МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет

імені Петра Могили

Факультет комп'ютерних наук

Кафедра комп'ютерної інженерії

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри,
д-р техн. наук, проф.

_____ І. М. Журавська

« __ » _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

Система логістичного моніторингу на базі Arduino, ODB-II для стивідорної компанії

Спеціальність 123 Комп'ютерна інженерія

123 – КБР.01 – 405.22010514

Студент _____ О. Ю. Новіков

підпис

«__» _____ 202__ р.

Керівник канд. фіз.-мат. наук, доцент _____ С. В. Пузирьов

підпис

«__» _____ 202__ р.

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра комп'ютерної інженерії

ЗАТВЕРДЖУЮ

Зав. кафедри _____ І. М. Журавська

«_____» _____ 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної бакалаврської роботи

Видано студенту групи 405 факультету комп'ютерних наук

Новіков Олександр Юрійович
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

Система логістичного моніторингу на базі Arduino, ODB-II для стивідорної компанії

Затверджена наказом по ЧНУ ім. Петра Могили від 30.01.2024 № 17.

2. Строк представлення кваліфікаційної роботи «____» _____ 20__ р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні
Розроблена система логістичного моніторингу на базі Arduino. Спеціальний девайс для працівників стивідорних компаній

4. Перелік питань, що підлягають розробці
Аналіз праці стивідорних компаній, їх різновиди. Які технології застосовують стивідори. Порівняльний аналіз існуючих технічних рішень. Вибір компонентів необхідних для реалізації системи. Реалізація СЛМ.

5. Перелік графічних матеріалів
Блок-схема занесення даних у СЛМ
Блок-схема зміни даних мітки контейнеру
Каркаси сайту

6. Завдання до спеціальної частини
Розрахунок освітлення в кімнаті для забезпечення належних умов охорони праці

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Макарова Олена Валеріївна	Кафедри екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Керівник роботи
доцент кафедри КІ, Пузирьов Сергій Володимирович
(посада, прізвище, ім'я, по батькові)

(підпис)

Завдання прийнято до виконання
Новіков Олександр Юрійович
(прізвище, ім'я, по батькові студента)

(підпис)

Дата видачі завдання «____» _____ 20_____ р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: Система логістичного моніторингу на базі Arduino, ODB-II для стивідорної компанії

№	Найменування роботи	Початок	Закінчення	Примітки
1	Розробка та затвердження завдання на виконання КР	30.01.24	31.01.24	Виконав
2	Огляд літератури за темою роботи	05.02.24	12.02.24	Виконав
3	Складання календарного плану БКР	16.02.24	26.02.24	Виконав
4	Аналіз предметної області	17.02.24	05.03.24	Виконав
5	Розробка проєктних рішень	06.03.24	08.03.24	Виконав
6	Моделювання структури системи логістичного моніторингу та конструкції девайсу на базі Arduino	08.03.24	22.03.24	Виконав
7	Конструювання системи логістичного моніторингу, перевірка її працездатності	24.04.24	11.06.24	Виконав
8	Оформлення БКР та презентації	07.05.24	14.06.24	Виконав
9	Перший предзахист	28.05.24	28.05.24	Виконав
10	Другий предзахист	04.06.24	04.06.24	Виконав
11	Завершення оформлення КР та презентації	14.06.24	14.06.24	Виконав
12	Захист бакалаврської кваліфікаційної роботи	.06.24	.06.24	Виконав

Розробив здобувач ВО ____Новіков Олександр Юрійович _____

(прізвище, ім'я, по батькові) *(підпис)*

«____» _____ 2024 р.

Керівник роботи доцент кафедри КІ, Пузирьов Сергій Володимирович _____

(посада, прізвище, ім'я, по батькові) *(підпис)*

«____» _____ 2024 р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи
«Система логістичного моніторингу на базі Arduino,
ODB-II для стивідорної компанії»
Студент 405 гр.: Новіков Олександр Юрійович
Керівник: канд. фіз.-мат. наук, доцент Пузирьов Сергій Володимирович

Актуальність: Розробка системи логістичного моніторингу є актуальною для підвищення ефективності стивідорних компаній, забезпечуючи контроль над вантажопотоками та оптимізуючи логістичні процеси.

Об'єкт дослідження: Процеси логістичного моніторингу у стивідорній компанії.

Предмет дослідження: Система логістичного моніторингу на базі Arduino.

Мета: Розробка та впровадження системи логістичного моніторингу для стивідорної компанії, яка забезпечить відстеження вантажів, збір даних про їх переміщення та стан, а також візуалізацію цієї інформації у вебзастосунку.

Практична значимість: Впровадження розробленої системи дозволить стивідорній компанії підвищити ефективність логістичних операцій, зменшити витрати на моніторинг та покращити якість обслуговування клієнтів.

В результаті проведеної роботи було розроблено вебзастосунок з використанням Next.js та Tailwind, який забезпечує зручну візуалізацію даних про переміщення та стан вантажів. Далі, було розроблено унікальний пристрій на базі Arduino, що дозволяє збирати дані про вантажі з використанням технології RC522. Цей пристрій інтегровано з розробленим вебзастосунком.

Кваліфікаційна робота складається з: 59 с., 24 рис., 4 табл., 25 джерел посилання, 2 додатки.

Ключові слова: *Next.js, React, Typescript, HTTPS-запити, NodeJS, Arduino, мікроконтролер, моніторинг, RC522.*

ABSTRACT
of the Bachelor's Thesis
"Logistics monitoring system based on Arduino,
ODB-II for a stevedoring company"
Student: Novikov Alexander
Supervisor: Puzyrev Sergey

Relevance: The development of a logistics monitoring system is relevant for increasing the efficiency of stevedoring companies by providing control over cargo flows and optimizing logistics processes.

Object of research: Logistics monitoring processes in a stevedoring company.

Subject of research: Arduino-based logistics monitoring system.

Objective: Development and implementation of a logistics monitoring system for a stevedoring company that will provide cargo tracking, collection of data on their movement and condition, and visualization of this information in a web application.

Practical significance: The implementation of the developed system will allow the stevedoring company to increase the efficiency of logistics operations, reduce monitoring costs and improve the quality of customer service.

As a result of the work carried out, a web application was developed using Next.js and Tailwind, which provides convenient visualization of data on the movement and condition of goods. Next, we developed a unique Arduino-based device that allows collecting cargo data using RC522 technology. This device is integrated with the developed web application.

The qualification work consists of: 59 p., 24 figs., 4 tables, 25 references, 2 appendices.

Keywords: Next.js, React, Typescript, HTTPS requests, NodeJS, Arduino, microcontroller, monitoring, RC522.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних
МД – мобільний девайс
ПК – персональний комп’ютер
СВК – система відстеження контейнерів
СЛМ – система логістичного моніторингу
СУТ – система управління терміналом

HTTPS – Hypertext Transfer Protocol Secure
IoT – Internet of Things
JSON – JavaScript Object Notation
npm – Node Package Manager
RFID – Radio Frequency Identification
SPI – Serial Peripheral Interface
UI/UX – User Interface/User Expirience
Wi-Fi – Wireless Fidelity

ВСТУП

Світ логістики динамічно розвивається, сягаючи $15.5 трлн. до 2025 року. Стивідорні компанії, що стикаються з жорсткою конкуренцією, змушені шукати шляхи оптимізації та економії. Система логістичного моніторингу (СЛМ) на базі Arduino, ODB-II пропонує рішення, яке може суттєво покращити їхню діяльність.

Система моніторингу дозволяє оптимізувати переміщення контейнерів, завантаження/розвантаження суден, відстеження вантажів, управління запасами. Arduino – доступний та простий у використанні, гнучкий та масштабований. ODB-II – стандартизований протокол, що забезпечує сумісність з широким колом транспортних засобів та легкість доступу до даних про стан та експлуатацію техніки.

На сучасному ринку існує безліч мікроконтролерів, які дозволяють людям розробляти власні системи відправлення інформації або керування пристроями. Одним з таких мікроконтролерів є Arduino, популярна торгівельна марка, яка давно зайняла своє місце на ринку. Через це створення такого девайсу на цій базі є доступним та легко адаптивним до нового функціоналу.

Також на ринку не існує готових рішень, які б повноцінно відповідали потребам стивідорних компаній. Існуючі системи моніторингу часто дорогі, складні у використанні та не пропонують необхідного набору функцій. Тому СЛМ на базі Arduino, ODB-II є актуальною та перспективною розробкою, яка може допомогти стивідорним компаніям суттєво покращити свою діяльність та отримати конкурентні переваги.

Мета кваліфікаційної бакалаврської роботи: розроблення СЛМ для стивідорної компанії на базі Arduino.

Поставлена мета вимагає вирішення таких **задач**:

1) аналіз потреб стивідорної компанії;
2) аналіз існуючих СЛМ;

3) визначення вимог до розроблюваної СЛМ;

4) визначення комплектації електронних компонентів;

5) проєктування та розробка СЛМ

6) проєктування та розробка апаратної частини логістичного моніторингу.

Об'єкт дослідження: СЛМ для стивідорної компанії.

Предмет дослідження: створення СЛМ з використанням OBD-2 для стивідорної компанії.

Практичне значення отриманих результатів полягає у розробці унікального девайсу та СЛМ для працівників стивідорних компаній, які полегшать їх працю.

1 АНАЛІЗ ПРОЦЕСУ ПРАЦІ СТИВІДОРНИХ КОМПАНІЙ ТА СЛМ

1.1 Теоретичні відомості про стивідорів.

Стивідорні компанії – це компанії, які займаються навантаженням і розвантаженням вантажів у портах. Це стосується всіх вантажів, що перевозяться морем [2, 23].

Спочатку існувала професія стивідора, відповідального за нагляд за перевалкою вантажів у портах. Зі збільшенням товарообігу в усьому світі зросла і галузь, і зараз існують повні стивідорні компанії з великою кількістю працівників. Ця професія залишається і сьогодні, але вже в рамках всієї компанії. Тому, окрім контролю за вантажно-розвантажувальними операціями, стивідори також тісно співпрацюють із відділом судноплавства порту для оформлення супровідних документів для роботи. Відповідальність за розробку плану робіт несе стивідорна компанія.

Як правило, стивідор бере на себе повну відповідальність за виконання загального плану робіт і дотримання його технічного регламенту. Стивідори мають право відсторонити працівників, якщо виявлено порушення ними трудового розпорядку. Він контролює графік обробки судна, несе відповідальність за позапланові простої вантажного обладнання та відстежує час стоянки. Він зобов'язаний погоджувати з капітаном усі поточні питання щодо ходу робіт на судні. Після закінчення зміни стивідори звітують диспетчеру зміни вантажної зони [13].

В залежності від умов фрахтового договору стивідорна компанія може бути найнятою фрахтувальником (власником) вантажу або судновласником. У другому випадку вартість стивідорних операцій розраховують за фрахтовою ставкою.

1.2 Види стивідорних компаній

В основному компанії поділяють на такі види:

За спеціалізацією стивідорні компанії можна поділити на такі типи:

– компанія «Універсал» – надає послуги з будь-яких відправлень (приймають будь які вантажі);

– спеціалізовані – обслуговування тільки певних вантажів (насипні вантажи, лісоматеріали, контейнерів тощо).

За кількістю вантажно-розвантажувального обладнання виділяються такі компанії:

– наявність власного обладнання;

– користуються орендованим обладнанням.

За умовами найму портових працівників підприємства:

– мають постійну бригаду докерів;

– не має власних докерів (наймають їх для виконання конкретних робіт);

– деякі докери працюють постійно (вантажні майстри, виконроби, висококваліфіковані спеціалісти), інші наймаються на тимчасовій основі.

Право вибору стивідорної компанії зазвичай належить стороні, яка оплачує вантажні операції. Це може бути і судновласник, і вантажовласник (фрахтувач судна). Сторона, яка оплачує стивідорні витрати, визначається договором купівлі-продажу та договором морського перевезення [4].

Приватні судноплавні компанії, як правило, належать окремим особам або корпораціям і працюють на комерційній основі, зосереджуючись на максимізації прибутку. Вони мають більш гнучку організаційну структуру і можуть швидко адаптуватися до змін ринку та вимог клієнтів. Державні перевантажувальні компанії, у свою чергу, належать уряду та можуть мати інші пріоритети, такі як забезпечення стратегічних інтересів держави, підтримка національної безпеки чи підтримка економічного розвитку регіону. Вони часто отримують державне фінансування і можуть мати доступ до

додаткових ресурсів і підтримки, але іноді прийняття рішень відбувається повільніше через бюрократичні процеси. Вибір між приватною або державною транспортною компанією залежить від конкретних потреб клієнта та умов, викладених у контракті [25].

1.3 Функції стивідорних підприємств

Окрім навантаження та розвантаження вантажів, стивідорні компанії виконують багато інших важливих функцій, таких як:

– планування та координація. стивідорна компанія розробляє плани вантажно-розвантажувальних робіт, координує дії різних учасників процесу (екіпаж, портові служби, вантажники) і забезпечує синхронізацію всіх систем;

– підготовка документів: стивідори готують необхідні документи, пов'язані з вантажно-розвантажувальними операціями, такі як вантажні декларації, акти прийому-передачі, коносаменти тощо;

– заходи безпеки: вантажники та розвантажувачі несуть відповідальність за дотримання правил техніки безпеки та ГіБП під час вантажно-розвантажувальних операцій;

– інші послуги: Стивідорні компанії можуть надавати додаткові послуги, такі як зберігання вантажів, страхування, митне оформлення тощо.

Стивідорні компанії постійно вдосконалюють свої технології та методи роботи для максимально ефективної та безпечної обробки вантажів. Вони також співпрацюють із судноплавними компаніями, портами та іншими учасниками транспортно-логістичного ланцюга, щоб забезпечити безперебійне функціонування усієї системи.

