МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Таврійський державний агротехнологічний університет
імені Дмитра Моторного
Факультет енергетики і комп'ютерних технологій

ЗАТВЕРДЖУЮ
Зав. каф. "Комп'ютерні науки"
доц.___ _____Сергій ШАРОВ
«16»_червня__2025 р.

Пояснювальна записка

до кваліфікаційної роботи здобувача СВО Бакалавр
(ступінь вищої освіти)

на тему: «Локальна інформаційна система моніторингу та управління виробничими процесами на підприємстві»

52/4КНД.11323044.000000ПЗ

Виконав: здобувач вищої освіти__2__курсу, групи 21С(МС)КН
спеціальності 122 Комп'ютерні науки
за ОПП Комп'ютерні науки
(шифр і назва спеціальності та ОПП)
_____Олександр МЕЛЬНИЧУК
(підпис)

Керівник доц._____Дмитро ЛУБКО
(підпис)

Консультант_____Лариса БОЛТЯНСЬКА
(підпис)

Консультант_____Михайло ЗОРЯ
(підпис)

Нормоконтроль, ст.викл._____Ольга ЗИНОВ'ЄВА
(підпис)

Рецензент_____Олекій НАУМУК_____
(підпис)

Запоріжжя – 2025 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Таврійський державний агротехнологічний університет імені Дмитра Моторного

Факультет: Енергетики і комп'ютерних технологій
Кафедра: Комп'ютерні науки
Ступінь вищої освіти: Бакалавр
Спеціальність: 122 Комп'ютерні науки
ОПП: Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри_КН
к.пед.н., доц. _____Сергій ШАРОВ
"10" _жовтня__2024__ року

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

__Мельничуку Олександру Олександровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи__«Локальна інформаційна система моніторингу та управління виробничими процесами на підприємстві»
керівник проєкту __Лубко Дмитро Вікторович, к.т.н., доцент_____
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом університету від "30" вересня_2024_року №_452-С
2. Строк подання здобувачем вищої освіти роботи __14 червня 2025_____
3. Вихідні дані до роботи: дані обстеження об'єкту, статистичні дані, технічне завдання на дипломне проєктування, матеріали виробничих практик, нормативні документи, науково-технічна література, електронні ресурси та ін.
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)____
___1. Опис предметної області та аналіз існуючих рішень._____
___2. Специфікація вимог до локальної інформаційної системи.
___3. Вибір та обґрунтування технологій розробки інформаційної системи.___
___4. Дослідна експлуатація інформаційної системи._____
___5. Тестування інформаційної системи._____
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників)
Діаграма варіантів використання, діаграми IDEF0, діаграма ER, прототип інтерфейсу, діаграма компонентів системи, інтерфейс програми.
Презентація – 11 слайдів

1. Титульний слайд
2. Предмет, об'єкт, мета дослідження
3. Завдання дослідження

4. Порівняльна характеристика аналогів програмного продукту
5. Діаграма варіантів використання
6. Інструментальні засоби
7. Інтерфейс і робота інформаційної системи
8. Інтерфейс і робота інформаційної системи
9. Тестування
10. Висновки
11. Дякую за увагу

6. Консультанти розділів кваліфікаційної роботи

Розділ/ Підрозділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
4.3	Болтянська Л. О.	15.10.2025	15.06.2025
4.4	Зоря М. В.	15.10.2025	15.06.2025

7. Дата видачі завдання___10 жовтня 2024 р._____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Аналіз предметної області	03.10.2024	Виконано
2	Специфікація вимог до локальної інформаційної системи	17.10.2024	Виконано
3	Проектування локальної інформаційної системи	28.10.2024	Виконано
4	Обґрунтування інструментальних засобів	18.11.2024	Виконано
5	Розробка локальної інформаційної системи	16.12.2024	Виконано
6	Тестування та налагодження локальної інформаційної системи	11.04.2025	Виконано
7	Підпис керівником роботи	14.06.2025	Виконано
8	Підпис завідувачем кафедри	16.06.2025	Виконано

Здобувач вищої освіти _____Олександр МЕЛЬНИЧУК__
(підпис) (власне ім'я та прізвище)

Керівник кваліфікаційної роботи _____Дмитро ЛУБКО_
(підпис) (власне ім'я та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра: 80 с., 18 рис., 13 табл., 21 посилання, 2 додатки.

Актуальність: підприємства важкого машинобудування потребують наскрізної цифрової підтримки для зниження витрат, мінімізації людського фактору та підвищення якості продукції. Розроблена ERP-система автоматизує увесь життєвий цикл замовлення електромашин: від прийому замовлення до відвантаження виробу, візуалізує ключові показники діяльності підприємства.

Об’єкт дослідження: процес управління виробничими операціями на підприємстві, що виготовляє важкі електромашини.

Предмет дослідження: інформаційна ERP-система та її архітектурні, функціональні й технологічні рішення.

Мета роботи: створення й упровадження адаптованої ERP-системи, яка забезпечує інтеграцію даних, контроль виробництва та автоматичне формування документації.

Завдання дослідження:

– проаналізувати існуючі IT-рішення й виявити їхні недоліки для виробництва важких електромашин;

– спроєктувати архітектуру та функціональні модулі системи;

– реалізувати програмний продукт засобами Python 3.10, PySide6 та SQLite;

– провести тестування й оцінити ефективність системи;

– підготувати проект до подальшого розвитку системи.

Практичне значення полягає у можливості використання системи на виробничому майданчику, що забезпечує прозоре управління замовленнями, скорочує час випуску та підвищує конкурентоспроможність підприємства .

Ключові слова: ERP-СИСТЕМА, ВИРОБНИЦТВО, ВАЖКІ ЕЛЕКТРОМАШИНИ, АВТОМАТИЗАЦІЯ, PYTHON, PYSIDE6.

SUMMARY

Bachelor's qualification work: 80 p., 18 fig., 13 tab., 21 references, 2 appendices.

Relevance: heavy-machinery factories need end-to-end digital support to cut costs, eliminate human error and keep their product quality high. The developed ERP system automates the full order lifecycle for large electric machines: from order intake to shipment, and visualizes key performance metrics.

Object of research: the production-management process at a heavy-electrical-machinery manufacturer.

Subject of research: the ERP information system together with its architectural, functional and technological solutions.

Purpose: to design and implement a tailored ERP platform that integrates data flows, monitors production stages and auto-generates business documents.

Research objectives:

– analyze current IT solutions and identify their shortcomings for heavy-machinery production;

– design the system architecture and functional modules;

– develop the software using Python 3.10, PySide6 and SQLite;

– run comprehensive testing and evaluate techno-economic efficiency;

– outline recommendations for future system evolution.

Practical significance: the system can be deployed on the shop floor, providing transparent order control, shortening lead-time and strengthening the company’s competitiveness.

Keywords: ERP SYSTEM, MANUFACTURING, HEAVY ELECTRIC MACHINES, AUTOMATION, PYTHON, PYSIDE6.

ЗМІСТ

ВСТУП

Нинішній етап розвитку інформаційних технологій створює безпрецедентні можливості для модернізації виробничих процесів, зокрема у важкій електротехніці. Інтеграція цифрових рішень у сферу виробництва дозволяє не лише підвищити ефективність роботи підприємства, а й забезпечити якісний контроль на всіх етапах технологічного процесу.

Зростання вимог до автоматизації виробничих процесів обумовлюється не лише потребою зниження виробничих витрат, але й необхідністю мінімізації людського фактору, що є основним джерелом помилок. Розробка спеціалізованої ERP–системи для підприємства, яке виробляє важкі електромашини, дозволяє ефективно організувати роботу з замовленнями, контролювати етапи виробництва та покращити взаємодію між підрозділами підприємства.

Об’єктом дослідження є процес управління виробничими операціями на підприємстві, що спеціалізується на виготовленні важких електромашин. Це включає інтеграцію всіх етапів виробництва – від прийому замовлення до доставки готової продукції, а також взаємодію між виробничим відділом, контролем якості, логістикою та адміністративним супроводом.

Предметом дослідження є розроблена інформаційна система (ERP-платформа) та її архітектура, що дозволяє автоматизувати управління виробничими процесами. Дослідження зосереджене на моделюванні бізнес-процесів, розробці функціональних модулів системи, а також на інтеграції різних підсистем в єдине інформаційне середовище.

Метою даної роботи є розробка та підготовка до використання адаптованої локальної інформаційної системи для автоматизації управління виробництвом на підприємстві, що виробляє важкі електромашини.

Система повинна забезпечувати повний контроль над виробничими процесами, інтегрувати дані між різними підрозділами та сприяти зниженню операційних витрат за рахунок мінімізації людських помилок.

Для досягнення поставленої мети необхідно виконати наступні завдання:

– провести аналіз сучасних інформаційних систем та визначити їх недоліки з огляду на специфіку виробництва важких електромашин;

– розробити концепцію та архітектуру системи з урахуванням вимог до автоматизації виробничих процесів;

– визначити функціональні модулі та розробити програмний продукт із застосуванням технологій Python, Qt PySide6 та SQLite;

– забезпечити тестування розробленої системи для виявлення та усунення помилок, а також перевірити її ефективність у реальних умовах експлуатації;

– оцінити практичну значущість використання системи, підготувати систему до подальшого розвитку.

Дипломна робота складається з чотирьох основних розділів. У першому розділі проведено аналіз предметної області, розглянуто сучасні тенденції цифровізації виробництва та проаналізовано існуючі аналогічні рішення. У другому розділі розроблено глосарій, визначенні функціональні та нефункціональні вимоги та складено концептуальну модель використання системи. Третій розділ присвячено розробці, реалізації та впровадженню програмного продукту. У четвертому розділі наведено інструкцію користувача, методику тестування продукту, оцінено техніко-економічні характеристики ІС та проконтрольовано техніку безпеки.

Дана робота є комплексним дослідженням, спрямованим на розробку ефективного інструменту управління виробничими процесами. Результати дослідження мають як теоретичну, так і практичну цінність, що підтверджується успішною розробкою системи та її потенціалом для подальшого використання в промисловій сфері.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Узагальнена характеристика предметної області

Предметна область охоплює управління виробництвом важких електромашин, зокрема генераторів та електродвигунів середньої, низької та високої напруги. Процес виробництва таких виробів є складним та багатоступеневим, що вимагає ретельного контролю на кожному етапі для забезпечення високої якості продукції. Для ефективного керування випуском продукції необхідно забезпечити тісну взаємодію між усіма підрозділами підприємства, починаючи від прийому замовлення та закінчуючи доставкою готової продукції клієнтам [20, с. 20].

Клієнтами підприємства є великі промислові об’єкти, такі як кар’єри, металургійні заводи, атомні електростанції, великі харчові виробництва та підприємства видобувної промисловості. Особливістю роботи з такими замовниками є високі вимоги до якості продукції та точності дотримання термінів поставок. Отже, ключовими аспектами управління виробництвом є своєчасний контроль кожного етапу виробництва та забезпечення швидкої комунікації між відділами підприємства.

1.1.1 Завдання, що вирішуються в предметні області

Підприємства важкого машинобудування, зокрема ті, що займаються випуском електромашин та супутнього обладнання, стикаються з рядом специфічних завдань, серед яких:

– автоматизація процесу обробки замовлень;

– контроль якості та стандартизація виробництва;

– оптимізація логістичних процесів;

– аналітика та прийняття управлінських рішень;

– інтеграція даних та взаємодія між різними підрозділами підприємства.

Всі зазначені завдання вимагають використання сучасних програмних рішень, які забезпечують автоматизацію рутинних операцій, мінімізують вплив людського фактору та підвищують загальну продуктивність підприємства. Для вирішення таких завдань використовуються ERP-системи [17].

1.1.2 Огляд програмних рішень у використанні

До використання ERP-платформи управління могло здійснюватися через системи Odoo або SAP ERP, але через складність впровадження, санкції на певні продукти, спричинені політичною ситуацією, та потребу у більш гнучкому та адаптованому до специфіки підприємства рішенні, було вирішено розробити власну інформаційну систему. Запропоноване рішення має мету забезпечити:

– повний контроль над виробництвом – можливість відстеження кожного замовлення в режимі реального часу;

– зв’язок між підрозділами – автоматизований обмін даними між відділами за допомогою єдиної бази даних;

– автоматизацію процесів – зменшення кількості рутинних операцій, таких як заповнення звітів вручну;

– збільшення ефективності комунікації – впровадження внутрішнього чату для інженерів, менеджерів та виробничих працівників;

– генерацію документації – формування договорів, накладних та інших документів у напівавтоматичному режимі.

Розробка нового програмного забезпечення має на меті не тільки автоматизацію стандартних процесів, але й забезпечення інтегрованої платформи, здатної адаптуватися до потреб підприємства, забезпечувати комплексний аналіз даних та підтримувати високий рівень якості обслуговування внутрішніх та зовнішніх користувачів.

1.1.3 Актуальність розробки спеціалізованого програмного забезпечення

Актуальність розробки сучасного інформаційного продукту для виробничих підприємств визначається кількома ключовими чинниками:

– цифровізація виробничих процесів;

– підвищення якості управління;

– аналітична підтримка прийняття рішень;

– гнучкість та масштабованість;

– інноваційність та використання сучасних технологій.

У сучасних умовах цифрова трансформація стає запорукою конкурентоспроможності підприємства [14]. Автоматизація та інтеграція даних дозволяють не лише зменшити операційні витрати, але й значно підвищити якість продукції та оперативність прийняття управлінських рішень.

Впровадження єдиної інформаційної системи дає підприємству можливість у режимі реального часу контролювати всі етапи виробництва — від замовлення до доставки, підвищуючи якість продукції та знижуючи ризик помилок.

Інтеграція аналітичних інструментів дозволяє глибоко аналізувати дані, оцінювати ефективність підрозділів, прогнозувати потреби ринку та оперативно реагувати на зміни. Сучасні програмні продукти повинні бути адаптивними й масштабованими, щоб забезпечувати довгострокову конкурентну перевагу.

Нове програмне забезпечення дозволяє впроваджувати інноваційні підходи до управління, зокрема мобільні додатки, веб-сервіси та аналітичні модулі, що сприяє автоматизації та інтеграції з іншими сучасними рішеннями.

Запропонована система оптимізує процеси й забезпечує високий рівень контролю за всіма етапами виробництва, що важливо для якості продукції та конкурентоспроможності на ринку.

1.2 Огляд і аналіз існуючих аналогів системи

Для управління випуском продукції та обліку замовлень на підприємствах використовуються різні інформаційні системи, такі як ERP (Enterprise Resource Planning), MRP (Material Requirements Planning) та CRM (Customer Relationship Management) [6]. Вони дозволяють контролювати всі етапи виробництва, покращувати комунікацію між відділами й автоматизувати рутинні завдання.

Буде проведено огляд найпопулярніших систем, які використовуються на організацій, подібних до розглянутої у проєкті, а також порівняння їхніх функціональних можливостей. Особлива увага приділяється системам Odoo, SAP ERP та MRPeasy як найбільш поширеним рішенням. На ринку є й інші лідери, які не розглядатимуться, оскільки вони не підходять для використання на українських підприємствах через санкції або невідповідність специфіці виробництва електродвигунів.

1.2.1 Огляд основних аналогів

Odoo – це модульна ERP-система з частково відкритим кодом, що дозволяє налаштувати функціонал під потреби конкретного підприємства. Вона включає модулі для управління виробництвом, обліку замовлень, контролю складів та комунікації з клієнтами (рис. 1.1).

Переваги:

– гнучкість завдяки модульній архітектурі;

– відкритий код дозволяє адаптувати систему під специфічні потреби;

– вбудовані інструменти для аналітики та формування звітів.

Недоліки:

– висока складність налаштування та впровадження;

– потрібен висококваліфікований IT-персонал для підтримки системи;

– деякі модулі доступні лише на платній основі.

