НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Навчально-науковий інститут атомної та теплової енергетики

Кафедра цифрових технологій в енергетиці

«До захисту допущено»
Завідувач кафедри
_____ Наталія АУШЕВА
“___”_____2026 р

Дипломна робота
на здобуття ступеня бакалавр

За освітньою програмою “Цифрові технології в енергетиці”

Спеціальності 122 “Комп’ютерні науки”

на тему: “CRM-система з аналітичним модулем прогнозування ефективності виробництва.”

Виконала: студентка 4 курсу, групи ТР-21

Слюсар Катерина Ігорівна _____
(прізвище, ім’я, по батькові) (підпис)

Керівник асистент Олександр ВОЛКОВ _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Рецензент ст.викладач, доктор філософії, Ольга ВЛАСЕНКО _____
(посада, науковий ступінь, вчене звання, ім’я, ПРІЗВИЩЕ) (підпис)

Н.контроль ст.викладач Ольга БЕСПАЛА _____
(посада, ім’я, ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент_____

Київ - 2026

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ АТОМНОЇ ТА ТЕПЛОВОЇ ЕНЕРГЕТИКИ

Кафедра ЦИФРОВИХ ТЕХНОЛОГІЙ В ЕНЕРГЕТИЦІ

Рівень вищої освіти – перший (бакалаврський)
спеціальність 122 "Комп'ютерні науки"
Освітньо-професійна програма "Цифрові технології в енергетиці"

ЗАТВЕРДЖУЮ
Завідувач кафедри ЦТЕ
_____ Наталія АУШЕВА
(підпис)
"___" _____2026 р.

ЗАВДАННЯ
на дипломну роботу студенту

Слюсар Катерині Ігорівні
(прізвище, ім'я, по батькові)

1. Тема роботи «CRM-система з аналітичним модулем прогнозування ефективності виробництва».
Науковий керівник Волков Олександр Володимирович, ассистент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від " 28 " травня 2026 року № 1992 -с

2. Термін подання студентом роботи 04.06.2026

3. Вихідні дані до роботи мова програмування Python; фреймворк FastAPI; бібліотека React; середовище Vite; база даних MySQL; REST API; моделі даних Pydantic; засоби візуалізації Chart.js / Recharts; предметна область виробничого обліку та планування.

4. Перелік питань, які потрібно розробити
1) провести аналіз серед наявних аналогів
2) визначити головні функції веб-застосунку, що керує послугами та ресурсами

3) аргументувати вибрані інструменти, алгоритми та технології для реалізації веб-системи

4) побудувати архітектуру програмного забезпечення та бази даних

5) створити прикладний програмний веб-інтерфейс

6) представити процес застосування веб-інтерфейсу

5. Орієнтовний перелік ілюстративного матеріалу діаграма прецедентів, архітектурна схема, узагальнена структура бази даних, алгоритм прогнозування, послідовність отримання прогнозу, схема модулів клієнтської частини, схема тестування.

6. Дата видачі завдання 15.09.2025

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Затвердження теми роботи	вересень 2025	Виконано
2	Аналіз предметної області та збір вимог	20-26.04.2026	Виконано
3	Проектування архітектури та бази даних	27.04-03.05.2026	Виконано
4	Розробка серверної частини API	04-08.05.2026	Виконано
5	Розробка клієнтського інтерфейсу	09-12.05.2026	Виконано
6	Реалізація аналітичного модуля	13-17.05.2026	Виконано
7	Захист програмного забезпечення	14.05.2026	Виконано
8	Оформлення пояснювальної записки	25.05-02.06.2026	Виконано
9	Передзахист	04.06.2026	Виконано
10	Захист	17.06.2026	Виконано

Студент _____ Слюсар К.І.
(підпис) (прізвище та ініціали)

Керівник _____ Волков О.В.
(підпис) (прізвище та ініціали)

.

АНОТАЦІЯ

Записка містить 60 сторінок, 35 рисунків, 10 таблиць, 2 додатки, 14 посилань.

Метою даної роботи є розробка CRM-системи з аналітичним модулем прогнозування ефективності виробництва, яка забезпечує централізований облік працівників, робочих змін, графіків, виробничих результатів, складських операцій, виробів і технологічних процесів. Система орієнтована на використання у виробничому середовищі, де важливо швидко отримувати актуальні дані про стан персоналу, виконання плану та потенційні ризики затримки.

У роботі розглянуто предметну область виробничого управління, проаналізовано підходи до побудови CRM- та ERP-подібних систем, обґрунтовано вибір клієнт-серверної архітектури, React для клієнтської частини, FastAPI для серверної частини та MySQL для збереження структурованих даних. Окрему увагу приділено проектуванню бази даних, організації REST API, розподілу ролей користувачів та розробці модуля прогнозування, який використовує норми часу технологічних операцій і дані тижневих графіків працівників.

Практична цінність роботи полягає у створенні програмного продукту, що допомагає керівнику виробництва приймати рішення на основі даних: контролювати завантаження працівників, бачити дефекти, аналізувати виконання виробничого плану, виявляти вузькі місця та формувати рекомендації щодо перерозподілу персоналу.

Ключові слова: CRM-СИСТЕМА, ВИРОБНИЦТВО, ПРОГНОЗУВАННЯ, FASTAPI, REACT, MYSQL, REST API, АНАЛІТИКА, ПРАЦІВНИКИ, СКЛАД, ТЕХНОЛОГІЧНИЙ ПРОЦЕС.

ABSTRACT

The explanatory note contains 60 pages, 35 figures, 10 tables, 2 applications, and 14 references.

The aim of the qualification work is to develop a CRM system with an analytical module for forecasting production efficiency. The system provides centralized accounting of employees, work shifts, schedules, production results, warehouse operations, products, and technological processes. It is intended for a production environment where managers need timely information about personnel workload, production performance, and possible delays.

The work analyzes the subject area of production management, reviews approaches to building CRM and ERP-like systems, and justifies the use of a client-server architecture. React is used for the client side, FastAPI for the server API, and MySQL for storing structured data. Special attention is paid to database design, REST API organization, user-role separation, and the forecasting module based on process time standards and weekly employee schedules.

The practical value of the system is that it supports data-driven production management. The developed software allows the manager to monitor employee workload, analyze defects, track production plan execution, detect bottlenecks, and generate recommendations for redistributing staff between production stages.

Keywords: CRM SYSTEM, PRODUCTION, FORECASTING, FASTAPI, REACT, MYSQL, REST API, ANALYTICS, EMPLOYEES, WAREHOUSE, TECHNOLOGICAL PROCESS.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ТЕРМІНІВ

API - прикладний програмний інтерфейс, через який програмні компоненти обмінюються даними.

CRM - система керування взаємодією з клієнтами та внутрішніми процесами підприємства; у межах роботи використовується як платформа виробничого обліку.

ERP - система планування ресурсів підприємства.

REST - архітектурний підхід до побудови web API на основі HTTP-запитів.

JSON - текстовий формат обміну структурованими даними.

CRUD - базові операції роботи з даними: створення, читання, оновлення, видалення.

UI - користувацький інтерфейс.

UX - досвід користувача під час роботи із системою.

БД - база даних.

SQL - мова структурованих запитів для роботи з реляційними базами даних.

HTTP - протокол передавання даних у web-застосунках.

KPI - ключовий показник ефективності.

Вузьке місце - етап виробництва, який обмежує загальну швидкість виконання плану.

Норма часу - очікувана тривалість виконання однієї технологічної операції для одиниці продукції.

ВСТУП

Сучасні виробничі підприємства працюють в умовах постійної потреби у швидкому доступі до актуальної інформації. Дані про працівників, зміни, графіки, складські залишки, дефекти, виконані операції та виробничі плани часто зберігаються у різних таблицях або передаються вручну між відповідальними особами. Такий підхід ускладнює контроль виробництва, знижує прозорість процесів і створює ризик помилок під час прийняття управлінських рішень.

Особливо важливою є проблема прогнозування ефективності виробництва. Керівнику потрібно не лише знати, скільки продукції вже виготовлено, а й розуміти, чи вистачить працівників для виконання плану в заданий термін, який технологічний етап може стати критичним і де доцільно змінити розподіл персоналу. Без спеціалізованого програмного забезпечення така оцінка виконується вручну, потребує часу і залежить від досвіду конкретного менеджера.

Актуальність теми обумовлена необхідністю створення інформаційної системи, яка поєднує виробничий облік, складський контроль, управління персоналом та аналітичний модуль прогнозування. Запропонована CRM-система розглядається не лише як інструмент роботи з контактами, а як комплексна платформа для організації внутрішніх процесів підприємства. Вона повинна забезпечувати єдине джерело даних, рольовий доступ, зручний інтерфейс для адміністратора і працівника, а також автоматизований розрахунок прогнозу виконання виробничого плану.

Метою дипломної роботи є розроблення CRM-системи з аналітичним модулем прогнозування ефективності виробництва, що дозволяє автоматизувати облік працівників, змін, графіків, виробів, технологічних процесів, складських операцій і на основі цих даних оцінювати терміни виконання виробничого плану.

Об’єктом дослідження є процеси управління виробничою діяльністю підприємства, зокрема облік персоналу, планування роботи, фіксація результатів виробництва та контроль складських ресурсів. Предметом дослідження є методи та

програмні засоби розроблення web-орієнтованої CRM-системи з модулем аналітичного прогнозування виробничої ефективності.

Для досягнення поставленої мети необхідно виконати такі завдання:

- проаналізувати предметну область виробничого обліку та визначити основні інформаційні потоки системи;

- сформувати функціональні вимоги до адміністративної панелі, кабінету працівника, складського модуля та модуля прогнозування;

- обґрунтувати вибір технологій для клієнтської частини, серверної частини та бази даних;

- спроєктувати клієнт-серверну архітектуру CRM-системи та структуру реляційної бази даних;

- реалізувати основні API-маршрути для роботи з працівниками, змінами, графіками, виробами, складом і прогнозуванням;

- розробити алгоритм розрахунку строків виконання виробничого плану та визначення вузького місця;

- провести тестування системи та описати сценарії взаємодії користувачів із програмним продуктом [12].

Практична цінність роботи полягає у можливості використання розробленої системи як внутрішнього інструменту виробничого підприємства. Система дозволяє керівнику бачити реальний стан роботи, аналізувати результати, своєчасно реагувати на відхилення та приймати рішення на основі структурованих даних, а не лише на основі ручних звітів або суб’єктивних оцінок.

1 РОЗРОБКА CRM-СИСТЕМИ ДЛЯ ПРОГНОЗУВАННЯ ЕФЕКТИВНОСТІ ВИРОБНИЦТВА

Розробка CRM-системи для виробничого підприємства передбачає поєднання декількох груп функцій: обліку персоналу, планування графіків, фіксації виробничих результатів, складського контролю, статистики та прогнозування. На відміну від простого електронного журналу, така система повинна працювати як інтегрований інформаційний простір, де дані, введені в одному модулі, використовуються в інших модулях без дублювання.

1.1 Постановка задачі розробки

Задача розробки полягає у створенні програмного продукту, який забезпечує комплексне керування виробничими процесами підприємства. Основні ролі та сценарії взаємодії користувачів з системою подано на рисунку 1.1.

Рисунок 1.1 - Діаграма прецедентів CRM-системи

Система повинна дати можливість адміністратору реєструвати працівників, переглядати їхній статус, формувати або контролювати графіки, створювати

вироби та технологічні процеси, працювати зі складом, аналізувати статистику виробництва і запускати прогноз виконання плану.

Для працівника система повинна бути простішою. Працівник має бачити особистий кабінет, подавати тижневий графік, починати і завершувати зміну, фіксувати паузи, вносити кількість якісної та дефектної продукції, а також переглядати власні показники. Такий поділ дозволяє не перевантажувати користувачів зайвими функціями та водночас забезпечити керівника повною картиною виробничого процесу.

Узагальнені вимоги наведено в таблиці 1.1.

Таблиця 1.1 - Узагальнені вимоги до CRM-системи

Група вимог	Зміст вимоги	Результат для користувача
Функціональні	Авторизація, робота з працівниками, змінами, графіками, виробами, складом і прогнозом	Користувач виконує потрібні дії в одному інтерфейсі
Інформаційні	Збереження структурованих даних у MySQL	Дані не губляться і можуть використовуватися для звітності
Аналітичні	Розрахунок строків виконання плану і вузьких місць	Керівник бачить ризики та затримки
Рольові	Поділ на адміністратора і працівника	Кожен користувач бачить лише потрібні функції
Експлуатаційні	Запуск у локальній мережі або на робочому комп’ютері	Система може працювати у внутрішньому контурі підприємства

Вхідними даними системи є дані працівників, ролі користувачів, робочі зміни, тижневі графіки, назви виробів, описи технологічних процесів, норми часу, складські операції, комплекти, виробничі журнали, кількість якісних і дефектних одиниць продукції. Вихідними даними є адміністративні таблиці, статистичні показники, прогнози тривалості виконання плану, перелік критичних процесів, повідомлення про проблеми із графіками та рекомендації щодо перерозподілу працівників.

Основною вимогою до системи є цілісність даних. Наприклад, виробничий запис повинен бути пов’язаний із конкретним працівником, зміною та кількістю виготовленої продукції. Технологічний процес повинен належати певному виробу, а прогноз має використовувати актуальні дані з графіків і норм часу. Така зв’язаність забезпечується через реляційну базу даних та API-рівень, який перевіряє вхідні дані перед записом.