1.4 Вплив СЛМ на роботу стивідорних компаній

Впровадження СЛМ дає стивідорним компаніям ряд переваг, які сильно впливають на ефективність праці. По перше йде підвищення ефективності. Системи моніторингу дозволяють відстежувати переміщення вантажів у

режимі реального часу, оптимізувати маршрути та мінімізувати час простою [24].

По-друге, це зниження витрат. За допомогою кращого планування та контролю можна зменшити витрати на паливо, електроенергію та інші ресурси. Також покращується безпека усього робочого процесу. Системи моніторингу допомагають виявляти та усувати потенційні небезпечні ситуації, що сприяє зниженню ризику травм і аварій.

Підсумовуючи, можна відзначити підвищення якості послуг. Завдяки доступності інформації про місцезнаходження та стан вантажів у режимі реального часу стивідорні компанії можуть надати своїм клієнтам кращий сервіс і виділитися серед конкурентів.

1.4.1 Типи СЛМ актуальні для стивідорів

Стивідорні компанії використовують різні типи СЛМ для відстеження та оптимізації своїх операцій. Деякі з найбільш поширених типів систем включають:

– система відстеження контейнерів (СВК): СВК використовується для відстеження розташування та стану контейнерів у всьому ланцюжку поставок. Це може допомогти стивідорним компаніям підвищити операційну ефективність і видимість;

– системи управління терміналом (СУТ): СУТ використовується для керування операціями терміналу, такими як відправлення суден, прийом і випуск контейнерів, а також керування транспортуванням вантажів. Це може допомогти стивідорним компаніям збільшити пропускну здатність терміналу та скоротити час простою;

– системи автоматичного розпізнавання номерних знаків (ANPR): ANPR використовується для автоматичного розпізнавання номерних знаків вантажівок і причепів. Це може допомогти стивідорним компаніям прискорити процес прийому та видачі вантажу;

– Radio Frequency Identification (RFID): RFID використовується для відстеження контейнерів та інших вантажів за допомогою радіочастотних міток. Це допомагає стивідорним компаніям підвищити видимість вантажу та запобігти втраті або крадіжці [3];

– сенсорні системи: датчики можна використовувати для збору даних про такі фактори, як температура, вологість і вібрація контейнера. Ці дані можна використовувати для моніторингу стану відправлення та запобігання пошкодженням.

Крім цих типів систем, стивідорні компанії також можуть використовувати інші логістичні технології, такі як програмне забезпечення для управління ланцюгами поставок (SCM) і системи планування ресурсів підприємства (ERP).

Основні переваги використання перерахованих систем включають підвищення ефективності та підвищення видимості. СЛМ може допомогти стивідорним компаніям підвищити ефективність роботи за рахунок скорочення часу простою, зменшення кількості помилок і прискорення обробки вантажів, а також допоможе стивідорним компаніям отримати кращу видимість у своїх ланцюжках поставок, допомагаючи їм приймати кращі рішення та реагувати на проблеми.

1.4.2 Виклики впровадження СЛМ

Впровадження СЛМ може значно підвищити ефективність будь-якого бізнесу, що оперує в логістичній сфері. Однак, як і будь-яка інноваційна технологія, вона має свої проблеми, які необхідно враховувати.

Впровадження СЛМ потребує значних інвестицій, включаючи закупівлю обладнання, програмного забезпечення, послуги з монтажу, навчання персоналу та подальше обслуговування. Їх може бути важко налаштувати та використовувати, особливо для підприємств, які не мають досвіду використання подібних технологій.

Іншою проблемою є інтеграція з існуючими системами, де СЛМ має інтегруватися з існуючими бізнес-системами, такими як системи управління запасами та системи планування маршрутів. Існуючі бізнес-процеси також можуть потребувати змін, що може призвести до опору співробітників.

Підіймається питання конфідеційності інформації. СЛМ, які збирають дані про переміщення та роботу транспортних засобів та співробітників, повинні діяти в рамках чинного законодавства про захист персональних даних та мати стабільну структуру, яка не допускає витоку інформації або доступу сторонніх осіб.

1.5 Аналіз існуючих СЛМ

Сьогодні доступно багато СЛМ, які можуть допомогти стивідорним компаніям покращити свою діяльність [17]. Серед найпоширеніших СЛМ у всесвіті можна виділити:

– **Navis N4**: ця система надає широкий спектр можливостей для моніторингу та керування вантажно-розвантажувальними операціями, включаючи відстеження контейнерів, планування маршруту, керування чергами та оптимізацію використання крана. Navis N4 використовується багатьма великими стивідорними компаніями по всьому світу;

– **Trimble Transportation Enterprise**: ця система надає комплексне рішення для моніторингу та керування транспортними засобами, включаючи відстеження GPS, відстеження вантажу, аналіз продуктивності водія та керування паливом. Багато стивідорних компаній використовують Trimble Transport Enterprise для моніторингу своїх транспортних засобів і вантажів;

– **JDA Warehouse Management**: ця система забезпечує функції управління запасами та складом, включаючи відстеження контейнерів, керування розміщенням, контроль запасів і планування завантаження. JDA Warehouse Management використовується деякими стивідорними компаніями для управління своїми складами та запасами;

– **OpenGTS**: ця платформа GPS-відстеження з відкритим кодом надає широкий спектр можливостей моніторингу транспортних засобів, включаючи GPS-відстеження, відстеження вантажу, аналіз продуктивності водія та керування паливом. Деякі стивідорні компанії використовують OpenGTS для моніторингу своїх транспортних засобів і вантажів;

– **Traccar**: ця платформа GPS-відстеження з відкритим кодом пропонує простий у використанні інтерфейс і широкі можливості моніторингу транспортних засобів, включаючи GPS-відстеження, відстеження вантажу, аналіз продуктивності водія та керування паливом. Деякі стивідорні компанії використовують Traccar для моніторингу своїх транспортних засобів і вантажів;

– **Log4j**: ця бібліотека журналювання з відкритим кодом надає гнучкі та потужні можливості журналювання для програм Java. Деякі СЛМ використовують Log4j для реєстрації подій і помилок.

Таблиця 1.1 – Порівняння систем СЛМ

Система	Функції	Ціна	Складність впровадження	Відгуки користувачів
Navis N4	Широкий спектр функцій	Висока	Складна	Позитивні
Trimble Transportation Enterprise	Моніторинг транспортних засобів та вантажів	Середня	Середня	Позитивні
JDA Warehouse Management	Управління запасами та складами	Висока	Складна	Змішані
OpenGTS	Широкий спектр функцій	Безкоштовна	Середня	Позитивні
Traccar	Простий у використанні інтерфейс	Безкоштовна	Проста	Позитивні
Log4j	Журнування	Безкоштовна	Проста	Позитивні

Після ретельного порівняння різних СЛМ стивідорні компанії можуть вибрати ту, яка найкраще відповідає їхнім потребам.

1.6 Технологічні тренди у стивідорній діяльності

Сучасні стивідорні компанії активно впроваджують новітні технології для підвищення ефективності та безпеки своїх операцій. Із основних технологічнів трендів, що впливають на стивідорну діяльність можна виділити наступні технології.

1.6.1 Автоматизація та роботизація

Автоматизація процесів завантаження та розвантаження контейнерів значно знижує час виконання операцій та мінімізує людський фактор, що зменшує ризик помилок та підвищує продуктивність. Використання роботизованих систем для переміщення вантажів дозволяє оптимізувати простір складу та знизити витрати на ручну працю. Автоматичні крани та транспортні засоби стають невід'ємною частиною сучасних терміналів, забезпечуючи безперебійну роботу та високу точність виконання завдань.

Крім того, автоматизовані системи можуть працювати 24/7 без потреби у відпочинку, що дозволяє значно підвищити швидкість обробки вантажів та зменшити час їх перебування на терміналі. Завдяки цьому підвищується пропускна здатність портів, що є критично важливим для ефективного управління вантажопотоками та задоволення потреб клієнтів.

1.6.2 Використання дронів та безпілотних технологій

Застосування дронів на контейнерних терміналах значно оптимізує процеси інспекції та моніторингу. Вони дозволяють швидко оцінювати стан контейнерів, виявляти пошкодження та забезпечувати візуальний контроль у важкодоступних зонах. Безпілотні наземні транспортні засоби, оснащені сенсорами та камерами, автоматично переміщують вантажі та виконують інші операції, зменшуючи потребу в ручній праці та підвищуючи безпеку.

Використання дронів сприяє зниженню витрат на інспекцію та охорону, забезпечуючи при цьому високу точність та оперативність отримання інформації. Крім того, вони можуть працювати в небезпечних умовах або за несприятливої погоди, що робить їх незамінним інструментом у сучасній логістиці.

1.6.3 Роль IoT у стивідорній діяльності

Internet of Things (IoT) об'єднує різні пристрої та системи в одну мережу, забезпечуючи безперервний обмін даними та централізоване управління. Це дозволяє відстежувати місцезнаходження та стан контейнерів, контролювати роботу приладів та отримувати аналітичні дані в режимі реального часу, що дозволяє оперативно реагувати на будь-які зміни та оптимізувати процеси.

Завдяки IoT компанії можуть передбачити потребу в обслуговуванні обладнання, уникаючи незапланованих простоїв і підвищуючи надійність системи. Інтеграція IoT з іншими технологіями, такими як блокчейн і штучний інтелект, відкриває нові можливості для автоматизації та оптимізації логістичних процесів, підвищення ефективності та конкурентоспроможності підприємств[5, 9].

1.6.4 Використання великих даних та аналітики

Аналіз великих даних допомагає стивідорним компаніям отримувати цінні інсайти з великих обсягів інформації, зібраної під час роботи. Використання аналітичних платформ дозволяє виявляти тренди, прогнозувати попит, оптимізувати логістичні маршрути та покращувати управління запасами. Це сприяє прийняттю більш обґрунтованих рішень та підвищує конкурентоспроможність компаній [7].

Аналітика великих даних також допомагає виявити потенційні проблеми до того, як вони стануть критичними, дозволяючи вживати запобіжних заходів і уникати простоїв або затримок. У результаті компанії можуть покращити обслуговування клієнтів, зменшити витрати на обробку вантажів і підвищити

ефективність своєї діяльності. Крім того, великі дані дозволяють точніше прогнозувати зміни ринку та адаптувати стратегії управління до нових умов, що є важливим фактором у динамічному логістичному середовищі.

1.6.5 Впровадження блокчейн-технологій

Блокчейн-технології знаходять все ширше застосування у сфері логістики завдяки своїй здатності забезпечувати прозорість та безпеку транзакцій. Використання блокчейну у стивідорній діяльності дозволяє відстежувати ланцюжок поставок, забезпечувати автентичність документів та знижувати ризики шахрайства. Це підвищує довіру між учасниками ланцюга поставок та покращує ефективність управління логістичними операціями.

Деякі стивідорні компанії вже успішно впровадили технології блокчейну для оптимізації своєї діяльності. Наприклад, порти Роттердама та Антверпена використовують блокчейн для відстеження контейнерних перевезень і спрощення митних процедур. Це дозволяє значно скоротити час перевірки документів і знизити логістичні витрати [20].

Висновки до розділу 1

Стивідорні компанії відіграють важливу роль у сучасному ланцюжку поставок, забезпечуючи завантаження та розвантаження вантажів у портах. Їхня робота стає дедалі складнішою через збільшення обороту та жорстку конкуренцію.

Впровадження СЛМ може допомогти судноплавним компаніям значно підвищити ефективність роботи. СЛМ можуть допомогти оптимізувати планування та координацію вантажно-розвантажувальних робіт, відстежувати переміщення вантажів у режимі реального часу, зменшити час простою та покращити безпеку.

Існує багато різних типів СЛМ, кожен з яких має свої власні функції та переваги. При виборі СЛМ стивідорні компанії повинні враховувати свої конкретні потреби та бюджет.

Деякі з найпоширеніших СЛМ, які використовуються стивідорними компаніями, включають Navis N4, Trimble Transportation Enterprise, JDA Warehouse Management, OpenGTS та Traccar.

Впровадження СЛМ може принести багато переваг стивідорним компаніям, але воно також пов'язане з певними викликами. До них належать високі витрати на впровадження, складність інтеграції з існуючими системами та необхідність навчання персоналу.

Незважаючи на ці виклики, СЛМ стають все більш важливим інструментом для стивідорних компаній, які прагнуть залишатися конкурентоспроможними на сучасному ринку.

2 КОНЦЕПТ СЛМ НА БАЗІ ARDUINO, ODB-II ДЛЯ СТИВІДОРНОЇ КОМПАНІЇ

2.1 Опис орієнтовочної компанії

СЛМ на основі Arduino та ODB-II розробляється з урахуванням потреб стивідорних компаній, які використовують контейнери. Це означає, що система буде зосереджена на зборі та аналізі даних, пов’язаних з переміщенням і завантаженням контейнерів, а також на забезпеченні ефективного управління логістичними процесами на вантажних платформах.

Вимоги до СЛМ базуються на конкретних обставинах контейнерної роботи стивідорної компанії та враховують наступні аспекти:

– типи контейнерів: система повинна підтримувати різні типи та розміри контейнерів, що використовуються в стивідорній діяльності;

– логістичні процеси: система повинна забезпечувати моніторинг та аналіз ключових логістичних процесів, таких як завантаження контейнерів, розвантаження, переміщення та зберігання;

– інтеграція з існуючими системами: система повинна мати можливість інтегруватися з існуючими системами керування стивідорним терміналом, такими як СУТ і СВК.

Таким чином, впровадження такої системи значно підвищить ефективність і точність роботи стивідорних компаній. Забезпечення інтеграції з існуючими системами скоротить час впровадження та адаптації нової технології. Крім того, гнучкість і масштабованість системи дозволить їй легко адаптуватися до змін вимог і умов роботи. Це створює можливості для подальшого розвитку та вдосконалення логістичних процесів, що сприятиме загальному підвищенню конкурентоспроможності компанії на ринку.

2.2 Функціональні вимоги до СЛМ

СЛМ на базі Arduino та ODB-II буде реалізована у вигляді вебзастосунку з функціональними можливостями, описаними нижче.

