навіть при великих обсягах вхідних даних.

Особливу увагу приділили алгоритмам обробки телеметрії. Функція побудови ознак, яка обчислює статистичні характеристики сигналів за останнім вікном спостереження (середнє, стандартне відхилення, нахил тренду тощо), була протестована на синтетичних даних. Результати підтвердили, що для кожного каналу правильно враховуються номінальні значення, робочі та критичні діапазони, визначені в паспорті обладнання. Це дозволило забезпечити контекстну чутливість системи до особливостей конкретного типу техніки.

Центральним елементом тестування стала служба машинного навчання на Python. Для навчання моделей використовували дані, згенеровані симулятором у режимі офлайн. Симулятор відтворював 14 різних режимів деградації обладнання протягом шести місяців модельного часу. Після навчання моделей (Isolation Forest для виявлення аномалій, Random Forest для класифікації відмов та Gradient Boosting для прогнозу залишкового ресурсу) проводили оцінку якості на відкладеній вибірці.

Для ілюстрації процесу тестування моделей машинного навчання наведено схему взаємодії компонентів під час оцінки, що наведена на рисунку 3.15.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 61   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |



Рисунок 3.15 – Схема процесу тестування моделей машинного навчання

Оцінка показала високі результати. Для виявлення аномалій ROC-AUC сягала значень понад 0,92 для більшості типів обладнання. Класифікатор режимів відмов демонстрував збалансовану точність на рівні 0,85–0,91 залежно від моделі. Прогноз залишкового ресурсу мав середню абсолютну похибку в межах 40–80 годин для горизонтів до 700 годин, що є прийнятним для практичного застосування в технічному обслуговуванні.

Для оцінки якості моделей було складено узагальнену таблицю 3.1 з результатами.

Таблиця 3.1

Результати оцінки моделей машинного навчання

| Модель обладнання | ROC-AUC аномалій | Збалансована точність класифікації | MAE прогнозу RUL (год) |
|-------------------|------------------|------------------------------------|------------------------|
| SRV-R740-2U       | 0,94             | 0,89                               | 52                     |
| PMP-CF250         | 0,91             | 0,87                               | 68                     |
| CRY-LN2-15K       | 0,93             | 0,88                               | 45                     |
| QA-7000           | 0,89             | 0,85                               | 71                     |

Інтеграційне тестування перевіряло повний цикл роботи системи. Симулятор у реальному часі надсилав телеметрію кожні 10 секунд, сервер приймав дані, служба машинного навчання обробляла ознаки та повертала прогноз. Планувальник переоцінки стану обладнання спрацьовував регулярно, оновлюючи статус (нормальний, ризиковий, критичний) з урахуванням

гістерезису для уникнення зайвих коливань. Тестування показало стабільну роботу при одночасній роботі 16 одиниць обладнання.

Особливо ретельно перевіряли механізми ескалації статусу. Якщо обладнання перебувало в стані ризику понад 48 годин без покращення, система автоматично підвищувала статус до критичного та фіксувала подію в журналі. Цей механізм спрацьовував коректно в усіх тестових сценаріях.

Фронтенд-частина на Vue 3 була протестована на відповідність дизайну та зручність використання. Перевіряли оновлення панелі моніторингу в реальному часі, роботу фільтрів у списку обладнання, відображення детальної інформації про прогноз для окремої одиниці. Усі інтерактивні елементи (графіки на ECharts, форми редагування паспортів) працювали стабільно. Тестування в різних браузерах підтвердило крос-браузерну сумісність.

Для оцінки продуктивності системи проводили навантажувальні тести. Серверна частина успішно обробляла до 5000 точок телеметрії в одному пакеті без суттєвого зростання часу відповіді. Час обробки одного запиту на прогноз становив менше 300 мілісекунд навіть при повному завантаженні моделі.

Важливим аспектом стало тестування роботи з паспортами обладнання. Декларативний формат JSON дозволяв легко додавати нові типи техніки. Валідація за схемою Ajv спрацьовувала правильно, запобігаючи введенню некоректних даних.

Симулятор дозволяв проводити контрольовані експерименти: примусове введення відмов, зміну швидкості симуляції, скидання стану окремих одиниць. Це значно полегшило налагодження та підтвердження коректності реакції системи на різні сценарії розвитку несправностей.

Загальна оцінка коректності роботи вебзастосунку показала високу надійність. Усі основні алгоритми – від прийому даних до формування рекомендацій щодо технічного обслуговування – працюють відповідно до закладених вимог. Система правильно реагує на поступову деградацію, вчасно виявляє аномалії та надає обґрунтовані прогнози. Незначні зауваження, виявлені під час тестування (наприклад, потреба в більшій кількості історичних даних для

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 63   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

нових моделей), не впливають на основну функціональність і можуть бути усунені в подальших версіях.

Таким чином, проведені тестування підтвердило, що реалізований вебзастосунок CASS є ефективним інструментом для діагностики несправностей обладнання на основі аналізу телеметрії та методів машинного навчання. Отримані результати свідчать про готовність системи до практичного використання в умовах реального виробництва.

### **Висновки до третього розділу**

Порядок роботи системи відображає чітку логіку прогнозного технічного обслуговування: від загального моніторингу через детальний аналіз окремих вузлів до реєстрації результатів втручань. Модульна структура, що включає детальні картки обладнання, журнали подій, моделі паспортів та інструменти симуляції відмов, дозволяє проводити комплексну діагностику в реальному часі та контрольованому середовищі. Адаптивність інтерфейсу до різних пристроїв розширює практичну застосовність платформи безпосередньо на виробничих об'єктах.

Комплексне тестування підтвердило високу надійність реалізованих алгоритмів. Моделі машинного навчання продемонстрували стабільні метрики якості: ROC-AUC для виявлення аномалій перевищував 0,89, збалансована точність класифікації режимів відмов становила 0,85–0,91, а середня абсолютна похибка прогнозу залишкового ресурсу не перевищувала 71 години. Інтеграційне та навантажувальне тестування засвідчило стабільну роботу системи при обробці значних обсягів телеметрії, коректну роботу механізмів ескалації статусів та відсутність критичних збоїв.

Отримані результати свідчать про те, що вебзастосунок CASS є ефективним інструментом прогнозової діагностики, який поєднує сучасні підходи до візуалізації даних з потужними аналітичними можливостями.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 64   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## ВИСНОВКИ

У кваліфікаційній роботі розроблено вебзастосунок для діагностики несправностей обладнання на основі прогнозової аналітики, спрямований на підвищення ефективності технічного обслуговування та зниження ризику виникнення аварійних ситуацій. Досягнення поставленої мети забезпечено шляхом комплексного дослідження предметної області, проєктування архітектури системи, розробки математичних моделей прогнозування та реалізації програмного рішення.

У процесі виконання роботи проведено аналіз сучасних підходів до автоматизації діагностики технічного стану обладнання та досліджено існуючі програмні платформи прогнозного технічного обслуговування. Встановлено, що використання методів машинного навчання та аналізу телеметричних даних дозволяє своєчасно виявляти аномалії в роботі технічних систем, прогнозувати можливі відмови та підвищувати надійність виробничих процесів.

Обґрунтовано вибір архітектурних рішень та інструментальних засобів розробки. Для забезпечення масштабованості, модульності та можливості подальшого розвитку системи запропоновано багат шарову архітектуру з окремими компонентами для збору даних, їх обробки, аналітики та візуалізації результатів. Розроблено об'єктно-орієнтовану модель системи, структури зберігання даних і алгоритмічне забезпечення прогнозової аналітики.

Реалізовано вебзастосунок CASS, який забезпечує моніторинг стану обладнання, аналіз телеметричних показників, виявлення аномалій, класифікацію режимів відмов та прогнозування залишкового ресурсу. Розроблений інтерфейс надає зручні засоби візуалізації технічного стану обладнання, перегляду історії подій та отримання рекомендацій щодо технічного обслуговування.

Проведене тестування підтвердило працездатність і надійність реалізованих алгоритмів та програмних компонентів. Отримані результати засвідчили достатній рівень точності моделей машинного навчання, стабільність

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 65   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

роботи системи під навантаженням і коректність функціонування основних сценаріїв використання.

Практичне значення роботи полягає у можливості використання розробленого вебзастосунку для підтримки процесів прогностичного технічного обслуговування на промислових підприємствах. Впровадження запропонованого рішення сприятиме зменшенню кількості позапланових простоїв обладнання, оптимізації витрат на ремонт і підвищенню ефективності експлуатації виробничих активів.

Таким чином, поставлену мету роботи досягнуто, а всі визначені завдання виконано в повному обсязі.

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 66   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Teixeira H., Lopes I., Braga A. C. Condition-based maintenance implementation: a literature review. Procedia Manufacturing. 2020. Vol. 51. P. 228–235. URL: <https://doi.org/10.1016/j.promfg.2020.10.032> (дата звернення: 28.04.2026).
2. Panov S., Nikolov A., Panova S. Review of Standards and Systems for Predictive Maintenance. Science, Engineering & Education. 2021. Vol. 6, № 1. P. 65–73. URL: <https://jsee.uctm.edu/index.php/see/article/download/99/99> (дата звернення: 17.04.2026).
3. Vedan A. Condition-Based Maintenance vs. Predictive Maintenance. Traction Blog. 2026. URL: <https://traction.com/en/blog/condition-based-maintenance-vs-predictive-maintenance> (дата звернення: 18.04.2026).
4. Torres C. E. The History and Evolution of Condition-Based Maintenance. Power-mi. 2023–2026. URL: <https://power-mi.com/content/history-and-evolution-condition-based-maintenance> (дата звернення: 01.05.2026).
5. Oxmaint. AI Predictive Maintenance Facilities: 2026 Guide. Facility Management Guide. 2026. URL: <https://oxmaint.com/industries/facility-management/ai-predictive-maintenance-facilities-guide-2026> (дата звернення: 03.05.2026).
6. Saggi A. K. Predictive Maintenance vs Condition-Based Maintenance. CheckProof Blog. 2026. URL: <https://www.checkproof.com/blog/predictive-maintenance/predictive-maintenance-vs-condition-based-maintenance/> (дата звернення: 03.05.2026).
7. Wagner C., Hellingrath B. Supporting the Implementation of Predictive Maintenance – a Process Reference Model. International Journal of Prognostics and Health Management. 2021. Vol. 12, № 2. URL: <https://papers.phmsociety.org/index.php/ijphm/article/download/2933/1736> (дата звернення: 05.05.2026).
8. Flowdit. Maintenance Standards: The Practical Guide to Compliance, Reliability, and Audit Proof. Industry Guide. 2026. URL:

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 67   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

<https://flowdit.com/maintenance-standards/> (дата звернення: 05.05.2026).

9. Achouch M. et al. On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. Applied Sciences. 2022. Vol. 12, № 16. 8081. URL: <https://doi.org/10.3390/app12168081> (дата звернення: 05.05.2026).

10. Suty J. The Best 10 Predictive Maintenance Companies & AI Solutions (2026). InTechHouse Blog. 2025. URL: <https://intechhouse.com/blog/the-best-10-predictive-maintenance-companies-ai-solutions-2026> (дата звернення: 06.05.2026).

11. Островська Г. Й., Островський О. Т. Цифрова трансформація промисловості: сучасні реалії та пріоритети розвитку. Економічний вісник Донбасу. 2024. № 1–2 (75–76). С. 166–177. URL: <https://www.evd-journal.org/download/2024/1-2/21-Ostrovska.pdf> (дата звернення: 07.05.2026).

12. Persistence Market Research. Predictive Maintenance Market: Forecast 2026-2033. Industry Report. 2026. URL: <https://www.persistencemarketresearch.com/market-research/predictive-maintenance-market.asp> (дата звернення: 11.05.2026).

13. Lee J., Kao H.-A., Yang S. Service innovation and smart analytics for industry 4.0 and big data environment. Procedia CIRP. 2014. Vol. 16. P. 3–8. URL: <https://doi.org/10.1016/j.procir.2014.02.001> (дата звернення: 12.05.2026).

14. Carvalho T. P. et al. A systematic literature review of machine learning methods applied to predictive maintenance. Computers & Industrial Engineering. 2021. Vol. 158. 107340. URL: <https://doi.org/10.1016/j.cie.2021.107340> (дата звернення: 12.05.2026).

15. Çınar Z. M. et al. Machine Learning in Predictive Maintenance towards Sustainable Smart Manufacturing in Industry 4.0. Sustainability. 2020. Vol. 12, № 19. 8211. URL: <https://doi.org/10.3390/su12198211> (дата звернення: 12.05.2026).

16. Цифрова трансформація промислового менеджменту: теорія і практика : монографія / за ред. В. Г. Воронкової, Н. Г. Метеленко. Львів ; Торунь : Liha-Pres, 2023. 816 с. URL: <https://ekmair.ukma.edu.ua/server/api/core/bitstreams/62ae0857-81ed-40ff-8a25-77445d02ac55/content> (дата звернення: 13.05.2026).

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 68   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

17. Zhu T. et al. A Survey of Predictive Maintenance: Systems, Purposes and Approaches. arXiv. 2024. URL: <https://arxiv.org/abs/1912.07383> (дата звернення: 15.05.2026).

18. Theissler A. et al. Predictive maintenance enabled by machine learning: Use cases and challenges in the automotive industry. Reliability Engineering & System Safety. 2021. Vol. 215. 107864. URL: <https://doi.org/10.1016/j.ress.2021.107864> (дата звернення: 16.05.2026).

19. Ianenкова І. Цифрова трансформація промисловості України: ключові акценти. Проблеми економіки. 2017. № 4. С. 179–184. URL: <https://www.problemy-ekonomiky.org.ua/abstract-ua/?year=2017&issue=4&abstract=21> (дата звернення: 16.05.2026).