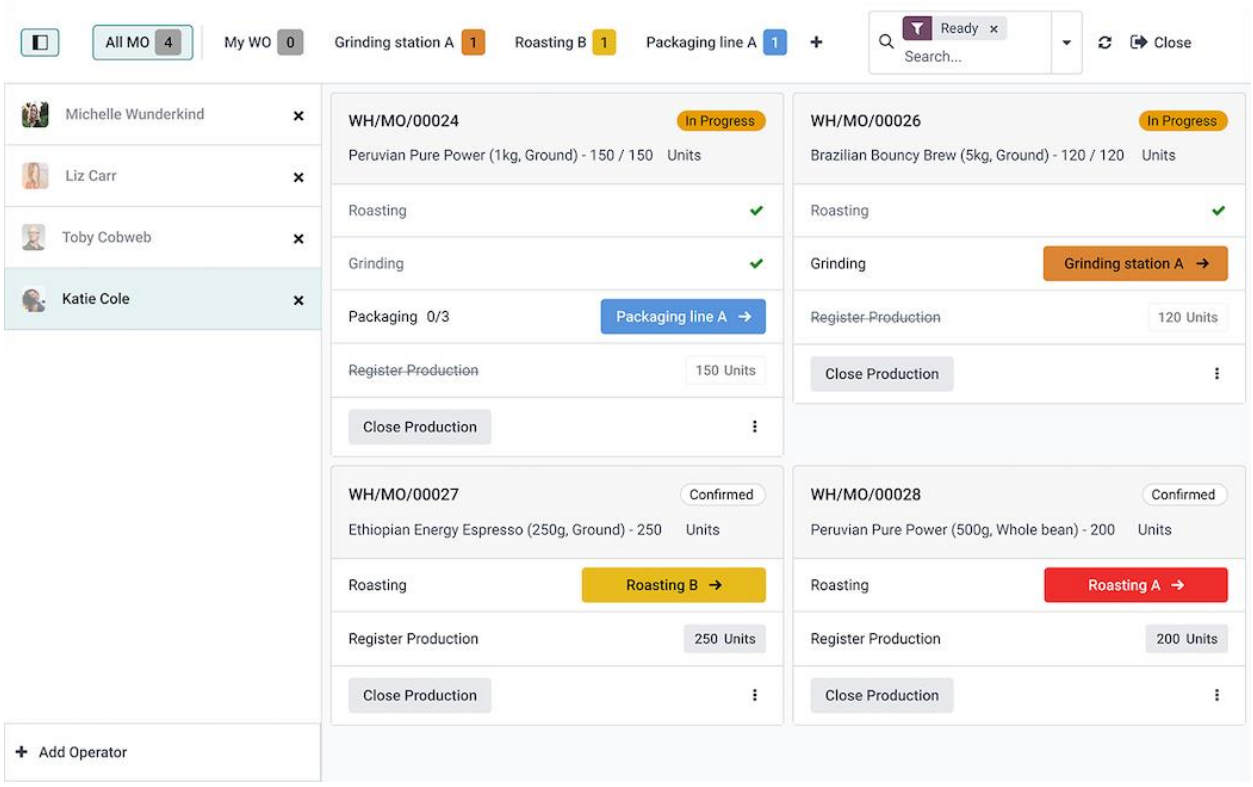

Рисунок 1.1 – Користувацький інтерфейс Odoo

Для підприємства, яке розглядається у проєкті, Odoo може бути корисною завдяки модулю Manufacturing, що дозволяє відстежувати всі етапи виробництва та контролювати виконання замовлень. Проте через складність налаштування та вартість ліцензій впровадження цієї системи може бути невиправдано комплексним та дорогим.

SAP ERP – це одна з найбільш відомих та потужних систем для управління бізнес-процесами великих підприємств. Вона забезпечує інтеграцію всіх функцій компанії, включаючи виробництво, логістику, фінанси та управління персоналом (рис. 1.2).

Переваги:

– висока надійність та масштабованість;

– широкий функціонал для великих виробничих підприємств;

– можливість інтеграції з іншими корпоративними системами.

Недоліки:

– висока вартість впровадження та підтримки;

– складність адаптації під специфічні потреби;

– довготривалий процес навчання персоналу.

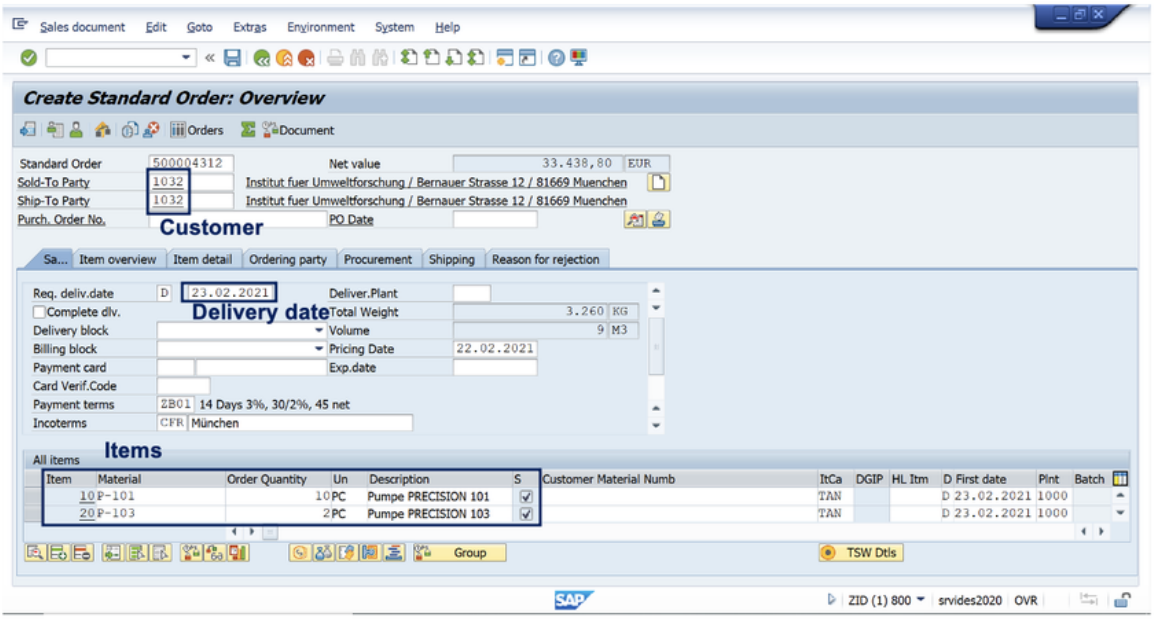

Рисунок 1.2 – Користувацький інтерфейс SAP ERP

Для заводу спеціалізованого на випуску важких електричних машин SAP може забезпечити контроль над усіма бізнес-процесами, але вартість та складність впровадження роблять цю систему недоцільною для середніх підприємств.

MRPeasy – це хмарна MRP-система, розроблена спеціально для малих та середніх виробничих підприємств. Вона дозволяє управляти замовленнями, контролювати запаси та планувати виробництво (рис. 1.3).

Переваги:

– сучасний та інтуїтивно зрозумілий інтерфейс;

– доступність завдяки хмарному рішенню та відносно невисокі вартості;

– швидке впровадження без необхідності залучення IT-фахівців.

Недоліки:

– обмежена функціональність для великих підприємств;

– обмежена можливість налаштування під специфічні потреби;

– залежність від інтернет-з'єднання через хмарну інфраструктуру.

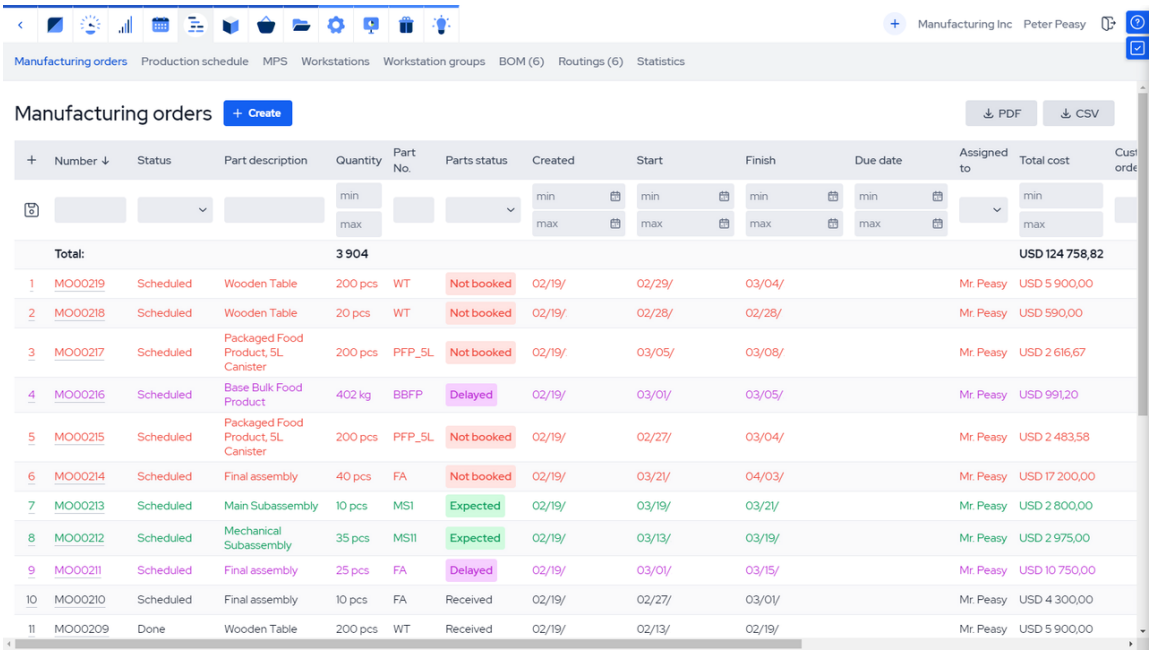

Рисунок 1.3 – Користувацький інтерфейс MRPeasy

Для середнього підприємства, яке розглядається в даному проєкті, MRPeasy може бути досить ефективним рішенням завдяки простоті використання та доступності. Проте обмежена функціональність та можливість масштабізації цієї системи не дозволяє повністю автоматизувати складні виробничі процеси та контролювати специфічні параметри продукції.

1.2.2 Аналіз існуючих аналогів обґрунтування вибору власної системи

Аналіз розглянутих систем дозволяє зробити такі висновки:

– Odoo ERP та SAP ERP є надто складними та дорогими для підприємства середнього розміру. Їхнє впровадження потребує значних інвестицій та тривалого навчання персоналу;

– MRPeasy, хоча і є більш доступним та простим у використанні, не має достатньої функціональності для управління виробництвом важких електромашин;

– інші учасники ринку, такі як Katana MRP, NetSuite ERP, Epicor ERP та Infor CloudSuite, також мають певні обмеження або надмірний функціонал, який не буде потрібен для даного типу підприємства.

Основними недоліками існуючих рішень є:

– висока вартість впровадження та підтримки систем для середніх та підприємств;

– обмежені можливості налаштування під специфічні процеси виробництва;

– складність інтеграції з існуючими бізнес-процесами підприємства;

– залежність від інтернет-з'єднання для хмарних рішень, що може бути критичним для безперервного виробництва.

З огляду на ці недоліки було прийнято рішення розробити власну ERP – систему з урахуванням специфіки розглянутої організації. Основними перевагами власного рішення є:

– адаптація до виробничих процесів: система спроєктована таким чином, щоб повністю відповідати послідовності етапів виготовлення електродвигунів – від збору замовлення до відправки готової продукції;

– простота використання: інтерфейс системи розроблено з урахуванням потреб працівників, які не мають спеціальних навичок роботи з комп’ютерними системами;

– автоматизація процесів: система автоматизує контроль за кожним етапом виробництва, що дозволяє зменшити кількість ручної роботи та мінімізувати ризик помилок;

– контроль якості: вбудовані інструменти для фіксації результатів тестування та перевірки якості забезпечують високу надійність продукції;

– зручна комунікація: вбудований чат дозволяє співробітникам швидко обмінюватися інформацією та координувати дії без використання сторонніх месенджерів;

– керування розробкою та поступовою мастшабізацією програмного інструменту у разі її потреби безпосередньо у межах підприємства;

– зменшення витрат: власна система дозволяє уникнути витрат на ліцензії та абонентську плату за використання сторонніх програмних продуктів.

Аналіз існуючих аналогів свідчить, що ринкові рішення мають як свої переваги, так і серйозні недоліки. У зв’язку з цим, розробка спеціалізованої ERP–системи, яка враховує специфіку виробництва важких електромашин та дозволяє гнучко адаптувати бізнес-процеси під потреби конкретного підприємства, є не лише доцільною, а й стратегічно важливою для підвищення ефективності управління виробництвом [21].

1.3 Розробка технічного завдання

На підставі обґрунтованих параметрів, розробляється технічне завдання на створення програмного забезпечення. Воно містить технічне завдання на розробку ERP–системи для автоматизації управління виробничими процесами на підприємстві важкого машинобудування. Дослідження охоплює основні аспекти розробки системи – від визначення її призначення до критеріїв прийому готової роботи.

1.3.1 Призначення та сфера застосування

Розроблювана система створена для автоматизації управління виробничими процесами на підприємствах, що займаються виготовленням важких електромашин та електродвигунів.

Основна мета – забезпечення інтеграції інформації між усіма підрозділами підприємства, що дозволяє:

– автоматизувати процес прийому, обробки та виконання замовлень;

– забезпечити оперативний моніторинг виробничих стадій;

– зменшити ризики людських помилок автоматизацією рутинних операцій;

– скоротити час виробництва та знизити витрати на супутні процеси.

Система орієнтована на середні підприємства машинобудування, де виробництво електромашин, електродвигунів і супутнього обладнання є ключовим напрямком діяльності. Вона застосовується для:

– контролю за виробничими процесами від складання до відвантаження готової продукції;

– управління документами: створення друк договорів, накладних, звітів;

– підтримки комунікації між співробітниками через вбудований чат;

– проведення аналітики та прийняття управлінських рішень на основі оперативних даних.

1.3.2 Вимоги до інтерфейсу

Інтерфейс системи має відповідати наступним вимогам:

– візуальна привабливість: застосування сучасних графічних елементів, плавних анімацій, контрастних кольорів для забезпечення читабельності.

– інтуїтивність та зручність: меню та навігація повинні бути логічно структуровані, забезпечуючи швидкий доступ до основних функцій системи;

– адаптивність до стандартного робочого обладнання: інтерфейс повинен коректно відображатися на моніторах стандартних розмірів із врахуванням сучасних тенденцій UX/UI;

– інформаційна насиченість: розподіл інформації за логічними блоками для швидкого сприйняття та ефективної взаємодії з системою.

1.3.3 Технічні вимоги

Основні технічні вимоги до системи включають:

– розробка системи проводиться на мові Python із застосуванням сучасної IDE PyCharm;

– використання PySide6 для створення графічного інтерфейсу;

– робота з легковісною СКБД SQLite та базою даних, приведеною до третьої нормальної форми;

– інтеграція з бібліотекою python-docx для автоматизованої генерації документів;

– застосування бібліотеки Qt Charts для побудови графіків та діаграм;

– модель побудови ПЗ із чітким розподілом функціональних модулів (автентифікація, управління замовленнями, виробничий контроль, документообіг, аналітика);

– забезпечення високої модульності для спрощення подальшої підтримки та розширення системи;

– реалізація шифрування критичних даних;

– реалізація системи логування дій користувачів;

– реалізація попереднього завантаження та звільнення пам’яті для швидкодії програми.

1.3.4 Етапи розробки

Процес розробки системи передбачає наступні етапи:

– аналіз вимог та планування: збір інформації про виробничі процеси підприємства, формулювання основних вимог та затвердження концепції проекту;

– проектування архітектури та бази даних: розробка структурної схеми системи, проектування бази даних, визначення таблиць, зв’язків та нормалізація даних;

– прототипування інтерфейсу користувача: створення та пошук перших прототипів UI з урахуванням вимог зручності та адаптивності;

– реалізація функціональних модулів: виведення функціоналу програми в окремі модулі для поліпшення досвіду розробки та користування, інтеграція модулів в єдину систему;

– інтеграційне тестування та оптимізація: перевірка сумісної роботи всіх компонентів, виправлення помилок та оптимізація швидкодії;

– впровадження та підготовка супровідної документації: остаточне налаштування системи, створення інструкції та технічної документації.

1.3.5 Критерії прийому роботи

Готовий проект вважається прийнятим за умови дотримання наступних критеріїв:

– функціональна відповідність: реалізовано всі основні та додаткові функції згідно з технічним завданням, кожен модуль працює коректно, забезпечуючи автоматизацію виробничих процесів;

– якість інтерфейсу: інтерфейс користувача є інтуїтивно зрозумілим, адаптивним та відповідає сучасним стандартам UI/UX, всі форми введення та відображення даних мають чітку візуальну структуру;

– продуктивність та надійність: час відгуку системи не перевищує встановлених норм, система забезпечує стабільну роботу та має налагоджену систему резервного копіювання;

– безпека: впроваджено надійну систему автентифікації та захисту даних, система стійка до основних типів атак (SQL–ін’єкції, несанкціонований доступ);

– документообіг: автоматична генерація договорів, накладних та звітів здійснюється відповідно до встановлених шаблонів, файли документів зберігаються у визначеній папці та мають коректний формат;

– тестування: проведено функціональне, інтеграційне, стресове та безпекове тестування без виявлення критичних помилок, отримано позитивні відгуки від користувачів під час пілотного запуску;

– супровідна документація: наявність технічної документації, користувацьких інструкцій та опису архітектури системи, що дозволяє забезпечити подальшу підтримку проекту.

Сформульоване технічне завдання являє собою детальний опис майбутньої системи, який визначає основні напрямки розробки, вимоги до функціональності, терміни виконання робіт та порядок проведення приймально-здавальних випробувань. Цей документ є незамінним інструментом для координації роботи розробників і замовника, що сприятиме успішному впровадженню програмного забезпечення у виробничий процес.

РОЗДІЛ 2. СПЕЦИФІКАЦІЯ ВИМОГ ДО ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Глосарій

Під час розробки системи було визначено глосарій – систематизований перелік основних термінів і понять, що використовуються у розробці та впровадженні інформаційної системи для автоматизації управління виробничими процесами. Глосарій спрямований на забезпечення уніфікованого розуміння термінології серед усіх зацікавлених сторін – розробників, замовників та користувачів. Чітко визначені поняття сприяють уникненню розпливчасті під час аналізу предметної області, побудови концептуальних моделей та формування функціональних і технічних вимог до системи.

Представлена таблиця 2.1 містить докладний опис ключових термінів, що відображають результати концептуального аналізу та слугують базою для подальшої деталізації моделі інформаційної системи.

Таблиця 2.1 – Глосарій

Термін	Опис терміну
1. Основні поняття предметної області	
Інформаційна система	Система, що забезпечує збір, зберігання, обробку та передачу інформації для підтримки прийняття рішень на підприємстві.
ERP-система	Інтегроване програмне забезпечення для управління підприємством, що об'єднує всі виробничі, комерційні та адміністративні процеси.