1) Відображення даних про контейнери:

Вебінтерфейс буде використовувати дані, отримані з пристроїв Arduino через Hypertext Transfer Protocol Secure-запити (HTTPS) до бекенду, для візуалізації інформації про контейнери, включаючи:

- ідентифікаційний номер контейнера (RFID-мітка);
- тип та розмір контейнера;
- місцезнаходження контейнера;
- статус завантаження/розвантаження;
- дата та час завантаження/розвантаження;
- інші релевантні дані;

2) **Звіти та аналітика:**

- вебінтерфейс буде надавати можливості для збору звітів та проведення аналітики даних, зібраних системою;
- аналітика даних допоможе користувачам виявити проблеми, оптимізувати логістичні процеси та приймати обґрунтовані управлінські рішення [1];

3) Інші функції:

- вебінтерфейс буде мати інтуїтивно зрозумілий та зручний інтерфейс користувача;
- система буде доступна через будь-який веббраузер на персональний комп’ютер (ПК) або мобільний девайс (МД).

Ці функції значно покращать управління контейнерами та забезпечать прозорість логістичних процесів. Користувачі зможуть отримувати точні та актуальні дані в режимі реального часу, що сприятиме швидкому прийняттю рішень. Інтуїтивно зрозумілий інтерфейс і доступність через різні пристрої зроблять систему зручною у використанні широкому колу користувачів.

Загалом впровадження такої системи дозволить оптимізувати робочі процеси та підвищити ефективність роботи експедиторських компаній.

2.3 Опис структури СЛМ

СЛМ на базі Arduino та ODB-II – це складний програмно-апаратний комплекс, який забезпечує збір, передачу та обробку даних, пов’язаних з контейнерними перевезеннями.

Апаратна частина системи включає пристрій Arduino, який знаходиться у користувача, за допомогою якого він відстежує дані контейнерів, заносить їх у СЛМ та змінює дані контейнерів. Зчитувач RFID ідентифікує кожен контейнер. Крім того на пристрій Arduino можна встановити інші датчики, для швидкої діагностики контейнеру. Наприклад перевірити температуру та вологість усередині контейнера.

Програмна частина СЛМ відповідає за обробку та інтерпретацію зібраних даних. Прошивка пристрою Arduino забезпечує функціональність для збору даних із датчиків, попередньої обробки та передачі на сервер. Сервер розроблено на Python або Node.js і приймає дані від Arduino, зберігає їх у базі даних (PostgreSQL, MySQL або MongoDB) і надає API для взаємодії з вебінтерфейсом. Вебінтерфейс, створений за допомогою React, Angular або Vue.js, забезпечує зручну візуалізацію даних, звітування та аналіз.

Комунікаційна частина СЛМ забезпечує надійну та ефективну передачу даних між компонентами системи. Модуль Wireless Fidelity (Wi-Fi) використовується для передачі даних на короткі відстані, наприклад, в межах складів.

Загалом, інтеграція різноманітних апаратних та програмних компонентів дозволяє створити комплексне рішення, яке забезпечує повний цикл управління контейнерами, від їх ідентифікації до моніторингу стану та аналітики.

2.3.1 Алгоритм зчитування даних

СЛМ використовує RFID для відстеження даних контейнерів на всіх етапах ланцюга поставок. Завдяки RFID, інформація про місцезнаходження, стан та вміст контейнерів автоматично зчитується і передається до централізованої системи.

Рисунок 2.1 – Блок-схема занесення даних у СЛМ

Ця схема забезпечує безперервний потік даних від RFID-міток до користувача через інтегровану систему збору та обробки інформації. Інформація постійно оновлюється, що дозволяє користувачам отримувати точні та актуальні дані в режимі реального часу. Це значно підвищує ефективність управління логістичними процесами та дозволяє оперативно реагувати на будь-які зміни чи проблеми.

2.3.2 Алгоритм запису даних

Зміна даних на RFID-мітці включає кілька етапів, починаючи від встановлення з'єднання зі зчитувачем, аутентифікації для доступу до мітки, і завершуючи записом нових даних. Цей процес забезпечує цілісність та безпеку інформації, що зберігається на RFID-мітці.

Рисунок 2.2 – Блок-схема зміни даних мітки контейнеру

Ця схема дозволяє користувачам легко і швидко змінювати інформацію на RFID-мітках, що важливо для динамічного моніторингу логістичних процесів.

Таким чином, інтеграція технології RFID з вебінтерфейсом і платформою Arduino забезпечує ефективний обмін даними в системі моніторингу логістики, дозволяє швидко реагувати на зміни і забезпечувати точність інформації.

2.4 Каркаси інтерфейсу для ПК та МД

Каркаси інтерфейсу користувача – це схематичні зображення інтерфейсу користувача, які показують розташування основних елементів на сторінці без детального дизайну. Вони служать основою для розробки дизайну і допомагають визначити функціональність і структуру майбутнього інтерфейсу. Каркаси дозволяють розробникам і дизайнерам швидко створювати макети сторінок, обговорювати та вносити зміни в структуру інтерфейсу на ранніх стадіях проєкту.

У цьому розділі представлені інтерфейси користувача для ПК і МД, створені за допомогою Balsamiq [19].

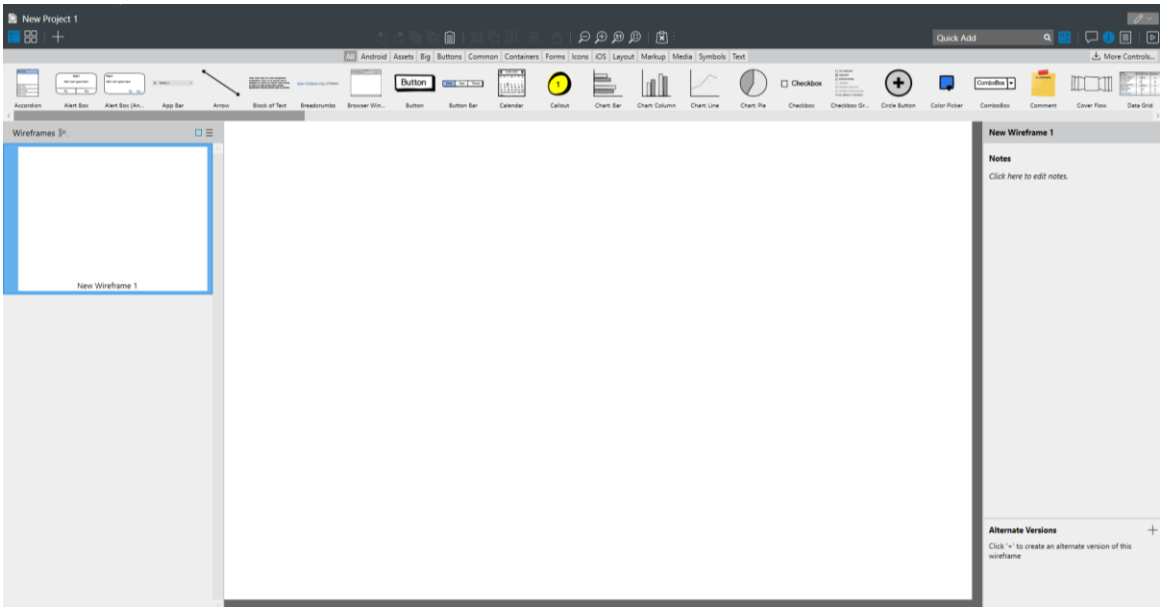


Рисунок 2.3 – Інтерфейс програми Balsamiq Wireframes

Balsamiq – популярний інструмент інтерфейсу користувача, який дозволяє швидко й легко створювати макети вебсайтів і програм. Програма має інтуїтивно зрозумілий інтерфейс і широкий набір готових елементів, таких як кнопки, поля введення, форми, меню та інші компоненти.

2.4.1 Каркас інтерфейсу для сторінки входу на ПК

Забезпечення безпечного доступу до СЛМ починається зі зручної та зрозумілої сторінки входу.

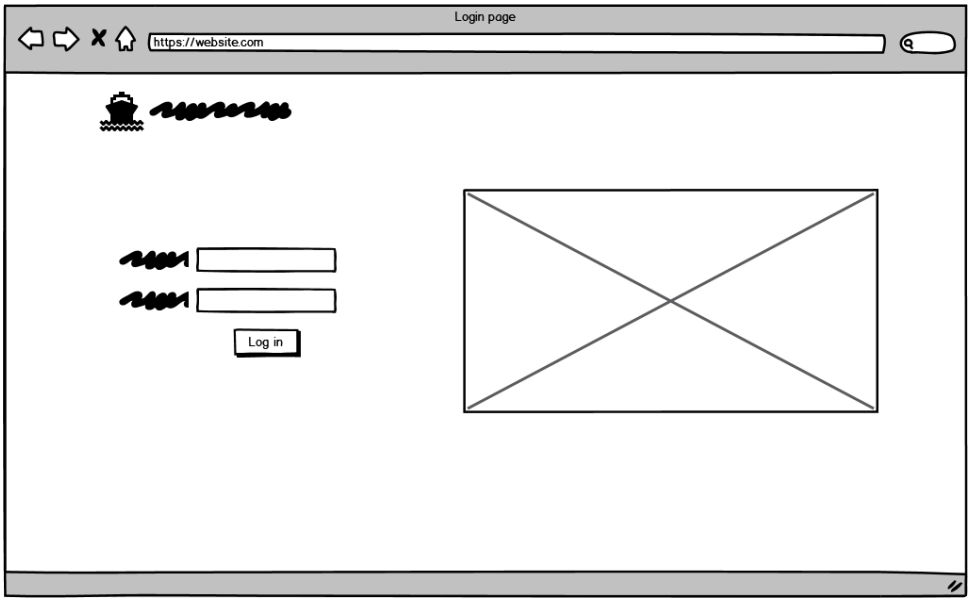


Рисунок 2.4 – Каркас для сторінки входу до СЛМ

На рисунку показаний каркас інтерфейсу для сторінки входу до ЛСМ на ПК версії. Сторінка містить такі основні елементи:

– логотип та назва компанії: Розташовані у верхньому лівому куті сторінки;

– поле введення імені користувача: Розташоване ліворуч під логотипом;

– поле введення пароля: Знаходиться під полем введення імені користувача;

– кнопка входу: Розташована під полем введення пароля;

– місце для банера або важливого повідомлення: Велика область праворуч від полів введення, яка може використовуватися для відображення зображень, новин або інших важливих повідомлень.

Така структура сторінки входу забезпечує простий та інтуїтивно зрозумілий процес авторизації користувачів у системі.

2.4.2 Каркас інтерфейсу для сторінки контейнерів на ПК

Для ефективного керування контейнером потрібен інтуїтивно зрозумілий інтерфейс. Інтерфейс користувача для сторінок-контейнерів на ПК розроблено з урахуванням цих вимог.

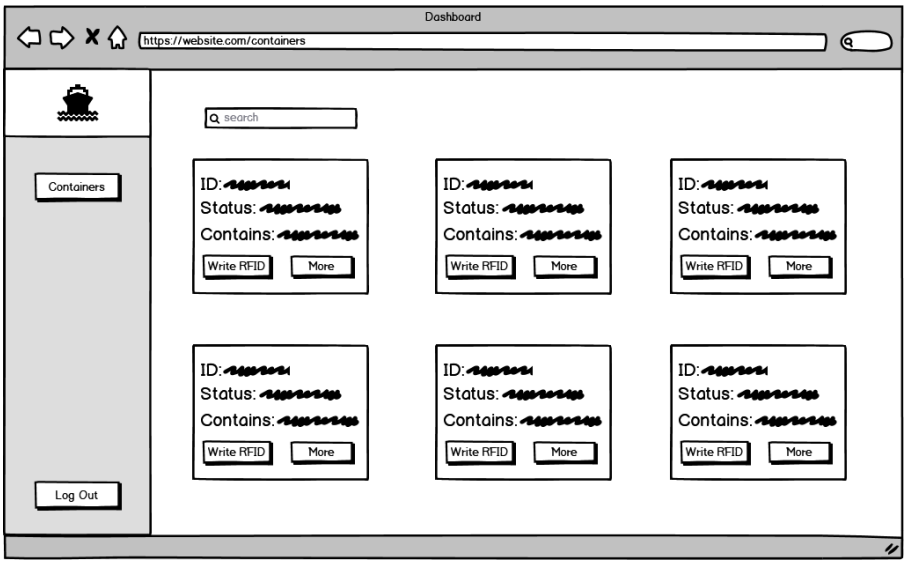


Рисунок 2.5 – Каркас для сторінки контейнерів

На рисунку показаний каркас інтерфейсу для сторінки контейнерів на ПК версії. Сторінка містить такі основні елементи:

– навігаційна панель: Розташована у лівій частині сторінки, містить посилання на інші сторінки або розділи системи. Може у майбутньому містити більше функціоналу;

– список контейнерів. Розташований у центральній частині сторінки, містить інформацію про кожен контейнер, включаючи ідентифікаційний номер, тип, розмір, статус завантаження/розвантаження та місцезнаходження;

– панель пошуку та фільтрації. Знаходиться у верхній частині сторінки, дозволяє користувачам шукати контейнери за різними критеріями (наприклад, за номером, статусом чи місцезнаходженням) і фільтрувати результати;

– кнопки дій. Розташовані на панелях у списку контейнерів. Дозволяють змінити дані на контейнеру або подивитись детальну інформацію.

Така структура забезпечує простоту використання та дозволяє користувачам ефективно взаємодіяти з СЛМ.

2.4.3 Каркас інтерфейсу для сторінки інформації контейнеру на ПК

Детальна інформація про контейнер є ключем до ефективного управління логістикою. Структура інтерфейсу користувача сторінки інформації про контейнер на ПК розроблена з урахуванням цієї вимоги.