1.2 Аналіз предметної області виробничого підприємства

Предметна область охоплює процеси планування та контролю виробництва. У типовому виробничому середовищі менеджер або адміністратор повинен знати, які працівники доступні на поточному тижні, які процеси вони виконують, які вироби перебувають у роботі, скільки одиниць продукції виготовлено, які дефекти були виявлені і чи вистачає складських ресурсів для подальшого випуску.

Однією з проблем є розпорошеність інформації. Дані про графіки можуть зберігатися в окремих документах, виробничі результати - у журналі або таблиці, складські залишки - в іншому обліковому файлі. Через це керівник отримує фрагментарну картину, а для прийняття рішення доводиться вручну порівнювати декілька джерел. CRM-система усуває цю проблему завдяки централізації даних.

Другою проблемою є складність оцінки майбутнього навантаження. Навіть якщо відома кількість працівників і запланована кількість продукції, без

формального розрахунку складно визначити, який процес стане найбільш повільним. Наприклад, етап тестування може мати більшу норму часу, ніж монтаж, але якщо на тестування призначено достатньо працівників, вузьким місцем може виявитися інший етап. Тому прогноз повинен враховувати не лише норму часу, а й фактичний розподіл персоналу за графіками.

У межах системи виробничий процес розглядається як послідовність технологічних операцій. Кожна операція має назву, норму часу та порядок виконання. Для кожного виробу зберігається набір таких операцій. Під час прогнозування система аналізує, скільки працівників у графіку призначено на відповідні процеси, і розраховує очікувану тривалість виконання партії.

1.3 Огляд існуючих підходів до автоматизації виробничих процесів

Для автоматизації виробничих процесів часто використовуються ERP-системи, CRM-системи, MES-рішення, електронні таблиці та спеціалізовані внутрішні застосунки [10] – [11]. Великі ERP-рішення мають широкий функціонал, але можуть бути складними для впровадження, потребувати значних ресурсів і не завжди відповідати локальним процесам конкретного підприємства. Електронні таблиці простіші, але не забезпечують належного контролю доступу, цілісності даних і автоматизованої аналітики. Порівняння наведено в таблиці 1.2.

Розроблення власної CRM-системи дозволяє адаптувати функціонал до конкретного виробничого сценарію. У такому підході система не перевантажується зайвими модулями, а включає саме ті функції, які потрібні підприємству: кадровий облік, зміни, тижневі графіки, облік продукції, склад, комплекти, статистика та прогнозування.

Важливою перевагою web-орієнтованого підходу є незалежність клієнтської частини від конкретного робочого місця [8] – [9]. Користувач може відкрити

систему в браузері, а серверна частина забезпечує єдину бізнес-логіку та доступ до бази даних. Це полегшує підтримку, оскільки оновлення логіки відбувається на сервері, а не на кожному комп’ютері окремо.

У розробленій системі поєднуються елементи CRM, ERP та виробничої аналітики. CRM-частина відповідає за користувачів і персонал, ERP-подібна частина - за ресурси, склад і плани, а аналітичний модуль - за прогнозування строків і виявлення вузьких місць. Саме таке поєднання дає змогу розглядати виробництво як єдиний керований процес.

Таблиця 1.2 - Порівняння підходів до автоматизації

Підхід	Переваги	Недоліки	Доцільність для роботи
Електронні таблиці	Простота, швидкий старт	Помилки, дублювання, слабкий контроль доступу	Підходить лише для малого обсягу даних
Готова ERP-система	Багато модулів, стандартизовані процеси	Висока складність, потреба в адаптації	Надлишкова для навчального і локального проєкту
Окремі облікові програми	Закривають конкретні задачі	Немає єдиного інформаційного простору	Не вирішують проблему інтеграції
Власна CRM-система	Адаптація до процесів, гнучкість, контроль архітектури	Потребує розробки і тестування	Обрано як основний підхід

1.4 Ролі користувачів і сценарії роботи

У системі виділено дві основні ролі: адміністратор і працівник. Адміністратор має доступ до повного набору функцій, тому може керувати працівниками, переглядати статистику, редагувати виробничі процеси, працювати зі складом, призначати графіки і запускати прогноз. Працівник має доступ до персонального кабінету, де може виконувати дії, пов’язані з власною роботою.

Рольовий поділ важливий не лише для зручності, а й для безпеки. Працівнику не потрібно бачити адміністративні таблиці або мати можливість змінювати дані інших користувачів. Адміністратор, навпаки, повинен мати цілісне бачення всього підприємства. Тому після авторизації клієнтська частина відкриває різні дашборди залежно від ролі користувача. Основні функції наведено в таблиці 1.3.

Таблиця 1.3 - Ролі користувачів CRM-системи

Роль	Основні функції	Дані
Адміністратор	Персонал, статистика, графіки, склад, вироби, прогнозування	Повні дані підприємства
Працівник	Особистий графік, зміна, виробіток, власна статистика	Власні записи та план роботи
Система	Перевірка даних, розрахунок прогнозу, email-нагадування	Дані з API та бази даних

Типовий сценарій адміністратора починається з входу до системи, перегляду дашборду, перевірки активних змін, аналізу статистики і переходу до необхідного модуля. Якщо потрібно оцінити виконання плану, адміністратор створює або вибирає виріб, задає кількість продукції та запускає прогноз. Система повертає очікуваний термін, список процесів, завантаження працівників і рекомендацію.

Типовий сценарій працівника є коротшим: вхід до кабінету, подання графіку, початок зміни, фіксація паузи за потреби, завершення зміни та внесення

виробничих результатів. Така логіка зменшує кількість ручних повідомлень менеджеру і забезпечує накопичення даних для подальшої аналітики.

2 ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ РОЗРОБКИ

Для реалізації CRM-системи обрано клієнт-серверну архітектуру (рис. 2.1). Клієнтська частина відповідає за відображення сторінок і взаємодію з користувачем, серверна частина - за бізнес-логіку та обробку HTTP-запитів, а база даних - за довготривале збереження інформації [9] – [10].

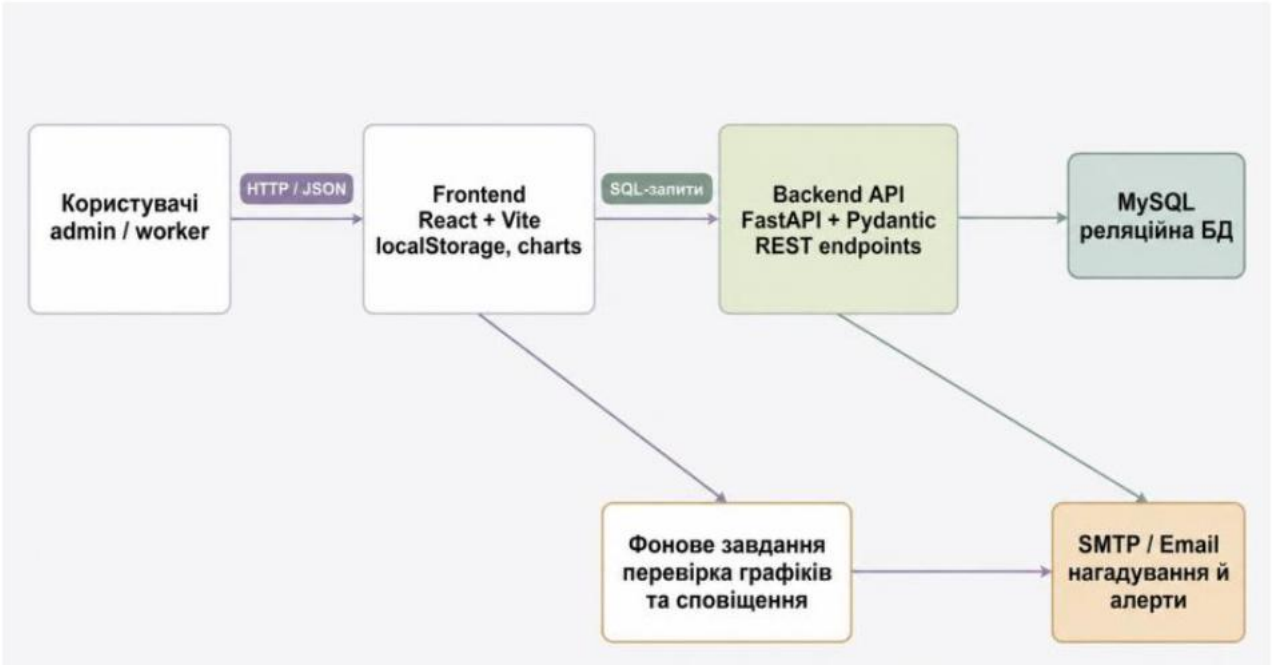

Рисунок 2.1 - Клієнт-серверна архітектура CRM-системи

Обрана архітектура є зручною для подальшого розвитку системи. За потреби можна окремо змінювати інтерфейс користувача, розширювати серверну логіку або оновлювати структуру бази даних без повної переробки всього застосунку. Крім того, такий підхід спрощує підтримку системи, підвищує її надійність і дозволяє поступово додавати нові функціональні модулі.

2.1 Клієнтська частина на React

Основні компоненти клієнтської частини наведено в таблиці 2.1. Тут React використовується для побудови клієнтської частини системи [2]. Основною перевагою такого підходу є компонентна структура. Кожний великий блок інтерфейсу реалізований як окремий компонент, що спрощує підтримку, повторне використання елементів та подальше розширення.

Таблиця 2.1 - Основні компоненти клієнтської частини

Компонент	Призначення
App.jsx	Зберігає стан авторизації, працює з localStorage, відкриває потрібний dashboard залежно від ролі
LoginScreen.jsx	Форма входу користувача до системи
AdminDashboard.jsx	Головна адміністративна панель, статистика, персонал, аналітика
WorkerDashboard.jsx	Особистий кабінет працівника, зміни, графік, виробіток
WarehouseDashboard.jsx	Складські операції, залишки, комплекти, історія
AdminProducts.jsx	Вироби, технологічні процеси, прогнозування
SyncManager.js	Логування дій, буферизація запитів у разі втрати мережі, синхронізація

У системі компонент App відповідає за стан авторизації та вибір dashboard-а. Якщо користувач не авторизований, відкривається екран LoginScreen. Якщо роль користувача дорівнює admin, відкривається AdminDashboard. Для звичайного

працівника відкривається WorkerDashboard. Така логіка дозволяє розділити функціональність без створення окремих застосунків.

Адміністративна панель містить модулі статистики, управління працівниками, графіками, виробами, складом, журналом змін і прогнозуванням. Кабінет працівника орієнтований на швидке виконання повсякденних дій: подання графіку, початок і завершення зміни, внесення виробітку, перегляд власних показників.

Для візуального представлення статистики використовуються бібліотеки Chart.js, react-chartjs-2 та Recharts [6] – [7]. Вони дають змогу будувати діаграми, які допомагають адміністратору швидше інтерпретувати дані: співвідношення якісної і дефектної продукції, активність працівників, виконання планів і динаміку виробництва.

2.2 Серверна частина на FastAPI

FastAPI використовується для реалізації серверної частини [1]. Він дозволяє описувати API-маршрути у вигляді функцій, працювати з моделями даних Pydantic, повертати відповіді у форматі JSON і швидко створювати REST API для клієнтської частини [1] – [5].

Серверна частина виконує роль посередника між інтерфейсом і базою даних. Коли користувач виконує дію у браузері, React надсилає HTTP-запит на відповідний endpoint. FastAPI приймає запит, перевіряє параметри, звертається до MySQL, виконує необхідну операцію і повертає результат.

Основні API-маршрути системи представлено в табл. 2.2.

Таблиця 2.2 - Основні API-маршрути системи

Група endpoint-ів	Приклади маршрутів	Призначення
Авторизація	POST /login	Вхід користувача і визначення ролі
Працівники	GET /employees, POST /employees, PUT /employees/{id}	Створення, редагування і перегляд працівників
Зміни	POST /shifts/start, PUT /shifts/{id}/pause, PUT /shifts/{id}/end	Початок, пауза і завершення зміни
Графіки	GET /schedules, POST /schedule/submit, POST /schedule/assign	Подання і призначення тижневих графіків
Виробництво	POST /production/log, GET /admin/stats	Фіксація виробітку і статистика
Вироби	GET /products, POST /products, PUT /products/{id}	Створення виробів і технологічних процесів
Склад	GET /inventory/balances, POST /inventory/operation	Залишки, прихід, списання, комплекти
Прогноз	GET /predict	Розрахунок строку виконання плану і вузького місця

У системі реалізовані API-маршрути для авторизації, роботи з працівниками, змінами, графіками, виробничими записами, виробами, складом, комплектами, історією складських операцій і прогнозуванням. Такий набір маршрутів дозволяє

розділити функції за логічними групами та зберегти зрозумілу структуру серверного коду.

Окремим елементом серверної частини є фонове завдання, яке перевіряє подання графіків і може формувати email-нагадування. Це демонструє, що backend виконує не лише відповідь на запити користувача, а й регулярні службові дії, пов’язані з організацією роботи підприємства.