Продовження таблиці 2.1

Виробничий процес	Сукупність технологічних операцій, що здійснюються для виготовлення продукції, зокрема важких електромашин.
Автоматизація	Використання сучасних інформаційних технологій для оптимізації та контролю виробничих процесів, мінімізації людського фактору та підвищення ефективності.
База даних	Організована структура збереження даних, що управляється системою керування базами даних (СКБД) і використовується для зберігання інформації ІС.
Контроль якості	Система заходів, спрямованих на перевірку відповідності продукції встановленим стандартам і забезпечення високої якості виробництва.
Логістика	Організація і управління процесами доставки сировини та готової продукції, а також координація між різними підрозділами підприємства.
Інтеграція даних	Об'єднання інформації з різних джерел у єдиний інформаційний простір для забезпечення комплексного аналізу і управління.
Модуль системи	Складова інформаційної системи, яка реалізує певну функціональну можливість або бізнес-процес.
Об’єктна модель	Модель, що описує об’єкти предметної області та їх взаємодію, що використовується при розробці об’єктно-орієнтованих систем.

Продовження таблиці 2.1

2. Користувачі системи	
Адміністратор	Користувач, відповідальний за налаштування системи, управління правами доступу та моніторинг роботи інформаційної системи.
Розробник	Фахівець, відповідальний за проектування, розробку та впровадження програмного забезпечення, що реалізує функціонал інформаційної системи.
Інженер зі стандартизації та якості	Працівник, що контролює та координує виробничі процеси, забезпечуючи своєчасне виконання замовлень і підтримання стандартів якості.
Менеджер з продажу	Спеціаліст, який обробляє замовлення, веде переговори з клієнтами та контролює процес продажів, забезпечуючи зв’язок між клієнтами та виробництвом.
Виробничий менеджер	Користувач, який організовує процеси транспортування та доставки продукції, аналіз над виробничими процесами.
3. Вхідні та вихідні документи	
Замовлення	Документ, що містить інформацію про потребу клієнта у виготовленні певної продукції, включаючи технічні та комерційні параметри замовлення.
Договір купівлі-продажу	Юридичний документ, який регламентує умови купівлі-продажу продукції (наприклад, двигунів), встановлює права та обов’язки сторін, що беруть участь у операції.

Продовження таблиці 2.1

Накладна на відвантаження	Документ, який підтверджує факт відвантаження продукції, містить інформацію про перевізника, транспортний засіб, водія та дату відвантаження.
Звіт про виробництво	Документ, що відображає статистичну інформацію про стан виробничих процесів, обсяг виконаних замовлень і поточний стан виробництва на підприємстві.
Журнал дій	Документ, у якому фіксуються всі операції та події, пов’язані з роботою системи, для подальшого аналізу, контролю та аудиту.
Технічний паспорт виробу	Документ, який містить інформацію про виріб: його технічні характеристики та тонкощі виробництва.

2.2 Специфікація функціональних та нефункціональних вимог

Для забезпечення чіткого взаєморозуміння та мінімізації ризиків двозначностей у процесі розробки було прийнято рішення детально формалізувати як функціональні, так і нефункціональні вимоги до системи (табл. 2.2-2.3). Це дозволило чітко визначити кожну вимогу з унікальним ідентифікатором, що спростило їх відстеження та управління змінами в процесі розробки.

Такий підхід гарантує прозорість пріоритизації задач і забезпечує достовірне джерело даних для всіх учасників проєкту, від бізнес-сторони до тестувальників. Формалізація вимог стала фундаментом для побудови якісного процесу розробки, тестування та подальшої підтримки інформаційної системи.

Таблиця 2.2 – Функціональні вимоги системи

ID вимоги	Назва вимоги	Атрибути вимог		
		пріоритет	складність	виконавець
FR-1	Управління замовленнями із оновленням статусу	Обов’язково	Середня	Розробник
FR-2	Контроль виробничих процесів, моніторинг етапів	Обов’язково	Висока	Провідний інженер
FR-3	Автоматизована генерація документів	Обов’язково	Середня	Розробник
FR-4	Управління користувачами	Обов’язково	Середня	Адміністратор системи
FR-5	Внутрішній чат для оперативної комунікації співробітників	Рекомендовано	Низька	Розробник
FR-6	Аналітика та звітність	Рекомендовано	Середня	Data-інженер
FR-7	Експорт журналу дій у PDF	Опційно	Низька	Розробник
FR-8	API для інтеграції зі сторонніми системами	Опційно	Висока	Інтеграційна команда

Формалізація нефункціональних параметрів гарантує, що система відповідатиме очікуванням за продуктивністю, надійністю та безпекою. Це дозволяє об’єктивно вимірювати та перевіряти відповідність реалізації встановленим стандартам і технологічним обмеженням.

Таблиця 2.3 – Нефункціональні вимоги системи

№	Назва вимоги	Характеристика, цільове значення
1	Застосовність	
1.1	Час навчання користувача	≤ 4 год для базових операцій
1.2	Час відгуку до типових дій	≤ 1 с при 100 одночасних користувачах
2	Надійність	
2.1	Середній час безвідмовної роботи	≥ 720 год
2.2	Середній час відновлення	≤ 2 год
3	Робочі характеристики	
3.1	Швидкодія відкриття замовлення	≤ 800 мс
3.2	Продуктивність (пропускна здатність)	≥ 300 транзакцій/хв
4	Проєктні обмеження	
4.1	Мова програмування	Python 3.10 + PySide6
4.2	СУБД	SQLite 3, 3НФ
4.3	Цільова ОС	Windows 10
5	Документація та довідка	Інструкція користувача
6	Інтерфейси	
6.1	Інтерфейси користувача	Qt GUI, адаптований від ≥ 1366 × 768
6.2	Програмні	CSV export
6.3	Комунікаційні	TCP/IP (порт 443), TLS 1.3
7	Ліцензування	MIT-ліцензія для внутрішнього використання
8	Стандарти	PEP-8, ISO 9241-11 (UX), OWASP ASVS L2 (безпека)

2.3 Концептуальна модель використання інформаційної системи.

Діаграма варіантів використання відображає загальну картину взаємодії різних категорій користувачів із системою управління виробничими процесами. Вона демонструє, як окремі актори: адміністратор та інженер – виконують свої основні завдання через набір специфічних функцій. Для побудови діаграми використання системи було вирішено скористуватися засобами UML (рис. 2.1).

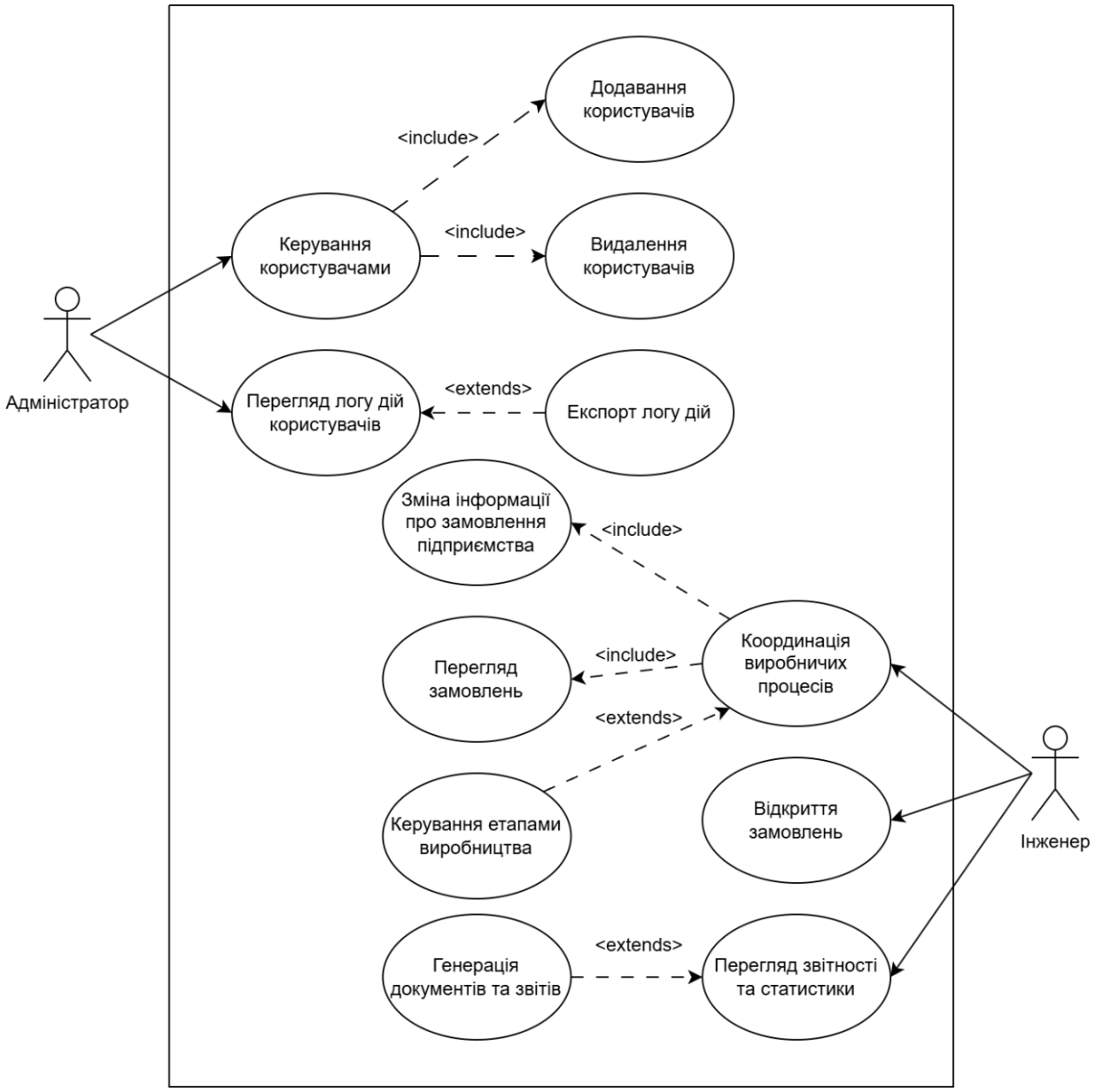

Рисунок 2.1 – Діаграма варіантів використання системи

На представленій діаграмі використано два типи зв’язків між варіантами використання: include, який вказує на обов’язкове виконання підпроцесу в рамках основної функції, та extends, що характеризує додаткове розширення основного сценарію дій.

Системний адміністратор здійснює управління користувачами та контроль за діяльністю системи, провідний інженер відповідає за обробку та координування замовлень, їх відкриття та моніторинг, має доступ до звітності та статистики. Розмежування функціональних можливостей за ролями дозволяє чітко окреслити обов’язки та забезпечити ефективну роботу системи.

Визначені таблиці 2.4–2.8 містять детальний опис основних варіантів використання системи.

Таблиця 2.4 – Керування користувачами

Контекст використання	Система надає можливість адміністратору виконувати операції зі створення та видалення користувачів, а також контролювати їх права доступу.
Дійові особи	Актор – адміністратор.
Передумова	Користувач успішно автентифікований як адміністратор.
Тригер	Вибір опцій «Додати користувача» або «Видалити користувача»
Сценарій	1. Адміністратор відкриває сторінку керування користувачами. 2. На екрані відображається список поточних користувачів. 3. Адміністратор може обрати додавання нового користувача або видалення існуючого запису.
Постумова	Дані користувачів оновлено, зміни збережено в базі даних.

Таблиця 2.5 – Керування замовленнями

Контекст використання	Система забезпечує функції для створення, редагування, відкриття та зміни інформації про замовлення підприємства, що включають роботу інженера та виробничого менеджера.
Дійові особи	Актор – інженер.
Передумова	Користувач авторизований з правами інженера, існує попередній доступний список замовлень, який сформовано та відображено у програмі.
Тригер	Обрана сторінка «Керування замовленнями»
Сценарій	1. Інженер відкриває модуль замовлень. 2. Відображаються картки замовлень з можливістю редагування та відкриття детальної інформації. 3. При виборі окремого замовлення – система надає опції для зміни інформації або відкриття повного перегляду замовлення.
Постумова	Дані про замовлення підприємства оновлено, інформація відображена згідно з останніми змінами.

Таблиця 2.6 – Експорт логу дій

Контекст використання	Система дозволяє експортувати дані логу дій у файл для подальшого аналізу або архівування.
Дійові особи	Актор – адміністратор.
Передумова	Користувач успішно авторизований як адміністратор. Лог дій сформовано та він доступний для перегляду. Фокус користувача на сторінці «Адмін-панель».
Тригер	Натискання кнопки «Експортувати лог».

Продовження таблиці 2.6

Сценарій	1. Адміністратор обирає функцію експорту логу дій. 2. Система генерує файл у форматі PDF із зібраними даними. 3. Файл зберігається на системі користувача, адміністратор отримує підтвердження операції.
Постумова	Лог дій збережено у файл, адміністратор отримує інформаційне повідомлення.

Таблиця 2.7 – Координація виробничих процесів

Контекст використання	Забезпечується моніторинг та координація етапів виробництва, що дозволяє користувачам контролювати виконання замовлень і логістику доставки продукції.
Дійові особи	Актор – інженер.
Передумова	Користувач авторизований. Замовлення в процесі виробництва.
Тригер	Обрана сторінка «Координація»
Сценарій	1. Спеціаліст відкриває розділ виробничих процесів. 2. Система відображає інформацію про поточні етапи виробництва, стан замовлень та інформацію про логістику. 3. Користувач може переглядати процес виробництва та керувати відправленнями, дослідити відповідність процесів встановленим стандартам, дозволяє перенести виріб на наступний етап виробництва.
Постумова	Замовлення переходять до наступних етапів виробництва, дані актуалізовано.

Таблиця 2.8 – Перегляд візуальної аналітики

Контекст використання	Система генерує звіти та графічну статистику, що дозволяє менеджеру аналізувати ефективність роботи підприємства та приймати рішення для оптимізації процесів.
Дійові особи	Актор – інженер.
Передумова	Дані замовлень та виробничих процесів накопичено в базі даних.
Тригер	Обрання сторінки «Статистика»
Сценарій	1. Спеціаліст обирає функцію перегляду звітності. 2. Система обчислює статистичну інформацію, яка формує графіки та з актуальними даними про продажі, виробництво та доставку. 3. Користувач може здійснювати аналіз даних за різними критеріями та періодами.
Постумова	Звіти сформовано, інформація доступна для перегляду та аналізу.

Діаграма варіантів використання дає змогу чітко окреслити інтегровану модель взаємодії між різними категоріями користувачів системи. Розподіл ролей і відповідних функціональних сценаріїв забезпечує оптимізацію робочих процесів та підвищення ефективності управління виробничими операціями. Детальний опис варіантів використання у таблицях дозволяє зрозуміти логіку взаємодії системи, що є надзвичайно важливим для подальшої розробки та практичного використання програмного продукту.

2.4 Розробка функціональної моделі

Для описання розроблюваного продукту було вирішено створити функціональну модель інформаційної системи за методологією IDEF0, яка відображає всі основні функції, що виконуються при розробці програмного забезпечення, а також потоки інформації, що забезпечують їх взаємозв’язок. Функціональна модель дозволяє систематизувати завдання, встановити взаємозв’язки між підсистемами та задокументувати перетворення вхідних даних у готові результати.

Основною функцією розроблюваної системи є моніторинг та управління виробничими процесами на підприємстві. Побудована контекстна діаграма IDEF0 представлена як єдиний функціональний блок із інтерфейсами (рис. 2.2).

Рисунок 2.2 – Контекстна діаграма розробленої системи за методологією IDEF0

Ця діаграма демонструє, що система приймає численні вхідні дані, що підлягають обробці згідно з встановленими нормативами та керівництвом, а результатом її роботи є формування звітів, статистичних даних та документів за участю користувача, програмного забезпечення та апаратних засобів.

Для детального відображення функціональних процесів основна функція розбивається на чотири підфункції, що описуються окремими блоками (прямокутниками №1-4) (рис. 2.3).