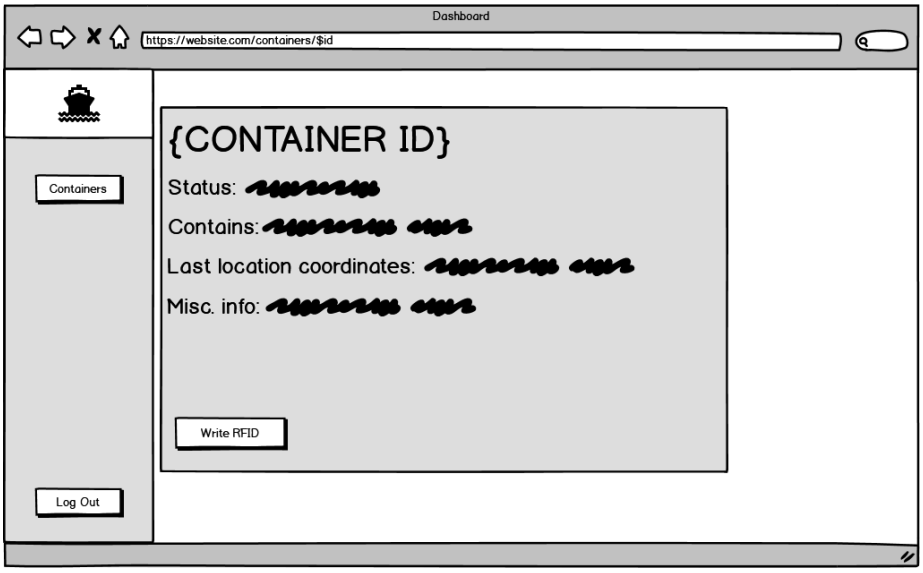


Рисунок 2.6 – Каркас для сторінки інформації контейнеру на ПК

На рисунку представлений каркас інтерфейсу для сторінки детальної інформації про контейнер. Ця сторінка містить такі основні елементи:

– заголовок: Відображає ідентифікатор контейнера (CONTAINER ID);

– блок інформації про контейнер: Містить детальну інформацію про контейнер;

– кнопка "Write RFID": призначена для запису інформації на RFID-мітку контейнера;

Окрім основної інформації, блок інформації про контейнер може включати додаткові дані, такі як історія переміщень і стан контейнера. Ці дані можуть бути корисними для відстеження контейнера, оптимізації логістики та забезпечення безпеки вантажу.

2.4.4 Каркас інтерфейсу для сторінки входу на МД

Каркас інтерфейсу для сторінки входу на МД має подібну структуру, але оптимізований для використання на менших екранах. Основні елементи мобільного інтерфейсу можуть бути описані наступним чином:

– логотип та назва компанії: Розташовані у верхньому положенні сторінки для кращої видимості на МД;

– поле введення логіну: Розташоване у центральній частині сторінки щоб користувач одразу його бачив;

– поле введення пароля: Знаходиться під полем введення логіну;

– кнопка входу: Розташована під полем введення пароля.

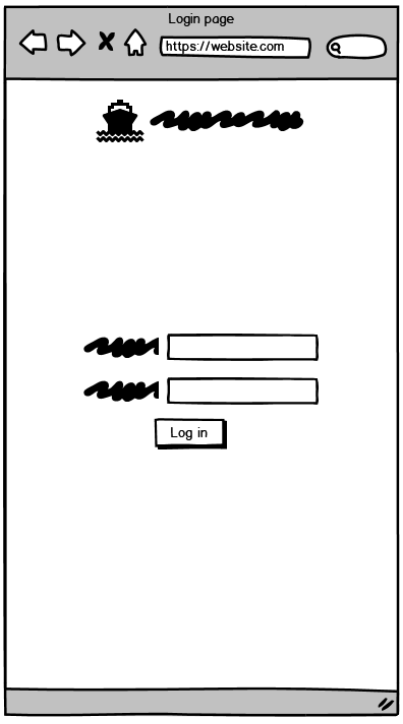


Рисунок 2.7 – Каркас для сторінки входу на МД

На МД прибирається банер на сторінці входу, щоб не загромождати маленький екран користувача. Відсутність банера дозволить зосередити увагу на основних елементах сторінки, таких як поля для введення логіна та пароля, а також кнопка входу. Відсутність банера сприятиме більш зручному та швидкому введенню даних для авторизації та покращить загальний досвід користувача.

2.4.5 Каркас інтерфейсу для сторінки контейнерів на МД

На МД каркас інтерфейсу для сторінки контейнерів має подібну структуру, проте оптимізований для використання на менших екранах.

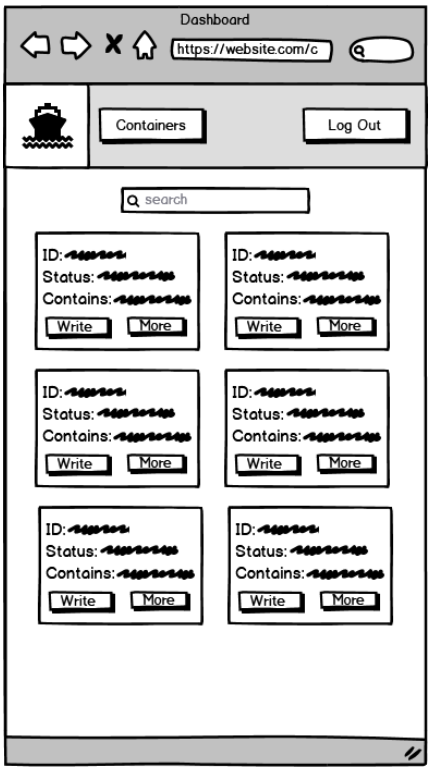


Рисунок 2.8 – Каркас для сторінки контейнерів на МД

Основні елементи сторінки включають:

– навігаційна панель: Розташована у верхній частині екрану для зручного доступу, містить посилання на інші сторінки або розділи системи. Може мати функцію згортатання для економії простору, при наявності більшого функціоналу;

– список контейнерів: Розташований у центральній частині сторінки, відображає інформацію про кожен контейнер.

– панель пошуку та фільтрації: Знаходиться у верхній частині екрану, дозволяє користувачам шукати контейнери за різними критеріями та фільтрувати результати;

– кнопки дій: Розташовані на панелях у списку контейнерів, дозволяють змінювати дані на контейнері або подивитись детальну інформацію.

Такий адаптивний дизайн забезпечує зручність використання системи на МД, дозволяючи користувачам ефективно керувати контейнерами.

2.4.6 Каркас інтерфейсу для сторінки інформації контейнеру на МД

Каркас інтерфейсу для сторінки інформації про контейнер на МД має за своєю основною структурою подібний до версії для ПК.

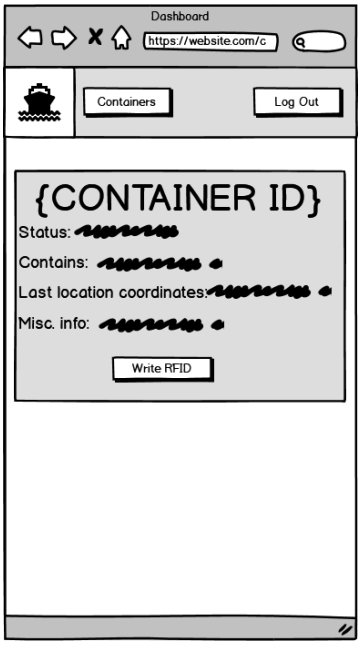


Рисунок 2.9 – Каркас для сторінки інформації контейнеру на МД

Основні елементи сторінки включають:

– заголовок: Відображає ідентифікатор контейнера;

– блок інформації про контейнер: Містить детальну інформацію про контейнер, таку як тип, розмір, стан та історія переміщень.

– кнопка "Write RFID": Призначена для запису інформації на RFID-мітку контейнера.

Такий дизайн забезпечує зручний доступ до детальної інформації про контейнер та можливість оперативно оновлювати дані на RFID-мітці за допомогою МД.

2.5 Безпека та захист даних

СЛМ на базі Arduino буде розроблена з урахуванням вимог безпеки та захисту даних.

По-перше, користувачі повинні будуть проходити аутентифікацію за допомогою імені користувача та пароля або інших методів автентифікації, перш ніж отримувати доступ до системи. Система буде надавати різні рівні доступу для різних користувачів, щоб обмежити доступ до даних лише авторизованими особами.

Крім того, всі дані, що передаються між пристроями Arduino, сервером, вебінтерфейсом і мобільним додатком, будуть зашифровані. Це включатиме дані про місцезнаходження, параметри мітки, дані про контейнери та інші конфіденційні дані.

Дані, що зберігаються на сервері, будуть захищені системами запобігання вторгненням та іншими заходами безпеки. Доступ до сервера буде надано лише авторизованим особам.

Для передачі даних використовуватимуться лише надійні протоколи, такі як HTTPS і TLS [8].

Крім технічних заходів, буде впроваджено політики безпеки, які включатимуть регулярні аудити безпеки, навчання персоналу з питань інформаційної безпеки та планування дій на випадок надзвичайних ситуацій. Будуть проведені тестування на проникнення для виявлення вразливостей та забезпечення високого рівня захисту даних.

Система буде відповідати міжнародним стандартам безпеки інформації, таким як ISO/IEC 27001, та слідувати рекомендаціям OWASP для захисту вебзастосунків.

Таким чином, СЛМ на базі Arduino буде забезпечувати високий рівень безпеки та захисту даних, зменшуючи ризики несанкціонованого доступу та захищаючи конфіденційну інформацію користувачів.

Висновки до розділу 2

У цьому розділі було представлено концепцію СЛМ на базі Arduino та ODB-II, призначеної для стивідорних компаній, що працюють з контейнерами. Було описано основні вимоги до системи, включаючи підтримку різних типів контейнерів, моніторинг логістичних процесів та інтеграцію з існуючими системами управління.

Детально розглянуто функціональні вимоги до СЛМ, такі як відображення даних про контейнери, звіти та аналітика, а також інші функції, що забезпечують зручність використання системи. Описано структуру СЛМ, включаючи апаратну частину (пристрій Arduino, зчитувач RFID, датчики), програмну частину (прошивка Arduino, сервер на Python/Node.js, вебінтерфейс на React/Angular/Vue.js) та комунікаційну частину (модуль Wi-Fi).

Також було представлено алгоритми зчитування та запису даних з використанням RFID-міток, що забезпечують ефективний обмін інформацією в системі. Нарешті, було розглянуто питання безпеки та захисту даних, включаючи аутентифікацію користувачів, шифрування даних, захист сервера та використання надійних протоколів передачі даних.

Додатково до цього, можна зазначити, що було надаго детальний огляд інтерфейсів користувача для різних пристроїв, що були створені за допомогою інструменту Balsamiq. Ці каркаси інтерфейсу служать основою для подальшого розроблення та дизайну інтерфейсу користувача системи.

3 РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

3.1 Вибір технологій для розробки вебзастосунку

Враховуючи специфіку проєкту та вимоги до продуктивності, безпеки та масштабованості, було проведено ретельний аналіз доступних технологій та інструментів. Особливу увагу було приділено стеку технологій, який би забезпечив оптимальне поєднання швидкості розробки, ефективності роботи вебзастосунку та можливості його подальшого розвитку. Вибір було зроблено на користь сучасних, перевірених часом рішень, які добре зарекомендували себе у розробці вебзастосунків різної складності.

3.1.1 Мова програмування та фреймворк

Для розробки вебзастосунку було обрано TypeScript як основну мову програмування. TypeScript є надмножиною JavaScript, що додає статичну типізацію, що дозволяє виявляти помилки на етапі компіляції, покращує читабельність та підтримуваність коду, особливо у великих проєктах.

Next.js було обрано як основний фреймворк для розробки як фронтенду, так і бекенду. Це дозволяє створити єдину, цілісну кодову базу, що полегшує розробку, підтримку та масштабування проєкту.

Із основних переваг використання Next.js можна виділити:

– Server-Side Rendering: Next.js дозволяє рендерити сторінки на сервері, що покращує продуктивність та SEO-оптимізацію сайту;

– API Routes: Вбудована підтримка створення API-маршрутів дозволяє легко розробляти бекенд-логіку;

– Automatic Code Splitting: Next.js автоматично розділяє код на менші частини, що завантажуються лише за необхідності, покращуючи продуктивність.

Next.js надає прості у використанні інструменти для створення компонентів інтерфейсу користувача, маршрутизації, керування станом та

інших можливостей, необхідних для створення сучасних вебзастосунків. React, як бібліотека для створення компонентів, використовується в Next.js для побудови інтерактивних та динамічних інтерфейсів користувача.

3.1.2 Бази даних

Для зберігання даних у системі логістичного моніторингу було обрано NoSQL БД Firebase. Цей вибір обумовлений специфікою даних та вимогами до системи [11].

Firebase забезпечує реальну відсутність сервера та автоматичне масштабування, що дозволяє легко додавати нові функції та змінювати структуру даних у процесі розробки та експлуатації системи. Це особливо важливо для проєктів, де вимоги до даних можуть змінюватися з часом. Firebase легко масштабується, що дозволяє розподілити навантаження на кілька серверів та забезпечити високу продуктивність системи навіть при великих обсягах даних [14].

Firebase оптимізована для роботи з великими обсягами даних та забезпечує швидкий доступ до них. Це важливо для системи логістичного моніторингу, де необхідно оперативно обробляти та відображати дані в реальному часі. З Firebase також легко працювати з мобільними та вебзастосунками, що забезпечує зручний доступ до даних з будь-якого пристрою.

Firebase є оптимальним вибором для системи логістичного моніторингу, оскільки вона забезпечує гнучкість, масштабованість, продуктивність та зручність роботи з реальними даними в реальному часі.

Firebase також забезпечує інтеграцію з іншими сервісами Google, що дозволяє додатково розширити функціональність системи, наприклад, використовуючи аналітику, автентифікацію користувачів та хмарне сховище.