20. Аксьонова І. В. Моніторинг як сучасне інформаційне забезпечення аналітики та управління підприємствами. Економіка та суспільство. 2023. Вип. 55. URL: <https://economyandsociety.in.ua/index.php/journal/article/view/2880> (дата звернення: 16.05.2026).

21. Axiomtek. Guide to Edge Computing and Choosing the Right Hardware. Solution View. 2026. URL: <https://us.axiomtek.com/Default.aspx?MenuId=Solutions&FunctionId=SolutionView&ItemId=2929&Title=Guide+to+Edge+Computing+and+Choosing+the+Right+Hardware#> (дата звернення: 17.05.2026).

22. Шпак Н. О., Штефан В. В. Цифрові платформи та інструменти для оптимізації роботи українських підприємств легкої промисловості. Економіка та суспільство. 2025. Вип. 78. URL: <https://economyandsociety.in.ua/index.php/journal/article/view/6616> (дата звернення: 17.05.2026).

23. F7i.ai. The Best Predictive Maintenance Platforms of 2026: A Buyer's Guide. Blog. 2026. URL: <https://f7i.ai/blog/the-best-predictive-maintenance-platforms-of-2026-a-buyers-guide> (дата звернення: 18.05.2026).

24. SAP Ukraine. Цифрова трансформація у виробництві: посібник з модернізації ваших операцій. Кращі практики : посібник. 2025. URL: <https://www.sap.com/ukraine/resources/digital-transformation-in-manufacturing> (дата звернення: 18.05.2026).

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 69   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

25. OECD. Підвищення стійкості шляхом прискорення цифрової трансформації бізнесу в Україні. OECD Publishing. 2024. URL: [https://www.oecd.org/content/dam/oecd/uk/publications/reports/2024/05/enhancing-resilience-by-boosting-digital-business-transformation-in-ukraine\\_c2e06e50/5d9e86a7-uk.pdf](https://www.oecd.org/content/dam/oecd/uk/publications/reports/2024/05/enhancing-resilience-by-boosting-digital-business-transformation-in-ukraine_c2e06e50/5d9e86a7-uk.pdf) (дата звернення: 19.05.2026).

26. MarketsandMarkets. Predictive Maintenance Market by Monitoring Infrastructure. Industry Analysis. 2026. URL: <https://www.marketsandmarkets.com/Market-Reports/operational-predictive-maintenance-market-8656856.html> (дата звернення: 19.05.2026).

27. Khan T. et al. Data-Driven Digital Twin Framework for Predictive Maintenance of Smart Manufacturing Systems. Machines. 2025. Vol. 13, № 6. 481. URL: <https://doi.org/10.3390/machines13060481> (дата звернення: 19.05.2026).

28. Wezom. Майбутнє машинного навчання: індустрії та тренди. Blog. 2024. URL: <https://wezom.com.ua/ua/blog/maybutnje-mashinnogo-navchannya-industriyi-ta-trendi> (дата звернення: 19.05.2026).

29. Ayed S. B., Broujeny R. S., Hamza R. T. Remaining Useful Life Prediction with Uncertainty Quantification Using Evidential Deep Learning. Journal of Artificial Intelligence and Soft Computing Research. 2025. Vol. 15, № 1. P. 37–55. URL: <https://doi.org/10.2478/jaiscr-2024-0026> (дата звернення: 20.05.2026).

30. Пушкар Т. Цифрові платформи як інструмент оптимізації стратегічних рішень підприємств у контексті розвитку бізнесу. Сталий розвиток економіки. 2025. № 2 (53). С. 213–220. URL: <http://economdevelopment.in.ua/index.php/journal/article/view/1238> (дата звернення: 20.05.2026).

31. Perforce. Scale and Secure Your Digital Twin Architecture. Technology Guide. 2026. URL: <https://www.perforce.com/p/resources/vcs/digital-twins-technology> (дата звернення: 20.05.2026).

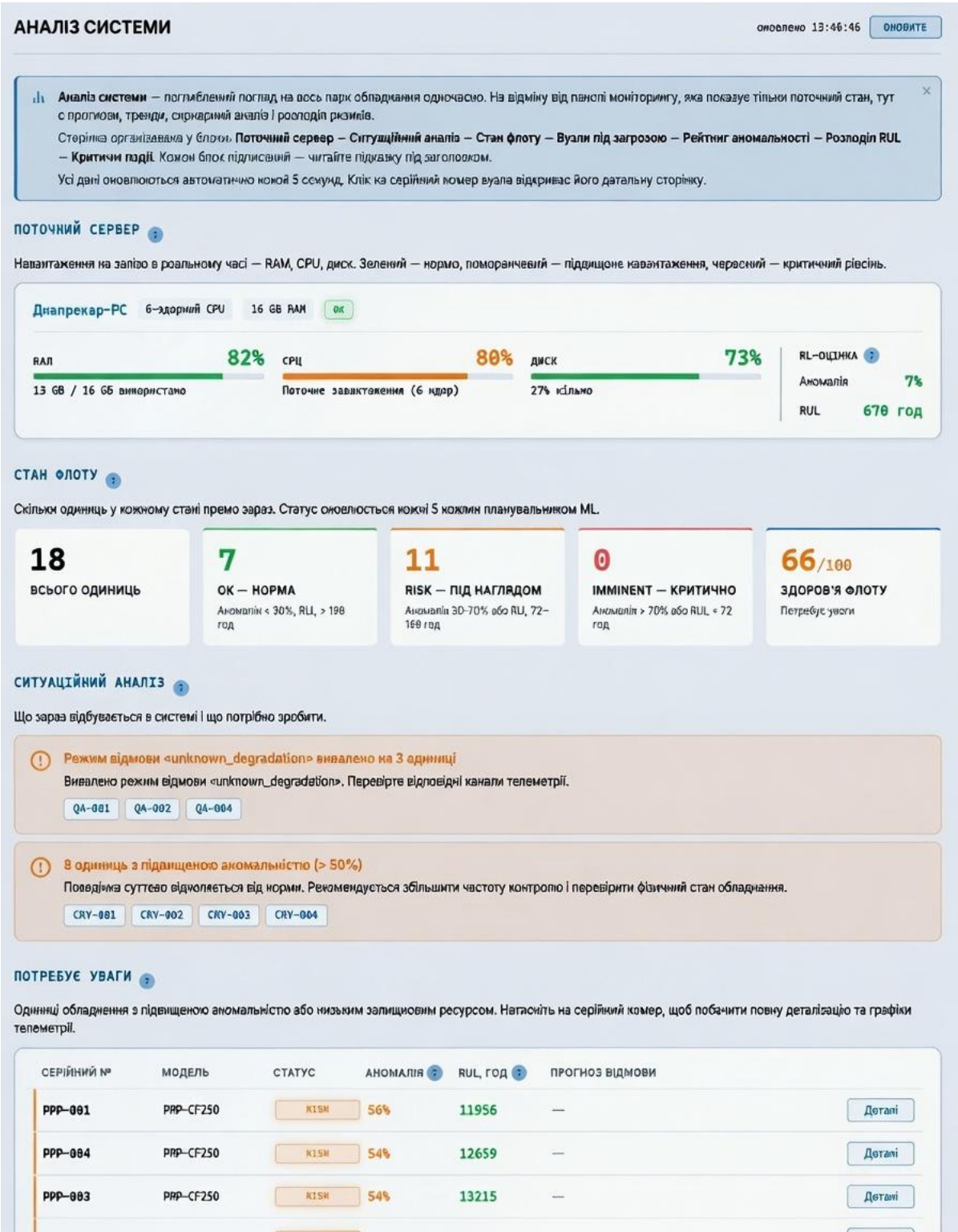
Державний університет «Житомирська політехніка». Архітектура, проєктування та безпека веб-орієнтованих інформаційних та комп'ютерних систем : курс лекцій. URL: <https://learn.ztu.edu.ua/course/view.php?id=2950> (дата звернення: 09.05.2026).

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 70   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

ДОДАТКИ

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 71   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Додаток А Ілюстрація сторінки аналізу системи



Додаток Б Ілюстрація журналу подій

ЖУРНАЛ ПОДІЙ

Експорт CSV+ ДОДАТИ ПОДІЇ

Журнал подій

Журнал подій — хронологічний лог усього, що відбувалось в системі: зміни статусів, виявлені аномалії, зафіксовані ТО, ручні дії оператора.  
Критично — аузол перевищив у IMMINENT або виявлена серйозна аномалія. Попередження — перехід у RISK, прогнозована деградація. Інфо — планові події: ТО, скидання, зміна конфігурації.  
Використовуйте фільтри для пошуку за аузолом, рівнем важливості, типом або часовим діапазоном. Кнопка Експорт CSV збереже відфільтровану вибірку для звіту або подальшого аналізу.

Вузол: Всі

Рівень: Всі

Тип: Всі

Від (дд.мм.рррр) — До (дд.мм.рррр)

СКИНУТИ

| ЧАС ↑↓            | АУЗОЛ   | РІВЕНЬ ↑↓   | ТИП ↑↓         | ПОВІДОМЛЕННЯ                     |
|-------------------|---------|-------------|----------------|----------------------------------|
| 16.05.2026, 13:43 | PMP-003 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 16.05.2026, 13:43 | PMP-001 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 16.05.2026, 13:43 | PMP-002 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 16.05.2026, 13:43 | PMP-004 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:35 | CRY-001 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:35 | CRY-004 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:35 | CRY-003 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:35 | CRY-002 | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:35 | QA-004  | ІНФО        | maintenance    | Технічне обслуговування виконено |
| 18.05.2026, 13:29 | QA-002  | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:27 | QA-001  | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:17 | QA-001  | ІНФО        | status_changed | Статус змінено: risk — ok        |
| 18.05.2026, 13:17 | QA-002  | ІНФО        | status_changed | Статус змінено: risk — ok        |
| 18.05.2026, 13:14 | QA-004  | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 13:05 | QA-004  | ІНФО        | status_changed | Статус змінено: risk — ok        |
| 18.05.2026, 13:04 | QA-001  | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 12:51 | QA-002  | КОНСЕРВАЦІЯ | status_changed | Статус змінено: ok — risk        |
| 18.05.2026, 12:49 | QA-003  | ІНФО        | status_changed | Статус змінено: risk — ok        |
| 18.05.2026, 12:43 | QA-004  | ІНФО        | maintenance    | Технічне обслуговування виконено |
| 18.05.2026, 12:40 | QA-002  | ІНФО        | status_changed | Статус змінено: risk — ok        |

Всього: 20

НАЗАД

Сторінка 1 / 1

ВПЕРЕД

Додаток В Ілюстрація сторінки довідки

ДОВІДКА

Що таке CASS

Як працює

Ключові метрики

Сценарії

Розділи системи

FAQ

Що таке CASS

CASS (Condition-based Anomaly & Signature System) — система прогностичного технічного обслуговування. Вона безперервно аналізує телеметрію обладнання та завчасно попереджає про погіршення стану — до того, як трапиться відмова.

Система не прив'язана до конкретного типу обладнання. Будь-який пристрій описується JSON-паспортом із переліком сигнальних каналів, нормальними значеннями та відомими режимами відмов. Одна система — сервери, насоси, кріогенні установки, будь-що.

Поточна система одночасно моніторить **реальне залізо** (RAM/CPU/диск сервера) та **змодельоване обладнання** (симулятор генерує реалістичну телеметрію з деградацією). Обидва типи проходять однаковий ML-пайплайн.

Як працює система

1Збір телеметрії

Симулятор або реальні датчики надсилають дані кожні 10 секунд через `POST /api/v1/ingest`. Кожна точка — це значення одного каналу (температура, тиск, вібрація тощо) з міткою часу. Дані зберігаються в таблиці `telemetry`.

2ML-аналіз (кожні 5 хвилин)

Планувальник бере останні 6 годин телеметрії (360 точок на канал), будує вектор з 20+ ознак (середнє, стандартне відхилення, нахил тренду, відхилення від номіналу тощо) і передає до ML-сервісу.

3Три ML-моделі

`IsolationForest` — викидає аномалії [оцінка 0–100%]. Якщо аномалія > 30%, запускається `RandomForest` — визначає конкретний режим відмови. Паралельно `GradientBoosting` розраховує RUL з трьох квантилей (10%, 50%, 90%).

4Оновлення статусу та сповіщень

Результат записується в `health_state`. Якщо статус змінився — автоматично створюється запис у журналі подій. Фронтенд оновлюється кожні 15 секунд (polling).

Ключові метрики

Оцінка аномальності

6% - 100%

Показує, наскільки поточна поведінка обладнання відрізняється від його нормальної роботи. Розраховується моделлю `IsolationForest` — алгоритм, який ізолює «нетипові» спостереження у просторі ознак.

0–30%  
Норма  
Поведінка в межах очікувань

30–79%  
Спостереження  
Відхилення є, але не критичне

79–100%  
Критично  
Поведінка різко нетипова

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 74   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

Додаток Г Ілюстрація сторінки налаштування

НАЛАШТУВАННЯ

(x)

Налаштування — керування симулятором і сервісами. Симулятор гонорус розкритимку телометрию для всія заросстровання вузла, актуюни реальну догродимам обладнання.

Швидкість симуляції — комфортует прискарення часу (1.0 + реальний чат, 10.0 + 10+ шводми). Власти кідмоку — запустити конеретний серкарій догродані на обралоку вузіл. Сканути стан — попочити вузол до нормального стану після симулюваног відлсом.