2.3 Використання MySQL та REST API

Обрано MySQL як реляційну базу даних, оскільки інформація системи має чіткі зв’язки [4]. Працівники мають зміни та графіки, вироби мають технологічні процеси, виробничі записи пов’язані з працівниками, складські операції впливають на залишки, а комплекти складаються з окремих компонентів.

Реляційна модель дозволяє підтримувати цілісність даних і виконувати аналітичні запити. Наприклад, статистика адміністратора розраховується на основі сумарної кількості якісної та дефектної продукції, активних змін і продуктивності працівників. Прогнозування використовує дані з таблиць products, product_processes та weekly_schedules.

Обмін між frontend і backend відбувається через REST API. Клієнт надсилає HTTP-запити у форматі JSON, а сервер повертає структуровану відповідь [8] – [9]. Такий підхід є зручним для web-застосунків, оскільки інтерфейс і бізнес-логіка не залежать від внутрішньої реалізації одне одного.

Завдяки REST API систему можна розширювати. У майбутньому до неї можуть бути додані мобільний застосунок, окремий аналітичний dashboard або інтеграція з обладнанням, якщо вони будуть використовувати ті самі API-маршрути.

2.4 Безпека, обмеження доступу та надійність

Безпека системи забезпечується насамперед рольовим розмежуванням доступу. Після входу користувач отримує роль, а клієнтська частина відкриває відповідний набір функцій. Адміністратор може працювати з управлінськими модулями, а працівник - лише зі своїм кабінетом.

На рівні серверної частини важливо перевіряти коректність вхідних даних. Для цього використовуються Pydantic-моделі, які описують очікувану структуру запиту [5]. Якщо користувач передає неправильні дані, сервер може повернути помилку замість запису некоректної інформації в базу даних.

Надійність клієнтської частини підвищує механізм SyncManager. Він фіксує дії користувача, може зберігати модифікуючі запити в локальний буфер, якщо мережа недоступна, і синхронізувати їх після відновлення підключення. Для виробничого середовища це важливо, оскільки короткочасна втрата мережі не повинна призводити до втрати введених даних.

Окремим питанням є захист конфігураційних параметрів і паролів. У фінальній версії системи такі дані доцільно переносити у змінні середовища або окремі конфігураційні файли, які не публікуються разом із репозиторієм. Це дозволить уникнути випадкового розкриття службової інформації.

3 МОДЕЛЬ ДАНИХ ТА АЛГОРИТМ ПРОГНОЗУВАННЯ

3.1 Структура бази даних

База даних містить таблиці для обліку працівників, робочих змін, графіків, виробів, технологічних процесів, виробничих записів, складських позицій, історії складських операцій, комплектів і компонентів (рис. 3.1) [4] – [14].

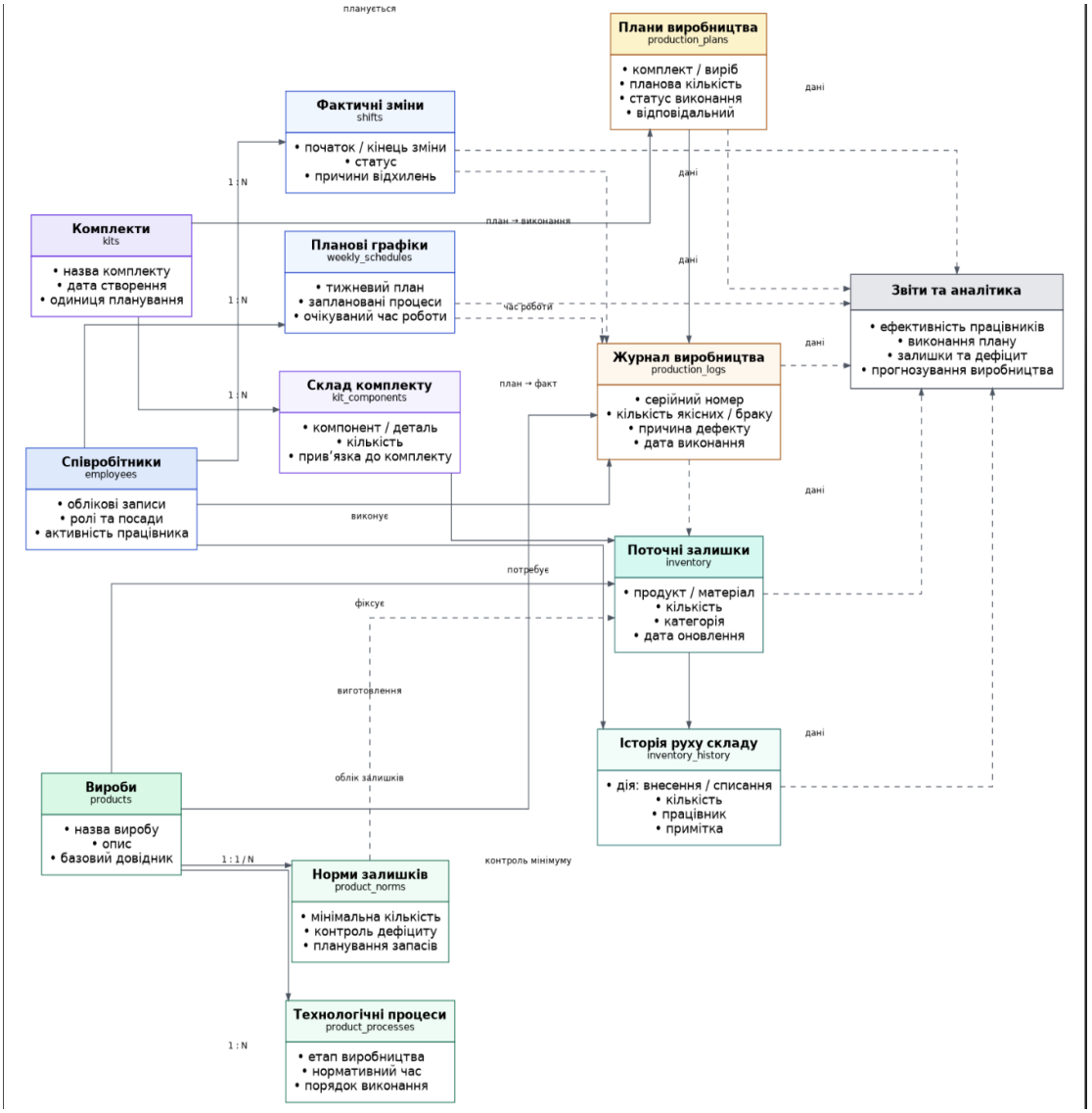

Рисунок 3.1 - Узагальнена структура бази даних CRM-системи

Така структура дозволяє охопити як адміністративний, так і виробничий контур системи.

Таблиця employees зберігає інформацію про працівників: ім’я, прізвище, email, посаду, роль, статус активності та дані для входу. Таблиця shifts фіксує початок, паузи і завершення робочої зміни. Таблиця weekly_schedules містить плани роботи на тиждень, зокрема процеси, на які працівник призначений у конкретні дні.

Виробничий блок представлений таблицями products, product_processes і production_logs. У products зберігаються вироби, у product_processes - технологічні операції з нормами часу, а production_logs фіксує фактичний виробіток працівників. Саме ці таблиці формують основу для статистики і прогнозування.

Складський блок містить таблиці inventory, inventory_history, kits і kit_components. Таблиця inventory показує актуальні залишки, inventory_history зберігає історію приходів і списань, а kits та kit_components дозволяють описувати комплекти, для виготовлення яких потрібно списати декілька позицій зі складу.

3.2 Математичне подання прогнозування

Аналітичний модуль прогнозування базується на припущенні, що для кожного виробу відома послідовність технологічних процесів і норма часу на виконання кожного процесу для однієї одиниці продукції. Також система використовує дані графіків, щоб оцінити, скільки працівників фактично призначено на кожний процес (рис. 3.2) [14].

Рисунок 3.2 - Алгоритм прогнозування виробничої ефективності

Для прогнозування система використовує планову кількість продукції, норму часу на виконання кожного виробничого процесу та кількість працівників, які призначені на ці процеси. Спочатку визначається, скільки часу потрібно для виконання кожного етапу виробництва. Після цього система враховує кількість доступних працівників і розраховує, скільки робочих днів може зайняти виконання відповідного процесу.

Якщо на певний процес не призначено жодного працівника, система не може коректно спрогнозувати строк виконання. У такому випадку цей процес вважається проблемним, а користувач отримує попередження про необхідність призначити персонал.

Загальний термін виконання партії визначається за процесом, який потребує

найбільше часу. Саме він вважається вузьким місцем виробництва, оскільки може сповільнювати виконання всього плану. На основі цього система формує рекомендацію для адміністратора, наприклад щодо перерозподілу працівників або посилення окремого етапу виробництва. Такий підхід допомагає адміністратору побачити, як можна збалансувати навантаження.

3.3 Алгоритм визначення вузького місця

Алгоритм прогнозування реалізований у серверному endpoint-і /predict. На вхід він отримує ідентифікатор виробу та планову кількість продукції. Після цього сервер звертається до таблиці product_processes і отримує перелік технологічних операцій з нормами часу.

Наступним кроком є вибір актуального тижня. Система шукає останній поданий тижневий графік і аналізує, які процеси вказані для працівників у понеділок, вівторок, середу, четвер і п’ятницю. Кожне призначення процесу збільшує лічильник змін для відповідного процесу. Після цього кількість змін ділиться на п’ять, щоб отримати середню кількість працівників на день.

Для кожного процесу система розраховує загальну потребу у хвилинах і кількість днів. Якщо працівників на процесі немає, формується попередження. Якщо всі процеси мають працівників, система знаходить максимальну тривалість і визначає відповідний процес як вузьке місце. Додатково формується текстова рекомендація: залишити розподіл, якщо він виглядає збалансовано, або перевести людей із найшвидшого процесу на критичний.

Результат повертається у форматі JSON і містить планову кількість, загальну оцінку строку, назву вузького місця, рекомендацію та деталізацію за процесами. Завдяки цьому клієнтська частина може не лише показати загальний прогноз, а й вивести таблицю з поясненням, чому отримано саме такий результат.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ РОЗРОБЛЕНОЇ СИСТЕМИ

Програмна реалізація виконана у вигляді web-застосунку з клієнтською частиною на React, серверною частиною на FastAPI та базою даних MySQL. Така структура дозволяє розділити відповідальність між модулями, спростити тестування і забезпечити можливість подальшого розширення.

4.1 Загальна структура системи та її компоненти

Загальна структура системи включає frontend, backend і database. Frontend відповідає за інтерфейс користувача, форми, таблиці, кнопки, графіки та відображення результатів прогнозування. Backend реалізує логіку авторизації, перевірку даних, виконання SQL-запитів і формування відповідей. Database зберігає всі робочі дані.

Клієнтська частина містить окремі компоненти для адміністратора і працівника. Такий підхід відповідає принципу розділення відповідальності: кожний компонент має власний стан, власні функції завантаження даних і власну логіку відображення [10]-[13]. Наприклад, AdminProducts відповідає за вироби та прогноз, а WarehouseDashboard - за складські операції.

Серверна частина складається з набору endpoint-ів. Кожний endpoint відповідає за конкретну операцію: створення працівника, оновлення графіку, початок зміни, запис виробітку, отримання статистики або розрахунок прогнозу. Це спрощує підтримку, оскільки логіку кожного процесу можна аналізувати окремо.

Для взаємодії з MySQL використовується функція підключення до бази даних. Після виконання SQL-запиту курсор закривається, а з’єднання завершується.

Такий підхід є простим і зрозумілим для навчального проєкту, однак у промисловій версії доцільно додати пул з’єднань і централізовану обробку помилок.

4.2 Реалізація функціональних модулів CRM

Функціональні модулі CRM-системи наведено в табл. 4.1.

Модуль авторизації перевіряє дані користувача і визначає його роль. Після успішного входу інформація про користувача зберігається у localStorage, що дозволяє не втрачати сесію при оновленні сторінки. Якщо користувач виходить із системи, запис видаляється, а інтерфейс повертається до екрана входу.

Адміністративний модуль надає керівнику доступ до статистики, працівників, журналу змін, графіків, складу, виробів і аналітичного прогнозу. На головному dashboard-і відображаються ключові показники: загальна кількість виготовленої продукції, кількість дефектів, активні зміни та ефективність працівників.

Модуль працівника призначений для щоденної роботи. Працівник може почати зміну, поставити паузу, завершити зміну, вказати причину запізнення або раннього виходу, подати графік на наступний тиждень і внести виробничий результат. Завдяки цьому дані про роботу надходять до системи без додаткового ручного перенесення.

Складський модуль підтримує операції приходу і списання, перегляд залишків, категорії товарів, комплекти та історію. Під час списання система перевіряє, чи достатньо залишку на складі. У разі нестачі повертається помилка, що запобігає формуванню некоректних залишків.

Модуль виробів дозволяє створювати виріб, додавати технологічні процеси та задавати норму часу для кожного етапу. Ці дані використовуються не лише для довідника, а й як основа аналітичного прогнозування. Якщо норми часу не задані або задані неправильно, прогноз буде неточним, тому цей модуль має важливе значення для всієї системи.