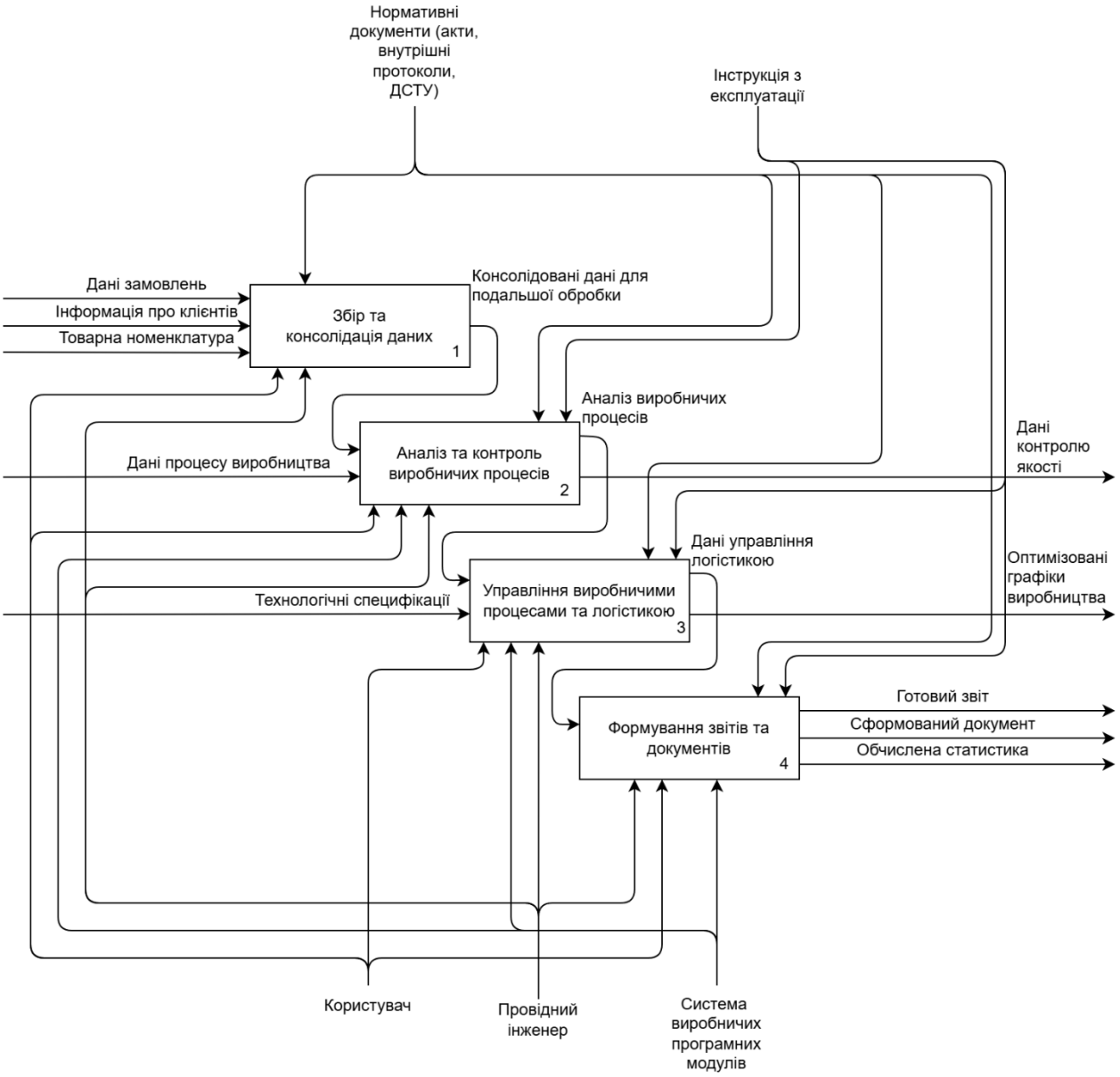

Рисунок 2.3 – Діаграма декомпозиції розроблюваної системи

Блок «Збір та консолідація даних» відповідає за первинний збір, об’єднання та попередню обробку даних із зовнішніх джерел.

Блок «Аналіз та контроль виробничих процесів» забезпечує аналіз зібраних даних та контроль якості проведення виробничих процесів. Результати аналізу передаються у блок 3, а дані контролю якості виходять із системи.

Блок «Управління виробничими процесами та логістикою» відповідає за прийняття управлінських рішень з оптимізації виробництва та логістики. Дані управління логістикою надходять у блок 4, натомість оптимізовані графіки виробництва виходять до зовнішнього середовища.

Блок «Формування звітів та документів» завершує процес, здійснюючи остаточне формування звітності та документації, що відображають результати управління виробництвом. Всі результати надходять до зовнішнього середовища та використовуються для прийняття управлінських рішень.

Розробка функціональної моделі за методологією IDEF0 дозволяє чітко визначити основні функції інформаційної системи та їх взаємозв’язки. Контекстна діаграма моніторингу та управління виробничими процесами на підприємстві демонструє загальний потік інформації: вхідні дані, що обробляються за допомогою встановлених нормативів та керівництва, з використанням наявних ресурсів, що призводить до формування звітів, статистики, оброблених даних та документів. Подальша декомпозиція цього блоку на підфункції дозволяє деталізувати процес збору, аналізу, управління та документування, що є необхідним для точного відображення робочого процесу та подальшого проектування архітектури системи.

За допомогою зазначених діаграм та їх детальної декомпозиції забезпечується повне узгодження функціональних вимог із зацікавленими сторонами та створюється надійна основа для розробки та впровадження інформаційної системи.

РОЗДІЛ 3. ОПИС ПРИЙНЯТИХ ПРОЄКТНИХ ТА ТЕХНОЛОГІЧНИХ РІШЕНЬ

3.1 Розробка об’єктної моделі

У цьому підрозділі здійснюється аналіз ключових об’єктів досліджуваної системи, визначаються їх властивості та причинно-наслідкові зв’язки, що дозволяє побудувати об’єктну модель, яка є основою для розробки програмного забезпечення. Важливою частиною даної роботи є створення UML-діаграми класів та ER-діаграми, оскільки для роботи продукту використовується база даних.

3.1.1 Аналіз системи

Аналіз вимог та процесів дозволив виділити наступні ключові об’єкти:

– користувачі: сутність, що включає дані про логін, пароль, ім’я, а також посаду, яку займає користувач у системі;

– посади: описують ролі користувачів, які мають визначені властивості та використовуються для управління доступом;

– замовлення: ключовий об’єкт, що містить інформацію про клієнта, виріб та умови замовлення;

– клієнти та двигуни: об’єкти, що слугують деталізацією даних замовлень і забезпечують контекст для генерації документів;

– виробничі етапи: окремі об’єкти, що відображають стан замовлення на певній стадії виробничого циклу;

–дії: об’єкти, які дозволяють відслідковувати активність користувачів, зокрема операції, що здійснюються в системі;

– повідомлення: об’єкт, що забезпечує комунікацію між користувачами в режимі реального часу.

3.1.2 Розробка UML-діаграми класів

На основі аналізу предметної області було створено UML-діаграму класів, (рис. 3.1) яка відображає атрибути основних класів та методи, що відповідають бізнес-логіці.

Зв’язки між класами демонструють:

– асоціацію, наприклад, «User belongs to Position», де кожен користувач має саме одну посаду, а кожна посада може бути закріплена за багатьма користувачами;

– відношення між замовленнями та виробничими етапами (1:1 взаємовиключне: одне замовлення може бути присутнім тільки в одному класі-етапі одночасно);

– залежності, наприклад, клас DocumentGenerator залежить від класів Order, Customer та Motor для генерації документів;

– наслідування і використання модулів у класі MainWindow, який наслідує додаткову функціональність.

3.1.3 Розробка ER-діаграми

ER-діаграма формалізує концептуальну модель даних через графічне відображення сутностей, їхніх атрибутів та зв’язків із кардинальністю. Вона служить структурою для аналітиків, розробників і замовника, закладає основу для перетворення логічної схеми в таблиці СУБД із їх ключами.

На основі аналізу визначено ключові сутності й встановлено між ними зв’язки: клієнт-замовлення (1:N), двигун-замовлення (1:N), замовлення-етап виробництва (1:1), користувач-посада (1:M), користувач-журнал дій (1:N). Ця структура гарантує цілісність даних і логічну послідовність обробки операцій.

Нижче наведено ER діаграму, яка наочно ілюструє логічну модель бази даних системи (рис. 3.2).

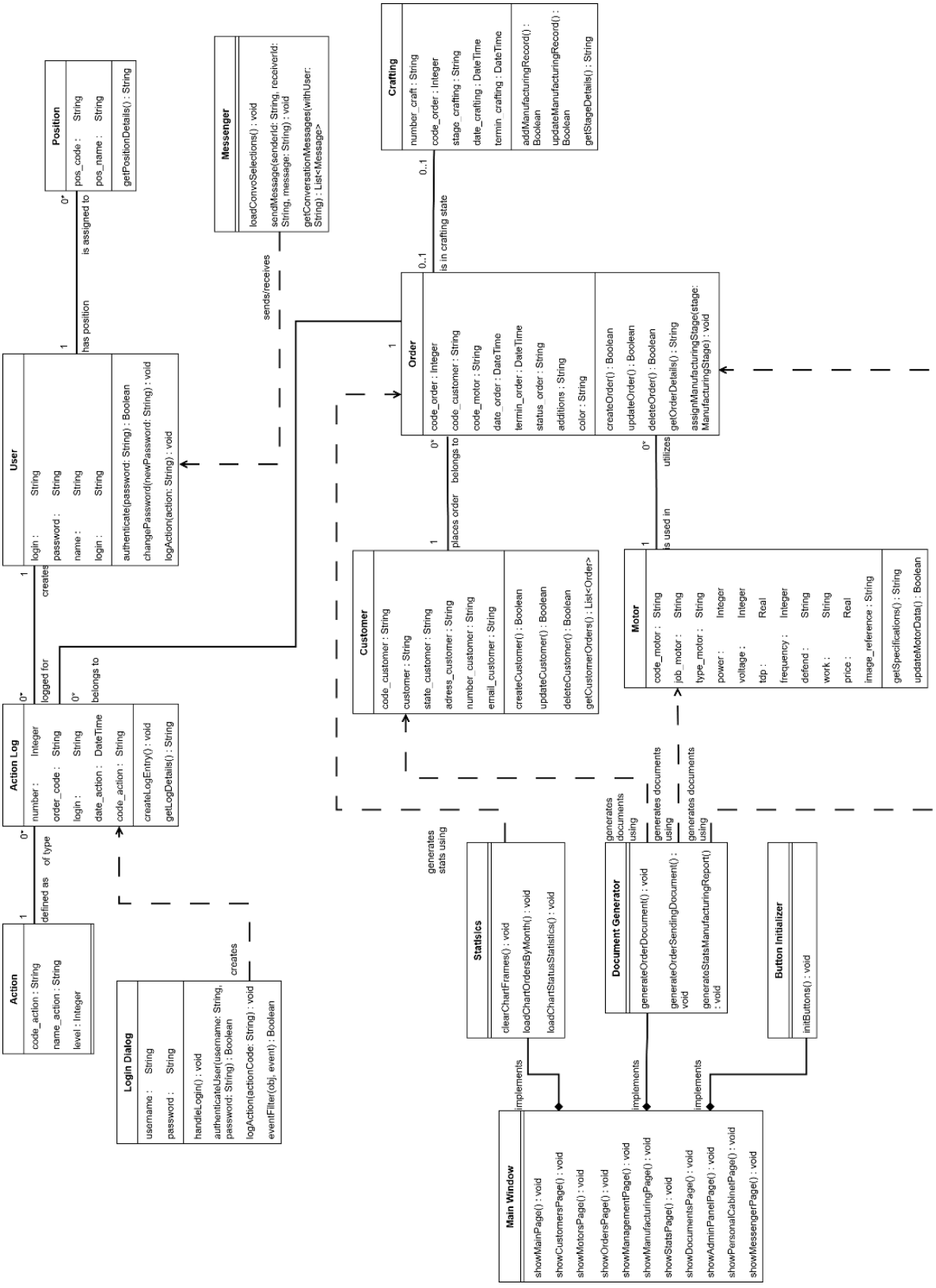

Рисунок 3.1 – Отримана діаграма класів розроблюваної системи

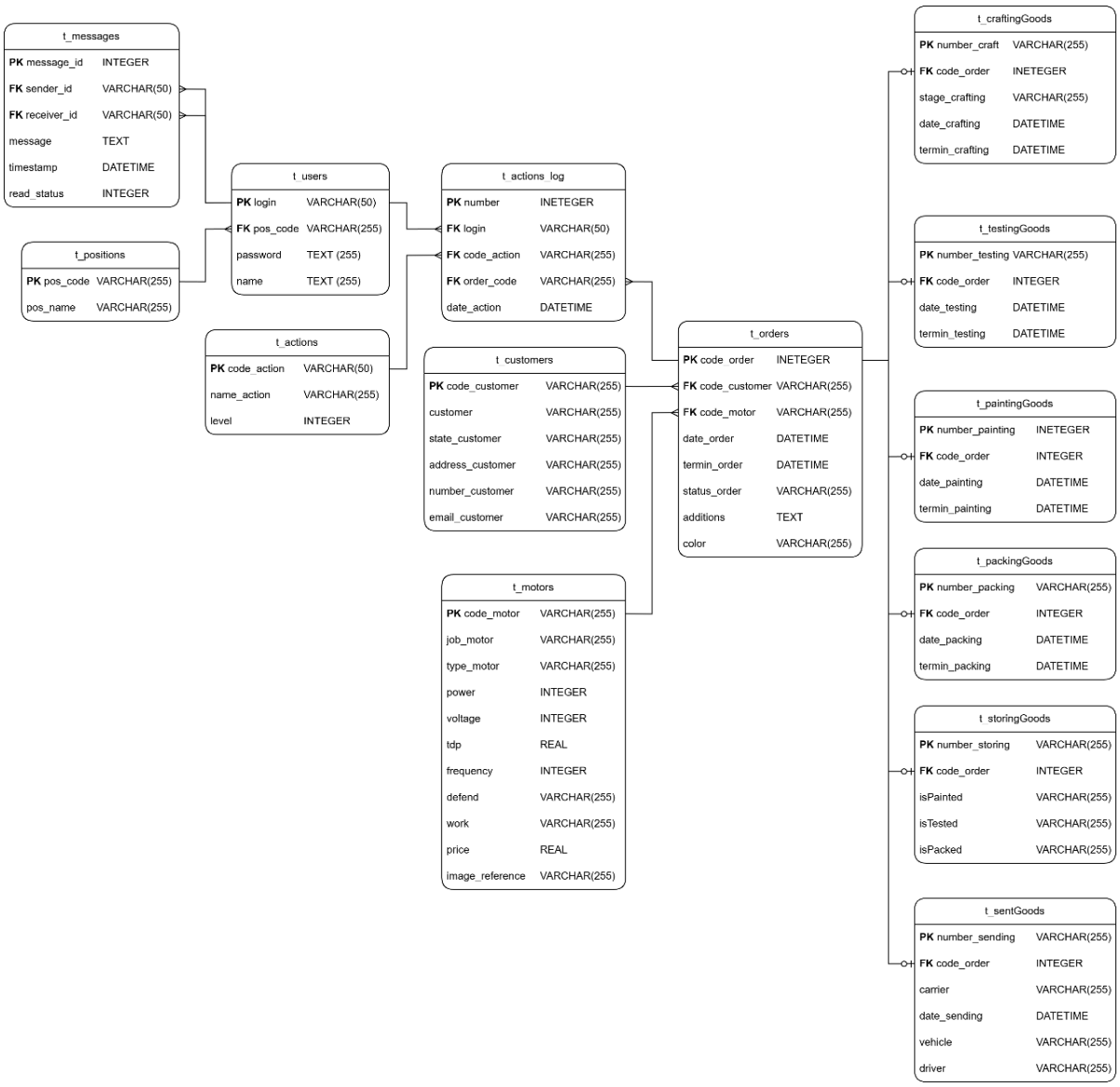

Рисунок 3.2 – Отримана ER-діаграма розроблюваної системи

Була забезпечена логічна цілісність даних за допомогою ER-діаграми, що полегшує управління базою даних і підтримку інформаційної системи. У поєднанні з UML-діаграмою класів ER-діаграма уточнює бізнес-логіку, визначає ролі об’єктів і методи, сприяючи створенню гнучкої, розширюваної архітектури з можливістю подальшої інтеграції додаткових модулів.

Таким чином, об’єктна модель виступає ключовою стадією в розробці системи, забезпечуючи глибоке розуміння причинно-наслідкових зв’язків, властивостей об’єктів та їх взаємодії у межах визначених цілями дослідження.

3.2 Розробка архітектури

У даному підрозділі детально розглянуто архітектурні рішення, прийняті під час створення інформаційної системи: від вибору загальної моделі до розподілу функцій між модулями та організації взаємодії між ними.

3.2.1 Вибір архітектурного підходу

Для застосунку обрано багатошарову компонентну архітектуру Model–View–Controller, що розділяє код на три логічні рівні:

– модель: SQLite-база даних із таблицями для користувачів, клієнтів, замовлень, виробничих етапів, логів і повідомлень;

– представлення: файли інтерфейсу користувача, згенеровані Qt Designer;

– контролери: методи програмного засобу, що реагують на сигнали інтерфейсу й виконують бізнес-логіку.

Таке розділення спрощує тестування, дозволяє незалежно розвивати інтерфейс і логіку та усуває жорсткі зв’язки між шарами.

3.2.2 Компонентна модель системи

Код програми згруповано у файли-модулі, які формують компоненти системи. Для візуалізації архітектури системи була створена діаграма компонентів (рис 3.2).

Верхній блок «Програмний засіб» узагальнює виконуваний пакет. Суцільні стрілки створення екземплярів: під час запуску формується LoginDialog, після аутентифікації – MainWindow.

Штриховані стрілки логічного доступу до БД показують, що всі тематичні плагіни мають окремі запити, але використовують спільну схему даних.

Штрих-пунктирні стрілки від MainWindow до міксинів означають динамічне підмішування – в UML це трактують як відношення «use / include».