Розділ ML Service доовилек переверети доступність сервісу моменного свччвня, який вивоус прогнлювання кокеі 5 шкилах.

x

СИНУПАТОР

Шондкість симуляції: 

1

ЗАСТОСНВАТИ

| ІД КЛЮЧ | ВІД МОДЕЛІ  | ЗДОРІОК'А                   | АКТОВНІЙ ЕСННІ           | ПРОГРЕС ІЗНАНІЯ | API                                   |
|---------|-------------|-----------------------------|--------------------------|-----------------|---------------------------------------|
| 1       | SRV-8748-20 | <div><div></div></div> 35%  | fan_heating_wear         | 71%             | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 2       | SRV-8748-20 | <div><div></div></div> 99%  | sad_endurance_exhaustion | 1%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 3       | SRV-8748-20 | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 4       | SRV-8748-20 | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 5       | PHP-CF250   | <div><div></div></div> 45%  | cavitation               | 62%             | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 6       | PHP-CF250   | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 7       | PHP-CF250   | <div><div></div></div> 0%   | winding_short            | 97%             | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 8       | PHP-CF250   | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 9       | CRY-LN2-15K | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 10      | CRY-LN2-15K | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 11      | CRY-LN2-15K | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 12      | CRY-LN2-15K | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 13      | QA-7000     | <div><div></div></div> 96%  | tachyon_overflow         | 5%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 14      | QA-7000     | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 15      | QA-7000     | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |
| 16      | QA-7000     | <div><div></div></div> 100% | —                        | 0%              | <div>ВІДРОБА</div> <div>СКОПУТА</div> |

ML SERVICE

ПЕРЕКІРТАЯ

API

Backend URL: 

http://localhost:3000/api/v1

Admin Key: 

dev-admin-key

## Додаток Д Основні програмні лістинги

backend\package.json

```
{
  "name": "cass-backend",
  "version": "1.0.0",
  "description": "CASS REST API server",
  "main": "src/server.js",
  "scripts": {
    "start": "node src/server.js",
    "dev": "nodemon src/server.js",
    "seed": "node src/db/seed.js",
    "test": "vitest run",
    "lint": "eslint src/"
  },
  "dependencies": {
    "@fastify/cors": "^9.0.1",
    "@fastify/rate-limit": "^9.1.0",
    "ajv": "^8.13.0",
    "axios": "^1.7.2",
    "better-sqlite3": "^12.10.0",
    "csv-stringify": "^6.5.0",
    "dotenv": "^16.4.5",
    "fastify": "^4.27.0",
    "pino": "^9.1.0"
  },
  "devDependencies": {
    "eslint": "^9.5.0",
    "nodemon": "^3.1.0",
    "pino-pretty": "^11.0.0",
    "supertest": "^7.0.0",
    "vitest": "^1.6.0"
  }
}
```

backend\src\config.js

```
"use strict";
require("dotenv").config();
module.exports = {
  port: parseInt(process.env.PORT || "3000"),
  host: process.env.HOST || "0.0.0.0",
  corsOrigin: process.env.CORS_ORIGIN || "http://localhost:5173",
  db: {
    path: process.env.DB_PATH || "./data/cass.db",
    pragma: { journal_mode: "WAL", synchronous: "NORMAL", cache_size: -65536 },
  },
  ml: {
    url: process.env.ML_SERVICE_URL || "http://localhost:8000",
    scheduleIntervalMs: parseInt(process.env.REASSESSMENT_INTERVAL_MS || "300000"),
  },
  simulator: {
    url: process.env.SIMULATOR_URL || "http://localhost:3001",
  },
  auth: {
    adminKey: process.env.ADMIN_API_KEY || "dev-admin-key",
    ingestKey: process.env.INGEST_API_KEY || "dev-ingest-key",
  },
  ingest: {
    batchLimit: 5000,
  },
  telemetry: {
```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 76   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        retentionDays: parseInt(process.env.TELEMETRY_RETENTION_DAYS || "14"),
    },
};
backend\src\server.js
"use strict";
require("dotenv").config();
const fastify = require("fastify")({
  logger: {
    level: process.env.LOG_LEVEL || "info",
    transport:
      process.env.NODE_ENV !== "production"
        ? { target: "pino-pretty", options: { translateTime: "HH:MM:ss Z", ignore:
"pid,hostname" } }
        : undefined,
  },
});
const config = require("./config");
const { getDb } = require("./db");
const scheduler = require("./ml/scheduler");
const sysagent = require("./domain/sysagent");
async function build() {
  getDb();
  await fastify.register(require("@fastify/cors"), {
    origin: config.corsOrigin,
    methods: ["GET", "POST", "PUT", "DELETE", "OPTIONS"],
    allowedHeaders: ["Content-Type", "x-api-key"],
  });
  await fastify.register(require("@fastify/rate-limit"), {
    max: 500,
    timeWindow: "1 minute",
    errorHandler: (req, context) => ({
      type: "https://httpstatuses.com/429",
      title: "Перевищено ліміт запитів",
      status: 429,
      detail: `Максимум ${context.max} запитів за ${context.after}. Спробуйте
пізніше.`,
      instance: req.url,
    }),
  });
  await fastify.register(require("./plugins/error_handler"));
  await fastify.register(require("./plugins/auth"));
  const API_PREFIX = "/api/v1";
  await fastify.register(require("./routes/healthz"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/models"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/units"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/ingest"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/forecast"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/events"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/dashboard"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/sim_proxy"), { prefix: API_PREFIX });
  await fastify.register(require("./routes/sysinfo"), { prefix: API_PREFIX });
  return fastify;
}
async function start() {
  try {
    await build();
    await fastify.listen({ port: config.port, host: config.host });
    scheduler.start(fastify.log);
    sysagent.start(fastify.log);
  } catch (err) {
    fastify.log.error(err);
    process.exit(1);
  }
}

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 77   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

    }
  }
  if (require.main === module) {
    start();
  }
  module.exports = { build };
  backend\src\db\index.js
  "use strict";
  const Database = require("better-sqlite3");
  const path = require("path");
  const fs = require("fs");
  const config = require("../config");
  let db;
  function getDb() {
    if (!db) {
      const dbPath = path.resolve(config.db.path);
      const dir = path.dirname(dbPath);
      if (!fs.existsSync(dir)) {
        fs.mkdirSync(dir, { recursive: true });
      }
      db = new Database(dbPath);
      for (const [k, v] of Object.entries(config.db.pragma)) {
        db.pragma(`${k} = ${v}`);
      }
      require("../migrations").run(db);
    }
    return db;
  }
  function closeDb() {
    if (db) {
      db.close();
      db = null;
    }
  }
  module.exports = { getDb, closeDb };
  backend\src\db\migrations.js
  "use strict";
  const SCHEMA = `
CREATE TABLE IF NOT EXISTS equipment_models (
  id          INTEGER PRIMARY KEY AUTOINCREMENT,
  model_code   TEXT NOT NULL UNIQUE,
  display_name TEXT NOT NULL,
  category     TEXT NOT NULL,
  passport_json TEXT NOT NULL,
  passport_hash TEXT NOT NULL,
  weibull_eta  REAL NOT NULL,
  weibull_beta REAL NOT NULL,
  created_at   TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now'))
);
CREATE TABLE IF NOT EXISTS equipment_units (
  id          INTEGER PRIMARY KEY AUTOINCREMENT,
  model_id    INTEGER NOT NULL REFERENCES equipment_models(id),
  serial_no   TEXT NOT NULL UNIQUE,
  location    TEXT,
  commissioned_at TEXT NOT NULL,
  hours_run   REAL NOT NULL DEFAULT 0,
  is_active   INTEGER NOT NULL DEFAULT 1,
  notes       TEXT,
  created_at  TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now')),
  updated_at  TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now'))
);

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 78   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

CREATE INDEX IF NOT EXISTS idx_units_model ON equipment_units(model_id);
CREATE INDEX IF NOT EXISTS idx_units_active ON equipment_units(is_active);
CREATE TABLE IF NOT EXISTS telemetry (
    id                INTEGER PRIMARY KEY AUTOINCREMENT,
    unit_id           INTEGER NOT NULL REFERENCES equipment_units(id),
    channel_code      TEXT NOT NULL,
    ts                TEXT NOT NULL,
    value             REAL NOT NULL,
    quality           INTEGER NOT NULL DEFAULT 1
);
CREATE INDEX IF NOT EXISTS idx_telemetry_unit_ts ON telemetry(unit_id, channel_code, ts DESC);
CREATE INDEX IF NOT EXISTS idx_telemetry_ts ON telemetry(ts);
CREATE TABLE IF NOT EXISTS ground_truth_state (
    unit_id           INTEGER PRIMARY KEY REFERENCES equipment_units(id),
    true_health       REAL NOT NULL,
    active_mode       TEXT,
    mode_started_at   TEXT,
    rul_true_hours    REAL,
    updated_at        TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now'))
);
CREATE TABLE IF NOT EXISTS health_state (
    unit_id           INTEGER PRIMARY KEY REFERENCES equipment_units(id),
    status            TEXT NOT NULL DEFAULT 'ok',
    anomaly_score     REAL NOT NULL DEFAULT 0,
    predicted_mode     TEXT,
    predicted_mode_conf REAL,
    rul_hours         REAL,
    rul_lower_hours   REAL,
    rul_upper_hours   REAL,
    model_version     TEXT,
    last_updated      TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now'))
);
CREATE TABLE IF NOT EXISTS event_log (
    id                INTEGER PRIMARY KEY AUTOINCREMENT,
    unit_id           INTEGER REFERENCES equipment_units(id),
    ts                TEXT NOT NULL DEFAULT (strftime('%Y-%m-%dT%H:%M:%SZ', 'now')),
    severity          TEXT NOT NULL DEFAULT 'info',
    event_type        TEXT NOT NULL,
    payload_json      TEXT NOT NULL DEFAULT '{}',
    message           TEXT NOT NULL
);
CREATE INDEX IF NOT EXISTS idx_events_unit_ts ON event_log(unit_id, ts DESC);
CREATE INDEX IF NOT EXISTS idx_events_severity ON event_log(severity, ts DESC);
CREATE TABLE IF NOT EXISTS ml_model_versions (
    id                INTEGER PRIMARY KEY AUTOINCREMENT,
    version_tag       TEXT NOT NULL UNIQUE,
    trained_at        TEXT NOT NULL,
    metrics_json      TEXT NOT NULL DEFAULT '{}',
    is_active         INTEGER NOT NULL DEFAULT 0,
    notes             TEXT
);
`;

function run(db) {
    db.exec(SCHEMA);
}
module.exports = { run };
backend\src\db\seed.js
"use strict";
require("dotenv").config();

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 79   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

const crypto = require("crypto");
const { getDb } = require("../index");
function hashPassport(obj) {
    return crypto.createHash("sha256").update(JSON.stringify(obj)).digest("hex").slice(0,
16);
}
const PASSPORTS = [
    {
        model_code: "SRV-R740-2U",
        display_name: "Серверна стійка Dell R740 (2U)",
        category: "compute",
        weibull_eta_hours: 43800,
        weibull_beta: 2.4,
        channels: [
            { code: "cpu_temp_pkg", class: "T", unit: "°C", nominal: 55, operating_range:
[30, 75], critical_range: [85, 105], inertia_seconds: 30, noise_sigma: 0.6 },
            { code: "cpu_temp_core_avg", class: "T", unit: "°C", nominal: 58,
operating_range: [35, 80], critical_range: [90, 110], inertia_seconds: 25, noise_sigma:
0.5 },
            { code: "cpu_load_pct", class: "F", unit: "%", nominal: 35, operating_range: [0,
95], critical_range: [98, 100], inertia_seconds: 5, noise_sigma: 2 },
            { code: "ram_usage_pct", class: "F", unit: "%", nominal: 50, operating_range: [0,
90], critical_range: [95, 100], inertia_seconds: 3, noise_sigma: 1 },
            { code: "ram_ecc_errors_rate", class: "Q", unit: "err/min", nominal: 0,
operating_range: [0, 2], critical_range: [10, 9999], inertia_seconds: 0, noise_sigma:
0.1 },
            { code: "ssd_wear_pct", class: "R", unit: "%", nominal: 0, operating_range: [0,
80], critical_range: [95, 100], inertia_seconds: 0, noise_sigma: 0 },
            { code: "ssd_temp", class: "T", unit: "°C", nominal: 40, operating_range: [25,
60], critical_range: [70, 85], inertia_seconds: 60, noise_sigma: 0.3 },
            { code: "fan_rpm_front", class: "F", unit: "rpm", nominal: 4500, operating_range:
[2000, 7000], critical_range: [0, 1500], inertia_seconds: 5, noise_sigma: 50 },
            { code: "fan_rpm_rear", class: "F", unit: "rpm", nominal: 4000, operating_range:
[2000, 7000], critical_range: [0, 1500], inertia_seconds: 5, noise_sigma: 50 },
            { code: "psu_voltage_12v", class: "E", unit: "V", nominal: 12.0, operating_range:
[11.4, 12.6], critical_range: [11.0, 13.0], inertia_seconds: 1, noise_sigma: 0.05 },
            { code: "psu_current", class: "E", unit: "A", nominal: 25, operating_range: [10,
50], critical_range: [0, 60], inertia_seconds: 2, noise_sigma: 0.5 },
            { code: "network_rx_errors_rate", class: "Q", unit: "err/s", nominal: 0,
operating_range: [0, 5], critical_range: [50, 9999], inertia_seconds: 0, noise_sigma:
0.2 },
            { code: "disk_io_latency_p99", class: "Q", unit: "ms", nominal: 5,
operating_range: [0, 20], critical_range: [100, 9999], inertia_seconds: 0, noise_sigma:
0.5 },
            { code: "uptime_hours", class: "R", unit: "h", nominal: 0, operating_range: [0,
9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0 },
            { code: "gpu_temp", class: "T", unit: "°C", nominal: 60, operating_range: [40,
85], critical_range: [95, 105], inertia_seconds: 20, noise_sigma: 0.8 },
        ],
        failure_modes: [
            { code: "thermal_runaway", name: "Тепловий розгін", affected_channels:
["cpu_temp_pkg", "fan_rpm_front", "gpu_temp"], signature: "rising_T_compensating_F",
typical_horizon_hours: 72 },
            { code: "fan_bearing_wear", name: "Знос підшипника вентилятора",
affected_channels: ["fan_rpm_front", "fan_rpm_rear", "cpu_temp_pkg"], signature:
"rpm_std_increase", typical_horizon_hours: 240 },
            { code: "ssd_endurance_exhaustion", name: "Вичерпання ресурсу SSD",
affected_channels: ["ssd_wear_pct", "disk_io_latency_p99", "ssd_temp"], signature:
"monotone_wear_increase", typical_horizon_hours: 720 },
            { code: "psu_degradation", name: "Деградація блоку живлення", affected_channels:
["psu_voltage_12v", "psu_current"], signature: "voltage_instability",
typical_horizon_hours: 168 },
        ]
    }
]

