

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

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

на тему:

**«Вебсистема автоматизації діяльності малого
підприємства каменеобробної галузі»**

Виконав студент групи ПЗ-22-2
КОРОЛЬ Віталій Валерійович

Керівник роботи:
ЛОКТИКОВА Тамара Миколаївна

Рецензент:
ЛОБАНЧИКОВА Надія Миколаївна

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

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення

Тетяна Вакалюк

«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОРОЛЬ Віталій Валерійович**

Керівник роботи: **ЛОКТИКОВА Тамара Миколаївна**

Тема роботи: **«Вебсистема автоматизації діяльності малого підприємства каменеобробної галузі»**, затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., №92/с**

Вихідні дані для роботи: **об'єктом дослідження є процес автоматизації управління клієнтами, замовленнями та виробничими операціями малого підприємства каменеобробної галузі, а предметом дослідження є методи та засоби розробки веборієнтованої системи для обліку клієнтів, сировини, готової продукції, працівників, замовлень, відправок, розрахунку вартості робіт і аналізу основних показників діяльності підприємства.**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Локтікова Т.М.	02.03.2026р.	02.03.2026р.
2	Локтікова Т.М.	02.03.2026р.	02.03.2026р.
3	Локтікова Т.М.	02.03.2026р.	02.03.2026р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з вебсистеми автоматизації діяльності малого підприємства каменеобробної галузі та пояснювальної записки. Пояснювальна записка до випускної роботи містить текстову частину, викладену на 101 сторінці друкованого тексту, список використаних джерел містить 40 найменувань і займає 4 сторінки. У роботі наведено 20 рисунків та 12 таблиць. У роботі є 29 сторінок додатків. Загальна кількість сторінок пояснювальної записки становить 130.

Метою роботи є розробка та реалізація вебсистеми автоматизації діяльності малого підприємства каменеобробної галузі, яка дозволяє централізовано вести облік клієнтів, замовлень, сировини, готової продукції, працівників, прайсу граніту, пакування, відправок, витрат, зарплат і фінансових показників діяльності підприємства.

Для реалізації вебсистеми обрано клієнт-серверну веборієнтовану архітектуру з використанням архітектурного підходу MVC. Серверну частину реалізовано мовою PHP, для збереження даних використано MySQL, а для побудови клієнтської частини застосовано HTML, CSS і JavaScript. Вебсистема має модульну структуру та охоплює авторизацію користувачів, роботу з клієнтами, замовленнями, прайсом граніту, поставками сировини, складом сировини, розкромом, складом готової продукції, пакуванням, відправками, зарплатами, витратами, інструментом, фінансовою аналітикою, підпискою та журналом дій.

КЛЮЧОВІ СЛОВА: ВЕБСИСТЕМА, АВТОМАТИЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ, МАЛЕ ПІДПРИЄМСТВО, КАМЕНЕОБРОБНА ГАЛУЗЬ, MVC, PHP, JAVASCRIPT, MYSQL.

					ІПЗ.КР.Б – 121 – 26 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Вебсистема автоматизації діяльності малого підприємства каменеобробної галузі. Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		Король В.В.						
Керівник		Локтікова Т.М					3	130
Рецензент		Лобанчикова Н. М.				Житомирська політехніка, група ІПЗ-22-2		
Зав. Каф.		Вакалюк Т.А.						

ABSTRACT

The bachelor's qualification work consists of a web system for automating the activities of a small enterprise in the stone processing industry and an explanatory note. The explanatory note contains the main text presented on 101 printed pages; the list of references includes 40 sources and occupies 4 pages. The work contains 20 figures and 12 tables. The appendices occupy 29 pages. The total number of pages of the explanatory note is 130.

The aim of the work is to develop and implement a web system for automating the activities of a small enterprise in the stone processing industry, which enables centralized accounting of clients, orders, raw materials, finished products, employees, granite price lists, packaging, shipments, expenses, salaries, and financial performance indicators of the enterprise.

A client-server web-oriented architecture using the MVC architectural pattern was chosen for the implementation of the web system. The server side was implemented in PHP, MySQL was used for data storage, and HTML, CSS, and JavaScript were applied to build the client side. The web system has a modular structure and includes user authorization, client management, order management, granite price list management, raw material deliveries, raw material warehouse accounting, cutting operations, finished product warehouse accounting, packaging, shipments, payroll, expenses, tools, financial analytics, subscription management, and an action log.

KEYWORDS: WEB SYSTEM, BUSINESS PROCESS AUTOMATION, SMALL ENTERPRISE, STONE PROCESSING INDUSTRY, MVC, PHP, JAVASCRIPT, MYSQL.

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ АВТОМАТИЗАЦІЇ ДІЯЛЬНОСТІ МАЛОГО ПІДПРИЄМСТВА КАМЕНЕОБРОБНОЇ ГАЛУЗІ	10
1.1. Постановка задачі	10
1.2. Аналіз аналогів програмного продукту	14
1.3. Вибір архітектури системи	23
1.4. Обґрунтування вибору інструментальних засобів реалізації	28
Висновки до розділу 1	33
РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБСИСТЕМИ	35
2.1. Визначення варіантів використання системи	35
2.2. Розробка бази даних системи	43
2.3. Проєктування та реалізація алгоритмів роботи системи	49
2.4. Проєктування та реалізація функціональних модулів системи	54
Висновки до розділу 2	65
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ	67
3.1. Порядок встановлення та налаштування параметрів системи	67
3.2. Структура інтерфейсу системи	74
3.3. Тестування роботи системи	83
Висновки до розділу 3	92
ВИСНОВКИ	94
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	98
ДОДАТКИ	102

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних.

СКБД – система керування базами даних.

API – Application Programming Interface, програмний інтерфейс застосунку.

CRM – Customer Relationship Management, система управління взаємовідносинами з клієнтами.

CSS –Cascading Style Sheets, каскадні таблиці стилів.

HTML – HyperText Markup Language, мова гіпертекстової розмітки.

HTTP – HyperText Transfer Protocol, протокол передавання гіпертексту.

HTTPS – HyperText Transfer Protocol Secure, захищений протокол передавання гіпертексту.

JS – JavaScript, мова програмування для реалізації інтерактивної поведінки вебсторінок.

JSON – JavaScript Object Notation, текстовий формат обміну даними.

MVC – Model-View-Controller, архітектурний шаблон побудови програмного забезпечення.

MySQL – система керування реляційними базами даних.

OAuth – Open Authorization, відкритий протокол авторизації.

PDO – PHP Data Objects, інтерфейс доступу PHP до баз даних.

PHP – Hypertext Preprocessor, серверна мова програмування.

SQL – Structured Query Language, мова структурованих запитів.

SMTP – Simple Mail Transfer Protocol, протокол передавання електронної пошти.

UI – User Interface, інтерфейс користувача.

URL – Uniform Resource Locator, уніфікований покажчик ресурсу.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми. У сучасних умовах розвитку малого бізнесу особливого значення набуває впровадження інформаційних технологій, які дають змогу автоматизувати облік, управління даними та контроль основних операційних процесів підприємства. Для невеликих підприємств це є особливо важливим, оскільки значна частина управлінських і виробничих операцій часто виконується вручну або за допомогою розрізнених інструментів: електронних таблиць, паперових записів, месенджерів, окремих файлів чи неструктурованих баз даних. Такий підхід ускладнює пошук актуальної інформації, підвищує ризик помилок, дублювання даних, втрати важливих відомостей і знижує ефективність управління підприємством.

Каменеобробна галузь має низку специфічних особливостей, які потребують окремого підходу до автоматизації. Діяльність малого підприємства у цій сфері пов'язана не лише з обліком клієнтів, а й із веденням замовлень, контролем сировини, обліком готової продукції, збереженням інформації про типи граніту, розміри матеріалів, дефекти, залишки, прайси, відправки, виконані роботи та розрахунок заробітної плати працівників. Кожне замовлення може містити індивідуальні параметри, залежати від конкретного виду матеріалу, типу обробки, розмірів деталей, додаткових послуг і стану виконання. Тому використання лише стандартних засобів обліку або універсальних CRM-систем не завжди повністю відповідає реальним потребам каменеобробного підприємства.

Актуальність розробки системи автоматизації діяльності малого підприємства каменеобробної галузі полягає у необхідності створення програмного рішення, яке дозволить централізовано зберігати та обробляти дані про основні бізнес-процеси підприємства. Така система має забезпечити зручну роботу із замовленнями, клієнтами, сировиною, продукцією, працівниками, відправками та фінансово-виробничими показниками. Її використання дає змогу

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

зменшити кількість ручних операцій, підвищити точність обліку, прискорити доступ до інформації та покращити контроль за виконанням замовлень.

Метою кваліфікаційної роботи є розробка алгоритмів роботи та реалізація вебсистеми автоматизації діяльності малого підприємства каменеобробної галузі, призначеної для обліку замовлень, сировини, готової продукції, клієнтів, працівників, прайсів граніту, відправок, зарплат та інших пов'язаних операційних процесів.

Для досягнення поставленої мети необхідно виконати такі **завдання**: провести аналіз предметної області та визначити основні процеси діяльності малого підприємства каменеобробної галузі, які потребують автоматизації; дослідити існуючі програмні рішення для автоматизації малого бізнесу, визначити їхні функціональні можливості, переваги та обмеження з урахуванням специфіки каменеобробного підприємства; сформулювати функціональні й нефункціональні вимоги до системи, визначити основні ролі користувачів і сценарії їхньої взаємодії з програмним продуктом; вибрати архітектуру системи; обґрунтувати вибір інструментальних засобів реалізації; спроектувати структуру системи; розробити базу даних; реалізувати основні функціональні модулі системи; забезпечити адаптивність інтерфейсу і провести тестування розробленого програмного продукту.

Об'єктом дослідження є процеси організації та управління діяльністю малого підприємства каменеобробної галузі, пов'язані з обліком замовлень, сировини, готової продукції, працівників, відправок, розрахунків і взаємодії з клієнтами.

Предметом дослідження є методи та засоби проектування й розробки системи автоматизації бізнес-процесів малого підприємства каменеобробної галузі з використанням клієнт-серверної архітектури, архітектурного підходу MVC, мови програмування PHP та бази даних MySQL.

Практичне значення отриманих результатів полягає у створенні програмного продукту, який може бути використаний малим підприємством каменеобробної галузі для автоматизації основних облікових, виробничих та

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						8
Змн.	Арк.	№ докум.	Підпис	Дата		

управлінських процесів. Розроблена система дозволяє централізувати дані про замовлення, сировину, готову продукцію, клієнтів, працівників, відправки, прайси граніту та зарплати. Її використання сприяє зменшенню обсягу ручної роботи, підвищенню точності обліку, спрощенню контролю за виконанням замовлень і покращенню організації діяльності підприємства.

За результатами випускної кваліфікаційної роботи бакалавра опубліковані тези:

Король В.В., Кушнір Н.О., Локтікова Т.М. Побудова та проектування вебсистеми управління взаємовідносинами з клієнтами для малого бізнесу: тези Всеукраїнської науково - практичної on-line конференції здобувачів вищої освіти і молодих учених, присвяченої Дню науки, м. Житомир, 11 – 15 травня 2026 р. Житомир: «Житомирська політехніка», 2026. С. 90 – 91 [1].

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ АВТОМАТИЗАЦІЇ ДІЯЛЬНОСТІ МАЛОГО ПІДПРИЄМСТВА КАМЕНЕОБРОБНОЇ ГАЛУЗІ

1.1. Постановка задачі

Мале підприємство каменеобробної галузі у процесі щоденної діяльності виконує значну кількість взаємопов'язаних облікових, виробничих, складських та управлінських операцій. До них належать приймання та облік сировини, збереження інформації про види граніту, формування замовлень, облік готової продукції, ведення прайсу, робота з клієнтами, контроль відправок, облік працівників, фіксація виконаних робіт і розрахунок заробітної плати. Ефективність роботи такого підприємства залежить від того, наскільки швидко та точно здійснюється обробка інформації про матеріали, замовлення, продукцію, працівників і поточний стан виконання робіт.

Особливістю каменеобробної галузі є те, що кожне замовлення має індивідуальний характер і може містити значну кількість параметрів. Під час його оформлення потрібно враховувати назву матеріалу, тип граніту, розміри деталей, товщину, кількість виробів, вид обробки, додаткові послуги, знижку, загальну вартість, відповідальних працівників і подальшу відправку готової продукції. Крім того, важливим є зв'язок замовлення із залишками сировини та готовою продукцією, оскільки від наявності потрібного матеріалу залежить можливість виконання роботи у визначені строки.

На практиці в малому бізнесі подібні процеси часто ведуться вручну або за допомогою окремих інструментів, які не мають єдиної інформаційної структури. Дані про клієнтів можуть зберігатися в телефоні або месенджері, інформація про замовлення – в електронній таблиці, залишки сировини – в окремому файлі, а відомості про виконані роботи та зарплати – у паперових записах чи інших документах. У результаті підприємство не має єдиного середовища, у якому можна швидко отримати повну й актуальну інформацію про поточний стан діяльності.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						10
Змн.	Арк.	№ докум.	Підпис	Дата		

Такий підхід створює низку практичних проблем. Ускладнюється пошук потрібних даних, зростає ризик дублювання записів, помилок під час розрахунків, втрати частини інформації або використання застарілих відомостей. Окремою проблемою є контроль статусів замовлень: без централізованої системи важко швидко визначити, які замовлення тільки створені, які перебувають у роботі, які вже виконані, а які підготовлені до відправки. Також ускладнюється контроль залишків сировини, наявності готової продукції та виконаних робіт працівників.

Стандартні CRM-системи частково вирішують задачу впорядкування інформації про клієнтів і продажі, однак вони не повністю відповідають потребам каменеобробного підприємства. Основний функціонал класичних CRM-систем зазвичай зосереджений на веденні клієнтської бази, угодах, комунікаціях, нагадуваннях і воронці продажів. Для підприємства каменеобробної галузі цього недостатньо, оскільки значну роль відіграють виробничий, складський і фінансово-обліковий напрями. Тому доцільною є розробка спеціалізованої системи автоматизації, яка враховує специфіку саме цієї предметної області.

Задачею кваліфікаційної роботи є проектування та реалізація системи автоматизації діяльності малого підприємства каменеобробної галузі, яка забезпечує централізоване зберігання, обробку та використання даних про основні бізнес-процеси підприємства. Розроблювана вебсистема має об'єднати в єдиному інформаційному середовищі дані про замовлення, клієнтів, сировину, готову продукцію, працівників, прайс граніту, відправки, виконані роботи та зарплати.

Однією з основних задач є реалізація модуля роботи із замовленнями. Цей модуль має забезпечувати створення нового замовлення, вибір клієнта, вибір матеріалу, зазначення розмірів, товщини, кількості деталей, додаткових послуг, коментарів, знижки та підсумкової вартості. Також необхідно передбачити можливість зміни статусу замовлення, що дозволить відстежувати його проходження від моменту створення до виконання або відправки. Замовлення є

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						11
Змн.	Арк.	№ докум.	Підпис	Дата		

центральною сутністю системи, оскільки саме воно поєднує клієнта, матеріал, виробничі параметри, фінансові розрахунки та стан виконання роботи.

Окремою задачею є реалізація клієнтського обліку. Незважаючи на те, що розроблювана система не обмежується лише взаємовідносинами з клієнтами, інформація про клієнтів залишається важливою частиною її функціональності. Система має забезпечувати збереження контактних даних, прив'язку клієнтів до замовлень, можливість пошуку та перегляду інформації про попередні звернення або виконані роботи. Це дозволить швидше працювати з повторними замовленнями та підтримувати цілісність даних про клієнтів.

Важливим напрямом автоматизації є облік сировини та готової продукції. У системі необхідно реалізувати можливість додавання матеріалів із зазначенням назви граніту, типу сировини, габаритних розмірів, товщини, дати надходження, постачальника, інформації про дефекти, корисні розміри та фотографії. Також система має забезпечувати збереження інформації про готові вироби, їхні розміри, матеріал, тип обробки, фотографії та поточний стан. Такий підхід дозволить створити структурований електронний склад, у якому користувач зможе швидко знаходити потрібний матеріал або готову продукцію.

Ще однією задачею є створення модуля прайсу граніту та розрахунку вартості замовлення. Оскільки вартість робіт залежить від матеріалу, товщини, одиниці вимірювання, площі або об'єму, система повинна забезпечувати збереження цінової інформації у структурованому вигляді. Це дозволить використовувати прайс під час оформлення замовлення, зменшити кількість ручних обчислень, знизити ризик помилок і зробити процес розрахунку більш прозорим.

Система також має підтримувати облік працівників і розрахунок заробітної плати. Для цього необхідно передбачити збереження інформації про працівників, їхні ролі, ставки, виконані роботи та нарахування. У каменеобробному підприємстві різні працівники можуть виконувати різні види робіт, зокрема фрезерування, полірування, роботу на верстатах із числовим програмним керуванням, допоміжні або адміністративні операції. Автоматизація цього

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						12
Змн.	Арк.	№ докум.	Підпис	Дата		

процесу дозволить спростити контроль виконаних робіт і зменшити ризик помилок під час розрахунку зарплати.

Окрему увагу потрібно приділити модулю відправок і контролю завершення замовлення. Після виконання робіт готова продукція може бути передана клієнту або підготовлена до транспортування. Система має забезпечувати збереження інформації про відправлення, пов'язані замовлення, статус відправки та інші необхідні дані. Це дозволить контролювати не лише процес виготовлення, а й завершальний етап роботи із замовленням.

Для забезпечення зручної та безпечної роботи необхідно реалізувати розподіл прав доступу між користувачами. Адміністратор повинен мати доступ до управління користувачами, клієнтами, замовленнями, сировиною, готовою продукцією, прайсом, працівниками, зарплатами, відправками та аналітичною інформацією. Працівник має отримувати доступ лише до тих функцій, які потрібні для виконання його завдань. За потреби клієнт може мати обмежений доступ до інформації про власні замовлення або їхній поточний стан.

Важливою задачею є створення зручного інтерфейсу користувача. Оскільки система орієнтована на практичне використання в малому бізнесі, її інтерфейс має бути зрозумілим, логічним і не перевантаженим зайвими елементами. Користувач повинен мати можливість швидко перейти до потрібного розділу, знайти необхідний запис, створити нове замовлення, додати матеріал, переглянути готову продукцію або перевірити інформацію про працівників. Також необхідно забезпечити адаптивність інтерфейсу, щоб система коректно працювала як на персональних комп'ютерах, так і на мобільних пристроях.

Очікуваним результатом виконання кваліфікаційної роботи є працездатна система автоматизації діяльності малого підприємства кам'янообробної галузі. Розроблений програмний продукт має забезпечувати централізовану роботу з основними даними підприємства, підвищувати точність обліку, зменшувати кількість ручних операцій, спрощувати контроль виконання замовлень і покращувати організацію виробничих, складських та управлінських процесів.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						13
Змн.	Арк.	№ докум.	Підпис	Дата		

Реалізація такої системи дозволить створити основу для подальшої цифровізації діяльності малого підприємства та підвищення ефективності його роботи.