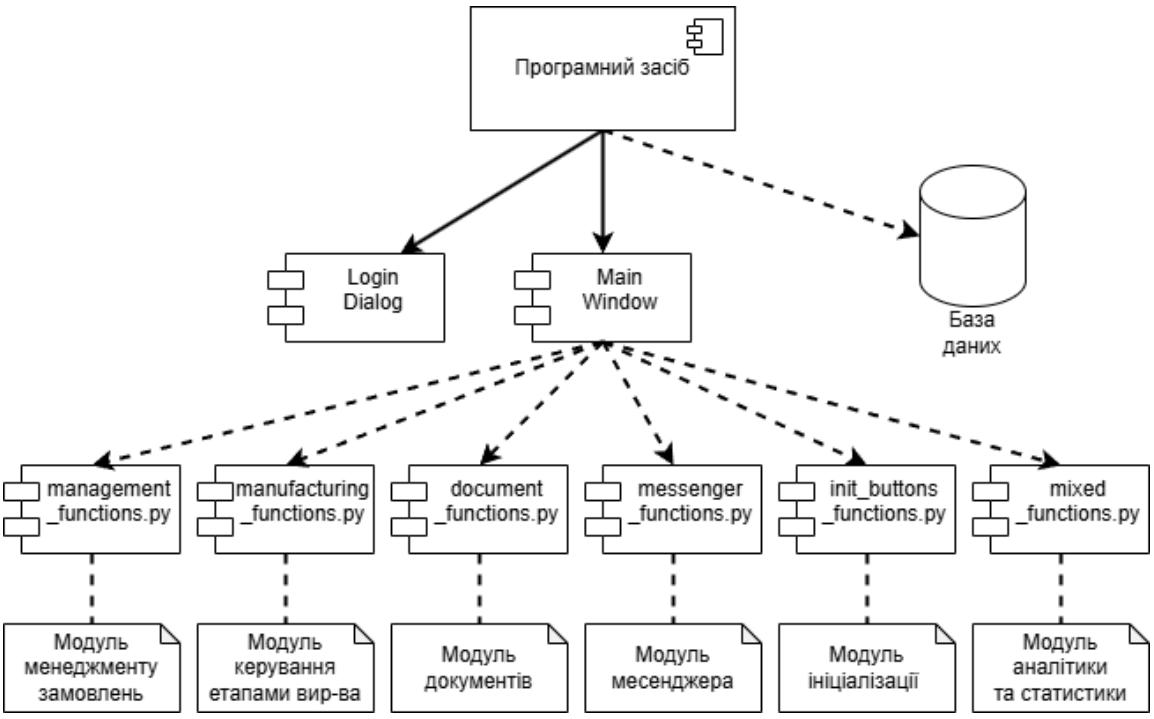

Рисунок 3.3 – Діаграма компонентів системи

3.2.3 Взаємодія компонентів

Кожний віджет генерує сигнал у момент взаємодії користувача, а клас-ініціалізатор кнопок ButtonsInitializer реєструє відповідні слоти, які викликають необхідні методи обробки. Такий підхід гарантує, що логіка реагування на події лишається централізованою та зрозумілою, а додавання нових елементів інтерфейсу не потребує змін у вже реалізованому зв’язуванні Signal-slot.

Головний клас MainWindow наслідує одразу п’ять міксинів, кожен із чіткою відповідальністю (Single Responsibility Principle). Це дає змогу підключати або відключати функціональність не порушуючи цілісність коду

Таким чином, обрана багатошарова компонентна архітектура забезпечує гнучкість, простоту розширення та відповідність нефункціональним вимогам. Міксиновий підхід у поєднанні з Qt-Signal/Slot-шиною та безпечною роботою з БД дозволив реалізувати легко підтримуваний і масштабований програмний продукт.

3.3 Проєктування інтерфейсу програмної системи

У цьому підрозділі розглянуто підходи до проєктування користувацького інтерфейсу розроблюваної системи: визначено основні принципи та вимоги до UI-дизайну, описано процес прототипування й побудову навігаційної структури, а також обґрунтовано вибір компонентів і стилістичних рішень.

Графічний інтерфейс призначений забезпечити оперативний доступ до ключових бізнес-функцій для користувачів із різними ролями. Під час проєктування були закладені принципи:

– єдність стилю: усі вікна використовують однакову палітру, шрифти сімейства Nunito та радіус заокруглення елементів 10 px;

– мінімізація шляху до дії: не більше трьох кліків до будь-якої операції;

– візуальна ієрархія: важлива аналітика виділяється у вигляді діаграм, другорядні дані – у вигляді списків або таблиць;

– доступність: контраст тексту ≥ 4.5:1, клавіатурна навігація через Tab-order.

Для пришвидшення проектування інтерфейсу програмного засобу було вирішено скористуватися вільним дизайном Figma Community (рис. 3.4), який було адаптовано під корпоративну айдентику підприємства та функціональні потреби системи (рис. 3.5).

Використання готового шаблону обґрунтовано дослідженнями в галузі UX: за даними Neilsen Norman Group [15], застосування перевірених UI-компонентів і патернів дозволяє скоротити час прототипування та знизити кількість помилок у реалізації інтерфейсу, при цьому підвищуючи його інтуїтивність.

У результаті було розроблено компонентний, ергономічний та масштабований інтерфейс, що забезпечує швидкий доступ до основних бізнес-процесів і може безболісно розширюватися разом із функціональними потребами підприємства.

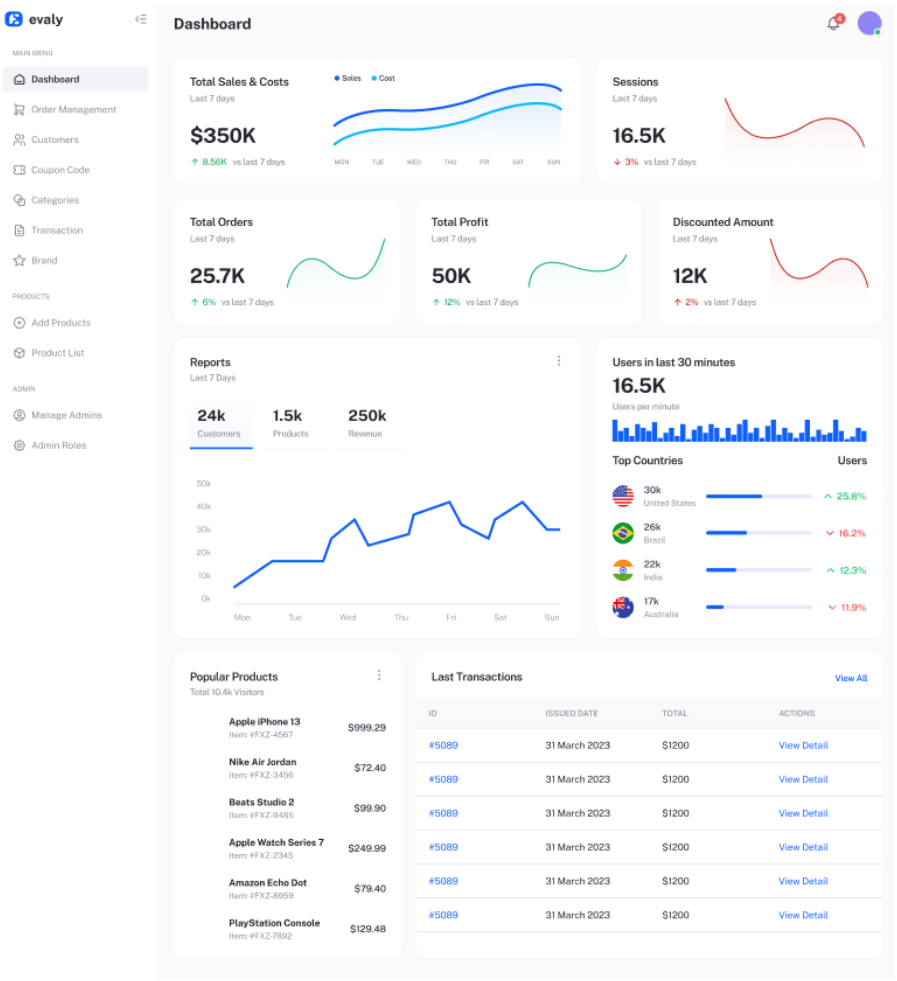

Рисунок 3.4 – Evaly e-commerce dashboard

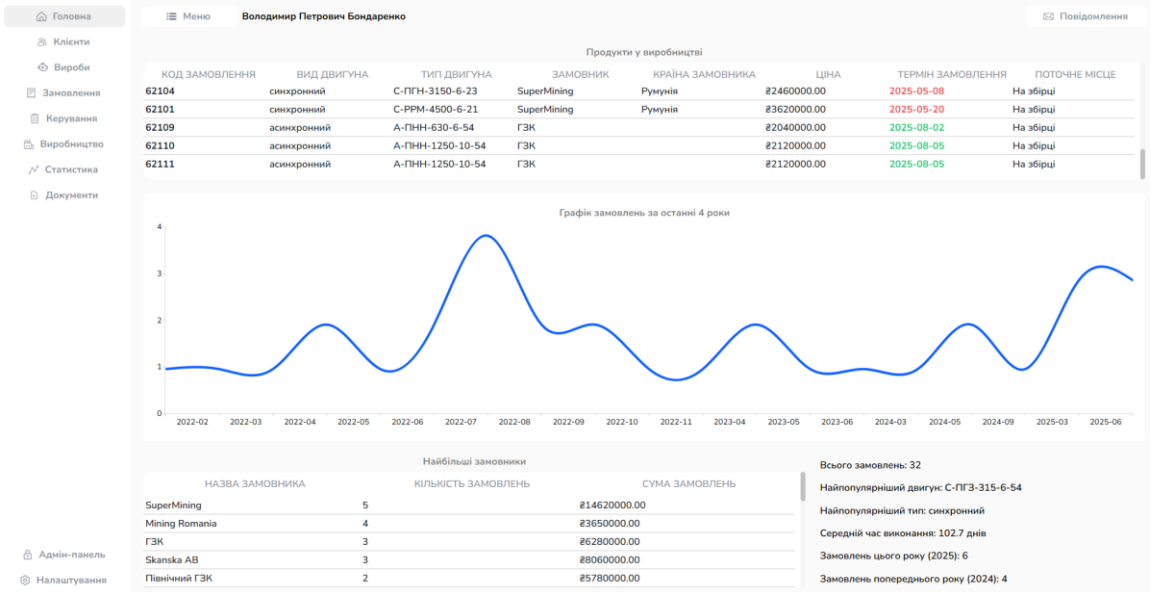

Рисунок 3.5 – Отриманий інтерфейс програми

3.4 Обґрунтування вибору мови програмування

Проєктування системи для управління виробництвом важких електромашин передбачає використання рішень, що гарантують не лише достовірність даних, а й швидку й безперебійну взаємодію між усіма модулями платформи.

Головними завданнями, виділеними для технологічного стеку стали:

– забезпечення інтегрованої взаємодії між підрозділами підприємства;

– автоматизація обробки замовлень та контролю якості на кожному етапі виробництва;

– гнучкість у налаштуванні інтерфейсу користувача під час розробки для поліпшення взаємодії всередині системи;

– надійність зберігання та обробки даних, що охоплюють як аналітичні, так і операційні аспекти діяльності підприємства.

Враховуючи ці вимоги, при виборі технологічного стеку для розробки ІС слід орієнтуватися на інструментальні засоби, що дозволяють швидко впровадити необхідний функціонал, забезпечити масштабованість, гнучкість та простоту підтримки програмного продукту.

3.4.1 Обґрунтування вибору мови розробки програмного забезпечення

Python – високорівнева мова з чистим та інтуїтивним синтаксисом, поєднуючи простоту з потужним набором інструментів, що підтримує швидку розробку та легке обслуговування коду. Це особливо важливо у проєктах з частими змінами бізнес-логіки та командною розробкою, де чітка структура модулів полегшує внесення змін і масштабування функціоналу.

Екосистема Python пропонує бібліотеки для роботи з базами даних (sqlite3), створення графічних інтерфейсів (PySide6) та обробки документів (python-docx).

Використання перевірених рішень скорочує час розробки та тестування, дозволяючи зосередитися на реалізації бізнес-вимог.

Для створення прототипу ERP-системи важлива можливість оперативно формувати робочі моделі, аналізувати їх та вносити корективи за результатами тестування й зворотного зв'язку. Динамічна природа Python дозволяє робити зміни без довготривалої компіляції та складних налаштувань середовища.

Кросплатформність обраної мови гарантує запуск розробленого продукту як на Windows, так і на Unix-подібних ОС, зменшуючи залежність від апаратної частини підприємства.

3.4.2 Аналіз переваг і недоліків Python у контексті проекту

При виборі Python для розробки важливо враховувати не тільки його численні переваги, а й потенційні обмеження. До переваг мови належать:

– швидке освоєння та підтримка: прозорий синтаксис скорочує час введення в проєкт нових учасників та полегшує супровід;

– багата екосистема: великий набір бібліотек і фреймворків (для аналізу даних, роботи з файлами, GUI тощо) дозволяє інтегрувати готові рішення замість власної розробки;

– активна спільнота: швидкий доступ до документації, прикладів і рішень із реальних проєктів підвищує надійність розробки.

Водночас була врахована низка обмежень мови, серед яких:

– продуктивність: інтерпретація коду може бути повільнішою за компільовані мови (C++, Java) [7], але для операцій із базами даних, обробки документів та GUI-логіки ця втрата зазвичай не критична;

– динамічна типізація: хоча спрощує прототипування, підвищує ризик runtime-помилок, тому необхідні автоматизовані тести і статичний аналіз коду;

– ресурсні обмеження при масштабуванні: у великих ERP-системах із значним обсягом обробки даних може виникнути потреба в оптимізації критичних ділянок коду або використанні розширень на C. Водночас для систем середнього масштабу Python забезпечує належний рівень продуктивності.

3.4.3 Використання суміжних технологій: PySide6 та SQLite

Окрім самої мови Python, у проєкті використовуються дві ключові технології: фреймворк PySide6 і СКБД SQLite.

PySide6 дозволяє створювати крос-платформні графічні інтерфейси з готовими віджетами. Надає широкий набір елементів для побудови функціонального інтерфейсу, забезпечує просту інтеграцію з Python-кодом, має відкритий вихідний код, що дозволяє адаптувати та розширювати бібліотеку.

SQLite – легка вбудована реляційна СКБД, що не потребує окремого сервера й ідеально підходить для середніх обсягів даних, що забезпечить стабільну продуктивність для ERP-системи обраного розміру. Вбудована підтримка в Python реалізована через стандартний модуль sqlite3.

3.4.4 Порівняльний аналіз альтернативних підходів

Незважаючи на значні переваги обраної мови програмування, варто розглянути альтернативні технології, які могли б бути застосовані для розробки ERP-систем.

Java та C# пропонують високу продуктивність, статичну типізацію та багату екосистему, але вимагають більше часу на налаштування та розробку GUI, що ускладнює швидке створення прототипів.

JavaScript та TypeScript ідеальні для динамічних веб-інтерфейсів, але в настільних ERP-застосунках інтеграція з локальними базами даних і реалізація складної бізнес-логіки виявляються трудомісткими.

C++ забезпечує максимальну швидкодію, проте потребує високої кваліфікації розробників, тривалішого циклу розробки та значних зусиль для побудови UI й роботи з БД.

3.4.5 Практичні кейси використання та досвід інтеграції

Python довів свою ефективність у проєктах різного масштабу – від відкритих ERP-систем до високонавантажених веб-сервісів.

Odoo – модульна ERP-платформа на Python, що охоплює бухгалтерію, CRM, логістику та виробництво. Завдяки гнучкій архітектурі Python реалізує високу модульність: нові функції додаються як окремі пакети без зміни ядра.

ERPNext – сучасна ERP-система, у якій Python забезпечує швидкі ітерації бізнес-логіки та ефективну обробку даних, що допомагає компаніям адаптувати процеси в режимі реального часу.

У великих сервісах, як Dropbox, YouTube та Instagram Python відповідає за серверну логіку, де потрібно обробляти мільйони запитів щодня. Мова забезпечує високу продуктивність навіть під великим навантаженням.

У розробці ERP для виробничого підприємства ключові кейси інтеграції:

– автоматизація документообігу: використання бібліотеки python-docx дозволяє автоматизовано формувати договори, накладні та інші документи;

– інтерфейс користувача та інтеграція модулів: PySide6 дозволяє створювати зручні крос-платформні користувацькі інтерфейси;

– обробка даних: використання SQLite у проекті дозволяє організувати зберігання даних без необхідності налаштування додаткового серверного ПЗ.

Таким чином, вибір мови програмування Python разом із с технологіями PySide6 та SQLite є обґрунтованим для розробки локальної ERP-системи. Отриманий результат підтверджується як теоретичним аналізом, так і практичними прикладами, реалізованими у відповідних модулях проекту.

3.5 Опис програмної реалізації

Інформаційна система розроблена із застосуванням мови Python та фреймворку Qt (PySide 6) із використанням патерна MVC та міксин-архітектури. Кожен міксин реалізує окремі модулі, а головний клас MainWindow успадковує їх, формуючи цілісну ІС.

3.5.1 Технологічний стек

Проєкт розгортається у середовищі Python 3.10 із використанням бібліотек:

– PySide6 для побудови UI й реактивного зв’язку через сигнали/слоти;

– sqlite3, як легка вбудована СУБД;

– python-docx для створення Word-документів;

– pandas для обробки статистичних даних;

– bcrypt для хешування та перевірки паролів.

3.5.2 Автентифікація та логування

Вікно входу реалізовано у класі LoginDialog. Паролі зберігаються у базі у вигляді хешу bcrypt. Метод автентифікації виконує запит до таблиці користувачів, порівнює введений пароль із хешем, а подію входу фіксує в t_actions_log. Фрагмент лістингу коду файлу «main.py», який виконує автентифікацію:

```
def authenticate_user(u, p):
    with sqlite3.connect('DBAdm.db') as c:
        h = c.execute(
            'SELECT pass_hash FROM t_users WHERE login=?', (u,)
        ).fetchone()
    return h and bcrypt.checkpw(p.encode(), h[0].encode())
```

Параметризований запит гарантує захист від SQL-ін’єкцій. У разі помилки генерується попередження типу QMessageBox.warning.