3.1.3 API та протоколи обміну даними

Для забезпечення ефективної та надійної комунікації між вебзастосунком та пристроями Arduino було обрано RESTful API

RESTful API – це архітектурний стиль для розробки вебсервісів, який базується на принципах HTTPS протоколу. Він забезпечує стандартизований спосіб взаємодії між різними компонентами системи, дозволяючи їм обмінюватися даними у форматі JavaScript Object Notation (JSON).

3.1.4 Проєктування бази даних

Оскільки було обрано Firebase як NoSQL базу даних (БД), структура буде гнучкою та адаптивною до потреб системи. Firebase, будучи базою даних в реальному часі, дозволяє зберігати дані у форматі JSON, що робить її схожою на MongoDB у відношенні до гнучкості структури даних. Основні колекції (аналог таблиць у реляційних БД):

1) devices: Зберігає інформацію про підключені пристрої Arduino (ID, тип, статус);

2) containers: Інформація контейнерів (час, тип події, ID пристрою/транспорту, дані).

Така структура забезпечує ефективне зберігання та швидкий доступ до даних про контейнери та пристрої для СЛМ. Крім того, завдяки здатності Firebase працювати в режимі реального часу, дані про статус контейнерів і пристроїв будуть постійно оновлюватися, що забезпечить актуальність інформації для користувачів системи.

3.2 Опис Next.js

Next.js – це потужний фреймворк з відкритим вихідним кодом, розроблений на основі React, який спрощує створення сучасних вебзастосунків. Він поєднує в собі найкращі практики розробки, пропонуючи

готові рішення для маршрутизації, рендерингу, оптимізації та інших важливих аспектів [12]. Із головних особливостей цього фреймворку можна виділити:

– Server-Side Rendering: Next.js дозволяє рендерити сторінки на стороні сервера, що покращує продуктивність, SEO та забезпечує швидке завантаження контенту;

– маршрутизація на основі файлової системи: Структура папок проєкту визначає маршрути додатку, що робить його інтуїтивно зрозумілим та легким у підтримці;

– підтримка CSS Modules та CSS-in-JS: Next.js надає гнучкі інструменти для стилізації компонентів;

– API Routes: Next.js дозволяє легко створювати API-маршрути для обробки запитів на стороні сервера;

– image optimization: Next.js автоматично оптимізує зображення для різних пристроїв та розмірів екрану.

Завдяки своїй гнучкості та інтуїтивно зрозумілій структурі Next.js став популярним вибором для розробки різноманітних проєктів, від простих цільових сторінок до складних корпоративних програм. Його активна спільнота та постійна підтримка забезпечують актуальність і розвиток фреймворку, що робить його одним із лідерів у сфері веброзробки.

3.2.1 Налаштування середовища розробки

Для створення та розробки вебзастосунків за допомогою Next.js необхідно попередньо підготувати середовище розробки. Першим кроком є встановлення Node.js та Node Package Manage (npm) або yarn, оскільки Next.js базується на Node.js та використовує ці інструменти для управління залежностями та виконання скриптів [18].

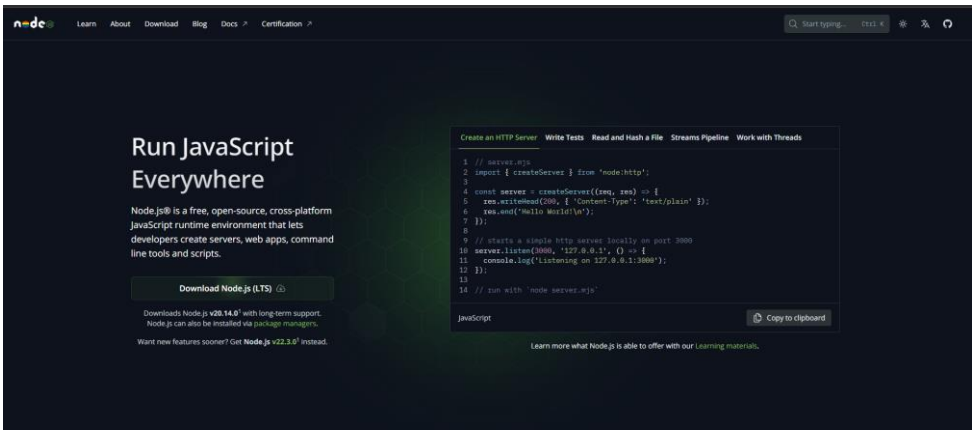


Рисунок 3.1 – Офіційна сторінка Node.js

Після встановлення Node.js та npm або yarn можна приступати до створення нового проєкту Next.js. Це здійснюється за допомогою команди npx create-next-app@latest, яку слід виконати у терміналі. Під час створення проєкту можна обрати шаблон "Dashboard", який надає базову структуру та компоненти для швидкого старту розробки панелі адміністратора.

Отриманий шаблон "Dashboard" можна легко модифікувати під конкретні вимоги проєкту. Це включає в себе зміну стилів, додавання або видалення компонентів, налаштування логіки роботи та інтеграцію з необхідними сервісами чи API. Таким чином, використання шаблону значно скорочує час на початкове налаштування проєкту та дозволяє зосередитися на реалізації основної функціональності додатку.

3.2.2 Структура проєкту Next.js

Next.js використовує специфічну структуру директорій для організації коду та ресурсів проєкту:

– /app: Ця директорія є нововведенням у Next.js 13 і вище. Вона містить компоненти Layout, Page та інші, відповідальні за візуальне представлення та логіку додатку. Компоненти Layout визначають загальну структуру сторінок (навбар, меню, футер), тоді як Page компоненти відповідають за унікальний вміст кожної сторінки;

– /pages: Традиційна директорія, яку можна використовувати для створення сторінок, особливо у проєктах, мігрованих зі старіших версій Next.js;

– /public: Містить статичні файли, такі як зображення, шрифти, іконки тощо, які будуть доступні за їхнім шляхом від кореня вебсайту;

– /styles: Призначена для глобальних CSS-файлів, які впливають на весь додаток;

Next.js підтримує створення односторінкових програм (SPA), де навігація між сторінками відбувається без повного перезавантаження. Це досягається за допомогою маршрутизації на стороні клієнта та таких інструментів, як React Router.

3.3 Використання Tailwind CSS

Tailwind CSS – це утилітарний фреймворк CSS, який пропонує набір готових класів для швидкого та зручного оформлення вебелементів. Він відрізняється від традиційних фреймворків CSS, таких як Bootstrap або Foundation, тим, що не надає готових компонентів, а зосереджується на інструментах низького рівня, які можна комбінувати для створення будь-якого дизайну [6].

3.3.1 Основні принципи роботи Tailwind CSS

Tailwind CSS працює за принципом «мобільні девайси перші» (mobile-first), що означає, що стилі за замовчуванням оптимізовані для МД, а потім адаптуються для більших екранів за допомогою медіа-запитів.

Фреймворк надає велику кількість класів для налаштування різних властивостей елементів, таких як:

– розміри (ширина, висота, padding, margin);

– кольори (тексту, фону, границь);

– шрифти (розмір, вага);

– flexbox та grid (для створення складних та адаптивних макетів);
– ефекти (тіні, переходи);
– інші (позиціонування, відображення, трансформації).

Класи Tailwind CSS інтегруються в HTML-розмітку через атрибут className. Наприклад, для встановлення синього фону можна використовувати клас bg-blue-500. Класи можуть бути легко комбіновані для досягнення бажаного вигляду:

```
<div className = “hidden h-auto w-full grow rounded-md md:block”></div>
<form>
<button className=”flex h-[48px] w-full grow items-center justify-center gap-2>
```

Рисунок 3.2 – Приклад використання Tailwind

Наведений код демонструє використання Tailwind CSS для створення адаптивного дизайну. Елемент div прихований на малих екранах і відображається на середніх та великих. Кнопка має фіксовану висоту та займає всю доступну ширину, а її вміст центрується по вертикалі та горизонталі.

Tailwind CSS активно використовується у даному проєкті для швидкої та ефективної верстки інтерфейсу користувача, забезпечуючи гнучкість та зручність у налаштуванні стилів [21].

3.4 Розробка сторінки логіну

Сторінка логіну розроблена з урахуванням сучасних тенденцій вебдизайну та принципів User Interface/User Expirience (UI/UX). Вона має мінімалістичний вигляд, що дозволяє користувачеві зосередитися на основній задачі – авторизації в системі.

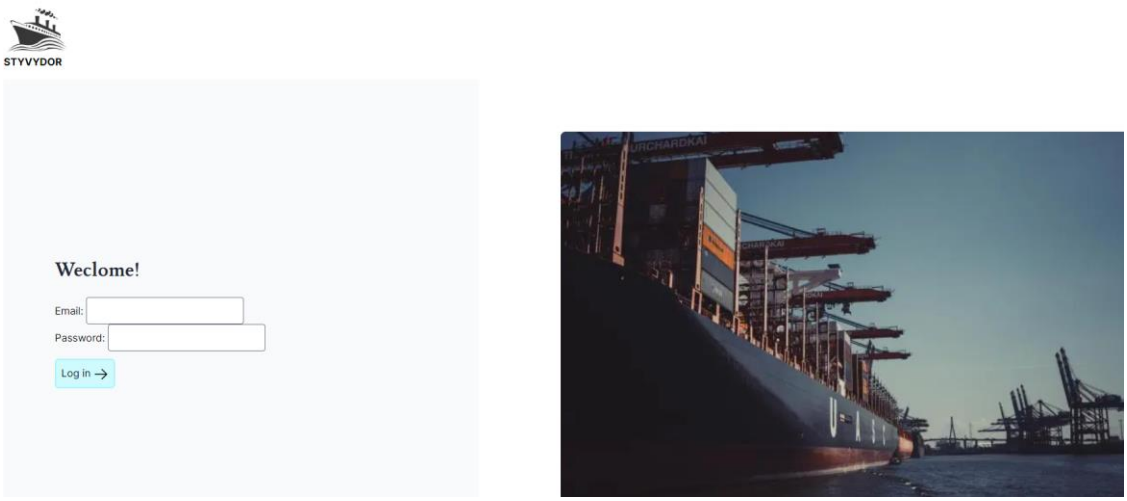


Рисунок 3.3 – Сторінка логіну

Сторінка містить лише необхідні елементи: поля для введення логіна та пароля, кнопку «Увійти». Такий підхід забезпечує інтуїтивно зрозумілий інтерфейс і швидкий доступ до функціональних можливостей системи.

3.4.1 Сценарії взаємодії користувача

Вхід до системи – це перший крок взаємодії користувача з додатком. Від того, наскільки цей процес буде зручним та безпечним, залежить загальне враження від використання продукту. Розглянуто два основні сценарії взаємодії користувача зі сторінкою на етапі входу:

1) **Успішний вхід:**
   - користувач вводить коректну електронну пошту та пароль;
   - після натискання кнопки "Увійти", система успішно автентифікує користувача;
   - користувач перенаправляється на головну сторінку або іншу захищену область додатку.
2) **Невдалий вхід:**
   - користувач вводить некоректну електронну пошту або пароль;
   - після натискання кнопки "Увійти", система не може автентифікувати користувача;

– система відображає повідомлення про помилку, вказуючи на некоректні дані.

Ці сценарії демонструють основні способи взаємодії користувачів зі сторінкою входу, забезпечуючи позитивний досвід під час успішного входу та повідомлення про помилку при невдалій спробі.

3.4.2 Реалізація логіну за допомогою Firebase

Для реалізації процесу входу у вебзастосунок було обрано службу Firebase Authentication, яка надає розробникам широкий спектр інструментів і функцій, що забезпечують зручну та безпечну автентифікацію користувачів. Однією з ключових переваг автентифікації Firebase є можливість автентифікації за допомогою адреси електронної пошти та пароля, що є методом, з яким знайома та зрозуміла більшість користувачів.

Інтеграція автентифікації Firebase у вебпрограму включає кілька кроків. Спочатку створюється проєкт Firebase і підключається до вебпрограми, налаштувавши відповідні параметри в консолі Firebase. Зокрема, необхідно було активувати метод автентифікації «Електронна пошта/Пароль», який дозволить користувачам входити в систему, використовуючи присвоєні їм адреси електронної пошти та паролі.

Наступним кроком є встановлення Firebase SDK у проєкті Next.js. SDK надає набір бібліотек і інструментів, які спрощують взаємодію з API Firebase і дозволяють реалізувати різні функції, включаючи автентифікацію. Після можна почати писати код для обробки входу за допомогою наданих Firebase API.

Процес входу починається з того, що користувач вводить адресу електронної пошти та пароль у відповідні поля форми на сторінці входу. Ці дані передаються на сервер, де викликається функція signInWithEmailAndPassword із Firebase SDK. Ця функція надсилає запит до Firebase Authentication, який перевіряє введені дані та повертає результат автентифікації.

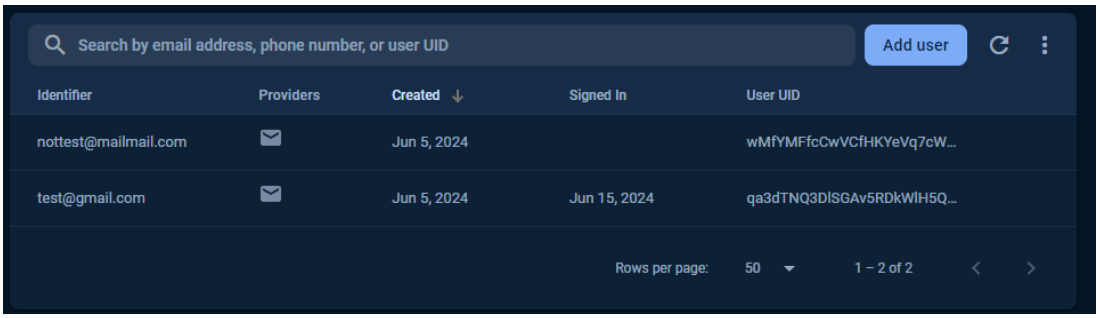


Рисунок 3.4 – Список користувачів у Firebase

Якщо користувач успішно ввійшов, вебпрограма отримає підтвердження від автентифікації Firebase, видасть маркер користувачеві, який увійшов, і переспрямує на сторінку контейнерів. Якщо автентифікація не вдається, автентифікація Firebase повертає повідомлення про помилку.