1.2. Аналіз аналогів програмного продукту

У сфері автоматизації діяльності малого бізнесу існує значна кількість програмних продуктів, призначених для впорядкування роботи з клієнтами, продажами, задачами, угодами, фінансовими документами та внутрішньою комунікацією. Найпоширенішим класом таких рішень є CRM-системи, які дозволяють зберігати інформацію про клієнтів, контролювати етапи продажів, фіксувати звернення, планувати задачі та аналізувати взаємодію з контрагентами [2]. Для багатьох підприємств такого функціоналу достатньо, оскільки основна частина їхньої діяльності пов'язана з продажами, обробкою заявок і підтриманням комунікації з клієнтами.

Разом з тим система, що розробляється в межах випускної кваліфікаційної роботи бакалавра, не обмежується лише класичним CRM-функціоналом. Її призначенням є автоматизація діяльності малого підприємства каменеобробної галузі, де поряд із клієнтами та замовленнями важливими є облік сировини, готової продукції, прайсу граніту, відправок, працівників і заробітної плати. Тому під час аналізу аналогів необхідно оцінити не лише загальні можливості CRM-систем, а й те, наскільки вони можуть бути пристосовані до специфічних виробничих і складських процесів каменеобробного підприємства [3, 4].

Для аналізу було обрано три програмні продукти: Uspasy, Perfex CRM та Pipedrive CRM. Таке поєднання аналогів дозволяє розглянути різні підходи до побудови CRM-рішень. Uspasy є українською CRM-платформою та цифровим робочим простором для організації комунікації, задач і продажів. Perfex CRM є CRM-системою, яка поєднує роботу з клієнтами, проектами, задачами, інвойсами та фінансовими документами. Pipedrive CRM є міжнародною CRM-системою, основний акцент якої зроблено на лідах, угодах, контактах і воронці продажів. Порівняння цих рішень із розроблюваною системою дає змогу

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дата		

визначити типовий функціонал CRM-продуктів, їхні обмеження та відмінності власної розробки.

Uspasy [5] – це українська CRM-орієнтована платформа, яка поєднує засоби комунікації, спільної роботи, управління задачами та продажами. Система позиціонується як єдиний цифровий робочий простір, у якому працівники компанії можуть вести комунікацію, працювати із задачами, клієнтами, угодами та іншими елементами робочого процесу. Такий підхід дає можливість зменшити кількість окремих сервісів, якими користується підприємство, та об’єднати основні процеси в одному середовищі.

Загальний вигляд CRM-модуля Uspasy наведено на рис. 1.1.

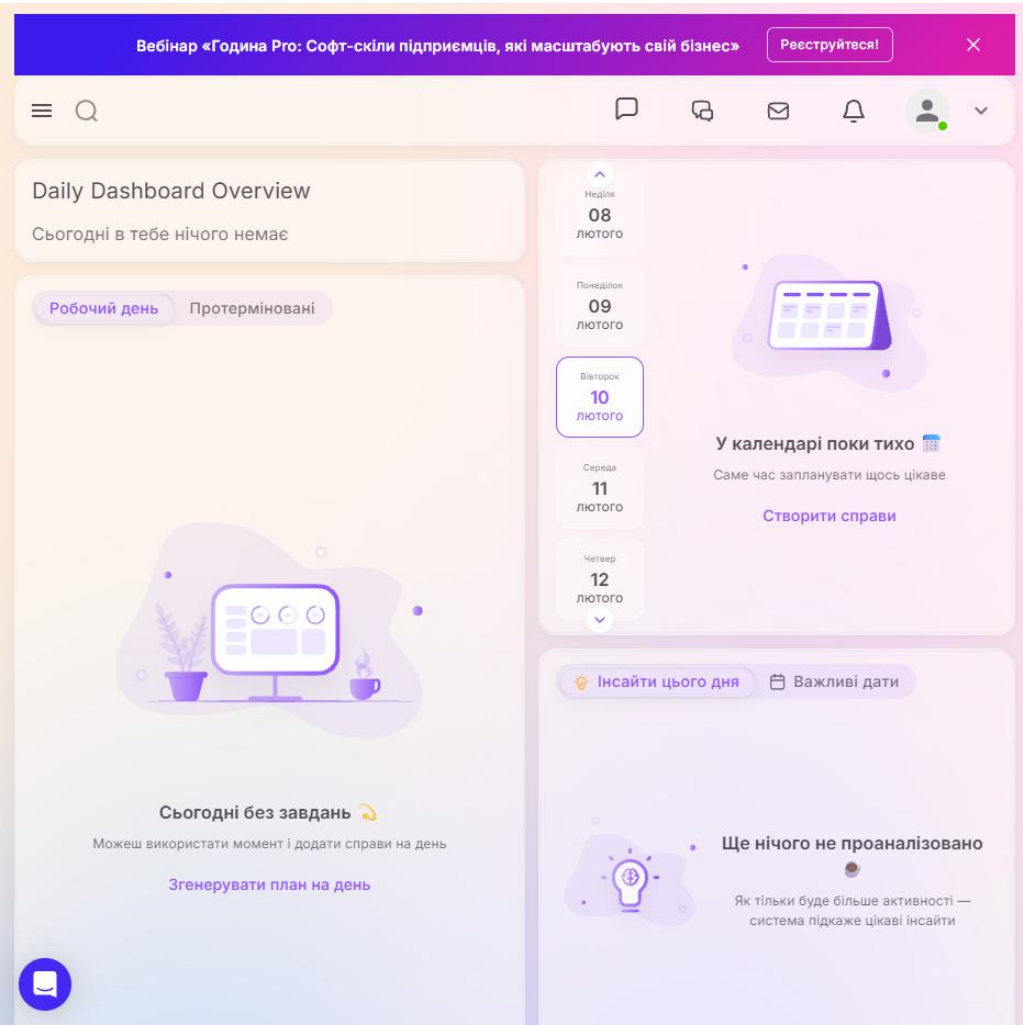


Рис. 1.1. Інтерфейс CRM-модуля Uspasy

Аналіз функціональних можливостей Uspasy показує, що система орієнтована на організацію роботи команди та впорядкування процесів продажу. У ній передбачено роботу з клієнтами, лідами, контактами, компаніями, угодами,

задачами та комунікацією між працівниками. Для малого бізнесу такий функціонал є корисним, оскільки дозволяє зберігати клієнтську інформацію в одному середовищі, контролювати етапи роботи з угодами та пов'язувати домовленості з конкретними задачами для працівників. Наявність цифрового простору для комунікації також зменшує потребу у використанні сторонніх месенджерів для обговорення робочих питань.

CRM-модуль Uspasy може бути використаний для ведення воронки продажів, контролю лідів та угод, перегляду історії взаємодії з клієнтами і постановки задач відповідальним працівникам. У контексті малого підприємства це дозволяє систематизувати роботу з потенційними та наявними клієнтами, зменшити ризик втрати звернень і підвищити прозорість продажів. Також важливим є те, що Uspasy орієнтована на український ринок, має зрозумілий для українських користувачів інтерфейс і може бути швидко впроваджена без складного налаштування серверної інфраструктури.

Водночас у контексті автоматизації каменеобробного підприємства Uspasy покриває лише частину необхідних процесів. Система добре підходить для роботи з клієнтами, задачами, угодами та комунікацією, однак не містить спеціалізованої логіки для обліку сировини, готової продукції, фізичних параметрів матеріалів, дефектів, корисних розмірів, типів обробки, прайсу граніту, відправок і розрахунку заробітної плати працівників. Каменеобробне підприємство потребує не лише контролю клієнтських звернень, а й повноцінного обліку матеріалів, виробів, виконаних робіт і фінансово-виробничих операцій.

Особливістю Uspasy є хмарна модель використання, яка спрощує доступ до системи, не потребує самостійного адміністрування серверної частини та дозволяє швидко розпочати роботу. Однак для розроблюваної системи важливою є можливість глибокої адаптації структури даних і бізнес-логіки під конкретні процеси каменеобробного підприємства. У готових CRM-рішеннях такі можливості можуть бути обмеженими, оскільки структура даних і логіка роботи визначаються можливостями самої платформи. Отже, Uspasy доцільно

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

розглядати як аналог у частині CRM-функцій, задач і комунікації, але не як повноцінне рішення для автоматизації всіх процесів каменеобробного підприємства.

Perfex CRM [6] – це CRM-система, орієнтована на малий бізнес, фрилансерів і компанії, які працюють із клієнтами, проектами, задачами, рахунками та пропозиціями. Система містить набір функцій для управління клієнтською базою, лідами, проектами, задачами, інвойсами, договорами, витратами, платежами та зверненнями до служби підтримки. Такий підхід дозволяє використовувати Perfex CRM не лише для фіксації контактів, а й для супроводу проєктів і фінансових документів, пов’язаних із клієнтами.

Головну інформаційну панель Perfex CRM наведено на рис. 1.2.

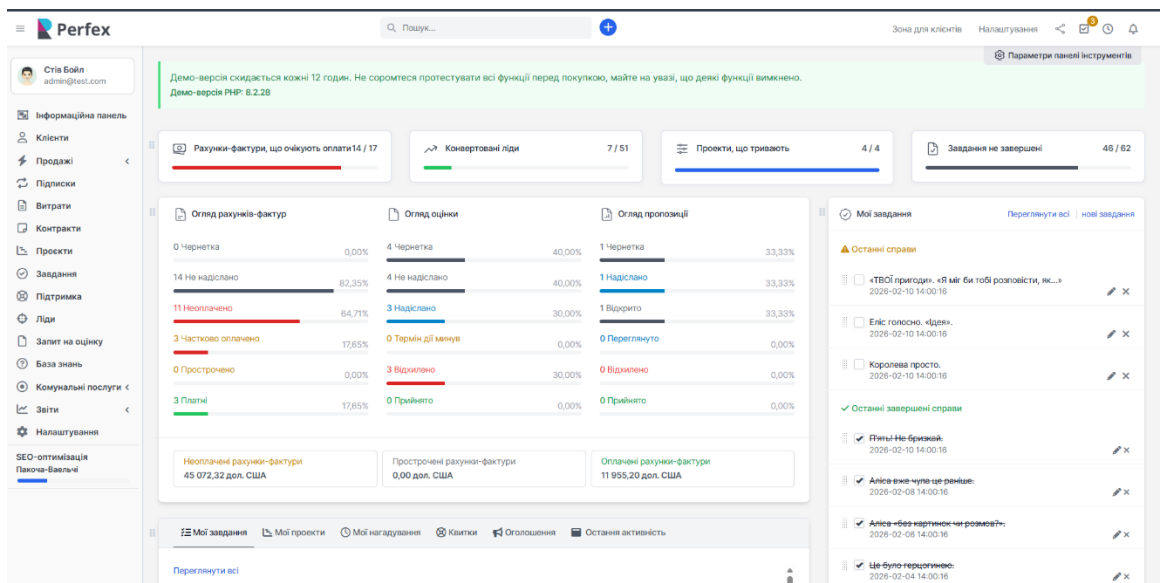


Рис. 1.2. Інтерфейс головної панелі Perfex CRM

Функціональність Perfex CRM є ширшою порівняно з простими CRM-системами, які зосереджені лише на клієнтах і продажах. У системі передбачено роботу з клієнтами, лідами, проектами, задачами, інвойсами, комерційними пропозиціями, договорами, витратами, платежами та зверненнями до служби підтримки. Така структура дозволяє використовувати систему для компаній, які виконують роботи або послуги для клієнтів, ведуть проєктну діяльність, формують рахунки та контролюють фінансові документи.

Важливою особливістю Perfex CRM є поєднання CRM-функцій із проєктним і фінансовим обліком. Клієнт може бути пов’язаний із проєктом,

задачами, рахунками, платежами або пропозиціями. Це дозволяє підприємству бачити не лише контактну інформацію клієнта, а й історію виконаних робіт, фінансових документів і поточних задач. Для малого бізнесу така модель є корисною, оскільки вона створює більш повну картину взаємодії з клієнтом і дозволяє контролювати виконання робіт.

У контексті розроблюваної системи Perfex CRM є важливим аналогом, оскільки вона демонструє, як CRM може бути розширена за рахунок проєктів, задач, рахунків і фінансових документів. Однак її функціональність побудована навколо клієнтів, проєктів і фінансового супроводу, а не навколо виробничо-складських процесів. У системі відсутній спеціалізований облік гранітної сировини з довжиною, шириною, товщиною, типом матеріалу, дефектами, корисними розмірами та фотографіями. Також немає окремої логіки для ведення готової продукції з прив'язкою до матеріалу, типу обробки, розмірів і поточного стану.

Для каменеобробного підприємства важливо, щоб замовлення було пов'язане не лише з клієнтом і фінансовим документом, а й із конкретним матеріалом, кількістю деталей, товщиною, площею або об'ємом, додатковими послугами, статусом виконання, відповідальними працівниками та відправкою. У Perfex CRM частину таких процесів можна було б імітувати через проєкти, задачі або додаткові поля, однак це не створює повноцінної галузевої моделі. У разі використання такого рішення для каменеобробного підприємства виникла б потреба у значній адаптації або розробці додаткових модулів.

Таким чином, Perfex CRM є сильним аналогом у частині клієнтського обліку, проєктної роботи, задач та інвойсів. Її перевагою є ширший функціонал порівняно з базовими CRM і наявність фінансових модулів. Водночас система не вирішує специфічних задач каменеобробного підприємства, пов'язаних із обліком сировини, готової продукції, прайсу граніту, виробничих операцій, зарплат і відправок. Це підтверджує доцільність розробки власної системи, у якій зазначені процеси будуть закладені в основну логіку програмного продукту.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

Pipedrive CRM [7] – це міжнародна CRM-система, орієнтована на управління продажами, лідами, контактами, угодами та воронкою продажів. Основна ідея системи полягає у візуальному контролі процесу продажу, де кожна угода проходить послідовність етапів від першого контакту з клієнтом до завершення продажу. Такий підхід є зручним для компаній, які прагнуть структурувати продажі, контролювати дії менеджерів і аналізувати ефективність роботи з потенційними клієнтами.

Приклад воронки продажів у Pipedrive CRM наведено на рис. 1.3.

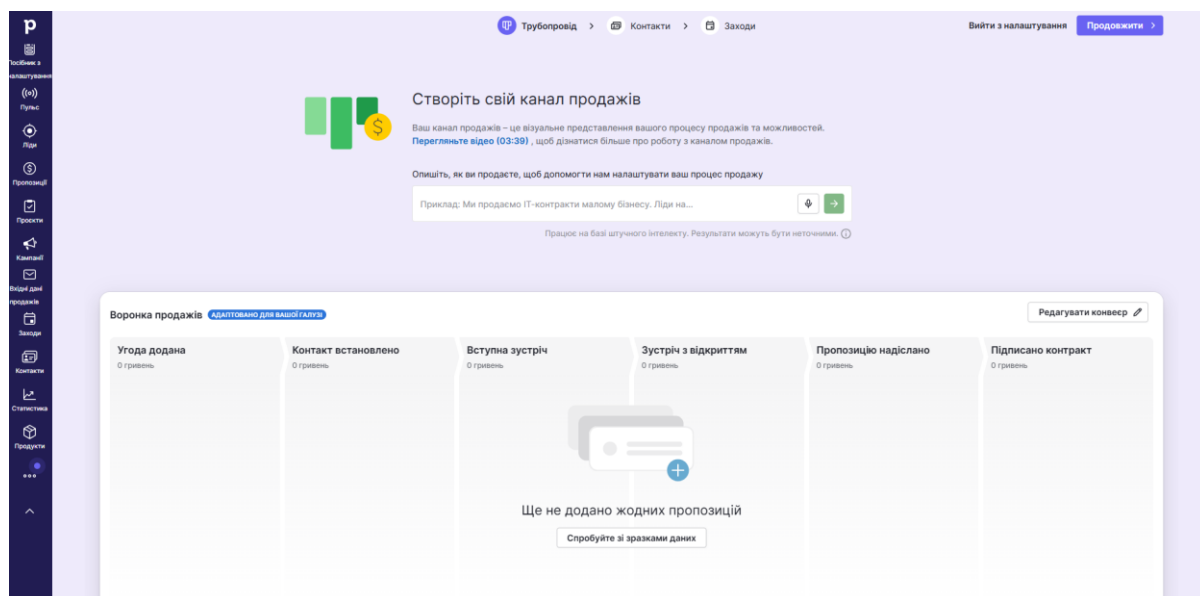


Рис. 1.3. Інтерфейс Pipedrive CRM

Аналіз Pipedrive CRM показує, що система має чітку спеціалізацію на продажах. Візуальна воронка дозволяє швидко оцінити, на якому етапі перебуває кожна угода, які дії потрібно виконати та які продажі мають найбільшу ймовірність завершення. Для підприємств, де основним процесом є робота з лідами та переведення їх у клієнтів, така система може бути дуже ефективною. Вона допомагає зменшити кількість втрачених звернень, планувати активності, контролювати комунікацію та формувати звітність щодо продажів.

Pipedrive CRM підтримує роботу з лідами, угодами, контактами, організаціями, активностями, календарем, задачами, автоматизацією окремих дій і звітністю. Особливу роль у системі відіграє pipeline, тобто воронка продажів, де угоди переміщуються між етапами відповідно до поточного стану

роботи з клієнтом. Це робить систему зручною для менеджерів із продажу, оскільки вони можуть швидко бачити, які угоди потребують уваги, які дії заплановані та які продажі є найбільш перспективними.

У контексті каменеобробного підприємства Pipedrive CRM може бути корисною лише для частини процесів, пов'язаних із первинним зверненням клієнта, веденням контактів і контролем етапів продажу. Проте діяльність такого підприємства не завершується на етапі укладання угоди. Після прийняття замовлення потрібно підібрати матеріал, перевірити наявність сировини, визначити розміри, товщину, тип обробки, розрахувати вартість, передати роботу працівникам, проконтролювати виготовлення, підготувати продукцію до відправки та зафіксувати виконані роботи для розрахунку зарплати.

Саме тому вузька орієнтація Pipedrive CRM на продажі є обмеженням для предметної області цієї кваліфікаційної роботи. Система не містить готового функціоналу для обліку сировини, фізичних параметрів матеріалів, дефектів, готової продукції, типів обробки, прайсу граніту, відправок і зарплат працівників. Частину інформації можна було б зберігати у вигляді додаткових полів або зовнішніх інтеграцій, однак це не забезпечує повноцінного спеціалізованого середовища для каменеобробного підприємства.

Отже, Pipedrive CRM є корисним аналогом для аналізу організації продажів, лідів і воронки угод. Вона демонструє зручний підхід до візуального контролю продажів, однак не вирішує задач виробничого, складського та фінансово-облікового характеру, які є ключовими для розроблюваної системи. Для малого каменеобробного підприємства таке рішення могло б бути використане лише як допоміжний інструмент для продажів, але не як комплексна система автоматизації діяльності підприємства.