3.5.3 Завантаження кнопок

Для зменшення комплексності коду програми, було створено окремі класи.

Клас-ініціалізатор ButtonsIntializer містить метод init_buttons(), де відбувається підключення всіх сигналів кнопок до методів-обробників. Фрагмент лістингу коду файлу «init_buttons_functions.py», який присвоює дії кнопкам:

```
def init_buttons(self):
 self.ui.mainButton.clicked.connect(self.show_main_page)
 self.ui.ordersButton.clicked.connect(self.show_orders_page)
 self.ui.manageFind.clicked.connect(self.on_manage_find)
 self.ui.docGenerate.clicked.connect(self.generate_order_doc)
```

3.5.4 Керування замовленнями (CRUD)

Модуль ManagementMixin інкапсулює створення, пошук, редагування та видалення записів у таблиці замовлень:

– пошук: SELECT … WHERE code_order=?;

– створення: INSERT, попередня перевірка на наявність клієнта та мотора;

– оновлення: UPDATE, валідація формату дат;

– видалення: DELETE з діалогом підтвердження.

Усі транзакції виконуються в with sqlite3.connect(); помилки перехоплюються та відображаються кінцевому користувачу.

Модуль ManufacturingMixin працює з шістьома таблицями виробничого процесу (збірка, тестування, фарбування, пакування, зберігання, відправлення). Для кожної з них використано спільний шаблон load_data. Фрагмент коду файлу «manufacturing_fucntions.py», який завантажує дані в таблиці етапів:

```
def load_data(self, table, widget):
 with sqlite3.connect('DBAdm.db') as c:
  rows = c.execute(f'SELECT * FROM {table}').fetchall()
 self.fill_table(widget, rows)
```

Операції locate / edit / complete виконують оновлення й моментально відображають зміни у QTableWidget.

3.5.5 Генерація документів та візуальна аналітика

Модуль DocumentsMixin автоматично створює договори купівлі-продажу, накладні, звіти за шаблонами python-docx.

Фрагмент коду файлу «document_functions.py», який формує договір купівлі-продажу за шаблоном:

```
doc = Document()
doc.add_heading('Договір купівлі-продажу', 0)
doc.add_paragraph(f'Замовлення № {order}')
doc.save(Path.home() / 'Documents' / f'Contract_{order}.docx')
```

Міксин клас MixedMixin реалізує аналітичні діаграми та графіки. Для візуалізації використовуються графіки QtCharts. Дані збираються через pandas.read_sql_query, обробляються й передаються в серії QLineSeries, QPieSeries і QbarSeries (рис. 3.6).

Рисунок 3.6 – Графік помісячної зміни кількості замовлень

3.5.6 Внутрішній чат

Підсистема обміну повідомленнями реалізована в класі MessengerMixin і включає три основні механізми: формування списку співрозмовників, завантаження історії листування та обмін повідомленнями.

Система забезпечує безпечну аутентифікацію, повний життєвий цикл замовлень, контроль виробничих етапів, автоматичні документи, внутрішню комунікацію та оперативну аналітику.

Модульний підхід на основі mixin-класів дозволяє масштабувати систему: додавати нові звіти, інтегрувати зовнішні сервіси чи розширювати UI без порушення існуючої логіки. Така архітектура сприяє швидкій підтримці та простоті розвитку продукту.

РОЗДІЛ 4. ДОСЛІДНА ЕКСПЛУАТАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

4.1 Інструкція користувача ІС

Розроблений програмний продукт забезпечує повний цикл супроводу виробничого замовлення: від первинної реєстрації клієнта до відвантаження готового виробу. Робота користувача відбувається через головне вікно з панеллю навігації, тоді як доступність окремих функцій регулюється роллю інженера або адміністратора.

4.1.1 Встановлення та запуск

Розпакуйте дистрибутив у робочу теку (рекомендується C:\Program Files\Manufacturer). Переконайтеся, що у системі був встановлений Python 3.10 й успішно інстальовані модулі PySide6, pandas, python-docx. Переконайтеся, що файл бази даних розташовано поряд із виконуваним файлом.

Запустіть програму подвійним клацанням по ярлику Manufacturer.exe Під час старту ініціалізуються елементи навігації та відбувається перевірка доступу до бази даних.

4.1.2 Авторизація та робота з обліковими даними

У стартовому вікні введіть логін і пароль, надані адміністратором бази. Натисніть «Увійти» або Enter. Система звіряє введені дані з таблицею користувачів та фіксує подію входу.

Для зміни пароля оберіть у бічному меню пункт «Кабінет», введіть свій логін у перше поле, старий пароль у друге поле, у третє – новий, натисніть «Змінити пароль». Після успішного збереження система повідомить про зміни.

4.1.3 Навігація головним вікном

Для згортання та розгортання бічної панелі праворуч робочої зони скористайтеся кнопкою «Меню». Основні сторінки бічної панелі представлені у таблиці 4.1.

Таблиця 4.1 – Сторінки головного вікна програми

Сторінка	Призначення
Головна	Зведені показники та швидка статистика
Клієнти	Довідник клієнтів
Вироби	Каталог двигунів
Замовлення	Перегляд і створення замовлень
Керування	Редагування замовлень
Виробництво	Відстеження етапів виробництва
Статистика	Візуальна статистика
Документи	Автоматичне формування договорів і накладних
Адмін-панель	Робота з логом дій та редагування користувачів
Налаштування	Змінення пароля для користувачів
Повідомлення	Внутрішні повідомлення

Перемикайтеся між розділами лівою кнопкою миші. Активний пункт підсвічується, вміст центральної області змінюється динамічно.

Сторінки «Клієнти» та «Вироби» надають можливості інтерактивної взаємодії із клієнтами та моторами. Натиснення на кнопку «Деталі» дозволяє редагувати інформацію про клієнта або виріб у окремій формі.

Натисніть на віджет клієнта або мотора лівою кнопкою миші для розширення списку замовлень, в яких фігурує обраний запис. Подвійний клік на обране замовлення спрямує користувача на сторінку «Замовлення» для деталізації обраного замовлення.

4.1.4 Операції над замовленнями

Для спрощеної навігації виробничим процесом, швидкого відкриття, редагування, або видалення замовлень відкрийте сторінку «Замовлення».

Для створення замовлень натисніть «Відкрити замовлення», новий код з’явиться у картці замовлення, інші поля підготуються для заповнення. У поле «Замовник» введіть назву або код замовника, якщо його не існує – програма надає можливість його створити. Редагуйте поля мотора, дати та терміну замовлення, кольору виробу та додаткових побажань. Після, натисніть «Відкрити замовлення» ще раз.

Для редагування інформації про замовлення введіть код потрібного замовлення у поле для пошуку, натисніть «Пошук», або використовуйте перемикання замовленнями для навігації. Редагуйте доступні поля, натисніть «Змінити» для підтвердження операції. Для видалення замовлень оберіть потрібне та натисніть «Видалити» для підтвердження операції.

Для знаходження замовлення на виробництві оберіть потрібне та натисніть «Знайти на виробництві». Програма перенесе вигляд користувача на поточний етап виробництва для обраного виробу.

Безпосередню роботу з журналом замовлень забезпечує сторінка «Керування». Використовуйте кнопку «Знайти» для того, щоб заповнити поля сторінки або використайте подвійний клік по обраному замовленню. Натисніть «Змінити» для редагування інформації, «Відкрити» для відкриття, «Видалити» для видалення замовлення.

4.1.5 Контроль виробництва

Для контролю виробничих етапів оберіть сторінку «Виробництво». Перемикайтеся шістьома етапами за допомогою верхніх кнопок «Збірка», «Тестування», «Фарбування», «Пакування», «Зберігання», «Відправлення».

Щоб сфокусуватися на виробі на етапі, введіть код замовлення у полі та натисніть кнопку «Знайти», або подвійний клік по обраному замовленню.

За потреби зміни інформації про виріб на етапі редагуйте окремі поля та натисніть «Змінити» для збереження змін. Після завершення робіт на поточному етапі натисніть «Push»: виріб переміститься до наступної таблиці, а відповідний статус автоматично оновиться у картці замовлення.

4.1.6 Генерація документів та аналітики

Для автоматизованої генерації документів та звітів оберіть сторінку «Документи». Центральна зона містить таблицю із поточними замовленнями для спрощення знаходження потрібного замовлення. Натисніть подвійним кліком на потрібне замовлення щоб сфокусуватися на ньому у полі для кодів замовлень.

Оберіть кнопку «Договір» або «Накладна» для генерації договору купівлі-продажу для виробу або для сформування накладної для замовлення. Натисніть «Виробництво» або кнопки окремих етапів для отримання звітів з виробництва. Документи будуть сформовані із відповідною назвою у теці «Документи».

Спрощену генерацію договору та накладної також представлено на сторінці «Замовлення». Натисніть відповідні кнопки для обраного виробу для формування документів.

Для перегляду графіків виробництва оберіть сторінку «Статистика». На робочій зоні буде представлена візуальна статистика виробництва.

4.1.7 Внутрішній месенджер

Для того, щоб скористуватися вбудованим месенджером оберіть «Месенджер» вгорі екрану. Ліворуч відображається перелік співробітників; праворуч – обраний діалог. Щоб перемикатися адресатами натисніть на їх картки ліворуч. Введіть текст у нижнє поле та натисніть Enter. Повідомлення буде відправленим і стане доступним адресату

4.1.8 Адміністрування та вихід з програми

У модулі «Адмін-панель» адміністратори можуть додавати нових користувачів у зоні створення користувача у полях «Логін» та «Пароль». Після натиснення «Створити» створиться новий користувач із зашифрованим паролем. Центральна область містить перегляд дій користувачів. Для експорту журналу натисніть «Експортувати лог»: програма створить журнал у форматі PDF у визначену папку.

Для коректного виходу натисніть закрийте головне вікно. Програма фіксує подію виходу та завершує підключення до бази даних. Додаткове збереження не потребується, оскільки всі зміни збережуються одразу після виконання операцій.

Інтерфейс автоматично приховує елементи, недоступні вашій ролі. Якщо певний функціонал відсутній або недоступний, зверніться до адміністратора для перегляду прав доступу.

4.2 Тестування програмного забезпечення

Тестування програмного забезпечення – заключний етап життєвого циклу розробки, що підтверджує відповідність готового продукту технічному завданню й очікуванням замовника. Метою випробувань є виявлення дефектів на рівні функціональності, інтерфейсу та цілісності даних, а також перевірка коректності взаємодії окремих модулів системи.

4.2.1 Мета й задачі тестування

Були визначені основні завдання підрозділу тестування:

– перевірка функціональної повноти: чи реалізовані всі, описані в ТЗ, можливості (авторизація, керування замовленнями, виробничий цикл, генерація документів тощо);

– перевірка стійкості до помилок вводу: адекватність повідомлень, відсутність неконтрольованих виключень;

– перевірка узгодженості даних: чи зберігається лиш одна «джерельна» копія інформації між модулями та базою SQLite, чи коректно оновлюються пов’язані таблиці;

– перевірка прав доступу: відповідність доступності елементів ролям;

– перевірка нефункціональних вимог: швидкість відгуку користувацького інтерфейсу, час формування документів, витоки пам’яті.

4.2.2 Організація випробувань

Після визначення завдань для тестування були проведені випробування програмного виробу. Поступово виконувалися сценарії, описані в таблиці 4.2. Після виконання кожного тесту складалися короткі звіти з описом фактичного результату. У разі відхилення фактичного результату від очікуваного негайно реєструвалася помилка для подальшого аналізу та виправлення.

Таблиця 4.2 – Тестові ситуації

№	Очікуваний результат	Статус
1	Авторизація з коректними обліковими даними.	Авторизація у головне вікно, у журналі подій з’явився запис входу.
2	Авторизація з хибним паролем.	Помилка «Некоректний пароль», сторінка входу залишається.
3	Додавання нового замовлення.	Запит з’явився у таблиці замовлень та на збірці, статус «Виробляється».
4	Додавання замовлення із кодом, який уже існує.	Програма підготовлює поля із автоматично згенерованим кодом.

Продовження таблиці 4.2

5	Спроба Push без вибраного рядка.	Попередження «Виберіть запис», зміни не внесені.
6	Видалення замовлення без відповідних прав.	Попередження «Недостатньо прав», зміни не внесені.
7	Введення даних у неправильному форматі.	Попередження «Неправильний формат поля», зміни не внесені.
8	Генерація договору купівлі-продажу за неіснуючим замовленням.	Попередження «Введіть правильний код замовлення», файл не створено.
9	Переміщення до замовлення з таблиць із сторінки «Клієнти».	Перехід на сторінку «Замовлення» до обраного коду замовлення.
10	Спроба SQL-ін’єкції у полі коду замовлення.	Повідомлення «Запис не знайдено», таблиця лишається неушкодженою.
11	Спроба відкрити Admin Panel користувачем-інженером.	Кнопка прихована, доступ заборонено.
12	Видалення відсутнього замовлення.	Попередження «Запис не знайдено»
13	Генерація номера на етапі при переході етапами.	Код автоматично згенеровано за заданим форматом.

4.2.4 Аналіз результатів

Результати проведених випробувань підтвердили, що:

– функціональні вимоги виконано повністю: усі ключові сценарії (авторизація, CRUD-операції над замовленнями, передача етапами виробництва, генерація документів, обмін повідомленнями) відпрацьовують коректно;

– стійкість до помилок вводу забезпечено: діалогові вікна попереджають про порожні чи некоректні поля, перехоплюються винятки sqlite3.Error, запобігаючи аварійному завершенню;

– консистентність БД не порушується: транзакції INSERT/UPDATE/DELETE супроводжуються commit/rollback; перехресні оновлення між таблицею замовлень і виробничими таблицями виконуються через відокремлені методи;

– розмежування доступу працює: адміністративні функції недоступні інженеру, невідповідні кнопки приховані при ініціалізації.

Проведене тестування програмного засобу довело працездатність і відповідність інформаційної системи вимогам технічного завдання. Зафіксовані зауваження мають косметичний характер і можуть бути усунені в ході регламентного супроводу. Програмний продукт вважається підготованим до використання в корпоративному середовищі.

4.3 Техніко-економічні показники ІС

Для оцінки ефективності створення та впровадження інформаційної системи необхідно проаналізувати її техніко-економічні показники. До них належать витрати на оплату праці, амортизаційні відрахування, витрати на матеріали, електроенергію та інші супутні витрати.

4.3.1 Витрати на оплату праці

До даних витрат належать витрати на оплату праці розробників з нарахуваннями, амортизаційні відрахування, витрати на матеріали, електроенергію та інші супутні витрати.

Витрати на оплату праці з нарахуваннями складаються з витрат на основну заробітну плату, додаткову заробітну плату та нарахувань згідно з чинного законодавства (22%).

Витрати на основну заробітну плату визначаються виходячи з місячного посадового окладу або вартості 1 години роботи розробника та витрат часу, понесених на розробку даного продукту.

Розрахунок основної оплати праці здійснюється за формулою:

$$ЗПосн = \frac{ЗПміс}{ФРЧ} \cdot Тр,$$

де ЗПосн – основна заробітна плата , грн;

ЗПміс – місячний посадовий оклад розробника, на підприємстві становить 14800 грн/міс;

ФРЧ – місячний фонд робочого часу, становить 176 (22 дні по 8 год.) год./міс;

Тр – трудомісткість (затрати часу) виготовлення програмного продукту, год.

Трудомісткість (затрати часу) розробника на створення продукту визначається виходячи з кількості витрачених на розробку робочих днів та тривалості робочого дня:

$$ФРЧ = Крдм \cdot Трз,$$

де Крд – кількість робочих днів на місяць, прийнято 20 дн.;

Трз – тривалість робочого дня (зміни), год.

Отже,

$$Тр = 20 \cdot 8 = 160год;$$

Таким чином:

$$ЗПосн = \frac{14800}{176} \cdot 160 = 13455 \text{ грн. .}$$

У випадку, якщо при розробці продукту задіяні декілька працівників, розмір основної заробітної плати розраховується за кожним.

Додаткова заробітна плата передбачає різноманітні види доплат за рівень кваліфікації, стаж роботи тощо. Розмір нарахувань здійснюється відповідно до норм прийнятих на підприємстві замовника. Відсоток нарахувань додаткової заробітної плати коливається в межах 10-25%. На даному підприємстві додаткова оплата праці становить 18,5%.

Сума додаткової заробітної плати розраховується від розміру основної:

$$ЗПдод = ЗПосн \cdot \frac{\%дод}{100} ,$$

де %дод – відсоток нарахувань додаткової заробітної плати, %;

$$ЗПдод = 13455 \cdot \frac{18,5}{100} = 2490 \text{ грн. .}$$

Нарахування на заробітну плату здійснюють від суми основної та додаткової зарплати. Розмір нарахувань становить 22%:

$$Нзп = (ЗПосн + ЗПдод) \cdot \frac{\%нар}{100} ,$$

де %нар – відсоток нарахувань заробітної плати, %.

Отже,

$$Нзп = (13455 + 2490) \cdot \frac{22}{100} = 3507,9 \text{ грн. .}$$

Розрахунок загальної суми витрат на оплату праці з нарахуваннями розробника наведений у вигляді таблиці 4.3.