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 80   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

    ],
  },
  {
    model_code: "PMP-CF250",
    display_name: "Відцентровий насос CF-250",
    category: "rotary",
    weibull_eta_hours: 26280,
    weibull_beta: 2.1,
    channels: [
      { code: "vibration_x_rms", class: "V", unit: "mm/s", nominal: 2.5,
        operating_range: [0, 7], critical_range: [11.2, 9999], inertia_seconds: 2, noise_sigma:
        0.2 },
      { code: "vibration_y_rms", class: "V", unit: "mm/s", nominal: 2.5,
        operating_range: [0, 7], critical_range: [11.2, 9999], inertia_seconds: 2, noise_sigma:
        0.2 },
      { code: "vibration_z_rms", class: "V", unit: "mm/s", nominal: 1.5,
        operating_range: [0, 4.5], critical_range: [7, 9999], inertia_seconds: 2, noise_sigma:
        0.1 },
      { code: "bearing_temp_de", class: "T", unit: "°C", nominal: 50, operating_range:
        [25, 85], critical_range: [95, 9999], inertia_seconds: 60, noise_sigma: 0.4 },
      { code: "bearing_temp_nd", class: "T", unit: "°C", nominal: 48, operating_range:
        [25, 85], critical_range: [95, 9999], inertia_seconds: 60, noise_sigma: 0.4 },
      { code: "winding_current", class: "E", unit: "A", nominal: 18, operating_range:
        [10, 25], critical_range: [32, 9999], inertia_seconds: 3, noise_sigma: 0.3 },
      { code: "winding_temp", class: "T", unit: "°C", nominal: 70, operating_range:
        [40, 110], critical_range: [130, 9999], inertia_seconds: 90, noise_sigma: 0.6 },
      { code: "pressure_discharge", class: "F", unit: "bar", nominal: 6.0,
        operating_range: [4, 8], critical_range: [2, 10], inertia_seconds: 5, noise_sigma: 0.05
        },
      { code: "flow_rate", class: "F", unit: "m³/h", nominal: 250, operating_range:
        [150, 320], critical_range: [100, 380], inertia_seconds: 10, noise_sigma: 2 },
      { code: "rpm", class: "F", unit: "rpm", nominal: 2960, operating_range: [2900,
        3020], critical_range: [2860, 3060], inertia_seconds: 3, noise_sigma: 5 },
      { code: "runtime_hours", class: "R", unit: "h", nominal: 0, operating_range: [0,
        9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0 },
    ],
    failure_modes: [
      { code: "bearing_wear", name: "Знос підшипника", affected_channels:
        ["vibration_x_rms", "vibration_y_rms", "bearing_temp_de"], typical_horizon_hours: 360
        },
      { code: "rotor_imbalance", name: "Дисбаланс ротора", affected_channels:
        ["vibration_x_rms", "vibration_y_rms", "vibration_z_rms"], typical_horizon_hours: 168
        },
      { code: "cavitation", name: "Кавітація", affected_channels:
        ["pressure_discharge", "vibration_x_rms", "winding_current"], typical_horizon_hours: 48
        },
      { code: "winding_short", name: "Замикання обмотки", affected_channels:
        ["winding_current", "winding_temp"], typical_horizon_hours: 24 },
    ],
  },
  {
    model_code: "CRY-LN2-15K",
    display_name: "Кріогенна установка LN2-15K",
    category: "cryo",
    weibull_eta_hours: 35040,
    weibull_beta: 1.8,
    channels: [
      { code: "dewar_pressure", class: "F", unit: "bar", nominal: 1.1, operating_range:
        [0.9, 1.5], critical_range: [2.0, 9999], inertia_seconds: 20, noise_sigma: 0.01 },
      { code: "cold_head_temp", class: "T", unit: "K", nominal: 4.2, operating_range:
        [4.0, 10], critical_range: [20, 9999], inertia_seconds: 120, noise_sigma: 0.05 },
      { code: "compressor_current", class: "E", unit: "A", nominal: 14,

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 81   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

operating_range: [10, 20], critical_range: [25, 9999], inertia_seconds: 5, noise_sigma:
0.2 },
    { code: "compressor_oil_temp", class: "T", unit: "°C", nominal: 55,
operating_range: [30, 80], critical_range: [95, 9999], inertia_seconds: 180,
noise_sigma: 0.3 },
    { code: "ln2_level_pct", class: "F", unit: "%", nominal: 70, operating_range:
[30, 95], critical_range: [0, 10], inertia_seconds: 60, noise_sigma: 0.5 },
    { code: "boil_off_rate", class: "F", unit: "l/h", nominal: 0.5, operating_range:
[0.2, 1.0], critical_range: [3.0, 9999], inertia_seconds: 30, noise_sigma: 0.02 },
    { code: "vacuum_pressure", class: "E", unit: "Pa", nominal: 0.0001,
operating_range: [0.00001, 0.001], critical_range: [0.01, 9999], inertia_seconds: 300,
noise_sigma: 0.000005 },
    { code: "condense_cycles_per_hour", class: "F", unit: "cyc/h", nominal: 0.5,
operating_range: [0, 2], critical_range: [5, 9999], inertia_seconds: 0, noise_sigma: 0
},
    { code: "runtime_hours", class: "R", unit: "h", nominal: 0, operating_range: [0,
9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0 },
    { code: "defrost_count", class: "R", unit: "count", nominal: 0, operating_range:
[0, 9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0 },
],
failure_modes: [
    { code: "vacuum_breach", name: "Витік ізоляції вакууму", affected_channels:
["vacuum_pressure", "boil_off_rate", "compressor_current"], typical_horizon_hours: 144
},
    { code: "compressor_wear", name: "Деградація компресора", affected_channels:
["compressor_current", "compressor_oil_temp", "condense_cycles_per_hour"],
typical_horizon_hours: 480 },
    { code: "filter_clog", name: "Забивання фільтра", affected_channels:
["dewar_pressure", "ln2_level_pct", "boil_off_rate"], typical_horizon_hours: 96 },
],
{
    model_code: "QA-7000",
    display_name: "Квантовий анігілятор QA-7000",
    category: "experimental",
    weibull_eta_hours: 17520,
    weibull_beta: 1.5,
    channels: [
        { code: "quantum_flux_pressure", class: "T", unit: "qfp", nominal: 42.0,
operating_range: [38, 47], critical_range: [52, 9999], inertia_seconds: 15,
noise_sigma: 0.3 },
        { code: "containment_field_strength", class: "E", unit: "kG", nominal: 12.5,
operating_range: [11.0, 14.0], critical_range: [0, 9.0], inertia_seconds: 10,
noise_sigma: 0.1 },
        { code: "tachyon_emission_rate", class: "Q", unit: "count/s", nominal: 0,
operating_range: [0, 3], critical_range: [15, 9999], inertia_seconds: 0, noise_sigma:
0.1 },
        { code: "vacuum_chamber_integrity", class: "F", unit: "%", nominal: 99.5,
operating_range: [98, 100], critical_range: [0, 95], inertia_seconds: 120, noise_sigma:
0.05 },
        { code: "entanglement_coherence", class: "Q", unit: "%", nominal: 87,
operating_range: [80, 95], critical_range: [0, 70], inertia_seconds: 30, noise_sigma:
0.5 },
        { code: "annihilation_cycle_count", class: "R", unit: "count", nominal: 0,
operating_range: [0, 9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0
},
        { code: "dimensional_resonance_freq", class: "V", unit: "Hz", nominal: 2400000,
operating_range: [2280000, 2520000], critical_range: [2040000, 2760000],
inertia_seconds: 5, noise_sigma: 500 },
        { code: "core_temp_kelvin", class: "T", unit: "K", nominal: 0.05,
operating_range: [0.02, 0.1], critical_range: [0.5, 9999], inertia_seconds: 300,
noise_sigma: 0.001 },

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 82   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        { code: "total_runtime_hours", class: "R", unit: "h", nominal: 0,
operating_range: [0, 9999], critical_range: [0, 0], inertia_seconds: 0, noise_sigma: 0
},
    ],
    failure_modes: [
        { code: "field_decoherence", name: "Декогеренція поля", affected_channels:
["containment_field_strength", "entanglement_coherence", "tachyon_emission_rate"],
typical_horizon_hours: 120 },
        { code: "tachyon_overflow", name: "Перевантаження тахіонів", affected_channels:
["tachyon_emission_rate", "quantum_flux_pressure"], typical_horizon_hours: 48 },
        { code: "containment_drift", name: "Дрейф утримуючого поля", affected_channels:
["containment_field_strength", "vacuum_chamber_integrity",
"dimensional_resonance_freq"], typical_horizon_hours: 240 },
    ],
};
const UNITS = [
    { model_code: "SRV-R740-2U", serial_no: "SRV-001", location: "ДЦ-1 стійка A1",
hours_run: 8000, commissioned_at: "2023-02-15T00:00:00Z" },
    { model_code: "SRV-R740-2U", serial_no: "SRV-002", location: "ДЦ-1 стійка A2",
hours_run: 15000, commissioned_at: "2022-07-20T00:00:00Z" },
    { model_code: "SRV-R740-2U", serial_no: "SRV-003", location: "ДЦ-2 стійка B1",
hours_run: 22000, commissioned_at: "2021-12-01T00:00:00Z" },
    { model_code: "SRV-R740-2U", serial_no: "SRV-004", location: "ДЦ-2 стійка B2",
hours_run: 3000, commissioned_at: "2023-10-05T00:00:00Z" },
    { model_code: "PMP-CF250", serial_no: "PMP-001", location: "НС-1 агрегат 1",
hours_run: 12000, commissioned_at: "2022-10-10T00:00:00Z" },
    { model_code: "PMP-CF250", serial_no: "PMP-002", location: "НС-1 агрегат 2",
hours_run: 18000, commissioned_at: "2022-02-14T00:00:00Z" },
    { model_code: "PMP-CF250", serial_no: "PMP-003", location: "НС-2 агрегат 1",
hours_run: 5000, commissioned_at: "2023-07-01T00:00:00Z" },
    { model_code: "PMP-CF250", serial_no: "PMP-004", location: "НС-2 агрегат 2",
hours_run: 9500, commissioned_at: "2023-01-18T00:00:00Z" },
    { model_code: "CRY-LN2-15K", serial_no: "CRY-001", location: "Лаб 101", hours_run:
7200, commissioned_at: "2023-04-20T00:00:00Z" },
    { model_code: "CRY-LN2-15K", serial_no: "CRY-002", location: "Лаб 203", hours_run:
14400, commissioned_at: "2022-06-01T00:00:00Z" },
    { model_code: "CRY-LN2-15K", serial_no: "CRY-003", location: "Лаб 305", hours_run:
2800, commenced_at: "2023-12-01T00:00:00Z", commissioned_at: "2023-12-01T00:00:00Z" },
    { model_code: "CRY-LN2-15K", serial_no: "CRY-004", location: "Лаб 401", hours_run:
21000, commissioned_at: "2021-10-15T00:00:00Z" },
    { model_code: "QA-7000", serial_no: "QA-001", location: "Сектор Альфа", hours_run:
4000, commissioned_at: "2023-09-01T00:00:00Z" },
    { model_code: "QA-7000", serial_no: "QA-002", location: "Сектор Бета", hours_run:
8800, commissioned_at: "2023-01-10T00:00:00Z" },
    { model_code: "QA-7000", serial_no: "QA-003", location: "Сектор Гамма", hours_run:
1200, commissioned_at: "2024-04-01T00:00:00Z" },
    { model_code: "QA-7000", serial_no: "QA-004", location: "Сектор Дельта", hours_run:
11000, commissioned_at: "2022-09-20T00:00:00Z" },
];
function seed() {
    const db = getDb();
    const existing = db.prepare("SELECT COUNT(*) as cnt FROM equipment_models").get();
    if (existing.cnt > 0) {
        console.log("Database already seeded, skipping.");
        return;
    }
    const insertModel = db.prepare(
        "INSERT INTO equipment_models (model_code, display_name, category, passport_json,
passport_hash, weibull_eta, weibull_beta) VALUES (?, ?, ?, ?, ?, ?, ?)"
    );
    const insertUnit = db.prepare(

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 83   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