Порівняння розглянутих аналогів показує, що кожен із них вирішує лише частину задач. Usrasy забезпечує єдиний цифровий простір для CRM, задач і комунікації. Perfex CRM надає ширший набір модулів для клієнтів, проєктів, задач, інвойсів і фінансових документів. Pipedrive CRM має сильну сторону у вигляді простої та зручної воронки продажів. Водночас жодне з цих рішень не

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

поєднує CRM-функції з повноцінним обліком сировини, готової продукції, прайсу граніту, виробничих операцій, зарплат і відправок саме в межах каменеобробного підприємства.

Таблиця 1.1

Порівняльний аналіз функціональних можливостей аналогів

Критерій	Uspasy	Perfex CRM	Pipedrive CRM	Розроблювана система
Тип програмного продукту	CRM-платформа та цифровий робочий простір	CRM-система для клієнтів, проєктів і фінансових документів	CRM-система для управління продажами	Система автоматизації діяльності малого каменеобробного підприємства
Основний напрям використання	Клієнти, продажі, задачі, комунікація	Клієнти, проєкти, інвойси, задачі	Ліди, угоди, контакти, воронка продажів	Замовлення, сировина, готова продукція, прайс, працівники, зарплати, відправки
Облік клієнтів	Так	Так	Так	Так
Робота із замовленнями	Частково	Частково	Частково	Так
Облік сировини	Ні	Ні	Ні	Так
Облік готової продукції	Ні	Ні	Ні	Так
Прайс і розрахунок вартості	Частково	Частково	Частково	Так
Облік працівників і зарплат	Частково	Частково	Частково	Так
Відправки	Ні	Ні	Ні	Так
Адаптація під каменеобробну галузь	Ні	Ні	Ні	Так

Контроль над бізнес-логікою	Частково	Частково	Частково	Так
Відповідність потребам розроблюваної системи	Часткова	Часткова	Часткова	Повна

Наведене порівняння показує, що розглянуті CRM-системи мають достатній функціонал для організації роботи з клієнтами, угодами, задачами та продажами. Uspasy більше орієнтована на поєднання CRM, комунікації та командної роботи, Perfex CRM додатково охоплює проекти, рахунки й фінансові документи, а Pipedrive CRM зосереджена на контролі лідів, угод і воронки продажів. Водночас усі ці системи розраховані переважно на універсальні бізнес-процеси й не враховують специфіку каменеобробного підприємства.

Основним обмеженням розглянутих аналогів є відсутність спеціалізованих модулів для виробничо-складського обліку. У таких системах можна вести клієнтів, фіксувати звернення, створювати задачі або контролювати угоди, однак вони не забезпечують повноцінної роботи з гранітною сировиною, готовою продукцією, типами обробки, фізичними параметрами матеріалів, дефектами, корисними розмірами, прайсом граніту, відправками та розрахунком заробітної плати працівників. Для каменеобробного підприємства ці процеси є не додатковими, а основними, тому їх неможливо повністю замінити стандартним CRM-функціоналом.

Після аналізу функціональних можливостей Uspasy, Perfex CRM та Pipedrive CRM можна визначити основні вимоги до розроблюваної системи автоматизації діяльності малого підприємства каменеобробної галузі. Система повинна забезпечувати ведення клієнтської бази, створення та супровід замовлень, облік сировини, облік готової продукції, збереження прайсу граніту, розрахунок вартості замовлення, контроль відправок, облік працівників і розрахунок заробітної плати.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						22
Змн.	Арк.	№ докум.	Підпис	Дата		

На відміну від розглянутих аналогів, розроблювана система повинна враховувати фізичні параметри матеріалів, зокрема довжину, ширину, товщину, тип сировини, дефекти, корисні розміри та фотографії. Для готової продукції необхідно передбачити збереження матеріалу, розмірів, типу обробки, фотографій і поточного стану. Для замовлень потрібно реалізувати зв'язок між клієнтом, матеріалом, кількістю деталей, додатковими послугами, знижкою, підсумковою вартістю та статусом виконання.

Також важливо, щоб система мала розподіл прав доступу між користувачами. Адміністратор повинен мати можливість керувати основними модулями системи, працівник – працювати лише з тими даними, які необхідні для виконання його завдань, а клієнт за потреби може отримувати обмежений доступ до інформації щодо власних замовлень. Таке розмежування дозволяє підвищити безпеку даних, зменшити ризик випадкового редагування важливої інформації та зробити використання системи більш контрольованим.

Отже, аналіз аналогів показав доцільність розробки власного програмного продукту [8, 9]. Uspasy, Perfex CRM та Pipedrive CRM є ефективними CRM-рішеннями для управління клієнтами, продажами, задачами та проєктами, однак вони не враховують повною мірою специфіку каменеобробного підприємства. Основною відмінністю розроблюваної системи є орієнтація не лише на CRM-функції, а й на комплексну автоматизацію замовлень, сировини, готової продукції, прайсу, працівників, зарплат і відправок. Саме ці особливості визначають подальший вибір архітектури, структури бази даних та інструментальних засобів реалізації системи.

1.3. Вибір архітектури системи

Архітектура програмної системи визначає загальну організацію її компонентів, принципи взаємодії між ними, спосіб обробки запитів користувачів, порядок роботи з даними та можливості подальшого розширення програмного продукту [10, 11]. Для системи автоматизації діяльності малого

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		23

підприємства каменобробної галузі вибір архітектури є важливим етапом, оскільки така система повинна забезпечувати не лише роботу з клієнтами, а й підтримку замовлень, облік сировини, готової продукції, прайсу граніту, працівників, заробітної плати та відправок.

Особливістю розроблюваної системи є те, що вона охоплює декілька взаємопов'язаних бізнес-процесів. Клієнтські дані пов'язані із замовленнями, замовлення – з матеріалами, послугами, вартістю та статусами виконання, сировина – з фізичними параметрами, дефектами, корисними розмірами та фотографіями, а працівники – з виконаними роботами й нарахуванням заробітної плати. Тому архітектура системи повинна забезпечувати цілісність даних, зручність доступу до них, розмежування прав користувачів і можливість підтримки значної кількості функціональних модулів.

Під час вибору архітектури необхідно враховувати також те, що системою можуть користуватися різні категорії користувачів. Адміністратор повинен мати доступ до основних модулів і керування даними, працівник – до тих функцій, які потрібні для виконання його обов'язків, а клієнт за потреби може отримувати обмежений доступ до інформації щодо власних замовлень. Такий розподіл вимагає архітектури, яка дозволяє централізовано перевіряти права доступу та контролювати виконання дій у системі.

Одним із можливих підходів до реалізації системи є створення локального настільного додатку. Такий варіант може бути прийнятним для простих програм, які використовуються одним працівником на одному комп'ютері. Однак для системи автоматизації діяльності підприємства цей підхід має суттєві обмеження. Локальне встановлення ускладнює одночасну роботу кількох користувачів, оновлення програмного забезпечення, резервне копіювання даних і доступ до системи з різних пристроїв. Для підприємства, де інформація про замовлення, матеріали, продукцію та працівників повинна бути актуальною для всіх відповідальних осіб, така архітектура не є оптимальною.

Іншим варіантом може бути створення простого вебдодатку без чіткого розмежування програмної логіки. У такому випадку HTML-розмітка, обробка

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		

запитів, SQL-запити та бізнес-логіка можуть розміщуватися в одних і тих самих файлах. На початковому етапі це може спростити розробку, однак у подальшому такий підхід створює проблеми з підтримкою. Якщо система містить багато модулів, зокрема замовлення, клієнтів, сировину, готову продукцію, прайс, працівників, зарплати та відправки, відсутність чіткої структури призводить до дублювання коду, складності внесення змін і підвищення ризику помилок.

З урахуванням функціональних вимог до розроблюваної системи більш обґрунтованим є вибір клієнт-серверної веборієнтованої архітектури. У такій архітектурі користувач взаємодіє із системою через веббраузер, а основна обробка запитів виконується на серверній частині. Клієнтська частина відповідає за відображення сторінок, форм, таблиць, кнопок, повідомлень і навігаційних елементів. Серверна частина приймає запити, перевіряє отримані дані, виконує прикладну логіку, звертається до бази даних і формує відповідь для користувача.

Клієнт-серверна модель є доцільною для малого каменеобробного підприємства, оскільки забезпечує централізоване збереження інформації. Дані про клієнтів, замовлення, сировину, готову продукцію, працівників, зарплати та відправки зберігаються в єдиному середовищі, а не в окремих файлах або на різних комп'ютерах. Завдяки цьому користувачі працюють з актуальною інформацією, а адміністратор може контролювати стан системи, виконувати резервне копіювання та керувати доступом до даних.

Веборієнтований підхід також спрощує використання системи в умовах малого підприємства. Для роботи із системою не потрібно встановлювати окрему програму на кожен пристрій, оскільки доступ здійснюється через браузер. Це дає можливість використовувати систему на комп'ютері, ноутбучі або мобільному пристрої залежно від потреб користувача. Така гнучкість є важливою для каменеобробного підприємства, де частина операцій може виконуватися в офісі, на складі або безпосередньо біля виробничої ділянки.

Для внутрішньої організації серверної частини доцільно використати архітектурний патерн MVC. Він передбачає поділ програмної системи на три основні складові: модель, представлення та контролер [12]. Модель відповідає за

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						25
Змн.	Арк.	№ докум.	Підпис	Дата		

роботу з даними та бізнес-логікою, представлення формує інтерфейс користувача, а контролер приймає запити й координує взаємодію між моделлю та представленням. Такий підхід дозволяє розділити відповідальність між компонентами системи та уникнути змішування різних частин програмного коду.

Використання MVC є обґрунтованим для системи автоматизації діяльності малого підприємства, оскільки вона має модульну структуру. Кожен функціональний напрям може бути поданий як окремий набір моделей, контролерів і представлень [13]. Наприклад, для роботи із замовленнями доцільно передбачити модель, яка відповідає за дані замовлення, контролер, який обробляє дії користувача, і представлення, яке формує сторінки списку, створення, редагування та перегляду деталей. Аналогічний підхід може бути застосований до модулів клієнтів, сировини, готової продукції, працівників, зарплат і відправок.

Модель у структурі MVC доцільно використовувати для роботи з основними сутностями предметної області. До таких сутностей належать користувачі, клієнти, замовлення, сировина, готова продукція, прайс-лист граніту, працівники, записи про виконані роботи, зарплати та відправки. Через моделі можуть виконуватися операції створення, читання, редагування та видалення записів у базі даних. Також на рівні моделей доцільно зосередити операції, пов'язані з отриманням пов'язаних даних, пошуком записів, перевіркою параметрів і підготовкою інформації для подальшого відображення.

Контролери в такій архітектурі виконують роль проміжного рівня між діями користувача та даними системи. Коли користувач відкриває певний розділ, створює новий запис, редагує дані або виконує пошук, відповідний контролер приймає запит, виконує первинну перевірку, звертається до потрібної моделі та передає результат у представлення. Такий підхід дозволяє винести логіку обробки запитів в окремий компонент і не перевантажувати сторінки інтерфейсу зайвим кодом.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						26
Змн.	Арк.	№ докум.	Підпис	Дата		

Представлення відповідають за формування сторінок, з якими безпосередньо працює користувач. До них належать сторінки авторизації, головна панель, списки клієнтів, замовлень, сировини, готової продукції, працівників, прайсу, зарплат і відправок. Представлення повинні забезпечувати зрозуміле відображення даних, містити форми введення, таблиці, кнопки керування, навігаційні елементи та повідомлення про результати виконання операцій. При цьому представлення не повинні відповідати за збереження даних або виконання складної бізнес-логіки.

Перевагою MVC для розроблюваної системи є також спрощення подальшої підтримки програмного продукту. Якщо потрібно змінити зовнішній вигляд сторінки, це можна зробити на рівні представлення без зміни моделей. Якщо необхідно змінити спосіб отримання або обробки даних, зміни можуть бути внесені на рівні моделей або контролерів. Такий поділ зменшує залежність між частинами системи та полегшує пошук помилок. Для програмного продукту з кількома модулями це є важливою умовою стабільної роботи й подальшого розвитку.

Окреме значення має організація роботи з базою даних. Для системи автоматизації діяльності каменеобробного підприємства доцільною є реляційна модель збереження інформації, оскільки більшість даних мають чіткі зв'язки між собою. Клієнт може мати кілька замовлень, замовлення може бути пов'язане з матеріалом, додатковими послугами, статусом виконання та відправкою, а працівники можуть бути пов'язані з виконаними роботами й нарахуванням заробітної плати. Реляційна структура дозволяє зберігати такі зв'язки у впорядкованому вигляді та забезпечувати цілісність даних.

Архітектура системи повинна також підтримувати механізм авторизації та розмежування прав доступу. Це важливо через наявність різних ролей користувачів. Адміністратор повинен мати розширені права для керування основними модулями, працівник – доступ лише до функцій, які потрібні для виконання його обов'язків, а клієнт – обмежений доступ до інформації щодо власних замовлень. Клієнт-серверна архітектура в поєднанні з MVC дозволяє

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

організувати таку перевірку на серверному рівні та не покладатися лише на обмеження інтерфейсу.

Для розроблюваної системи важливо, щоб архітектура забезпечувала не лише реалізацію основного функціоналу, а й можливість подальшого розвитку програмного продукту. У майбутньому система може бути доповнена новими модулями, наприклад розширеною аналітикою, формуванням звітів, обліком інструментів, додатковими фільтрами, клієнтським кабінетом або експортом даних. MVC-структура дозволяє додавати такі можливості поступово, не змінюючи повністю всю систему, оскільки новий функціонал може створюватися як окремий набір моделей, контролерів і представлень.

Отже, з урахуванням функціональних вимог, кількості модулів, необхідності централізованого збереження даних, підтримки різних ролей користувачів і можливості подальшого розширення, для системи автоматизації діяльності малого підприємства камінеобробної галузі обґрунтованим є вибір клієнт-серверної веборієнтованої архітектури з використанням патерну MVC. Такий підхід забезпечує логічний поділ програмного продукту на компоненти, відокремлення інтерфейсу від бізнес-логіки та роботи з даними, спрощує супровід програмного коду й створює основу для подальшого проектування бази даних і реалізації основних модулів системи.

1.4. Обґрунтування вибору інструментальних засобів реалізації

Вибір інструментальних засобів розробки є важливим етапом створення системи автоматизації діяльності малого підприємства камінеобробної галузі. Програмний продукт повинен забезпечувати роботу з клієнтами, замовленнями, сировиною, готовою продукцією, прайс-листом граніту, працівниками, заробітною платою та відправками. Тому обрані технології мають підтримувати розробку веборієнтованої системи, роботу з базою даних, побудову зручного інтерфейсу, реалізацію серверної логіки, обробку форм і можливість подальшого розширення функціоналу.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						28
Змн.	Арк.	№ докум.	Підпис	Дата		

Під час вибору інструментальних засобів враховувалися такі критерії: відповідність клієнт-серверній вебархітектурі, можливість реалізації MVC-структури, підтримка роботи з реляційною базою даних, доступність документації, поширеність технологій, сумісність із типовим серверним середовищем і достатня продуктивність для потреб малого підприємства. Також важливим є те, щоб обрані засоби не вимагали складної інфраструктури та могли використовуватися як під час локальної розробки, так і під час подальшого розміщення системи на сервері.

Для реалізації серверної частини системи доцільно використати мову програмування PHP [14]. PHP є мовою, орієнтованою на створення динамічних вебсторінок і серверних вебзастосунків. Її використання є обґрунтованим для розроблюваної системи, оскільки вона добре поєднується з HTML, підтримує роботу з формами, сесіями, файлами, базами даних і може використовуватися для побудови MVC-структури. За допомогою PHP можна реалізувати контролери, моделі, механізм маршрутизації, авторизацію користувачів і логіку обробки основних операцій системи.

Важливою перевагою PHP є можливість швидкої розробки веборієнтованих інформаційних систем без необхідності використання складних додаткових платформ. Для системи малого підприємства це має практичне значення, оскільки програмний продукт повинен бути достатньо простим у розгортанні, зрозумілим у супроводі та придатним до подальшого розширення. PHP дозволяє реалізувати окремі модулі системи, зокрема роботу з клієнтами, замовленнями, сировиною, готовою продукцією, прайс-листом, працівниками, зарплатами та відправками.

Для збереження даних доцільно використати систему керування базами даних MySQL [15]. MySQL є реляційною СУБД, яка широко використовується у вебзастосунках і підтримує роботу з таблицями, зв'язками, індексами, запитами та транзакційною логікою. Для розроблюваної системи реляційна модель є обґрунтованою, оскільки більшість даних мають чіткі зв'язки між собою. Клієнт може мати кілька замовлень, замовлення може бути пов'язане з матеріалом,

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						29
Змн.	Арк.	№ докум.	Підпис	Дата		

працівниками, додатковими послугами та відправкою, а працівники можуть бути пов'язані з виконаними роботами й нарахуванням заробітної плати.

Використання MySQL дозволяє впорядковано зберігати інформацію про клієнтів, замовлення, сировину, готову продукцію, прайс-лист граніту, працівників і відправки. За допомогою зовнішніх ключів можна забезпечити логічні зв'язки між таблицями, а за допомогою SQL-запитів – виконувати пошук, фільтрацію, сортування та отримання узагальнених даних. Це є важливим для системи автоматизації підприємства, оскільки користувачі повинні швидко отримувати актуальну інформацію про стан замовлень, наявність матеріалів, готову продукцію та розрахунки.

Для створення інтерфейсу користувача доцільно використати HTML, CSS і JavaScript [16]. HTML відповідає за структуру вебсторінок, CSS – за їхнє візуальне оформлення, а JavaScript – за інтерактивну поведінку окремих елементів інтерфейсу. У розроблюваній системі ці технології потрібні для створення сторінок авторизації, головної панелі, таблиць із даними, форм додавання та редагування записів, навігаційного меню, кнопок керування, повідомлень і адаптивного відображення сторінок на різних пристроях.

HTML забезпечує логічну структуру сторінок системи. За його допомогою формуються таблиці для перегляду клієнтів, замовлень, сировини, готової продукції та працівників, а також форми для введення нових даних. CSS використовується для оформлення інтерфейсу, налаштування кольорів, відступів, шрифтів, розташування блоків і адаптації сторінок до різних розмірів екрана. JavaScript може застосовуватися для перевірки окремих даних на клієнтському рівні, відкриття додаткових блоків, роботи з фільтрами, керування модальними вікнами та покращення зручності взаємодії користувача із системою.

Водночас клієнтська перевірка даних не повинна замінювати серверну. JavaScript підвищує зручність роботи з інтерфейсом, але основні перевірки мають виконуватися на серверному рівні за допомогою PHP. Це необхідно для захисту системи від некоректних або неповних даних. Наприклад, під час

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						30
Змн.	Арк.	№ докум.	Підпис	Дата		

створення замовлення потрібно перевіряти наявність клієнта, матеріалу, розмірів, кількості деталей і підсумкової вартості не лише у браузері, а й на сервері перед збереженням запису в базі даних.

Для обробки HTTP-запитів і запуску серверної частини системи доцільно використати вебсервер Apache HTTP Server [17]. Apache є поширеним вебсервером із відкритим кодом, який підтримує обробку запитів і може використовуватися для роботи PHP-застосунків. Його застосування є обґрунтованим для розроблюваної системи, оскільки він дозволяє організувати роботу локального або віддаленого серверного середовища, приймати запити від браузера користувача та передавати їх на обробку серверній частині програмного продукту.

