

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

**Навчально-науковий інститут
«Інститут інформаційних технологій в економіці»**

Кафедра інформаційних систем в економіці

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Інформаційні управляючі системи і технології»**

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

Форма навчання: очна (денна)

КВАЛІФІКАЦІЙНА МАГІСТЕРСЬКА РОБОТА

на тему: «Розроблення веб-платформи для транспортних задач»

здобувача Микитюка Максима Олександроча

Науковий керівник: к.т.н., доцент, Крячок Олександр Степанович

(підпис)

**Робота допущена до захисту перед екзаменаційною
комісією з атестації здобувачів вищої освіти (ЕК)**

Завідувач кафедри: _____
(науковий ступінь, учене звання, ПІБ)

(підпис)

Київ 2025

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА**

Навчально-науковий інститут «Інститут інформаційних технологій в економіці»

Кафедра інформаційних систем в економіці

**ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА
«Інформаційні управляючі системи і технології»**

галузь знань 12 Інформаційні
технології

спеціальність 122 Комп'ютерні науки

ПОГОДЖЕНО:

Керівник проєктної групи (гарант)
освітньо-професійної програми

_____ Устенко С.В.
« ____ » _____ 202_ р.

ЗАТВЕРДЖУЮ:

Завідувач кафедри інформаційних
систем в економіці

_____ Тішков Б.О.
« ____ » _____ 202_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ

здобувача вищої освіти Микитюка Максима Олександровича
очної форми навчання

на підготовку кваліфікаційної магістерської роботи
на тему: «Розроблення веб-платформи для транспортних задач»

Тему затверджено наказом ректора Університету від « ____ » _____ 202_ р. № ____

Кваліфікаційна магістерська робота виконується на матеріалах

План кваліфікаційної магістерської роботи

Розділ I

ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ
УПРАВЛЯЮЧИХ СИСТЕМ У ЛОГІСТИЦІ

Розділ II

СТРУКТУРА І ХАРАКТЕРИСТИКА СИСТЕМИ

Розділ III

РОЗРОБЛЕННЯ ПРОЄКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ

Об'єкт дослідження: інформаційні управляючі системи для підприємств логістичного профілю, орієнтовані на автоматизацію процесів, пов'язаних із перевезеннями.

Предмет дослідження: технології та інструменти автоматизації збору, обробки, збереження та подання логістичних даних у межах веб-застосунку.

Мета кваліфікаційної магістерської роботи: розробити інформаційний веб-застосунок для моніторингу логістики вагонів, що автоматизує процес оновлення даних, забезпечує їх збереження у базі даних та відображення через веб-інтерфейс для подальшого використання у процесах логістичного управління.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I

Дослідити особливості логістичної сфери та автоматизації процесів у логістиці. Здійснити аналіз існуючих рішень, обґрунтувати обрані технології та рішення для розробки.

У розділі II

Надати вичерпну характеристику системи, її структури та функціональності.

У розділі III

Розробити веб-платформу, надати результати роботи та описати етапи розробки.

Завдання підготував

науковий керівник

к.т.н., доцент, Крячок Олександр Степанович

« ____ » _____ 202_ р.

Завдання одержав

здобувач

Микитюк Максим Олександрович

« ____ » _____ 202_ р.

Реферат

Кваліфікаційна магістерська робота містить 74 сторінки, 5 таблиць, 18 рисунків, список використаних джерел з 51 найменувань, додатки.

«Розроблення веб-платформи для транспортних задач»

Об'єктом дослідження кваліфікаційної магістерської роботи є інформаційні управляючі системи для підприємств логістичного профілю, орієнтовані на автоматизацію процесів, пов'язаних із перевезеннями.

Предметом дослідження є технології та інструменти автоматизації збору, обробки, збереження та подання логістичних даних у межах веб-застосунку.

Мета і завдання дослідження. Основною метою кваліфікаційної магістерської роботи є розробка інформаційного веб-застосунку для моніторингу логістики вагонів, що автоматизує процес оновлення даних, забезпечує їх збереження у базі даних та відображення через веб-інтерфейс для подальшого використання у процесах логістичного управління.

Відповідно до поставленої мети визначені такі *завдання*:

- провести аналіз існуючих рішень у сфері транспортної логістики та інформаційних систем;
- спроектувати та реалізувати реляційну базу даних на основі PostgreSQL для структурованого зберігання отриманої інформації;
- визначити та реалізувати алгоритми, що забезпечують автоматизацію моніторингу й оптимізацію логістичних процесів;
- створити веб-інтерфейс для доступу до даних та управління логістикою;
- розробити веб-застосунок для інтеграції зі стороннім API з можливістю автоматизованого отримання логістичних даних;
- провести тестування розробленого рішення з метою перевірки його стабільності, точності та продуктивності;
- забезпечити хмарне розгортання системи на платформі Azure, що дозволяє гарантувати її масштабованість, доступність та безпечність.

Теоретична, методична та практична значущість отриманих результатів. Під час дослідження було досліджено предметну область, проаналізовано оптимальні шляхи розробки та обрано ефективні технологічні інструменти. В ході роботи було розроблено веб-платформу для транспортних задач, в які входить, зокрема, регулярне оновлення даних про перевезення, заповнення бази даних та їх відображення у формі звітів через веб-інтерфейс.

Рік виконання кваліфікаційної магістерської роботи – 2025.

Рік захисту роботи – 2025.

Ключові слова: веб-додаток, логістика, автоматизація.

Рецензія

на кваліфікаційну магістерську роботу
здобувача вищої освіти

Микитюка Максима Олександровича

на тему: « Розроблення веб-платформи для транспортних задач»

Тема магістерської роботи є надзвичайно актуальною в умовах стрімкої цифровізації логістичних процесів та необхідності підвищення ефективності перевезень у галузі залізничного транспорту. Автор обґрунтував важливість створення автоматизованої веб-платформи для моніторингу логістики вагонів, що відповідає реальним потребам підприємств у сфері управління транспортними ресурсами. В роботі акцентовано увагу на необхідності уніфікації джерел даних, їх оперативної обробки, автоматизації збору інформації та зручного подання користувачам. Особливу доцільність проєкту підкреслює воєнний контекст, який вимагає підвищеного рівня безпеки та автономності логістичних систем.

Робота вирізняється високим рівнем теоретичної підготовки та практичної реалізації. Проведено ґрунтовний аналіз предметної області, систематизовано сучасні підходи до побудови логістичних інформаційних систем, виконано огляд і порівняння існуючих рішень (Transporeon, RailSys, RLMS, CargoTrack тощо), що демонструє аналітичне мислення автора.

У технічному аспекті робота реалізована на базі сучасного стеку технологій: ASP.NET Core, PostgreSQL, Razor, DI, MVC, DTO, фонові обробка, хмарне розгортання (Azure). Створено UML-діаграми, описано архітектуру системи, механізми захисту та масштабування. Робота містить візуалізацію ключових компонентів, діаграми компонентів, таблиці моделей, пояснення структури сховища — усе це вказує на комплексний підхід до проєктування системи.

Робота магістранта Микитюка М.О. містить ряд позитивних рис та характеристик, зокрема, здійснено ґрунтовний теоретичний аналіз предметної області з використанням сучасних підходів до архітектури програмного забезпечення; також в роботі реалізовано прототип веб-застосунку із повноцінним інтерфейсом, фоновим збором даних та хмарним розгортанням.

Але, водночас, робота б виграла, якщо б в розділі з реалізацією системи було докладніше описано виклики, з якими зіштовхнувся магістрант під час розробки, та як вони були вирішені. Це зауваження носить рекомендаційний характер і не знижує загального позитивного враження.

Таким чином, магістерська робота Микитюка Максима є самостійним, якісним і науково обґрунтованим дослідженням. Робота відповідає вимогам, що висуваються до випускних кваліфікаційних робіт освітнього рівня «магістр» і рекомендована до захисту в Екзаменаційній комісії з високою оцінкою.

Д.е.н., професор кафедри економічної кібернетики
Київського національного університету
імені Тараса Шевченка

Олена БАЖЕНОВА



Відгук
на кваліфікаційну магістерську роботу
здобувача вищої освіти навчально-наукового інституту «Інститут інформаційних
технологій в економіці» КНЕУ

Микитюка Максима Олександровича
на тему «**Розроблення веб-платформи для транспортних задач**»

Тема роботи є актуальною через потребу підприємств у сучасних програмних інструментах для автоматизованого моніторингу логістичних даних. У сфері транспортної логістики швидкий і безперешкодний доступ до інформації є критично важливим для ухвалення ефективних рішень. Попри наявність API-провайдерів, бракує зручних рішень для обробки й візуалізації даних. Запропонована Микитюком М.О. веб-платформа вирішує цю задачу, підвищуючи прозорість процесів і зменшуючи залежність від людського чинника.

Магістерська робота відзначається глибоким опрацюванням літератури та предметної області, що забезпечило розуміння поточних тенденцій у сфері логістичних інформаційних систем. Автор застосував конструктивні підходи до розробки, зокрема, використання актуальних технологій, інтеграцію з API та реалізацію зручного веб-інтерфейсу. В результаті було створено програмний застосунок, який відповідає реальним потребам галузі й має високу практичну цінність.

У межах магістерської роботи реалізовано цілісну інформаційну систему, яка включає реляційну базу даних на PostgreSQL, веб-додаток з інтуїтивно зрозумілим інтерфейсом, а також фонові сервіси для автоматичного збору та оновлення даних. Для перевірки працездатності та тестування системи створено демо-сервер, що імітує роботу з реальними даними.

Робота містить важливі теоретичні узагальнення щодо побудови логістичних інформаційних систем, а також практичні рекомендації, що можуть бути використані при впровадженні подібних рішень у корпоративне середовище. Це підвищує прикладну значущість дослідження.

Серед деяких недоліків можна відзначити відсутність елементів просунутої інтелектуальної обробки даних, таких як прогнозування чи машинне навчання, що могло б розширити функціональність системи. Крім того, впровадження подібних рішень у реальному середовищі пов'язане з низкою технічних, організаційних та безпекових ризиків, які потребують додаткового аналізу та планування.

Загалом вважаю, що кваліфікаційна магістерська робота на тему «Розроблення веб-платформи для транспортних задач» відповідає встановленим вимогам та рекомендується до захисту з високою оцінкою, а її автор *Микитюк*

Максим Олександрович заслуговує на присвоєння освітньо-кваліфікаційного рівня «магістр» за спеціальністю 122 – «Комп’ютерні науки».

Науковий керівник к.т.н., доцент, доцент кафедри
інформаційних систем в економіці

Навчально-наукового інституту «Інститут

інформаційних технологій в економіці» КНЕУ



О.С. Крячок

«20» травня 2025 р.

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ У ЛОГІСТИЦІ	8
1.1. Аналіз існуючих логістичних інформаційних систем.....	8
1.1.1. Дослідження інформаційних систем у логістиці	8
1.1.2. Огляд існуючих рішень	14
1.2. Обґрунтування вибору підходів і технологій.....	17
РОЗДІЛ 2. СТРУКТУРА І ХАРАКТЕРИСТИКА СИСТЕМИ	21
РОЗДІЛ 3. РОЗРОБЛЕННЯ ПРОЄКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ.....	27
3.1. Проєктування бази даних	27
3.1.1. Логічна модель БД	27
3.1.2. Фізична модель БД.....	30
3.1.3. Контрольний приклад та обсяг БД.....	31
3.3. Програмне забезпечення.....	33
3.3.1. Проєктування ПЗ.....	33
3.3.2. Вибір архітектури ПЗ.....	34
3.3.3. Інструментальні засоби	37
3.3.4. Тестування ПЗ	39
3.4. Технічне забезпечення.....	41
3.5. Реалізація системи.....	43
3.6. Економічна ефективність	51
3.7. Оцінка ризиків при впровадженні	57
3.8. Організаційне забезпечення	62
ВИСНОВКИ	65
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	69
ДОДАТКИ.....	72

ВСТУП

Упродовж останнього десятиліття інформаційні технології зазнали значного прогресу, суттєво трансформуючи всі сфери підприємницької діяльності — від управлінських процесів до безпосереднього виробництва товарів і надання послуг. Їхній вплив особливо відчутний у галузях, де щоденно обробляються великі масиви даних, що надходять з різних джерел — як внутрішніх, так і зовнішніх. У таких умовах критично важливою стає здатність систем до оперативної, точної та ефективної обробки інформації. Саме тут на перший план виходить роль автоматизації бізнес-процесів та попередньої програмної обробки даних. Вони дозволяють не лише оптимізувати робочі процеси, але й мінімізувати вплив людського чинника на виконання рутинних завдань. Завдяки цьому працівники мають змогу зосередитися на стратегічних аспектах діяльності, які вимагають професійної компетентності, креативного підходу та аналітичного мислення. Таким чином, впровадження сучасних ІТ-рішень стає не просто засобом підвищення продуктивності, а важливим чинником конкурентоспроможності підприємства в умовах динамічного ринку.

Однією зі сфер які зазнали впливу є логістика. Це зумовлено тим, що логістичні процеси нерозривно пов'язані з постійним моніторингом транспортних засобів, як з метою контролю виконання перевезень, так і для ефективного управління матеріально-технічними активами підприємства [1, 2, 3]. Йдеться, зокрема, про потребу в регулярному відслідковуванні технічного стану транспорту, плануванні та фіксації ремонтних робіт, а також контролю пробігу, витрат пального та дотримання маршрутів. У таких умовах застосування автоматизованих інформаційних систем стає вирішальним чинником забезпечення оперативності, точності та прозорості логістичних операцій [4]. Класичні ERP-системи з модульною структурою поступово втрачають актуальність, поступаючись місцем новітнім рішенням, що забезпечують комплексну інтеграцію бізнес-процесів [5]. Такі системи здатні самотійно, без залучення людини, оновлювати інформацію у реальному часі, здійснювати первинну обробку даних, виявляти відхилення від

нормативних показників та формувати звітність у зручному для користувачів форматі. У результаті, працівники отримують доступ до актуальних і структурованих даних, що значно підвищує ефективність прийняття управлінських рішень та сприяє оптимізації логістичних витрат, відкриваючи простір для вдосконалення логістичних маршрутів [6].

На даний момент тема достатньо досліджена в науковій спільноті, існує достатньо робіт які покривають логістичні інформаційні системи абсолютно різних масштабів, зокрема ті які включають екосистеми для підтримки товарних магазинів та ті які використовують для функціонування інтернет речей. Втім, якщо орієнтуватись на більш вузьке, практичне завдання, як-от суто моніторинг даних які надаються провайдером, їх аналіз та конкретна імплементація, джерел стає менше. Зокрема важливим фактором є постійне оновлення технологій, що призводить до невеликих змін у найбільш оптимальних шляхах впровадження тих чи інших рішень на прикладному рівні. В роботі було використано ряд джерел українських авторів, розміщених у загальному доступі. Вони надають різні перспективи на інформаційні логістичні системи, що є корисним у досягненні мети роботи. Також було використано посилання на статистичні дані, документацію окремих технологій та оглядові статті в інтернеті від розробників програмного забезпечення з інформацією про деталі розробки подібних додатків. Останнє є чи не найбільш корисним, адже відомості від людей які працюють в даній сфері станом на поточний момент є цінною.

Розробка інформаційних систем для моніторингу та управління логістикою вагонів має ключове значення в умовах інтеграції транспортних систем. Такі системи сприяють не лише вдосконаленню технологічних процесів, але й підвищенню прозорості логістичного ланцюга, що є актуальним у контексті цифровізації економіки. Важливість цього дослідження підкріплена зростаючими потребами підприємств у скороченні витрат і поліпшенні якості обслуговування клієнтів.

Актуальність насамперед визначається роботою з даними. У сучасних умовах розвитку логістичної галузі своєчасність, точність та доступність даних відіграють

ключову роль у процесі ухвалення ефективних управлінських рішень. Особливо це стосується підприємств, що здійснюють перевезення за допомогою наземного транспорту, де навіть незначні затримки або помилки у даних можуть призвести до значних фінансових втрат або порушень логістичних ланцюгів [1]. Багато компаній активно використовують API, надані транспортними сервісами, для отримання необхідної інформації. Проте, попри доступність цих даних, компанії часто не мають зручного інструментарію для їх постійного моніторингу, обробки, збереження та подання у зручному форматі. У цьому контексті створення автоматизованої системи, здатної ефективно збирати, зберігати та візуалізувати логістичні дані, є актуальним завданням, що відповідає реальним потребам галузі [2, 7].

Мета дослідження — розробка інформаційного веб-застосунку для моніторингу логістики вагонів, що автоматизує процес оновлення даних, забезпечує їх збереження у базі даних та відображення через веб-інтерфейс для подальшого використання у процесах логістичного управління.

Об'єктом дослідження є інформаційна управляюча система для підприємств логістичного профілю, орієнтована на автоматизацію процесів, пов'язаних із перевезеннями.

Предметом дослідження виступають технології та інструменти автоматизації збору, обробки, збереження та подання логістичних даних у межах веб-застосунку.

Ключова проблема — уніфікація даних із різних джерел (ERP, GPS, звіти тощо). Обробка великих масивів інформації потребує оптимізації, щоб уникнути перевантаження системи. Також важливі безперебійність роботи, особливо в кризових умовах, і захист даних. Для ефективності система має бути простою у використанні, щоб диспетчери й менеджери могли працювати без технічних труднощів.

У межах роботи розглядається процес розробки програмного забезпечення, призначеного для автоматизації обробки та оновлення даних у корпоративному середовищі. Основна функціональність розроблюваної системи полягає у використанні фонових сервісів, які забезпечують регулярне та безперервне

оновлення інформації з визначених джерел. Отримані та оброблені дані автоматично завантажуються до бази даних підприємства, що сприяє забезпеченню їхньої актуальності, цілісності та доступності для користувачів різних рівнів. Крім того, кінцева розробка містить веб-інтерфейс, який дозволяє зручно переглядати, аналізувати та інтерпретувати зібрану інформацію. Такий підхід дозволяє не лише мінімізувати вплив людського чинника в рутинних операціях, але й створює передумови для підвищення прозорості внутрішніх процесів, що, у свою чергу, позитивно впливає на загальну ефективність управлінських рішень у межах підприємства.

Для досягнення поставленої мети визначено такі завдання:

- провести аналіз існуючих рішень у сфері транспортної логістики та інформаційних систем;
- спроектувати та реалізувати реляційну базу даних на основі PostgreSQL для структурованого зберігання отриманої інформації;
- визначити та реалізувати алгоритми, що забезпечують автоматизацію моніторингу й оптимізацію логістичних процесів;
- створити веб-інтерфейс для доступу до даних та управління логістикою;
- розробити веб-застосунок для інтеграції зі стороннім API з можливістю автоматизованого отримання логістичних даних;
- провести тестування розробленого рішення з метою перевірки його стабільності, точності та продуктивності;
- забезпечити хмарне розгортання системи на платформі Azure, що дозволяє гарантувати її масштабованість, доступність та безпечність.

Також для досягнення поставленої мети було проведено всебічний аналіз наукових і технічних джерел, що охоплюють як друковані праці українських і зарубіжних авторів, так і сучасні електронні ресурси, зокрема документацію до технологій, статті з профільних сайтів, дослідження у сфері логістики та розробки інформаційних систем. На їх основі спроектовано та реалізовано веб-застосунок для моніторингу логістики, його було протестовано в умовах, наближених до

реальних, для перевірки функціональності, стійкості до навантажень, точності обробки даних та зручності користування.

Для реалізації розробки важливе застосування сучасних інструментів, які дозволятимуть системі гідно зустрічати виклики великих обсягів даних та мінливості перевезень, даватимуть можливість гнучко відповідати на вимоги підприємства та забезпечуватимуть надійність та безпеку даних.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ТА АНАЛІЗ ПІДХОДІВ ДО СТВОРЕННЯ ІНФОРМАЦІЙНИХ УПРАВЛЯЮЧИХ СИСТЕМ У ЛОГІСТИЦІ

1.1. Аналіз існуючих логістичних інформаційних систем

1.1.1. Дослідження інформаційних систем у логістиці

Перевезення, отримання своєчасних відомостей про них та прийняття рішень на основі останніх даних є критично важливим для компаній, які займаються логістикою. Вона має вплив не лише на прибутки, а й на імідж та життєздатність компанії як такої, а в умовах війни деякі товари та їх своєчасна доставка мають вплив на країну (в локальних масштабах) та на людей, які їх очікують. Дослідження [8] стверджує, що оптимізовані логістичні рішення дозволяють підприємствам швидше реагувати на зміни ринку та забезпечувати безперебійне постачання. Це підвищує їхню стійкість і прибутковість у довгостроковій перспективі.

Основними аспектами управління логістикою є планування руху, моніторинг місцезнаходження транспорту, управління завантаженням і розвантаженням, а також оцінка технічного стану [9]. Розроблення рішень для автоматизації роботи з даними, а саме їх обробку, збереження та аналіз, збільшує точність, оперативність прийняття рішень, покращує ефективність працівників які за них відповідають та зменшує кількість непотрібної роботи, так званого «тертя», як-от пошук потрібної інформації на робочій пошті або в масиві не зв'язаних між собою на перший погляд документів. Втім, логістичні інформаційні системи спрямовані на забезпечення всіх етапів, включаючи координацію діяльності, управління матеріальним потоком і поповнення запасів, що реалізується завдяки інтеграції апаратного та програмного забезпечення. Використання інформаційних технологій у виробничо-логістичних системах дозволяє підвищити загальну ефективність ланцюга поставок, підтримуючи прийняття рішень і вдосконалення планування на основі спеціалізованих інформаційних платформ [10].

Інформаційні технології допомагають досягнути оптимізаційних задач, забезпечуючи автоматизацію та інтеграцію процесів через ERP, MRP, DRP та інші системи [8]. Вони підвищують точність прогнозування, оптимізують управління запасами й покращують координацію постачань, сприяючи зниженню витрат і підвищенню загальної ефективності логістичних операцій. Дослідження [11] підкреслює зростаюче значення інформаційної логістики в умовах цифрової трансформації бізнесу, особливо в контексті глобалізації та зростаючих вимог до управління інформаційними потоками. Попри її потенціал, більшість країн, зокрема Україна, поки що перебувають у фазі фрагментарного впровадження таких рішень. Інформаційна логістика розглядається як управління інформаційними процесами за логістичними принципами та як стратегічне використання ІТ для підвищення ефективності логістичних операцій, що робить її ключовим напрямком розвитку сучасного бізнесу. Також, дослідження [12] акцентує увагу на необхідності впровадження сучасних інформаційних технологій для комплексної оцінки логістичної діяльності підприємств. Авторка зазначає, що підприємства часто орієнтуються на кількісні показники, ігноруючи економічну ефективність, що може обмежувати можливості оптимізації. Формування бази даних логістичних показників дозволяє не лише оцінювати продуктивність і надійність логістичної системи, а й використовувати бенчмаркінг для виявлення прихованих витрат і потенціалу вдосконалення, що сприяє підвищенню конкурентоспроможності підприємств. А власне без впровадженої інформаційної системи створення такої бази показників буде неефективним та затратним в плані часу та ресурсів.

В умовах сучасного розвитку технологій особлива увага також приділяється автоматизації створення звітів, які використовуються як для внутрішнього обігу, так і для взаємодії із зовнішніми підрозділами. Такий підхід дозволяє покращити обґрунтованість управлінських рішень, мінімізувати затримки в обміні даними та підвищити загальну продуктивність роботи залізничного транспорту. Звіти можуть бути періодичними, що створюються за шаблоном, або формуватись за запитом, що забезпечує гнучкість у роботі з інформацією. Окремий акцент робиться на розподілі інформаційних потоків для уникнення перевантаження каналів зв'язку.

Це особливо важливо в умовах великого обсягу даних, що циркулюють у системах транспортної логістики. Рациональний розподіл інформації між виконавцями та керівництвом сприяє зниженню рівня дублювання даних і підвищенню оперативності прийняття рішень. Основні операції обробки даних, як правило, виконуються на рівні виконавців, що дозволяє функціонувати в автономному режимі, забезпечуючи стабільність та безперебійність роботи системи. Зокрема, використання комп'ютерної техніки та програмного забезпечення дозволяє замінити традиційні методи обміну інформацією електронними комунікаціями. Це забезпечує не лише пришвидшення обробки даних, але й можливість інтеграції гнучкого зворотного зв'язку, що є критично важливим у динамічних умовах транспортних перевезень [13].

У контексті безпеки систем транспортної логістики вагомим є дотримання вимог до захисту інформації. Ідентифікація та авторизація користувачів, контроль цілісності системи, аудит подій та архівація даних формують основу стабільної роботи системи. Використання криптографічних засобів і захисту від вірусів, їхня інтеграція є важливим напрямом для подальшого розвитку технологій. Ці аспекти сприяють підвищенню працездатності систем і надійності передачі конфіденційної інформації, що має вирішальне значення для ефективного управління вагонними парками у залізничному транспорті.