Таблиця 4.1 - Функціональні модулі CRM-системи

Модуль	Користувач	Основний результат
Авторизація	Адміністратор, працівник	Вхід до системи та визначення ролі
Працівники	Адміністратор	Створення, редагування, звільнення, перегляд персоналу
Зміни	Працівник, адміністратор	Фіксація фактичного робочого часу
Графіки	Працівник, адміністратор	Планування роботи за днями тижня
Вироби і процеси	Адміністратор	Опис продукції та норм часу
Склад	Адміністратор	Контроль залишків і операцій
Прогноз	Адміністратор	Оцінка строків і вузьких місць

4.3 Реалізація API-маршрутів

У системі API-маршрути реалізують основні сценарії системи та забезпечують обмін даними між клієнтською і серверною частинами [1]-[9]. Для авторизації використовується POST /login. Для працівників передбачені маршрути перегляду, створення, редагування та звільнення. Для змін реалізовані маршрути початку, паузи, завершення, адміністративного редагування і видалення.

Для виробничої статистики використовується POST /production/log, що

записує виробіток працівника, а також GET /admin/stats, який повертає сумарні показники для адміністративного dashboard-а. Для графіків використовуються маршрути подання графіку працівником і призначення графіку адміністратором.

Складський API містить маршрути для перегляду залишків, виконання складських операцій, редагування позицій, роботи з нормами, комплектами, історією та планами. Завдяки цьому складська логіка не змішується з виробничим журналом, але обидва модулі можуть бути використані в межах одного інтерфейсу.

Маршрут GET /predict є центральним для аналітичного модуля. Він отримує product_id і target_qty, завантажує процеси виробу, аналізує актуальні графіки, розраховує кількість працівників на кожному процесі, визначає очікуваний строк і формує рекомендацію. Клієнтська частина відображає отриманий результат у зрозумілому вигляді.

Приклади взаємодії з API представлено в табл. 4.2.

Таблиця 4.2 - Приклади взаємодії з API

Сценарій	Запит	Очікувана відповідь
Вхід користувача	POST /login	Дані користувача та роль
Початок зміни	POST /shifts/start	Ідентифікатор активної зміни
Запис виробітку	POST /production/log	Підтвердження запису
Отримання статистики	GET /admin/stats	Сумарні показники і продуктивність
Створення виробу	POST /products	Повідомлення про створення
Прогнозування	GET /predict product_id=...&target_qty=...	Дні, вузьке місце, рекомендація, деталізація

4.4 Тестування і налагодження системи

Тестування системи виконувалося за основними сценаріями користувачів. Приклад, можливих випадків наведено в табл. 4.3.

Таблиця 4.3 - Основні тестові випадки

№	Перевірка	Очікуваний результат
1	Вхід адміністратора	Відкривається адміністративний дашборд
2	Вхід працівника	Відкривається кабінет працівника
3	Створення виробу з процесами	Виріб зберігається, процеси відображаються у списку
4	Початок і завершення зміни	Створюється запис зміни зі статусом completed
5	Запис виробітку	У статистиці збільшується кількість продукції
6	Списання зі складу при недостатньому залишку	Система повертає помилку і не змінює залишок
7	Прогноз при відсутності працівника на процесі	Система показує попередження про неможливість виконання
8	Прогноз при збалансованому графіку	Система повертає строк виконання і позитивну рекомендацію

Спочатку перевіряється авторизація: правильний вхід адміністратора, правильний вхід працівника, реакція на неправильні дані та коректне завершення сесії. Далі перевіряються CRUD-операції для працівників, виробів, процесів і складських позицій.

Окремо тестується робота змін. Перевіряється початок зміни, пауза, завершення, наявність незавершеної зміни, редагування зміни адміністратором і

відображення записів у журналі. Це важливо, оскільки дані про зміни впливають на статистику та можуть використовуватися для оцінки фактичного робочого часу.

Для модуля прогнозування тестування виконується на різних наборах даних. Перевіряються ситуації, коли всі процеси мають працівників, коли один із процесів не має працівника, коли планова кількість мала, коли планова кількість велика, а також коли розподіл працівників явно незбалансований. У кожному випадку очікується, що система сформує зрозумілу рекомендацію.

Під час налагодження також перевіряються відповіді API, статуси HTTP, коректність SQL-запитів, формат дат і часу, а також відображення повідомлень в інтерфейсі. Для клієнтської частини важливим є тестування реакції на недоступність backend-а або втрату мережі.

5 ВЗАЄМОДІЯ КОРИСТУВАЧА З CRM-СИСТЕМОЮ

Взаємодія користувача із системою побудована навколо двох основних сценаріїв: роботи адміністратора і роботи працівника. Інтерфейс повинен бути зрозумілим, оскільки система використовується у виробничому середовищі, де користувачі не повинні витрачати багато часу на пошук потрібної функції.

5.1 Вимоги до встановлення та запуску

Для запуску системи потрібні встановлені Python, Node.js, MySQL та браузер. Backend запускається як FastAPI-сервер, який приймає запити на локальному порту. Frontend запускається через Vite і відкривається у браузері [1]-[4]. База даних MySQL повинна містити необхідні таблиці та тестові або робочі дані.

Перед запуском потрібно налаштувати параметри підключення до бази даних. У фінальній версії рекомендовано зберігати їх у змінних середовища, щоб не розміщувати паролі у вихідному коді. Також потрібно переконатися, що CORS дозволяє запити з адреси, на якій працює frontend.

Типовий порядок запуску включає запуск MySQL, запуск backend-команди FastAPI/Uvicorn, запуск frontend-команди npm run dev і відкриття сторінки у браузері. Після цього користувач може увійти до системи під роллю адміністратора або працівника. Узагальнення по вимогах представлено в табл. 5.1.

Таблиця 5.1 - Вимоги до запуску системи

Компонент	Потрібне середовище	Призначення
Backend	Python, FastAPI, Uvicorn	Обробка запитів і бізнес-логіка
Frontend	Node.js, Vite, React	Інтерфейс користувача
Database	MySQL	Збереження даних
Browser	Chrome, Edge, Safari або інший сучасний браузер	Доступ до web-інтерфейсу

5.2 Опис інтерфейсу користувача

Інтерфейс CRM-системи побудовано таким чином, щоб забезпечити зручну роботу як адміністратора, так і звичайного працівника. Основна ідея полягає у розділенні функціональних можливостей залежно від ролі користувача. Адміністратор має доступ до керування працівниками, графіками, виробництвом, складом, звітами та аналітикою. Працівник використовує спрощений інтерфейс, у якому зосереджено лише ті дії, що потрібні для щоденної роботи: перегляд зміни, подання графіка, фіксація виробітку та перегляд власної статистики.

Такий підхід дозволяє не перевантажувати інтерфейс зайвими елементами та одночасно підвищує безпеку системи. Користувач бачить тільки ті модулі й кнопки, які відповідають його ролі та правам доступу. Це зменшує ймовірність випадкових змін у критично важливих даних, наприклад у графіках роботи, складських залишках або виробничих планах.

5.2.1 Вхід у систему

Першим екраном системи є форма авторизації користувача (рис. 5.1). Вона призначена для перевірки особи користувача перед наданням доступу до основного функціоналу CRM-системи. На формі розміщено поля для введення логіна та пароля, кнопку входу, а також посилання для подання запиту на доступ до системи.

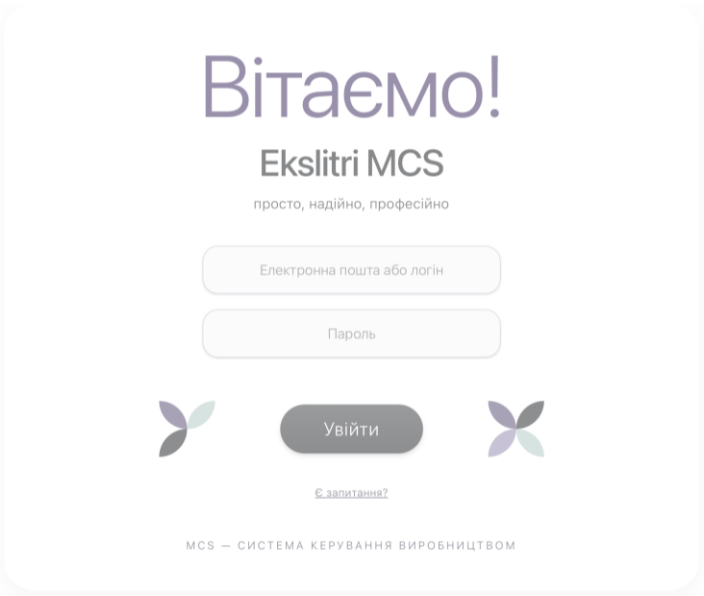

Рисунок 5.1 - Вхід у систему

Після введення логіна і пароля користувач натискає кнопку входу. Система перевіряє введені дані та визначає роль користувача. Якщо обліковий запис належить адміністратору, відкривається адміністративна панель. Якщо авторизується працівник, система перенаправляє його до персонального кабінету. У разі неправильного введення даних система повідомляє користувача про помилку та не надає доступ до внутрішніх модулів.

Окремою функцією є можливість подати запит на доступ до системи. Це

потрібно у випадку, коли працівник ще не має створеного облікового запису або його дані відсутні в базі. Для цього на екрані входу передбачено посилання на форму запиту доступу (рис. 5.2). Такий механізм дозволяє уникнути самостійної реєстрації без контролю адміністратора.

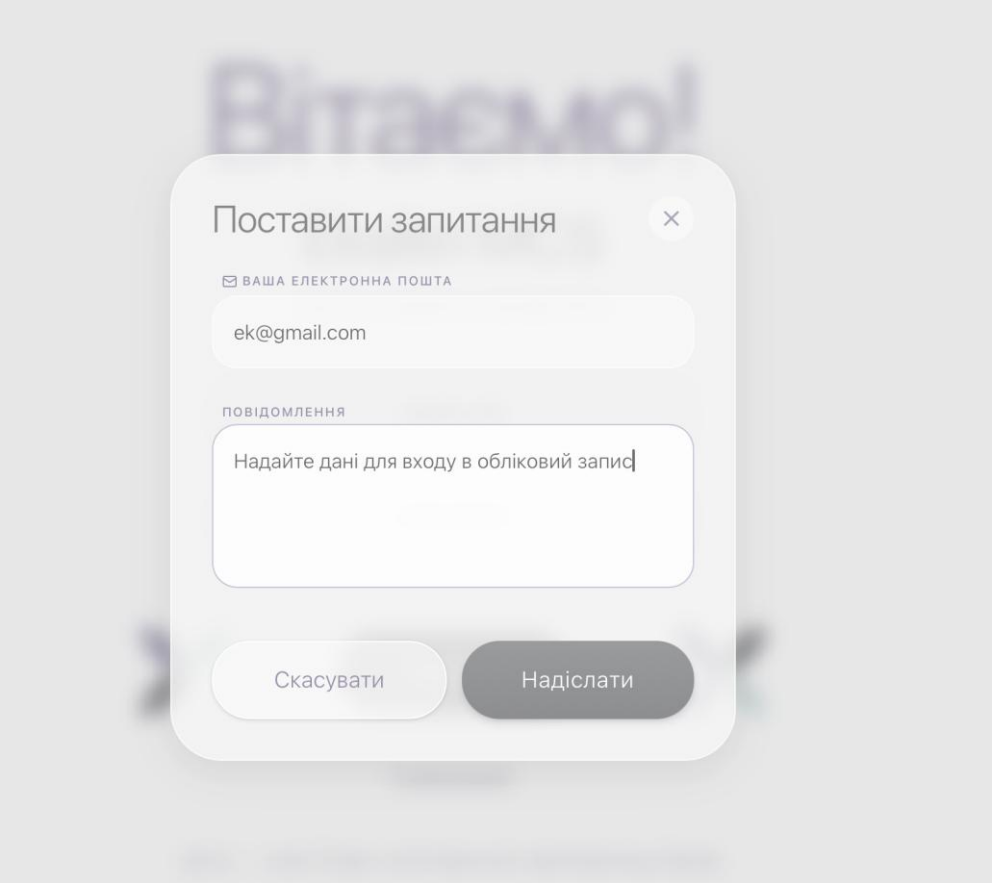

Рисунок 5.2 - Перехід до форми запиту доступу

У формі запиту доступу користувач заповнює основну інформацію про себе: прізвище та ім’я, контактні дані, бажану роль або посаду, а також може залишити короткий коментар для адміністратора. Після заповнення форми користувач надсилає запит на розгляд (рис. 5.3). Система не створює обліковий запис автоматично, а лише передає інформацію відповідальній особі.

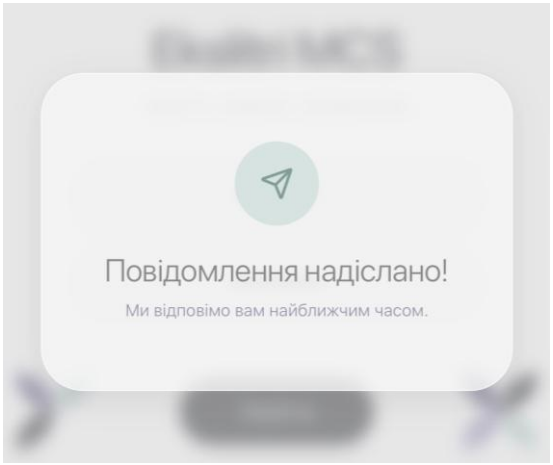

Рисунок 5.3 – Форма подання запиту на доступ

Після надсилання запиту адміністратор отримує повідомлення на електронну пошту (рис. 5.4). У листі містяться дані користувача, який подав заявку, та інформація, необхідна для прийняття рішення. Адміністратор може перевірити дані працівника, створити для нього обліковий запис, призначити роль і надати відповідні права доступу.