"INSERT INTO equipment_units (model_id, serial_no, location, commissioned_at,
hours_run) VALUES (?, ?, ?, ?, ?)"
);
const insertHealth = db.prepare(
"INSERT INTO health_state (unit_id, status, anomaly_score) VALUES (?, 'ok', 0)"
);
const doSeed = db.transaction(() => {
const modelIdByCode = {};
for (const passport of PASSPORTS) {
const passport_json = JSON.stringify(passport);
const passport_hash = hashPassport(passport);
const { lastInsertRowid } = insertModel.run(
passport.model_code,
passport.display_name,
passport.category,
passport_json,
passport_hash,
passport.weibull_eta_hours,
passport.weibull_beta
);
modelIdByCode[passport.model_code] = lastInsertRowid;
}
for (const u of UNITS) {
const model_id = modelIdByCode[u.model_code];
if (!model_id) throw new Error(`Unknown model_code: ${u.model_code}`);
const { lastInsertRowid } = insertUnit.run(
model_id,
u.serial_no,
u.location,
u.commissioned_at,
u.hours_run
);
insertHealth.run(lastInsertRowid);
}
});
doSeed();
console.log(`Seeded ${PASSPORTS.length} models and ${UNITS.length} units.`);
}
seed();
backend\src\domain\health.js
"use strict";
const { getDb } = require("../db");
const THRESHOLDS = {
imminent: { score: 0.7, rul: 72 },
risk: { score: 0.3, rul: 168 },
};
const HYSTERESIS = {
score: 0.1,
rulFactor: 0.9,
};
function classify(anomaly_score, rul_hours, current_status) {
const rul = rul_hours != null ? rul_hours : Infinity;
if (anomaly_score >= THRESHOLDS.imminent.score || rul <= THRESHOLDS.imminent.rul) {
return "imminent";
}
if (anomaly_score >= THRESHOLDS.risk.score || rul <= THRESHOLDS.risk.rul) {
return "risk";
}
if (current_status === "imminent") {
const stricter_score = THRESHOLDS.imminent.score - HYSTERESIS.score;
const stricter_rul = THRESHOLDS.imminent.rul / HYSTERESIS.rulFactor;
if (anomaly_score >= stricter_score || rul <= stricter_rul) {

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 84   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

        return "imminent";
    }
}
if (current_status === "risk") {
    const stricter_score = THRESHOLDS.risk.score - HYSTERESIS.score;
    const stricter_rul = THRESHOLDS.risk.rul / HYSTERESIS.rulFactor;
    if (anomaly_score >= stricter_score || rul <= stricter_rul) {
        return "risk";
    }
}
return "ok";
}
const findByUnitId = (unit_id) =>
    getDb().prepare("SELECT * FROM health_state WHERE unit_id=?").get(unit_id) || null;
const upsert = (unit_id, prediction) => {
    const db = getDb();
    const current = db.prepare("SELECT status FROM health_state WHERE
unit_id=?").get(unit_id);
    const current_status = current ? current.status : "ok";
    const new_status = classify(
        prediction.anomaly_score,
        prediction.rul_hours != null ? prediction.rul_hours : Infinity,
        current_status
    );
    db.prepare(
        `INSERT INTO health_state (unit_id, status, anomaly_score, predicted_mode,
predicted_mode_conf,

        rul_hours, rul_lower_hours, rul_upper_hours, model_version, last_updated)
VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, strftime('%Y-%m-%dT%H:%M:%SZ', 'now'))
ON CONFLICT(unit_id) DO UPDATE SET
    status=excluded.status,
    anomaly_score=excluded.anomaly_score,
    predicted_mode=excluded.predicted_mode,
    predicted_mode_conf=excluded.predicted_mode_conf,
    rul_hours=excluded.rul_hours,
    rul_lower_hours=excluded.rul_lower_hours,
    rul_upper_hours=excluded.rul_upper_hours,
    model_version=excluded.model_version,
    last_updated=excluded.last_updated`
    ).run(
        unit_id,
        new_status,
        prediction.anomaly_score,
        prediction.predicted_mode ?? null,
        prediction.predicted_mode_conf ?? null,
        prediction.rul_hours ?? null,
        prediction.rul_lower_hours ?? null,
        prediction.rul_upper_hours ?? null,
        prediction.model_version ?? null
    );
    return { new_status, previous_status: current_status, changed: new_status !==
current_status };
};
const escalateStaleRisk = () => {
    const db = getDb();
    const threshold_ts = new Date(Date.now() - 48 * 60 * 60 *
1000).toISOString().replace("T", "T").slice(0, 19) + "Z";
    const stale = db
        .prepare(
            "SELECT unit_id FROM health_state WHERE status='risk' AND last_updated <= ?"
        )

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 85   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

        .all(threshold_ts);
        const updateStmt = db.prepare(
            "UPDATE health_state SET status='imminent', last_updated=strftime('%Y-%m-%dT%H:%M:%SZ', 'now') WHERE unit_id=?"
        );
        const insertEvent = db.prepare(
            `INSERT INTO event_log (unit_id, severity, event_type, payload_json, message)

            VALUES (?, 'warning', 'status_escalated', ?, ?)`
        );
        const doEscalate = db.transaction((units) => {
            for (const { unit_id } of units) {
                updateStmt.run(unit_id);
                insertEvent.run(
                    unit_id,
                    JSON.stringify({ previous_status: "risk", new_status: "imminent", reason:
                        "risk_timeout_48h" }),
                    "Статус автоматично підвищено до imminent: стан risk тривав понад 48 годин"
                );
            }
        });
        doEscalate(stale);
        return stale.length;
    };
    module.exports = { classify, findByUnitId, upsert, escalateStaleRisk };
}
backend\src\domain\models.js
"use strict";
const crypto = require("crypto");
const Ajv = require("ajv");
const { getDb } = require("../db");
const { PASSPORT_SCHEMA } = require("../lib/passport_schema");
const ajv = new Ajv({ allErrors: true });
const validatePassport = ajv.compile(PASSPORT_SCHEMA);
function hashPassport(obj) {
    return crypto.createHash("sha256").update(JSON.stringify(obj)).digest("hex").slice(0, 16);
}
function notFound(id) {
    const err = new Error(`Модель #${id} не знайдена`);
    err.code = "NOT_FOUND";
    return err;
}
function validateOrThrow(passport) {
    const valid = validatePassport(passport);
    if (!valid) {
        const err = new Error("Паспорт не відповідає схемі: " +
            ajv.errorsText(validatePassport.errors));
        err.statusCode = 422;
        err.validation = validatePassport.errors;
        throw err;
    }
}
const list = () =>
    getDb()
        .prepare(
            "SELECT id, model_code, display_name, category, weibull_eta, weibull_beta,
            created_at FROM equipment_models ORDER BY display_name"
        )
        .all();
const findById = (id) => {
    const row = getDb().prepare("SELECT * FROM equipment_models WHERE id = ?").get(id);

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 86   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

    return row || null;
};
const create = ({ passport }) => {
    validateOrThrow(passport);
    const passport_json = JSON.stringify(passport);
    const passport_hash = hashPassport(passport);
    const { lastInsertRowid } = getDb()
        .prepare(
            "INSERT INTO equipment_models (model_code, display_name, category, passport_json,
passport_hash, weibull_eta, weibull_beta) VALUES (?, ?, ?, ?, ?, ?, ?)"
        )
        .run(
            passport.model_code,
            passport.display_name,
            passport.category,
            passport_json,
            passport_hash,
            passport.weibull_eta_hours,
            passport.weibull_beta
        );
    return findById(lastInsertRowid);
};
const update = (id, { passport }) => {
    const existing = findById(id);
    if (!existing) throw notFound(id);
    validateOrThrow(passport);
    const passport_json = JSON.stringify(passport);
    const passport_hash = hashPassport(passport);
    getDb()
        .prepare(
            "UPDATE equipment_models SET model_code=?, display_name=?, category=?,
passport_json=?, passport_hash=?, weibull_eta=?, weibull_beta=? WHERE id=?"
        )
        .run(
            passport.model_code,
            passport.display_name,
            passport.category,
            passport_json,
            passport_hash,
            passport.weibull_eta_hours,
            passport.weibull_beta,
            id
        );
    return findById(id);
};
const remove = (id) => {
    const existing = findById(id);
    if (!existing) throw notFound(id);
    const inUse = getDb().prepare("SELECT 1 FROM equipment_units WHERE model_id=? AND
is_active=1").get(id);
    if (inUse) {
        const err = new Error("Модель має активні одиниці обладнання та не може бути
видалена");
        err.code = "CONFLICT";
        throw err;
    }
    getDb().prepare("DELETE FROM equipment_models WHERE id=?").run(id);
};
module.exports = { list, findById, create, update, remove };
backend\src\domain\sysagent.js
"use strict";
const os = require("os");

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 87   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

const fs = require("fs");
const { getDb } = require("../db");
const { ingest } = require("../telemetry");
const MODEL_CODE = "LOCAL-SRV";
const SERIAL_NO = os.hostname();
const INTERVAL_MS = 30_000;
const PASSPORT = {
  model_code: MODEL_CODE,
  display_name: "Local Server Hardware",
  category: "server",
  weibull_eta_hours: 87600,
  weibull_beta: 1.5,
  channels: [
    {
      code: "ram_used_pct",
      class: "Q",
      unit: "%",
      nominal: 50,
      operating_range: [0, 85],
      critical_range: [85, 100],
      noise_sigma: 0.5,
    },
    {
      code: "cpu_load_pct",
      class: "Q",
      unit: "%",
      nominal: 30,
      operating_range: [0, 90],
      critical_range: [90, 100],
      noise_sigma: 1.0,
    },
    {
      code: "disk_used_pct",
      class: "Q",
      unit: "%",
      nominal: 30,
      operating_range: [0, 90],
      critical_range: [90, 100],
      noise_sigma: 0.1,
    },
  ],
  failure_modes: [
    {
      code: "memory_pressure",
      name: "Перевантаження оперативної пам'яті",
      affected_channels: ["ram_used_pct"],
      typical_horizon_hours: 72,
    },
    {
      code: "cpu_overload",
      name: "Перевантаження процесора",
      affected_channels: ["cpu_load_pct"],
      typical_horizon_hours: 24,
    },
    {
      code: "disk_full",
      name: "Заповнення диску",
      affected_channels: ["disk_used_pct"],
      typical_horizon_hours: 168,
    },
  ],
},
];
};

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 88   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

async function getCpuLoadPct() {
  const cpus1 = os.cpus();
  await new Promise((r) => setTimeout(r, 200));
  const cpus2 = os.cpus();
  let totalDiff = 0;
  let idleDiff = 0;
  for (let i = 0; i < cpus1.length; i++) {
    const sum1 = Object.values(cpus1[i].times).reduce((a, b) => a + b, 0);
    const sum2 = Object.values(cpus2[i].times).reduce((a, b) => a + b, 0);
    totalDiff += sum2 - sum1;
    idleDiff += cpus2[i].times.idle - cpus1[i].times.idle;
  }
  return totalDiff > 0 ? Math.round((1 - idleDiff / totalDiff) * 100) : 0;
}
async function getDiskUsedPct() {
  try {
    const drivePath = process.cwd().slice(0, 2) + "/";
    const stats = await fs.promises.statfs(drivePath);
    const total = stats.blocks * stats.bsize;
    const free = stats.bfree * stats.bsize;
    return total > 0 ? Math.round(((total - free) / total) * 100) : null;
  } catch {
    return null;
  }
}
function getRamUsedPct() {
  const total = os.totalmem();
  const free = os.freemem();
  return Math.round(((total - free) / total) * 100);
}
async function collectMetrics() {
  const [cpu_load_pct, disk_used_pct] = await Promise.all([getCpuLoadPct(),
getDiskUsedPct()]);
  return {
    ram_used_pct: getRamUsedPct(),
    cpu_load_pct,
    disk_used_pct,
    ram_total_mb: Math.round(os.totalmem() / 1048576),
    ram_free_mb: Math.round(os.freemem() / 1048576),
    cpu_count: os.cpus().length,
    platform: os.platform(),
    hostname: os.hostname(),
  };
}
function ensureModel(db) {
  const existing = db.prepare("SELECT id FROM equipment_models WHERE
model_code=?").get(MODEL_CODE);
  if (existing) return existing.id;
  const crypto = require("crypto");
  const passport_json = JSON.stringify(PASSPORT);
  const passport_hash = crypto
    .createHash("sha256")
    .update(passport_json)
    .digest("hex")
    .slice(0, 16);
  const { lastInsertRowid } = db
    .prepare(
      "INSERT INTO equipment_models (model_code, display_name, category, passport_json,
passport_hash, weibull_eta, weibull_beta) VALUES (?, ?, ?, ?, ?, ?, ?)"
    )
    .run(
      PASSPORT.model_code,

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 89   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

        PASSPORT.display_name,
        PASSPORT.category,
        passport_json,
        passport_hash,
        PASSPORT.weibull_eta_hours,
        PASSPORT.weibull_beta
    );
    return lastInsertRowid;
}
function ensureUnit(db, model_id) {
    const existing = db
        .prepare("SELECT id FROM equipment_units WHERE serial_no=? AND model_id=?")
        .get(SERIAL_NO, model_id);
    if (existing) return existing.id;
    const today = new Date().toISOString().slice(0, 10);
    const tx = db.transaction(() => {
        const { lastInsertRowid } = db
            .prepare(
                "INSERT INTO equipment_units (model_id, serial_no, location, commissioned_at,
notes) VALUES (?, ?, ?, ?, ?)"
            )
            .run(model_id, SERIAL_NO, "Local Server", today, "Auto-registered by sysagent");
        db.prepare("INSERT INTO health_state (unit_id, status, anomaly_score) VALUES (?,
'ok', 0)").run(lastInsertRowid);
        return lastInsertRowid;
    });
    return tx();
}
let unitId = null;
function ensureRegistered() {
    const db = getDb();
    const model_id = ensureModel(db);
    unitId = ensureUnit(db, model_id);
    return unitId;
}
async function tick(log) {
    try {
        if (!unitId) ensureRegistered();
        const m = await collectMetrics();
        const ts = new Date().toISOString();
        const points = [];
        points.push({ unit_id: unitId, channel_code: "ram_used_pct", ts, value:
m.ram_used_pct });
        if (m.cpu_load_pct !== null) {
            points.push({ unit_id: unitId, channel_code: "cpu_load_pct", ts, value:
m.cpu_load_pct });
        }
        if (m.disk_used_pct !== null) {
            points.push({ unit_id: unitId, channel_code: "disk_used_pct", ts, value:
m.disk_used_pct });
        }
        ingest(points);
        if (log) log.debug({ ram: m.ram_used_pct, cpu: m.cpu_load_pct, disk:
m.disk_used_pct }, "sysagent tick");
    } catch (err) {
        if (log) log.warn({ err: err.message }, "sysagent tick failed");
    }
}
function start(log) {
    try {
        ensureRegistered();
        if (log) log.info({ unit_id: unitId, serial_no: SERIAL_NO }, "sysagent: Local