Також, якщо зосередитись на вагонному парку, управлінням якого займається компанія, то вдало створена розробка може допомогти з оптимізацією використання ресурсів. Наприклад, використання методів кластерного аналізу може допомогти у групуванні вагонів за параметрами їхньої завантаженості чи призначення [14]. Себто, подібним чином можна посприяти рівномірному розподілу вантажів і зменшенню експлуатаційних витрат. Звісно, подібні системи не можуть існувати ізольовано, і зазвичай вони складаються з інфраструктурних мереж і мають багатокомпонентний, складений характер. Це включає взаємодію з системами управління станціями, складськими приміщеннями та перевантажувальними пунктами. Подібна інтеграція забезпечує єдність

інформаційного середовища та зменшує ймовірність виникнення помилок через розрізненість даних.

У дослідженні логістики вагонів мають велике значення способи роботи з даними – даних багато, вони часто можуть приймати неоднорідну структуру, а рішення які приймаються на їх основі варіюються в залежності від мети. Наприклад, моделі теорії графів використовуються для побудови оптимальних маршрутів, тоді як методи лінійного програмування допомагають розв’язувати задачі розподілу ресурсів. Подібні підходи знижують витрати на перевезення та забезпечують високий рівень обслуговування клієнтів.

Хоч і менш очевидний та пріоритетний особливо в умовах кризової ситуації, в логістиці безперечно задіяний і екологічний аспект. Зменшення порожнього пробігу вагонів і оптимізація маршрутів сприяють зниженню викидів парникових газів, що відповідає сучасним вимогам до сталого розвитку транспорту [15].

Логістичні компанії здійснюють обробку великих обсягів інформації, що стосується переміщення вантажів, організації перевезень, моніторингу стану транспорту та планування логістичних операцій. Усі ці процеси вимагають високої точності, швидкості обробки даних та злагодженості між різними підрозділами компанії. Інформація, яку обробляють логістичні компанії, включає дані про вантажі, маршрути, транспорти, транспортні засоби, графіки перевезень, а також умови зберігання і доставки вантажу. Для ефективного управління цими процесами необхідна система, яка дозволить автоматизувати збори, обробку та аналіз інформації.

Обробка інформації в логістичній компанії починається з отримання даних від різних джерел: це можуть бути внутрішні системи, зовнішні API (як у випадку з залізничними перевезеннями), а також документи та звіти [16]. Для автоматизації цього процесу використовуються спеціалізовані інформаційні системи, які здатні оперативно зібрати дані з різних джерел і перетворити їх у зручний для подальшого використання формат. Однією з основних задач є інтеграція зовнішніх систем, таких як API залізничних компаній або інших транспортних операторів, що надають дані про рух транспорту, зміни в графіках чи ситуацію на маршруті.

Ключовими етапами обробки інформації є її трансформація, зберігання та аналіз. Трансформація даних передбачає їх приведення до єдиного формату, що забезпечить коректне використання в системі. Це важливий етап, оскільки отримані дані можуть бути в різних форматах, наприклад, у вигляді JSON або XML, тому вони потребують відповідних перетворень для інтеграції з базою даних. Власне, зберігання інформації здійснюється в реляційних базах даних, що дозволяє забезпечити надійність та швидкий доступ до великих обсягів даних, гарантуючи їх цілісність.

Після зберігання даних у базі, наступним етапом є їх аналіз. Аналіз може здійснюватися для різних цілей: оптимізації маршрутів, передбачення затримок у перевезеннях, аналізу ефективності роботи транспорту та персоналу, оцінки запасів та потреб у постачаннях. Для цього використовуються різноманітні аналітичні інструменти, починаючи від простих звітів і закінчуючи складними аналітичними платформами, які можуть застосовувати машинне навчання або прогностичні моделі для виявлення закономірностей у великих наборах даних.

Один із важливих аспектів автоматизації процесів обробки та аналізу даних в логістичних компаніях полягає в обробці і відображенні інформації в режимі реального часу. Наприклад, для контролю за станом вагонів чи інших транспортних одиниць важливо мати змогу миттєво реагувати на будь-які зміни, що стосуються їх місцеположення, стану чи зупинок. Інформаційні системи в таких випадках можуть забезпечувати моніторинг в реальному часі, сповіщення про зміни стану транспорту та автоматичне оновлення даних [17].

Крім того, важливим завданням є управління обсягами інформації, що зберігаються в системі. Логістичні компанії, особливо великі, працюють з величезними обсягами даних, що вимагає розробки механізмів для ефективного їх зберігання, пошуку та обробки. Одним із підходів до вирішення цієї проблеми є нормалізація бази даних та використання індексації, що дозволяє прискорити пошук необхідної інформації та знизити навантаження на систему при великих обсягах даних. Звісно, системи для автоматизації логістичних процесів водночас повинні бути гнучкими і здатними адаптуватися до змін у зовнішніх умовах. Це

особливо важливо у випадку, коли компанія працює з різними видами транспорту або змінюються умови перевезень. Система повинна бути здатна швидко вносити зміни в графіки або маршрути перевезень, враховувати нові вимоги до безпеки або змінювати план за необхідності.

Одним із важливих аспектів є також інтеграція таких інформаційних систем із зовнішніми партнерами, постачальниками або клієнтами. Це дозволяє автоматизувати передачу даних і забезпечити безперервний обмін інформацією між усіма учасниками процесу. Дослідження [18] підтверджує ключову роль цифрових технологій у підвищенні ефективності логістики, підкреслюючи важливість інтегрованих інформаційних систем. Недостатнє використання ІТ-рішень може призвести до перебоїв у постачанні та втрати конкурентних переваг. І при впровадженні, така система навпаки принести колосальні переваги – вона може підключати зовнішні служби для прогнозування ситуації на маршрутах, аналізу погодних умов або пошуку альтернативних шляхів. Висока швидкість обробки та аналізу даних, інтеграція з різними джерелами інформації та можливість швидкого прийняття рішень створюють потужну основу для оптимізації всіх етапів логістичних операцій.

Щорічне перевезення значних обсягів вантажів, зокрема 26,2 млн тонн сипучих матеріалів, нафти та будівельних матеріалів [19], вимагає ефективного управління. Припустимо, що тоннаж одного вагона складає 71 тону [20]. Тоді отримуємо приблизно 69 тис. вагонів, що курсують протягом року, створюють значне навантаження на транспортну інфраструктуру, що ускладнюється нестабільною ситуацією на окупованих територіях. Враховуючи наявність понад 1447 станцій та 2268 зупинок (станом на 2016 рік) [21], та необхідність збереження даних про кожну операцію з парком вагонів (зокрема пов'язані з технічним станом), оптимізація моніторингу логістики є критично важливою для підвищення ефективності перевезень і раціонального використання ресурсів.

1.1.2. Огляд існуючих рішень

З розвитком інформаційних технологій, сфера логістики залізничних перевезень почала активно інтегруватися із сучасними веб-застосунками, що дозволяють здійснювати моніторинг, аналіз та управління рухом вагонів. Відомо кілька розвинутих платформ, які виконують подібні функції – ці рішення охоплюють різні аспекти логістики, включаючи відстеження вантажів, прогнозування маршрутів та управління складськими запасами.

Одним із таких великих рішень є система Transporeon – платформа для управління різними типами перевезень включно із залізничним транспортом. Вона забезпечує їх повну видимість, дозволяючи користувачам отримувати дані в реальному часі про розташування вагонів, прогнозувати терміни доставки та ефективно управляти вантажами (рисунок 1.1.1.). Ця система також інтегрується з іншими інформаційними системами для автоматизації та оптимізації процесів [22]. Але як часто буває з подібними рішеннями, що розробляються на масового користувача, а не під конкретну компанію, вона не дає можливостей для гнучкості малому та середньому бізнесу, особливо якщо їхній кейс відрізняється від більшості компаній.

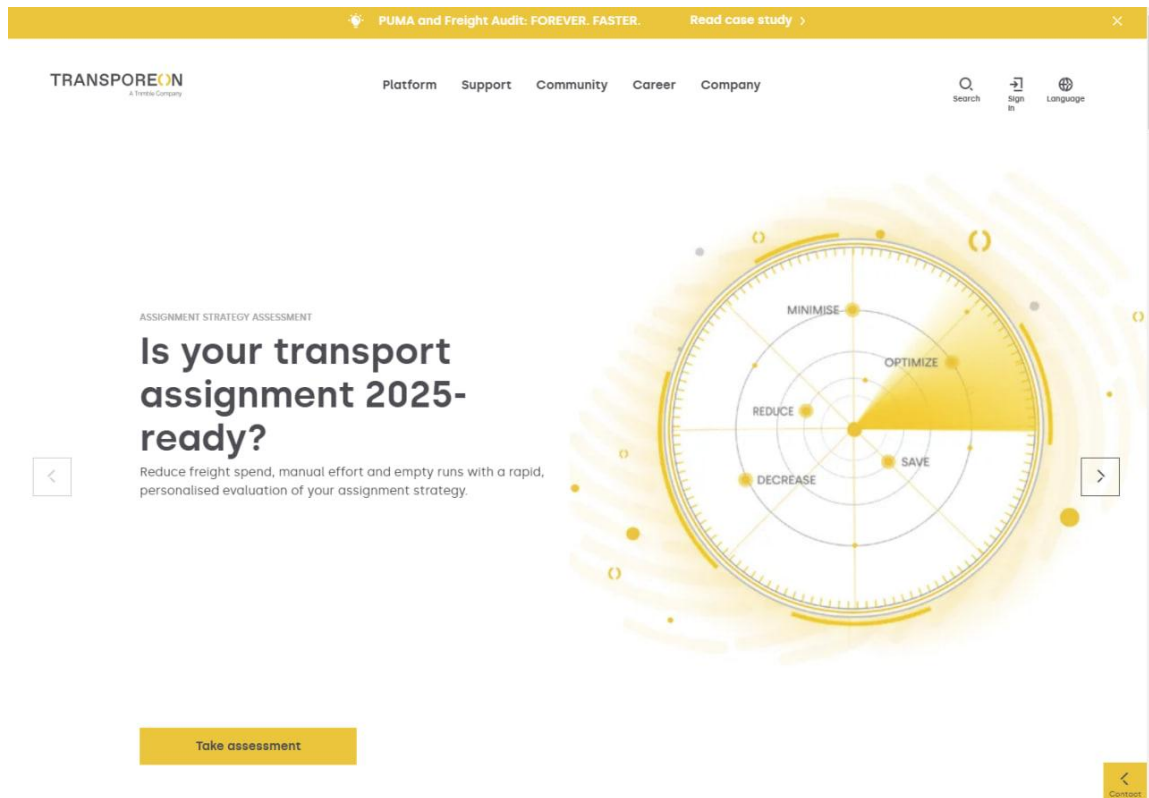


Рисунок 1.1.2.1. – Сайт сервису Transporeon

Іншою веб-системою яка пропонує схожий функціонал є RailSys, яка надає комплексні рішення для моделювання та аналізу залізничного транспорту. Вона включає інструменти для симуляції руху поїздів, управління розкладами, а також обробки даних про вантажі та інфраструктуру [23]. Повний функціонал даного рішення представлений на рисунку 1.1.2. Однак цей продукт орієнтований більше на великих операторів залізничного транспорту, ніж на невеликі компанії або локальні проекти. Система вимагає значних витрат на впровадження та налаштування, а її інтерфейс може бути занадто складним для кінцевого користувача, що створює бар'єри для інтеграції в більш дрібні компанії.

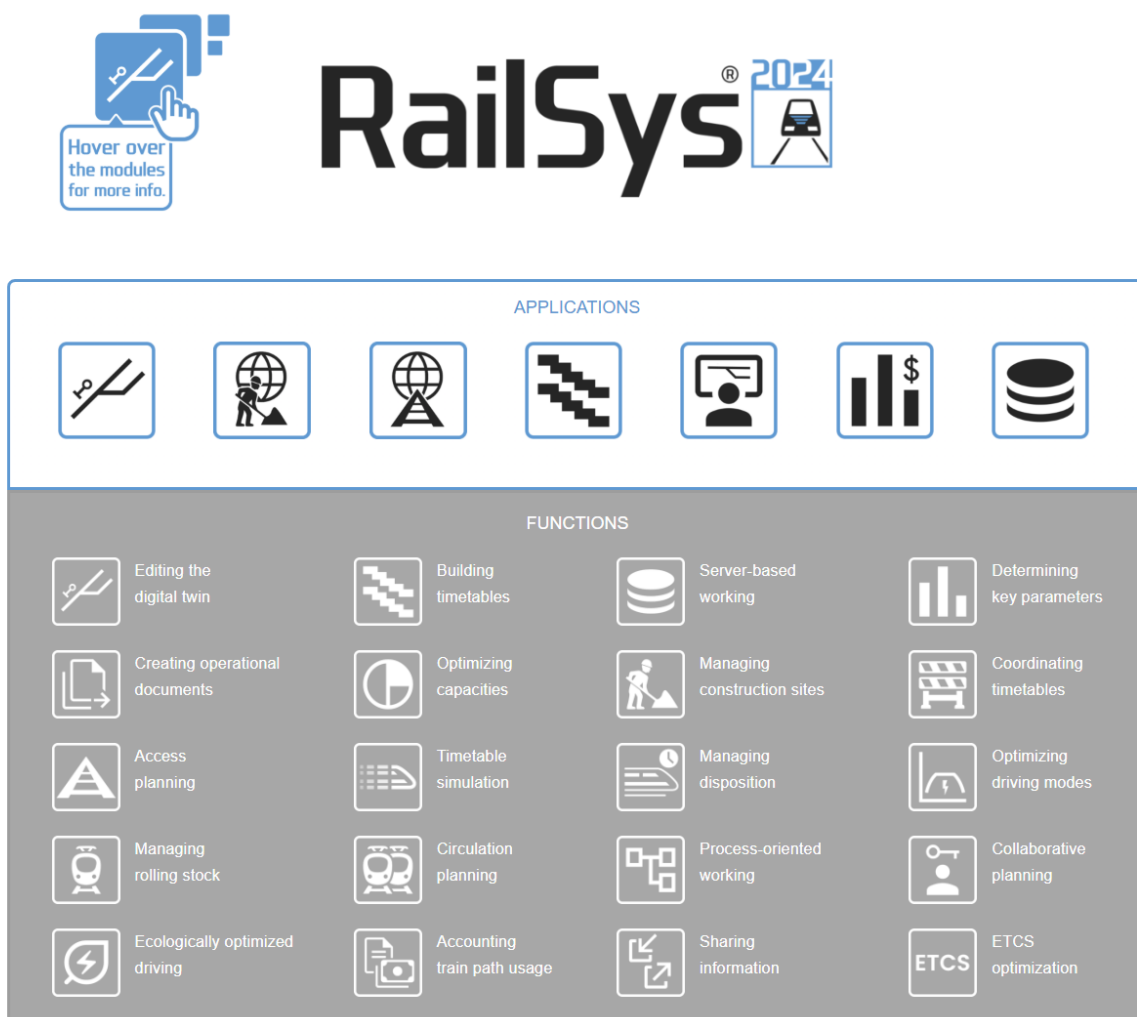


Рис. 1.1.2.2. – Сайт сервісу RailSys з описом функцій

Також слід згадати RLMS – Railway Logistics Management System, веб-додаток для відстеження вантажів у реальному часі та управління розкладами перевезень. Ця система дозволяє інтегрувати дані з різних джерел і забезпечує користувачів зручним інтерфейсом для управління вантажами і складання оптимальних маршрутів [24]. Та знову, система потребує значних інвестицій в налаштування та підтримку, а її можливості для персоналізації та адаптації під конкретні умови є обмеженими, як і у вищеназваних конкурентів.

Інноваційна платформа CargoTrack дещо схожа, вона дозволяє відстежувати вантажі на всіх етапах транспортування, використовуючи IoT-технології та забезпечує інтеграцію з іншими транспортними засобами, такими як вантажівки та судна [25]. Втім, і ця система має високі вимоги до обладнання і з'єднання, що може

бути невідомим для більшості локальних перевізників, а також вимагає значних затрат на впровадження та обслуговування.

Крім того, існують рішення, орієнтовані на оптимізацію технічного стану вагонів, як, наприклад, Asset Health Management Systems (AHMS), які використовуються для моніторингу стану технічних засобів, включаючи вагонні парки. Ці системи дають змогу прогнозувати можливі поломки та оптимізувати технічне обслуговування, знижуючи витрати на ремонти та забезпечуючи безперебійність перевезень [26]. Проте вони, як правило, орієнтовані на великі залізничні оператори та потребують складної інфраструктури для ефективного функціонування.

На ринку існує велика кількість різноманітних рішень для управління логістикою вагонів, що включають різні функціональні можливості, від моніторингу перевезень до прогнозування технічного стану вагонів [27]. Однак одним із найбільших недоліків більшості існуючих рішень є висока вартість впровадження та складність у налаштуванні, що обмежує їх застосування в рамках дрібних і середніх компаній. Крім того, через умови воєнного часу окрім великих витрат та недоліків гнучкості присутні також і безпекові ризики. Через те що інформація часто може бути недоступною а залізниця є об'єктом, критично важливим для держави, розробка окремого дотатка під ситуацію конкретної компанії-перевізника є більш доцільним рішенням, особливо якщо враховуючи що розробка такої системи та її впровадження в діяльність компанії є більш ніж доступною витратою.

1.2. Обґрунтування вибору підходів і технологій

При розробці вибір правильних технологій є критично важливим для забезпечення ефективності, масштабованості та надійності системи. Основна архітектура модулю для роботи з даними, його структура, продиктована функціоналом зовнішнього API, особливостями роботи з ним. Саме він дозволяє

отримати дані необхідні підприємству. Для цього використовуються бекграунд-сервіси, які автоматизують процес отримання даних, їх обробку та перетворення в зручний формат для збереження в базі даних. Для перетворення даних з формату JSON застосовуються кастомні конвертори, що дозволяють здійснювати точний мапінг полів і забезпечувати ефективну обробку великих обсягів даних в потоковому режимі. Використання бекграунд-сервісів дозволяє реалізувати асинхронний доступ до API та оновлення даних у реальному часі без впливу на продуктивність основного додатку.

У проектуванні застосовувались діаграми UML, які призначені і часто використовуються для опису та моделювання систем. Їх використання допомагає чітко формулювати вимоги до системи, структурувати її компоненти та визначити функціональні можливості. Поза контекстом розробленого рішення це також є корисним інструментом для командної роботи, оскільки дає змогу чітко передати архітектурні рішення та логіку взаємодії між компонентами всім учасникам процесу розробки. Якщо ж поглянути на це зі сторони додатка який було розроблено, подібні діаграми можуть ще й краще пояснити систему людям які будуть нею користуватись та розробникам, у разі залучення їх до процесів оптимізації та масштабування.

Архітектура ґрунтується на принципах MVC (Model-View-Controller) та Clean Architecture. Вони – широко розповсюджена практика у світі сучасної розробки, оскільки роблять код підтримуваним та масштабованим. Архітектура MVC забезпечує розділення логіки обробки даних, інтерфейсу користувача та бізнес-логіки, завдяки чому додаток стає зрозумілим та зручним [28]. Clean Architecture забезпечує чітке розмежування різних рівнів додатку (на відміну від MVC, це більш глобальний концепт, який, наприклад, охоплює ще й рівень даних), що сприяє зручності, гнучкості та розширюваності системи [29].

Використання паттерну репозиторіїв у розробці створює можливості для абстракції доступу до даних, що полегшує тестування та підвищує підтримуваність коду. Репозиторії зберігають логіку роботи з базою даних у окремому класі, що

спрощує модифікації та обслуговування додатку, дозволяючи замінювати реалізації доступу до даних без зміни інших компонентів системи [30].

Останнім важливим елементом є Dependency Injection (DI), яка дозволяє ефективно управляти залежностями між компонентами. Використання DI дає змогу інкапсулювати створення залежностей, що забезпечує більш зручну заміну реалізацій та тестування окремих компонентів системи. Цей підхід підвищує гнучкість та знижує рівень зв'язності між різними частинами додатку.

Один із перших і найважливіших етапів — вибір основної платформи. Сучасний ринок пропонує широкий спектр технологій, що відрізняються за функціональністю, продуктивністю та сферою застосування, про що детальніше йдеться у статті [31]. Для безпосередньо розробки програмної частини було обрано платформу .NET та технологію ASP.NET Core, яка є постійно оновлюваною, підтримуваною, має великий вибір із доступних бібліотек і є одним із поширених виборів для розробки високофункціональних веб-додатків [32]. Дана платформа є вельми сприятливою для бізнес-рішень, про що йдеться в статті [33], а стаття [34] також вказує про особливо високу ефективність при роботі з даними, що є критично важливим при більших обсягах даних, як у логістиці.

В якості сховища було обрано саме реляційні бази даних – вони дозволяють чітко і структуровано зберігати інформацію про вагонний парк, перевезення та інші логістичні дані. Використання нормалізації забезпечує мінімізацію дублювання і підвищення ефективності обробки. Окрім того, індексація полів значно прискорює виконання запитів, що є важливим у випадку роботи з великими обсягами даних. Встановлення зв'язків між таблицями в базі даних дозволяє зберігати цілісність даних і полегшує обробку запитів. Щодо конкретного рішення, то для зберігання та обробки даних у веб-додатку доцільно використовувати PostgreSQL, оскільки ця система керування базами даних є відкритим програмним забезпеченням, підтримує складні механізми індексації та роботу з JSON-форматом. Це забезпечує швидкий доступ до інформації та підвищує ефективність обробки логістичних задач. Потім, потрібно вибрати оптимальну ORM або технологію доступу до даних.

EF Core є одним із сучасних рішень, що забезпечує зручність роботи з даними та високу продуктивність додатка.

Важливо, як комунікує додаток з API. Найчастіше використовується протокол HTTP, а сама архітектура API будується за принципами REST. Однак вибір підходу залежить від вимог проєкту, тому необхідно аналізувати умови та адаптувати рішення відповідно до конкретних потреб.

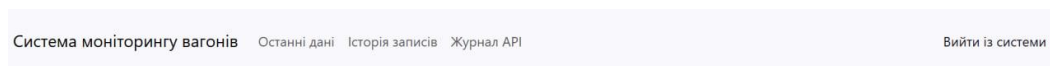
Що стосується проектування інтерфейсу, для його створення використовується Razor. Ця технологія дозволяє створювати динамічні сторінки, інтегруючи серверний код із HTML-розміткою. Razor дає можливість ефективно працювати з моделями та даними, отриманими з сервера, створюючи зручний і інтуїтивно зрозумілий інтерфейс для користувачів. Для управління аутентифікацією та авторизацією користувачів застосовується ASP.NET Identity, що забезпечує високий рівень безпеки і зручність у реалізації механізмів входу та реєстрації користувачів. Завдяки цьому можливо ефективно реалізовувати різні рівні доступу та системно працювати в умовах високих вимог до безпеки.

Існує теж безліч допоміжних технологій, які варто враховувати. Сюди входять маппери, SignalR, кешування та інструменти роботи з JSON. Вони сприяють оптимізації обробки даних та підвищенню продуктивності системи. Зважаючи на значні обсяги інформації, правильний вибір технологічного стеку забезпечує ефективність і масштабованість розробки.

Такий набір технологій та методів насамперед зумовлено потребами забезпечення високої ефективності, безпеки, масштабованості та підтримуваності системи. Кожен з обраних підходів дозволяє досягти оптимальних результатів у розробці, гарантуючи зручність роботи з даними, гнучкість інтерфейсу та надійність архітектури додатку. Що не менш важливо, вони використовуються та підтримуються в комерційній розробці на даний момент, що зумовлено їхньою надійністю та сучасністю.

РОЗДІЛ 2. СТРУКТУРА І ХАРАКТЕРИСТИКА СИСТЕМИ

Розроблений веб-застосунок є прикладом спеціалізованого програмного забезпечення, орієнтованого на потреби підприємства, що функціонує у сфері промислової логістики. Система розміщена на внутрішніх серверах компанії та призначена для щоденного використання працівниками, які займаються моніторингом перевезень, обробкою логістичної інформації та аналітикою. Архітектура застосунку включає три основні компоненти: базу даних, веб-інтерфейс та фоновий сервіс. Фоновий сервіс здійснює автоматизовану комунікацію з зовнішніми джерелами даних, зокрема з сервером даних, звідки отримує актуальну інформацію щодо вантажних перевезень. Після отримання, дані проходять попередню обробку та зберігаються у внутрішній базі даних підприємства. Веб-інтерфейс (рисунок 2.1.) застосунку надає користувачам зручний доступ до інформації: вона відображається у структурованому вигляді, з можливістю фільтрації, сортування та експорту у форматі Excel для подальшої роботи.



Вітаємо в системі!