3.4.3 Безпека логін системи

Для забезпечення безпеки логін-системи вжито наступних заходів:

– хешування паролів: Паролі зберігаються в базі даних Firebase у захешованому вигляді, що ускладнює їх розшифровку у разі витоку даних;

– захист від brute-force атак: Firebase автоматично блокує IP-адреси, з яких надходять численні невдалі спроби входу;

– валідація даних на стороні клієнта та сервера: Перевірка коректності введених даних здійснюється як на стороні клієнта, так і на стороні сервера ;

– використання HTTPS: Для захисту передачі даних між клієнтом та сервером використовується протокол HTTPS.

Тільки адміністратори системи мають можливість додавати нові облікові записи користувачів через консоль Firebase. При цьому паролі нових користувачів генеруються автоматично та надсилаються їм електронною поштою.

3.5 Розробка сторінки контейнерів

Сторінка контейнерів розроблена з акцентом на зручність користувача та ефективне відображення інформації. Її дизайн спрямований на забезпечення швидкого доступу до даних про контейнери та можливостей для їх управління.

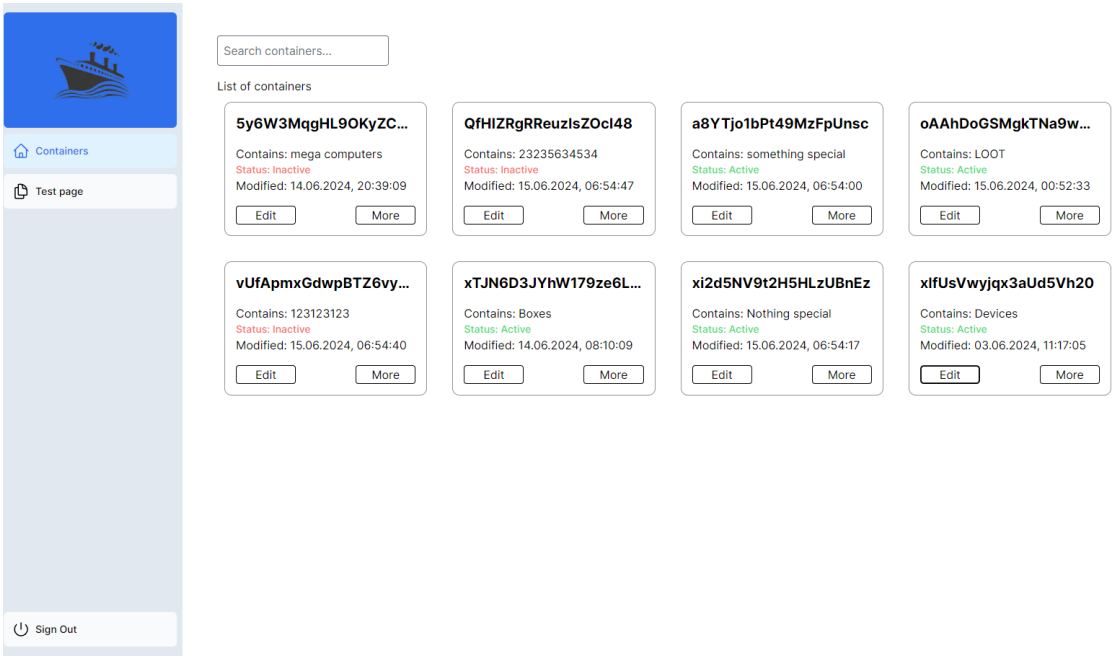


Рисунок 3.5 – Сторінка з інформацією контейнерів

Також присутне навігаційне меню, що розташоване зліва. Іконка компанії та пункт «Containers» відправляють на головну сторінку контейнерів. У нижній частині меню знаходиться кнопка «Sign Out», яка дозволяє користувачеві вийти з облікового запису та змінити аккаунт.

3.5.1 Логіка відображення контейнерів

Логіка відображення контейнерів на сторінці базується на отриманні даних з Firebase Realtime Database. Після завантаження сторінки відбувається фетчинг даних про контейнери з бази даних. Отримані дані потім обробляються та використовуються для формування карток контейнерів, які відображаються користувачеві на сторінці. Кожна картка містить ключову

інформацію про контейнер, таку як його назва, тип, статус та дата створення, що дозволяє користувачеві швидко оцінити вміст та стан кожного контейнера.

3.5.2 Маніпуляція даними контейнерів

Сторінка контейнерів пропонує користувачеві широкий спектр інструментів для ефективної взаємодії з даними. Пошуковий рядок дозволяє швидко та зручно знаходити потрібний контейнер за ключовими словами, такими як унікальний ідентифікатор, вміст контейнера або будь-які інші релевантні атрибути. Завдяки цьому користувач може миттєво знайти необхідну інформацію, не витрачаючи час на перегляд усього списку контейнерів.

Крім того, сторінка надає можливість фільтрувати контейнери за різними параметрами, такими як тип, статус та дата створення. Це дозволяє користувачеві звузити результати пошуку та отримати лише ті контейнери, які відповідають заданим критеріям. Така функціональність значно спрощує роботу з великою кількістю контейнерів та дозволяє швидко знаходити потрібну інформацію.

Для отримання детальної інформації про конкретний контейнер, користувач може натиснути на кнопку "more" на відповідній картці. Це відкриває модальне вікно, яке містить розширену інформацію про контейнер, включаючи його опис, вміст, дату останнього оновлення та інші релевантні дані.

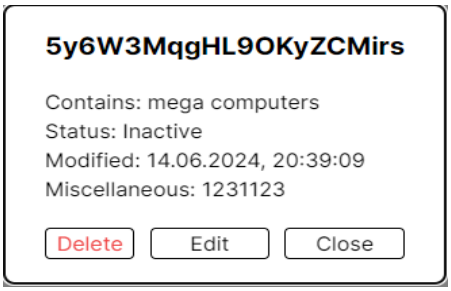


Рисунок 3.6 – Модальне вікно з інформацією контейнера

Процес редагування даних контейнера здійснюється через форму редагування, яка відкривається після натискання кнопки "Редагувати" на картці контейнера або у модальному вікні.

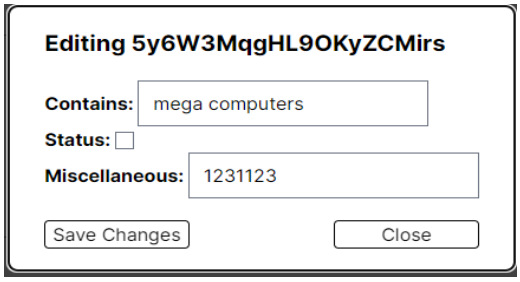


Рисунок 3.7 – Модальне вікно редагування інформації

Користувач може змінити будь-які дані контейнера, після чого зберегти зміни. Після збереження, оновлені дані автоматично відправляються до бази даних Firebase Realtime Database, де вони зберігаються та синхронізуються з іншими користувачами системи.

Висновки до розділу 3

У третьому розділі було детально розглянуто процес розробки вебзастосунку для моніторингу та управління контейнерами, використовуючи сучасні технології та інструменти. Було обрано стек технологій, що включає TypeScript, Next.js та Firebase, який забезпечує високу продуктивність, масштабованість та зручність розробки.

Детально описано структуру проєкту Next.js, яка дозволяє організувати код та ресурси у логічні блоки, спрощуючи процес розробки та підтримки. Також було розглянуто використання Tailwind CSS, утилітарного фреймворку CSS, який надає широкий набір готових класів для швидкої та зручної стилізації елементів інтерфейсу користувача.

Особливу увагу було приділено розробці сторінки логіну, яка забезпечує безпечний доступ до системи, використовуючи Firebase Authentication для автентифікації користувачів за допомогою електронної пошти та паролю. Було

описано заходи безпеки, вжиті для захисту логін-системи від несанкціонованого доступу та зловмисних дій.

Також було детально розглянуто розробку сторінки контейнерів, яка дозволяє користувачам переглядати, редагувати та видаляти інформацію про контейнери. Було описано логіку відображення та маніпуляції даними контейнерів, а також взаємодію користувача зі сторінкою.

Важливо зазначити, що створений вебзастосунок буде адаптований під апаратну частину роботи, забезпечуючи інтеграцію з пристроями Arduino та обробку даних, отриманих з них. Це дозволить створити комплексну систему моніторингу та управління контейнерами, яка буде ефективно працювати як на рівні програмного забезпечення, так і на рівні апаратного забезпечення.

4 РЕАЛІЗАЦІЯ АПАРАТНОЇ ЧАСТИНИ СИСТЕМИ

Розробка апаратної складової системи логістичного моніторингу вимагала ретельного аналізу та вибору оптимальної платформи, яка б відповідала всім вимогам проєкту. Основними критеріями при виборі мікроконтролера були вартість, функціональність, простота використання та енергоефективність.

4.1 Вибір мікроконтролера

Зважаючи на необхідність забезпечення бездротового зв'язку, обробки даних з RFID-міток, а також враховуючи обмеження у розмірах та енергоспоживанні, було розглянуто декілька варіантів мікроконтролерів.

4.1.1 Характеристики Arduino Uno

Arduino Uno – популярна та добре документована платформа, яка широко використовується в навчальних та хобі-проєктах.

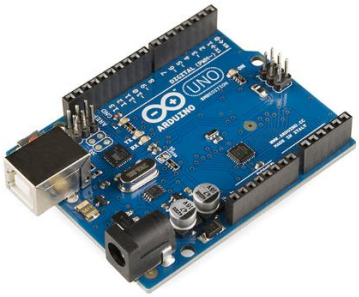


Рисунок 4.1 – Мікроконтролер Arduino Uno

Таблиця 4.1 – Характеристики мікроконтролеру Arduino Uno

Характеристика	Значення
Мікроконтролер	ATmega328P
Тактова частота	16 МГц
Напруга живлення	5 В
Цифрові піни	14 (з яких 6 можуть використовуватися як ШІМ)
Аналогові піни	6
Інтерфейси	UART, SPI, I2C

Arduino Uno, незважаючи на свою популярність та простоту використання, виявився непридатним для даного проєкту через відсутність вбудованого Wi-Fi модуля. Це означало б необхідність підключення зовнішнього модуля Wi-Fi, що збільшило б розміри та вартість пристрою, а також ускладнило б його програмування та налаштування [16].

4.1.2 Характеристики ESP8266

ESP8266 – це недорогий мікроконтролер з вбудованим Wi-Fi модулем, що робить його привабливим варіантом для проєктів, пов'язаних з Інтернетом речей.

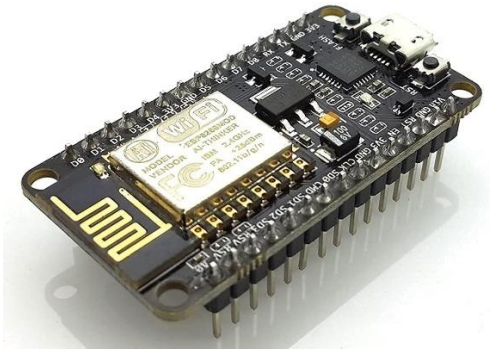


Рисунок 4.2 – Мікроконтролер ESP8266

Таблиця 4.2 – Характеристики мікроконтролеру ESP8266

Характеристика	Значення
Мікроконтролер	ESP8266
Тактова частота	80 МГц (може бути розігнаний до 160 МГц)
Напруга живлення	3.3 В
Цифрові піни	11
Аналогові піни	1 (з обмеженим діапазоном)
Інтерфейси	UART, SPI, I2C

ESP8266 привернув увагу завдяки вбудованому Wi-Fi модулю та відносно низькій вартості. Однак, обмежена кількість пінів вводу/виводу могла б створити труднощі при підключенні всіх необхідних сенсорів та

модулів, таких як RFID-зчитувач та GPS-модуль. Крім того, хоч ESP8266 і має достатню продуктивність для багатьох завдань, він може виявитися недостатньо потужним для обробки великих обсягів даних у режимі реального часу [10].

4.1.3 Характеристики Wemos D1

Wemos D1 – це мікроконтролер на базі популярної платформи Arduino, який має вбудований Wi-Fi модуль ESP8266. Це поєднання забезпечує зручність програмування, знайому користувачам Arduino [22].

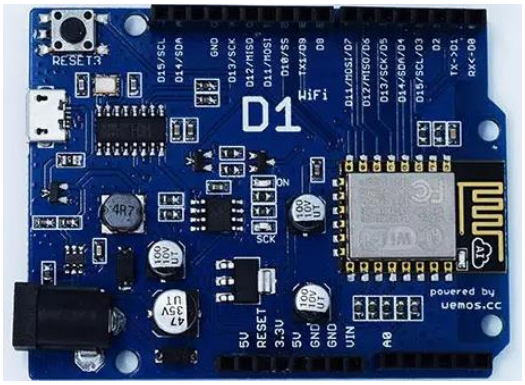


Рисунок 4.3 – Мікроконтролер Wemos D1

Таблиця 4.3 – Характеристики мікроконтролеру Wemos D1

Характеристика	Значення
Мікроконтролер	ESP8266
Тактова частота	80 МГц (може бути розігнаний до 160 МГц)
Напруга живлення	5 В
Цифрові піни	11
Аналогові піни	1 (з обмеженим діапазоном)
Інтерфейси	UART, SPI, I2C

Wemos D1, заснований на тому ж чіпі ESP8266, що й ESP-01, виявився оптимальним вибором для даного проєкту. Він усуває недоліки ESP-01, пропонуючи розширену кількість пінів вводу/виводу, зручний форм-фактор та можливість живлення від 5 В. Це дозволяє легко підключати різноманітні

сенсори та модулі, забезпечуючи необхідну функціональність системи. Крім того, Wemos D1 підтримує програмування через Arduino IDE, що робить його доступним навіть для розробників з невеликим досвідом.