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 90   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

Server unit registered");
    } catch (err) {
        if (log) log.warn({ err: err.message }, "sysagent: registration failed, will retry
on first tick");
    }
    tick(log);
    setInterval(() => tick(log), INTERVAL_MS);
}
module.exports = { start, collectMetrics, getUnitId: () => unitId };
backend\src\domain\telemetry.js
"use strict";
const { getDb } = require("../db");
const ingest = (points) => {
    const db = getDb();
    const stmt = db.prepare(
        "INSERT INTO telemetry (unit_id, channel_code, ts, value, quality) VALUES (?, ?, ?,
?, ?)"
    );
    const updateHours = db.prepare(
        "UPDATE equipment_units SET hours_run=?, updated_at=strftime('%Y-%m-
%dT%H:%M:%SZ','now') WHERE id=? AND hours_run < ?"
    );
    const doIngest = db.transaction((rows) => {
        const maxTsByUnit = {};
        for (const r of rows) {
            stmt.run(r.unit_id, r.channel_code, r.ts, r.value, r.quality ?? 1);
            if (!maxTsByUnit[r.unit_id] || r.ts > maxTsByUnit[r.unit_id].ts) {
                maxTsByUnit[r.unit_id] = r;
            }
        }
        for (const [unit_id, latest] of Object.entries(maxTsByUnit)) {
            const latestRow = db
                .prepare(
                    "SELECT MAX(CAST(strftime('%s', ts) AS REAL)) as max_epoch,
MIN(CAST(strftime('%s', ts) AS REAL)) as min_epoch FROM telemetry WHERE unit_id=?"
                )
                .get(unit_id);
            if (latestRow && latestRow.max_epoch && latestRow.min_epoch) {
                const spanHours = (latestRow.max_epoch - latestRow.min_epoch) / 3600;
                const unit = db.prepare("SELECT hours_run FROM equipment_units WHERE
id=?").get(unit_id);
                if (unit) {
                    const newHours = Math.max(unit.hours_run, spanHours);
                    updateHours.run(newHours, unit_id, newHours + 1);
                }
            }
        }
    });
    doIngest(points);
    return points.length;
};
const query = ({ unit_id, channel, from, to, limit = 1000 }) => {
    let sql = "SELECT id, unit_id, channel_code, ts, value, quality FROM telemetry WHERE
unit_id=?";
    const params = [unit_id];
    if (channel) {
        sql += " AND channel_code=?";
        params.push(channel);
    }
    if (from) {
        sql += " AND ts>=?";
        params.push(from);
    }

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 91   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

    }
    if (to) {
        sql += " AND ts<=?";
        params.push(to);
    }
    sql += " ORDER BY ts DESC LIMIT ?";
    params.push(Math.min(limit, 10000));
    return getDb().prepare(sql).all(...params);
};

const getFeatureWindow = (unit_id, n = 360) =>
    getDb()
        .prepare(
            "SELECT channel_code, ts, value FROM telemetry WHERE unit_id=? ORDER BY ts DESC
LIMIT ?"
        )
        .all(unit_id, n);

function buildFeatures(rows, passport) {
    if (!rows || rows.length === 0) return {};
    const byChannel = {};
    for (const r of rows) {
        if (!byChannel[r.channel_code]) byChannel[r.channel_code] = [];
        byChannel[r.channel_code].push(r.value);
    }
    const chMeta = {};
    if (passport && Array.isArray(passport.channels)) {
        for (const ch of passport.channels) {
            chMeta[ch.code] = ch;
        }
    }
    const features = {};
    for (const [ch, values] of Object.entries(byChannel)) {
        const n = values.length;
        const mean = values.reduce((a, b) => a + b, 0) / n;
        const variance = values.reduce((acc, v) => acc + (v - mean) ** 2, 0) / n;
        const std = Math.sqrt(variance);
        const sorted = [...values].sort((a, b) => a - b);
        const min = sorted[0];
        const max = sorted[n - 1];
        const mid = Math.floor(n / 2);
        const median = n % 2 === 0 ? (sorted[mid - 1] + sorted[mid]) / 2 : sorted[mid];
        let slope = 0;
        if (n >= 2) {
            const xMean = (n - 1) / 2;
            let num = 0;
            let den = 0;
            for (let i = 0; i < n; i++) {
                num += (i - xMean) * (values[i] - mean);
                den += (i - xMean) ** 2;
            }
            slope = den !== 0 ? num / den : 0;
        }
        features[`_${ch}__mean`] = mean;
        features[`_${ch}__std`] = std;
        features[`_${ch}__min`] = min;
        features[`_${ch}__max`] = max;
        features[`_${ch}__median`] = median;
        features[`_${ch}__slope`] = slope;
        const meta = chMeta[ch];
        if (meta) {
            const nominal = meta.nominal ?? 0;
            const opRange = meta.operating_range ?? [0, 1];
            const critRange = meta.critical_range ?? [null, null];

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 92   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

const opLow = opRange[0];
const opHigh = opRange[1];
const opWidth = Math.max(1, Math.abs(opHigh - opLow));
const critLow = critRange[0];
const critHigh = critRange[1];
features[`${ch}__dev_from_nominal`] = (mean - nominal) / opWidth;
features[`${ch}__time_above_op`] = values.filter((v) => v > opHigh).length / n;
features[`${ch}__time_below_op`] = values.filter((v) => v < opLow).length / n;
if (critHigh !== null && critLow !== null) {
  if (critHigh > critLow) {
    features[`${ch}__in_critical`] = values.filter((v) => v >= critLow && v <=
critHigh).length / n;
  } else {
    features[`${ch}__in_critical`] = values.filter((v) => v <= critHigh).length /
n;
  }
}
}
}
return features;
}
module.exports = { ingest, query, getFeatureWindow, buildFeatures };
backend/src/domain/units.js
"use strict";
const { getDb } = require("../db");
function notFound(id) {
  const err = new Error(`Одиниця обладнання #${id} не знайдена`);
  err.code = "NOT_FOUND";
  return err;
}
const list = ({ model_id, status, q } = {}) => {
  let sql = `
    SELECT u.id, u.model_id, u.serial_no, u.location, u.commissioned_at,
      u.hours_run, u.is_active, u.notes, u.created_at, u.updated_at,
      m.model_code, m.display_name as model_display_name, m.category,
      h.status, h.anomaly_score, h.rul_hours, h.predicted_mode, h.last_updated as
health_updated
    FROM equipment_units u
    JOIN equipment_models m ON m.id = u.model_id
    LEFT JOIN health_state h ON h.unit_id = u.id
    WHERE u.is_active = 1
  `;

  const params = [];
  if (model_id) {
    sql += " AND u.model_id = ?";
    params.push(model_id);
  }
  if (status) {
    sql += " AND h.status = ?";
    params.push(status);
  }
  if (q) {
    sql += " AND (u.serial_no LIKE ? OR u.location LIKE ?)";
    params.push(`%${q}%`, `%${q}%`);
  }
  sql += " ORDER BY u.serial_no";
  return getDb().prepare(sql).all(...params);
};
const findById = (id) => {
  const row = getDb()

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 93   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

.prepare(
  `SELECT u.id, u.model_id, u.serial_no, u.location, u.commissioned_at,

          u.hours_run, u.is_active, u.notes, u.created_at, u.updated_at,
          m.model_code, m.display_name as model_display_name, m.category,
          m.passport_json, m.weibull_eta, m.weibull_beta,
          h.status, h.anomaly_score, h.rul_hours, h.rul_lower_hours,
h.rul_upper_hours,
          h.predicted_mode, h.predicted_mode_conf, h.model_version, h.last_updated
as health_updated
  FROM equipment_units u
  JOIN equipment_models m ON m.id = u.model_id
  LEFT JOIN health_state h ON h.unit_id = u.id
  WHERE u.id = ?`
)
.get(id);
return row || null;
};
const create = ({ model_id, serial_no, location, commissioned_at, notes }) => {
  const db = getDb();
  const modelExists = db.prepare("SELECT id FROM equipment_models WHERE
id=?").get(model_id);
  if (!modelExists) {
    const err = new Error(`Модель #${model_id} не знайдена`);
    err.statusCode = 422;
    throw err;
  }
  const doCreate = db.transaction(() => {
    const { lastInsertRowid } = db
      .prepare(
        "INSERT INTO equipment_units (model_id, serial_no, location, commissioned_at,
notes) VALUES (?, ?, ?, ?, ?)"
      )
      .run(model_id, serial_no, location ?? null, commissioned_at, notes ?? null);
    db.prepare("INSERT INTO health_state (unit_id, status, anomaly_score) VALUES (?,
'ok', 0)").run(lastInsertRowid);
    return lastInsertRowid;
  });
  const id = doCreate();
  return findById(id);
};
const update = (id, fields) => {
  const existing = findById(id);
  if (!existing) throw notFound(id);
  const allowed = ["location", "notes", "is_active", "hours_run"];
  const sets = [];
  const params = [];
  for (const key of allowed) {
    if (Object.prototype.hasOwnProperty.call(fields, key)) {
      sets.push(`_${key}=`);
      params.push(fields[key]);
    }
  }
  if (sets.length === 0) return existing;
  sets.push("updated_at=strftime('%Y-%m-%dT%H:%M:%SZ', 'now')");
  params.push(id);
  getDb()
    .prepare(`UPDATE equipment_units SET ${sets.join(",")} WHERE id=?`)
    .run(...params);
  return findById(id);
};
const softDelete = (id) => {

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 94   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

const existing = findById(id);
if (!existing) throw notFound(id);
getDb()
  .prepare("UPDATE equipment_units SET is_active=0, updated_at=strftime('%Y-%m-%dT%H:%M:%SZ','now') WHERE id=?")
  .run(id);
};
module.exports = { list, findById, create, update, softDelete };
backend/src/lib/passport_schema.js
"use strict";
const PASSPORT_SCHEMA = {
  type: "object",
  required: ["model_code", "display_name", "category", "weibull_eta_hours",
    "weibull_beta", "channels"],
  additionalProperties: true,
  properties: {
    model_code: { type: "string", minLength: 1 },
    display_name: { type: "string", minLength: 1 },
    category: { type: "string", minLength: 1 },
    weibull_eta_hours: { type: "number", exclusiveMinimum: 0 },
    weibull_beta: { type: "number", exclusiveMinimum: 0 },
    channels: {
      type: "array",
      minItems: 1,
      items: {
        type: "object",
        required: ["code", "class", "unit", "nominal", "operating_range",
          "critical_range"],
        additionalProperties: true,
        properties: {
          code: { type: "string", minLength: 1 },
          class: { type: "string", enum: ["T", "F", "E", "Q", "R", "V"] },
          unit: { type: "string" },
          nominal: { type: "number" },
          operating_range: { type: "array", items: { type: "number" }, minItems: 2,
            maxItems: 2 },
          critical_range: { type: "array", items: { type: "number" }, minItems: 2,
            maxItems: 2 },
          inertia_seconds: { type: "number", minimum: 0 },
          noise_sigma: { type: "number", minimum: 0 },
        },
      },
    },
    failure_modes: {
      type: "array",
      items: {
        type: "object",
        required: ["code", "name"],
        additionalProperties: true,
        properties: {
          code: { type: "string" },
          name: { type: "string" },
          affected_channels: { type: "array", items: { type: "string" } },
          typical_horizon_hours: { type: "number" },
        },
      },
    },
  },
};
module.exports = { PASSPORT_SCHEMA };

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 95   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

## Додаток Е Інтеграція мотедів Python

```
ml_service\training\train.py
from __future__ import annotations

import argparse
import json
import logging
import sys
from datetime import datetime, timezone
from pathlib import Path
from typing import Any

import joblib
import numpy as np
import pandas as pd
from sklearn.ensemble import (
    GradientBoostingRegressor,
    IsolationForest,
    RandomForestClassifier,
)
from sklearn.model_selection import GroupShuffleSplit

# Add project root to path so training/ can import from app/
PROJECT_ROOT = Path(__file__).parent.parent
sys.path.insert(0, str(PROJECT_ROOT))

from app.features import extract_features # noqa: E402

TRAINING_DATA_DIR = PROJECT_ROOT / "training_data"
MODELS_DIR = PROJECT_ROOT / "models"

logging.basicConfig(
    stream=sys.stdout,
    level=logging.INFO,
    format="%(asctime)s %(levelname)-8s %(message)s",
    datefmt="%Y-%m-%dT%H:%M:%S",
)
logger = logging.getLogger(__name__)

# -----
# Data loading and windowing
# -----

def load_training_data(model_code: str) -> list[dict[str, Any]]:
    """Load flat telemetry + ground-truth records for one model_code."""
    path = TRAINING_DATA_DIR / f"{model_code}.json"
    if not path.exists():
        logger.error(
            "Training data not found: %s\n\n"
            "To generate it, run the simulator in offline mode:\n"
            "    cd simulator\n"
            "    node src/offline.js --model-code %s --output"
            "../ml_service/training_data/%s.json\n"
            "\nThe simulator must be running and the CASS backend seeded with equipment"
            "models.",
            path, model_code, model_code,
        )
        sys.exit(1)
```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 96   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

with open(path, "r", encoding="utf-8") as fh:
    data = json.load(fh)

logger.info("Loaded %d telemetry rows for model_code=%s", len(data), model_code)
return data

def build_windowed_features(
    records: list[dict[str, Any]],
    passport: dict[str, Any],
    window_minutes: int = 60,
) -> pd.DataFrame:
    """Group records into 60-minute windows per unit and extract features.