Для подальшої роботи натисніть одне з посилань на панелі зверху

Рисунок 2.1. – Початкове вікно програми

Ці опції дозволяють оперативно відстежувати переміщення вантажів, контролювати технічний стан вагонів та аналізувати ефективність їхнього використання. Зокрема, система надає можливість визначити кількість здійснених поїздок кожним вагоном, виявити нерівномірності у навантаженні та ідентифікувати ресурси, що використовуються неефективно. Крім того,

збереження історичних даних у базі створює передумови для проведення глибокої аналітики — зокрема, аналізу виконаних замовлень, виявлення тенденцій у логістичних процесах та формування статистичної звітності. Таким чином, застосунок не лише автоматизує рутинні операції, а й виступає як інструмент стратегічного планування та підвищення ефективності логістичних процесів.

Основне призначення розробки це автоматизація процесів отримання та відображення даних через веб-додаток. Також ключовим моментом є фільтрація, перетворення та нормалізація даних та збереження їх в БД, для використання у майбутньому. Розробка складається з різних модулів – вони комунікують між собою та мають визначені ролі, що допомагає уникнути так званого spaghetti code – нерозбірливо написаної програмної частини (а зараз в контексті поточної роботи і хаотичної технічної, організаційної архітектури). Підтримуваність, ефективність, надійність – основні пріоритети системи.

Що стосується функціональної характеристики, то для її розробки та проектування було створено діаграму UML use-case, яка зображає можливі дії користувача із системою та безпосередньо всі варіації її поведінки. Дану діаграму наведено на рисунку 2.2.

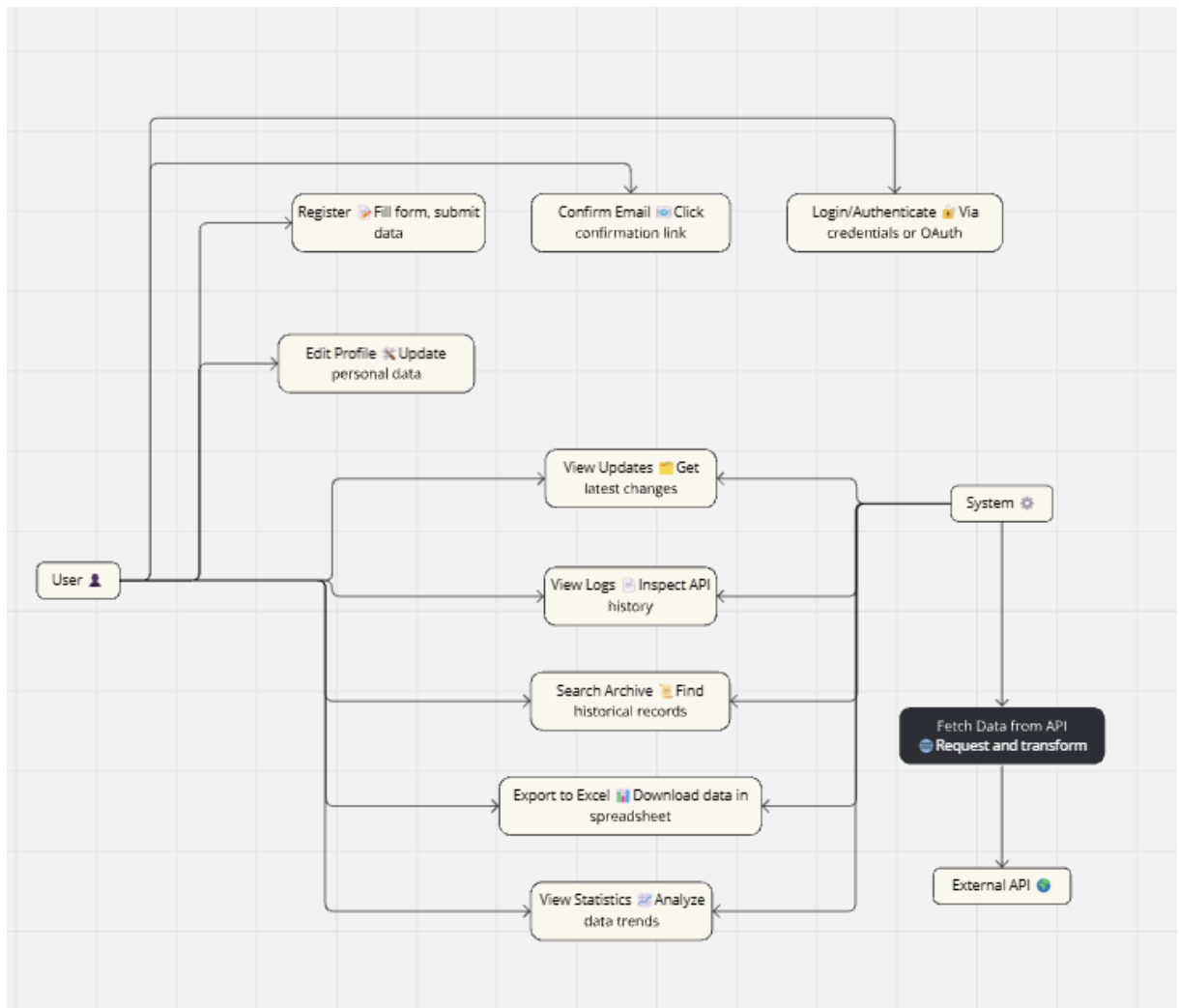


Рис. 2.2. – Діаграма використання системи

Однією з головних частин є модуль збору та обробки даних. Цей компонент отримує інформацію через API залізниці, перетворює її у формат, зручний для подальшої обробки, і передає в базу даних. Для ефективного збору даних використовуються асинхронні бекграунд-сервіси (додаток А), які відповідають за можливості системи реагувати на зміни у реальному часі та регулярно оновлювати інформацію про стан вагонів. Ці сервіси реалізовані з використанням черг для обробки запитів та збереження результатів у базі даних.

Для взаємодії з користувачами та відображення даних за допомогою технологій Razor та ASP.NET Identity було створено веб-інтерфейс. Ці інструменти дозволяють створити інтерактивний інтерфейс для моніторингу транспорту, перегляду звітів, а також для налаштування профілю користувача. UI також включає систему аутентифікації та авторизації, що забезпечує безпеку доступу до

чутливої інформації про вантажі та маршрути. Приклад сторінки, повністю створеною базовою функцією ASP.NET Identity, профілю користувача міститься в додатку Б.

Архітектура програми побудована на основі паттерну MVC (Model-View-Controller) з метою впровадження чіткого розділу логіки додатка, користувацького інтерфейсу і даних. Це забезпечує зручність для розробників при подальшому розширенні та підтримці системи. Згідно з цим паттерном, бізнес-логіка розміщується в моделях, сервісах та інших структурах, а контролери відповідають за обробку запитів від користувачів та передачу даних до відповідних представлень.

Частина системи, що забезпечує доступ до даних, реалізована через паттерн репозиторіїв. Репозиторії забезпечують абстракцію над конкретною реалізацією доступу до бази даних, що спрощує тестування та модифікацію коду. Це дозволяє гнучко змінювати схему бази даних або переходити на інші технології зберігання даних без суттєвого впливу на інші компоненти системи.

Dependency Injection дозволяє легко тестувати та модифікувати різні частини системи, оскільки залежності передаються в компоненти через конструктори, а не створюються безпосередньо всередині них. Він є неодмінним складником розробки на платформі .NET. Такий підхід знижує рівень жорсткої прив'язаності між компонентами та підвищує гнучкість архітектури.

Для відображення архітектури та взаємодії основних компонентів системи було створено діаграму компонентів яка представлена на рисунку 2.3. Ця діаграма слугує ключовим інструментом для візуалізації архітектурної структури програмного забезпечення, зокрема функціональної взаємодії між модулями.

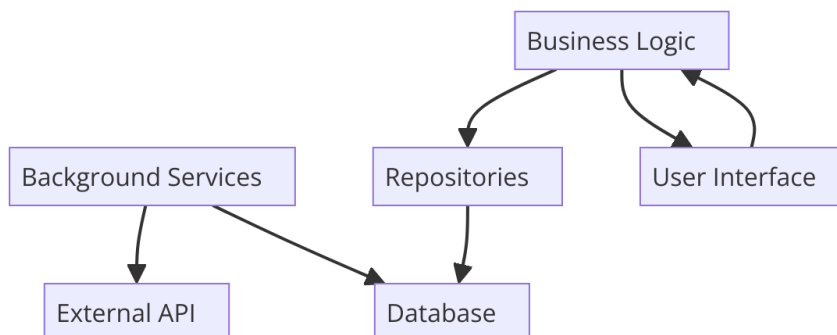


Рис. 2.3. – Діаграма компонентів

На діаграмі зображені основні компоненти системи, кожен із яких відповідає за певний набір завдань, а також взаємозв'язки між ними. Компоненти бізнес-логіки займають центральне місце в системі та виступають своєрідним "серцем" програмного забезпечення. Вони відповідають за обробку запитів, виконання алгоритмів, взаємодію з базою даних і забезпечення правильності роботи ключових функцій системи. Також на діаграмі окремо виділені модулі обробки даних, які забезпечують отримання, трансформацію та передачу інформації між компонентами. Ці модулі взаємодіють як з внутрішніми джерелами даних (база даних, кеш-сервіси), так і з зовнішніми (сторонні API, інтеграційні сервіси). Умовні стрілки на діаграмі вказують на напрямок передачі даних між модулями, що дозволяє зрозуміти потоки інформації в системі.

Ключові елементи UML-діаграми включають компоненти користувацького інтерфейсу, що ініціюють запити до бізнес-логіки і відображають результати користувачам, бізнес-логіку, компоненти бази даних та модулі інтеграції зі сторонніми системами, які відповідають за отримання інформації з API або інших зовнішніх джерел і приведення її до потрібного формату.

Діаграма компонентів дозволяє оцінити структуру та ієрархію системи, зрозуміти, які частини взаємодіють між собою, і проаналізувати залежності між компонентами. Це є важливим етапом у процесі проектування, оскільки допомагає знайти оптимальні шляхи взаємодії, зменшити складність реалізації та уникнути зайвих дублювань функціональності.

Загалом система має модульну структуру, що дозволяє легко додавати нові функціональні можливості, такі як розширення підтримки інших видів транспорту або інтеграція з іншими інформаційними системами. Крім того, система спроектована таким чином, щоб легко обробляти великі обсяги даних, забезпечуючи високу продуктивність при роботі з численними запитами та звітами. Вона також підтримує масштабованість і може бути адаптована до зростання обсягів даних, що особливо важливо для логістичних компаній, які працюють з

великою кількістю вантажів і транспортних одиниць. Підсумовуючи, було реалізовано функціональний веб-застосунок із такими можливостями:

- регулярне автоматизоване отримання актуальних логістичних даних із зовнішнього API;
- збереження, обробка та оновлення отриманої інформації у реляційній базі даних;
- відображення структурованих даних у веб-інтерфейсі з функціями сортування, фільтрації та експорту у форматі Excel;
- хмарне розгортання на платформі Azure, що забезпечує стабільний доступ до системи з будь-якого пристрою та дозволяє масштабувати її відповідно до потреб підприємства.

Щодо практичного значення, то розроблене програмне рішення демонструє приклад побудови сучасної прикладної інформаційної управляючої системи з елементами автоматизації та хмарної доступності. Його можна адаптувати під специфіку інших логістичних сервісів або інтегрувати в існуючу корпоративну IT-інфраструктуру.

РОЗДІЛ 3. РОЗРОБЛЕННЯ ПРОЄКТНИХ РІШЕНЬ ТА ЇХ РЕАЛІЗАЦІЯ

3.1. Проєктування бази даних

3.1.1. Логічна модель БД

Система як така не пропонує рішень фахівцям які працюють з нею, її мета надання показової інформації для підтримки їх прийняття. На даний момент основним алгоритмом для роботи системи є алгоритм В-дерева для пошуку записів та їх відтворення з бази даних. Його складність $O(\log N)$, де N – кількість записів, з яких здійснюється вибір. Для оптимізації та прискорення даного алгоритму використовуються індекси. Серед індексів розрізняються кластерні (фізично переставляють дані в таблиці згідно з порядком індексу) та некластерні (зберігаються окремо від основної таблиці, але містять вказівники на відповідні рядки). Обидва ці типи активно використовуються в розробленому рішенні та мають ключову роль в ефективній обробці великих обсягів даних.

Основними сутностями якими оперує система є вагони, документи, потяги та ін. Вони є наслідком нормалізації сховища та оптимізації подальшого використання даних. Перелік моделей з описом наведений у таблиці 3.1.1.1.

Назва моделі	Опис
ApiLog	Модель для збереження та відтворення на стороні інтерфейсу повідомлень при роботі з API залізниці
Car	Модель що зберігає дані про вагони, що належать компанії
Cargo	Вантаж, який перевозять вагоні
DataSession	Модель, функціонально необхідна для роботи з API. Зберігає інформацію про останній забір даних.
Document	Модель яка описує документи, пов'язані з тим чи іншим вагоном
Message	Повідомлення, коментарі до записів вагонів
Operation	Операції здійснені над вагонами
Participant	Сторони договору про перевезення
Train	Поїзди, до яких прикріплені ті чи інші вагони
User	Користувачі зареєстровані у системі

Таблиця 3.1.1.1. – Моделі розроблюваної системи

У процесі розробки програмного забезпечення активно використовуються об'єкти передачі даних, або DTO (Data Transfer Object), які є важливим інструментом для оптимізації роботи з інформацією. DTO створені для представлення структурованих даних у вигляді об'єктів, що полегшує їх передачу між різними компонентами системи.

DTO застосовуються в двох основних сценаріях: обробка початкових, "сирих" даних та переміщення даних всередині системи. Коли інформація надходить до системи з зовнішніх джерел, таких як API, бази даних або інші сервіси, вона може мати неоднорідний, надлишковий чи нефіксований формат. DTO дозволяють адаптувати ці дані до структури, зручною для обробки у додатку, фільтруючи зайві поля, нормалізуючи значення і перетворюючи їх до єдиного вигляду. Це забезпечує підвищену безпеку та стабільність у роботі бізнес-логіки. У бізнес-логіці DTO використовуються для передачі інформації між різними модулями та компонентами. Вони дозволяють мінімізувати залежність між компонентами, зберігаючи чіткі контракти даних. Завдяки цьому будь-які зміни в одному модулі не вимагають суттєвих коригувань у пов'язаних частинах системи, що сприяє модульності та масштабованості розробки.

Структура DTO зазвичай включає тільки необхідні поля для певного контексту використання, що дозволяє зменшити обсяг переданих даних і оптимізувати використання ресурсів системи. Наприклад, DTO для отримання інформації про користувача може містити лише ідентифікатор, ім'я та роль, тоді як DTO для авторизації — додатково включати електронну пошту та токен доступу. Цей підхід сприяє ефективному функціонуванню програмного забезпечення, оскільки зменшує навантаження на мережу за рахунок передачі мінімально необхідного обсягу даних, підвищує продуктивність завдяки зменшенню обсягу обчислень і спрощенню обробки та забезпечує зрозумілу та логічно організовану структуру даних, яка спрощує налагодження та підтримку системи. Таким чином, використання DTO є не тільки технічним прийомом, але й одним із ключових аспектів забезпечення надійності, гнучкості та продуктивності програмного

забезпечення. Дані моделі та структура забезпечують оптимальне використання ресурсів та ефективне функціонування розробки.

Система сховища є критично важливою компонентою розробки, оскільки неправильна її імплементація унеможливить саму роботу з системою – дані провантажуються довго, зберігаються неналежним чином, некоректно; подібні події повністю нівелюють успішну роботу інших модулів. У рамках цієї системи використовуються реляційні бази даних для зберігання великих обсягів інформації, що надходить з зовнішніх джерел через API залізничних компаній. Процес проектування та реалізації бази даних має на меті забезпечити ефективне зберігання, обробку та доступ до даних, що дозволяє підтримувати високий рівень продуктивності та гнучкості при масштабуванні системи.

З метою оптимізації структури бази даних і мінімізації дублювання інформації було проведено процес нормалізації. На першому етапі дані були логічно розподілені за категоріями та структуровані у вигляді дев'яти окремих таблиць. Такий підхід дозволив суттєво зменшити кількість повторюваних записів, забезпечивши одночасно гнучкість і масштабованість моделі даних. Однією з ключових вимог замовника було збереження кожного рядка вхідних даних у разі наявності хоча б одного відмінного поля, та, відповідно, відмова від збереження запису, якщо всі його поля ідентичні вже наявному. Це ускладнювало логіку перевірки на унікальність і потребувало ретельного підходу до структурування.

Під час аналізу було виявлено, що ефективне повторне використання даних можливе завдяки переходу до зіркоподібної (star schema) архітектури, яка дозволяє зосередити статичні дані в окремих таблицях-довідниках. Проте надмірна нормалізація могла б негативно позначитись на швидкодії операцій вибірки та модифікації, особливо в умовах великого обсягу даних. Тому було прийнято збалансоване рішення щодо глибини нормалізації. Результатом стала структура, яка дозволила скоротити ймовірність дублювання щонайменше удвічі порівняно з початковим варіантом. Для автоматизації процесу перевірки унікальності записів було впроваджено додаткові поля хеш-кодів, сформованих на основі вмісту всіх інших полів рядка. Цей підхід значно прискорює і спрощує процедуру виявлення

дублікатів, водночас знижуючи навантаження на систему при обробці великих обсягів даних.

3.1.2. Фізична модель БД

У межах розробки веб-додатку було здійснено налаштування контексту даних, що включало визначення сутностей, їхніх взаємозв'язків, а також конфігурацію параметрів доступу до даних. З метою підвищення ефективності запитів до бази даних було проведено індексацію полів, які, згідно з аналітичними припущеннями, найчастіше використовуватимуться під час пошуку. Це дозволяє значно зменшити час, необхідний для пошуку, сортування та фільтрації, що є критичним для системи, яка обробляє великий обсяг даних в реальному часі. Індеси допомагають скоротити час виконання складних запитів, що важливо для забезпечення високої продуктивності при взаємодії з користувачем або в процесі генерації звітів.

Під час роботи з часовими даними виникли труднощі, зумовлені специфікою обробки форматів дати та часу у PostgreSQL. Зокрема, застосування стандартного формату UTC призводило до потенційних непорозумінь при інтерпретації часових позначок у межах локальної інфраструктури. Було прийнято рішення перейти на локальний часовий пояс, що виявилось прийнятним компромісом, оскільки вся логістична діяльність здійснюється в межах однієї країни, без перетину часових зон.

Для забезпечення доступу застосовується Entity Framework Core (EF Core), потужний інструмент для роботи з реляційними базами даних в екосистемі .NET. EF Core дозволяє працювати з базою даних на більш високому рівні абстракції, без необхідності вручну писати SQL-запити для виконання стандартних операцій, таких як вставка, оновлення, видалення чи вибірка даних [35]. Для зв'язків між таблицями використовуються зовнішні ключі, які дозволяють зберігати цілісність даних при виконанні операцій вставки, оновлення або видалення записів. Зв'язки

між таблицями відображають результати нормалізації вищезгаданих моделей, які в свою чергу є наборами інформації про ті чи інші сутності. Структуризацію взаємозв'язків між моделями реалізовано вручну з використанням Fluent API, що надається платформою Entity Framework Core. Такий підхід забезпечує повний контроль над конфігурацією моделей, дозволяє чітко визначити правила зв'язку між сутностями і зменшує ймовірність неоднозначностей [36]. Усі визначення були централізовано зосереджені в одному місці, що сприяє кращій підтримованості й масштабованості проєкту.

Для забезпечення тестування та перевірки коректності реалізованої логіки було створено локальну копію бази даних за допомогою механізму міграцій. Цей процес надалі повторювався і для хмарної інфраструктури, що дозволило проводити тестування окремих функціональних блоків (зокрема, механізмів десеріалізації вхідних даних) автономно, без необхідності постійних звернень до віддаленого сервера. Такий підхід значно зменшив ризики втрати даних та сприяв ефективному виявленню і усуненню помилок на ранніх етапах розробки.

Важливою характеристикою бази даних є її масштабованість. Оскільки система повинна обробляти значні обсяги даних, зокрема у разі збільшення кількості вантажів та вагонів, передбачається використання відповідних технік для масштабування. Для цього можуть бути застосовані як вертикальне, так і горизонтальне масштабування. Вертикальне масштабування передбачає збільшення потужностей окремих серверів для обробки більшого обсягу даних, тоді як горизонтальне масштабування передбачає додавання нових серверів до існуючої інфраструктури для балансування навантаження та підвищення загальної потужності системи.

3.1.3. Контрольний приклад та обсяг БД

Створена база даних підтримує збереження як поточних даних, так і історичних відомостей про рух вантажів і транспорту. Це дозволяє здійснювати аналіз ефективності логістичних операцій, виводити статистику, порівнювати

показники роботи компанії в різні періоди часу та прогнозувати майбутні тренди. Історичні дані зберігаються окремо від поточних, що дозволяє оптимізувати роботу системи, не навантажуючи її зайвими записами.

При використанні тестових даних був створений простий застосунок, Web API на платформі ASP.NET який комунікує схожим чином з реальним сервісом, тому не потребує особливих змін в основній програмі. Власне, всі дані наведені в роботі є тестовими, згенерованими випадковим чином. При тестуванні обидва проекти запускались локально, комунікуючи через локальні порти, передаючи дані через HTTP протокол.

Оцінку обсягів реальних даних провести важко, але в межах доби кількість записів дорівнювала близько двом тисячам, в залежності від обставин. Тобто, протягом місяця обсяги складатимуть близько шістдесяти тисяч записів.

Для демонстрації структури таблиць та відносин між ними, за допомогою середовища pgAdmin 4, було створено діаграму бази даних, наведену у додатку В. Як можна бачити, основний акцент робиться на таблиці з вагонами, що пояснюється запитом компанії-перевізника. Це здебільшого нагадує зіркову схему, яка поширена в роботі з Power BI та великими обсягами даних. Звісно, цілком можливо нормалізувати її в ще більшій мірі, втім, з точки зору роботи з реляційною базою даних це не буде так ефективно, адже тоді порушиться цілісність представлення даних на стороні користувача і суттєво збільшиться час виконання запитів, що буде перевантажувати та гальмувати систему.

Додатково, з метою забезпечення високої доступності та відмовостійкості бази даних також використовуються механізми резервного копіювання та відновлення даних, що дозволяють зберігати цілісність та доступність інформації навіть у разі збоїв у системі або аварійних ситуацій.

3.3. Програмне забезпечення

3.3.1. Проєктування ПЗ

На початковому етапі розробки особливу увагу було приділено визначенню обсягу та типів даних, необхідних для повноцінного функціонування програмного забезпечення. Цей процес передбачав не лише вивчення технічної документації, а й безпосередню комунікацію з працівниками, які працюють з логістичними даними на практиці — зокрема, з диспетчерами, що відповідають за моніторинг перевезень. Завдяки таким розмовам вдалося з'ясувати, які саме дані є критично важливими в їхній щоденній роботі, у якому вигляді вони мають бути представлені, а також якого роду фільтрування, сортування чи обробка застосовується до цих даних на практиці.

Паралельно з цим проводився детальний аналіз документації до API, з якого надходитимуть зовнішні дані. Було визначено технічні особливості роботи з цим API, включаючи формат переданих даних (зокрема, необхідність десеріалізації), специфіку формування запитів, обмеження за частотою звернень, типи відповідей, обробку помилок тощо. На основі отриманої інформації були сформульовані технічні вимоги до архітектури взаємодії застосунку з API, а також виписані необхідні методи, що забезпечують коректну інтеграцію.

Також було підібрано відповідні шаблони проєктування (патерни), що відповідають логіці поведінки API і дозволяють реалізувати стабільну, масштабовану та легко підтримувану взаємодію. Зокрема, визначено стратегії обробки відповідей, механізми повторних запитів у разі збоїв, а також реалізовано гнучку систему логування для подальшого моніторингу працездатності інтеграційного модуля.

Програмне забезпечення було спроектовано як комплексна система, що включає в себе веб-застосунок та базу даних, розгорнуті на серверному обладнанні компанії. Такий підхід забезпечує централізоване управління всіма компонентами, полегшує адміністрування та підвищує рівень безпеки даних.

3.3.2. Вибір архітектури ПЗ

Веб-застосунок структуровано на основі набору модулів, кожен із яких відповідає за виконання певної функціональності. Ці модулі були умовно поділені на кілька категорій залежно від їхніх завдань і призначення. У таблиці 3.3.2.1 наведено розподіл модулів за категоріями з детальним описом їх функцій. Наприклад, до першої категорії входять модулі-контролери, які відповідають за бізнес-логіку та перетворення даних перед демонстрацією їх користувачам. Такий підхід забезпечує зручність розробки та підтримки системи, дозволяючи масштабувати функціональність або змінювати окремі компоненти без значного впливу на інші частини програмного забезпечення.