Таблиця 4.3 – Розрахунок витрат на оплату праці з нарахуваннями розробників програмного продукту

Найменування посади	Основна заробітна плата, грн.	Додаткова заробітна плата, год.	Розмір нарахувань на оплату праці, грн.	Всього витрат на оплату праці з нарахуваннями, грн.
Інженер-програміст	13455	2490	3507,9	19452,9

4.3.2 Амортизаційні відрахування

Розрахунок суми амортизаційних відрахувань обладнання, яке використовується при розробці продукту здійснюється найбільш поширеним прямолінійним методом. При створенні продукту використовувався ноутбук та принтер. Відповідно до податкового кодексу України електронно-обчислювальні машини (ПК) відносять до 4 групи, з терміном корисного використання 2 роки. Відповідно річна норма амортизації становить 50%.

Балансова вартість ПК (або ноутбуку) на момент придбання (розрахунку) становить 17900 грн, принтера - початкова вартість 9800 грн, приймається річнп норму амортизації в розмірі 20% (термін використання 5 років).

Розрахунок річної суми амортизаційних відрахувань:

$$\text{Аріч} = \text{ББобл} \cdot \frac{\%\text{аморт}}{100},$$

де БВобл. - ціна придбання, або балансова вартість комп’ютера на момент придбання, грн.;

%аморт – річний відсоток нарахувань додаткової заробітної плати, %.

$$\text{Аріч. комп.} = 17900 \cdot \frac{50}{100} = 8950\text{грн.} \ ,$$

$$\text{Аріч. принт.} = 9800 \cdot \frac{20}{100} = 1960\text{грн.} \ .$$

Суму амортизаційних відрахувань, що увійде до собівартості програмного продукту розраховується із врахуванням часу використання обладнання при розробці продукту:

$$\text{А} = \text{Аріч} \cdot \frac{\text{Твик(днів)}}{365},$$

де Твик – термін використання персонального комп’ютера при розробці програмного продукту, міс. або дні.

Таким чином, загальна сума амортизаційних відрахувань, становитиме:

$$\text{А} = (8950 + 1960) \cdot \frac{20}{365} = 597{,}8 \text{ грн.} \ .$$

Витрати на матеріали визначаються відповідно до кількості витрачених матеріалів на розробку продукту та ціни на даний матеріал, на момент розрахунку. Витрати на матеріали, використані на розробку продукту наведені в таблиці 4.4.

Таблиця 4.4 – Витрати на матеріали, використані в розробці продукту

Найменування матеріалу	Ціна за одиницю матеріалу, грн.	Витрачено, од	Вартість витрачених матеріалів, грн.
Чорнила, банка	259	3	777
Папір, уп.	219	3	657
Разом витрат			1434

Витрати на електроенергію визначаються з потужності обладнання, що використовується при розробці продукту (П=0.51), коефіцієнту використання потужності (Квп=0,95), фактичного часу (трудомісткості) на розробку та ціни електроенергії, що склалась на момент розрахунку (Ц1кВт=4,32 грн.):

$$\text{Вел.} = \text{П} \cdot \text{Квп} \cdot \text{Тр} \cdot \text{Ц1кВт} = 0{,}51 \cdot 0{,}95 \cdot 4{,}32 \cdot 160 = 334{,}6 \text{ грн.}.$$

Розрахунок суми витрат на розробку відображений даними таблиці 4.5.

Таблиця 4.5 – Розрахунок витрат на створення продукту

Найменування витрат	Вартість, грн.
Витрати на оплату праці з нарахуваннями, грн.	19452,9
Амортизаційні відрахування, грн	597,8
Витрати на матеріали, грн	1434
Витрати на електроенергію, грн	334,6
Разом витрат	21819,3

4.3.3 Розрахунок собівартості одиниці копії програмного продукту

Сукупні витрати на створення та впровадження програмного продукту складаються з загальної суми витрат:

$$\text{Соб. прод} = \text{Вроз.} + \text{Вінш.}\ ,$$

де Врозр. – витрати на розробку програмного продукту, грн;

Вінш. – інші витрати, які не включені до витрат розробки, коливаються в межах 10-30% , прийняті на рівні 22,5%.

$$\text{Вінш.} = 21819{,}3 \cdot \frac{225}{1000} = 4909{,}3 \text{ грн.}\ .$$

Таким чином, витрати на створення продукту складатимуть:

$$\text{Вствор.} = 21819{,}3 + 4909{,}3 = 26728{,}6 \text{ грн.}\ .$$

Оскільки, програмний продукт приймається та впроваджується за завданням підприємства, розповсюдження програми не передбачається. Таким чином, собівартість одиниці продукту дорівнює витратам на його створення.

4.3.4 Розрахунок заощаджень від впровадження програмного продукту

Оскільки програма не продається стороннім споживачам, економічний ефект визначається через зниження витрат або підвищення продуктивності підприємства. Для цього розраховується очікувана економія від впровадження програми, рівень якої прийнятий в 40% від собівартості продукту на рік:

$$E_{ек} = \text{Вствор.} \cdot \text{Рек} = 26728{,}6 \cdot \frac{40}{100} = 10691{,}5 \text{ грн.}\ ,$$

де Рек – рівень економії від впровадження програми на рік, грн..

4.3.5 Термін окупності

Термін окупності характеризує період часу протягом якого окупляться витрати на розробку програмного продукту замовником.

Розрахунок періоду окупності здійснюється за формулою:

$$\text{Ток} = \frac{\text{Вствор.}}{\text{Е}_{\text{ек}}};$$

$$\text{Ток} = \frac{26728{,}6}{10691{,}5} = 2{,}49 \text{ роки.}$$

Таким чином, створення та подальший продаж розробленого програмного продукту є достатньо ефективним для підприємства. Витрати на створення продукту становлять 26,7 тис. грн. Із визначеним рівнем економії від провадження програми у 40% економічний ефект склав 10,7 тис. грн. на рік, а період окупності при даних показниках – 2,5 роки. Слід зауважити, що показники ефективності розраховувались з урахуванням мінімальної очікуваної економії витрат підприємства. При цьому реальний економічний ефект від впровадження програми може бути вищим, що підтверджує доцільність та ефективність її використання у виробничому процесі.

4.4 Техніка безпеки

Коротка характеристика приміщення і виконуваних робіт: у приміщенні операторів є два комп'ютери. Комп'ютери об'єднані в локальну мережу. До складу основного обладнання входять ПК та друкуючий пристрій типу Canon PIXMA. У приміщенні є один вхід. Обладнання розміщується таким чином, щоб був забезпечений вільний прохід до всіх робочих місцях. Робота в основному проводиться за ПК.

Планування і розміщення робочих місць: відповідно до загальних ергономічних вимог за ГОСТ 12.2.049-80 виробниче обладнання повинно відповідати антропометричним, фізіологічним, психофізіологічним, властивостям людини та обумовленим цими властивостями гігієнічними вимогами з метою збереження здоров'я людини і досягнення збільшення ефективності праці, зниження стомлюваності.

Столи робочих місць, на якому виробляються діагностика, мають висоту 800 мм, довжину 1400 мм, ширину 800 мм, висота сидіння 450 мм, висота простору для ніг – 650 мм, що забезпечує зручність робочого місця. Висота і конструкція робочого столу вибрані так, щоб було легко переходити з робочого положення сидячи в положення стоячи.

4.4.1 Розрахунок освітлення

Приміщення має розміри 6 м на 9 м. Таким чином, площа складе S = 54 м^2. При висоті H = 2.5 м обсяг приміщення складе V = 135 м^3. Природне освітлення – одностороннє. Приміщення має три вікна висотою 1,5 м, і шириною 2 м, розташованих на одній стіні. Згідно БНІП П-479, коефіцієнт природної освітленості (КПО) при бічному висвітленні повинен складати $e_н$=1,5%. Фактичний коефіцієнт природної освітленості становить 8.33%, що перевищує нормативне значення 1.5%, тобто, КПО у приміщенні достатньо.

Коефіцієнт природної освітленості (КПО) - це процентне відношення природної освітленості у будь-якій точці в середині приміщення до одночасно виміряної на однаковому рівні освітленості зовнішньої горизонтальної площини рівномірно розсіяним (дифузійним) світлом усього небосхилу

Враховуючи коефіцієнт світлового клімату m = 0,8 і коефіцієнт сонячності клімату при тому, що вікна приміщення виходять на східний бік, C = 0,7, визначаємо нормоване значення КПО для даного приміщення:

$$e = e_H \cdot m \cdot C = 1{,}5 \cdot 0{,}5 \cdot 0{,}7 = 0{,}84\% .$$

При бічному освітленні КПО можна оцінити за формулою:

$$e = \frac{\tau_0 \cdot r_1 \cdot S_0}{K_з \cdot \mu_0 \cdot K_{зd} \cdot S_n},$$

де $S_п$ – площа підлоги приміщення, м^2;

S_0– площа світлових прорізів, м^2;

$К_з$ – коефіцієнт запасу (приймається в межах від 1,2 до 2,0 залежно від можливого забруднення світлових прорізів кіптявою);

μ_0 – світлова характеристика вікон;

$К_{зд}$ – коефіцієнт, що враховує підвищення коефіцієнта завдяки світлу, відбитому від поверхні приміщення та підстилаючих шару, який прилягає до будівлі;

τ_0 – загальний коефіцієнт світлопропускання, що визначається за формулою:

$$\tau_0 = \tau_1 + \tau_2 + \tau_3 + \tau_4,$$

де τ_1 – коефіцієнт світлопропускання матеріалу (для різних типів скла приймається в межах від 0,65 до 0,9);

τ_2– коефіцієнт, що враховує втрати світла в межах вікна (в залежності від виду палітурки приймається в межах від 0,5 до 0,9);

τ_3 – коефіцієнт, що враховує втрати світла в несучих конструкціях (від 0,8 до 0,9);

τ_4 – коефіцієнт, що враховує втрати світла в сонцезахисних пристроях (при їх відсутності τ_4=1, при їх наявності тоді приймається від 0,6 до 0,9).

Площа світлових прорізів становить S_0=15,75м^2. Площа підлоги приміщення становить $S_п$ =60 м^2. Для складальних цехів коефіцієнт запасу приймається $К_з$=1,2.

Виходячи зі співвідношення розмірів приміщення і вікон, світлова характеристика вікон μ_0=10,5. Коефіцієнт, що враховує КПО за рахунок 0 відбиття від поверхонь приміщення і підстилаючого шару, для даного приміщення становить r_1 = 1,3.

Для вікон з подвійними рамами коефіцієнт світлопропускання τ_1 =0,8...1. Оскільки палітурка вікна подвійна металопластикова, то τ_2 = 0,6. При бічному освітленні τ_3=1. Як сонцезахисних пристроїв використовуються регульовані 3 внутрішні штори, тому τ_4= 0,9. Таким чином, отримуємо:

$$\tau_1 = 0.8 \cdot 0.6 \cdot 1 \cdot 0.9 = 0.43\,,$$

$$e = \frac{(0.43 \cdot 1.3 \cdot 15.75)}{1.2 \cdot 10.5 \cdot 1 \cdot 60} \cdot 100\% = 1.17\%\,.$$

Тож коефіцієнт природного освітлення e = 1,17% задовольняє нормі. Природне освітлення істотно залежить від погодних умов, отже, необхідно передбачити штучне освітлення в похмуру погоду.

Штучне освітлення необхідно ще й тому, що в приміщенні ведуться роботи не тільки у світлий час доби, але і в темний.

4.4.2 Обґрунтування робочого місця оператора

Виконаємо обґрунтування робочого місця оператора. Зонування моторного поля робочого місця оператора наведено на рисунку 4.1.

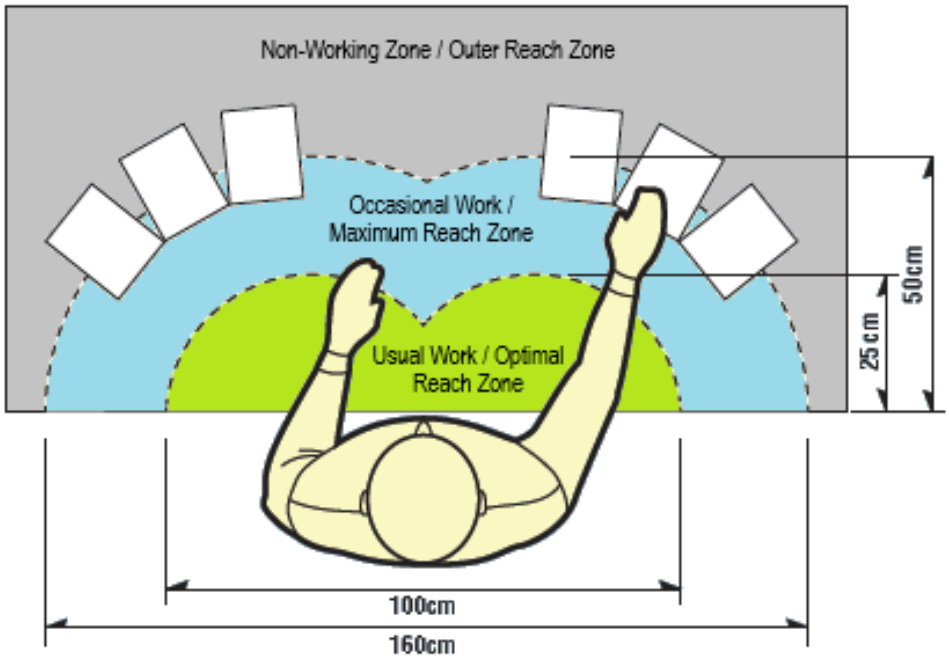

Рисунок 4.1 – Зонування моторного поля оператора

Клавіатуру слід розташовувати на поверхні столу на відстані 100-300 мм від краю, який повернутий до користувача. Кут нахилу до панелі клавіатури має знаходитись в межах від 5° до 15°. Висота клавіатури на рівні середнього ряду не повинна перевищувати 30 мм. Правильні та неправильні положення тіла оператора при роботі за комп’ютером наведені на рис. 4.2.

Рисунок 4.2 – Правильні та неправильні положення тіла оператора (програміста) при роботі за комп’ютером

4.4.3 Ергономічні вимоги до стільців та правильне положення при сидінні

Для забезпечення комфортної та безпечної роботи операторів особливу увагу слід приділити вибору та налаштуванню стільців. Висота сидіння повинна бути відрегульована так, щоб при вертикальному положенні тулуба стегна були майже горизонтальними, а стопи стояли повністю на підлозі; оптимальна висота сидіння – 420-520 мм від підлоги. Кут між стегнами та гомілками має становити приблизно 90°-100°, що знижує навантаження на колінні суглоби.

Спинка стільця повинна мати регульований нахил у межах 100°-110° до сидіння, а також висувну поперекову опору, що повторює природний вигин хребта, для зменшення напруги в поперековому відділі. Кут нахилу сидіння вперед (похил) рекомендується встановити в діапазоні 0°-5°, що сприяє рівномірному перерозподілу тиску та активному положенню тазу.

При тривалій роботі допускається періодичне змінювання кута нахилу (наприклад, 5-15° декілька разів за годину) за допомогою синхромеханізму стільця.

Підлокітники мають бути регульованими за висотою, на рівні 200–250 мм вище сидіння, й шириною, щоб передпліччя спиралися на них під кутом близько 90° у ліктьовому суглобі. Відстань від краю сидіння до коліщок стільця повинна бути не менше 50 мм, щоб уникати надмірного тиску на задню поверхню стегон.

Дотримання цих параметрів дозволяє зменшити статичне навантаження, попередити розвиток м’язово-скелетних розладів та підвищити продуктивність праці операторів. Періодичні перерви кожні 30-45 хвилин та виконання простих гімнастичних вправ сприяють додатковому зняттю напруги і підтриманню здорового тонусу м’язів.

Підберемо робочий стіл та стілець, які задовольняють моторній зоні оператора. На рис. 4.3 зображено модель робочого столу, а на рис. 4.4 зображено модель робочого стільця оператора (програміста).

Рисунок 4.3 – Модель робочого столу оператора (програміста)

Рисунок 4.4 – Модель робочого стільця оператора (програміста)

ВИСНОВКИ

Були виконані проектування та програмна реалізація локальної інформаційної системи моніторингу та управління виробничими процесами на підприємстві.

У роботі проведено аналіз предметної області та існуючих програмних рішень, узагальнено їхні характеристики та сформовано технічне завдання, у якому визначено призначення, функціональні модулі та інтерфейс інформаційної системи. Виконано специфікацію функціональних і нефункціональних вимог, створено глосарій термінів, розроблено концептуальну та функціональну моделі, що відображають потік даних і логіку взаємодії користувачів із системою.

Спроектовано об’єктну модель із UML та ER-діаграмами, у яких сформульовано сутності, атрибути та зв’язки для забезпечення цілісності даних. Розроблено багатошарову компонентну архітектуру за принципом MVC, що гарантує гнучкість, простоту тестування та подальшого розширення. Обґрунтовано вибір Python із фреймворком PySide6 та СКБД SQLite, що забезпечило швидку прототипізацію, кросплатформеність та надійність.

Реалізовано графічний інтерфейс, створено всі необхідні модулі для керування замовленнями, контролю виробничих процесів, внутрішнього чату й автоматичної генерації документів. Проведено дослідну експлуатацію: підготовлено інструкцію користувача, виконано тестування системи, оцінено техніко-економічні показники, забезпечено відповідність вимогам безпеки праці.