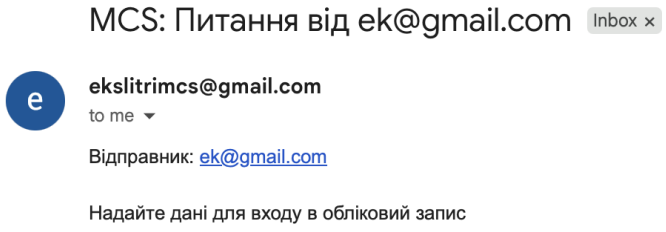

Рисунок 5.4 - Лист адміністратору із запитом на доступ

Такий підхід є важливим для безпеки системи, оскільки доступ до виробничих, складських та персональних даних не може надаватися без підтвердження відповідальної особи. Крім того, адміністратор одразу контролює, до якої ролі має належати новий користувач і які функції системи йому потрібно відкрити.

5.2.2 Інтерфейс адміністратора

Після успішної авторизації адміністратор переходить до адміністративної панелі CRM-системи. Вона є основним робочим середовищем для керування підприємством і містить навігацію між ключовими модулями. На головному екрані розміщено меню, через яке можна перейти до розділів обліку працівників, графіків, виробництва, складу, статистики, звітів, документації та аналітики.

Адміністративна панель містить короткі інформаційні блоки, які дозволяють швидко оцінити поточний стан підприємства (рис. 5.5). На такому екрані доцільно відображати кількість працівників на зміні, виробничі показники за день, інформацію про виконання плану, наявність проблемних процесів, а також основні повідомлення системи.

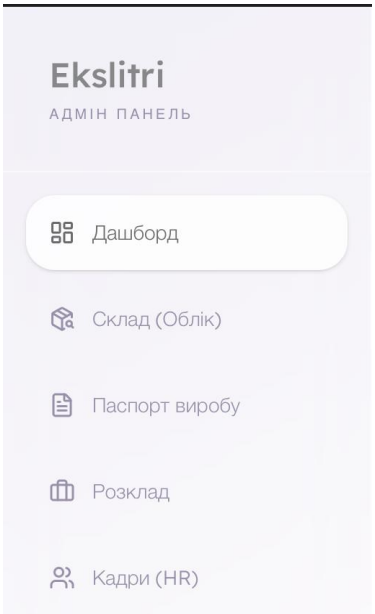

Рисунок 5.5 - Головна панель адміністратора

Головна сторінка адміністратора - це дашборд, на якому відображаються ключові показники роботи підприємства. У верхній частині показано короткі статистичні картки: кількість придатної продукції, кількість браку, кількість активних змін і кількість активних працівників у штаті (рис. 5.6). Такі показники

дозволяють швидко оцінити поточний стан виробництва без переходу до окремих таблиць.

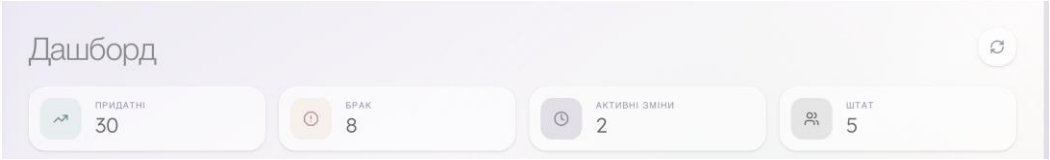

Рисунок 5.6 - Статистичні картки адміністративного дашборду

У вкладці продуктивності адміністратор бачить графічну аналітику за виробітком. Система відображає співвідношення придатної продукції та браку, а також продуктивність окремих працівників. Нижче розміщується таблиця, у якій для кожного працівника показано кількість придатних одиниць, кількість браку та відсоток якості (рис. 5.7).

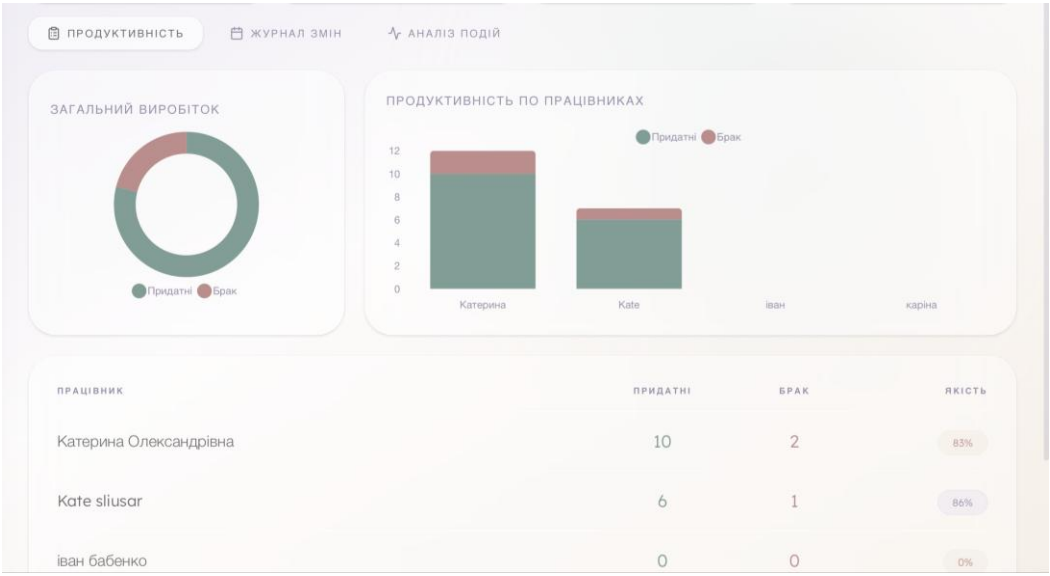

Рисунок 5.7 - Аналітика продуктивності працівників

Окремою вкладкою у дашборді є журнал змін (рис. 5.8). У цьому модулі адміністратор може переглядати робочі зміни працівників, їхній час початку та

завершення, статус зміни, а також редагувати або видаляти помилкові записи. Для зручності передбачено фільтри за датою та статусом: весь час, сьогодні, вчора, поточний місяць, активні зміни, призупинені та завершені зміни.

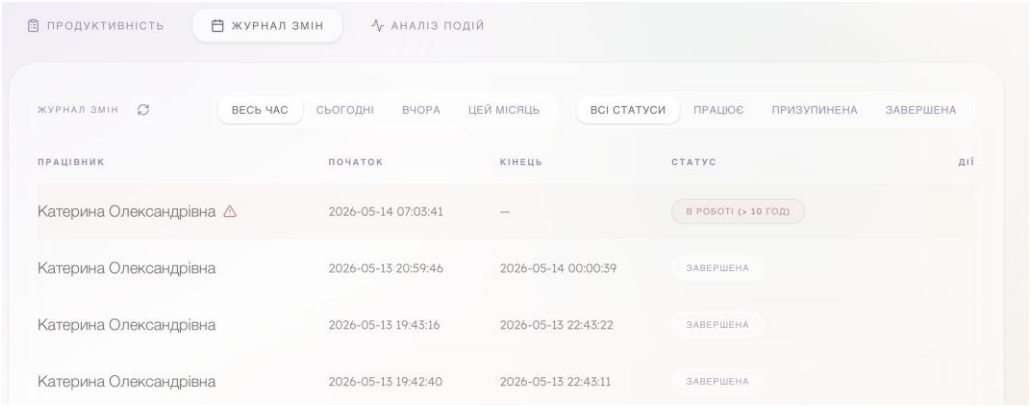

Рисунок 5.8 - Журнал робочих змін

Якщо зміна триває занадто довго, система візуально позначає її як підозрілу. Це допомагає адміністратору знайти ситуації, коли працівник забув завершити зміну або дані були внесені некоректно. У разі потреби адміністратор може відкрити форму редагування зміни, змінити час початку, час завершення або статус зміни (рис. 5.9).

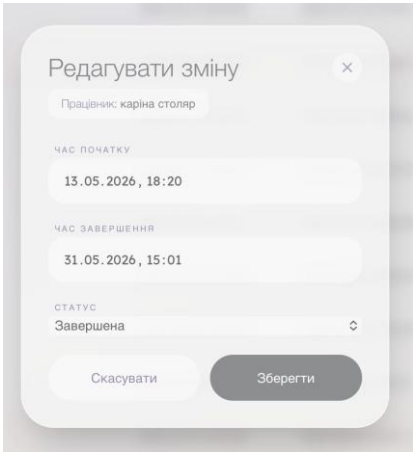

Рисунок 5.9 - Редагування робочої зміни адміністратором

У вкладці аналізу подій система відображає інформацію про причини браку, запізнення, раннє завершення зміни, затримки після завершення робочого часу та паузи (рис. 5.10). Такий модуль дозволяє не просто зберігати виробничі факти, а аналізувати причини проблем. Наприклад, якщо працівник вніс брак, він повинен вказати причину дефекту. Надалі ці причини доступні адміністратору для аналізу.

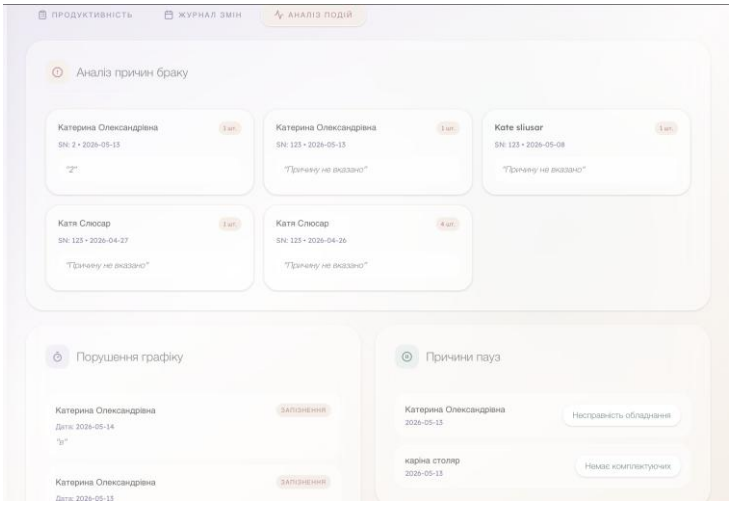

Рисунок 5.10 - Аналіз виробничих подій

Кадровий модуль призначений для керування працівниками підприємства (рис. 5.11). У ньому адміністратор бачить список активних працівників, їхні посади,

логіни та електронні адреси. Адміністратор може додати нового працівника, відредагувати його дані або перевести працівника в архів.

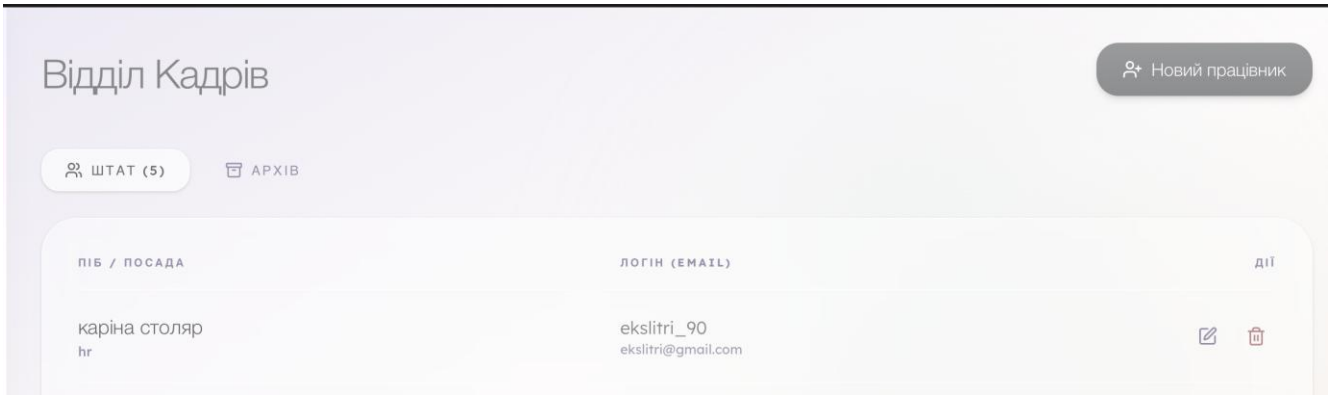

Рисунок 5.11 - Кадровий модуль системи

Під час створення нового працівника адміністратор вказує ім’я, прізвище, email, посаду, логін, пароль і роль користувача (рис. 5.12). Якщо пароль не задано вручну, система може згенерувати його автоматично. Це спрощує створення нових облікових записів і зменшує ризик дублювання логінів.