Назва	Опис
Контролери	Складова MVC, призначена для функцій пов'язаних з веб-сторінками застосунку
Дані	Безпосередня робота з базою даних; там встановлюються зв'язки між моделями за допомогою Fluent API, додається індексація.
Репозиторії	Реалізація відповідного паттерну, що дозволяє інкапсулювати роботу з даними та отримати їх для подальшого використання на стороні інтерфейсу
Сервіси	Набір сервісів для реалізації бізнес-логіки системи. Містить у собі конвертери (сервіси для перетворення даних), фонові сервіси (для періодичного звертання до API), сервіси для безпосередньої роботи з API (звертання за допомогою HTTP протоколів), мануальна версія для роботи через консоль, сервіси для збереження даних до БД
Моделі, Mapper, DTOs (data transfer objects)	Моделі та профілі для роботи з даними; включають в себе моделі з БД, моделі для роботи всередині програми (для покращеної та більш ефективної комунікації між сервісами) та профілі для більш ефективної конвертації
Views, ViewModels	Моделі та сторінки, які використовуються користувацьким інтерфейсом

Табл. 3.3.2.1. Опис модулів системи

Особливо варто відзначити різні рівні розробки: наприклад, сервіси відповідають за логіку отримання, перетворення та збереження даних, а контролери та репозиторії за їх витягнення та показ користувачу. В програмному коді вони знаходяться в різних папках для зручного використання та модифікації. Файлова структура проекту наведена на рисунку 3.3.2.1.

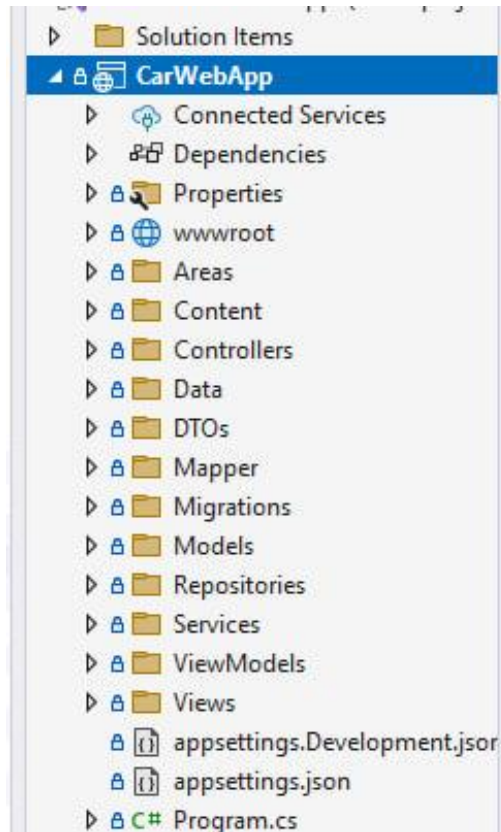


Рисунок 3.3.2.1. – Структура проекту

Всі класи із категорії сервісів попередньо реєструються в класі Program відповідно до їх життєвого циклу. Це дозволяє в майбутньому ін'єкцією залежностей використовувати їх при взаємодії одне з одним. Також для опису функціонування системи було створено діаграму UML Sequence, яка зображує запит даних, їх отримання, трансформацій та відображення кінцевому користувачу. Діаграму наведено на рисунку 3.3.2.2.

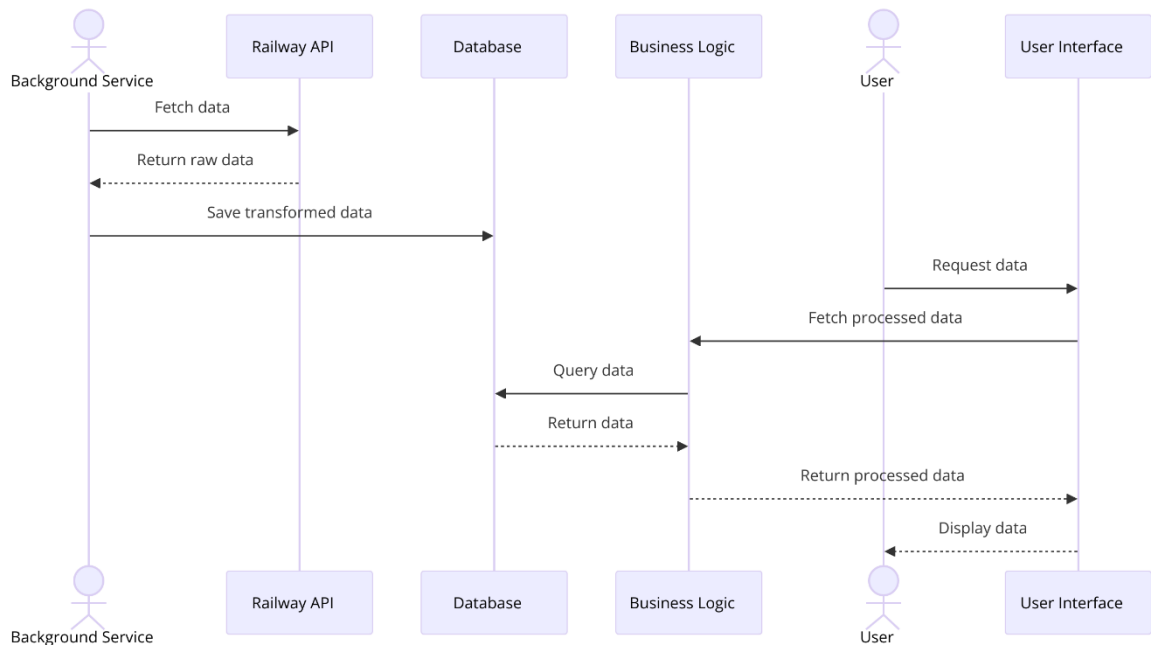


Рисунок 3.3.2.2. – Діаграма послідовності

Під час проектування програмного забезпечення було обрано підхід Clean Architecture [37], що дозволяє забезпечити чітке розділення відповідальностей між компонентами системи, спрощує її масштабування та підтримку, а також підвищує тестованість коду. Цей архітектурний підхід передбачає виділення окремих рівнів — доменної логіки, інтерфейсів взаємодії, інфраструктури та зовнішніх джерел даних — із чітко визначеними залежностями лише в одному напрямку: від зовнішнього до внутрішнього. Така структура дозволяє уникнути надмірної зв'язаності між модулями та підвищує гнучкість системи.

У реалізації взаємодії між компонентами було застосовано механізм Dependency Injection, що є вбудованим у платформу .NET. Його використання дозволило централізовано керувати залежностями між класами та сервісами, спрощуючи конфігурацію і тестування окремих модулів. Крім того, під час розробки враховано принципи SOLID [38], зокрема принципи єдиної відповідальності та відкритості/закритості, що стало основою для створення логічно ізольованих сервісів, кожен з яких виконує чітко визначену функцію. Для ефективного перетворення об'єктів між різними шарами системи, зокрема між моделями домену та передаваними структурами, використовувалась технологія AutoMapper — популярний інструмент для автоматизованого мапінгу об'єктів. З

метою організації внутрішнього обміну даними було побудовано узгоджений ланцюг сервісів, де кожен модуль взаємодіє з іншими через чітко визначені інтерфейси, що дозволяє зберігати модульність та масштабованість проєкту. Такий підхід гарантує гнучкість у розширенні функціоналу в майбутньому та полегшує адаптацію системи до нових бізнес-вимог.

3.3.3. Інструментальні засоби

На етапі постановки та реалізації завдання особливу увагу було приділено уточненню функціональних вимог та узгодженню потреб кінцевих користувачів. Для цього проводилися цільові комунікації з працівниками підприємства, які безпосередньо працюватимуть із системою. Було детально проаналізовано технічне завдання, а також вивчено документацію до API, з якого передбачається отримання зовнішніх даних для подальшого опрацювання.

На основі зібраних вимог було визначено технологічний стек, оптимальний для реалізації поставлених цілей. Для розробки було обрано фреймворк ASP.NET з використанням архітектурного шаблону MVC, що забезпечує логічне розділення відповідальностей у коді та полегшує його підтримку. Для роботи з базою даних використано ORM-технологію Entity Framework Core, яка дозволяє ефективно взаємодіяти з реляційною СУБД PostgreSQL. Остання була обрана з огляду на її продуктивність, розвинуті можливості індексації, масштабованість, а також відкриту ліцензію, що виключає необхідність додаткових витрат на програмне забезпечення.

Окрім основних технологій, було також підібрано низку допоміжних бібліотек і пакетів, що сприяють підвищенню швидкодії системи та дозволяють реалізовувати сучасні підходи до розробки програмного забезпечення — зокрема, полегшену обробку запитів, логування, кешування тощо. Питання бюджету та ресурсного забезпечення проєкту обговорювалося на початковому етапі розробки,

що дало змогу раціонально підійти до вибору інструментів, не жертвуючи якістю кінцевого продукту.

Для створення зручного та інтуїтивно зрозумілого інтерфейсу користувача було розроблено кілька сторінок за допомогою Razor — потужного інструмента для створення динамічного HTML-коду в середовищі .NET. Razor дозволяє ефективно поєднувати серверний і клієнтський код, що сприяє спрощенню процесу розробки та забезпеченню високої продуктивності веб-додатка. Як зазначалося в попередніх розділах, Razor було обрано завдяки його здатності швидко інтегруватися з іншими технологіями, такими як ASP.NET MVC.

Для забезпечення безпеки системи та реалізації процесу авторизації користувачів було використано вбудований функціонал ASP Identity. Цей механізм дозволяє керувати обліковими записами користувачів, підтримувати реєстрацію, вхід, вихід, а також обробляти ролі та права доступу. ASP Identity забезпечує гнучку архітектуру, яка дозволяє адаптувати механізми авторизації до конкретних потреб додатка, зберігаючи при цьому високий рівень безпеки завдяки вбудованим функціям хешування паролів та використанню токенів доступу.

На рисунку 3.3.3.1. наведено приклад реалізації сторінки авторизації, яка демонструє, як Razor використовується для створення HTML-форми, а ASP Identity — для обробки запитів користувачів. Сторінка містить поля для введення логіна та пароля, а також функції обробки помилок та забезпечення коректного переходу до захищених розділів системи.

Система моніторингу вагонів Домашня сторінка Форма 1892 Форма 8858 Журнал API Зареєструватись Увійти

Log in

Email
Password
☐ Remember me?

Log in

[Forgot your password?](#)

[Register as a new user](#)

Рисунок 3.3.3.1. – Приклад сторінки авторизації користувача

3.3.4. Тестування ПЗ

У процесі розробки активно використовувався режим налагодження (debug) середовища розробки Visual Studio. Основними інструментами виступали брейкпоінти, які дозволяли зупиняти виконання програми в ключових точках, а також вбудовані засоби відстеження значень змінних у реальному часі. Це забезпечувало можливість детального аналізу логіки виконання коду, своєчасного виявлення помилок і корекції поведінки програми на ранніх етапах.

Паралельно здійснювався моніторинг використання системних ресурсів як на рівні апаратного забезпечення (через стандартні системні інструменти операційної системи), так і засобами самого середовища розробки. Такий підхід дозволив виявити потенційні прогалини у продуктивності та уникнути надмірного споживання оперативної пам'яті чи процесорного часу, що є критичним для тривалого функціонування веб-додатку в умовах реального навантаження.

Була впроваджена система логування, яка охоплює як фронтенд, так і бекенд-частину додатку. Повідомлення про помилки, статуси запитів та відповіді API спрямовувалися в інтерфейс користувача згідно з технічними вимогами — це дозволяло оперативно отримувати зворотний зв'язок щодо дій системи. Окремо реалізовано виведення детальних діагностичних логів у консоль — цей канал використовувався виключно в режимі розробки й тестування, з метою відслідковування поведінки системи під час виконання (рисунок 3.3.4.1).

```

-----
info: CarWebApp.Services.ExternalApiService[0]
      Response Status Code: OK
info: CarWebApp.Services.ExternalApiService[0]
      Response Header: Date - Thu, 08 May 2025 00:41:48 GMT
info: CarWebApp.Services.ExternalApiService[0]
      Response Header: Server - Kestrel
info: CarWebApp.Services.ExternalApiService[0]
      Response Header: Transfer-Encoding - chunked
info: CarWebApp.Services.ExternalApiService[0]
      Response Content: Content too long (success)
-----
info: CarWebApp.Services.ExternalApiService[0]
      GET: Data added to result
info: System.Net.Http.HttpClient.IExternalApiService.LogicalHandler[100]
      Start processing HTTP request GET https://localhost:7166/api/DataGen
info: System.Net.Http.HttpClient.IExternalApiService.ClientHandler[100]
      Sending HTTP request GET https://localhost:7166/api/DataGen
info: System.Net.Http.HttpClient.IExternalApiService.ClientHandler[101]
      Received HTTP response headers after 488.5217ms - 200
info: System.Net.Http.HttpClient.IExternalApiService.LogicalHandler[101]
      End processing HTTP request after 509.033ms - 200
-----

```

Рисунок 3.3.4.1. – Приклад логів програми

Для тестування стабільності роботи додатку програма запускала на локальному комп'ютері в умовах тривалого сеансу (в середньому по 8 годин безперервної роботи). Це дозволило перевірити систему на предмет витоків пам'яті, перевантаження ресурсів і непередбачуваних виняткових ситуацій. Протягом тестування було виявлено ряд проблем, пов'язаних з нестабільною поведінкою стороннього API, до якого здійснювались запити. Через недостатню деталізацію документації API, значну частину адаптації логіки доводилося виконувати вручну, шляхом емпіричного тестування та поступової модифікації коду згідно з фактичною поведінкою зовнішньої системи. Це потребувало високого рівня гнучкості та багаторазових повторень тестових сценаріїв, аби забезпечити коректну інтеграцію.

На завершальному етапі реалізації відбувається безпосереднє введення розробленого веб-додатку в експлуатацію. Цей процес є поступовим і включає кілька контрольованих фаз, спрямованих на забезпечення максимальної стабільності та надійності системи в умовах реального користування.

Первинне введення системи в дію здійснюється розробником у режимі технічного моніторингу, що передбачає постійне відстеження логів, поведінкових відхилень, а також навантаження на серверну частину. На цьому етапі особливу

увагу приділяють виявленню залишкових помилок, тестуванню граничних сценаріїв роботи та перевірці відповідності системи очікуваному функціоналу.

Наступною фазою є обмежене впровадження серед цільової групи кінцевих користувачів — обраного кола працівників, які взаємодіють із системою в умовах часткового навантаження. Такий підхід дозволяє зібрати первинний зворотний зв'язок, виявити потенційні проблеми в інтерфейсі або логіці роботи додатку, що могли бути неочевидними під час внутрішнього тестування. У цей період продовжується активний моніторинг технічного стану системи з боку розробника.

Лише після успішного проходження вищезгаданих етапів та підтвердження стабільної роботи веб-додатку, система вводиться в повноцінну експлуатацію. Це означає, що доступ до функціоналу отримують усі відповідні працівники, додаток переходить у стандартний робочий режим без обмежень, а процес технічного супроводу переходить до стадії обслуговування та підтримки.

3.4. Технічне забезпечення

Для успішної реалізації інформаційної системи моніторингу та управління логістикою вагонів, необхідно не лише правильно обрати програмне забезпечення, але й забезпечити відповідне технічне оснащення, яке дозволить ефективно інтегрувати систему в інфраструктуру логістичної компанії. Технічне забезпечення повинно відповідати вимогам до продуктивності, надійності, безпеки та масштабованості, що дозволяє забезпечити безперебійну роботу системи в умовах великих обсягів даних та високих навантажень.

Щоб описати приблизні групи потрібного устаткування було створено діаграму розгортання UML. Вона зображає як різні категорії технічного обладнання взаємодіють між собою та що є кінцевим пристроєм даної мережі (рисунок 3.4.1.).

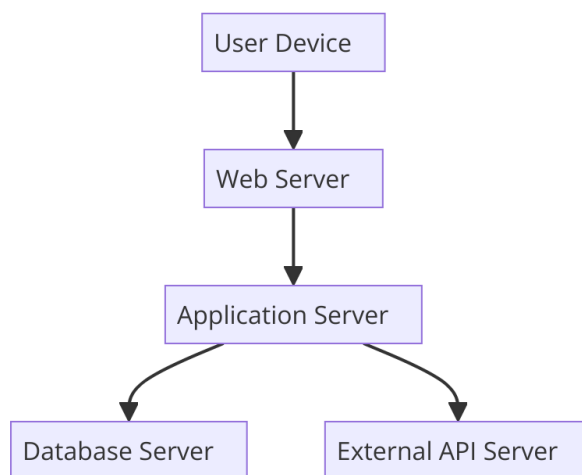


Рисунок 3.4.1. – Діаграма розгортання

Перш за все, для роботи з великими обсягами даних, що генеруються під час моніторингу вагонів, необхідне потужне серверне обладнання. Використання високопродуктивних серверів з достатньою кількістю процесорних ядер та великим обсягом оперативної пам'яті дозволяє забезпечити швидку обробку запитів, обчислень та зберігання даних. Важливим параметром для серверів є наявність швидких дискових підсистем, що забезпечують високу швидкість читання та запису даних, що критично для систем, що працюють з великими базами даних.

Також для забезпечення безперебійної роботи необхідно впровадити технології резервного копіювання та відновлення даних. Це може бути досягнуто через використання кластеризації серверів, реплікації баз даних та використання систем зберігання даних, що підтримують автоматичне резервування. Важливою складовою є також наявність систем моніторингу, які дозволяють своєчасно виявляти та усувати неполадки в роботі сервісів та інфраструктури.

Для реалізації інтерфейсу з користувачем, а також для забезпечення доступу до системи з різних пристроїв, необхідно передбачити використання мережевого обладнання, яке забезпечить стабільне з'єднання з інтернетом. Використання маршрутизаторів та комутаторів, що підтримують високу пропускну здатність та можливість сегментації мережі для безпеки, забезпечить необхідний рівень ефективності та безпеки при роботі з системою.

Ще одним важливим елементом технічного забезпечення є робочі станції співробітників логістичної компанії. Це можуть бути як персональні комп'ютери, так і мобільні пристрої, які повинні бути оснащені сучасними веб-браузерами, що підтримують всі функціональні можливості системи. Для забезпечення безпеки роботи та конфіденційності даних на кожному робочому місці необхідно впровадити сучасні системи антивірусного захисту та криптографії для захисту переданих даних.

Для здійснення технічної підтримки та обслуговування системи на стороні логістичної компанії потрібно забезпечити відповідну інфраструктуру для моніторингу її стану, оновлення та виправлення помилок. Це включає в себе сервери для зберігання логів, систему для централізованого управління оновленнями та інструменти для виявлення та усунення можливих проблем в роботі програмного забезпечення.

І наостанок, в системі повинні бути передбачені засоби для налаштування та управління безпекою на фізичному та програмному рівнях. Їй потрібно мати надійні механізми для аутентифікації та авторизації користувачів, а також для контролю доступу до чутливих даних. Використання сучасних криптографічних методів для захисту даних є необхідною умовою для забезпечення конфіденційності та цілісності інформації, що передається між різними частинами системи.

3.5. Реалізація системи

Веб-застосунок виконує роль основного інтерфейсу для взаємодії користувачів із системою. Він був реалізований із використанням сучасних веб-технологій, що дозволяє забезпечити високу швидкодію, адаптивність до різних типів пристроїв і зручність у користуванні. Основні завдання застосунку включають обробку запитів від користувачів, взаємодію з базою даних, виконання бізнес-логіки та відображення результатів роботи системи.

База даних, розгорнута на тому ж сервері, є централізованим сховищем усієї інформації, необхідної для роботи програмного забезпечення. Вона зберігає дані про користувачів, логіку авторизації та аутентифікації, результати обробки API, історичні дані для аналізу та багато іншого. Використання локальної бази даних дозволяє забезпечити мінімальні затримки в обробці даних та швидкий доступ до них.

Щоб забезпечити кінцевим користувачам можливість зручно відстежувати останні оновлення даних і отримувати інформацію про взаємодію системи зі стороннім залізничним API, було розроблено спеціалізовану сторінку, яка виконує функцію журналу API. Ця сторінка надає доступ до інформації про всі події, що виникають у процесі роботи з сервісом ExternalApiService, включаючи успішні виклики, помилки, затримки та інші важливі повідомлення.

Основу цієї функціональності становить модель ApiLog, яка була створена для зберігання всіх автоматично зареєстрованих повідомлень від ExternalApiService. Кожен запис у цьому журналі містить ключові атрибути, такі як час події, тип повідомлення (інформаційне, попередження, помилка тощо), опис події та додаткову інформацію, що допомагає деталізувати обставини виникнення. Такий підхід дозволяє не лише оперативно відслідковувати стан роботи API, але й проводити аналіз історичних даних для виявлення тенденцій, потенційних проблем або вузьких місць у роботі системи.

Сторінка журналу API була інтегрована в загальний інтерфейс користувача, забезпечуючи зрозумілий і структурований вигляд записів. Користувач може сортувати або фільтрувати дані за різними параметрами, такими як дата, тип повідомлення чи статус. Це дозволяє швидко знаходити потрібну інформацію, що особливо важливо у випадках виникнення проблем з оновленням даних або роботою стороннього API.

На рисунку 3.5.1 представлено приклад реалізації цієї сторінки, яка демонструє, як виглядають повідомлення від системи. Сторінка наочно показує структуру та зміст журналу, що є ключовим інструментом для контролю за

роботою API та отримання зворотного зв'язку для подальшого вдосконалення сервісу.

Система моніторингу вагонів		Домашня сторінка	Форма 1892	Форма 8858	Журнал API	Вітаємо, admin@admin.com! Вийти із системи
Журнал API						
Час	Повідомлення					
03.10.2024 16:40:48	Successfully fetched data for 1892					
03.10.2024 16:40:32	Authorization for the 1892 data request failed: AUTH: refresh token expired					
03.10.2024 15:35:04	Successfully fetched data for 1892					
03.10.2024 15:34:59	Authorization for the 1892 data request failed: AUTH: refresh token expired					
02.10.2024 12:46:25	Successfully fetched data for 8858					
02.10.2024 12:45:54	Successfully fetched data for 8858					
02.10.2024 12:44:59	Authorization for the 8858 data request failed: Error making the authorization request: Response status code does not indicate success: 500 (Internal Server Error).					
02.10.2024 12:44:58	Successfully fetched data for 8858					
02.10.2024 12:44:04	Successfully fetched data for 8858					
02.10.2024 12:43:11	Successfully fetched data for 8858					
02.10.2024 12:42:17	Successfully fetched data for 8858					
02.10.2024 12:41:25	Authorization for the 8858 data request failed: Error making the authorization request: Response status code does not indicate success: 500 (Internal Server Error).					
02.10.2024 12:41:24	Successfully fetched data for 8858					

Рисунок 3.5.1. – Сторінка для перегляду повідомлень при роботі з API

Одним із ключових викликів під час реалізації проєкту стала оптимізація обробки даних, адже неефективне опрацювання суттєво впливає на загальну продуктивність системи. Значна частина часу була присвячена вдосконаленню механізмів взаємодії з базою даних, зокрема оптимізації трекінгу сутностей, що є специфічною особливістю Entity Framework Core. Вимкнення автоматичного трекінгу у випадках, де це не було необхідно, дозволило зменшити навантаження на контекст даних і підвищити загальну швидкодію під час обробки великих масивів інформації.

Окрему складність становила обробка вхідних JSON-даних, оскільки структура та формат, які надавав сервер, не відповідали стандартним очікуванням десеріалізаторів. Для вирішення цієї проблеми було розроблено низку кастомних конвертерів, які дозволили коректно обробляти нетипові формати, забезпечуючи стабільність системи під час імпорту даних. Також була впроваджена потокова

обробка JSON-файлів, що мінімізувало затримки при роботі з великими обсягами вхідної інформації.

З метою забезпечення зручності для кінцевих користувачів було створено та заповнено словники відповідності кодів і їхніх текстових назв. Ці словники формувалися на основі відкритої технічної документації, яка містила перелік кодів об'єктів (наприклад, станцій) та їхніх найменувань. Отримані дані були імпортовані в систему та реалізовані у вигляді функціональності, що дозволяє конвертувати коди в текстові позначення і навпаки. В інтерфейсі користувача це виглядає як подання одночасно коду (наприклад, 394830) та назви (наприклад, ЧЕРКАСИ), що значно полегшує ідентифікацію об'єктів. Для підвищення продуктивності словники кешуються на стороні клієнта, що дозволяє здійснювати миттєві перетворення без додаткових запитів до сервера.

Щоб забезпечити стабільну роботу системи при роботі з великими наборами даних, було впроваджено механізм розбиття на сторінки (pagination), а також фільтрацію на рівні представлення даних (view models). Приклади зображені на рисунках 3.5.2 та 3.5.3. Це означає, що під час формування відповідей з бази даних до клієнта передаються лише ті поля, які є необхідними для поточної задачі працівника. Таким чином, створюється новий об'єкт з мінімізованим набором полів, що оптимізує обмін даними й запобігає перевантаженню системи.