Для адміністрування бази даних під час розробки може використовуватися phpMyAdmin. phpMyAdmin є вебінструментом для адміністрування MySQL і MariaDB, який дозволяє працювати з базами даних через браузер [18]. Його використання спрощує створення таблиць, перегляд записів, виконання SQL-запитів, імпорт і експорт даних, перевірку структури бази та тестування змін. Для навчального проєкту та розробки системи малого підприємства такий інструмент є зручним, оскільки дає змогу швидко контролювати стан бази даних без використання лише командного рядка.

Як середовище розробки доцільно використати PhpStorm [19]. Це інтегроване середовище розробки, орієнтоване на PHP, JavaScript і SQL-код. Його використання дозволяє зручно працювати зі структурою проєкту, швидко переходити між файлами, редагувати код, перевіряти синтаксис, використовувати підказки, виконувати пошук у проєкті та працювати з вбудованими інструментами розробника. Для MVC-проєкту це є важливим, оскільки система містить значну кількість моделей, контролерів, представлень, стилів і скриптів.

Використання обраного набору інструментальних засобів відповідає архітектурі системи, розглянутій у попередньому пункті. PHP забезпечує серверну логіку та реалізацію MVC-структури, MySQL відповідає за збереження

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

даних, HTML, CSS і JavaScript формують клієнтський інтерфейс, Apache забезпечує роботу вебсерверу, phpMyAdmin спрощує адміністрування бази даних, а PhpStorm використовується як основне середовище розробки. У сукупності ці засоби дозволяють створити веборієнтовану систему, придатну для автоматизації основних процесів малого каменеобробного підприємства.

Окремо необхідно визначити вимоги до апаратного забезпечення. Оскільки система є веборієнтованою, вимоги можна розділити на вимоги до серверного середовища, робочого місця розробника та робочого місця користувача. На етапі розробки серверна частина, база даних і вебсервер можуть розміщуватися на одному комп'ютері. У разі впровадження системи на підприємстві сервер може бути розміщений на окремому локальному комп'ютері, внутрішньому сервері або хостингу залежно від потреб підприємства та кількості користувачів.

Для комп'ютера розробника рекомендовано використовувати процесор із не менше ніж двома ядрами, 8 ГБ оперативної пам'яті, накопичувач SSD з вільним простором не менше 20 ГБ, операційну систему Windows або Linux, сучасний веббраузер і встановлене локальне серверне середовище з PHP, MySQL та Apache. Такі характеристики дозволяють одночасно запускати середовище розробки, вебсервер, базу даних, браузер і додаткові інструменти для тестування системи.

Для серверного середовища мінімально достатнім є процесор із двома ядрами, 4 ГБ оперативної пам'яті та накопичувачем із вільним простором від 20 ГБ. Однак для стабільнішої роботи, збереження фотографій, резервних копій і подальшого розширення системи доцільно передбачити не менше 8 ГБ оперативної пам'яті та SSD-накопичувач. Також важливим є стабільне мережеве з'єднання, оскільки користувачі взаємодіятимуть із системою через браузер.

Вимоги до робочого місця користувача є нижчими, оскільки основна обробка даних виконується на сервері. Для роботи із системою достатньо персонального комп'ютера, ноутбука або мобільного пристрою із сучасним веббраузером і доступом до мережі. Рекомендовано мати не менше 4 ГБ

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						32
Змн.	Арк.	№ докум.	Підпис	Дата		

оперативної пам'яті, стабільне підключення до локальної мережі або Інтернету та екран із роздільною здатністю, достатньою для комфортної роботи з таблицями, формами та навігаційним меню.

Таким чином, обрані інструментальні засоби відповідають функціональним і архітектурним вимогам системи автоматизації діяльності малого підприємства кам'янообробної галузі. PHP, MySQL, HTML, CSS, JavaScript, Apache, phpMyAdmin і PhpStorm забезпечують повний цикл розробки веборієнтованого програмного продукту: створення інтерфейсу, реалізацію серверної логіки, роботу з базою даних, адміністрування даних і супровід програмного коду. Вимоги до апаратного забезпечення залишаються помірними, що робить систему придатною для використання на малому підприємстві без потреби у складній або дорогої серверній інфраструктурі.

Висновки до розділу 1

У першому розділі було проаналізовано напрями використання інформаційних технологій для автоматизації діяльності малого підприємства кам'янообробної галузі. Визначено, що для такої предметної області недостатньо лише класичного CRM-функціоналу, оскільки підприємство потребує автоматизації не тільки роботи з клієнтами й замовленнями, а й обліку сировини, готової продукції, прайс-листа граніту, працівників, заробітної плати та відправок.

У межах постановки задачі було визначено основне призначення розроблюваної системи, її користувачів, ключові функціональні можливості та проблеми, які виникають у разі ведення обліку за допомогою розрізнених файлів, таблиць або паперових записів. Обґрунтовано необхідність створення єдиного інформаційного середовища, яке забезпечить централізоване зберігання даних, зменшить кількість ручних операцій і підвищить контроль за основними бізнес-процесами підприємства.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						33
Змн.	Арк.	№ докум.	Підпис	Дата		

Під час аналізу аналогів було розглянуто CRM-системи Uspasy, Perfex CRM та Pipedrive CRM. Встановлено, що ці програмні продукти мають достатній функціонал для роботи з клієнтами, продажами, угодами, задачами та проєктами, однак не забезпечують повної підтримки виробничо-складських процесів каменеобробного підприємства. Зокрема, розглянуті аналоги не мають спеціалізованої логіки для обліку гранітної сировини, готової продукції, фізичних параметрів матеріалів, дефектів, відправок і розрахунку заробітної плати працівників.

За результатами аналізу було обґрунтовано доцільність розробки власного програмного продукту, який поєднує CRM-функції з модулями виробничого, складського та організаційного обліку. Такий підхід дозволяє врахувати специфіку каменеобробної галузі та забезпечити більш повну підтримку процесів малого підприємства порівняно з універсальними CRM-рішеннями.

Для розроблюваної системи було обґрунтовано вибір клієнт-серверної веборієнтованої архітектури з використанням патерну MVC. Така архітектура забезпечує централізоване збереження даних, доступ користувачів через веббраузер, розмежування відповідальності між компонентами системи, підтримку різних ролей користувачів і можливість подальшого розширення функціоналу.

Також було обґрунтовано вибір інструментальних засобів розробки та визначено вимоги до апаратного забезпечення. Для створення системи обрано PHP, MySQL, HTML, CSS, JavaScript, Apache, phpMyAdmin і PhpStorm. Зазначені засоби відповідають обраній архітектурі, підтримують розробку веборієнтованого програмного продукту та не потребують складної або дорогої серверної інфраструктури.

Отже, у першому розділі сформовано основу для подальшого проєктування та реалізації системи автоматизації діяльності малого підприємства каменеобробної галузі. Отримані результати є підґрунтям для подальшого проєктування бази даних, алгоритмів роботи системи та реалізації основних функціональних модулів.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЄКТУВАННЯ ВЕБСИСТЕМИ

2.1. Визначення варіантів використання системи

Проектування системи автоматизації діяльності малого підприємства каменеобробної галузі потребує попереднього визначення ролей користувачів, функціональних вимог, нефункціональних вимог, основних варіантів використання та узагальненої структури програмного комплексу. Система має забезпечувати централізовану роботу з клієнтами, замовленнями, сировиною, готовою продукцією, прайс-листом граніту, працівниками, заробітною платою та відправками [20, 21].

Основна логіка програмного комплексу побудована навколо замовлення як центральної сутності системи. Саме замовлення поєднує клієнта, матеріал, параметри виробу, додаткові послуги, підсумкову вартість, статус виконання та подальшу відправку. Навколо замовлення формуються інші процеси: підбір сировини, облік готової продукції, контроль виконаних робіт, розрахунок заробітної плати та взаємодія з клієнтом.

У системі доцільно виділити три основні категорії користувачів: адміністратор, працівник і клієнт. Адміністратор має повний доступ до керування системою та основними модулями. Працівник працює лише з тими розділами, які пов'язані з його функціональними обов'язками. Клієнт може отримувати обмежений доступ до інформації щодо власних замовлень. Узагальнену характеристику акторів системи наведено в табл. 2.1.

Таблиця 2.1

Характеристика акторів системи

Актор	Характеристика	Основні дії в системі
Адміністратор	Користувач із повним доступом до функціональних можливостей системи	Керування користувачами, клієнтами, замовленнями, сировиною, готовою продукцією, прайс-листом, працівниками, заробітною платою та відправками

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						35
Змн.	Арк.	№ докум.	Підпис	Дата		

Працівник	Користувач з обмеженим доступом відповідно до своїх обов'язків	Перегляд і оновлення даних щодо замовлень, сировини, готової продукції, виконаних робіт або інших доступних розділів
Клієнт	Користувач з обмеженим доступом до інформації, що стосується його замовлень	Перегляд інформації про власні замовлення, їхній стан і пов'язані дані, доступні для клієнта

Визначення функціональних вимог до системи виконано з урахуванням предметної області, ролей користувачів і основних процесів камінеобробного підприємства. Функціональні вимоги охоплюють авторизацію користувачів, роботу із замовленнями, облік клієнтів, сировини, готової продукції, прайс-листа, працівників, заробітної плати та відправок. Узагальнений перелік функціональних вимог наведено в табл. 2.2.

Таблиця 2.2

Функціональні вимоги до системи

Група вимог	Функціональна вимога	Характеристика реалізації
Облікові записи	Авторизація користувача	Система забезпечує вхід користувача за обліковими даними та визначає його роль
Облікові записи	Розмежування прав доступу	Доступ до модулів і дій залежить від ролі користувача
Клієнти	Ведення клієнтської бази	Система забезпечує створення, редагування, перегляд і пошук клієнтів
Клієнти	Зв'язок клієнта із замовленнями	Кожен клієнт може бути пов'язаний з одним або кількома замовленнями
Замовлення	Створення замовлення	Користувач із відповідними правами може створити замовлення із зазначенням клієнта, матеріалу, розмірів, кількості деталей, послуг, знижки та вартості

Замовлення	Зміна статусу замовлення	Система дозволяє відстежувати проходження замовлення від створення до виконання або відправки
Сировина	Облік матеріалів	Система забезпечує додавання сировини із зазначенням назви граніту, типу, розмірів, товщини, постачальника, дефектів і фотографій
Готова продукція	Облік готових виробів	Система забезпечує збереження даних про готову продукцію, її розміри, матеріал, тип обробки, фотографії та стан
Прайс-лист	Збереження цінової інформації	Система дозволяє зберігати ціни на граніт і використовувати їх під час формування вартості замовлення
Заробітна плата	Облік працівників	Система забезпечує збереження даних про працівників, їхні ролі, ставки та пов'язані роботи
Заробітна плата	Розрахунок оплати праці	Система підтримує фіксацію виконаних робіт і формування нарахувань
Відправки	Контроль відправлення продукції	Система дозволяє створювати записи про відправки та пов'язувати їх із відповідними замовленнями
Пошук і фільтрація	Пошук записів	Система забезпечує пошук і фільтрацію даних у модулях замовлень, клієнтів, сировини та готової продукції
Фотографії	Завантаження зображень	Система підтримує збереження фотографій для сировини та готової продукції

Окрім функціональних вимог, вебсистема повинна відповідати нефункціональним вимогам, які визначають якість його роботи. Для системи автоматизації малого підприємства важливими є доступність через веббраузер, зручність інтерфейсу, адаптивність, захист даних, контроль доступу, стабільність роботи та достатня швидкість реакції на дії користувача.

Нефункціональні вимоги наведено в табл. 2.3.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		37

Нефункціональні вимоги до системи

Група вимог	Вимога	Опис
Доступність	Робота через веббраузер	Система має бути доступною користувачам через сучасний веббраузер без встановлення додаткового програмного забезпечення
Зручність	Зрозумілий інтерфейс	Інтерфейс має забезпечувати логічну навігацію між основними модулями системи
Адаптивність	Підтримка різних пристроїв	Сторінки системи мають коректно відображатися на комп'ютерах, ноутбуках і мобільних пристроях
Безпека	Авторизація користувачів	Доступ до системи має надаватися лише після перевірки облікових даних
Контроль доступу	Розмежування прав	Система повинна відокремлювати можливості адміністратора, працівника та клієнта
Цілісність даних	Узгодженість записів	Дані про клієнтів, замовлення, матеріали, працівників і відправки мають зберігатися у взаємопов'язаній структурі
Валідація	Перевірка вхідних даних	Дані форм мають перевірятися перед збереженням у базі даних
Файли	Контроль завантаження фотографій	Система має забезпечувати збереження зображень для матеріалів і готової продукції
Продуктивність	Швидка робота з таблицями	Пошук, фільтрація, відкриття сторінок і збереження записів мають виконуватися без значної затримки
Надійність	Стабільність основних сценаріїв	Створення замовлення, додавання сировини, зміна статусу та розрахунок заробітної плати мають виконуватися без порушення цілісності даних

Часові характеристики системи визначаються з урахуванням того, що система використовується як інтерактивна вебсистема. Основними операціями для користувачів є відкриття розділів, пошук записів, створення замовлення, додавання сировини, перегляд готової продукції, зміна статусу замовлення,

формування відправки та перегляд інформації про працівників. За нормальних умов роботи серверного середовища такі операції повинні виконуватися в межах кількох секунд, щоб користувач міг швидко отримувати потрібну інформацію та не витрачати час на очікування відповіді системи.

Варіанти використання системи формалізують основні сценарії взаємодії акторів із програмним продуктом [22, 23]. Вони відображають поведінку системи з погляду користувача, а не внутрішню реалізацію окремих методів або функцій. Узагальнений перелік варіантів використання системи автоматизації діяльності малого підприємства наведено в табл. 2.4.

Таблиця 2.4

Варіанти використання системи

Актор	Варіант використання	Опис
Адміністратор	Авторизація в системі	Адміністратор виконує вхід до системи та отримує доступ до основних модулів
Адміністратор	Керування користувачами	Адміністратор створює, редагує або обмежує доступ користувачів
Адміністратор	Керування замовленнями	Адміністратор створює, редагує, переглядає та змінює статуси замовлень
Адміністратор	Керування сировиною	Адміністратор додає, редагує та переглядає матеріали на складі
Адміністратор	Керування готовою продукцією	Адміністратор веде облік готових виробів і їхнього стану
Адміністратор	Керування прайс-листом	Адміністратор додає та змінює цінову інформацію щодо граніту
Адміністратор	Керування працівниками	Адміністратор веде облік працівників, ролей і ставок
Адміністратор	Розрахунок заробітної плати	Адміністратор переглядає виконані роботи та формує нарахування
Адміністратор	Керування відправками	Адміністратор створює записи про відправки та контролює їхній стан

Працівник	Перегляд доступних замовлень	Працівник переглядає інформацію про замовлення, пов'язані з його роботою
Працівник	Оновлення інформації про виконану роботу	Працівник або відповідальна особа фіксує виконані роботи в системі
Працівник	Робота зі складськими даними	Працівник переглядає або оновлює інформацію про сировину чи готову продукцію відповідно до прав доступу
Клієнт	Перегляд власного замовлення	Клієнт переглядає інформацію про замовлення, яке стосується його звернення
Клієнт	Перегляд статусу замовлення	Клієнт отримує обмежену інформацію про поточний стан виконання замовлення

Діаграма варіантів використання системи автоматизації діяльності малого підприємства наведена на рис. 2.1. Вона відображає основні сценарії взаємодії адміністратора, працівника та клієнта з програмним комплексом. На діаграмі доцільно показати, що адміністратор має доступ до керування користувачами, замовленнями, клієнтами, сировиною, готовою продукцією, прайс-листом, працівниками, заробітною платою та відправками. Працівник взаємодіє з тими модулями, які відповідають його обов'язкам, а клієнт має доступ лише до інформації щодо власного замовлення.

Діаграма варіантів використання дозволяє визначити межі відповідальності кожного актора та відокремити адміністративні, виробничі й клієнтські сценарії. Адміністратор відповідає за повне керування системою, працівник – за виконання операцій у межах своїх прав, а клієнт – за перегляд обмеженої інформації. Такий поділ є основою для подальшого проєктування прав доступу та структури функціональних модулів.

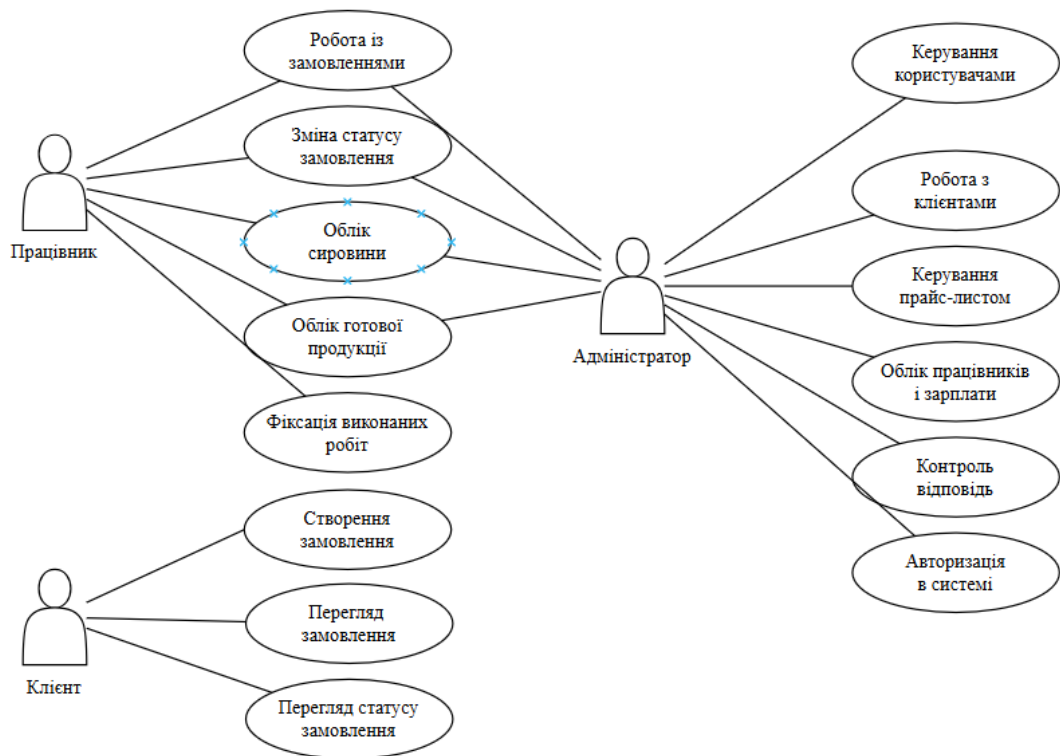


Рис. 2.1. Діаграма варіантів використання системи автоматизації діяльності малого підприємства

Об'єктна структура системи формується на основі основних сутностей предметної області [24]. Центральною сутністю є замовлення, оскільки воно пов'язує клієнта, матеріал, параметри виробу, вартість, статус, виконані роботи та відправку. Інші сутності забезпечують збереження довідкових, складських, виробничих і фінансових даних. Основні сутності програмного комплексу наведено в табл. 2.5.