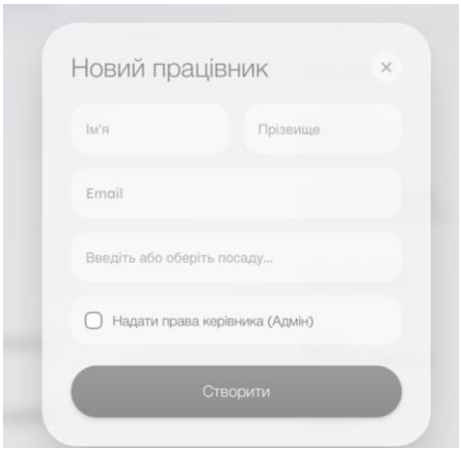

Рисунок 5.12 - Додавання нового працівника

Якщо працівник більше не працює в системі, адміністратор не видаляє його повністю, а переводить в архів із зазначенням причини (рис. 5.13). Це дозволяє

зберегти історію його змін, виробітку та виробничих подій, але заблокувати подальший доступ до системи.

Рисунок 5.13 - Архів працівників

5.2.3 Модуль розкладу та планування

Модуль розкладу використовується для планування роботи працівників на поточний або наступний тиждень (рис. 5.14). Адміністратор може переглядати, хто з працівників подав графік, які дні є робочими, а які позначені як вихідні.

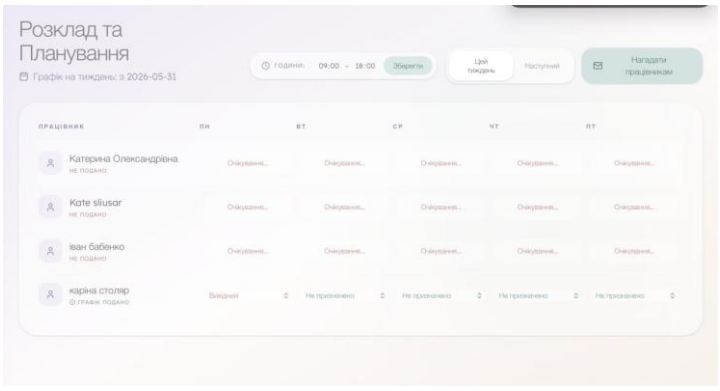

Рисунок 5.14 - Модуль розкладу та планування

У верхній частині модуля адміністратор може вибрати тиждень, встановити

загальні години роботи для працівників, а також надіслати нагадування тим, хто ще не подав графік. Нагадування надсилається на електронну пошту працівника. Це дозволяє зменшити кількість ситуацій, коли працівники забувають подати графік на наступний тиждень.

Після того як працівник подав графік, адміністратор може призначити йому конкретний виробничий процес на кожний робочий день. Наприклад, для одного працівника може бути призначено процес збірка, для іншого - пайку, тестування або інший етап виробництва (рис. 5.15). Список процесів формується на основі паспортів виробів, створених у системі.

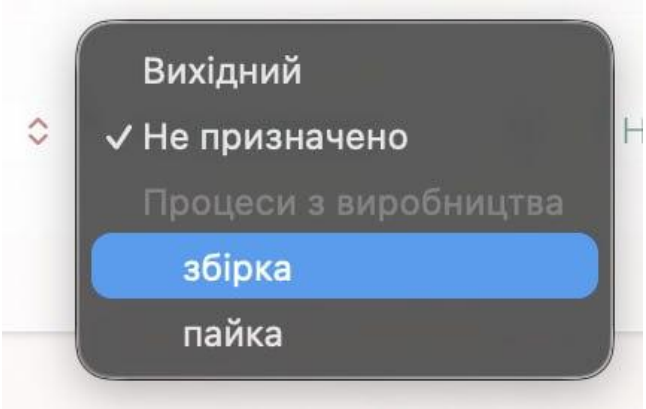

Рисунок 5.15 - Призначення виробничого процесу працівнику

Якщо працівник ще не подав графік, у таблиці відображається статус очікування. Це дає адміністратору змогу швидко побачити, які графіки вже доступні для планування, а які ще потребують уточнення.

5.2.4 Модуль паспортів виробів і прогнозування

Модуль паспортів виробів призначений для опису технологічної структури продукції (рис. 5.16). У ньому адміністратор може створювати нові вироби,

редагувати існуючі та додавати до них послідовність виробничих процесів.

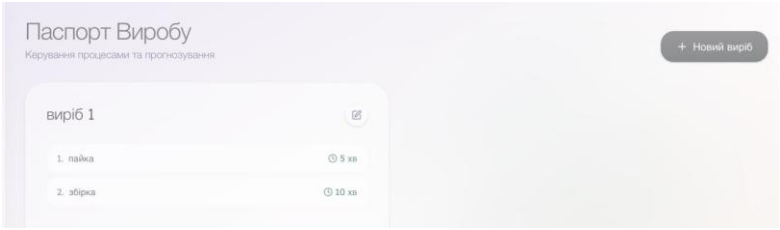

Рисунок 5.16 - Модуль паспортів виробів

Під час створення паспорта виробу адміністратор вказує назву виробу, його опис, а також перелік технологічних етапів. Для кожного етапу задається назва процесу, норма часу в хвилинах на одну одиницю продукції та порядок виконання (рис. 5.17). Наприклад, для виробу можуть бути задані процеси: збірка, пайка, тестування.

Рисунок 5.17 - Створення технологічної карти виробу

Після створення паспорта виробу система дозволяє виконати прогнозування строків виробництва. Для цього адміністратор вводить планову кількість одиниць продукції та натискає кнопку розрахунку. Система аналізує технологічні процеси виробу, норми часу та фактичний розподіл працівників за виробничими процесами у поданих графіках.

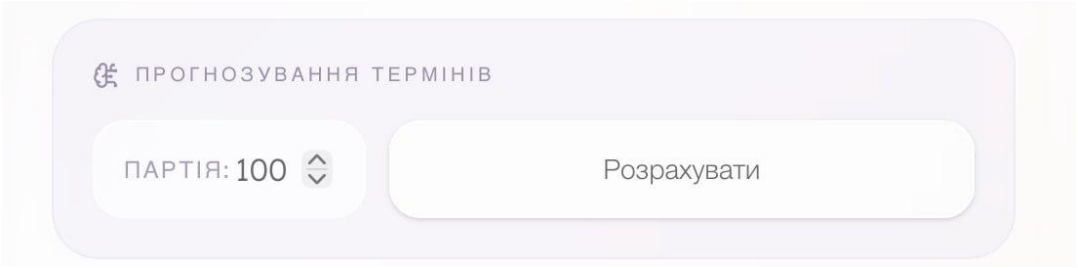

Рисунок 5.18 - Запуск прогнозування строків виробництва

Результат прогнозування відображається у вигляді окремого вікна (рис. 5.19). У ньому показується орієнтовна кількість робочих днів для виконання партії, навантаження по кожному процесу, фактична кількість працівників, оптимальна кількість працівників та можливе вузьке місце виробництва.

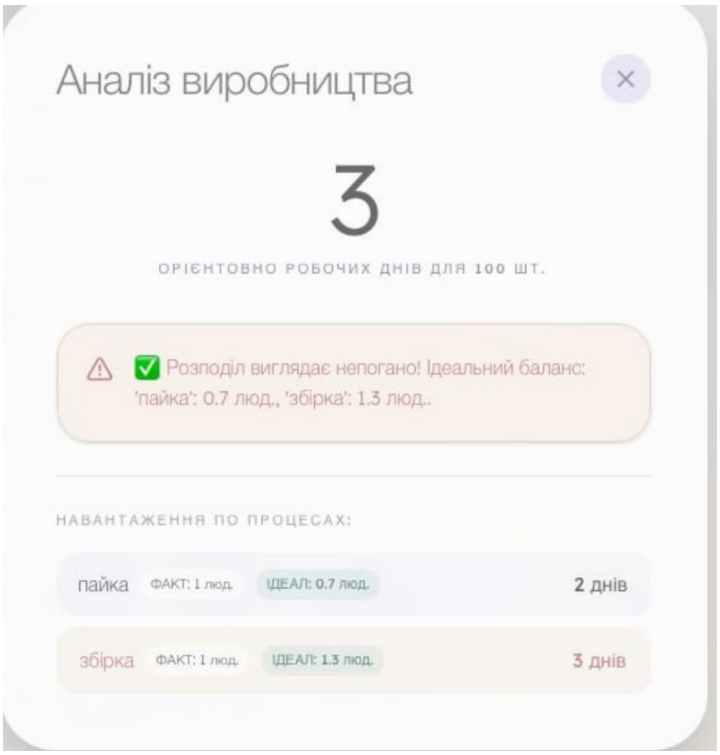

Рисунок 5.19 - Результат прогнозування виробництва

Якщо на певний процес не призначено жодного працівника, система показує попередження, що виконання партії неможливе. Якщо працівники призначені, але один із процесів потребує значно більше часу, ніж інші, система визначає його як

вузьке місце та пропонує перерозподілити персонал. Якщо розподіл працівників є збалансованим, система показує позитивне повідомлення та рекомендований оптимальний розподіл.

Окремо в модулі передбачено блок конвеєрної аналітики (рис. 5.20).

Рисунок 5.20 - Конвеєрна аналітика виробу

Він дозволяє отримати зведення за виробничими показниками: кількість придатних одиниць, кількість браку, відсоток якості та деталізацію по процесах Якщо рівень якості нижчий за допустимий поріг, система виводить попередження про необхідність перевірки виробничої лінії.

5.2.5 Складський модуль

Складський модуль призначений для обліку комплектуючих, готової продукції, складських операцій, комплектів і виробничих планів (рис. 5.21). Він дозволяє адміністратору контролювати залишки та виконувати операції приходу, списання й виробництва комплектів.

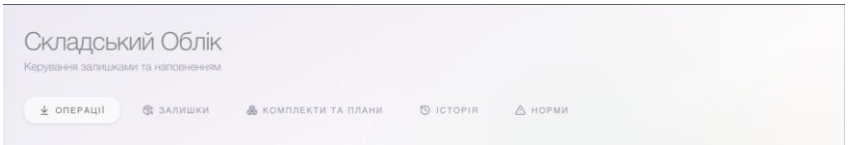

Рисунок 5.21 - Складський модуль системи

У вкладці руху товару користувач може вибрати тип операції: внести товар на склад, винести товар зі складу або виробити комплект (рис. 5.22). Для операцій приходу та списання вказується назва товару, кількість і примітка. Якщо додається новий товар, система пропонує вибрати категорію. Це дозволяє одразу структурувати номенклатуру складу.

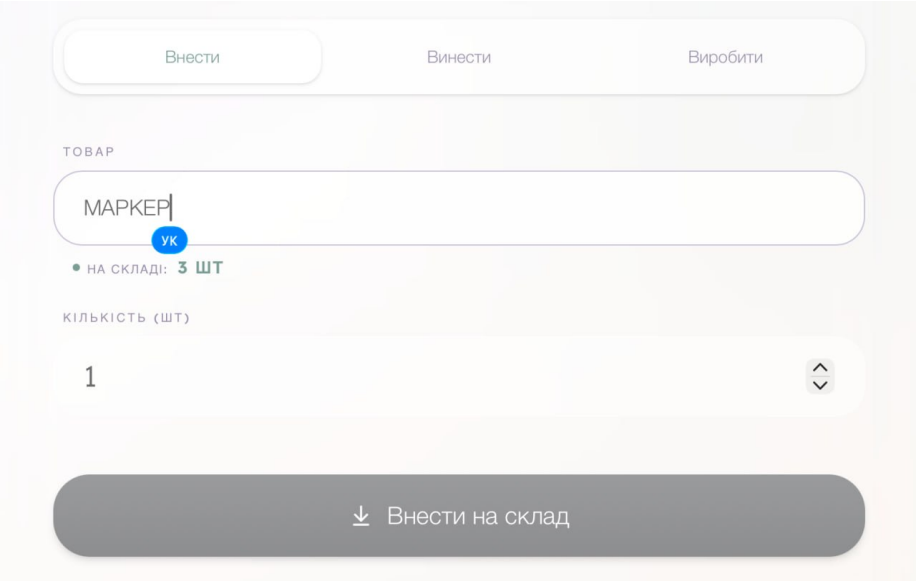

Рисунок 5.22 - Виконання складської операції

У вкладці залишків відображається таблиця товарів, їхні категорії та поточна кількість на складі (рис. 5.23). Для зручності реалізовано пошук за назвою товару та фільтрацію за категоріями. Якщо для товару задано мінімальну норму, а

фактичний залишок менший за цю норму, система візуально позначає дефіцит.