131014	БАТА ЦЕЛЮЛОЗНА	№ 11	01.06.2020 0:00:00	90	37
391233	НАСОСИ ВЕЛОСИПЕДНІ	НОВОКРАСНЕ	01.06.2020 0:00:00	57	3

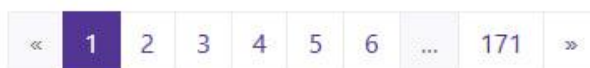


Рисунок 3.5.2. – Підвантаження даних по сторінках

Номер вагона	Операція			
	Код операції	Код операції (поїзд)	Дата та час здійснення операції	Станція здійснення операції
294892				
708523	117	105	09.05.2020 0:00:00	МИКОЛАЇВ-ДНІСТРОВСЬКИЙ
151965	15	230	09.05.2020 0:00:00	ЖАРКИ
716224	605	ВИВЛС	10.05.2020 0:00:00	МОЖАРОВЕ

Рисунок 3.5.3. – Поле для фільтрування даних

Значні зусилля були спрямовані на тестування та оптимізацію SQL-запитів, аби досягти максимальної продуктивності під час роботи з реальними сценаріями. В ході тестування із застосуванням програмного таймера було встановлено, що в типовому режимі використання (перегляд, фільтрація, оновлення даних) затримки у взаємодії з системою майже не відчуються. Єдине виключення — масове вивантаження даних за тривалий період (наприклад, експорт усіх записів за останній рік), що включає сотні тисяч рядків. У таких випадках час очікування є об’єктивно вищим, однак залишається в межах прийнятного для задач даного типу.

Для реалізації інтерфейсу користувача було використано безкоштовну тему на базі фреймворку Bootstrap [39], яка вирізняється мінімалістичним дизайном. Такий підхід забезпечив візуальну легкість, швидке завантаження сторінок і комфорт при тривалому перегляді даних. При створенні представлень використовувалася мова Razor (.cshtml), що дозволяє ефективно поєднувати HTML-розмітку з серверною логікою на мові С#. Це надало гнучкість у відображенні динамічних даних та адаптації інтерфейсу під потреби користувачів.

Для подання даних була реалізована таблиця, яка стилістично та структурно наближена до тих інтерфейсів, з якими користувачі вже мали попередній досвід роботи (рисунок 3.5.4).

Система моніторингу вагонів

Останні дані

Історія записів

Журнал API

Вийти із системи

Відвантажити в ексел

Номер вагона	Операція				Потяг		Код ЄДРПОУ підприємства, яке здійснює оперативне управління вагонами	Вантаж					Сторони		Станція призначення вагона	
	Код операції	Код операції (потяг)	Дата та час здійснення операції	Станція здійснення операції	Станція формування потяга	Станція призначення потяга		Ознака навантаженого вагона	Вага вантажу	Код вантажу	Назва вантажу	Станція навантаження	Дата навантаження	Код вантажоодержувача		Код вантажовідправника
Номери вагонів																Станція призначення
089714	ВІДМС	ВІВЗ	13.09.2024 00:00:00	ЯГОТИН	ІМ. БОРИСА ОЛІЙНИКА	1370 КМ	53877	НАВ	30085	712095	ВІНІЛХЛОРИД (ВІНІЛ ХЛОРИСТИЙ) ІНГІБОВАНИЙ	303 КМ	22.09.2024 00:00:00	47	57	ЛІСОВІ ПОЛІ
584304	ОБЕРТ	ПРИЧ	20.04.2023 00:00:00	УДЯНСЬКА	СТАРА ЖАДОВА	БЕЗРУЧКИ	285581	ПОР	54959	467057	АНТРАХІНОН	ІВАШКИ	25.04.2023 00:00:00	28	2	ДОБРІПІ
802780	496	103	08.08.2024 00:00:00	ПІРНИК	ДОПОМОЖНА	СОЛЯЧНА	800640	ПОР	77015	582043	ГРУШІ СУШЕНІ	299 КМ	16.08.2024 00:00:00	99	12	ЄРЕСТОВ
312295	12	ГУ78К	25.12.2023 00:00:00	КАРХІВКА	КОВІЛОВИЙ	ДІБРОВА	984212	НАВ	3523	713168	ХЛОРЦІАН СТАБІЛІЗОВАНИЙ	ГРЕЙГОВЕ	02.01.2024 00:00:00	73	66	141 КМ

Рисунок 3.5.4. – Сторінка з таблицею даних

Такий підхід дозволив зменшити час на адаптацію до нової системи та підвищити ефективність роботи персоналу. Крім базового відображення, таблиця включає поля для фільтрації, функціональні переходи за посиланнями (відповідно до побажань кінцевих користувачів), а також форму для пошуку записів на сторінці історії (рисунок 3.5.5.) — усі ці компоненти підвищують інтерактивність та зручність навігації.

Показати фільтри

Фільтри

Введіть номери вагонів:

Номери...

Приклад: 55608, 34342, 45407

Початковий час здійзн. оп. з об'єктом

дд.мм.rrrr --:--

Приклад: 1.01.2021

Кінцевий час здійзн. оп. з об'єктом

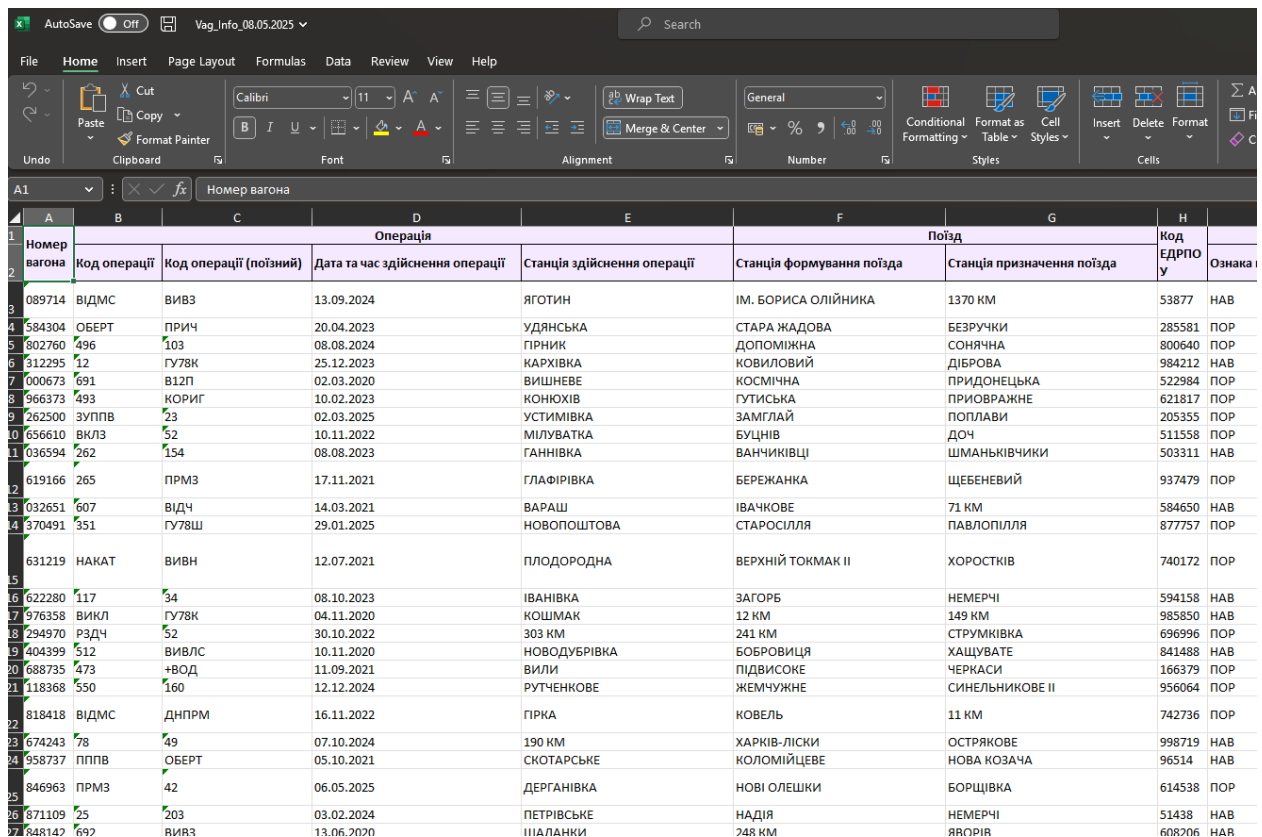
дд.мм.rrrr --:--

Приклад: 21.01.2021

Знайти

Рисунок 3.5.5. – Форма для пошуку в історії перевезень

Окрему увагу було приділено реалізації функціональності експорту даних у формат Excel, яка була серед пріоритетних вимог. Для цього було впроваджено відповідну кнопку, яка викликає серверну логіку експорту, та реалізовано кешування параметрів запиту з метою зменшення навантаження при повторному використанні однакових фільтрів. Результуючий файл Excel був стилізований таким чином, щоб максимально відповідати структурі таблиці у веб-інтерфейсі, що полегшує роботу з даними поза системою і підвищує зручність при звітуванні чи подальшій обробці (рисунок 3.5.6.).



Номер вагона	Код операції	Код операції (поїзний)	Дата та час здійснення операції	Станція здійснення операції	Станція формування поїзда	Станція призначення поїзда	Код ЕДРПОУ	Ознака
089714	ВІДМС	ВІВЗ	13.09.2024	ЯГОТИН	ІМ. БОРИСА ОЛІЙНИКА	1370 КМ	53877	НАВ
584304	ОБЕРТ	ПРИЧ	20.04.2023	УДЯНСЬКА	СТАРА ЖАДОВА	БЕЗРУЧКИ	285581	ПОР
802760	496	103	08.08.2024	ПРНИК	ДОПОМІЖНА	СОНЯЧНА	800640	ПОР
312295	12	ГУ78К	25.12.2023	КАРХІВКА	КОВИЛОВИЙ	ДІБРОВА	984212	НАВ
000673	691	В12П	02.03.2020	ВИШНЕВЕ	КОСМІЧНА	ПРИДОНЕЦЬКА	522984	ПОР
966373	493	КОРИГ	10.02.2023	КОНОХІВ	ГУТИСЬКА	ПРИОБРАЖНЕ	621817	ПОР
262500	ЗУППВ	23	02.03.2025	УСТИМІВКА	ЗАМГЛАЙ	ПОПЛАВИ	205355	ПОР
656610	ВКЛЗ	52	10.11.2022	МІЛУВАТКА	БУЦНІВ	ДОЧ	511558	ПОР
036594	262	154	08.08.2023	ГАННІВКА	ВАНЧИКІВЦІ	ШМАНЬКІВЧИКИ	503311	НАВ
619166	265	ПРМЗ	17.11.2021	ГЛАФІРІВКА	БЕРЕЖАНКА	ЩЕБЕНЕВИЙ	937479	ПОР
032651	607	ВІДЧ	14.03.2021	ВАРАШ	ІВАЧКОВЕ	71 КМ	584650	НАВ
370491	351	ГУ78Ш	29.01.2025	НОВОПОШТОВА	СТАРОСІЛЛЯ	ПАВЛОПІЛЛЯ	877757	ПОР
631219	НАКАТ	ВІВН	12.07.2021	ПЛОДОРІДНА	ВЕРХНІЙ ТОКМАК II	ХОРОСТКІВ	740172	ПОР
622280	117	34	08.10.2023	ІВАНІВКА	ЗАГОРБ	НЕМЕРЧІ	594158	НАВ
976358	ВІКЛ	ГУ78К	04.11.2020	КОШМАК	12 КМ	149 КМ	985850	НАВ
294970	РЗДЧ	52	30.10.2022	303 КМ	241 КМ	СТРУМКІВКА	696996	ПОР
404399	512	ВІВЛС	10.11.2020	НОВОДУБРІВКА	БОБРОВИЦЯ	ХАЩУВАТЕ	841488	НАВ
688735	473	+ВОД	11.09.2021	ВИЛИ	ПІДВИСОКЕ	ЧЕРКАСИ	166379	ПОР
118368	550	160	12.12.2024	РУТЧЕНКОВЕ	ЖЕМЧУЖНЕ	СИНЕЛЬНИКОВЕ II	956064	ПОР
818418	ВІДМС	ДНПРМ	16.11.2022	ПРКА	КОВЕЛЬ	11 КМ	742736	ПОР
674243	78	49	07.10.2024	190 КМ	ХАРКІВ-ЛІСКИ	ОСТРЯКОВЕ	998719	НАВ
958737	ППВ	ОБЕРТ	05.10.2021	СКОТАРСЬКЕ	КОЛОМІЙЦЕВЕ	НОВА КОЗАЧА	96514	НАВ
846963	ПРМЗ	42	06.05.2025	ДЕРГАНІВКА	НОВІ ОЛЕШКИ	БОРЩІВКА	614538	ПОР
871109	25	203	03.02.2024	ПЕТРІВСЬКЕ	НАДІЯ	НЕМЕРЧІ	51438	НАВ
848147	692	ВІВЗ	13.06.2020	ПІЛАДНИКИ	248 КМ	ЯГОРІЯ	608206	НАВ

Рисунок 3.5.6. – Експортована Excel-таблиця з даними

Значна частина часу була витрачена на поступові вдосконалення, пов'язані з деталями інтерфейсу, які на перший погляд можуть здатися незначними, але мають безпосередній вплив на загальний користувацький досвід. Зокрема, були виправлені потенційні вразливості на клієнтському боці, пов'язані з текстовими полями вводу (як-от на сторінці входу, зображений на рисунку 3.5.7.), включно з обмеженням довжини введення, валідацією типів даних та попередженням XSS-атаки. Також було проведено оптимізацію стилів та сценаріїв, аби забезпечити стабільну роботу інтерфейсу на різних пристроях та в різних браузерах.

Система моніторингу вагонів Останні дані Історія записів Журнал API Увійти

Увійти

Email

Password

☐ Запам'ятати мене

Увійти

Рисунок 3.5.7. – Сторінка входу

Важливим етапом у реалізації проєкту стало впровадження системи контролю версій на базі Git, що є незамінним інструментом при роботі над будь-яким сучасним програмним продуктом. Git забезпечує повний контроль над історією змін у коді, дозволяє ефективно відслідковувати правки, повертатись до попередніх версій, а також інтегруватися з хмарними сервісами для автоматизованого розгортання. З цією метою було створено окремий репозиторій на платформі GitHub, який надалі був зв'язаний із хмарним середовищем Azure відповідно до технічних вимог проєкту.

Azure було обрано як основну платформу для хостингу веб-додатку та розгортання бази даних. Це рішення дозволило реалізувати надійне серверне середовище з можливістю масштабування та централізованого адміністрування. Для налаштування CI/CD (безперервної інтеграції та доставки) використовувалась функціональність автоматичної публікації: після кожного оновлення основної гілки репозиторію GitHub додаток автоматично деплоївся на Azure App Service.

У ході первинного розгортання було створено необхідні ресурси Azure — зокрема App Service для хостингу веб-додатку та SQL Database для зберігання даних. Налаштування параметрів таких як географічне розміщення, обсяг пам'яті, тип сховища тощо здійснювалися системним адміністратором відповідно до внутрішніх політик компанії. В ході роботи було виконано налаштування

підключення, зокрема — коректна конфігурація рядків з'єднання, дозволів фаєрволу, а також логіка взаємодії додатку з БД.

Завдяки використанню підходу Code First у рамках технології Entity Framework Core, процес ініціалізації бази даних був автоматизований. Після активації хмарного контейнера виконується запуск початкової міграції. У разі відсутності бази даних за вказаним підключенням автоматично створюється новий екземпляр відповідно до опису моделей у коді. Якщо ж база вже існує — здійснюється її підключення без втрати наявних даних.

Після успішної ініціалізації бази даних було здійснено локальний запуск веб-додатку, проведено заповнення тестовими даними, а далі — повноцінне розгортання в хмарному середовищі. Додаток був підключений до розгорнутої бази, що дозволило протестувати працездатність усієї інфраструктури та переконатися у коректності маршрутизації запитів, стабільності з'єднання та відповідності продуктивності очікуваним критеріям.

3.6. Економічна ефективність

Перед розрахунком економічної ефективності необхідно чітко окреслити сценарій, у межах якого проводиться оцінка. Це особливо важливо для ІТ-рішень, які суттєво варіюються за масштабом, а отже — і за обсягом витрат. У даному випадку розглядається типовий приклад із галузі залізничної логістики. В українських реаліях, де приватні залізниці відсутні, а вся транспортна інфраструктура централізовано управляється Укрзалізницею, технічна сторона реалізації значно спрощується. Компаніям, які володіють вагонами або користуються залізничним транспортом, не потрібно розгортати власну мережу датчиків чи створювати окрему інфраструктуру — достатньо підключитися до наявних сервісів та API, що надає УЗ.

На основі цього реалізується веб-додаток, який забезпечує повний цикл роботи з даними: від отримання інформації з зовнішнього API, через її обробку, нормалізацію та очищення, до збереження у власну базу. Далі дані повторно

аналізуються і перетворюються у зручні для сприйняття звітні форми — наприклад, у вигляді ВІ-панелей або графіків. Уся ця аналітика інтегрується в користувацький інтерфейс для подальшого використання співробітниками компанії. Такий сценарій можна вважати мінімально достатнім для автоматизації логістичних процесів із реальним ефектом уже на стартових етапах впровадження.

Далі звернемось до архітектури. В оцінці обрано монолітну архітектуру — власне, коли система не включає в себе підсистеми або мікросервіси. Всі компоненти включені в один додаток, та працюють водночас (за винятком бази даних). Дослідження [40] також згадує про телекомунікаційну інфраструктуру підприємства, яка передбачає налагоджений зв'язок між підрозділами та офісами, постачальниками та споживачами. Цей момент може ускладнюватись в залежності від підприємства, яке послуговується системою, але у випадку поточної роботи достатньо використовувати існуючі канали зв'язку та керування доступом до веб-ресурсу після його розміщення. Також ускладнення в архітектурі та зв'язках можливі внаслідок залучення більшої кількості технологій, розширення сфери впливу логістичної системи, що зображується в дослідженні [41]. Обов'язковою є наявність інформаційно-аналітичного центру, важливість впровадження якого розглядається в дослідженні [42].

Недоліками найпростішого варіанту розробки є звісно ж складність масштабування та оптимізації, можливі також труднощі в подальших змінах. Втім, даний приклад мотивується тим, що для нього простіше вирахувати терміни та ресурси, потрібні на інвестицію, водночас продемонструвавши всі переваги, адже навіть простий додаток при правильній реалізації буде більш корисним ніж не мати додатку взагалі.

В даній статті [31] наведені приблизні оцінки витрат по часу на систему, втім, вони розраховані на більш масштабний продукт та на певну команду. Звісно, чим більша команда, тим швидше отримаєте результат. І відповідно, чим складніший продукт, тим довше на нього чекатимете. В даному розділі коротко описано основні етапи та оцінки по часу на одну людину. Очікується, що розробкою займається спеціаліст, що вже мав досвід розробки подібних додатків.

Також, для того щоб обрати структуру етапів розробки було взято за основу звичайний цикл життя програмного забезпечення [43] з невеликими модифікаціями під сценарій взятий в якості прикладу:

- проектування – 1 тиждень;
- розробка бази даних – 1 тиждень;
- налаштування комунікації з API – 3 тижні;
- розробка інтерфейсу – 3 тижні;
- оптимізація – 2 тижні;
- налаштування логіки (збереження та обробка даних) – 3 тижні;
- узгодження та правки замовника – 3 тижні;
- амортизація – 3 тижні.

Кожен етап включає в себе амортизацію (певний запас днів) у розмірі 30% від початкового терміну та загальну амортизацію у розмірі 3 тижнів. Це допомагає здати проект вчасно та підготуватись на випадок форс-мажорів, оскільки в даному випадку розробник залежить як від системи зовнішнього API, так і від компанії-замовника додатку. Сумарно термін розробки в прикладі займає 19 тижнів, приблизно 4 з половиною місяці.

Як було неодноразово зазначено, в залежності від бажаного результату необхідні інвестиції можуть варіюватись. Зокрема, в статті [44] розглядається вартість утримування команди спеціалістів для розробки продукту. Виходячи з наведеної інформації, вартість команди погодинно коливається від 221 до 399 дол. за годину. Якщо підприємство не може дозволити собі команду спеціалістів, це не проблема – достатньо зменшити власні вимоги та найняти одного або кількох fullstack-розробників, або замовити розробку повного продукту без подальшої підтримки.

У статті [31] теж згадується розмір інвестицій. За словами авторів, він може коливатись від 4 тис. дол. до 40 тис. дол. і більше. Варто зауважити, що у розглянутих матеріалах термін «додаток для логістики» розглядається у ширшому сенсі ніж у даній роботі, адже загалом він може передбачати і інтеграцію з

інтернетом речей, і поєднання різних видів транспорту, платформ для роботи (веб/мобільні) тощо.

Основними ресурсами, необхідними для впровадження, є хостинг, технічна підтримка та оплата роботи розробників. Вартість розробки може варіюватися: у випадку особистого чи експериментального проєкту вона може бути безкоштовною, тоді як для повноцінного комерційного використання потребує фінансування. Це відбувається також внаслідок взаємодії з ліцензіями ПЗ для розробки. Наприклад, ліцензія Visual Studio Community 2022 дозволяє безкоштовне комерційне користування лише індивідуальним розробникам та невеликим командам [45].

Окрему увагу слід приділити безпеці, оскільки вибір розробників без належного досвіду, зокрема на фриланс-платформах, може створити ризики, внаслідок великої кількості чутливої інформації. Оптимальним варіантом є співпраця з компанією, що має відповідний досвід та імідж у сфері.

Щодо інструментів, використання open-source рішень, зокрема s, значно знижує витрати. Наприклад, комерційна ліцензія Microsoft SQL Server може коштувати до \$12 000, тоді як PostgreSQL є безкоштовним [46]. Відсутність офіційної підтримки може бути недоліком, однак для малих компаній або експериментальних проєктів доступність такого рішення є суттєвою перевагою, враховуючи альтернативу у розповсюдженості open-source рішень та форумів для вирішення проблем.

На етапі підтримки додатку можуть виникати додаткові витрати, пов'язані з виправленням помилок, оптимізацією роботи з даними та оновленням функціоналу. Вартість цих послуг значною мірою залежить від розробника, який може встановлювати довільні тарифи, особливо якщо початковий код програми не переданий замовнику. Тому важливо заздалегідь обговорювати умови співпраці та обирати перевірених виконавців, які гарантують прозорість та доступність коду.

Для коректної роботи важливі належні налаштування взаємодії з API, оскільки зміни у зовнішніх сервісах впливають на функціональність програми. Підтримка додатку передбачає регулярний моніторинг змін в API, оновлення

відповідних модулів та забезпечення безперебійного обміну даними. Неправильна або несвоєчасна адаптація до нових умов може спричинити збої у роботі ключових функцій.

Також підтримка передбачає наявність людини, яка виконуватиме обов'язки системного адміністратора в компанії, та наявність адміністративного інтерфейсу в розробленому додатку, який дозволяє керувати доступом, налаштуваннями та даними. Важливо, щоб права адміністратора були чітко визначені програмним забезпеченням, а також контролювалась безпека входу та зміни конфігурацій. Відсутність такого механізму може бути фатальною для системи в цілому.

Для розгортання хмарної інфраструктури та введення додатку в експлуатацію є два основних варіанти – використання послуг хмарних провайдерів та розгортання на власних серверах компанії.

Власні сервери можуть бути привабливими для компаній, які вже мають необхідне обладнання, оскільки це дає більше контролю над власним продуктом та зменшує безпекові ризики. Втім, в умовах де неможливо гарантувати цілісність фізичної інфраструктури та її безперебійну роботу (через відключення електроенергії, наприклад), цей варіант краще залишити для кращих часів, зробивши вибір на користь провайдерів.

Додатково, розгортання на власних серверах та налаштування програми – це по визначенню складніша робота, ніж розгортання на хмарному сервісі і потребує більш досвідчених фахівців, що може привести до більших термінів та коштів необхідних на розробку та впровадження. Тому, в якості прикладу розглянемо розгортання на платформі Azure, яка має чудово налагоджену інтеграцію з .NET, оскільки обидві є продуктами компанії Microsoft.

Хмарні сервіси зазвичай мають безліч переваг, серед них масштабованість, оплата по факту використання (без переплати), завчасно налаштоване середовище розгортання. Azure також має доступний користувацький інтерфейс, з яким може впоратись навіть людина без DevOps-досвіду, хоч це і потребуватиме трохи часу.

Для запуску найпростішого застосунку необхідно два основних ресурси – SQL база даних та сервіс веб-застосунку. Щоб попередньо розрахувати вартість

подібного розгортання можна скористатись калькулятором [47] від Azure. Прайсинг прикладу зображений на рисунку 3.6.1.