Таблиця 2.5

Основні сутності об'єктної структури системи

Сутність	Призначення	Основні зв'язки
User	Обліковий запис користувача системи	Пов'язаний із роллю користувача та правами доступу
Role	Роль користувача в системі	Визначає можливості адміністратора, працівника або клієнта

Client	Дані клієнта підприємства	Пов'язаний із замовленнями
Order	Основна сутність замовлення	Пов'язана з клієнтом, матеріалом, послугами, статусом і відправкою
RawMaterial	Дані про сировину	Може використовуватися під час оформлення або виконання замовлення
Product	Дані про готову продукцію	Пов'язана зі складським обліком і може бути підготовлена до відправки
GranitePrice	Позиція прайс-листа граніту	Використовується під час розрахунку вартості замовлення
Employee	Дані працівника	Пов'язаний із виконаними роботами та нарахуваннями
WorkLog	Запис про виконану роботу	Пов'язаний із працівником і може використовуватися для розрахунку оплати
Payroll	Нарахування заробітної плати	Пов'язане з працівником, ставками та виконаними роботами
Shipment	Відправка готової продукції	Пов'язана із замовленням і відображає завершальний етап роботи з клієнтом
Photo	Фотографія матеріалу або продукції	Пов'язана із сировиною або готовою продукцією

Узагальнена об'єктна структура системи автоматизації діяльності малого підприємства наведена на рис. 2.2. На цій діаграмі показані зв'язки між користувачами, ролями, клієнтами, замовленнями, сировиною, готовою продукцією, прайс-листом, працівниками, виконаними роботами, заробітною платою та відправками. Центральним елементом схеми є сутність Order, оскільки саме замовлення об'єднує основні бізнес-процеси підприємства.

Об'єктна структура системи відображає логіку предметної області та створює основу для подальшого проектування бази даних. Вона дозволяє визначити основні сутності, їхнє призначення та зв'язки між ними.

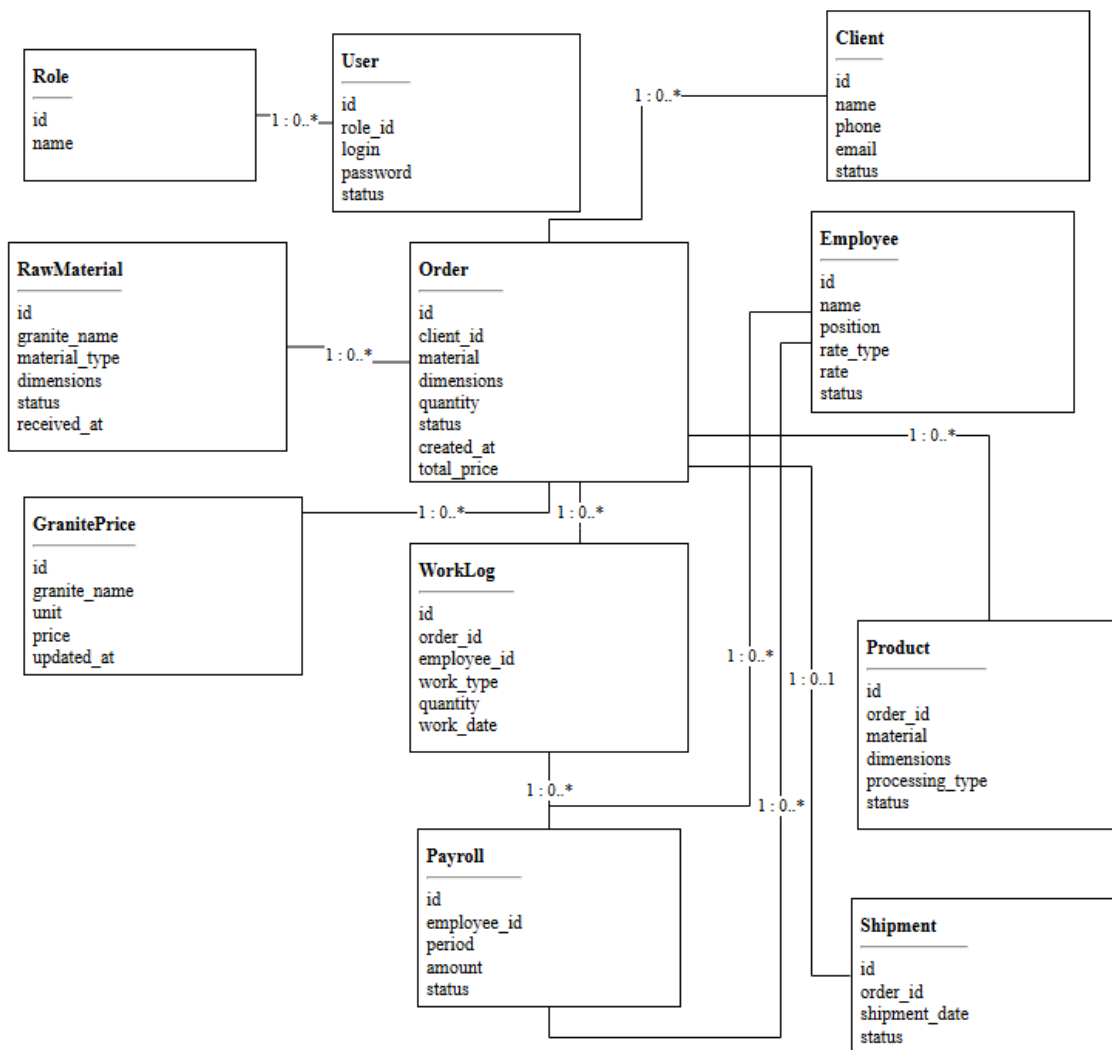


Рис. 2.2. Узагальнена об'єктна структура системи автоматизації діяльності малого підприємства

2.2. Розробка бази даних системи

База даних є основою інформаційного забезпечення системи автоматизації діяльності малого підприємства камінеобробної галузі. Вона призначена для централізованого зберігання даних про користувачів, клієнтів, замовлення, сировину, готову продукцію, маркування, відправки, виробничі операції та оплату праці. Проектування бази даних виконано з урахуванням функціональних вимог, визначених у підрозділі 2.1, а також особливостей предметної області камінеобробного підприємства.

Для реалізації бази даних використовується реляційна модель даних [25, 26]. Такий підхід є доцільним, оскільки основні об'єкти системи мають чітко визначену структуру та пов'язані між собою через ідентифікатори. Реляційна модель дозволяє подати інформацію у вигляді таблиць, забезпечити зв'язки між сутностями, виконувати пошук, фільтрацію та оновлення даних без дублювання основної інформації.

База даних побудована на основі СКБД MySQL із використанням механізму збереження InnoDB. Для таблиць застосовується кодування utf8mb4, що забезпечує коректне збереження українських текстів, назв матеріалів, коментарів, адрес, статусів і службових даних. Використання InnoDB є доцільним, оскільки цей механізм підтримує індекси, транзакції та зовнішні ключі, що важливо для системи, у якій дані різних таблиць логічно пов'язані між собою [27].

Центральною сутністю бази даних є таблиця orders, яка зберігає інформацію про замовлення. Саме замовлення об'єднує клієнта, матеріал, параметри виробу, кількість деталей, спосіб розрахунку, вартість, статус виконання та строк завершення. Інші таблиці доповнюють цю структуру даними про клієнтів, сировину, готову продукцію, прайс граніту, маркування та відправки.

Під час проєктування бази даних враховано такі вимоги:

- збереження основних даних про клієнтів, замовлення, матеріали та продукцію;
- підтримка зв'язку між клієнтом і його замовленнями;
- можливість обліку сировини, поставок і готової продукції;
- підтримка розрахунку площі, об'єму, ваги та вартості виробів;
- збереження фотографій продукції та матеріалів;
- підтримка маркування об'єктів системи;
- облік відправок готової продукції;
- можливість подальшого розширення виробничого та фінансового обліку.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						44
Змн.	Арк.	№ докум.	Підпис	Дата		

Основні таблиці бази даних наведено в табл. 2.6.

Таблиця 2.6

Основні таблиці бази даних системи

Таблиця	Призначення	Основні дані, що зберігаються
users	Зберігання облікових записів користувачів системи	ім'я, прізвище, телефон, email, пароль, роль, виробнича посада
production_positions	Зберігання виробничих посад	назва посади, ключ посади, виробничий профіль, активність
clients	Зберігання даних про клієнтів підприємства	ім'я, прізвище, телефон, email, місто, адреса
orders	Зберігання даних про замовлення	клієнт, матеріал, розміри, кількість, ціна, знижка, статус, строк виконання
order_items	Зберігання позицій замовлення	ідентифікатор замовлення, кількість, ціна
granite_types	Зберігання довідника типів граніту	назва граніту, ціна за м ² , ціна за м ³ , густина, активність
raw_deliveries	Зберігання даних про поставки сировини	номер поставки, постачальник, граніт, вартість, статус, дата надходження
raw_materials	Зберігання даних про сировину	маркування, назва граніту, тип матеріалу, розміри, кількість, стан, дефекти

products	Зберігання даних про готову продукцію	маркування, тип граніту, тип обробки, розміри, кількість, статус пакування
product_photos	Зберігання фотографій продукції	ідентифікатор продукції, шлях до фото, назва файлу, тип файлу
markings	Зберігання маркувань об'єктів	код маркування, тип об'єкта, ідентифікатор об'єкта, назва
shipments	Зберігання даних про відправки	номер відправки, клієнт, замовлення, адреса, служба доставки, статус
shipment_batches	Зберігання партій, пов'язаних із відправкою	ідентифікатор відправки, партія, номер партії, вага

Таблиця clients призначена для збереження інформації про клієнтів підприємства. Вона містить контактні дані клієнта, зокрема ім'я, прізвище, номер телефону, електронну пошту, місто та адресу. Наявність окремої таблиці клієнтів дозволяє не дублювати контактні дані в кожному замовленні та забезпечує можливість перегляду історії звернень конкретного клієнта.

Таблиця orders є центральною таблицею бази даних. Вона містить дані про замовлення: клієнта, тип виробу, матеріал, довжину, ширину, товщину, кількість, площу, об'єм, вагу, тип розрахунку, ціну матеріалу, додаткові послуги, знижку, підсумкову вартість, статус і строк виконання. Така структура дозволяє зберігати як виробничі, так і фінансові параметри замовлення.

Таблиця order_items використовується для деталізації позицій замовлення. Вона пов'язана із таблицею orders за допомогою поля order_id. Такий підхід дає змогу в межах одного замовлення зберігати кілька позицій або окремих складових.

Таблиця `granite_types` виконує роль довідника типів граніту. У ній зберігається назва граніту, ціна за квадратний метр для різної товщини, ціна за кубічний метр, густина матеріалу та ознака активності. Дані цієї таблиці використовуються під час розрахунку вартості замовлень і визначення маси матеріалу.

Таблиця `raw_deliveries` призначена для зберігання інформації про поставки сировини. Вона містить номер поставки, постачальника, назву граніту, транспортні параметри, вартість сировини, вартість доставки, загальну вартість, кількість блоків, об'єм, вагу, статус і дату надходження. Ця таблиця дозволяє контролювати джерело надходження сировини на склад.

Таблиця `raw_materials` використовується для обліку окремих одиниць сировини. У ній зберігаються код маркування, назва граніту, тип матеріалу, розміри, площа, об'єм, вага, кількість, одиниця вимірювання, постачальник, дата надходження, статус, інформація про списання, дефекти, корисні розміри та фотографія. Через поле `raw_delivery_id` сировина може бути пов'язана з конкретною поставкою.

Таблиця `products` призначена для обліку готової продукції. Вона містить маркування, тип граніту, тип обробки, розміри, площу, об'єм, вагу, кількість, статус поверхні, цілісність, опис дефектів, ідентифікатори клієнта, замовлення, сировини та виробничої операції. Це дозволяє простежувати походження готового виробу від сировини або операції розкрою до подальшого пакування чи відправки.

Таблиця `product_photos` використовується для збереження фотографій готової продукції. Винесення фотографій в окрему таблицю є доцільним, оскільки один виріб може мати кілька зображень. У таблиці зберігається шлях до файлу, початкова назва, MIME-тип, порядок сортування та дата створення.

Таблиця `markings` забезпечує збереження кодів маркування об'єктів системи. Вона містить код, тип об'єкта, ідентифікатор об'єкта та назву. Така структура дає змогу застосовувати єдиний підхід до маркування сировини,

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						47
Змн.	Арк.	№ докум.	Підпис	Дата		

продукції або інших об’єктів, не створюючи окрему таблицю маркувань для кожного модуля.

Таблиця shipments використовується для обліку відправок. Вона містить номер відправки, клієнта, замовлення, адресу доставки, дату відправки, службу доставки, номер відстеження, спосіб доставки, транспортні параметри, загальну вагу, статус і коментарі. Таблиця shipment_batches деталізує партії, які входять до конкретної відправки.

Схему взаємодії основних таблиць бази даних наведено на рис. 2.3.

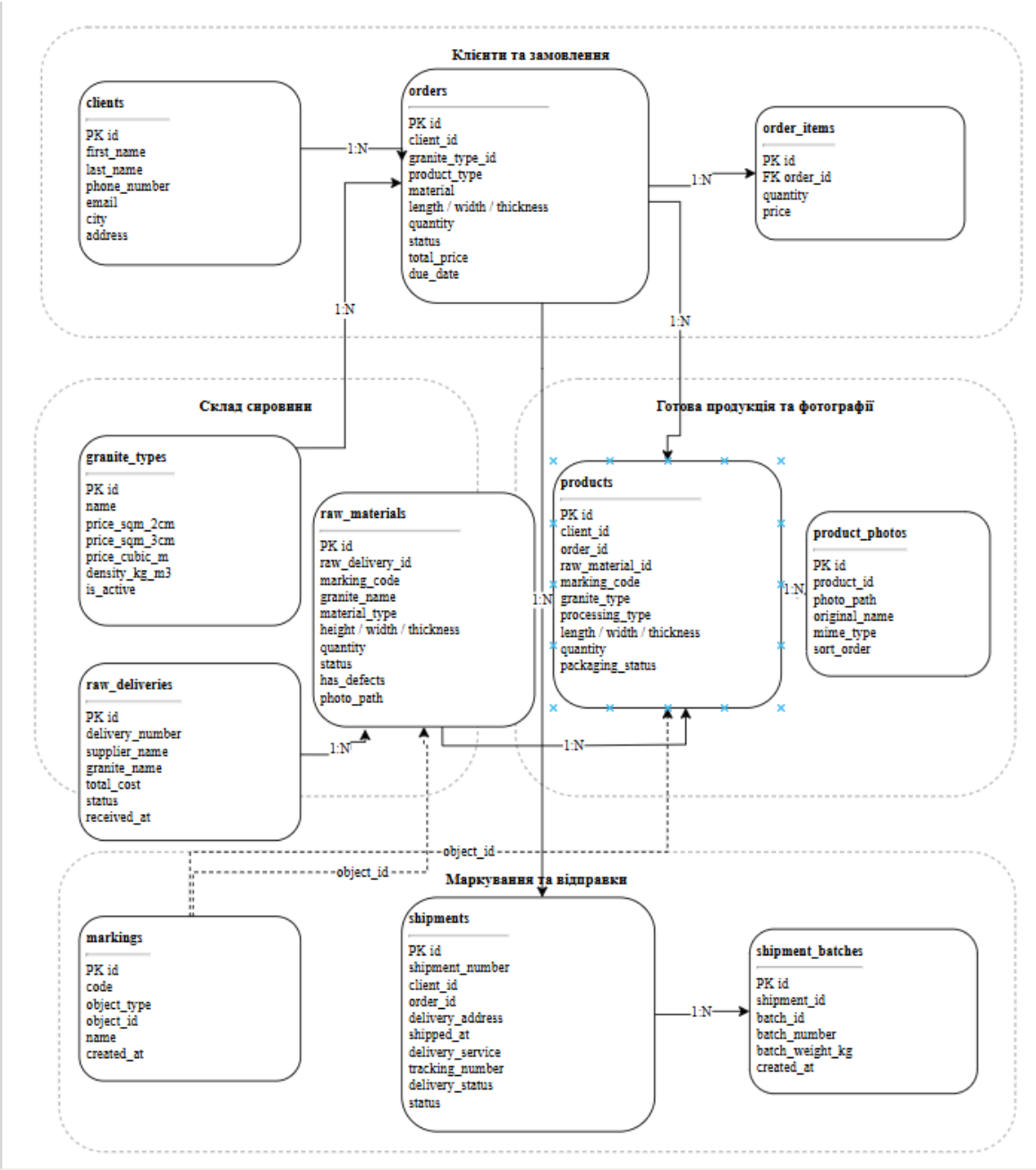


Рис. 2.3. Схема взаємодії основних таблиць бази даних системи

У базі даних частина зв'язків реалізована за допомогою зовнішніх ключів, а частина – через логічні ідентифікатори, які використовуються програмним кодом. Наприклад, фізично заданий зв'язок між order_items і orders забезпечує видалення позицій замовлення разом із самим замовленням. Також таблиця products пов'язана з clients, а таблиця users – з production_positions. Інші зв'язки, зокрема між замовленнями, продукцією, сировиною, маркуваннями та відправками, підтримуються на рівні структури полів і логіки програмного комплексу.

Запропонована структура бази даних забезпечує збереження основних даних, необхідних для роботи системи автоматизації малого каменеобробного підприємства. Вона дозволяє вести облік клієнтів, замовлень, сировини, поставок, готової продукції, фотографій, маркувань і відправок. Використання реляційної моделі створює основу для узгодженої роботи функціональних модулів системи та подальшої реалізації алгоритмів, розглянутих у наступному підрозділі.

2.3. Проектування та реалізація алгоритмів роботи системи

Алгоритми роботи системи автоматизації діяльності малого підприємства каменеобробної галузі визначають послідовність дій користувачів і програмних компонентів під час виконання основних виробничих, складських, фінансових та адміністративних процесів. Розроблювана вебсистема не обмежується збереженням довідкових даних, а відображає повний цикл роботи каменеобробного підприємства: від прийняття замовлення та надходження сировини до виготовлення продукції, її пакування, відправки клієнту та подальшого аналізу результатів діяльності.

Основна логіка системи побудована навколо замовлення та руху матеріалу. Замовлення визначає потребу клієнта, параметри майбутнього виробу, матеріал, кількість деталей, строки виконання та вартість. Поставка сировини забезпечує підприємство матеріалами для виконання замовлень. Далі сировина проходить

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						49
Змн.	Арк.	№ докум.	Підпис	Дата		

класифікацію, маркування, розкрій на деталі, фрезерування, полірування, переміщення на склад готової продукції, пакування та відправлення. Паралельно система фіксує виконані роботи, витрати, використання матеріалів і дані для розрахунку заробітної плати.