4.2 Структура апаратного забезпечення

Щоб досягти ретельного спостереження за контейнерами та їх мобільністю, апаратна частина системи об’єднує цілий ряд компонентів, кожен з яких виконує життєво важливу роль у зборі, аналізі та передачі даних. В основі системи лежить мікроконтролер, який керує роботою всіх інших модулів і забезпечує безперебійний зв’язок із вебпрограмою.

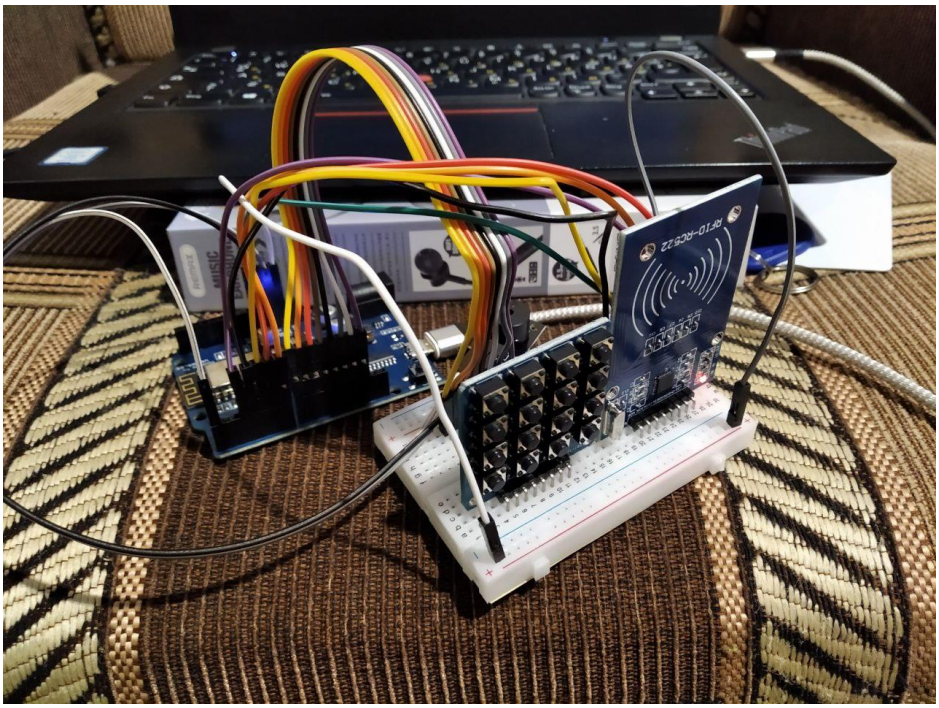


Рисунок 4.3 – Апаратна частина СЛМ

Структура апаратного забезпечення складається з наступних компонентів:

– Wemos D1: Мікроконтролер на базі Arduino, який виконує основні функції обробки даних, управління модулями та передачі інформації на сервер. ESP8266 забезпечує можливість підключення до Wi-Fi мережі, що дозволяє передавати дані на сервер бездротовим шляхом;

– RC522: Модуль RFID, що працює на частоті 13.56 МГц та підтримує стандарти ISO/IEC 14443 A та ISO/IEC 14443 B. Цей модуль відіграє ключову роль в ідентифікації контейнерів, зчитуючи унікальні ідентифікатори з RFID-міток, які можуть бути прикріплені до контейнерів. Вбудована антена забезпечує зчитування міток на відстані до 5 см [15];

– YL-102: модуль кнопок, який використовується для ручного керування процесом збору даних. Він дозволяє користувачеві ініціювати зчитування RFID-мітки та передачу даних на сервер, забезпечуючи додаткову гнучкість та контроль над системою.

Така структура апаратного забезпечення забезпечує комплексний підхід до моніторингу контейнерів, дозволяючи збирати дані про їх ідентифікацію, місцезнаходження та стан, що сприяє підвищенню ефективності та прозорості логістичних операцій.

4.2.1 Під'єднання модулів до Wemos D1

Під'єднання модуля RC522:

– VCC: Під'єднується до 3.3V;

– GND: Під'єднується до GND;

– MISO: Під'єднується до D6;

– MOSI: Під'єднується до D7;

– SCK: Під'єднується до D5;

– SDA (SS): Під'єднується до D8;

– RST: Під'єднується до D0.

Під'єднання модуля YL-102:

Оскільки YL-102 має 8 портів, для під'єднання до Wemos D1 ми вокористовуємо піни що залишилися. D1, D2, D3, D4, D9, D10, RX, TX

4.3 Взаємодія апаратної частини з вебзастосунком

Для забезпечення безпечної та надійної передачі даних між апаратною частиною системи (мікроконтролером Wemos D1) та вебзастосунком використовується протокол HTTPS. Цей протокол шифрує дані, що передаються, що запобігає їх перехопленню та несанкціонованому доступу.

Взаємодія між апаратною частиною та вебзастосунком здійснюється за допомогою HTTPS-запитів, які мікроконтролер відправляє на сервер, де розгорнуто вебзастосунок. Сервер, у свою чергу, обробляє ці запити та повертає відповідь мікроконтролеру.

Взаємодія апаратної частини з вебзастосунком здійснюється за допомогою двох основних типів запитів:

1) **запит на зчитування даних**: Цей запит ініціюється натисканням кнопки на мікроконтролері. При цьому відбувається зчитування даних з RFID-мітки, яка знаходиться в полі дії зчитувача RC522. Отримані дані відправляються на сервер у вигляді HTTPS-запиту. Сервер обробляє цей запит, зберігає дані у базі даних та оновлює відповідну інформацію на сторінці контейнерів у вебзастосунку;

2) **запит на редагування інформації**: Цей запит може бути ініційований двома способами:

– через модальне вікно: Користувач може відкрити модальне вікно для конкретного контейнера та внести зміни до його даних. Після збереження змін, вебзастосунок відправляє запит на сервер з новими даними, які необхідно записати на RFID-мітку. Сервер перевіряє, чи відповідає ID мітки, вказаний у запиті, фактичному ID мітки, прикріпленої до контейнера. Якщо дані збігаються, сервер відправляє команду мікроконтролеру на перезапис даних на мітку. Після успішного перезапису мікроконтролер відправляє підтвердження на сервер, і дані про контейнер оновлюються у базі даних та на сторінці контейнерів у вебзастосунку;

– через редагування даних існуючої мітки: Користувач може відредагувати дані контейнера безпосередньо на сторінці контейнерів. Після збереження змін, вебзастосунок відправляє запит на сервер з оновленими даними. Подальший процес аналогічний до описаного вище: сервер перевіряє ID мітки, відправляє команду на перезапис даних, отримує підтвердження та оновлює інформацію у базі даних та сторінці контейнерів.

Такий підхід забезпечує двосторонній обмін даними між апаратною та програмною частинами СЛМ, що дозволяє ефективно керувати та відстежувати стан контейнерів у режимі реального часу.

4.4 Бібліотеки для апаратної частини

Робота з RFID-зчитувачем та Wi-Fi модулем ESP8266 потребують встановлення спеціальних бібліотек. Вони забезпечать зручний інтерфейс для взаємодії з апаратними компонентами, дозволять підключитися до Wi-Fi мережі та обмінюватися даними з RFID-мітками.

1Бібліотека ESP8266WiFi є основою для взаємодії з Wi-Fi модулем ESP8266. Вона надає функції для підключення до мережі, налаштування параметрів з'єднання, отримання інформації про статус Wi-Fi тощо. Завдяки їй можна інтегрувати пристрій у локальну мережу або підключити його до Інтернету.

Бібліотека SPI (Serial Peripheral Interface) забезпечує стандартний інтерфейс для спілкування між мікроконтролером та периферійними пристроями. Вона дозволяє обмінюватися даними по послідовному каналу, використовуючи окремі лінії для передачі даних, тактування та вибору пристрою. SPI спрощує роботу з різними модулями, що підтримують цей інтерфейс, включаючи RFID-зчитувач RC522.

Бібліотека MFRC522. Ця бібліотека розроблена спеціально для роботи з RFID-зчитувачем MFRC522. Вона містить функції для ініціалізації модуля, читання та запису даних на RFID-мітки, а також налаштування різних

параметрів роботи зчитувача. MFRC522 значно полегшує процес інтеграції RFID-технології у проєкт.

Бібліотека HTTPClient надає зручний спосіб відправки HTTPS-запитів (GET, POST тощо) та обробки відповідей сервера. За допомогою HTTPClient можна отримувати дані з Інтернету, відправляти інформацію на сервер або взаємодіяти з хмарними сервісами.

ArduinoJson – це потужна і ефективна бібліотека, розроблена спеціально для роботи з даними у форматі JSON на мікроконтролерах Arduino та інших вбудованих системах. Вона надає зручні інструменти для:

– створення (серіалізації) JSON: Легко формуйте JSON-об'єкти та масиви з ваших даних для відправки на сервер або збереження у файли;

– розбору (десеріалізації) JSON: Зручно витягуйте потрібні значення з отриманих JSON-даних для подальшої обробки та використання у вашій програмі;

– економії ресурсів: Бібліотека оптимізована для роботи на пристроях з обмеженою пам'яттю та обчислювальною потужністю.

Бібліотека Keypad спрощує роботу з матричними клавіатурами, які часто використовуються в embedded-системах. Вона дозволяє легко зчитувати натискання клавіш, обробляти їх комбінації та навіть реалізовувати функціонал утримання клавіші. Keypad приховує складні деталі роботи з портами вводу-виводу, роблячи код більш читабельним та зрозумілим.

4.5 Написання та налагодження програмного коду

Для програмування мікроконтролера Wemos D1 використовується середовище розробки Arduino IDE. Цей вибір обумовлений його простотою використання, наявністю великої кількості бібліотек та документації.

На початку коду необхідно підключити всі необхідні бібліотеки, які будуть використовуватись для роботи з Wi-Fi модулем, RFID-зчитувачем та іншими компонентами.

Після підключення бібліотек, необхідно ініціалізувати компоненти та задати необхідні параметри для їх роботи. Для передачі даних на сервер необхідно встановити підключення до Wi-Fi мережі:

```
const char* ssid = "Xiaomi_34A4";
const char* password = "15426378";
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");
  SPI.begin();
mfrc522.PCD_Init();
}
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {D1, D2, D3, D4};
byte colPins[COLS] = {D9, D10, RX, TX};
```

Рисунок 4.4 – Конфігурація підключення до інтернету та матриці кнопок

У циклі loop() відбувається основна робота зчитування RFID-міток та передачі даних на сервер за допомогою HTTPS-запитів:

```
void loop() {
  if (WiFi.status() == WL_CONNECTED) {
    char key = keypad.getKey();
    if (key) {
      handleKeyPress(key);
    }
  }
}
void handleKeyPress(char key) {
  switch(key) {
    case 'A':
      readRFID();
      break;
    case 'B':
      WriteToRFID
      break;}
}
```

Рисунок 4.5 – Логіка зчитування та запису карток

Юзер сам вирішає чи зчитувати дані, чи оновити/записати нові на мітку. Керується це все через матрицю кнопок. За допомогою цієї матриці є можливість розширювати функціонал мікроконтроллера.

```
int httpCode = http.POST(jsonString);
    if (httpCode > 0) {
      String payload = http.getString();
      Serial.println("Server response: " + payload);} else {
      Serial.println("Error on sending POST: " + String(httpCode));}
```

Рисунок 4.6 – Логіка відправлення зчитаних даних до вебзастосунку

Наведений фрагмент коду дозволяє користувачеві відправляти дані у форматі JSON на вебсервер за допомогою POST-запиту. Після успішної відправки, користувач отримує відповідь від сервера, яка може містити інформацію про результат обробки даних або інші повідомлення. У разі невдачі, користувач отримує повідомлення про помилку з відповідним кодом.

4.5.1 Запис даних на мітку

Для запису даних на RFID-мітку, мікроконтролер повинен спочатку отримати необхідну інформацію з вебзастосунку.

```
int httpCode = http.GET();
    if (httpCode > 0) {
        if (httpCode == HTTP_CODE_OK) {
            String payload = http.getString();
            Serial.println("Received payload: " + payload);
            StaticJsonDocument<200> jsonDocument;
            DeserializationError error = deserializeJson(jsonDocument, payload);
            if (error) {
                Serial.println("Failed to parse JSON.");
                return false;}
            String contains = jsonDocument["contains"];
            bool status = jsonDocument["status"];
            String misc = jsonDocument["misc."];
```

Рисунок 4.7 – Отримання даних з застосунку для запису

Наведений фрагмент коду демонструє процес отримання даних з вебзастосунку для подальшого запису на RFID-мітку. За допомогою HTTPS

GET-запиту мікроконтролер отримує JSON-дані, які потім розбираються за допомогою бібліотеки ArduinoJson. З отриманого JSON-об'єкта витягуються значення полів «contains», «status» та «misc», які можуть містити інформацію про вміст, статус та додаткові дані, що будуть записані на мітку.

Висновки до розділу 4

У цьому розділі було проведено ретельний аналіз та вибір оптимальної апаратної платформи для системи логістичного моніторингу. Після розгляду різних варіантів мікроконтролерів, було обрано Wemos D1 завдяки його функціональності, зручності використання та сумісності з Arduino IDE.

Розроблена апаратна структура включає мікроконтролер Wemos D1, який слугує «мозком» системи, забезпечуючи обробку даних, управління модулями та комунікацію з вебзастосунком через Wi-Fi. Також використовується RFID-модуль RC522 для зчитування унікальних ідентифікаторів з міток, що кріпляться до контейнерів, та модуль кнопок YL-102 для ручного керування процесом збору та запису даних.

Взаємодія апаратної частини з вебзастосунком відбувається через протокол HTTPS. Мікроконтролер відправляє запити на сервер для зчитування даних RFID-міток, а також отримує запити для запису інформації на мітки.