Your Estimate

▼ Azure SQL Database	Single Database, vCore, General Purpose, Provisi...	Upfront: \$0.00	Monthly: \$372.97
▼ App Service	Basic Tier; 1 B1 (1 Core(s), 1.75 GB RAM, 10 GB St...	Upfront: \$0.00	Monthly: \$54.75
▼ Microsoft Cost Management	No charge for managed Azure spend. 0 Manage...	Upfront: \$0.00	Monthly: \$0.00

Support

SUPPORT:

Basic (Included)	\$0.00
------------------	--------

Рис. 3.6.1. – Приклад сервісів з цінами на місяць.

Звісно, це не мінімальна ціна, але і не максимальна. Дуже багато залежить від налаштувань і функцій як додатку, так і бази даних. Базу даних можливо встановити за 18 дол. на місяць, з програмою можна встановити ціну по факту використання ресурсів, що цілком реалістично може обходитись в 50 дол. на місяць. Швидше за все, таку ціну можуть дозволити навіть невеликі підприємства.

Втім, окрім мінімуму, хмарний сервіс пропонує також вкрай багато додаткових функцій, починаючи від моніторингу розгорнутих ресурсів і закінчуючи автоматизацією певних дій з ними, резервних копій та іншого. Це може видатись привабливим для окремих компаній які готові вкласти більші кошти в проект.

Насамперед ефективність впровадження системи залежить від конкретного підприємства, його потреб та цілей. Із переваг які може запропонувати додаток можна виділити наступні:

- детальна аналітика на основі сформованої бази даних може допомогти у визначенні ефективності наявної логістичної мережі шляхом використання точних метрик на основі великого датасету;
- при відсутності єдиної системи інформація може надходити електронними листами одразу на певну кількість електронних адрес, може існувати в паперовому форматі або може узгоджуватись по телефону. Хоч варіанти окрім першого не такі ймовірні, все ж, дані які оновлюються щосекунди

будуть надходити із запізненням. При наявності системи замовник може сам обрати частоту оновлень;

- додаток дає можливість зібрати всі джерела інформації в одному місці. Це допомагає уникнути збитків пов'язаних із раптовою втратою інформації (внаслідок резервного копіювання), зменшує час на пошуки та відкриває нові перспективи для подальших вдосконалень.

Втім, є і перепони, які стоять на шляху до впровадження нової системи. Серед них варто виділити необхідний час на освоєння продукту персоналом (що залежатиме від самого персоналу та компанії), питання безпеки (як на етапі розробки так і на етапі експлуатації) та самі дані, які можуть стати перепорою. Дані які надходять про логістичні об'єкти перш за все різнопланові. Швидше за все вони будуть використовуватись людьми з різними задачами та компетенціями – головне питання чи можливо буде це поєднати в кінцевому продукті та чи буде це зручно для всіх сторін.

3.7. Оцінка ризиків при впровадженні

При розробці та впровадженні інноваційних рішень важливим етапом є оцінка ризиків. Це особливо важливо в контексті таких чутливих сфер, як логістика, де одна похибка може призвести до критичних наслідків для компанії. У дослідженні [48] стверджується, що управління проєктними ризиками – насамперед безперервний процес, заснований на системному підході. Він передбачає своєчасне виявлення, оцінку та реагування на ризики з метою мінімізації їхнього впливу на результати проєкту, а це натомість забезпечує узгоджене реагування на загрози на всіх етапах життєвого циклу проєкту.

Для поточної розробки оптимальними є методи «опитування експертів» та «картки Кроуфорда». Оскільки проєкт має небагато складників, очікується менше етапів на яких можуть впливати ризики, а оскільки проєкт також не має великої складності в плані вимог, більшість кваліфікованих фахівців з релевантним досвідом на тому чи іншому етапі кар'єри з великою ймовірністю були залучені в

проект такого стибу. Звідси, «опитування експертів» у вигляді персональних інтерв'ю надасть достатню кількість якісної інформації для подальшої роботи та не перевантажить аналіз надмірними даними, так званий принцип «леза Оккама».

В якості альтернативи також цілком ефективно обрати «картки Кроуфорда», через чітку пріорітизацію ризиків у даному методі. Часто при наявності обмежених ресурсів (особливо часових) надміру важливо закривати ті чи інші ризики в розробці по мірі їх важливості. Наприклад, краще закрити безпекову прогалину в розробці і провалити терміни здачі, ніж навпаки.

Після аналізу можна виділити наступні ризики, зображені в таблиці (Додаток Г).

Для якісного аналізу ризиків необхідно розробити систему оцінювання, адаптовану до специфіки проекту. Зокрема, важливо визначити шкалу ймовірності виникнення кожного ризику, а також окрему шкалу для оцінки його потенційних наслідків. Шкали зображені на таблицях 3.7.1. та 3.7.2.

Шкала оцінки ймовірності настання ризиків		
Ранг	Значення	Критерій
5	Майже вірогідно	Шанси виникнення події надзвичайно великі (ймовірність > 95%)
4	Дуже ймовірно	Шанси настання досить великі (70% < ймовірність < 95%)
3	Можливо	Шанси виникнення події помірні (30% < ймовірність < 70%)
2	Мало ймовірно	Виникнення події досить сумнівне (5% < ймовірність < 30%)
1	Майже неможливо	Виникнення події вкрай сумнівне (ймовірність < 5%)

Таблиця 3.7.1. – Шкала оцінки ймовірності настання ризиків

Шкала оцінки тяжкості наслідків настання ризиків		
Ранг	Значення	Критерій
5	Катастрофічні	Втрати до 400 роб. год. виконавця та/або 100 000 грн. та більше
4	Критичні	Втрати менше 160 роб. год. виконавця та/або менше 16 000 грн.
3	Відчутні	Втрати менше 80 роб. год. виконавця та/або менше 8 000 грн.
2	Помірні	Втрати менше 40 роб. год. виконавця та/або менше 4 000 грн.
1	Допустимі	Втрати менше 10 роб. год. виконавця та/або менше 1 000 грн.

Таблиця 3.7.2. – Шкала оцінки тяжкості наслідків настання ризиків

Для систематичної оцінки ризиків, що можуть вплинути на успішність реалізації проєкту, було застосовано методіку ранжування на основі двох параметрів: ймовірності виникнення та тяжкості наслідків. Кожен ризик отримує оцінку за обома шкалами за п'ятибальною системою, де 1 — мінімальне значення, а 5 — максимальне. Ранг ризику обчислюється як добуток цих двох величин і дозволяє визначити пріоритетність реагування. У таблиці нижче наведено перелік виявлених ризиків, відповідні оцінки та обчислений ранг кожного з них. Це дає змогу зосередити увагу на критичних аспектах і своєчасно розробити заходи з їхньої мінімізації. Таблиця з розрахованими рангами знаходиться в додатку Г.

Для кількісного аналізу варто використовувати одночасно 3 методи: аналогів, аналіз сценаріїв та імітаційні методи. Аналоги були обрані через загальну схожість подібних проєктів та можливість віднайдення потрібної інформації як в мережі інтернет, так і у кваліфікованих експертів. Разом з тим, цей метод взаємодоповнюючий з аналізом сценаріїв, який було обрано через його конструктивність та чіткий план дій, що охоплює від кращого до гіршого варіантів розвитку подій. І імітаційні методи можуть допомогти з моделюванням проблемних ситуацій пов'язаних з технічними ризиками, як-от високе

навантаження системи. Тобто, послідовність наступна: Метод аналогій; З урахуванням відомостей із попереднього методу, аналіз сценаріїв; До найгірших сценаріїв застосовувати імітаційні методи (де можливо).

При реагуванні на ризики тим чи іншим чином будуть необхідні всі методи окрім передачі ризику, а саме ухилення, зниження та прийняття ризиків. Передача ризиків не застосовна конкретно до перелічених у якісному аналізі ризиків, тому що їх нікому передати. Навіть якщо є взаємодія з третьою стороною на яку можна перекласти відповідальність, майже неймовірно настання сценарію в якому третя сторона візьме на себе цей ризик. Тому реагувати доведеться на стороні, яка впроваджує проект, підбираючи реакцію в залежності від рангу та категорії ризику, а також доступних стратегій.

Максимальний ранг у ризику «Неправильно розрахована вартість хостингу», даний ризик має різницю у 3 пункти з найближчим ризиком із великим впливом. Щоб знизити його вплив, мінімізувати або знешкодити цілком достатньо кілька простих кроків:

- вичерпний моніторинг проекту на етапі розробки. На цьому етапі можливо дослідити ресурси, які використовуватиме система, при цьому слід проводити тестування в умовах, якнайбільше наближених до реальних;
- початковий хостинг на серверах компанії. Даний крок потенційно потребуватиме більших інвестицій, втім, під час хостингу на серверах компанії можливо подивитись як насправді поводитиметься проект і мати точне уявлення про необхідні ресурси;
- підвищення кваліфікації фахівців. Є цілком доречним підвищити кваліфікацію персоналу який буде відповідати за взаємодію із хмарними провайдерами. Здійснити це можна, наприклад, за допомогою курсів;
- посилений моніторинг кожного етапу розгортання та впровадження. Хмарні провайдери зазвичай надають інформацію про поточні витрати по кожному проекту, і якщо фахівець помітить надмірне зростання витрат на певному етапі, він може зупинити потік до настання найгірших наслідків. Через сам

алгоритм роботи програми, при постійному моніторингу проблему можна помітити на самих ранніх етапах.

Детальний перелік дій-реакцій на ризики міститься в таблиці 3.7.3.

Фаза життєвого циклу	Етап проекту	Дії з управління ризиками
I	Планування	Проведення окремої зустрічі для ознайомлення розробників з природою даних та потребами працівників; Спільне складання та узгодження плану, термінів та бюджету
	Проектування	Включення аварійних функцій серед компонент системи, таких як резервне копіювання, механізм на випадок відмови зовнішнього API, механізми на випадок витоків даних чи кібератак
II	Розробка	Налагодження регулярної комунікації замовників з командою розробників; Впровадження та контроль безпекових протоколів; Постійне узгодження проекту з останніми версіями стороннього API
	Тестування	Проведення окремої зустрічі для ознайомлення розробників з природою даних та потребами працівників; Спільне складання та узгодження плану, термінів та Проведення стрес-тестів під максимальним навантаженням та тестування різних сценаріїв відповідно до ризиків з попередніх таблиць; Ознайомлення працівників із демо-версією продукту, надання зворотного зв'язку
III	Впровадження	Початковий хостинг продукту на серверах компанії; Ретельний моніторинг потracених ресурсів програмою, можливе задіювання спеціалістів із тестів з проникнення
	Експлуатація	Постійна підтримка зворотнього зв'язку розробників із працівниками; Регулярні перевірки на відповідність безпековим протоколам та аналіз вразливостей

Таблиця 3.7.3. – Заходи із реагування на ризики

Підсумовуючи, аналіз ризиків виявив низку потенційних загроз, які потребують постійної уваги на всіх етапах проекту. Зведені дані наведено в

таблиці, що розміщена в додатку Д. Такий аналіз є критично важливим, оскільки навіть незначна помилка може призвести до серйозних наслідків, включно з повним припиненням роботи системи. Крім того, оцінка ризиків дозволяє краще розуміти як технічні виклики розробки, так і їхній вплив на економічну ефективність усередині логістичної компанії.

3.8. Організаційне забезпечення

Організаційне забезпечення системи є важливим компонентом, що визначає ефективність її використання в реальних умовах експлуатації. Воно охоплює структуру управління проектом, розподіл обов'язків серед учасників, а також процедури, необхідні для забезпечення безперебійної роботи системи на всіх етапах її життєвого циклу. Важливими аспектами організаційного забезпечення є управління проектом, планування ресурсів, організація навчання користувачів та забезпечення технічної підтримки [49].

Управління проектом передбачає розробку детального плану реалізації системи, визначення етапів її впровадження, а також контроль за виконанням кожного з них. Для досягнення поставлених цілей важливо правильно визначити терміни, бюджет, ресурси та персонал, що братимуть участь у розробці та впровадженні системи. Під час управління проектом необхідно постійно моніторити прогрес, виявляти ризики та проблеми на ранніх етапах і вживати необхідних заходів для їх усунення.

Планування ресурсів включає як матеріальні, так і людські ресурси, необхідні для забезпечення ефективної роботи системи. Ресурсне забезпечення повинно охоплювати необхідне обладнання, програмне забезпечення, а також кваліфікацію персоналу, який працюватиме з системою. Для цього потрібно провести аналіз існуючих ресурсів та розробити план по їх оптимальному використанню, що дозволить мінімізувати витрати та максимізувати ефективність впровадження.

Важливим етапом підготовки до впровадження системи є організація навчання користувачів. Навчання має на меті ознайомити з основними функціями системи, її інтерфейсом, принципами роботи з даними та взаємодії з іншими компонентами. Для цього можуть бути проведені тренінги, семінари, створені навчальні посібники або відеоуроки. Важливо, щоб навчання було доступним для всіх категорій користувачів, враховуючи рівень їх технічної підготовки та особливості роботи з інформаційними системами.

Також неодмінним елементом організаційного забезпечення є розробка документації для користувачів і адміністраторів системи. Це може включати інструкції щодо налаштування, використання та обслуговування програмного забезпечення, а також опис процедур для розв'язання можливих технічних проблем. Якісна документація є необхідною для забезпечення ефективного використання системи в умовах, коли необхідна швидка реакція на виникаючі ситуації.

Для забезпечення безперебійної роботи системи необхідно організувати постійну технічну підтримку. Сюди входить надання допомоги користувачам, усунення неполадок, а також своєчасне оновлення програмного забезпечення та забезпечення безпеки. Технічна підтримка має бути доступною для користувачів у будь-який час, оскільки система логістики вагонів є критично важливою для нормальної роботи компанії. Для цього можуть бути створені багатоканальні служби підтримки, включаючи телефонні лінії, онлайн-чати, електронну пошту та інші засоби комунікації.

Всі процеси, пов'язані з обробкою та зберіганням даних, повинні відповідати вимогам законодавства та політикам компанії щодо захисту інформації. Важливо встановити чіткі правила доступу до даних, шифрування інформації та захисту від несанкціонованого доступу. Це також передбачає використання засобів моніторингу та аудитів для відстеження потенційних загроз та реакцію на них.

Також, важливою частиною організаційного забезпечення є організація взаємодії між різними підрозділами компанії, які будуть використовувати систему. Цього можливо досягнути лише об'єднавши ефективну координацію між ІТ-

відділом, логістичним відділом, керівництвом та іншими службами, що забезпечують виконання функцій системи. Комунікація між цими підрозділами є критично важливою для оперативного вирішення виникаючих питань і забезпечення безперебійної роботи всієї системи.

Завершальним етапом організаційного забезпечення є планування розвитку та вдосконалення системи після її впровадження. Враховуючи швидкий розвиток технологій, важливо передбачити можливість масштабування та модернізації системи, щоб вона могла відповідати новим вимогам та викликам. Потрібно розробити стратегії оновлення, інтеграції нових функцій, а також планувати подальше навчання користувачів та технічної підтримки для збереження актуальності і ефективності системи в майбутньому.

Організаційне забезпечення є невід'ємною частиною системи, її успішної реалізації та експлуатації. Без належного виконання та реалізації це загрожує передчасному старінню, або неналежній експлуатації системи в моменті, відповідно, ставлячи під сумнів її доцільність.

ВИСНОВКИ

В ході проведеної роботи було досліджено можливості впровадження веб-застосунку для логістики, визначено ресурси для розробки та підтримки системи, зазначено переваги та недоліки. Розроблений продукт забезпечує ефективну обробку та представлення логістичних даних у зручному для користувача вигляді, а також дозволяє значно скоротити обсяг ручної праці, пов'язаної зі збором і аналізом інформації. Упровадження системи дозволяє відслідковувати актуальні логістичні операції, аналізувати рух транспортних активів, виявляти простой, оцінювати ефективність перевезень та формувати висновки щодо поточного стану логістичних процесів. Зберігання історичних даних у централізованій базі відкриває широкі можливості для подальшої аналітики та підвищення рівня прозорості управління.

Варто зазначити, що ефективність функціонування системи значною мірою залежить від стабільності роботи зовнішнього API, який є джерелом первинних даних. Проте навіть з урахуванням цього фактору розроблена ІУС забезпечує значний приріст функціональності порівняно з наявними на підприємстві підходами, які здебільшого базувалися на ручному введенні та розрізненому зберіганні інформації.

Найважливішим досягненням є не лише реалізація технічної частини системи, а й створення гнучкої архітектури, яка дозволяє масштабувати рішення відповідно до майбутніх вимог. Зокрема, створено всі передумови для подальшого впровадження інтелектуальних компонентів: алгоритмів прогнозування затримок, оптимізації маршрутів, оцінки навантаження тощо. Система вже зараз підтримує фільтрацію, агрегацію, експорт та візуалізацію даних, що може стати основою для побудови аналітичних панелей або впровадження методів машинного навчання [50]. Таким чином, відбувається поступовий перехід від традиційної інформаційної системи до елементів систем підтримки прийняття рішень.

Робота над проектом дозволила поглибити знання щодо побудови архітектури прикладних систем, взаємодії з API, роботи з ORM-фреймворками (EF

Core), впровадження хмарної інфраструктури, а також оптимізації продуктивності та безпеки веб-додатків.

Логістика вагонів у залізничному транспорті включає великі обсяги даних, недоліки відсутності автоматизації яких цілком очевидні. Водночас, реалізація інформаційних систем для процесів логістики включає певні виклики, такі як безпека даних, потреби конкретного підприємства та інтеграція з існуючими системами, потреба в точності та надійності, себто ціна помилки вкрай висока.

Існує ряд рішень які були створені для покращення та автоматизації процесів логістики. Втім, хоч вони і пропонують широкий функціонал, їхня орієнтованість на середньостатистичного користувача є недоліком, що обмежує можливості їх використання на підприємствах малого та середнього рівня. Для подібних компаній-перевізників пріоритетними радше слугують гнучкість системи, вартість розгортання та можливість видозміни розробки в залежності від потреб та ситуації, особливо в кризові періоди.

Система складається з різних модулів, які виконують ті чи інші етапи роботи з даними. Сюди входить інтерфейс користувача, база даних, сервіс для роботи із зовнішнього API для отримання даних. Бізнес-логіка є важливою компонентою – саме вона відповідає за обробку даних, перед відображенням користувачу. Архітектура також відповідає принципам Clean Architecture та MVC, що робить систему підтримуваною та масштабованою. Задачі передусім диктуються потребами підприємства, і включають автоматичний збір даних за допомогою фонових сервісів, збереження всіх даних у сховище, їх трансформація для перегляду та експортування у формі звітів та можливість взаємодіяти із даними, зокрема шукати необхідну інформацію та сортувати результати пошуків.

В якості моделей було обрано реальні сутності які є наслідком нормалізації початкового датасету, наданого стороннім API. Вони допомагають зменшити кількість дубльованої інформації та зберігають показовість даних, що має критичне значення для кінцевого користувача. Розроблене сховище даних відповідає вимогам до розробки, зберігаючи гнучкість та масштабованість.

Ключовою таблицею є таблиця вагонів, оскільки саме потребою в моніторингу вагонного парку була зумовлена розробка системи.

Розроблене програмне забезпечення складається з різних модулів, що теж сприяє підтримуваності та масштабованості. Більша частини логіки інкапсульована в сервіси, а користувацький інтерфейс виокремлений згідно до парадигми MVC у контролери, представлення та моделі представлень. Зокрема, наявні фонові сервіси, активність яких забезпечує постійне оновлення даних згідно документації стороннього API.

Технічне оснащення необхідне для впровадження системи складається із серверів для хостингу сховища та веб-додатку, мережевого обладнання, яке потрібне як для комунікації додатка з джерелом даних так і для доступу до додатка, персональних пристроїв робітників, для користування системою в робочих цілях. Організаційне забезпечення включає проведене навчання робітників для користування системою, розробка документації, яка буде точною та показовою, організація взаємодії між різними підрозділами та виконавцями які будуть послуговуватись системою.

Важливим напрямом удосконалення є інтеграція моделей та алгоритмів для покращення аналітики. Це дозволить підприємствам приймати обґрунтовані рішення, знижувати витрати та підвищувати точність прогнозування. Враховуючи тенденції цифровізації, подальший розвиток таких систем має значний потенціал у сфері логістики.

Водночас дослідження [51] вказує на необхідність удосконалення логістичних інформаційних систем, оскільки існуючі рішення часто є недостатньо гнучкими і не завжди ефективно реагують на кризові ситуації. Впровадження концепції глобальної транспортної інформаційної системи (GTIS) розглядається як перспективний напрямок, що передбачає інтеграцію різних видів перевезень, автоматизацію процесів та застосування передових технологій, таких як IoT, AI та Big Data.

Згідно з дослідженням [41], іншим основним напрямом розвитку є взаємодія фізичних і цифрових компонентів, що дозволяє автоматизувати управління

ланцюгами поставок, оптимізувати маршрути та підвищувати точність прогнозування попиту. У майбутньому цифрові логістичні системи стануть ще більш адаптивними завдяки використанню глибокого аналізу даних і прогнозного моделювання. Це дозволить компаніям не лише оперативно реагувати на зміни, а й передбачати можливі ризики, підвищуючи ефективність управління ресурсами. Синергія сучасних цифрових технологій сприятиме створенню нових бізнес-моделей, орієнтованих на сталий розвиток і персоналізоване обслуговування клієнтів. Серед дискусійних питань які не були в повній мірі розкриті хочеться виділити безпеку даних та механізми її збереження, впровадження штучного інтелекту та його роль в логістичних системах.