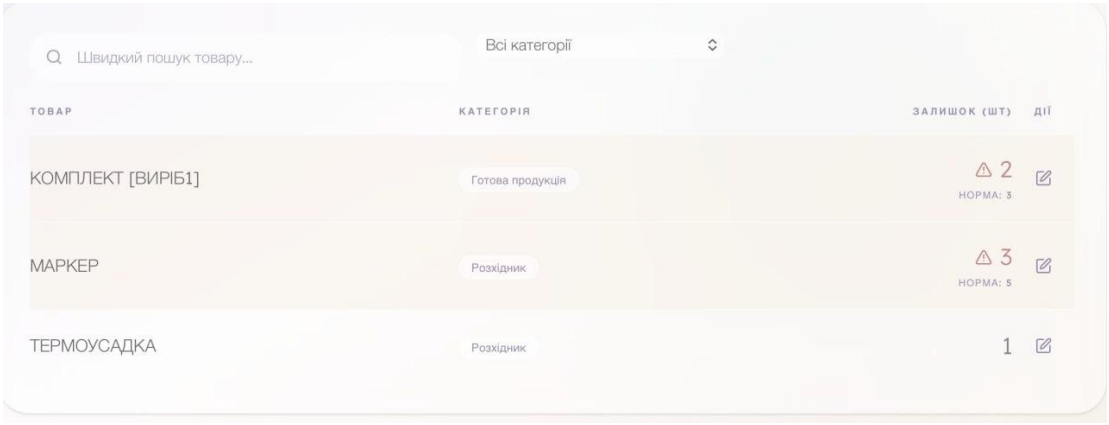

Рисунок 5.23 - Перегляд складських залишків

Адміністратор може редагувати назву товару та категорію. При зміні назви система оновлює пов’язані записи в нормах, компонентах комплектів та історії операцій. Це важливо, оскільки дозволяє уникнути розриву зв’язків між складськими даними.

Окремо реалізовано роботу з комплектами. Адміністратор може створити комплект, додати до нього перелік компонентів і вказати кількість кожної складової (рис. 5.24). Під час операції виробництва система перевіряє, чи достатньо комплектуючих на складі. Якщо хоча б одного компонента недостатньо, операція не виконується, а користувач отримує повідомлення про нестачу.

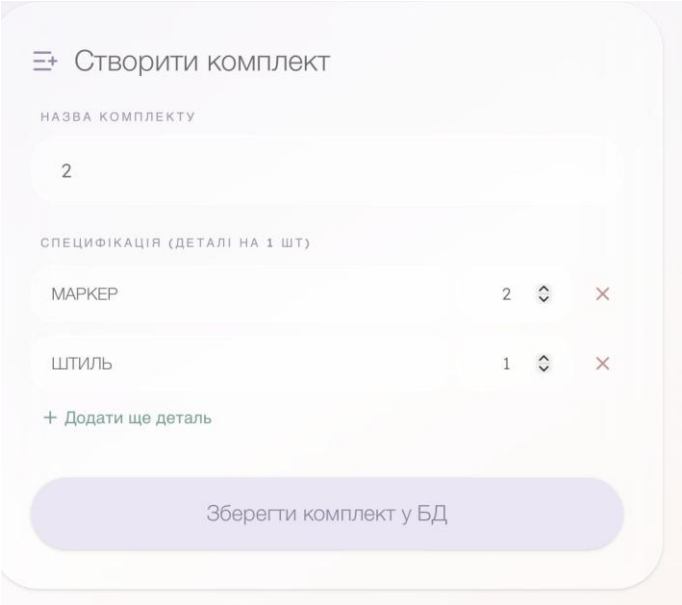

Рисунок 5.24 - Створення комплекту з компонентів

Після успішного виробництва комплекту система автоматично списує компоненти зі складу та додає готову продукцію. Усі операції записуються до історії руху товарів. Це дозволяє відстежити, хто виконав операцію, коли вона була виконана, який товар був змінений і в якій кількості.

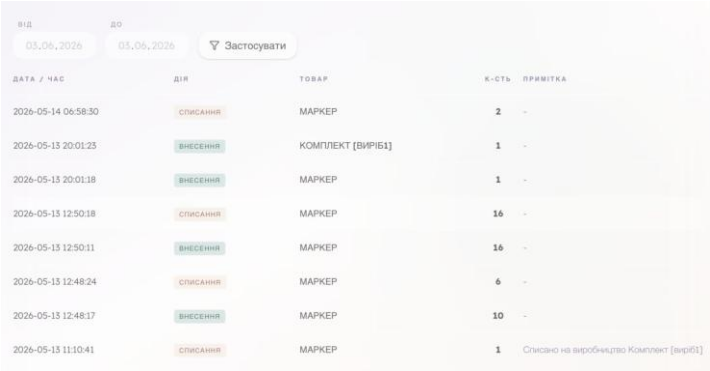

Рисунок 5.25 - Історія складських операцій

Також у складському модулі можна задавати мінімальні норми залишків. Якщо після списання або виробництва кількість товару стає нижчою за норму,

система формує попередження про дефіцит і надсилає повідомлення адміністраторам на електронну пошту.

5.2.2 Інтерфейс працівника

Після входу працівника відкривається персональний кабінет (рис. 5.26). Його інтерфейс простіший, ніж адміністративна панель, оскільки містить лише ті функції, які потрібні працівнику під час робочого дня.

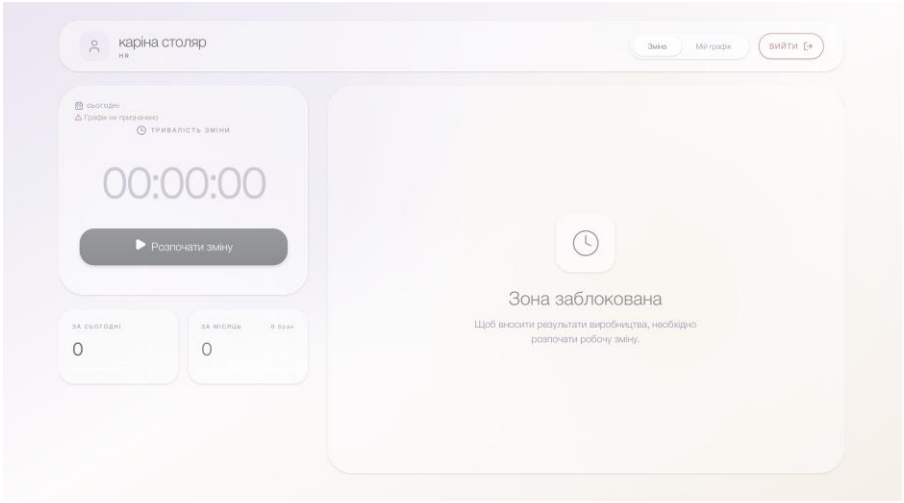

Рисунок 5.26 - Персональний кабінет працівника

У верхній частині кабінету відображається ім’я працівника, його посада, кнопка виходу із системи та перемикач між вкладками Зміна і Мій графік. Основна вкладка призначена для початку зміни, постановки зміни на паузу, завершення зміни та внесення результатів виробництва.

Якщо зміна ще не розпочата, зона внесення результатів заблокована. Це зроблено для того, щоб працівник не міг вносити виробіток без відкритої робочої зміни.

Після натискання кнопки початку зміни система створює запис у таблиці shifts. Якщо працівник починає зміну пізніше запланованого часу більше ніж на 10 хвилин, система відкриває форму введення причини запізнення. Після підтвердження причина зберігається, а адміністратор отримує повідомлення про запізнення працівника (рис. 5.27).

Рисунок 5.27 - Фіксація причини запізнення

Під час активної зміни працівник може вносити результати виробництва. Для цього він зазначає серійний номер, кількість придатних одиниць та кількість браку (рис. 5.28).

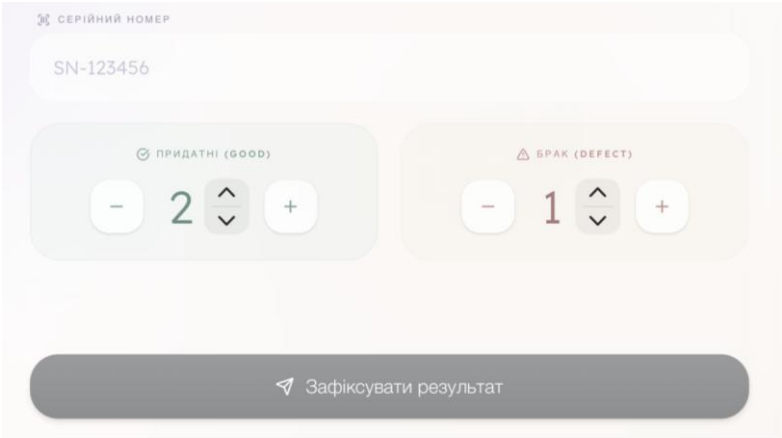

Рисунок 5.28 - Внесення результатів виробництва

Якщо кількість браку більша за нуль, система відкриває додаткову форму, у якій необхідно описати причину дефекту (рис. 5.29).

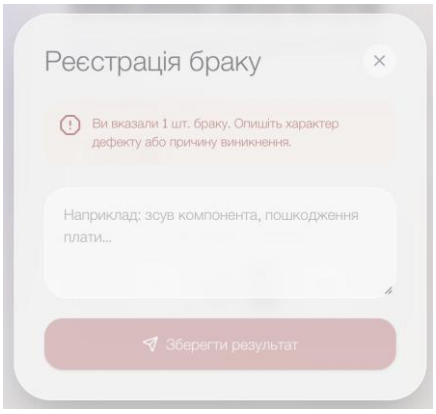

Рисунок 5.29 - Фіксація причини браку

Працівник також може поставити зміну на паузу. У системі передбачено кілька типових причин паузи: обідня перерва, очікування деталей, поломка обладнання, а також можливість ввести власну причину (рис. 5.30). Ці дані надалі відображаються в адміністративному модулі аналізу подій.

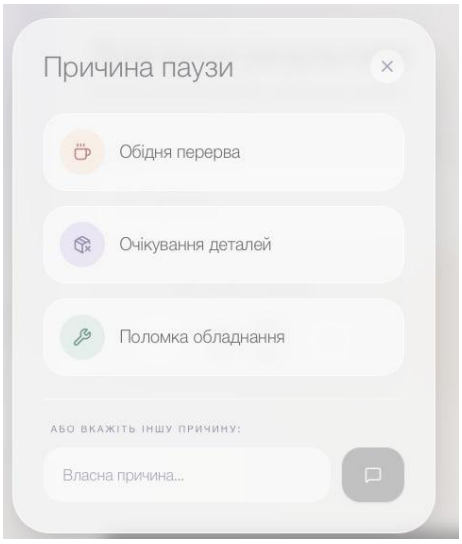

Рисунок 5.30 - Вибір причини паузи

Під час завершення зміни система перевіряє, чи було внесено результати роботи. Якщо працівник не зафіксував жодного результату, завершення зміни

блокується. Також система перевіряє час завершення відносно графіка. Якщо працівник завершує зміну раніше або затримується більше ніж на 30 хвилин після запланованого часу, потрібно вказати причину.

У вкладці Мій графік працівник може переглядати поточний або наступний тиждень. Якщо графік ще не подано, працівник вибирає робочі та вихідні дні й надсилає графік до системи (рис. 5.31). Після подання графіка адміністратор може призначити працівнику конкретні виробничі процеси.

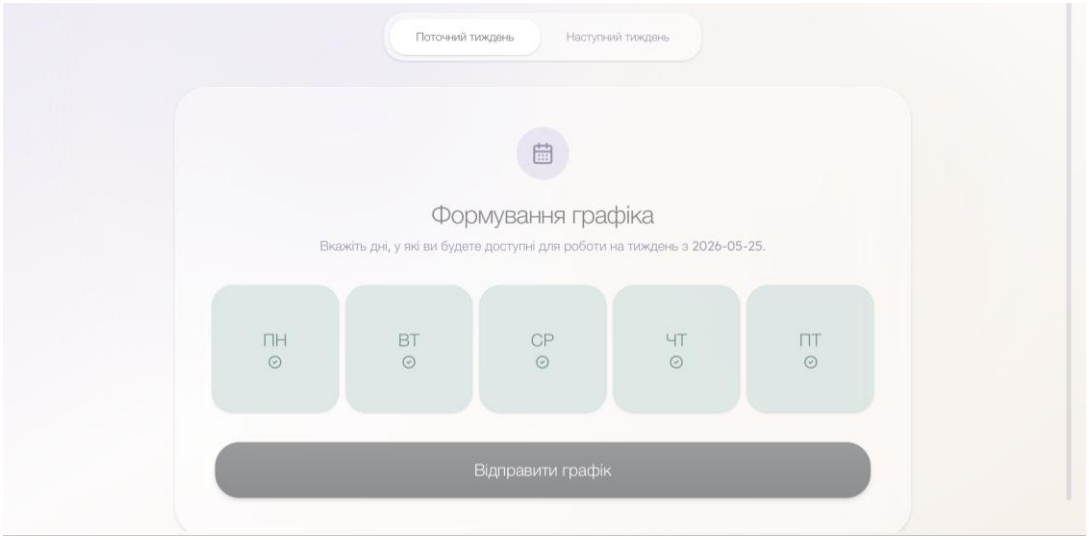

Рисунок 5.31 - Подання графіка працівником

Система також має механізм офлайн-буфера. Якщо під час виконання дії зникає з’єднання з мережею, зміни, які передбачають запис даних, можуть бути тимчасово збережені локально. Після відновлення з’єднання система автоматично синхронізує накопичені запити із сервером. Це підвищує надійність роботи у виробничих умовах, де інтернет-з’єднання може бути нестабільним.

5.3 Практичний сценарій використання

У таблиці 5.2 наведено типовий сценарій використання аналітичного модуля прогнозування.