    Expected record fields:
        unit_id, model_code, ts (ISO-8601), channel_code, value,
        true_health, active_mode (str | null), rul_true_hours

    Returns a DataFrame where each row is one window with feature columns
    plus target columns ``label`` (str mode code or "HEALTHY") and
    ``rul_hours`` (float).
    """
    df = pd.DataFrame(records)

    required = {"unit_id", "ts", "channel_code", "value"}
    missing = required - set(df.columns)
    if missing:
        raise ValueError(f"Training data missing required columns: {missing}")

    df["ts"] = pd.to_datetime(df["ts"], utc=True)
    df["window_idx"] = df["ts"].dt.floor(f"{window_minutes}min")

    # Target: use the most recent value in the window for active_mode and rul
    if "active_mode" not in df.columns:
        df["active_mode"] = None
    if "rul_true_hours" not in df.columns:
        df["rul_true_hours"] = np.nan

    rows: list[dict[str, Any]] = []

    groups = df.groupby(["unit_id", "window_idx"])
    total = len(groups)
    logger.info("Windowing %d groups (unit x 60-min window) ...", total)

    for (unit_id, window_ts), group in groups:
        window_records = group[["channel_code", "value"]].to_dict("records")
        if len(window_records) < 5:
            continue # skip nearly empty windows

        features = extract_features(window_records, passport)
        if not features:
            continue

        # Take the most recent row's labels
        last_row = group.sort_values("ts").iloc[-1]
        active_mode = last_row.get("active_mode")
        rul = last_row.get("rul_true_hours", np.nan)

        label = str(active_mode) if active_mode and pd.notna(active_mode) else
"HEALTHY"
        features["_unit_id"] = int(unit_id)
        features["_label"] = label

```

|      |      |               |        |      |                           |      |
|------|------|---------------|--------|------|---------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ІІЗ | Арк. |
|      |      | Власенко О.В. |        |      |                           | 97   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                           |      |

```

        features["_rul_hours"] = float(rul) if pd.notna(rul) else np.nan

        rows.append(features)

    if not rows:
        raise ValueError("No valid windows extracted – check training data format")

    result = pd.DataFrame(rows).fillna(0.0)
    logger.info(
        "Extracted %d windows, %d feature columns, labels: %s",
        len(result),
        len([c for c in result.columns if not c.startswith("_")]),
        result["_label"].value_counts().to_dict(),
    )
    return result

# -----
# Training
# -----

def train_model_bundle(
    model_code: str,
    windowed_df: pd.DataFrame,
    ) -> dict[str, Any]:
    """Train all sub-models and return the bundle dict."""
    feature_cols = sorted([c for c in windowed_df.columns if not
c.startswith("_")])

    X = windowed_df[feature_cols].values.astype(np.float64)
    y_label = windowed_df["_label"].values
    y_rul = windowed_df["_rul_hours"].values
    groups = windowed_df["_unit_id"].values

    # --- Train / validation split (80/20 by unit_id to avoid leakage) ---
    splitter = GroupShuffleSplit(n_splits=1, test_size=0.2, random_state=42)
    train_idx, val_idx = next(splitter.split(X, y_label, groups=groups))

    X_train, X_val = X[train_idx], X[val_idx]
    y_label_train, y_label_val = y_label[train_idx], y_label[val_idx]
    y_rul_train, y_rul_val = y_rul[train_idx], y_rul[val_idx]

    logger.info("Train set: %d windows | Val set: %d windows", len(X_train),
len(X_val))

    # --- 1. IsolationForest – fit ONLY on HEALTHY samples ---
    healthy_mask = y_label_train == "HEALTHY"
    X_healthy = X_train[healthy_mask]
    if len(X_healthy) < 10:
        logger.warning(
            "Only %d HEALTHY windows for IsolationForest – "
            "consider generating more healthy baseline data",
            len(X_healthy),
        )
        X_iso_fit = X_train # fallback: fit on all data
    else:
        X_iso_fit = X_healthy

    logger.info("Training IsolationForest on %d HEALTHY windows ...",
len(X_iso_fit))
    iso = IsolationForest(
        n_estimators=200,

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 98   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        contamination=0.05,
        max_samples="auto",
        random_state=42,
        n_jobs=-1,
    )
    iso.fit(X_iso_fit)

# --- 2. RandomForestClassifier (failure mode classification) ---
unique_labels = sorted(set(y_label_train))
logger.info("Training RandomForestClassifier on %d classes: %s ...",
len(unique_labels), unique_labels)
clf = RandomForestClassifier(
    n_estimators=300,
    max_depth=12,
    min_samples_leaf=3,
    class_weight="balanced",
    random_state=42,
    n_jobs=-1,
)
clf.fit(X_train, y_label_train)
clf_acc = float(np.mean(clf.predict(X_val) == y_label_val))
logger.info("Classifier val accuracy: %.3f", clf_acc)

# --- 3. GradientBoostingRegressor x 3 (quantile RUL regression) ---
# Only train where RUL is available (non-NaN)
rul_mask_train = np.isfinite(y_rul_train)
rul_mask_val = np.isfinite(y_rul_val)

X_rul_train = X_train[rul_mask_train]
y_rul_train_clean = y_rul_train[rul_mask_train]
X_rul_val = X_val[rul_mask_val]
y_rul_val_clean = y_rul_val[rul_mask_val]

logger.info("Training GBR quantile regressors on %d windows with RUL labels
...", len(X_rul_train))

rul_q10 = rul_q50 = rul_q90 = None
if len(X_rul_train) >= 20:
    common_gbr_params = dict(
        n_estimators=300,
        max_depth=4,
        learning_rate=0.05,
        min_samples_leaf=5,
        random_state=42,
    )
    rul_q10 = GradientBoostingRegressor(loss="quantile", alpha=0.1,
**common_gbr_params)
    rul_q10.fit(X_rul_train, y_rul_train_clean)

    rul_q50 = GradientBoostingRegressor(loss="quantile", alpha=0.5,
**common_gbr_params)
    rul_q50.fit(X_rul_train, y_rul_train_clean)

    rul_q90 = GradientBoostingRegressor(loss="quantile", alpha=0.9,
**common_gbr_params)
    rul_q90.fit(X_rul_train, y_rul_train_clean)

    if len(X_rul_val) > 0:
        preds_median = rul_q50.predict(X_rul_val)
        mae = float(np.mean(np.abs(preds_median - y_rul_val_clean)))
        logger.info("RUL q50 val MAE: %.1f hours", mae)
    else:

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 99   |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        logger.warning(
            "Not enough RUL-labelled windows (%d) to train quantile regressors
_ "
            "RUL prediction will be unavailable in this bundle",
            len(X_rul_train),
        )

# --- Feature importances and per-feature means ---
importances = dict(zip(feature_cols, clf.feature_importances_.tolist()))
feature_means = dict(zip(feature_cols, X_train.mean(axis=0).tolist()))

bundle: dict[str, Any] = {
    "isolation_forest": iso,
    "classifier": clf,
    "rul_q10": rul_q10,
    "rul_q50": rul_q50,
    "rul_q90": rul_q90,
    "feature_names": feature_cols,
    "feature_importances": importances,
    "feature_means": feature_means,
    "failure_modes": [m for m in unique_labels if m != "HEALTHY"],
    "version": "v1.0.0",
    "trained_at": datetime.now(timezone.utc).isoformat(),
}
return bundle

# -----
# Entry point
# -----

def get_available_model_codes() -> list[str]:
    """Return model codes for which training data exists."""
    if not TRAINING_DATA_DIR.exists():
        return []
    return [p.stem for p in TRAINING_DATA_DIR.glob("*.json")]

def train_one(model_code: str, passport: dict[str, Any]) -> None:
    """Load data, engineer features, train, and save bundle for one
model_code."""
    records = load_training_data(model_code)
    windowed_df = build_windowed_features(records, passport)
    bundle = train_model_bundle(model_code, windowed_df)

    MODELS_DIR.mkdir(parents=True, exist_ok=True)
    out_path = MODELS_DIR / f"{model_code}.pkl"
    joblib.dump(bundle, out_path, compress=3)
    logger.info("Saved bundle to %s (version=%s)", out_path, bundle["version"])

def main() -> None:
    parser = argparse.ArgumentParser(
        description="Train CASS ML model bundles from simulator-generated
training data.",
    )
    parser.add_argument(
        "--model-code",
        type=str,
        default=None,
        help="Equipment model code to train (e.g. SRV-R740-2U). "
        "Omit to train all available models.",
    )

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 100  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

    )
    parser.add_argument(
        "--passport",
        type=str,
        default=None,
        help=(
            "Path to a JSON file containing the equipment model passport. "
            "Required for proper contextual feature computation. "
            "If omitted, an empty passport is used (contextual features will be
zero)."
        ),
    )
    args = parser.parse_args()

    if not TRAINING_DATA_DIR.exists():
        logger.error(
            "training_data/ directory not found at %s\n\n"
            "Generate training data first by running the simulator in offline
mode:\n"
            "          cd simulator && node src/offline.js --output
../ml_service/training_data/",
            TRAINING_DATA_DIR,
        )
        sys.exit(1)

    # Load passport if provided; otherwise use empty (contextual features will
be zeros)
    if args.passport:
        with open(args.passport, "r", encoding="utf-8") as fh:
            passport = json.load(fh)
            logger.info("Using passport from %s", args.passport)
    else:
        passport = {"channels": []}
        logger.warning(
            "No --passport provided – contextual features (dev_from_nominal, "
            "time_above_op, etc.) will all be zero. "
            "Pass --passport path/to/passport.json for best results."
        )

    model_codes: list[str]
    if args.model_code:
        model_codes = [args.model_code]
    else:
        model_codes = get_available_model_codes()
        if not model_codes:
            logger.error("No training data files found in %s",
TRAINING_DATA_DIR)
            sys.exit(1)
        logger.info("Training all available models: %s", model_codes)

    for model_code in model_codes:
        logger.info("=" * 60)
        logger.info("Training model_code=%s", model_code)
        logger.info("=" * 60)
        try:
            train_one(model_code, passport)
        except Exception:
            logger.exception("Training failed for model_code=%s – skipping",
model_code)

    logger.info("Done. Bundles saved to %s", MODELS_DIR)

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіщук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 101  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        if __name__ == "__main__":
            main()

ml_service\app\inference.py
from __future__ import annotations

import logging
from typing import Optional

import numpy as np

from .features import build_feature_vector
from .registry import registry
from .schemas import PredictRequest, PredictResponse

logger = logging.getLogger(__name__)

# -----
# Heuristic fallback
# -----

def heuristic_predict(model_code: str, features: dict[str, float]) -> PredictResponse:
    """Compute anomaly score and RUL without a trained model.

    The heuristic aggregates deviation-related feature suffixes:
        __dev_from_nominal - normalised distance from nominal
        __time_above_op    - fraction of time above operating range
        __time_below_op   - fraction of time below operating range
        __in_critical      - fraction of time in critical zone (weighted x4)

    Final anomaly_score = clip(p80(indicators) * 0.5 + 0.05, 0, 0.95).
    RUL is inversely proportional to anomaly_score over a 0-720 hour range.
    """
    anomaly_indicators: list[float] = []

    for name, value in features.items():
        if name.endswith("__dev_from_nominal"):
            anomaly_indicators.append(abs(value))
        elif name.endswith("__time_above_op") or name.endswith("__time_below_op"):
            anomaly_indicators.append(float(value) * 2.0)
        elif name.endswith("__in_critical"):
            anomaly_indicators.append(float(value) * 4.0)

    # Also consider slope-based trend indicators
    for name, value in features.items():
        if name.endswith("__slope") and abs(value) > 0:
            anomaly_indicators.append(min(1.0, abs(value) * 10.0))

    if not anomaly_indicators:
        anomaly_score = 0.05
    else:
        raw = float(np.percentile(anomaly_indicators, 80))
        anomaly_score = min(0.95, raw * 1.0 + 0.05)

    # Find the most anomalous channel by dev_from_nominal
    worst_channel: Optional[str] = None
    worst_val = 0.0
    for name, val in features.items():
        if name.endswith("__dev_from_nominal") and abs(val) > worst_val:
            worst_val = abs(val)

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІПЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 102  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        worst_channel = name[: -len("__dev_from_nominal")]

# RUL: inversely proportional to anomaly_score, range 1-720 hours
rul = max(1.0, (1.0 - anomaly_score) * 720.0)

predicted_mode: Optional[str] = None
predicted_mode_conf: Optional[float] = None
if anomaly_score >= 0.3:
    predicted_mode = "unknown_degradation"
    predicted_mode_conf = round(anomaly_score, 4)

top_features: list[dict] = []
if worst_channel:
    top_features = [
        {"name": f"{worst_channel}__dev_from_nominal", "contribution":
round(worst_val, 4)}
    ]

return PredictResponse(
    anomaly_score=round(anomaly_score, 4),
    predicted_mode=predicted_mode,
    predicted_mode_conf=predicted_mode_conf,
    rul_hours=round(rul, 1),
    rul_lower_hours=round(rul * 0.6, 1),
    rul_upper_hours=round(rul * 1.5, 1),
    model_version="heuristic-v1",
    explanation={"top_features": top_features},
)

# -----
# ML bundle prediction
# -----

def _top_features_from_bundle(
    bundle: dict,
    features: dict[str, float],
    feature_names: list[str],
    n: int = 5,
) -> list[dict]:
    """Compute top-N feature contributions.