Отже, як показують дослідження, розробка та впровадження системи є доцільним та необхідним, оскільки це допоможе разюче збільшити ефективність роботи з розрізненими логістичними даними, покращити точність та надійність, структурувати історію записів та приймати більш обґрунтовані рішення на їх основі. Співвідношення її ефективності до необхідного устаткування разом із перевагами гнучкості та адаптивності робить можливим впровадження такої системи на малих та середніх підприємствах, слугуючи значно кращою альтернативою паперовим документам та розрізненій звітності. На поточному етапі система не містить вбудованих інтелектуальних модулів, однак її архітектура закладає надійне підґрунтя для їхнього впровадження. Розроблений фрагмент ІУС не лише вирішує ряд актуальних завдань підприємства, але й відкриває перспективу для створення повноцінної інтелектуальної системи логістичного управління. Він має високу практичну цінність і є масштабованим, що підтверджує актуальність та ефективність обраного технічного рішення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Харчук О. Г. Зарубіжний та вітчизняний досвід впровадження інформаційних технологій в залізничному транспорті. Збірник наукових праць Державного університету інфраструктури та технологій. 2013. Т. 23–24. С. 197–202. URL: http://nbuv.gov.ua/UJRN/Znpdetut_eiu_2013_23-24_30.
2. YC.Market. Цифрові технології у логістиці [Електронний ресурс]. Блог YC.Market. 25.06.2024. Режим доступу: <https://blog.youcontrol.market/tsifrovi-tiekhnologhiyi-u-loghistitsi/>.
3. Тимощук О. М., Мельник О. В. Інформаційно-логістичні системи в сучасних транспортних технологіях. Інвестиції: практика та досвід. 2015. № 22. С. 79–82. URL: <http://www.investplan.com.ua/?op=1&z=4729&i=15>
4. Короленко О. Б. Вплив інформаційних технологій на ефективність логістичних ланцюгів. Здобутки економіки: перспективи та інновації. 2025. Т. 15. DOI: <https://doi.org/10.5281/zenodo.14914740>.
5. Крамаренко О., Самойленко А. Особливості створення сучасних логістичних інформаційних систем. Маркетингові стратегії, підприємництво і торгівля: сучасний стан, напрямки розвитку. 2024. С. 450–453. URL: <http://ndeks.lviv.ua/pdf/21.06.2024.pdf#page=450> (дата звернення: 12.05.2025).
6. Використання інформаційних технологій для оптимізації автомобільних перевезень в межах транспортної системи міста / Т. В. Болвановська та ін. Транспортні системи та технології перевезень. 2025. вип. 29. С. 59–65. DOI: <https://doi.org/10.15802/tstt2025/325475>.
7. Петренко О., Корнійко Я., Невідім Є. Вплив інформаційних технологій на бізнес-процеси логістичних підприємств. Економіка і управління: Збірник наукових праць. 2024. вип. 56. С. 76–84. DOI: <https://doi.org/10.32703/2664-2964-2024-56-76-84>.
8. Ляліна Н. С., Панчук А. С., Матвієнко-Біляєва Г. Л. Впровадження сучасних методів логістики в підприємницькій діяльності. 2020. DOI: <https://doi.org/10.32840/2522-4263/2020-2-20>

9. Дмитрів Д.В. Майбутнє економіки в епоху інформаційного суспільства: матеріали міжнар. наук. - практич. Інтернет-конф., 11 квітня 2017 року. – Вінниця, ВНАУ, 2017. – 165 с. / Дмитрів Д.В, Рогатинська О.Р. URL: <http://repository.vsau.org/getfile.php/13060.pdf>
10. Харламов П.О. Інформаційні системи виробництва та логістики / П.О. Харламов // Міжнародна транспортна інфраструктура, індустріальні центри та корпоративна логістика : матеріали 16-ї науково-практичної міжнародної конференції (4-5 червня 2020 р., м. Харків). – Харків : УкрДУЗТ, 2020. – С. 82-83. URL: <http://lib.kart.edu.ua/handle/123456789/12299>
11. Гарафонов О. І. Теоретичні засади впровадження інформаційної логістики в туризмі. Сучасні тенденції та проблеми управління. 2015. С. 76–80. URL: <https://core.ac.uk/download/pdf/373098016.pdf>.
12. Чернописька Н. В. Методичні підходи оцінювання логістичної діяльності підприємства. Вісник Національного університету "Львівська політехніка". 2008. Т. 623 : Логістика. С. 265–271. URL: https://vlp.com.ua/files/38_4.pdf.
13. Використання інформаційних технологій обробки даних в сучасних системах транспортної логістики / Л. М. Дегтярьова та ін. Сучасні інформаційні технології у сфері безпеки та оборони. 2020. Т. 37. С. 139–144. DOI: <https://doi.org/10.33099/2311-7249/2020-37-1-139-144>.
14. Яценко, Р. М. Інформаційні системи в логістиці : навч. посіб. / Р. М. Яценко, І. В. Ніколаєв. – Харків : ХНЕУ, 2012. – 232 с. URL: <https://duikt.edu.ua/ua/lib/1/category/1137/view/1483>
15. Culver M. What is Rail Logistics and Why Does it Matter [Електронний ресурс] / Matt Culver // Режим доступу до ресурсу: <https://www.rsilogistics.com/blog/what-is-rail-logistics-and-why-does-it-matter/>.
16. Мельник Р. В. Інформаційні системи як важлива складова розвитку логістики на підприємстві / Р. В. Мельник. – ВНТУ, 2024. URL: <https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/41625/20907.pdf?sequence=3&isAllowed=y>

17. Why information systems are critical in logistics [Електронний ресурс]. Режим доступу: <https://inlogica.com/en/articles/why-information-systems-are-critical-in-logistics/>
18. Кривов'язюк І. Оцінювання ефективності впровадження цифрових технологій в сфері управління логістикою підприємств. Collection of Scientific Papers «Scientia». 2023. С. 34–38. URL: <https://previous.scientia.report/index.php/archive/article/view/703>.
19. Транспорт і зв'язок [Електронний ресурс]. Укрстат. Режим доступу: https://www.ukrstat.gov.ua/operativ/menu/menu_u/tr.htm.
20. Кар'єри України [Електронний ресурс] . Режим доступу: <https://www.karer.in.ua/vagon.php>.
21. Статистичні дані про Українські залізниці [Електронний ресурс]. Режим доступу: <https://mtu.gov.ua/content/statistichni-dani-pro-ukrainski-zaliznici.html>.
22. Transporeon [Електронний ресурс]. Режим доступу: <https://www.transporeon.com/en>
23. Rail Management Consultants International [Електронний ресурс]. Режим доступу: <https://rmcon-int.de/en/railsys-suite-en/>
24. Rail-Flow [Електронний ресурс]. Режим доступу: <https://www.rail-flow.com/en/rail-transport-management/>
25. CargoTrack [Електронний ресурс]. Режим доступу: <https://en.cargotrack.ro/>
26. Mottesi C. The Basics of Asset Health Management [Електронний ресурс] / Celeste Mottesi. – 2024. – Режим доступу: <https://blog.invgate.com/asset-health-management>.
27. Wood, L. C., Reiners, T., & Pahl, J. (2015). Manufacturing and logistics information systems. In M. Khosrow-Pour (Ed.), Encyclopedia of Information Science and Technology (3rd ed., pp. 5136-5144). DOI:10.4018/978-1-4666-5888-2.ch507
28. MVC Architecture [Електронний ресурс]. Режим доступу: <https://www.ramotion.com/blog/mvc-architecture-in-web-application/>
29. Martin R. C. The Clean Architecture [Електронний ресурс]. Robert Martin – Режим доступу: <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>.

30. Langlade A. The repository design pattern [Електронний ресурс]. Arnaud Langlade – Режим доступу: <https://www.arnaudlanglade.com/repository-design-pattern/>.
31. Mariam A. Trangotech [Електронний ресурс]. Режим доступу: <https://trangotech.com/blog/logistics-app-development-cost/>.
32. Overview of ASP.NET Core [Електронний ресурс]. Режим доступу: <https://learn.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-9.0>
33. Smith A. Why ASP.NET Development is Best for Scalable Web Apps. Medium [Електронний ресурс]. Режим доступу: <https://web-and-mobile-development.medium.com/why-asp-net-development-is-best-for-scalable-web-apps-faea5cacb53d>.
34. Музика Р., Васюта В. Pros and Cons of .NET Software Development [Електронний ресурс]. Leobit. Режим доступу: <https://leobit.com/blog/pros-and-cons-of-net-software-development/>.
35. Overview of Entity Framework Core – EF Core [Електронний ресурс] // Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/ef/core/>
36. Ardil Özkan. Fluent API Explained: Simplify Object-Oriented Programming with Method Chaining [Електронний ресурс] . Medium. 08.12.2024. Режим доступу: <https://medium.com/@ozkanardil/fluent-api-explained-simplify-object-oriented-programming-with-method-chaining-d81655d71bd5>.
37. Taylor J. Clean Architecture with .NET Core: Getting Started [Електронний ресурс]. 05.02.2020. Режим доступу: <https://jasontaylor.dev/clean-architecture-getting-started/>.
38. Yiğit K. E. The SOLID Principles of Object-Oriented Programming Explained in Plain English [Електронний ресурс]. freeCodeCamp. 20.08.2020. Режим доступу: <https://www.freecodecamp.org/news/solid-principles-explained-in-plain-english/>.
39. Bootswatch [Електронний ресурс]. Режим доступу: <https://bootswatch.com/>.
40. Лисько Т. Г., Луценко І. С. Основні засади функціонування логістичної системи підприємства. Актуальні проблеми економіки та управління : збірник наукових праць молодих вчених. 2012. вип. 6. URL: <https://ela.kpi.ua/handle/123456789/22619>.

41. Скіцько В. І. Синергія цифрових технологій в логістичних системах. Інвестиції: практика та досвід. 2018. вип. 15. С. 18–24. URL: <http://www.investplan.com.ua/?op=1&z=6240&i=3>.
42. Белова І., Мушеник І. Інформаційно-аналітичне забезпечення дослідження ринку логістичних послуг. Інститут бухгалтерського обліку, контроль та аналіз в умовах глобалізації. 2021. С. 7–23. DOI: <https://doi.org/10.35774/ibo2021.01-02.007>.
43. QualityAssurance Group [Електронний ресурс]. Режим доступу: <https://qagroup.com.ua/publications/6-osnovnykh-etapiv-sdlc/>.
44. Altynpara E., Ropalo Y. Logistics Mobile App Development: Types, Cost, and All to Know About It [Електронний ресурс]. Cleveroad. Режим доступу: <https://www.cleveroad.com/blog/logistics-app-development-cost/>.
45. License Terms [Електронний ресурс]. Microsoft Visual Studio. Режим доступу: <https://visualstudio.microsoft.com/license-terms/vs2022-ga-community/>.
46. PostgreSQL [Електронний ресурс]. Режим доступу: <https://www.postgresql.org/>.
47. Microsoft Azure. Pricing Calculator [Електронний ресурс]. Режим доступу: <https://azure.microsoft.com/en-us/pricing/calculator/>.
48. Дорошенко Т. Вдосконалення системи управління проєктними ризиками. Економічні, соціальні та інформаційні механізми формування та вдосконалення системи управління проєктами. 2021. С. 337–350.
49. Нікітюк В. Г. Організаційне забезпечення формування якісно нових логістичних систем на підприємствах машинобудування. Наукові записки Національного університету «Острозька академія». Серія «Економіка» : науковий журнал. Острог : Вид-во НаУОА, березень 2020. № 16(44). С. 81–89. URL: <https://journals.oa.edu.ua/Economy/article/download/2727/2495>
50. Петренко О. І., Черниш В. М. Інновації в логістичній діяльності, їх переваги та недоліки. Наукова спільнота. Сек. Логістика і транспорт. 2020. №15. URL: <http://www.spilnota.net.ua/ua/article/id-3441/>
51. Чобіток В., Літвінчик С. Системи інформаційного забезпечення транспортної логістики в підприємницькій діяльності. Herald of Khmelnytskyi National University.

Economic Sciences. 2024. T. 332. C. 14–21. DOI: <https://doi.org/10.31891/2307-5740-2024-332-2>.

ДОДАТКИ

ДОДАТОК А

Код фонового сервісу для оновлення даних

```
4 using Microsoft.IdentityModel.Tokens;
5 using System.Text;
6
7 namespace CarWebApp.Services
8 {
9     4 references
10     public class CarBackgroundService : BackgroundService
11     {
12         private readonly ILogger<CarBackgroundService> _logger;
13         private readonly IServiceScopeFactory _scopeFactory;
14         private readonly IConversionService _conversionService;
15         private readonly IConfiguration _configuration;
16
17         private int _triggerPeriodMinutes;
18
19     0 references
20     public CarBackgroundService(IServiceScopeFactory scopeFactory, ILogger<CarBackgroundService> logger,
21         IConversionService conversionService, IConfiguration configuration)
22     {
23         _logger = logger;
24         _scopeFactory = scopeFactory;
25         _conversionService = conversionService;
26         _configuration = configuration;
27
28         _triggerPeriodMinutes = _configuration.GetValue("BackgroundService:TriggerPeriodMinutes", 300);
29     }
30
31     0 references
32     protected override async Task ExecuteAsync(CancellationToken stoppingToken)
33     {
34         await Task.Delay(TimeSpan.FromMinutes(0), stoppingToken);
35         bool errorDelayFlag = false;
36
37         try
38         {
39             while (!stoppingToken.IsCancellationRequested)
40             {
41                 Console.WriteLine("=====");
42                 _logger.LogInformation("BCHGRND: Initiating data fetching");
43
44                 using (var scope = _scopeFactory.CreateScope())
45                 {
46                     var externalApiService = scope.ServiceProvider.GetRequiredService<IExternalApiService>();
47                     var dbService = scope.ServiceProvider.GetRequiredService<IDbService>();
48                     var apiLoggingService = scope.ServiceProvider.GetRequiredService<IApiLoggingService>();
49
50                     try
51                     {
52                         _logger.LogInformation("BCHGRND: Initiating external API");
53                         var data = await externalApiService.GetDataAsync();
54
55                         if (data.IsNullOrEmpty())
56                         {
57                             _logger.LogInformation("BCHGRND: Break, data is empty");
58                             break;
59                         }
60
61                         _logger.LogInformation("BCHGRND: Converting data...");
62                         var dataMidTier = _conversionService.ConvertInputToMidTier(data);
63                         _logger.LogInformation("BCHGRND: Saving data to db");
64                         await dbService.SaveDataAsync(dataMidTier);
65                         _logger.LogInformation("BCHGRND: Success");
66                         await apiLoggingService.LogAsync("Background service finished");
67                     }
68                     catch (Exception ex)
69                     {
70                         _logger.LogError($"BCHGRND: Loop critical error. {ex.Message}", ex);
71                         await apiLoggingService.LogAsync($"BCHGRND: Critical error occurred. {ex.Message}");
72                         throw;
73                     }
74                 }
75
76                 _logger.LogInformation("BCHGRND: Data fetching process ended");
77                 Console.WriteLine("=====");
78
79                 _triggerPeriodMinutes = _configuration.GetValue("BackgroundService:TriggerPeriodMinutes", 140);
80                 await Task.Delay(TimeSpan.FromMinutes(1), stoppingToken);
81             }
82         }
83         catch (Exception ex)
84         {
85             _logger.LogError($"BCHGRND: Critical error. {ex.Message}", ex);
86             _logger.LogInformation("BCHGRND: Data fetching process ended");
87             Console.WriteLine("=====");
88         }
89     }
90 }
91
92 }
```

Зразок профілю користувача, створеного за допомогою ASP.NET Identity

Система моніторингу вагонів

Домашня сторінка

Форма 1892

Форма 8858

Журнал API

Вітаємо, admin@admin.com!

Вийти із системи

Manage your account

Change your account settings

Profile

Email

Password

Two-factor authentication

Personal data

Profile

Thank you for confirming your email.

Username

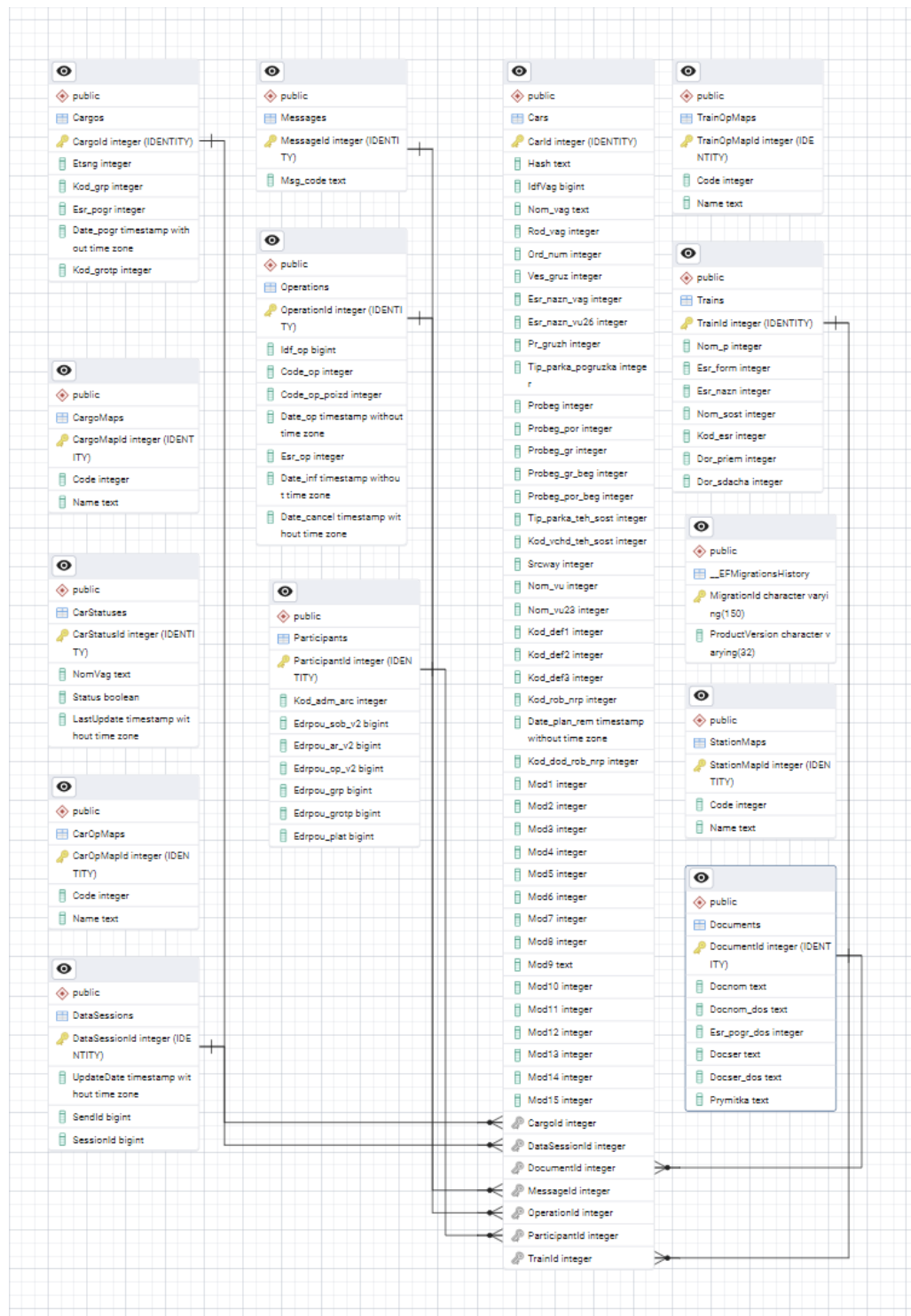
admin@admin.com

Phone number

Save

ДОДАТОК В

Діаграма бази даних



ДОДАТОК Г

Описова таблиця ризиків

Найменування ризику	Причина виникнення	Умови виникнення	Наслідки	Збитки та завдана шкода
Невідповідність розробленої функціональності до вимог	Брак комунікації розробників із замовниками	Наявність нечітко сформульованих вимог на стороні замовника	Некоректний, нефункціональний продукт на виході	Затримка в термінах завершення, трудові витрати
Нереалістичні терміни	Хибна оцінка початкових термінів виконавцем; Нерозуміння складності задачі замовником	Відсутність конструктивного проговорення умов кваліфікованими сторонами	Протерміновані дедлайни	Затримка в термінах завершення, трудові витрати
Недієздатність сторонніх API сервісів	Помилки на стороні підрядників	Ненадійність сторонніх API сервісів	Відсутність безперебійного функціонування, неможливість функціонування загалом	Повна відміна проекту, трудові, бюджетні та часові витрати
Недієздатність фізичних серверів	Збої в системі енергоживлення, пошкодження фізичної інфраструктури	Хостинг на фізичних серверах	Неможливість експлуатації продукту працівниками	Бюджетні, трудові витрати від переходу на інше середовище
Складність системи для персоналу	Невідповідність кваліфікації персоналу та вимог системи	Відсутність досвіду персоналу з подібними системами	Довга крива навчання, подовження періоду впровадження	Трудові витрати
Оверінженіринг	Надмірне впровадження технологій та функцій розробником	Внесення непотрібних та складних функцій розробником	Складність у використанні продукту	Трудові витрати
Помилки в роботі з даними	Нерозуміння суті даних виконавцем замовлення	Наявність комплексних та складних даних без відповідної документації	Некоректне відображення даних, ненадійність системи	Трудові витрати

Недостатня продуктивність системи	Погано налаштована система	Велика кількість даних для обробки	Довге провантаження даних, критичний спад в ефективності використання системи	Затримка в термінах завершення, трудові витрати
Неправильно розрахована потенційна вартість хостингу	Брак досвіду, брак розуміння кінцевих властивостей даних	Хостинг на серверах хмарних провайдерів	Вихід за межі бюджету	Бюджетні витрати
Ризики пов'язані з доступністю даних (внаслідок безпекової ситуації)	Кризова безпекова ситуація у країні, вразливість даних	Наявність гострої кризової безпекової ситуації	Неможливість застосування продукту	Повна відміна проекту, трудові, бюджетні та часові витрати
Витік даних	Хиби в протоколах безпеки при розробці	Наявність вразливих даних	Компрометовані логістичні ланцюги	Потенційна зміна логістичних ланцюгів, втрата клієнтів, безпекові ризики для вагонного парку
Невідповідність останній версії стороннього API	Брак комунікації зі стороною підрядника	Оновлення стороннього API	Необхідність в додатковій корекції та розробці	Затримка в термінах завершення, трудові витрати
Відсутність засобів захисту від кібератак	Недбалість розробника	Наявність цінних даних та зловмисників, які можуть бути в них зацікавлені	Компрометовані логістичні ланцюги, збитки пов'язані з потенційним шантажем	Потенційна зміна логістичних ланцюгів, втрата клієнтів, безпекові ризики для вагонного парку, бюджетні витрати
Відсутність засобів резервного відновлення	Недбалість розробника	Відсутність можливостей відновлення даних ззовні системи	Втрата цінних даних	Трудові та бюджетні витрати на відновлення
Застарілість довідників, що використовуються в програмі	Нестача надійних джерел актуальних даних	Використання довідників для обробки даних	Некоректні дані на виході, ненадійність системи	Трудові витрати

ДОДАТОК Г

Таблиця розрахунку рангу ризиків

Найменування ризику	Ймовірність	Наслідки	Ранг ризику
Невідповідність розробленої функціональності до вимог	4	2	8
Нереалістичні терміни	3	2	6
Недієздатність сторонніх API сервісів	3	3	9
Недієздатність фізичних серверів	2	2	4
Складність системи для персоналу	1	2	2
Оверінженіринг	1	2	2
Помилки в роботі з даними	2	3	6
Недостатня продуктивність системи	1	2	2
Неправильно розрахована потенційна вартість хостингу	3	4	12
Ризики пов'язані з доступністю даних (внаслідок безпекової ситуації)	2	3	6
Витік даних	1	5	5
Невідповідність останній версії стороннього API	2	2	4
Відсутність засобів захисту від кібератак	1	5	5
Відсутність засобів резервного відновлення	1	3	3
Застарілість довідників, що використовуються в програмі	4	1	4

ДОДАТОК Д

Зведена таблиця ризиків

Опис ризиків проекту «Впровадження веб-додатку для моніторингу логістики»								
Помилки в роботі з даними	Оверінженіринг	Складність системи для персоналу	Недіздатність фізичних серверів	Недіздатність сторонніх API сервісів	Нереалістичні терміни	Невідповідність розробленої функціональності до вимог	Найменування ризику	
Нерозуміння суті даних виконавцем замовлення	Надмірне впровадження технологій та функцій розробником	Невідповідність кваліфікації персоналу та вимог системи	Збої в системі енергоживлення, пошкодження фізичної інфраструктури	Помилки на стороні підрядників	Хибна оцінка початкових термінів виконавцем; Нерозуміння складності задачі замовником	Брак комунікації розробників із замовниками	Причина виникнення	
Некоректне відображення даних, ненадійність системи	Складність у використанні продукту	Довга крива навчання, подовження періоду впровадження	Неможливість експлуатації продукту працівниками	Відсутність безперебійного функціонування, неможливість функціонування загалом	Протерміновані дедлайни	Некоректний, нефункціональний продукт на виході	Наслідки	
2	1	1	2	3	3	4	Ймовірність	
Кінець терміну	Кінець терміну	Кінець терміну	Етап експлуатації	Від етапу створення MVP до етапу експлуатації	Кінець терміну	Кінець терміну	Можливі терміни настання	
Внутрішні (організаційні)	Внутрішні (організаційні)	Внутрішні (організаційні)	Внутрішні (технічні)	Зовнішні	Внутрішні (організаційні)	Внутрішні (організаційні)	Категорія ризику	
Помірний	Вкрай низький	Вкрай низький	Низький	Високий	Помірний	Високий	Вплив ризику	
3	2	2	2	3	2	2	Наслідки	
6	2	2	4	9	6	8	Ранг ризику	

Застарілість довідників, що використовуються в програмі	Відсутність засобів резервного відновлення	Відсутність засобів захисту від кібератак	Невідповідність останньої версії стороннього API	Витік даних	Ризики пов'язані з доступністю даних (внаслідок безпекової)	Неправильно розрахована потенційна вартість хостингу	Недостатня продуктивність системи
Нестача надійних джерел актуальних даних	Недбалість розробника	Недбалість розробника	Брак комунікації зі стороною підрядника	Хиби в протоколах безпеки при розробці	Кривозна безпекова ситуація у країні, вразливість даних	Брак досвіду, брак розуміння кінцевих властивостей даних	Погано налаштована система
Некоректні дані на виході, ненадійність системи	Втрата цінних даних	Компрометовані логістичні ланцюги, збитки пов'язані з потенційним шантажем	Необхідність в додатковій корекції та розробці	Компрометовані логістичні ланцюги	Неможливість застосування продукту	Вихід за межі бюджету	Довге провантаження даних, критичний спад в ефективності використання системи
4	1	1	2	1	2	3	1
Від етапу створення MVP до етапу експлуатації	Від етапу створення MVP до етапу експлуатації	Від етапу створення MVP до етапу експлуатації	Від етапу створення MVP до етапу експлуатації	Від етапу створення MVP до етапу експлуатації	Від етапу створення MVP до етапу експлуатації	Етап експлуатації	Від етапу створення MVP до етапу експлуатації
Зовнішні	Внутрішні (технічні)	Зовнішні	Зовнішні	Внутрішні (технічні)	Зовнішні	Внутрішні (організаційні)	Внутрішні (технічні)
Низький	Вкрай низький	Помірний	Низький	Помірний	Помірний	Вкрай високий	Вкрай низький
1	3	5	2	5	3	4	2
4	3	5	4	5	6	12	2



Звіт подібності

метадані

Назва організації

Kyiv National Economic University named after Vadym Hetman KNEU

Заголовок

Розроблення веб-платформи для транспортних задач

Автор

Науковий керівник / Експерт

Микитюк Максим ОлександровичКрячок О.С.

підрозділ

кафедра інформаційних систем в економіці

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.