Для полегшення розробки та забезпечення ефективної роботи системи було використано ряд бібліотек, таких як ESP8266WiFi для роботи з Wi-Fi модулем, SPI для комунікації з периферійними пристроями, MFRC522 для роботи з RFID-зчитувачем, HTTPClient для відправки HTTPS-запитів, Keypad для роботи з матрицею кнопок та ArduinoJson для обробки даних у форматі JSON.

Програмний код для мікроконтролера було написано та налагоджено в середовищі Arduino IDE. Код забезпечує зчитування даних з RFID-міток, їх передачу на вебсервер, отримання та запис нової інформації на мітки, а також ручне керування процесом за допомогою кнопок.

ВИСНОВКИ

У цій кваліфікаційній бакалаврській роботі було успішно розроблено СЛМ для стивідорної компанії. Завдяки інтеграції апаратного забезпечення Arduino та сучасних вебтехнологій, вдалося створити комплексне рішення для оптимізації логістичних процесів та підвищення ефективності роботи.

Для досягнення цієї мети було проведено детальний аналіз потреб стивідорних компаній та існуючих СЛМ. На основі отриманих результатів були визначені вимоги до розроблюваної СЛМ, що включають як функціональні можливості, так і технічні характеристики. З урахуванням цих вимог, було здійснено ретельний підбір електронних компонентів для апаратної частини системи.

Наступним етапом стало комплексне проєктування та розробка СЛМ, що охоплює як програмну, так і апаратну складові. Особлива увага була приділена розробці унікального та недорогого пристрою на базі Arduino, що дозволяє збирати дані про контейнери за допомогою RFID-технології. Також було створено зручний вебзастосунок для моніторингу та управління контейнерами, використовуючи сучасні технологій: Next.js, Firebase та Tailwind CSS.

Ключовим результатом роботи є створення унікального та недорогого пристрою на базі Arduino. Цей пристрій дозволяє збирати дані про контейнери за допомогою RFID-технології, забезпечуючи точне відстеження їхнього місцезнаходження та стану. Завдяки своїй простоті та гнучкості, пристрій легко інтегрується в існуючу інфраструктуру стивідорної компанії та може бути налаштований під її специфічні потреби.

Розроблена СЛМ має значний потенціал для покращення роботи стивідорних компаній. Вона дозволяє не лише оптимізувати логістичні процеси, скоротити час простою контейнерів та знизити операційні витрати, але й суттєво підвищити якість обслуговування клієнтів завдяки точному та швидкому відстеженню вантажів.

У перспективі, подальший розвиток системи може включати додавання нових функцій, таких як прогнозування попиту на контейнери, оптимізація маршрутів транспортування з урахуванням дорожніх умов та обмежень, а також інтеграція з іншими системами управління підприємством, що дозволить створити єдину інформаційну платформу для управління всіма аспектами діяльності стивідорної компанії. Крім того, можна розглянути можливість використання передових технологій, таких як машинне навчання та штучний інтелект, для аналізу накопичених даних та виявлення прихованих закономірностей, що може привести до подальшого покращення системи та підвищення її ефективності.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Рубець А. В. Система моніторингу для прогнозування використання ресурсів у розподілених системах. *Наукові нотатки*. 2020. № 68. С. 86–90. DOI: 10.36910/6775.24153966.2019.68.13.

2. Про морські порти України : Закон України від 17.05.2012 р. № 4709-VI. URL: https://is.gd/EbPDhe (дата звернення: 08.05.2024).

3. Резнік Н., Іванець І. Використання RFID-технології в області логістики та управління ланцюгами постачань. переваги та недоліки використання технології. Молодий вчений. 2022. № 2 (102). С. 76–81. DOI: 10.32839/2304-5809/2022-2-102-15.

4. Стивідорні компанії: їх особливості | Інформаційно-аналітичний порта. Інформаційно-аналітичний портал НБРА-IBRA / Інтернет-газета Курс Дня | І. URL: https://www.ibra.com.ua/analytics/164170-stividorni-kompaniyi-yih-osoblivosti (дата звернення: 07.06.2024).

5. Automotive, IoT & Industrial Solutions | NXP Semiconductors. URL: https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf (Last accessed: 08.06.2024).

6. Azhariyah S., Muhammad Mukhlis. Framework CSS: tailwind CSS untuk front-end website store PT. XYZ. Jurnal informatika. 2024. Vol. 3, no. 1. P. 30–36. DOI: 10.57094/ji.v3i1.1601.

7. Bourne S., Rihal T. Big data analytics. University of western ontario medical journal. 2019. Vol. 87, no. 2. P. 24–26. DOI: 10.5206/uwomj.v87i2.1149.

8. Clark T. Https. Encyclopedia of systems biology. New York, NY, 2013. P. 926. DOI: 10.1007/978-1-4419-9863-7_1571.

9. Dangat P. M. T. Industrial internet of things (IIOT). International journal for research in applied science and engineering technology. 2024. Vol. 12, no. 3. P. 2721–2726. DOI: 10.22214/ijraset.2024.59103.

10. Das M. Home automation using ESP8266. Transactions on machine design (TMD). 2018. Vol. 6, no. 2. P. 47. DOI: 10.6025/tmd/2018/6/2/43-46.

11. De Carli S. Advanced firestore. Build mobile apps with swiftui and firebase. Berkeley, CA, 2023. P. 95–104. DOI: 10.1007/978-1-4842-9452-9_5.

12. Next.js. Next.js by Vercel The React Framework. URL: https://nextjs.org/ (Last accessed: 13.06.2024).

13. Engber D. What does a port operator do, anyway?. Slate Magazine. URL: https://slate.com/news-and-politics/2006/02/what-does-a-port-operator-do-anyway.html (Last accessed: 06.05.2024).

14. Enhancing Android UIUX with Firestore. International research journal of modernization in engineering technology and science. 2024. DOI: 10.56726/irjmets48154.

15. Gaikwad O. RFID Attendence using RC522. International journal for research in applied science and engineering technology. 2020. Vol. 8, no. 5. P. 2386–2392. DOI: 10.22214/ijraset.2020.5392.

16. Kumar A. Home automation using arduino uno. International journal of scientific and research publications (IJSRP). 2019. Vol. 9, no. 12. DOI: 10.29322/ijsrp.9.12.2019.p9614.

17. Monitoring Z. Cra agenda: monitoring notes. Independently Published, 2020. 104 p.

18. Node.js. Carl Hanser Verlag GmbH & Co. 86 p.

19. Peldi Guilizzoni. Balsamiq. Version 4.6.3. Peldi Guilizzoni, 2008. URL: https://balsamiq.com/wireframes/ (Last accessed: 04.06.2024).

20. Pincheira M., Vecchio M., Giaffreda R. Characterization and costs of integrating blockchain and iot for agri-food traceability systems. Systems. 2022. Vol. 10, no. 3. P. 57. DOI: 10.3390/systems10030057.

21. Rappin N. Modern CSS with tailwind: flexible styling without the fuss. O'Reilly Media, Incorporated, 2022. 90 p.

22. Septiyanto A., Warta J., Sari R. Aplikasi pendeteksi kebocoran gas LPG berbasis wemos ESP8266 menggunakan peringatan notifikasi pada whatsapp. Journal of students' research in computer science. 2021. Vol. 2, no. 1. P. 1–10. DOI: 10.31599/jsrcs.v2i1.549.

23. Simulation analysis of seaport rijeka operations with established dry port / I. Lovrić et al. Pomorstvo. 2020. Vol. 34, no. 1. P. 129–145. DOI: 10.31217/p.34.1.15.

24. Terminal operating systems: main features, integration, and. AltexSoft. URL: https://www.altexsoft.com/blog/terminal-operating-system/ (Last accessed: 06.05.2024).

25. Zamlynskyi V. Development of stevidor enterprises as one of the means of improving the economy of the region. ECONOMIC SCIENTIFIC PORTAL – ECONOMIC SCIENTIFIC PORTAL. URL: https://economics.net.ua/files/archive/2021/No3/41.pdf (Last accessed: 04.06.2024).

ДОДАТОК А

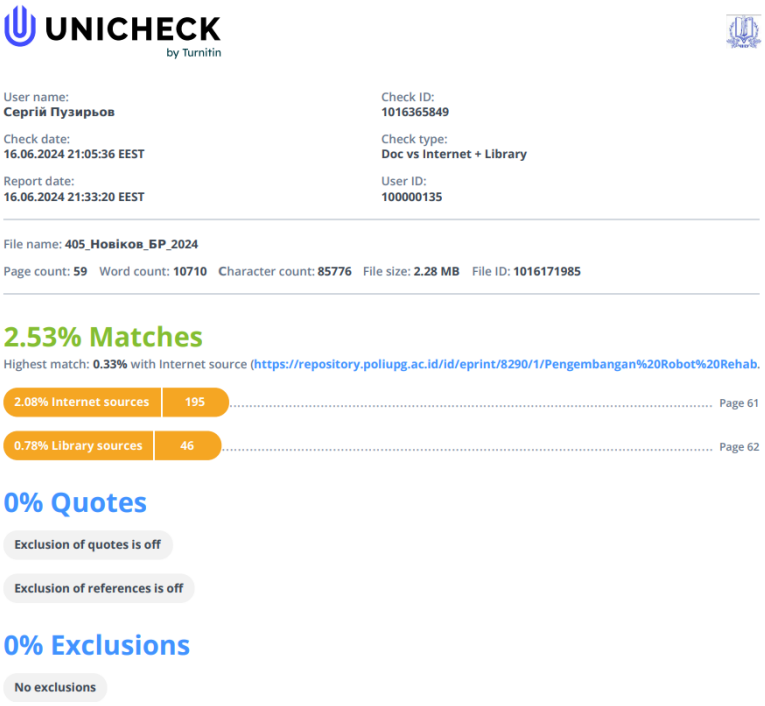
Довідка про перевірку на унікальність пояснювальної записки

бакалаврської кваліфікаційної роботи на тему:
«Система логістичного моніторингу на базі Arduino, ODB-II для стивідорної компанії»

студента спеціальності 123 «Комп'ютерна інженерія», 405 групи
Новіков Олександр Юрійович
прізвище, ім'я, по-батькові

Перевірку тексту здійснено сервісом: онлайн-сервіс Unicheck

Результат перевірки тексту бакалаврської кваліфікаційної роботи: схожість складає 2,53 %.


UNICHECK
by Turnitin

User name: Сергій Пузирьов
Check ID: 1016365849
Check date: 16.06.2024 21:05:36 EEST
Check type: Doc vs Internet + Library
Report date: 16.06.2024 21:33:20 EEST
User ID: 100000135

File name: 405_Новіков_БР_2024
Page count: 59 Word count: 10710 Character count: 85776 File size: 2.28 MB File ID: 1016171985

2.53% Matches
Highest match: 0.33% with Internet source (https://repository.poliupg.ac.id/id/eprint/8290/1/Pengembangan%20Robot%20Rehab.

2.08% Internet sources 195 Page 61
0.78% Library sources 46 Page 62

0% Quotes
Exclusion of quotes is off
Exclusion of references is off

0% Exclusions
No exclusions

Здобувач:

Керівник:
канд. фіз.-мат. наук, доцент

_____ Новіков О.Ю.
підпис ініціали, прізвище

_____ Пузирьов С. В.
підпис ініціали, прізвище

Дата: «___» _____ 2024 р.

ДОДАТОК Б

Код Arduino

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClientSecure.h>
#include <MFRC522.h>
#include <Keypad.h>
const char* ssid = "Xiaomi_34A4";
const char* password = "15426378";
#define RST_PIN D1
#define SS_PIN D2
MFRC522 mfrc522(SS_PIN, RST_PIN);

#define BUZZER_PIN D3

const byte ROWS = 4;
const byte COLS = 4;
char keys[ROWS][COLS] = {
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};
byte rowPins[ROWS] = {D4, D5, D6, D7};
byte colPins[COLS] = {D8, D9, D10, D11};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
const char* server = "https://nextjs-app-url.com/api/containers";

void sendTagData(String uid) {
  WiFiClientSecure client;
  HTTPClient https;

  client.setInsecure();

  if (https.begin(client, server)) {
    https.addHeader("Content-Type", "application/json");
    String postData = "{\"uid\":\"" + uid + "\"}";

    int httpCode = https.POST(postData);
    if (httpCode > 0) {
      String payload = https.getString();
      Serial.println(payload);
    } else {
      Serial.printf("POST...          failed,          error:          %s\n",
https.errorToString(httpCode).c_str());
    }
    https.end();
  }
}
```

```
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");

  SPI.begin();
  mfrc522.PCD_Init();
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, LOW);
}

void loop() {
  char key = keypad.getKey();

  if (key) {
    if (key == '1') {
      // Зчитування мітки
      if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
        String uid = "";
        for (byte i = 0; i < mfrc522.uid.size; i++) {
          uid += String(mfrc522.uid.uidByte[i], HEX);}
        uid.toUpperCase();
        Serial.println("UID tag: " + uid);
        sendTagData(uid);
        mfrc522.PICC_HaltA();
        mfrc522.PCD_StopCrypto1();
      } else {
        Serial.println("No card present");}
    } else if (key == '2') {
      // Запис даних на мітку
      // Отримання даних з вебзастосунку (можливо через іншу HTTPS функцію)
      // Запис цих даних на мітку
      String dataToWrite = "data_from_web_app";

      if (mfrc522.PICC_IsNewCardPresent() && mfrc522.PICC_ReadCardSerial()) {
        MFRC522::StatusCode    status    =    mfrc522.MIFARE_Write(4,    (byte
*)dataToWrite.c_str(), 16);
        if (status == MFRC522::STATUS_OK) {
          Serial.println("Data written to card");
          digitalWrite(BUZZER_PIN, HIGH);
          delay(100);
          digitalWrite(BUZZER_PIN, LOW);
        } else {
          Serial.println("Writing failed");}
        mfrc522.PICC_HaltA();
        mfrc522.PCD_StopCrypto1();
      } else {
        Serial.println("No card present");}}}}
```