    Method: multiply the classifier's feature importances by the absolute
    deviation of the feature value from its mean importance-weighted value.
    Falls back to raw importances if per-feature mean is unavailable.
    """
    importances: dict[str, float] = bundle.get("feature_importances", {})
    if not importances:
        return []

    scored: list[tuple[float, str]] = []
    feature_means: dict[str, float] = bundle.get("feature_means", {})

    for name in feature_names:
        imp = importances.get(name, 0.0)
        if imp <= 0.0:
            continue
        val = features.get(name, 0.0)
        baseline = feature_means.get(name, 0.0)
        contribution = imp * abs(val - baseline)
        scored.append((contribution, name))

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 103  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        scored.sort(reverse=True)
        return [{"name": name, "contribution": round(score, 4)} for score, name in
scored[:n]]

def ml_predict(bundle: dict, features: dict[str, float]) -> PredictResponse:
    """Run the full ML prediction pipeline using a loaded model bundle."""
    iso_forest = bundle.get("isolation_forest")
    classifier = bundle.get("classifier")
    rul_models: Optional[dict] = bundle.get("rul_q10") and bundle    # check via
individual keys below
    feature_names: list[str] = bundle.get("feature_names", [])
    version: str = bundle.get("version", "unknown")
    failure_modes: list[str] = bundle.get("failure_modes", [])

    if not feature_names:
        logger.warning("Bundle has no feature_names – falling back to heuristic")
        return heuristic_predict("", features)

    # Step 1 – Build feature vector
    X = build_feature_vector(features, feature_names)

    # Step 2 – Anomaly score from IsolationForest
    if iso_forest is not None:
        raw_score = float(iso_forest.decision_function(X)[0])
        # decision_function returns higher values for normal samples.
        # We map it to [0, 1] where 1 = most anomalous.
        anomaly_score = float(np.clip(0.5 - raw_score / 2.0, 0.0, 1.0))
    else:
        logger.warning("Bundle %s has no isolation_forest – anomaly_score set to 0.1",
version)
        anomaly_score = 0.1

    # Step 3 – Failure mode classification
    predicted_mode: Optional[str] = None
    predicted_mode_conf: Optional[float] = None

    if classifier is not None and anomaly_score >= 0.3:
        try:
            proba = classifier.predict_proba(X)[0]
            best_idx = int(np.argmax(proba))
            predicted_mode = str(classifier.classes_[best_idx])
            predicted_mode_conf = round(float(proba[best_idx]), 4)
            # If the classifier labels the sample as HEALTHY, don't report a mode
            if predicted_mode.upper() == "HEALTHY":
                predicted_mode = None
                predicted_mode_conf = None
        except Exception:
            logger.exception("Classifier.predict_proba failed")

    # Step 4 – RUL quantile regression
    rul_hours: Optional[float] = None
    rul_lower_hours: Optional[float] = None
    rul_upper_hours: Optional[float] = None

    rul_q10 = bundle.get("rul_q10")
    rul_q50 = bundle.get("rul_q50")
    rul_q90 = bundle.get("rul_q90")

    # Support legacy bundles where RUL models were stored under "rul_regressor" dict
    if rul_q10 is None and rul_q50 is None:
        legacy = bundle.get("rul_regressor")

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 104  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        if isinstance(legacy, dict):
            rul_q10 = legacy.get("q10")
            rul_q50 = legacy.get("q50")
            rul_q90 = legacy.get("q90")

    if rul_q50 is not None:
        try:
            rul_hours = float(max(0.0, rul_q50.predict(X)[0]))
        except Exception:
            logger.exception("rul_q50.predict failed")
    if rul_q10 is not None:
        try:
            rul_lower_hours = float(max(0.0, rul_q10.predict(X)[0]))
        except Exception:
            logger.exception("rul_q10.predict failed")
    if rul_q90 is not None:
        try:
            rul_upper_hours = float(max(0.0, rul_q90.predict(X)[0]))
        except Exception:
            logger.exception("rul_q90.predict failed")

    # Step 5 – Explanation: top feature contributions
    top_feats = _top_features_from_bundle(bundle, features, feature_names, n=5)

    return PredictResponse(
        anomaly_score=round(anomaly_score, 4),
        predicted_mode=predicted_mode,
        predicted_mode_conf=predicted_mode_conf,
        rul_hours=round(rul_hours, 1) if rul_hours is not None else None,
        rul_lower_hours=round(rul_lower_hours, 1) if rul_lower_hours is not None else
None,
        rul_upper_hours=round(rul_upper_hours, 1) if rul_upper_hours is not None else
None,
        model_version=version,
        explanation={"top_features": top_feats},
    )

# -----
# Unified entry point
# -----

def predict(req: PredictRequest) -> PredictResponse:
    """Route a predict request to the ML bundle or heuristic fallback."""
    bundle = registry.get(req.model_code)

    if bundle is None:
        logger.info(
            "No trained bundle for model_code=%s – using heuristic predictor",
            req.model_code,
        )
        return heuristic_predict(req.model_code, req.features)

    return ml_predict(bundle, req.features)
ml_service\app\features.py
from __future__ import annotations

from typing import Any

import numpy as np

try:

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 105  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

from scipy.stats import skew as scipy_skew, kurtosis as scipy_kurtosis

_SCIPY_AVAILABLE = True
except ImportError:
    _SCIPY_AVAILABLE = False

# -----
# Internal helpers
# -----

def _skew(arr: np.ndarray) -> float:
    """Pearson's moment coefficient of skewness (bias-corrected if scipy available)."""
    if len(arr) < 3:
        return 0.0
    if _SCIPY_AVAILABLE:
        return float(scipy_skew(arr, bias=False))
    std = float(np.std(arr, ddof=1))
    if std < 1e-12:
        return 0.0
    return float(np.mean(((arr - np.mean(arr)) / std) ** 3))

def _kurtosis(arr: np.ndarray) -> float:
    """Excess kurtosis (Fisher's definition, bias-corrected if scipy available)."""
    if len(arr) < 4:
        return 0.0
    if _SCIPY_AVAILABLE:
        return float(scipy_kurtosis(arr, fisher=True, bias=False))
    std = float(np.std(arr, ddof=1))
    if std < 1e-12:
        return 0.0
    return float(np.mean(((arr - np.mean(arr)) / std) ** 4) - 3)

def _slope(arr: np.ndarray) -> float:
    """Slope of a least-squares linear fit over the array (per-sample rate)."""
    n = len(arr)
    if n < 2:
        return 0.0
    x = np.arange(n, dtype=float)
    coeffs = np.polyfit(x, arr, 1)
    return float(coeffs[0])

def _fft_bands(arr: np.ndarray, sample_rate_hz: float = 1.0) -> tuple[float, float, float, float]:
    """Compute fractional spectral power in three bands and dominant frequency.

    Bands:
        low: 0 - 10 Hz
        mid: 10 - 100 Hz
        high: 100+ Hz

    Returns:
        (fft_low, fft_mid, fft_high, fft_peak)
        All power fractions are in [0, 1] and sum to ~1.
        fft_peak is the dominant frequency in Hz.
    """
    n = len(arr)
    if n < 4:
        return 0.0, 0.0, 0.0, 0.0

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 106  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

windowed = arr * np.hanning(n)
spectrum = np.abs(np.fft.rfft(windowed))
power = spectrum ** 2

total_power = float(np.sum(power))
if total_power < 1e-30:
    return 0.0, 0.0, 0.0, 0.0

freqs = np.fft.rfftfreq(n, d=1.0 / sample_rate_hz)

mask_low = freqs <= 10.0
mask_mid = (freqs > 10.0) & (freqs <= 100.0)
mask_high = freqs > 100.0

fft_low = float(np.sum(power[mask_low]) / total_power)
fft_mid = float(np.sum(power[mask_mid]) / total_power)
fft_high = float(np.sum(power[mask_high]) / total_power)

peak_idx = int(np.argmax(power))
fft_peak = float(freqs[peak_idx])

return fft_low, fft_mid, fft_high, fft_peak

# -----
# Public API – offline feature extraction
# -----

def extract_features(
    telemetry_window: list[dict[str, Any]],
    passport: dict[str, Any],
) -> dict[str, float]:
    """Extract a named feature vector from a raw telemetry window.

    Args:
        telemetry_window: List of telemetry row dicts. Each dict must have
            at minimum ``channel_code`` (str) and ``value`` (float).
            Optionally ``ts`` (ISO-8601 string, used for ordering).
        passport: Equipment model passport dict as stored in the DB.
            Must contain ``channels`` list with per-channel metadata.

    Returns:
        Flat dict mapping feature_name -> float.
        Missing channels are silently ignored (not filled); the returned
        dict covers only channels present in the window.

    Notes:
        The function is deliberately side-effect-free and does not access
        any external state. It is used both in the offline training
        pipeline and can be called from unit tests.
    """
    # Build channel metadata lookup: code -> channel dict
    ch_meta: dict[str, dict] = {ch["code"]: ch for ch in passport.get("channels", [])}

    # Group values by channel, preserving insertion order (== time order)
    by_channel: dict[str, list[float]] = {}
    for row in telemetry_window:
        code = row.get("channel_code", "")
        val = row.get("value")
        if code and val is not None:
            try:

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 107  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        by_channel.setdefault(code, []).append(float(val))
    except (TypeError, ValueError):
        pass

features: dict[str, float] = {}

for code, values in by_channel.items():
    if not values:
        continue

    arr = np.array(values, dtype=float)
    ch = ch_meta.get(code, {})
    ch_class = ch.get("class", "")
    nominal = ch.get("nominal", 0.0)
    op_range: list[float] = ch.get("operating_range", [0.0, 1.0])
    crit_range: list[float] = ch.get("critical_range", [float("-inf"),
float("inf")])

    # Protect against degenerate ranges
    op_low = float(op_range[0])
    op_high = float(op_range[1])
    op_width = max(1.0, abs(op_high - op_low))
    crit_low = float(crit_range[0])
    crit_high = float(crit_range[1])

    mean_val = float(np.mean(arr))
    p = code # prefix alias

    # --- Statistical features ---
    features[f"{p}__mean"] = mean_val
    features[f"{p}__std"] = float(np.std(arr, ddof=0))
    features[f"{p}__min"] = float(np.min(arr))
    features[f"{p}__max"] = float(np.max(arr))
    features[f"{p}__median"] = float(np.median(arr))
    features[f"{p}__skew"] = _skew(arr)
    features[f"{p}__kurt"] = _kurtosis(arr)
    features[f"{p}__slope"] = _slope(arr)
    features[f"{p}__last"] = float(arr[-1])

    # --- Contextual features (relative to passport) ---
    features[f"{p}__dev_from_nominal"] = (mean_val - nominal) / op_width
    features[f"{p}__time_above_op"] = float(np.mean(arr > op_high))
    features[f"{p}__time_below_op"] = float(np.mean(arr < op_low))

    # Critical range fraction: fraction of points falling in the critical band.
    # Critical range is defined as the dangerous zone – may be above or below
    # the operating range depending on the channel.
    if crit_high > crit_low:
        features[f"{p}__in_critical"] = float(np.mean((arr >= crit_low) & (arr <=
crit_high)))
    else:
        # Inverted critical (e.g. fan RPM where critical is [0, 1500])
        features[f"{p}__in_critical"] = float(np.mean(arr <= crit_high))

    # --- Spectral features for V-class channels ---
    if ch_class == "V":
        # Assume 1 Hz equivalent spacing when exact timestamps are absent.
        # In production the backend should pass the actual sample rate.
        fft_low, fft_mid, fft_high, fft_peak = _fft_bands(arr, sample_rate_hz=1.0)
        features[f"{p}__fft_low"] = fft_low
        features[f"{p}__fft_mid"] = fft_mid
        features[f"{p}__fft_high"] = fft_high

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 108  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |

```

        features[f"{p}__fft_peak"] = fft_peak

# --- Cross-channel correlations ---
# Correlate T-V, T-F, E-F pairs.
# Only compute when both channels have the same number of samples.
class_map: dict[str, list[str]] = {}
for code in by_channel:
    ch_class = ch_meta.get(code, {}).get("class", "")
    if ch_class:
        class_map.setdefault(ch_class, []).append(code)

pair_classes = [("T", "V"), ("T", "F"), ("E", "F")]
for cls_a, cls_b in pair_classes:
    codes_a = class_map.get(cls_a, [])
    codes_b = class_map.get(cls_b, [])
    for ca in codes_a:
        for cb in codes_b:
            arr_a = np.array(by_channel[ca], dtype=float)
            arr_b = np.array(by_channel[cb], dtype=float)
            min_len = min(len(arr_a), len(arr_b))
            if min_len < 4:
                continue
            arr_a = arr_a[:min_len]
            arr_b = arr_b[:min_len]
            std_a = float(np.std(arr_a))
            std_b = float(np.std(arr_b))
            if std_a < 1e-12 or std_b < 1e-12:
                corr = 0.0
            else:
                corr = float(np.corrcoef(arr_a, arr_b)[0, 1])
                if not np.isfinite(corr):
                    corr = 0.0
            features[f"corr_{ca}_{cb}"] = corr

return features

# -----
# Public API – inference-time vector assembly
# -----

def build_feature_vector(
    features: dict[str, float],
    feature_names: list[str],
) -> np.ndarray:
    """Assemble an ordered numpy array from a named feature dict.

    Missing feature names are filled with 0.0. Extra keys in ``features``
    that are not in ``feature_names`` are silently ignored.

    Args:
        features: Named feature dict as received in PredictRequest.features.
        feature_names: Ordered list of feature names as stored in the model bundle.

    Returns:
        2-D numpy array of shape (1, len(feature_names)) with dtype float64.
    """
    vec = np.array(
        [features.get(name, 0.0) for name in feature_names],
        dtype=np.float64,
    ).reshape(1, -1)
    return vec

```

|      |      |               |        |      |                          |      |
|------|------|---------------|--------|------|--------------------------|------|
|      |      | Фіцук Є.О.    |        |      | ІІЗ.КР.Б – 121 – 26 – ПЗ | Арк. |
|      |      | Власенко О.В. |        |      |                          | 109  |
| Змн. | Арк. | № докум.      | Підпис | Дата |                          |      |