КП 1

25

Довжина фрази для коефіцієнта подібності 2



КП 2

14266

Кількість слів



КЦ

112133

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		1
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		1

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Колір тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Колір тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://kneu.edu.ua/files/news/news_image/2019_POLOJENNYA_PRO_KVALFKATSYN_ROBOTI.pdf	23 0.16 %
2	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	20 0.14 %
3	https://ela.kpi.ua/bitstream/123456789/58806/1/Kostrova_bakalavr.pdf	11 0.08 %

4	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	10 0.07 %
5	Чуківський Микола Створення онлайн-магазину за допомогою технології ASP.NET 6/17/2024 Professional college of communication and informatization of State University of Intellectual Technologies and Communications (Professional college of communication and informatization of State University of Intellectual Technologies and Communications)	8 0.06 %
6	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	6 0.04 %

з бази даних RefBooks (0.00 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-----------	--

з домашньої бази даних (0.25 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Інформаційна система аналізу CEO показників сайту 6/22/2020 Kyiv National Economic University named after Vadym Hetman KNEU (кафедра інформаційних систем в економіці)	36 (3) 0.25 %

з програми обміну базами даних (0.06 %)

ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	Чуківський Микола Створення онлайн-магазину за допомогою технології ASP.NET 6/17/2024 Professional college of communication and informatization of State University of Intellectual Technologies and Communications (Professional college of communication and informatization of State University of Intellectual Technologies and Communications)	8 (1) 0.06 %

з Інтернету (0.24 %)

ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)
1	https://kneu.edu.ua/files/news/news_image/2019_POLOJENNYA_PRO_KVALFKATSYN_ROBOTI.pdf	23 (1) 0.16 %
2	https://ela.kpi.ua/bitstream/123456789/58806/1/Kostrova_bakalavr.pdf	11 (1) 0.08 %

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)
------------------	-------	---------------------------------------

УДК 004.42

JEL Classification: M1, O1, O3

Максим МИКИТЮК

*здобувач інституту інформаційних технологій в економіці,
Київський національний економічний університет імені Вадима Гетьмана,*

Олександр КРЯЧОК

*к.т.н., доцент кафедри інформаційних систем в економіці
Київський національний економічний університет імені Вадима Гетьмана*

Maksym MYKYTIUK

*Applicant of the Institute of Information Technologies in Economics,
Kyiv National Economic University named after Vadym Getman,
maksym.mykytiuk@kneu.ua*

Alexandr KRIACHOK

*Candidate of technical sciences, docent,
Department of Information Systems in Economics,
Kyiv National Economic University named after Vadym Getman
kriachok.oleksandr@kneu.edu.ua*

ІНФОРМАЦІЙНИЙ ВЕБ-ЗАСТОСУНОК ДЛЯ МОНІТОРИНГУ ЛОГІСТИКИ

WEB-BASED INFORMATION APPLICATION FOR LOGISTICS MONITORING

Анотація. Розглянуто теоретичні аспекти розробки та впровадження інформаційних веб-застосунків для моніторингу логістики. Досліджено специфічні задачі, які виникають в процесі розробки застосунків, запропоновано оптимальні рішення для них. Запропоновано перспективні напрямки та сфери для подальших досліджень.

Ключові слова: інформаційні системи, логістика, автоматизація процесів, інновації.

Abstract. The theoretical aspects of the development and implementation of web-based information applications for logistics monitoring are considered. Investigated the specific tasks that arise in the process of application development, and optimal solutions for them are proposed. Promising directions and areas for further research are suggested.

Keywords: information systems, logistics, process automation, innovations.

Розвиток інформаційних технологій невинно збільшує кількість інновацій та рішень, що спроможні суттєво покращити та доповнити різні сфери діяльності. Зокрема, інформаційні системи в логістиці здатні вирішити цілий ряд задач, починаючи від ефективності роботи окремих працівників і закінчуючи якістю вихідних даних та покращеною безпекою вразливої інформації.

З іншого боку, логістика є критично важливою сферою для підприємств, адже ефективне управління постачаннями, транспортуванням і складськими операціями безпосередньо впливає на їхню конкурентоспроможність. Втім, сучасні логістичні процеси стикаються з численними викликами, зокрема через великий обсяг даних, які часто розпорошені між різними джерелами. Наявність надійної інформаційної системи дозволяє централізувати дані, мінімізувати ризики помилок і підвищити загальну ефективність управління логістикою. Використання баз даних та автоматизації не лише спрощує обробку

інформації, а й створює основу для застосування штучного інтелекту та аналітики, що відкриває додаткові можливості для оптимізації процесів.

Основна проблема моніторингу логістики полягає в необхідності уніфікації великої кількості джерел інформації. Дані надходять із різних систем — ERP, APS, GPS-трекерів, внутрішніх звітів — і їх потрібно консолідувати, щоб забезпечити коректну аналітику. Крім того, обробка значних обсягів інформації вимагає оптимізації, адже нераціональне використання ресурсів може призвести до перевантаження системи. Важливим фактором є й безперебійність роботи інфраструктури, особливо в умовах війни, коли доступ до даних має бути гарантованим 24/7. При цьому безпека виходить на перший план, оскільки витік інформації може становити загрозу не лише для бізнесу, а й для людей. Щоб система була ефективною, вона має бути зручною для кінцевого користувача — диспетчерів, менеджерів та інших фахівців, які не є технічними експертами. Інтуїтивний інтерфейс мінімізує помилки та спрощує роботу з даними.

Аналіз праць науковців [1, 2, 3] дозволив виокремити ключові аспекти розвитку інформаційних систем для моніторингу логістики. Зокрема, дослідження [1] висвітлює виклики у впровадженні веб-моніторингу, пов'язані з якістю та стандартизацією даних. Інтеграція різних інформаційних систем, таких як ERP і APS, залишається складним завданням, що впливає на точність аналітики та ухвалення рішень. Крім того, повільне впровадження технологій RFID і GPS-трекінгу обмежує можливості відстеження логістичних процесів у реальному часі. Робота [2] акцентує увагу на ключових напрямках застосування ІТ у логістиці, зокрема на автоматизації документообігу, інтеграції інформаційних потоків та управлінні транспортними ресурсами. Використання сучасних технологій дозволяє значно підвищити швидкість обробки даних, оптимізувати маршрути перевезень і покращити якість управлінських рішень. Впровадження таких систем також сприяє зменшенню експлуатаційних витрат і підвищенню ефективності транспортних підприємств. Окремо підкреслюється важливість логістики як ключового елемента економічного розвитку, що забезпечує ефективний рух товарів, послуг та інформації. Використання інформаційних технологій дозволяє оптимізувати процеси, мінімізувати витрати та підвищити конкурентоспроможність компаній. Водночас дослідження [3] вказує на необхідність удосконалення логістичних інформаційних систем, оскільки існуючі рішення часто є недостатньо гнучкими і не завжди ефективно реагують на кризові ситуації. Впровадження концепції глобальної транспортної інформаційної системи (GTIS) розглядається як перспективний напрямок, що передбачає інтеграцію різних видів перевезень, автоматизацію процесів та застосування передових технологій, таких як IoT, AI та Big Data.

Для розробки ефективного веб-застосунку для моніторингу логістики необхідно використовувати продуктивні та надійні технології. Оптимальним вибором є платформа .NET, яка забезпечує масштабованість, високу продуктивність і підтримку сучасних інструментів розробки. Велика спільнота користувачів і широкий вибір бібліотек спрощують процес створення та тестування системи. Для роботи з даними доцільно використовувати PostgreSQL — потужну реляційну базу даних з ефективною системою індексації. Її відкритий код і відсутність ліцензійних витрат дозволяють мінімізувати фінансові затрати, а можливості оптимізації запитів забезпечують швидку обробку великих обсягів інформації.

З метою забезпечення стабільної роботи системи та її безперебійного функціонування доцільно використовувати хмарну інфраструктуру Microsoft Azure. Вона гарантує високу доступність, автоматичне масштабування та має налагоджену інтеграцію з .NET, що спрощує розгортання і адміністрування веб-застосунку. Для отримання та оновлення даних варто впровадити фонові сервіси, які працюватимуть через HTTP-запити. Це особливо важливо у випадках, коли зовнішні API, наприклад у залізничній логістиці, є нестабільними або обмеженими. Автоматизований збір інформації забезпечить актуальність даних і підвищить точність моніторингу.

Перспективи подальших досліджень охоплюють кілька ключових напрямів. Одним із найважливіших є застосування штучного інтелекту для автоматизації бізнес-аналітики та звітності. Машинне навчання може виявляти тренди, аномалії та закономірності в логістичних даних, допомагаючи ухвалювати обґрунтовані рішення. Проте якість вхідних даних залишається критичним фактором, що потребує подальших досліджень, оскільки неточна або неповна інформація може спотворювати результати аналізу.

Ще один перспективний напрям — інтеграція з мобільними додатками для сповіщень. Це дозволить оперативно інформувати користувачів про зміни в логістичних процесах і забезпечить миттєвий доступ до даних у реальному часі. Така функціональність покращить швидкість реакції на критичні події та сприятиме підвищенню ефективності управління логістикою. Також важливою є інтеграція з картографічними сервісами, що дасть змогу візуалізувати маршрути, оптимізувати логістичні ланцюги та відстежувати вантажі. Це підвищить зручність використання веб-застосунку й надасть аналітичні інструменти для точнішого планування перевезень.

Література

1. Wood L., Reiners T., Pahl J. Manufacturing and Logistics Information Systems. Encyclopedia of Information Science and Technology. 2015. С. 5136–5144. URL: <https://doi.org/10.4018/978-1-4666-5888-2.ch507>.
2. Використання інформаційних технологій обробки даних в сучасних системах транспортної логістики / Л. М. Дегтярьова та ін. Сучасні інформаційні технології у сфері безпеки та оборони. 2020. Т. 37. С. 139–144. URL: <https://doi.org/10.33099/2311-7249/2020-37-1-139-144>.
3. Чобіток В., Літвінчик С. Системи інформаційного забезпечення транспортної логістики в підприємницькій діяльності. Herald of Khmelnytskyi National University. Economic Sciences. 2024. Т. 332. С. 14–21. URL: <https://doi.org/10.31891/2307-5740-2024-332-2>.

References

1. Wood L., Reiners T., Pahl J. Manufacturing and Logistics Information Systems. Encyclopedia of Information Science and Technology. 2015. С. 5136–5144. URL: <https://doi.org/10.4018/978-1-4666-5888-2.ch507>.
2. The use of information technologies for data processing in modern transport logistics systems / L.M. Degtyareva et al. Modern information technologies in the field of security and defense. 2020. Т. 37. С. 139–144. URL: <https://doi.org/10.33099/2311-7249/2020-37-1-139-144>.
3. Chobitok V., Litvinchuk S. Systemy informatiinoho zabezpechennia transportnoi lohistyky v pidpriemnytskii diialnosti. Economic Sciences. 2024. Т. 332. С. 14–21. URL: <https://doi.org/10.31891/2307-5740-2024-332-2>.

UDC 004.42

JEL Classification: M1, O1, O3

Maksym MYKYTIUK

*Applicant of the Institute of Information Technologies in Economics,
Kyiv National Economic University named after Vadym Getman,
maksym.mykytiuk@kneu.ua*

Alexandr KRIACHOK

*Candidate of technical sciences, docent,
Department of Information Systems in Economics,
Kyiv National Economic University named after Vadym Getman
kriachok.oleksandr@kneu.edu.ua*

Максим МИКИТЮК

*здобувач інституту інформаційних технологій в економіці,
Київський національний економічний університет імені Вадима Гетьмана,*

Олександр КРЯЧОК

*к.т.н., доцент кафедри інформаційних систем в економіці
Київський національний економічний університет імені Вадима Гетьмана*

WEB-BASED INFORMATION APPLICATION FOR LOGISTICS MONITORING

ІНФОРМАЦІЙНИЙ ВЕБ-ЗАСТОСУНОК ДЛЯ МОНІТОРИНГУ ЛОГІСТИКИ

Abstract. The theoretical aspects of the development and implementation of web-based information applications for logistics monitoring are considered. Investigated the specific tasks that arise in the process of application development, and optimal solutions for them are proposed. Promising directions and areas for further research are suggested.

Keywords: information systems, logistics, process automation, innovations.

Анотація. Розглянуто теоретичні аспекти розробки та впровадження інформаційних веб-застосунків для моніторингу логістики. Досліджено специфічні задачі, які виникають в процесі розробки застосунків, запропоновано оптимальні рішення для них. Запропоновано перспективні напрямки та сфери для подальших досліджень.

Ключові слова: інформаційні системи, логістика, автоматизація процесів, інновації.

The development of information technology is steadily increasing the number of innovations and solutions that can significantly improve and complement various areas of activity. In particular, information systems in logistics can solve a number of problems, ranging from the efficiency of individual employees to the quality of output data and improved security of vulnerable information.

On the other hand, logistics is a critical area for businesses, as efficient management of supply, transportation, and warehouse operations directly affects their competitiveness. However, modern logistics processes face numerous challenges, in particular due to the large amount of data that is often scattered among different sources. Having a reliable information system allows you to centralize data, minimize the risk of errors, and increase the overall efficiency of logistics management. The use of databases and automation not only simplifies information processing, but

also creates the basis for the application of artificial intelligence and analytics, which opens up additional opportunities for process optimization.

The main problem with logistics monitoring is the need to unify a large number of information sources. Data comes from different systems - ERP, APS, GPS trackers, internal reports - and needs to be consolidated to provide correct analytics. In addition, the processing of large amounts of information requires optimization, as irrational use of resources can lead to system overload. The uninterrupted operation of the infrastructure is also an important factor, especially in times of war, when access to data must be guaranteed 24/7. At the same time, security comes to the fore, as information leakage can pose a threat not only to business but also to people. For the system to be effective, it must be convenient for the end user - dispatchers, managers, and other professionals who are not technical experts. An intuitive interface minimizes errors and simplifies data management.

An analysis of the works of scientists [1, 2, 3] allowed us to identify key aspects of the development of information systems for monitoring logistics. In particular, study [1] highlights the challenges in implementing web-based monitoring related to data quality and standardization. Integration of different information systems, such as ERP and APS, remains a challenge that affects the accuracy of analytics and decision-making. In addition, the slow adoption of RFID and GPS tracking technologies limits the ability to track logistics processes in real time. Paper [2] focuses on the key areas of IT application in logistics, in particular, document flow automation, integration of information flows, and transport resource management. The use of modern technologies can significantly increase the speed of data processing, optimize transportation routes, and improve the quality of management decisions. The introduction of such systems also helps to reduce operating costs and increase the efficiency of transport companies. The article emphasizes the importance of logistics as a key element of economic development that ensures the efficient movement of goods, services and information. The use of information technology allows to optimize processes, minimize costs and increase the competitiveness of companies. At the same time, study [3] points to the need to improve logistics information systems, as existing solutions are often not flexible enough and do not always respond effectively to crisis situations. The introduction of the concept of a global transportation information system (GTIS) is seen as a promising direction that involves the integration of different types of transportation, process automation, and the use of advanced technologies such as IoT, AI, and Big Data.

To develop an effective web-based application for logistics monitoring, you need to use productive and reliable technologies. The best choice is the .NET platform, which provides scalability, high performance, and support for modern development tools. A large community of users and a wide selection of libraries simplify the process of creating and testing the system. To work with data, it is advisable to use PostgreSQL, a powerful relational database with an efficient indexing system. Its open source code and the absence of license costs minimize financial expenses, and query optimization capabilities ensure fast processing of large amounts of information. To ensure stable operation of the system and its uninterrupted functioning, it is advisable to use the Microsoft Azure cloud infrastructure. It guarantees high availability, automatic scaling, and has a well-established integration with .NET, which simplifies the deployment and administration of the web application. To receive and update data, it is worth implementing background services that will work through HTTP requests. This is especially important in cases where external APIs, for example, in railway logistics, are unstable or limited. Automated information collection will ensure that the data is up-to-date and increase the accuracy of monitoring.

Prospects for further research cover several key areas. One of the most important is the use of artificial intelligence to automate business intelligence and reporting. Machine learning can identify trends, anomalies, and patterns in logistics data, helping to make informed decisions. However, the quality of the input data remains a critical factor that requires further research, as inaccurate or incomplete information can distort the results of the analysis.

Another promising area is integration with mobile applications for notifications. This will allow users to promptly inform about changes in logistics processes and provide instant access to real-time data. This functionality will improve the speed of response to critical events and help increase the efficiency of logistics management. Integration with mapping services is also important, as it will allow visualizing routes, optimizing supply chains, and tracking cargo. This will improve the usability of the web application and provide analytical tools for more accurate transportation planning.

References

1. Wood L., Reiners T., Pahl J. Manufacturing and Logistics Information Systems. Encyclopedia of Information Science and Technology. 2015. C. 5136-5144. URL: <https://doi.org/10.4018/978-1-4666-5888-2.ch507>.
2. The use of information technologies for data processing in modern transport logistics systems / L.M. Degtyareva et al. Modern information technologies in the field of security and defense. 2020. T. 37. C. 139-144. URL: <https://doi.org/10.33099/2311-7249/2020-37-1-139-144>.
3. Chobitok V., Litvinchuk S. Systemy informatiinoho zabezpechennia transportnoi lohistyky v pidpriemnytskii diialnosti. Economic Sciences. 2024. T. 332. C. 14-21. URL: <https://doi.org/10.31891/2307-5740-2024-332-2>.

Література

1. Wood L., Reiners T., Pahl J. Manufacturing and Logistics Information Systems. Encyclopedia of Information Science and Technology. 2015. C. 5136–5144. URL: <https://doi.org/10.4018/978-1-4666-5888-2.ch507>.
2. Використання інформаційних технологій обробки даних в сучасних системах транспортної логістики / Л. М. Дегтярьова та ін. Сучасні інформаційні технології у сфері безпеки та оборони. 2020. Т. 37. С. 139–144. URL: <https://doi.org/10.33099/2311-7249/2020-37-1-139-144>.
3. Чобіток В., Літвінчук С. Системи інформаційного забезпечення транспортної логістики в підприємницькій діяльності. Herald of Khmelnytskyi National University. Economic Sciences. 2024. Т. 332. С. 14–21. URL: <https://doi.org/10.31891/2307-5740-2024-332-2>.

ДО ПИТАННЯ ПРО МОНІТОРИНГ ЛОГІСТИКИ

Стрімкий розвиток інформаційних технологій постійно вносить нові інновації та рішення, які здатні суттєво трансформувати та вдосконалити різні галузі економіки. У сфері логістики інформаційні системи дозволяють вирішувати широкий спектр завдань – від підвищення продуктивності праці окремих співробітників до поліпшення якості вихідної інформації та зміцнення захисту конфіденційних даних.

Водночас, логістика залишається стратегічно важливою галуззю для компаній, оскільки якісне управління постачанням, перевезеннями й складськими операціями безпосередньо впливає на рівень їхньої конкурентоспроможності. Втім, нині існує низка проблем, таких як обробка великих обсягів даних, що часто зберігаються у розрізнених джерелах, або перебої з їхньою доступністю. Надійна інформаційна система дозволяє централізувати інформацію, зменшити ймовірність помилок і підвищити загальну ефективність управлінських процесів. Завдяки базам даних та автоматизації спрощується обробка, а також з'являється можливість інтеграції штучного інтелекту та аналітики для подальшої оптимізації.

Ключовою складністю в організації моніторингу логістики є необхідність уніфікації численних джерел надходження даних, таких як ERP, APS, GPS-трекерів, внутрішньої звітності. При цьому виникає потреба консолідації для забезпечення якісної аналітики. Окрім того, обробка великих обсягів даних потребує раціонального підходу, адже неефективне використання ресурсів може призвести до перевантаження систем. Іншим важливим аспектом є безперервна робота інфраструктури, особливо в умовах військового стану, коли доступ до даних повинен бути безперервним. Безпека при цьому має пріоритетне значення, оскільки витік інформації може становити загрозу не лише для бізнесу, а й для життя людей.

Аналіз досліджень [1 - 3] дає змогу додатково акцентувати увагу на важливості оптимізації логістичних процесів, користі цифрових технологій та ролі моніторингу логістики для прийняття рішень. У дослідженні [1] зазначається, що впровадження оптимізованих логістичних підходів дає змогу підприємствам оперативно адаптуватися до змін на ринку та підтримувати стабільність постачань. Це, у свою чергу, зміцнює їхню фінансову стійкість і забезпечує довгострокову прибутковість. Досягнення логістичної оптимізації значною мірою забезпечується завдяки використанню інформаційних технологій, зокрема систем ERP, MRP, DRP тощо, які дозволяють автоматизувати й інтегрувати процеси, що сприяє покращенню прогнозування, ефективнішому управлінню запасами та кращій координації в ланцюгах постачання, в підсумку призводячи до зниження витрат і зростання загальної ефективності операцій. Дослідження [2] наголошує на критичній ролі цифрових технологій у підвищенні ефективності логістики та важливості впровадження інтегрованих інформаційних систем. Ігнорування ІТ-рішень може призвести до збоїв у постачанні та втрати конкурентних переваг. Натомість, у дослідженні [3] розглядається важливість моніторингу як інструменту, що забезпечує ухвалення обґрунтованих рішень на базі достовірної інформації. Логістика зображається як складна система, де цифрові технології та інформаційне забезпечення (це зокрема IoT) мають вирішальне значення для її ефективного функціонування. Відсутність уніфікованої методики ускладнює управління логістичними сервісами, особливо в умовах швидких змін на ринку. Слід підкреслити необхідність створення надійної інформаційної основи,

використання статистичних даних і сучасних аналітичних методів для моніторингу. Без інтеграції інформаційних потоків, автоматизації та постійного аналізу даних ефективне функціонування логістики неможливе.

Одним із ключових рішень для підвищення ефективності логістичного моніторингу є впровадження цифрових інструментів, що дозволяють автоматизувати процеси, інтегрувати розрізнені джерела даних та забезпечити аналітичну підтримку управлінських рішень. У складній структурі логістичних ланцюгів важливо не лише отримувати достовірну інформацію в режимі реального часу, а й мати можливість її оперативно обробляти та використовувати для прогнозування. Саме тому створення уніфікованої системи з підтримкою технологій штучного інтелекту, IoT та аналітики стає стратегічним завданням. Таке рішення дозволяє зменшити кількість помилок, мінімізувати ризики та своєчасно реагувати на зміни ринкової ситуації.

Розробка високопродуктивного веб-застосунку, здатного забезпечити централізований доступ до актуальної інформації з різних джерел у режимі реального часу, може стати актуальним рішенням для малого та середнього бізнесу. За рахунок масштабованості, його розробка не вимагає залучення таких інвестицій як розгортання комплексної системи, та залишає простір для подальшого вдосконалення й інтеграції додаткових модулів при виникненні відповідних запитів. Веб-застосунок дозволить автоматизувати обробку даних, візуалізувати логістичні процеси, швидко виявляти відхилення та формувати аналітичні звіти. Додатково, підтримка мобільних пристроїв і систем сповіщень сприятиме оперативному реагуванню на зміни, підвищуючи контроль над логістичними операціями.

Майбутні дослідження мають охоплювати низку важливих напрямів, один із ключових – це впровадження штучного інтелекту для автоматизації бізнес-аналітики та звітності. Технології машинного навчання дозволять виявляти закономірності, аномалії та тренди у логістичних даних, що сприятиме прийняттю обґрунтованих управлінських рішень. Водночас, якість вхідної інформації залишається критичним аспектом, адже неточні чи фрагментарні дані можуть суттєво вплинути на достовірність результатів аналізу. Це питання потребує додаткового вивчення. Ще один перспективний напрям – інтеграція з мобільними застосунками для оперативних сповіщень. Це дозволить своєчасно інформувати користувачів про зміни в логістичних процесах і надавати миттєвий доступ до даних у режимі реального часу. Не менш важливою є інтеграція з картографічними сервісами, яка дає можливість візуалізувати маршрути, оптимізувати логістичні ланцюги та контролювати переміщення вантажів. Це не лише підвищує зручність роботи з веб-застосунком, а й забезпечує аналітичні можливості для точнішого планування перевезень.

Список використаних джерел

1. Ляліна Н.С., Панчук А.С., Матвієнко-Біляєва Г.Л. Впровадження сучасних методів логістики в підприємницькій діяльності. 2020. DOI: <https://doi.org/10.32840/2522-4263/2020-2-20>.
2. Кривов'язюк І. Оцінювання ефективності впровадження цифрових технологій в сфері управління логістикою підприємств. *Collection of Scientific Papers «SCIENTIA»*. 2023. С. 34–38. URL: <https://previous.scientia.report/index.php/archive/article/view/703>.
3. Белова І., Мушеник І. Інформаційно-аналітичне забезпечення дослідження ринку логістичних послуг. *Інститут бухгалтерського обліку, контроль та аналіз в умовах глобалізації*. 2021. С. 7–23. URL: <https://doi.org/10.35774/ibo2021.01-02.007>.

Науковий керівник: Крячок О. С., к.т.н., доцент.