Усі поставлені завдання у ході виконання проекту виконано: проведено аналіз і розробку, спроектовано архітектуру та інтерфейс, реалізовано та протестовано програмний продукт. Інформаційна система готова до практичного використання та забезпечує ефективну автоматизацію виробничих процесів згідно з вимогами технічного завдання.

СПИСОК ЛІТЕРАТУРИ

1. Авраменко А.С., Авраменко В.С., Косенюк Г.В. Тестування програмного забезпечення : навч. посіб. Черкаси : ЧНУ імені Богдана Хмельницького, 2017. 284 с.

2. Бусигін Б.С., Дівізінюк М.М., Півняк Г.Г. та ін. Тлумачний словник з інформатики : словник термінів. Дніпро : Нац. гірничий ун-т, 2010. 600 с.

3. Гринчак М.В., Шаповалов А.Л., Кузьмичова К.В., Волков Д.О. Інформаційні системи й технології на підприємстві : конспект лекцій для студентів 4 курсу заочної форми навчання за напрямом підготовки 0501 “Економіка і підприємництво”. Харків : ХНАМГ, 2009. 84 с.

4. Дудзяний І.М. Об’єктно-орієнтоване моделювання програмних систем : навч. посіб. Львів : Видавничий центр ЛНУ імені Івана Франка, 2007. 108 с.

5. Жученко А.І., Ярощук Л.Д. Проєктування інформаційних систем : бази даних : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп’ютерноінтегровані технології». КПІ ім. Ігоря Сікорського. 2-ге вид., допов. Київ : КПІ ім. Ігоря Сікорського, 2022. 166 с.

6. Карпусь Я.В. Переваги та недоліки впровадження ERP-систем для підвищення ефективності діяльності підприємства. *Розвиток європейського простору очима молоді : економічні, соціальні та правові аспекти* : матеріали Всеукр. наук.-практ. конф. Харків : ХНЕУ ім. С. Кузнеця, 2016. С. 554–559.

7. Новокшонов А.К. Аналіз ефективності реалізації арифметичних алгоритмів на мовах програмування C++ та Python. *Проблеми програмування*. № 2-3(спец. вип.). 2016. С. 26-31.

8. Пересадько Г.О. Важка промисловість України: історія, сучасний стан та перспективи розвитку. *Ринкова система України: стан та перспективи розвитку: монографія* / за заг. ред. О.В. Макарюка, В.М. Жмайлова, Ю.І. Данька. Харків : Міськдрук, 2011. С. 775–804.

9. Пилипенко Л.М., Редько М.О. Аналіз переваг та недоліків упровадження ERP-системи на підприємствах. *Приазовський економічний вісник*. 2019. № 6(17). С. 172–178

10. Сазонець О.М. Інформаційні системи і технології в управлінні зовнішньоекономічною діяльністю: навч. посіб. Київ : Центр учбової літератури, 2014. 256 с.

11. Ушакова І.О., Плеханова Г.О. Інформаційні системи та технології на підприємстві : конспект лекцій. Харків : Вид. ХНЕУ, 2009. 128 с.

12. Шаров С.В., Лубко Д.В., Зінов'єва О.Г. Методичні рекомендації до виконання та захисту кваліфікаційної роботи бакалавра спеціальності 122 «Комп'ютерні науки». Запоріжжя : ТДАТУ, 2025. 63 с.

13. Шибаєва, Н.О., Березоруцька О.В., Бут Н.В. Особливості архітектурного патерну MVC : *Інформатика, інформаційні системи та технології*: тези доповідей шістнадцятої всеукраїнської конференції студентів і молодих науковців. Одеса, 23 квітня 2021 р. С. 71-73.

14. Crawford M. How Industry 4.0 impacts engineering design : [Електронний ресурс] / ASME. 2018. URL: https://www.asme.org/topics-resources/content/industry-40-impacts-engineering-design

15. Farrell, S., Nielsen, J. Strategic design for frequently asked questions : Nielsen Norman Group. Fremont, CA : Nielsen Norman Group, 2012. 69 с.

16. Fitzpatrick M. Create GUI applications with Python & Qt6: the hands-on guide to making apps with Python. Version 5.0. Self-published via Leanpub: Martin Fitzpatrick, 2022. 805 с.

17. Information system: automation, data processing, decision support : [Електронний ресурс]. Encyclopædia Britannica. URL: https://www.britannica.com/topic/information-system/Operational-support-and-enterprise-systems

18. Kreibich J. A. Using SQLite. O'Reilly Media, Sebastopol, CA, 2010. 503 с.

19. Matthes E. Python crash course: a hands-on, project-based introduction to programming – 2nd ed. San Francisco : No Starch Press, 2019. 544 c.

20. Tripathi V., Chattopadhyaya S., Mukhopadhyay A.K., Saraswat S., Sharma S., Li C., Rajkumar S., Georgise F.B., Chang K.H. A novel smart production management system for the enhancement of industrial sustainability in Industry 4.0. *Mathematical Problems in Engineering*. 2022. С. 1–24.

21. Voggesser R. How data analytics is transforming the manufacturing industry: [Електронний ресурс]. RSM Technology Blog. 12 листопада 2024. URL: https://technologyblog.rsmus.com/industry/manufacturing/how-data-analytics-is-transforming-the-manufacturing-industry/

ДОДАТОК А

Сторінки інтерфейсу програми

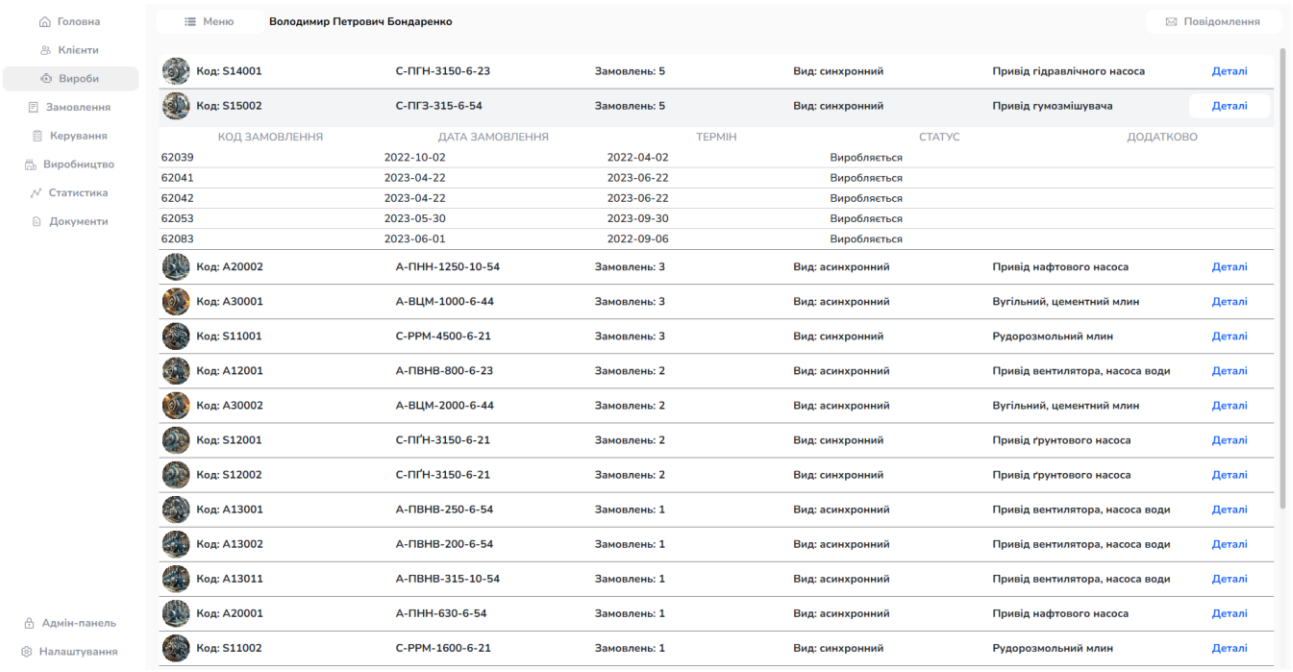

Рисунок А.1 – Сторінка виробів

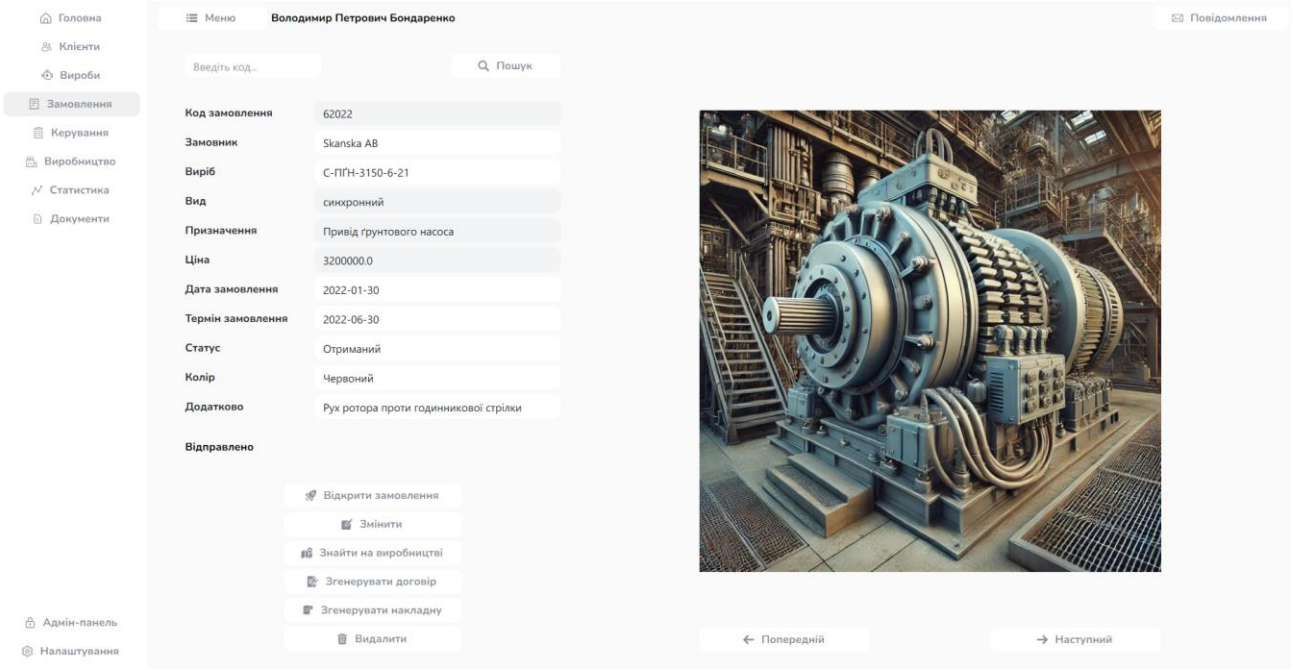

Рисунок А.2 – Сторінка карток замовлень

ДОДАТОК Б

Фрагмент лістингу коду програми файлу «main.py», який виконує налаштування таблиці поточних замовлень на головній сторінці

```
     # Головна сторінка

def main_load_current_orders(self):
    # Завантажує та відображає поточні замовлення на головній
сторінці.

    try:
        # Підключення до бази даних
        connection = sqlite3.connect("DBAdm.db")
        connection.row_factory = sqlite3.Row
        cursor = connection.cursor()

        # SQL-запит для отримання поточних замовлень з обчисленням
поточного місця
        query = """
            SELECT
                o.code_order,
                m.job_motor AS engine_name,
                m.type_motor AS engine_type,
                c.customer AS customer_name,
                c.state_customer AS customer_country,
                m.price AS price,
                o.termin_order,
                CASE
                    WHEN o.status_order = 'Отримано' THEN
'Отриманий'
                    WHEN EXISTS (SELECT 1 FROM t_sentGoods WHERE
code_order = o.code_order) THEN 'Відправлений'
                    WHEN EXISTS (SELECT 1 FROM t_craftingGoods
WHERE code_order = o.code_order) THEN 'На збірці'
                    WHEN EXISTS (SELECT 1 FROM t_paintingGoods
WHERE code_order = o.code_order) THEN 'На фарбуванні'
                    WHEN EXISTS (SELECT 1 FROM t_testingGoods WHERE
code_order = o.code_order) THEN 'На тестуванні'
                    WHEN EXISTS (SELECT 1 FROM t_packingGoods WHERE
code_order = o.code_order) THEN 'На пакуванні'
                    WHEN EXISTS (SELECT 1 FROM t_storingGoods WHERE
code_order = o.code_order) THEN 'На складі'
                    ELSE 'На виробництві'
                END AS current_location
            FROM t_orders o
            JOIN t_motors m ON o.code_motor = m.code_motor
```

```
            JOIN t_customers c ON o.code_customer = c.code_customer
            WHERE o.code_order NOT IN (SELECT code_order FROM
t_sentGoods)
              AND o.status_order <> 'Отримано'
            ORDER BY o.date_order ASC
        """
        cursor.execute(query)
        orders = cursor.fetchall()

        # Визначення заголовків таблиці
        headers = [
            "КОД ЗАМОВЛЕННЯ",
            "ВИД ДВИГУНА",
            "ТИП ДВИГУНА",
            "ЗАМОВНИК",
            "КРАЇНА ЗАМОВНИКА",
            "ЦІНА",
            "ТЕРМІН ЗАМОВЛЕННЯ",
            "ПОТОЧНЕ МІСЦЕ"
        ]
        table_widget = self.ui.main_current_orders_tableWidget
        table_widget.setRowCount(0)
        table_widget.setColumnCount(len(headers))
        table_widget.setHorizontalHeaderLabels(headers)
        table_widget.setItemDelegate(CustomDelegate())

table_widget.horizontalHeader().setSectionResizeMode(QHeaderView.St
retch)
        table_widget.verticalHeader().setVisible(False)
        table_widget.setShowGrid(False)

table_widget.setEditTriggers(QAbstractItemView.NoEditTriggers)

        # Встановлення шрифтів: звичайний та жирний
        base_font = QFont()
        bold_font = QFont()
        bold_font.setBold(True)
        today = datetime.datetime.now().date()

        # Заповнення таблиці рядками даних
        for row_index, order in enumerate(orders):
            table_widget.insertRow(row_index)

            # Колонка 0: Код замовлення – жирним шрифтом
            item_code = QTableWidgetItem(str(order["code_order"]))
            item_code.setFont(bold_font)
            table_widget.setItem(row_index, 0, item_code)

            # Колонка 1: Вид двигуна
```

```
            item_engine = QTableWidgetItem(order["engine_name"] if
order["engine_name"] else "")
            item_engine.setFont(base_font)
            table_widget.setItem(row_index, 1, item_engine)

            # Колонка 2: Тип двигуна
            item_engine_type =
QTableWidgetItem(order["engine_type"] if order["engine_type"] else
"")
            item_engine_type.setFont(base_font)
            table_widget.setItem(row_index, 2, item_engine_type)

            # Колонка 3: Назва замовника
            item_customer = QTableWidgetItem(order["customer_name"]
if order["customer_name"] else "")
            item_customer.setFont(base_font)
            table_widget.setItem(row_index, 3, item_customer)

            # Колонка 4: Країна замовника
            item_country =
QTableWidgetItem(order["customer_country"] if
order["customer_country"] else "")
            item_country.setFont(base_font)
            table_widget.setItem(row_index, 4, item_country)

            # Колонка 5: Ціна – форматована з символом гривні
            price_value = order["price"] if order["price"] is not
None else 0
            item_price = QTableWidgetItem(f"₴{price_value:.2f}")
            item_price.setFont(base_font)
            table_widget.setItem(row_index, 5, item_price)

            # Колонка 6: Термін замовлення – з кольоровим
виділенням залежно від терміну
            termin_text = order["termin_order"] if
order["termin_order"] else ""
            item_termin = QTableWidgetItem(termin_text)
            item_termin.setFont(bold_font)
            if termin_text:
                try:
                    termin_date =
datetime.datetime.strptime(termin_text, "%Y-%m-%d").date()
                    delta = (termin_date - today).days
                    if delta > 7:
                        bg_color = QColor(40, 199, 111)  # зелений
                    elif 0 <= delta <= 7:
                        bg_color = QColor("#ffc600")
                    else:
                        bg_color = QColor(234, 84, 85)  # червоний
```

```
                    item_termin.setForeground(QBrush(bg_color))
                    font = item_termin.font()
                    font.setBold(True)
                    item_termin.setFont(font)
                except ValueError:
                    pass
            table_widget.setItem(row_index, 6, item_termin)

            # Колонка 7: Поточне місце
            item_location =
QTableWidgetItem(order["current_location"] if
order["current_location"] else "")
            item_location.setFont(base_font)
            table_widget.setItem(row_index, 7, item_location)

        # Застосування стилів до таблиці, включаючи стилі для
заголовків і скролбарів
        table_widget.setStyleSheet("""
            QTableWidget {
                background-color: #ffffff;
                border: none;
                border-radius: 12px;
                gridline-color: transparent;
                color: #23272e;
                font: 600 12pt "Nunito"
            }
            QHeaderView,
            QHeaderView::section,
            QTableCornerButton {
                background: transparent;
                border: none;
                color: #8b909a;
                font: 600 12pt "Nunito";
                margin: 5px;
            }
            QScrollBar:vertical {
                background: transparent;
                width: 12px;
                margin: 0px;
            }
            QScrollBar::handle:vertical {
                background: gray;
                border-radius: 5px;
    ...
```