У межах програмного комплексу алгоритми реалізуються через взаємодію контролерів, моделей, представлень, клієнтських JavaScript-сценаріїв і бази даних. Користувач виконує дію в інтерфейсі, після чого запит передається відповідному контролеру. Контролер перевіряє права доступу, виконує обробку вхідних даних і звертається до моделі. Модель працює з базою даних, створює або оновлює записи, а результат повертається у представлення у вигляді оновленої сторінки, таблиці, форми або повідомлення.

Початковим етапом роботи є вхід користувача до системи. Після введення облікових даних система перевіряє їх коректність, створює активну сесію та визначає роль користувача. Адміністратор отримує доступ до основних модулів системи: клієнтів, замовлень, команди, сировини, продукції, маркування, відправок, зарплат, витрат і фінансової аналітики. Працівник отримує доступ лише до тих розділів, які пов'язані з його виробничими обов'язками.

Після авторизації користувач може виконувати базові адміністративні дії: додавати клієнтів, створювати замовлення, додавати працівників до команди, налаштовувати виробничі посади та ставки оплати. Дані клієнтів використовуються під час створення замовлень, а дані працівників – під час фіксації виконаних робіт і розрахунку заробітної плати.

Одним із ключових алгоритмів є прийняття та створення замовлення. Користувач обирає наявного клієнта або створює нового, після чого вводить параметри замовлення: тип виробу, матеріал, довжину, ширину, товщину, кількість деталей, додаткові послуги, знижку, строк виконання та примітки. На основі введених параметрів система розраховує площу, об'єм, орієнтовну масу та підсумкову вартість. Після збереження замовлення отримує початковий статус і може бути передане у виробничий процес.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						50
Змн.	Арк.	№ докум.	Підпис	Дата		

Паралельно із замовленнями в системі ведеться облік поставок сировини. Під час надходження матеріалу користувач створює запис про поставку, зазначає постачальника, назву граніту, транспортні параметри, кількість блоків, загальну вагу, вартість сировини, вартість доставки та дату надходження. Після цього окремі одиниці сировини додаються на склад із зазначенням типу матеріалу, розмірів, кількості, статусу, дефектів і фотографій.

Після надходження на склад сировина проходить класифікацію. На цьому етапі визначаються назва граніту, тип матеріалу, розміри, корисні розміри, наявність дефектів, стан матеріалу та можливість подальшого використання. Для кожного об'єкта може формуватися або зберігатися маркування, що дозволяє швидко ідентифікувати матеріал у подальших виробничих операціях.

Наступним етапом є розкрій сировини на деталі. У процесі розкрою система фіксує операцію, вихідні матеріали, отримані деталі, обсяг використаного матеріалу, відходи та залишки. Після розкрою отримані деталі знову можуть проходити класифікацію, оскільки для кожної деталі необхідно зафіксувати розміри, стан, придатність до подальшої обробки та зв'язок із вихідною сировиною або замовленням.

Після розкрою деталі передаються на подальшу обробку. Залежно від виробничого процесу вони можуть проходити фрезерування, полірування або інші операції. Система дозволяє фіксувати виконані роботи, відповідального працівника, тип роботи, кількість або обсяг виконання та дату. Ці дані використовуються не лише для контролю виробничого процесу, а й для подальшого розрахунку оплати праці.

Після завершення обробки вироби переміщуються на склад готової продукції. У таблиці продукції зберігаються маркування, тип граніту, тип обробки, розміри, кількість, стан поверхні, цілісність, наявність дефектів, зв'язок із клієнтом, замовленням, сировиною або виробничою операцією. За потреби до продукції додаються фотографії, які дозволяють візуально контролювати стан виробів.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						51
Змн.	Арк.	№ докум.	Підпис	Дата		

Після потрапляння на склад готової продукції вироби можуть бути передані на пакування. Алгоритм пакування передбачає формування партії, додавання до неї окремих виробів або деталей, фіксацію кількості, ваги та статусу пакування. Такий підхід дозволяє контролювати, які саме вироби вже підготовлені до відправки, а які залишаються на складі.

Завершальним виробничо-логістичним етапом є відправлення продукції клієнту. Під час створення відправки користувач зазначає клієнта, замовлення, адресу доставки, місто, дату відправлення, службу доставки, номер відстеження, транспортні параметри, загальну вагу та статус. Якщо відправка складається з кількох партій, вони фіксуються окремо й пов'язуються із загальним записом відправки.

Маркування супроводжує декілька етапів роботи системи. Воно може використовуватися для сировини, готової продукції, партій або інших об'єктів. Завдяки маркуванню система забезпечує простежуваність матеріалу: від поставки та складу сировини до розкрою, виготовлення деталей, складу готової продукції, пакування і відправки.

Окремим напрямом алгоритмів є фінансовий облік. Система фіксує витрати підприємства, вартість сировини, доставку, виробничі операції, додаткові послуги та нарахування працівникам. На основі цих даних може визначатися собівартість виробів, витрати за період, орієнтовний прибуток і ефективність виконаних замовлень. Дані про виконані роботи використовуються для розрахунку заробітної плати відповідно до посад, ставок і фактичного обсягу виконання.

Для розширеного аналізу передбачено модуль фінансової аналітики. Він може працювати як платна функція, доступна за активною підпискою. Перед відкриттям аналітичного модуля система перевіряє наявність активної підписки користувача або підприємства. Якщо підписка активна, користувач отримує доступ до фінансових показників: доходів, витрат, собівартості, прибутку, ефективності замовлень, динаміки продажів і витрат за обраний період. Якщо

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						52
Змн.	Арк.	№ докум.	Підпис	Дата		

підписка неактивна, система обмежує доступ до аналітики та пропонує активувати відповідну можливість.

Пошук і фільтрація даних реалізуються за єдиним алгоритмом у більшості модулів. Користувач вводить пошуковий запит або задає параметри фільтрації, після чого система формує SQL-запит до відповідної таблиці та повертає відфільтрований результат [28]. Такий механізм використовується в модулях клієнтів, замовлень, сировини, готової продукції, відправок і витрат.

Узагальнений алгоритм роботи системи автоматизації діяльності малого каменеобробного підприємства наведено на рис. 2.4.

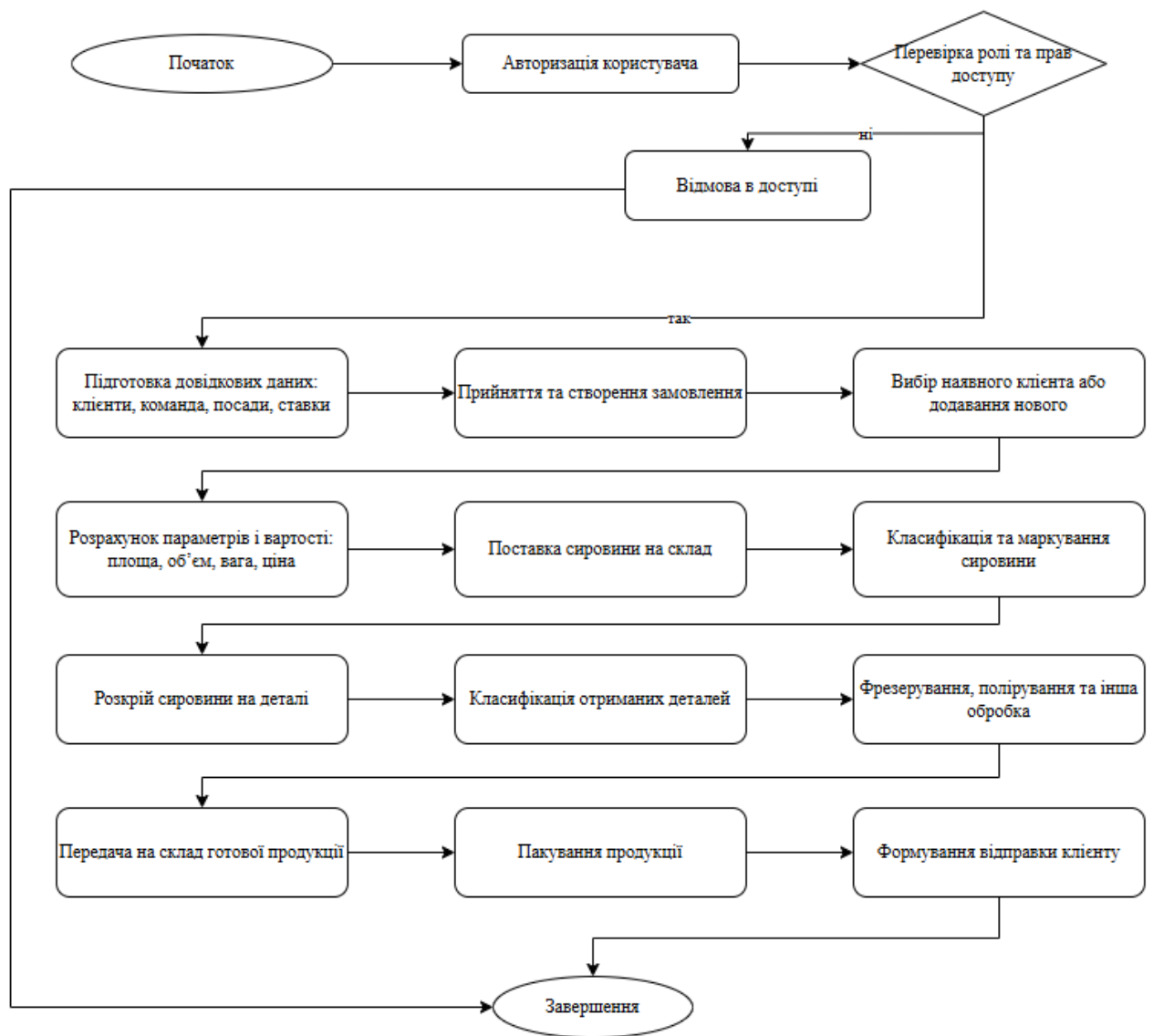


Рис. 2.4. Узагальнений алгоритм роботи системи автоматизації діяльності малого каменеобробного підприємства

Отже, проєктування алгоритмів роботи системи дозволило формалізувати основні сценарії діяльності підприємства. Розроблені алгоритми охоплюють адміністративні дії, прийняття замовлень, облік клієнтів і команди, поставку та класифікацію сировини, розкрій, виробничу обробку, склад готової продукції, пакування, відправлення, маркування, облік витрат, розрахунок заробітної плати та фінансову аналітику. Така логіка забезпечує цілісну автоматизацію виробничого та управлінського процесу підприємства.

2.4. Проєктування та реалізація функціональних модулів системи

Реалізація системи автоматизації діяльності малого підприємства каменеобробної галузі виконана у вигляді взаємопов'язаних функціональних модулів, які забезпечують роботу адміністративної, виробничої, складської, логістичної та фінансової частин програмного комплексу. Система охоплює керування користувачами, клієнтами, командою, замовленнями, прайсом граніту, поставками сировини, складом сировини, виробничими операціями, готовою продукцією, маркуванням, пакуванням, відправками, зарплатами, витратами, інструментом, фінансовою аналітикою та підпискою на платний функціонал.

Систему побудовано на основі власної MVC-структури. Вхідною точкою системи є файл `index.php`, який ініціалізує роботу застосунку та запускає ядро системи. У каталозі `core` розміщено основні службові компоненти, які забезпечують маршрутизацію, роботу контролерів, моделей, бази даних, сесій, шаблонів, захисту CSRF та обробки помилок. Клас `Core` відповідає за ініціалізацію базових компонентів застосунку, `Router` – за розбір маршруту й виклик відповідного контролера, `Controller` – за спільні методи контролерів, `Model` – за базові операції з даними, а `DataBase` – за виконання SQL-запитів до MySQL через PDO [29]. Компонент `Session` використовується для підтримки активної сесії користувача після авторизації [30].

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						54
Змн.	Арк.	№ докум.	Підпис	Дата		

Структура програмного комплексу розділена на каталоги контролерів, моделей, представлень, сервісів, клієнтських сценаріїв і стилів. У каталозі MVC/controllers розміщено контролери, які приймають HTTP-запити, перевіряють права доступу, виконують валідацію та координують прикладну логіку. Каталог MVC/models містить моделі предметної області, які працюють із таблицями бази даних. Представлення розташовані в MVC/views і відповідають за формування сторінок інтерфейсу, форм, таблиць, фільтрів і службових повідомлень. Клієнтські сценарії та стилі розміщено в каталогах MVC/views/js і MVC/views/css.

Окрему роль у реалізації системи відіграє каталог MVC/services. У ньому винесено бізнес-логіку, яка не повинна перевантажувати контролери або моделі. До таких сервісів належать ProductionAccountingService, WorkerProductionService, AnalyticsService, ExpenseService, ToolPurchaseService, SubscriptionService, AuditService, PhotoUploadService, WorkProfileAccessService та інші службові компоненти. Сервіси використовуються для обліку поставок і розкрою, роботи з виробничими операціями, формування аналітики, обліку витрат, перевірки підписки, журналювання дій, роботи з фотографіями та контролю доступу.

Узагальнену відповідність між функціональними модулями системи та основними компонентами їх реалізації наведено в табл. 2.7.

Таблиця 2.7

Реалізація основних функціональних модулів системи

Функціональний модуль	Основне призначення	Основні компоненти реалізації
Авторизація та користувачі	Вхід до системи, вихід, реєстрація, відновлення пароля, керування обліковими записами та ролями	UserController, ProfileController, Users, views/users, views/profile, PasswordResetService, RegistrationVerificationService, GoogleAuthService, MailService, таблиця users

Клієнти	Облік клієнтів підприємства та використання їхніх даних у замовленнях	ClientsController, Clients, views/clients, PhoneService, таблиця clients
Команда та виробничі посади	Облік працівників, посад, виробничих профілів і доступу до робіт	UsersController, ProductionPosition, views/users/team, views/users/positions, WorkProfileAccessService, ProductionPositionService, таблиці users, production_positions
Замовлення	Створення, редагування, перегляд, розрахунок і супровід замовлень	OrderController, Order, views/order, OrderCompletionService, таблиці orders, order_items, clients, granite_types
Прайс граніту	Збереження типів граніту, цін, густини та стану активності матеріалу	GraniteController, GraniteTypes, views/granite, DomainValueValidator, таблиця granite_types
Поставки сировини	Фіксація надходження сировини, постачальника, вартості, доставки та блоків	RawdeliveriesController, RawDeliveries, views/rawdeliveries, ProductionAccountingService, таблиці raw_deliveries, raw_materials
Склад сировини	Облік матеріалів, розмірів, дефектів, фото, статусів, списання та залишків	RawmaterialsController, RawMaterials, views/rawmaterials, RawMaterialTypeService, PhotoUploadService, таблиці raw_materials, raw_material_photos
Розкрій і виробничі операції	Фіксація операцій розкрою, отриманих деталей, залишків, відходів і робочих дій	CuttingoperationsController, WorkerController, CuttingOperations, views/cuttingoperations, views/worker, ProductionAccountingService, WorkerProductionService, таблиці cutting_operations, cutting_outputs, raw_materials, products

Склад готової продукції	Облік готових виробів, їхніх параметрів, статусів, походження, дефектів і списання	ProductsController, Products, views/products, PhotoUploadService, таблиці products, product_photos
Фотографії продукції	Завантаження, збереження та відображення фотографій готової продукції	ProductsController, partials/photo_uploader, PhotoUploadService, photo-uploader.js, таблиця product_photos
Маркування	Створення, збереження, перегляд і друк кодів маркування об'єктів системи	MarkingsController, Markings, views/markings, AuditObjectLabelService, таблиця markings
Пакування	Формування партій готової продукції, додавання виробів, зміна статусів і підготовка до відправки	PackingController, views/packing, таблиці packing_batches, packing_batch_items, products, shipments
Відправки	Облік доставки продукції клієнту, партій відправки, статусів і транспортних параметрів	ShipmentsController, Shipments, views/shipments, таблиці shipments, shipment_batches, packing_batches
Зарплати	Облік виконаних робіт, ставок, відвідуваності та розрахунок оплати праці	PayrollsController, views/payrolls, WorkerProductionService, AttendanceService, SalaryRateService, таблиці employee_work_logs, salary_rates, users
Витрати	Облік витрат підприємства, категорій, сум, дат і скасування помилкових записів	ExpensesController, BusinessExpense, views/expenses, ExpenseService, expense-form.js, таблиці business_expenses, expense_categories
Інструмент	Облік інструменту, закупівель, списання та зв'язку з витратами	ToolsController, Tools, views/tools, ToolPurchaseService, таблиці tools, tool_assignments, business_expenses

Фінансова аналітика	Відображення доходів, витрат, зарплат, вартості сировини, інструменту та підсумкових показників	AnalyticsController, views/analytics, AnalyticsService, analytics.js, таблиці orders, business_expenses, employee_work_logs, raw_deliveries, tools
Підписка на платний функціонал	Перевірка доступу до фінансової аналітики, створення платежів і обробка webhook	BillingController, WebhooksController, SubscriptionService, SubscriptionRepository, MonobankAcquiringClient, таблиці production_subscriptions, subscription_payments
Журналювання дій	Фіксація важливих дій користувачів і перегляд журналу подій	AuditController, AuditService, AuditObjectLabelService, views/audit, таблиця audit_logs

Модуль авторизації та користувачів забезпечує вхід до системи, вихід, реєстрацію, відновлення пароля, підтвердження контактних даних і роботу з профілем. Його реалізація пов'язана з UsersController, ProfileController, моделлю Users і службовими сервісами для відновлення пароля, перевірки контактних даних, надсилання листів і авторизації через зовнішні облікові записи. Дані користувачів зберігаються в таблиці users. Через роль користувача та виробничу посаду система визначає, які функціональні модулі доступні конкретному користувачу.

Модуль клієнтів реалізує створення, редагування, перегляд і пошук клієнтів підприємства. Контролер ClientsController обробляє запити, пов'язані з додаванням і зміною даних клієнта, модель Clients забезпечує доступ до таблиці clients, а представлення views/clients формують відповідні сторінки інтерфейсу. Цей модуль використовується разом із модулем замовлень, оскільки кожне замовлення може бути пов'язане з конкретним клієнтом.

Модуль команди та виробничих посад використовується для обліку працівників, їхніх посад і доступу до виробничих функцій. У межах цього модуля адміністратор може додавати працівників, призначати їм ролі, виробничі

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						58
Змн.	Арк.	№ докум.	Підпис	Дата		

профілі та посади. Реалізація модуля пов'язана з UsersController, моделлю ProductionPosition, сервісами WorkProfileAccessService і ProductionPositionService, а також таблицями users і production_positions. Така структура дозволяє розділити адміністративні та виробничі права користувачів.

Модуль замовлень є одним із центральних у системі. Він реалізований через OrderController, модель Order, представлення views/order і таблиці orders та order_items. У цьому модулі користувач створює замовлення, обирає клієнта, вказує матеріал, розміри, кількість, додаткові послуги, знижку та строк виконання. Система виконує розрахунок площі, об'єму, ваги та вартості замовлення, після чого зберігає результат у базі даних. Статуси замовлення використовуються для контролю його виконання.