Таблиця 5.2 - Сценарій використання модуля прогнозування

Крок	Дія користувача	Дія системи
1	Адміністратор створює виріб	Система зберігає запис у таблиці products
2	Адміністратор додає технологічні процеси	Система зберігає записи у product_processes
3	Працівники подають графік	Система зберігає дані у weekly_schedules
4	Адміністратор призначає процеси працівникам	Система оновлює графіки та закріплює процеси за днями
5	Адміністратор вводить планову кількість	Frontend формує запит на прогнозування
6	Backend виконує розрахунок	Система визначає строк виконання, навантаження та вузьке місце
7	Адміністратор переглядає результат	На екрані показано прогноз, деталізацію та рекомендацію

Спочатку адміністратор створює паспорт виробу, наприклад «Виріб 1». У паспорті він додає технологічні етапи: пайка, збірка, тестування. Для кожного етапу задається норма часу на одну одиницю продукції.

Після цього працівники подають свої графіки на тиждень. Адміністратор переглядає подані графіки та призначає працівникам конкретні процеси. Саме ці призначення потім використовуються системою для розрахунку доступної кількості працівників на кожному етапі виробництва.

Коли потрібно оцінити термін виконання партії, адміністратор відкриває модуль паспортів виробів, вводить планову кількість продукції та запускає прогнозування. Frontend формує запит до серверної частини. Backend отримує процеси виробу з таблиці product_processes, аналізує тижневі графіки з таблиці weekly_schedules, визначає кількість працівників на кожному процесі та розраховує орієнтовну кількість робочих днів.

Якщо на один із процесів не призначено жодного працівника, система показує значення нескінченності для цього етапу та повідомляє, що виконання партії неможливе без призначення персоналу. Якщо всі процеси мають працівників, система визначає найтриваліший етап як потенційне вузьке місце. Якщо різниця між найшвидшим і найповільнішим процесом значна, система пропонує перерозподілити працівників.

Таким чином, аналітичний модуль допомагає адміністратору не лише фіксувати виробничі дані, а й приймати управлінські рішення. Завдяки прогнозуванню можна заздалегідь побачити ризик затримки, змінити графік працівників або посилити проблемний етап виробництва.

ВИСНОВКИ

У дипломній роботі розроблено CRM-систему з аналітичним модулем прогнозування ефективності виробництва. Система поєднує функції обліку працівників, робочих змін, тижневих графіків, виробів, технологічних процесів, складських операцій, виробничих результатів і статистики. Основною особливістю є наявність модуля прогнозування, який дозволяє оцінити строк виконання виробничого плану та визначити вузьке місце.

Під час виконання роботи було проаналізовано предметну область виробничого управління і визначено ключові проблеми: розпорошеність даних, складність контролю графіків, потреба у швидкому аналізі виробітку та відсутність автоматизованої оцінки майбутнього навантаження. На основі цього сформовано вимоги до системи та визначено ролі користувачів.

Для реалізації обрано клієнт-серверну архітектуру. Клієнтська частина створена на React, серверна частина - на FastAPI, а для збереження даних використовується MySQL. Такий стек технологій забезпечує модульність, зрозумілу структуру, можливість розширення і зручну взаємодію через REST API.

Спроєктовано модель даних, що включає таблиці працівників, змін, графіків, виробів, технологічних процесів, виробничих журналів, складських залишків, історії операцій і комплектів. Завдяки зв’язкам між сутностями система може використовувати одні й ті самі дані для оперативного обліку, статистики і прогнозування.

Розроблено алгоритм прогнозування, який враховує планову кількість продукції, норми часу технологічних процесів і фактичний розподіл працівників за графіками. Алгоритм визначає орієнтовну кількість днів для кожного процесу, знаходить вузьке місце і формує рекомендацію щодо оптимізації розподілу персоналу.

Проведено опис основних сценаріїв тестування: авторизація, робота з працівниками, змінами, виробами, складом, записом виробітку та прогнозуванням.

Результати показують, що система здатна підтримувати основні процеси виробничого підприємства і може бути використана як основа для подальшого розвитку.

Подальший розвиток системи може включати розширення прав доступу, додавання детальнішої аналітики, інтеграцію з обладнанням або сканерами, покращення механізму безпеки, використання змінних середовища для конфігурації, створення звітів у форматі Excel або PDF та впровадження більш складних моделей прогнозування на основі історичних даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. FastAPI documentation : офіційна документація. URL: https://fastapi.tiangolo.com/ (дата звернення: 25.05.2026).

2. React documentation : офіційна документація. URL: https://react.dev/ (дата звернення: 25.05.2026).

3. Vite documentation : офіційна документація. URL: https://vite.dev/ (дата звернення: 25.05.2026).

4. MySQL 8.0 Reference Manual : офіційна документація. URL: https://dev.mysql.com/doc/ (дата звернення: 25.05.2026).

5. Pydantic documentation : офіційна документація. URL: https://docs.pydantic.dev/ (дата звернення: 25.05.2026).

6. Chart.js documentation : офіційна документація. URL: https://www.chartjs.org/docs/latest/ (дата звернення: 25.05.2026).

7. Recharts documentation : офіційна документація. URL: https://recharts.org/ (дата звернення: 25.05.2026).

8. MDN Web Docs. HTTP overview : офіційна документація. URL: https://developer.mozilla.org/ (дата звернення: 25.05.2026).

9. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures : doctoral dissertation. Irvine : University of California, 2000.

10. Sommerville I. Software Engineering. 10th ed. Pearson, 2016.

11. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. 9th ed. McGraw-Hill, 2019.

12. ISO/IEC/IEEE 29148:2018. Systems and software engineering — Life cycle processes — Requirements engineering.

13. ISO/IEC/IEEE 12207:2017. Systems and software engineering — Software life cycle processes.

14. Матеріали переддипломної практики за темою «CRM-система з аналітичним модулем прогнозування ефективності виробництва». Київ, 2026.

ДОДАТОК А

Фрагменти коду основних складових застосунку

«CRM-система з аналітичним модулем прогнозування ефективності виробництва»

УКР.НТУУ «КПІ ім. Ігоря Сікорського»_ІАТЕ_ЦТЕ_ТР_21

Аркушів 4

Київ — 2026

Програмні засоби, використані у застосунку:

– мова програмування клієнтської частини – JavaScript / JSX;

– бібліотека для створення інтерфейсу – React;

– інструмент збирання frontend-застосунку – Vite;

– стилізація інтерфейсу – Tailwind CSS;

– мова програмування серверної частини – Python;

– фреймворк для створення API – FastAPI;

– система керування базами даних – MySQL;

– підключення до бази даних – mysql.connector;

– обмін даними між клієнтом і сервером – HTTP-запити у форматі JSON;

– побудова графіків та аналітичних блоків – Chart.js / Recharts;

– локальне збереження даних користувача – localStorage.

Реалізація вибору інтерфейсу залежно від ролі користувача

Цей фрагмент демонструє логіку розмежування доступу до системи залежно від ролі авторизованого користувача. Якщо користувач ще не увійшов у систему, йому відображається форма авторизації. Після успішного входу система перевіряє роль: адміністратор переходить до адміністративної панелі, а працівник - до персонального кабінету.

```
if (!user) return ;
if (user.role === 'admin') {
return ;
}
return ;
```

Реалізація авторизації користувача

Фрагмент відповідає за перевірку логіна або електронної пошти та пароля користувача. Сервер звертається до таблиці працівників і шукає відповідний активний обліковий запис. Якщо дані введено неправильно або користувач неактивний, система повертає повідомлення про помилку та не надає доступ до функціоналу.

```
@app.post("/login")
def login(req: LoginRequest):
conn = get_db_connection()
cursor = conn.cursor(dictionary=True)
cursor.execute(
    """
    SELECT id, first_name, last_name, role, is_active
    FROM employees
    WHERE (username = %s OR email = %s) AND password = %s
    """,
    (req.username, req.username, req.password)
)

user = cursor.fetchone()

if not user:
    raise HTTPException(
        status_code=401,
        detail="Невірний логін або пароль"
    )

if not user["is_active"]:
    raise HTTPException(
        status_code=403,
        detail="Обліковий запис неактивний"
    )

return user
```

Реалізація алгоритму прогнозування ефективності виробництва

Цей фрагмент відображає основну логіку роботи аналітичного модуля прогнозування. Система проходить по кожному технологічному процесу, визначає кількість призначених працівників і розраховує орієнтовний час виконання етапу. Якщо на процес не призначено жодного працівника, він автоматично визначається як проблемний.

```
for process in processes:
workers = worker_distribution.get(process.name, 0)
total_minutes = target_quantity * process.norm_minutes
if workers <= 0:
    bottleneck = process.name
    warning = "На процес не призначено працівників"
else:
    days_needed = ceil(total_minutes / (workers * 480))
    update_bottleneck_if_needed(process.name, days_needed)
```

Реалізація формування рекомендації за результатами прогнозування

Фрагмент демонструє формування текстової рекомендації для адміністратора. Якщо система виявляє процес без працівників, вона пропонує призначити персонал. Якщо один із процесів значно сповільнює виробництво, система рекомендує перерозподілити працівників для вирівнювання навантаження.

```
if warning:
suggestion = (
"Необхідно призначити працівників "
"на процес: " + bottleneck
)
elif max_days > min_days * 2:
suggestion = (
"Процес '" + bottleneck +
"' гальмує виробництво. "
```

```
"Доцільно перерозподілити працівників."
)
else:
suggestion = "Розподіл працівників є збалансованим."
```

Реалізація офлайн-буфера клієнтської частини

Цей фрагмент показує механізм збереження запитів у разі тимчасової втрати інтернет-з’єднання. Якщо користувач виконує дію, що змінює дані, але мережа недоступна, запит тимчасово зберігається в локальному буфері. Після відновлення з’єднання система автоматично синхронізує накопичені запити із сервером.

```
if (!navigator.onLine && isModificationRequest) {
queueRequest(url, options);
return { ok: true, isOffline: true };
}
window.addEventListener('online', () => {
syncQueue();
});
```

ДОДАТОК Б

Апробація

Матеріали міжнародної наукової конференції SCIENTIA

«Scientific method: reality and future trends of researching»

«Development of a CRM System with an Analytical Module for Predicting Production Efficiency»

УКР.НТУУ «КПІ ім. Ігоря Сікорського»_ІАТЕ_ЦТЕ_ТР_21

Аркушів 2

Київ — 2026

Sliusar Kateryna Igorivna
4th-year student of the Department of Digital Technologies in Energy
National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Ukraine

Scientific director: Volkov Oleksandr Volodymyrovych
Department of Digital Technologies in Energy
National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute», Ukraine

DEVELOPMENT OF A CRM SYSTEM WITH AN ANALYTICAL MODULE FOR PREDICTING PRODUCTION EFFICIENCY

Modern manufacturing enterprises increasingly require comprehensive information systems that allow viewing the production process as a single controlled structure [4]. In traditional approaches, employee schedules, inventory balances, and production reports are often stored in separate tables or recorded manually. This complicates analysis, creates a risk of data duplication, and reduces the speed of making managerial decisions. Under these conditions, a CRM system can act as a centralized platform for managing personnel, shifts, production operations, and warehouse analytics [5].

The purpose of the study is the development of a CRM system with an analytical module for predicting production efficiency. Such a system is designed to automate the tracking of employees, work shifts, production results, and warehouse operations [6].

The general structure of the software product is implemented based on a client-server architecture. The server part is developed using the FastAPI framework, which is well-suited for creating REST APIs and supports data models through Pydantic [1]. The client interface is built with React, enabling a component-based approach and convenient user state management [2]. A relational MySQL database is used to store structured data regarding employees, products, technological processes, schedules, and production logs [3].

The developed system divides user interactions into administrator and worker roles. The administrator has access to the full functionality, including personnel management, warehouse control, and forecasting, while the worker operates within a personal cabinet to submit schedules, log shifts, and record production results.

A key feature of the software is the analytical forecasting module. Its algorithm calculates the estimated time required to manufacture a specified quantity of products. The calculation is based on the technological processes of the product, the

time standard for each stage in minutes, and the submitted weekly schedules of the workers. The system calculates the necessary duration in days for each stage, and the process with the longest duration is identified as the production "bottleneck". Furthermore, the module generates textual recommendations for optimization, such as suggesting the reallocation of workers from the fastest process to the critical one to balance the workflow.

In conclusion, the developed CRM system eliminates the fragmentation of manufacturing data by providing a single source of relevant information. The integration of the analytical module allows management to not only monitor the current state of production but also proactively evaluate future workloads, identify critical limitations, and promptly adjust the distribution of personnel to successfully fulfill production plans [7].

References:

1. FastAPI Documentation [Electronic resource]. – Available at: https://fastapi.tiangolo.com/
2. React Documentation [Electronic resource]. – Available at: https://react.dev/
3. MySQL 8.0 Reference Manual [Electronic resource]. – Available at: https://dev.mysql.com/doc/
4. Smith, J. (2025). Integrating Microcontrollers and IoT Systems in Manufacturing Execution. Journal of Industrial Automation, 12(3), 45-58.
5. Doe, A. (2024). Technical Project Management: Agile Methodologies for Full-Stack Applications. Tech Management Review, 8(1), 112-125.
6. Johnson, M. (2025). Optimization of Warehouse Logistics using 3D Printed Components. Advanced Manufacturing Practices, 4(2), 77-89.
7. Brown, K. (2026). Economic Efficiency and Forecasting in Serial Production. Journal of Business and Financial Economics, 15(4), 210-225.