Фрагмент програмної реалізації розрахунку вартості замовлення наведено нижче. Повний код відповідного методу подано в додатку А.1.

```
public static function calculatePrice($data){
$length = max(0, (float)($data['length'] ?? 0));
$width = max(0, (float)($data['width'] ?? 0));
$thickness = max(0, (float)($data['thickness'] ?? 0));
$quantity = max(1, (int)($data['quantity'] ?? 1));
$area = max(0, ($length * $width) / 1000000);
$volume = max(0, ($length * $width * $thickness) / 1000000000);
$unitPrice = (float)($data['granite_unit_price'] ?? $data['material_price'] ?? 0);
$singleDetailPrice = ($thickness >= 40 ? $volume : $area) * $unitPrice;
$base = $singleDetailPrice * $quantity;
$extras = max(0, (float)($data['extra_services_price'] ?? 0));
$subtotal = $base + $extras;
$discountPercent = min(100, max(0, (float)($data['discount_percent'] ?? 0)));
return round($subtotal - ($subtotal * $discountPercent / 100), 2);}
```

У наведеному фрагменті показано, що система виконує розрахунок площі, об'єму, знижки та підсумкової вартості на рівні моделі. Централізація розрахункової логіки в моделі дозволяє використовувати її в різних частинах програмного комплексу та уникати дублювання формул у контролерах або представленнях.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		59

Модуль прайсу граніту реалізує довідник матеріалів. Він пов'язаний із GraniteController, моделлю GraniteTypes, представленнями views/granite і таблицею granite_types. У межах цього модуля зберігаються назви типів граніту, ціни за квадратний метр для різної товщини, ціна за кубічний метр, густина матеріалу та ознака активності. Дані з прайсу використовуються під час створення та розрахунку замовлення.

Модуль поставок сировини відповідає за фіксацію надходження матеріалу на підприємство. Його реалізація пов'язана з RawdeliveriesController, моделлю RawDeliveries, представленнями views/rawdeliveries і сервісом ProductionAccountingService. У цьому модулі зберігається інформація про постачальника, номер поставки, назву граніту, вартість матеріалу, вартість доставки, кількість блоків, загальну вагу та дату надходження. Дані поставки можуть бути пов'язані з окремими одиницями сировини на складі.

Модуль складу сировини реалізує облік матеріалів, які перебувають на підприємстві. Контролер RawmaterialsController обробляє дії користувача, модель RawMaterials працює з таблицею raw_materials, а представлення views/rawmaterials забезпечують перегляд, фільтрацію, додавання, редагування, історію та списання матеріалів. У цьому модулі також використовується RawMaterialTypeService для нормалізації типів матеріалів і PhotoUploadService для роботи з фотографіями [31]. Окремо в службовій частині системи передбачено захист критичних POST-запитів від CSRF-атак за допомогою перевірки токенів [32, 33].

Модуль розкрою та виробничих операцій забезпечує фіксацію виробничих дій, пов'язаних із перетворенням сировини на деталі або готову продукцію. Управлінський розкрій реалізовано через CuttingoperationsController і представлення views/cuttingoperations. Робочі сторінки для виконання окремих операцій пов'язані з WorkerController і views/worker. Сервіси ProductionAccountingService і WorkerProductionService відповідають за створення виробничих записів, формування результатів операцій, перерахунок залишків, створення продукції та фіксацію виконаних робіт працівників.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						60
Змн.	Арк.	№ докум.	Підпис	Дата		

Модуль складу готової продукції реалізований через ProductsController, модель Products, представлення views/products і таблицю products. Він забезпечує облік готових виробів, їхніх розмірів, типу граніту, типу обробки, кількості, стану поверхні, цілісності, дефектів, зв'язку із замовленням, клієнтом, сировиною або виробничою операцією. Для продукції можуть додаватися фотографії, які зберігаються в таблиці product_photos.

Модуль фотографій продукції забезпечує завантаження та відображення зображень виробів. Для цього використовується PhotoUploadService, часткове представлення photo_uploader, клієнтський сценарій photo-uploader.js і таблиця product_photos. Винесення фотографій в окремий модуль спрощує повторне використання завантажувача та дозволяє зберігати декілька зображень для одного виробу.

Модуль маркування призначений для створення та збереження кодів маркування об'єктів системи. Його реалізація пов'язана з MarkingsController, моделлю Markings, представленнями views/markings і таблицею markings. Маркування може використовуватися для сировини, продукції, партій та інших об'єктів. Це забезпечує простежуваність матеріалу на різних етапах виробничого процесу.

Модуль пакування реалізований через PackingController і представлення views/packing. Він використовує таблиці packing_batches і packing_batch_items для формування партій продукції. У межах цього модуля користувач може створити партію пакування, додати до неї вироби, змінити статус, розпакувати позиції або підготувати партію до відправки. Пакування є проміжним етапом між складом готової продукції та логістичним модулем відправок.

Модуль відправок відповідає за контроль передавання продукції клієнту. Він реалізований через ShipmentsController, модель Shipments, представлення views/shipments і таблиці shipments та shipment_batches. У цьому модулі зазначаються клієнт, замовлення, адреса доставки, служба доставки, номер відстеження, транспортні параметри, вага та статус відправлення. Якщо

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						61
Змн.	Арк.	№ докум.	Підпис	Дата		

відправка складається з кількох партій, вони пов'язуються із загальним записом відправки.

Модуль зарплат призначений для фіксації виконаних робіт, налаштування ставок і розрахунку оплати праці. Його реалізація пов'язана з PayrollsController, представленнями views/payrolls, сервісами WorkerProductionService і AttendanceService, а також таблицями employee_work_logs і salary_rates. Дані цього модуля використовуються для контролю виробничої активності працівників і подальшого фінансового аналізу.

Модуль витрат реалізує облік фінансових витрат підприємства. Він пов'язаний з ExpensesController, моделлю BusinessExpense, представленнями views/expenses, сервісом ExpenseService і таблицями business_expenses та expense_categories. У межах цього модуля можна додавати витрати, обирати категорію, зазначати суму, дату, опис і скасовувати помилкові записи.

Модуль інструменту призначений для обліку інструменту, закупівель і списання. Його реалізація пов'язана з ToolsController, моделлю Tools, представленнями views/tools, сервісом ToolPurchaseService і таблицями tools та tool_assignments. Закупівля інструменту може бути пов'язана з витратами підприємства, тому цей модуль взаємодіє з фінансовою частиною системи.

Модуль фінансової аналітики реалізований через AnalyticsController, представлення views/analytics, клієнтський сценарій analytics.js і сервіс AnalyticsService. Він формує показники доходів, витрат, зарплат, вартості сировини, інструменту, собівартості та результативності діяльності підприємства. Оскільки фінансова аналітика є платним функціоналом, доступ до неї перевіряється через SubscriptionService.

Фрагмент перевірки активної підписки на фінансову аналітику наведено нижче. Повний код сервісу подано в додатку А.2.

```
public function hasActiveFeature(int $productionId, string $featureKey): bool
{
    return $this->subscriptions->findActiveFeature(
        $productionId,
        $featureKey,
```

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						62
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$this->now()
) !== null;
}
public function getFeatureSubscription(int $productionId, string $featureKey): ?object
{
    return $this->subscriptions->findFeatureForProduction($productionId, $featureKey);
}
public function canAccessFeature(int $productionId, string $featureKey): bool
{
    return $this->hasActiveFeature($productionId, $featureKey);
}

```

У наведеному фрагменті показано, що перед відкриттям аналітичного модуля система перевіряє наявність активної підписки. Такий підхід дозволяє відокремити базовий функціонал системи від платного аналітичного блоку.

Модуль підписки на платний функціонал реалізований через BillingController, WebhooksController, SubscriptionService, SubscriptionRepository і MonobankAcquiringClient. У базі даних для цього використовуються таблиці production_subscriptions і subscription_payments. Модуль відповідає за створення платежу, перевірку статусу підписки, обробку відповіді платіжного сервісу та відкриття доступу до платної фінансової аналітики.

Модуль журналювання дій забезпечує фіксацію важливих операцій користувачів. Він реалізований через AuditController, AuditService, AuditObjectLabelService, представлення views/audit і таблицю audit_logs. Журналювання дозволяє переглядати історію дій, відстежувати зміни в системі та підвищує контрольованість роботи програмного комплексу.

Компонентна структура функціональних модулів системи наведена на рис. 2.5. Вона відображає взаємодію користувача з інтерфейсом, контролерами, моделями, сервісами, базою даних, файловою підсистемою, платіжним сервісом і аналітичним модулем.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		63

Компонентна структура функціональних модулів системи

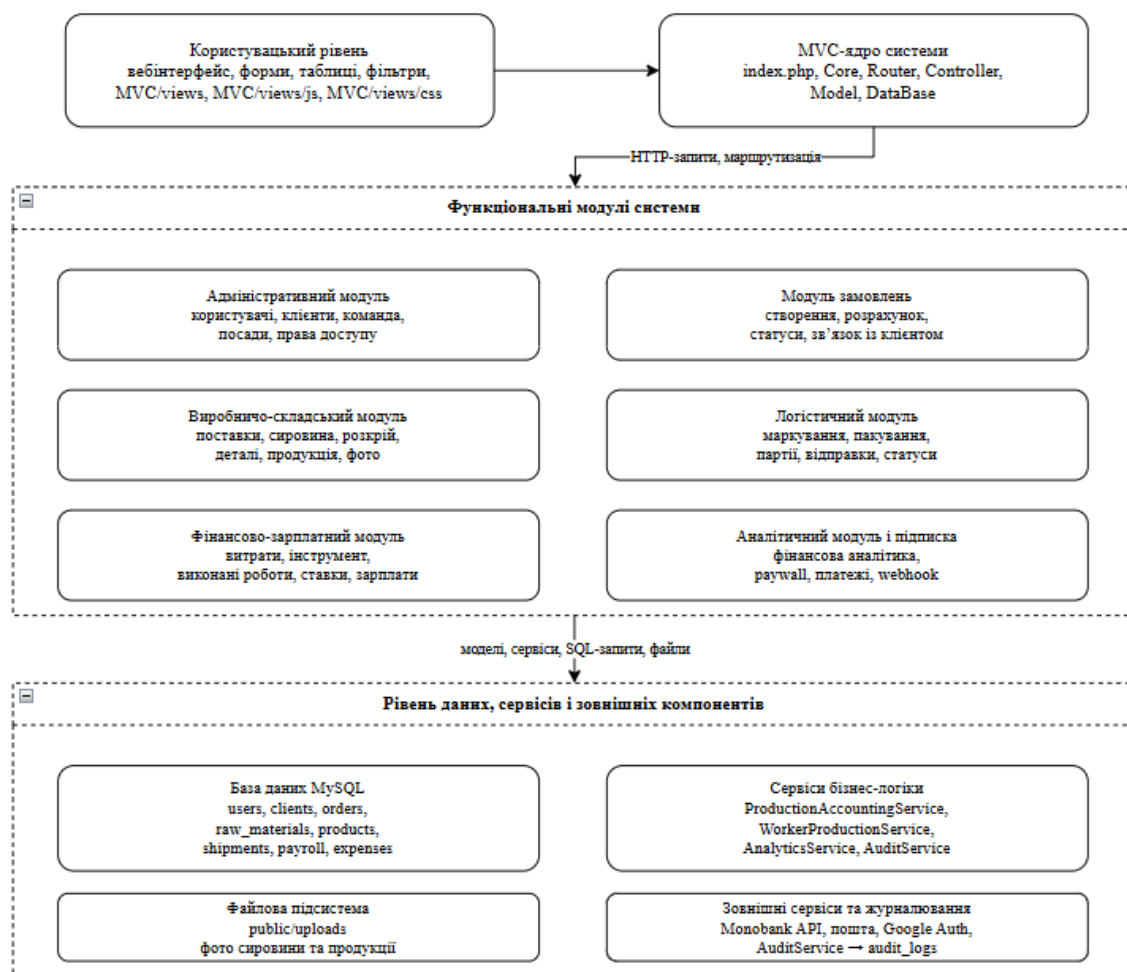


Рис. 2.5. Компонентна структура функціональних модулів системи автоматизації діяльності малого підприємства

Отже, реалізація функціональних модулів системи автоматизації діяльності малого каменеобробного підприємства забезпечує повний цикл роботи програмного комплексу: від авторизації користувача, обліку клієнтів і створення замовлення до поставки сировини, розкрою, виготовлення продукції, пакування, відправки, розрахунку зарплат, обліку витрат, фінансової аналітики та журналювання дій. Використання MVC-структури, окремих сервісів і модульної організації коду спрощує підтримку системи, зменшує дублювання логіки та створює основу для подальшого розвитку функціональності.

Висновки до розділу 2

У другому розділі було виконано проектування та обґрунтування основних технічних рішень для системи автоматизації діяльності малого підприємства каменеобробної галузі. На основі аналізу предметної області визначено загальну структуру програмного комплексу, основні категорії користувачів, функціональні та нефункціональні вимоги, варіанти використання, об'єктну структуру, базу даних, алгоритми роботи та функціональні модулі системи.

У межах проектування загальної структури програмного комплексу було визначено три основні категорії учасників взаємодії із системою: адміністратора, працівника та клієнта. Для кожної категорії встановлено межі доступу та основні сценарії роботи. Центральною сутністю системи визначено замовлення, оскільки саме воно поєднує клієнта, матеріал, параметри виробу, додаткові послуги, вартість, статус виконання та подальшу відправку. Такий підхід дозволяє побудувати логіку системи навколо реального виробничого процесу підприємства.

Під час проектування бази даних обґрунтовано використання реляційної моделі даних і СКБД MySQL. Визначено основні таблиці, необхідні для збереження даних про користувачів, клієнтів, замовлення, прайс граніту, поставки сировини, склад сировини, готову продукцію, фотографії, маркування та відправки. Запропонована структура бази даних забезпечує логічні зв'язки між основними сутностями системи, підтримує пошук, фільтрацію, облік виробничих даних і створює основу для подальшого розширення фінансового та виробничого функціоналу.

У процесі проектування алгоритмів роботи системи було формалізовано основний цикл діяльності каменеобробного підприємства: авторизація користувача, створення клієнта та замовлення, надходження сировини, її класифікація, маркування, розкрій, обробка деталей, переміщення продукції на склад, пакування та відправлення клієнту. Окремо враховано алгоритми обліку витрат, фіксації виконаних робіт, розрахунку заробітної плати та доступу до

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						65
Змн.	Арк.	№ докум.	Підпис	Дата		

фінансової аналітики. Це дозволило описати не лише окремі дії користувачів, а й повний рух даних і матеріалів у межах програмного комплексу.

Також було спроектовано та описано функціональні модулі системи. Вебсистему реалізовано на основі MVC-структури з поділом на контролери, моделі, представлення, сервіси, клієнтські сценарії та стилі. Такий підхід забезпечує логічне розділення відповідальності між частинами системи: контролери обробляють запити користувачів, моделі працюють із даними, представлення формують інтерфейс, а сервіси містять бізнес-логіку для виробничого обліку, аналітики, підписок, роботи з фотографіями та журналювання дій.

У результаті виконаного проєктування було сформовано технічну основу для реалізації системи автоматизації малого каменеобробного підприємства. Запропоновані рішення забезпечують централізований облік клієнтів, замовлень, сировини, продукції, працівників, зарплат, витрат, відправок і фінансових показників. Розроблена структура є достатньо гнучкою для подальшого розвитку системи, додавання нових модулів і розширення функціональності відповідно до потреб підприємства.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		66

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ

3.1. Порядок встановлення та налаштування параметрів системи

Розгортання системи автоматизації діяльності малого підприємства каменеобробної галузі передбачає підготовку серверного середовища, бази даних, конфігураційних файлів, файлової підсистеми та початкових параметрів роботи програмного комплексу. Оскільки система реалізована як веборієнтований PHP-застосунок із власною MVC-структурою, для її роботи необхідні вебсервер, інтерпретатор PHP, система керування базами даних MySQL або сумісна MariaDB, а також доступ до файлової системи для збереження фотографій сировини, продукції та службових файлів.

Система може бути розгорнута як у локальному середовищі розробки, так і на віртуальному хостингу або сервері. Для локального запуску доцільно використовувати середовище, яке містить Apache або Nginx, PHP та MySQL. Для роботи з кодом може використовуватися середовище розробки PhpStorm, а для адміністрування бази даних – phpMyAdmin або інший клієнт MySQL. У разі розгортання на хостингу необхідно переконатися, що сервер підтримує потрібну версію PHP, розширення для роботи з MySQL, файлові завантаження та сесії користувачів.

Основним середовищем виконання системи є PHP. Система використовує об'єктно-орієнтований підхід, простори імен, класи контролерів, моделей і сервісів, тому для стабільної роботи доцільно використовувати сучасну версію PHP. Для збереження даних використовується реляційна база даних MySQL. Такий вибір є обґрунтованим, оскільки система працює з великою кількістю взаємопов'язаних сутностей: користувачами, клієнтами, замовленнями, поставками сировини, складом сировини, готовою продукцією, пакуванням, відправками, зарплатами, витратами, інструментом, підписками та журналом дій.

Для коректної роботи програмного комплексу на сервері мають бути доступні основні PHP-розширення. Розширення pdo та pdo_mysql

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						67
Змн.	Арк.	№ докум.	Підпис	Дата		

використовуються для підключення до бази даних MySQL, session – для підтримки авторизації користувачів, json – для обробки структурованих даних, mbstring – для коректної роботи з українськими текстами, fileinfo, gd та exif– для перевірки й обробки зображень, curl та openssl – для взаємодії із зовнішніми сервісами, а hash, filter і ctype – для перевірки та обробки вхідних даних.

Вхідною точкою програмного комплексу є файл index.php. Після надходження HTTP-запиту вебсервер передає його застосунку, де запит обробляється ядром системи. Основні службові компоненти розміщено в каталозі core. До них належать класи Core, Router, Controller, Model, DataBase, Session, Template, Security та інші допоміжні компоненти. Клас Router визначає потрібний контролер і метод дії, Controller забезпечує спільні методи обробки запитів і перевірки доступу, Model задає основу для роботи з даними, а DataBase відповідає за виконання SQL-запитів до MySQL через PDO.

Файли програмного комплексу мають бути розміщені зі збереженням структури каталогів. Каталог MVC/controllers містить контролери, які обробляють дії користувачів. У MVC/models розміщено моделі, що працюють із таблицями бази даних. Каталог MVC/views містить сторінки інтерфейсу, форми, таблиці та службові шаблони. У MVC/services винесено бізнес-логіку, пов'язану з виробничим обліком, зарплатами, аналітикою, підписками, фотографіями та журналюванням дій. Клієнтські сценарії та стилі розміщено в MVC/views/js і MVC/views/css.

Першим етапом розгортання є створення бази даних. Для цього в MySQL створюється порожня база, після чого імпортується SQL-файл зі структурою таблиць. У базі мають бути створені таблиці для користувачів, клієнтів, замовлень, прайсу граніту, поставок сировини, складу сировини, продукції, фотографій, маркування, пакування, відправок, зарплат, витрат, інструменту, підписок і журналу дій. Після імпорту структури необхідно перевірити наявність основних таблиць і коректність їхніх зв'язків.

Підключення до бази даних налаштовується у конфігураційному файлі, розміщеному в каталозі config. У цьому файлі зазначаються адреса сервера бази

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						68
Змн.	Арк.	№ докум.	Підпис	Дата		

даних, назва бази, ім'я користувача та пароль. Користувач бази даних повинен мати права на читання, створення, зміну й видалення записів у межах відповідної бази. У production-середовищі конфігураційні дані, що містять пароль або інші секретні значення, не повинні публікуватися у відкритому доступі.

Окремо налаштовується файлова підсистема. Вона використовується для збереження фотографій сировини, фотографій готової продукції, службових файлів, тимчасових файлів і журналів роботи системи. Після перенесення проєкту на сервер потрібно перевірити, чи існують відповідні директорії для завантажень, а також чи має вебсервер права на запис у ці каталоги. Якщо права доступу налаштовані неправильно, не працюватиме завантаження фотографій або збереження службових даних.

За потреби налаштовуються поштові параметри [34]. Поштовий сервіс може використовуватися для відновлення пароля, підтвердження контактних даних або надсилання службових повідомлень. У конфігурації задаються SMTP-хост, порт, тип шифрування, логін, пароль або app password, адреса відправника та базова адреса застосунку. Базова адреса потрібна для правильного формування посилань у листах.

Якщо в системі використовується авторизація через Google, додатково налаштовуються параметри Google OAuth. Для цього в конфігурації зовнішніх сервісів задаються client_id, client_secret і redirect_uri. Значення redirect_uri має відповідати маршруту, який обробляє повернення користувача після авторизації. У разі зміни домену або протоколу сайту цей параметр потрібно оновити як у конфігурації проєкту, так і в налаштуваннях Google Cloud Console [35].

Для роботи платної фінансової аналітики може використовуватися інтеграція з monobank [36]. У цьому випадку в конфігурації задаються токен API, публічна адреса застосунку, адреса повернення після оплати та webhook URL. Webhook використовується для отримання повідомлень від платіжного сервісу про зміну статусу платежу або підписки. У production-середовищі важливо забезпечити перевірку достовірності повідомлень платіжного сервісу.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						69
Змн.	Арк.	№ докум.	Підпис	Дата		

Після завершення налаштування необхідно перевірити запуск системи. Насамперед перевіряється відкриття головної сторінки, робота маршрутизації, підключення до бази даних, запуск сесій, авторизація користувача та доступність основних розділів. Далі перевіряються прикладні сценарії: створення клієнта, створення замовлення, додавання сировини, завантаження фотографій, створення продукції, формування пакування, створення відправки, облік витрат, фіксація виконаних робіт, розрахунок зарплати та відкриття фінансової аналітики.

Узагальнений перелік основних параметрів конфігурації вебсистеми наведено в табл. 3.1.

Таблиця 3.1

Основні параметри конфігурації вебсистеми

Параметр	Призначення	Місце налаштування	Примітка
dbHost	Адреса сервера бази даних	config/database.php	Зазвичай localhost для локального сервера
dbName	Назва бази даних	config/database.php	Має відповідати створеній MySQL-базі
dbLogin	Ім'я користувача бази даних	config/database.php	Користувач повинен мати права на читання і запис
dbPassword	Пароль користувача бази даних	config/database.php	Не публікується у відкритому доступі
appBaseUrl	Базова адреса застосунку	конфігураційні файли системи	Використовується для формування посилань
mailHost	SMTP-сервер	config/mail.php	Потрібний для надсилання email

Продовження табл. 3.1

mailPort	Порт SMTP	config/mail.php	Залежить від поштового провайдера
mailUsername	Логін поштового акаунта	config/mail.php	Не публікується у відкритому доступі
mailPassword	Пароль або app password	config/mail.php	Має зберігатися як секретне значення
google.client_id	Ідентифікатор OAuth-клієнта	config/services.php	Використовується для входу через Google
google.client_secret	Секрет OAuth-клієнта	config/services.php	Не публікується у відкритому доступі
google.redirect_uri	Адреса повернення Google OAuth	config/services.php	Має відповідати маршруту callback
monobank.token	Токен API monobank	конфігурація платіжного сервісу	Не публікується у відкритому доступі
monobank.webhook_url	Адреса webhook	конфігурація платіжного сервісу	Потрібна для підтвердження платежів
uploadPath	Директорія завантажених файлів	public/uploads або службова конфігурація	Використовується для фотографій
APP_ENV	Режим роботи застосунку	конфігурація середовища	У production не слід показувати детальні помилки
CSRF-токени	Захист POST-запитів	службовий компонент захисту	Токени зберігаються в сесії

Схему ресурсів вебсистеми наведено на рис. 3.1. Вона відображає основні компоненти середовища розгортання: браузер користувача, вебсервер, PHP-застосунок, MVC-ядро, базу даних MySQL, файлову підсистему та зовнішні

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						71
Змн.	Арк.	№ докум.	Підпис	Дата		

сервіси. Користувач взаємодіє із системою через браузер. Запит надходить на вебсервер, після чого передається PHP-застосунку. Серверна частина обробляє запит через власну MVC-структуру, звертається до бази даних, файлової підсистеми або зовнішніх сервісів і повертає результат користувачу.

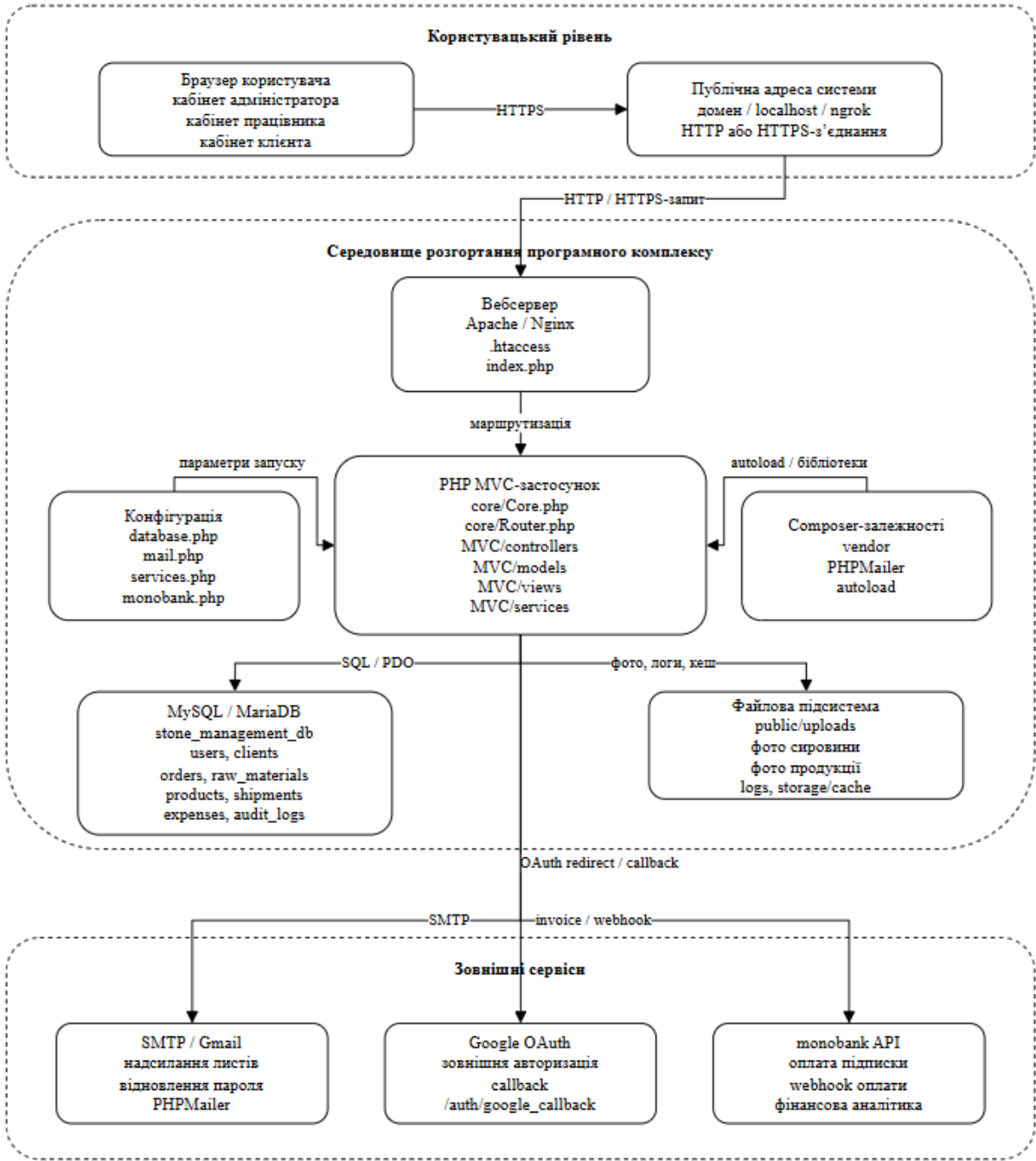


Рис. 3.1. Схема ресурсів системи автоматизації діяльності малого підприємства каменеобробної галузі

Налаштування прав доступу виконується на основі ролей користувачів і виробничих посад. Адміністратор має доступ до всіх основних модулів системи:

користувачів, клієнтів, замовлень, сировини, продукції, прайсу граніту, пакування, відправок, зарплат, витрат, інструменту та фінансової аналітики. Працівник отримує доступ лише до тих розділів, які відповідають його виробничим обов'язкам. Клієнт має обмежений доступ до інформації, що стосується його власних замовлень.

Узагальнений розподіл прав доступу користувачів системи наведено в табл. 3.2.

Таблиця 3.2

Розподіл прав доступу користувачів системи

Роль	Умови отримання доступу	Основні доступні функції
Адміністратор	Авторизація в системі з роллю адміністратора	Повне керування користувачами, клієнтами, замовленнями, сировиною, продукцією, прайсом, зарплатами, витратами, аналітикою та налаштуваннями
Працівник	Авторизація в системі та наявність відповідної виробничої посади	Робота з доступними виробничими операціями, замовленнями, сировиною або продукцією відповідно до прав
Клієнт	Авторизація в системі або створення клієнтського облікового запису	Перегляд обмеженої інформації про власні замовлення та їхній стан
Система	Виконання службових процесів без прямої участі користувача	Журналювання дій, перевірка підписки, обробка службових подій і оновлення статусів

Після налаштування ролей потрібно перевірити, що працівник не має доступу до адміністративних розділів, клієнт не може переглядати чужі замовлення, а адміністратор має доступ до всіх службових модулів. Така перевірка є важливою, оскільки помилки в розмежуванні прав можуть призвести до несанкціонованого доступу до виробничої, клієнтської або фінансової інформації.

Отже, розгортання та налаштування програмного комплексу охоплює підготовку PHP-середовища, MySQL-бази даних, конфігураційних файлів, файлової підсистеми, зовнішніх сервісів і прав доступу. Після виконання цих дій

система може бути запущена на локальному сервері або віртуальному хостингу і використана для демонстрації основних сценаріїв роботи: авторизації, керування клієнтами, створення замовлень, обліку сировини, готової продукції, пакування, відправок, зарплат, витрат і фінансової аналітики.

3.2. Структура інтерфейсу системи

Інтерфейс системи автоматизації діяльності малого підприємства каменеобробної галузі побудовано як вебсистему із розділенням доступу відповідно до ролі користувача. Основними категоріями користувачів є адміністратор, працівник і клієнт. Адміністратор отримує доступ до керування клієнтами, замовленнями, сировиною, продукцією, пакуванням, відправками, працівниками, витратами та фінансовою аналітикою. Працівник працює з обмеженим набором виробничих модулів, які відповідають його посаді та функціональним обов'язкам. Клієнт має доступ лише до інформації, що стосується його власних замовлень.

Загальна структура інтерфейсу складається з верхньої панелі, бокового навігаційного меню та основної робочої області [37, 38]. Верхня панель містить назву системи, елементи керування меню та ім'я поточного користувача. Бокове меню використовується для переходу між основними модулями: фінансовою аналітикою, витратами, замовленнями, клієнтами, поставками сировини, прайсом граніту, складом сировини, розкромом, складом продукції, командою виробництва, пакуванням, відправками, маркуванням, обліком інструментів, профілем і виходом із системи. Основна робоча область змінюється залежно від вибраного модуля та містить таблиці, форми, фільтри, картки, статуси, кнопки дій і службові повідомлення.

Початковою сторінкою для роботи з програмним комплексом є сторінка авторизації. Вона призначена для входу користувача до системи за email або номером телефону та паролем. У верхній частині сторінки розміщено назву програмного комплексу й кнопки переходу до сторінок входу та реєстрації.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						74
Змн.	Арк.	№ докум.	Підпис	Дата		

Центральну частину інтерфейсу займає форма авторизації, яка містить поле для введення email або телефону, поле пароля, кнопку перегляду пароля, кнопку входу, посилання для відновлення доступу та перехід до реєстрації нового облікового запису. Після успішної авторизації система визначає роль користувача та відкриває доступ до відповідних функціональних модулів.

Сторінка авторизації користувача в системі показана на рис. 3.2.

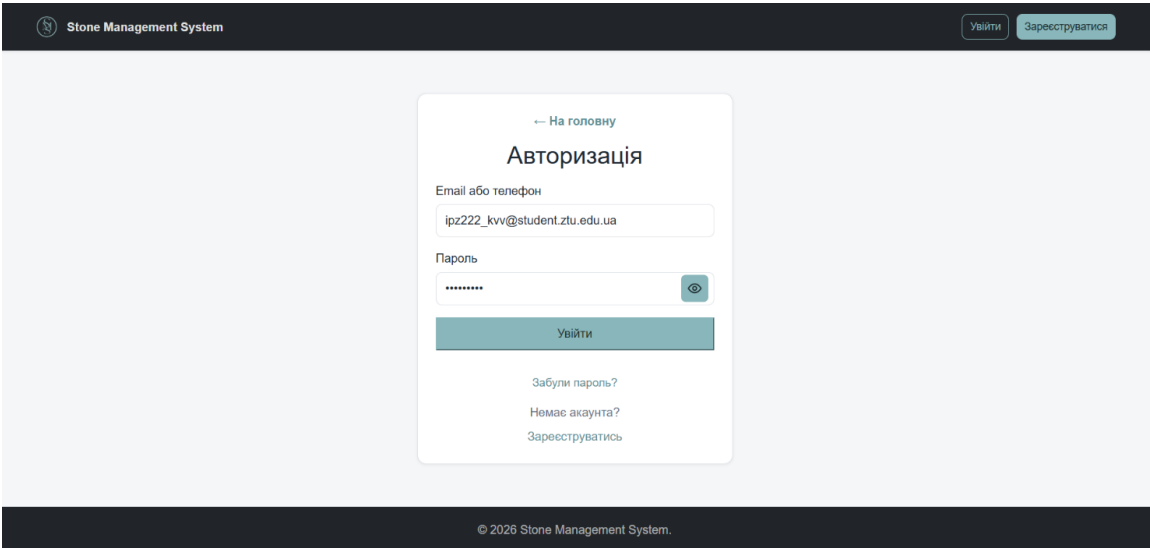


Рис. 3.2. Сторінка авторизації користувача в системі

Після входу користувач потрапляє до внутрішньої частини системи, де доступні основні робочі модулі. На прикладі сторінки фінансової аналітики видно, що деякі можливості системи можуть бути обмежені залежно від стану підписки. Якщо доступ до аналітичного модуля не активовано, система відображає повідомлення про недоступність модуля та пропонує перейти до керування підпискою. Такий підхід дозволяє відокремити базовий функціонал системи від платного аналітичного блоку. При цьому користувач усе одно бачить загальну структуру інтерфейсу, бокове меню та доступні для нього модулі.

Сторінка інтерфейсу системи після авторизації з обмеженим доступом до фінансової аналітики показана на рис. 3.3.

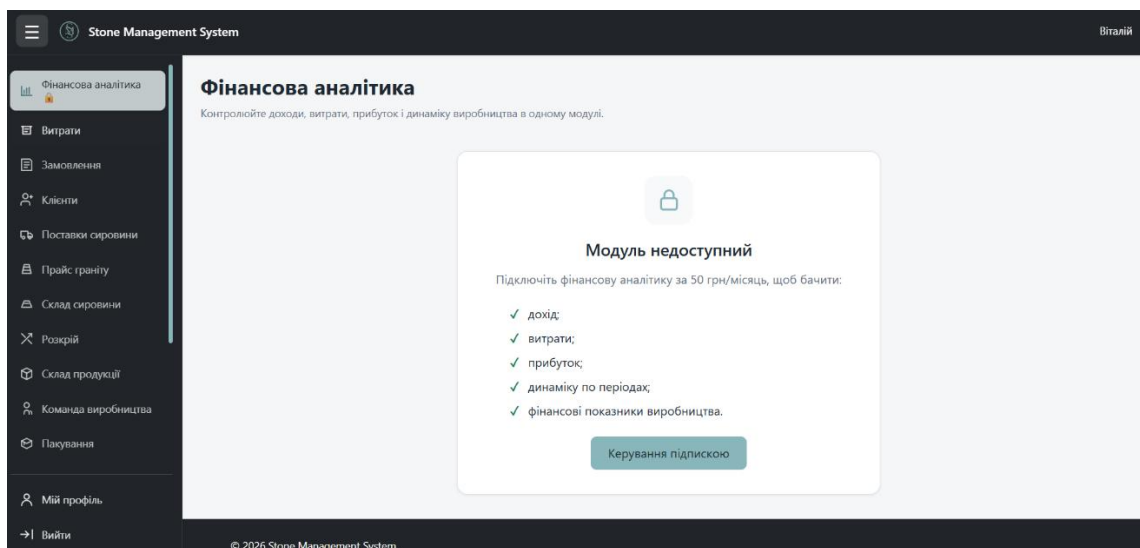


Рис. 3.3. Інтерфейс системи після авторизації з обмеженим доступом до фінансової аналітики

Одним із центральних модулів системи є модуль замовлень. Він використовується для створення, перегляду та супроводу замовлень клієнтів. На сторінці замовлень відображається таблиця з основними даними: номером замовлення, клієнтом, видом граніту, типом виробу, розмірами, кількістю деталей, площею, об'ємом, типом розрахунку та додатковими послугами. У верхній частині сторінки розміщено кнопку пошуку й фільтрації, а також кнопку створення нового замовлення. Така структура дозволяє швидко переглядати поточні замовлення, знаходити потрібний запис і переходити до створення або редагування даних.

Сторінка інтерфейсу модуля керування замовленнями показана на рис. 3.4.

Для детального перегляду окремого замовлення передбачено окрему сторінку, на якій виводиться повна інформація про вибраний запис. У верхній частині сторінки відображається номер замовлення та кнопка повернення до списку. Основна частина сторінки містить дані про клієнта, вид граніту, густину матеріалу, розміри виробу, загальну площу, орієнтовну вагу, ціну з прайсу, вартість усіх деталей, початкову суму, додаткові послуги та підсумкову суму до оплати.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		76

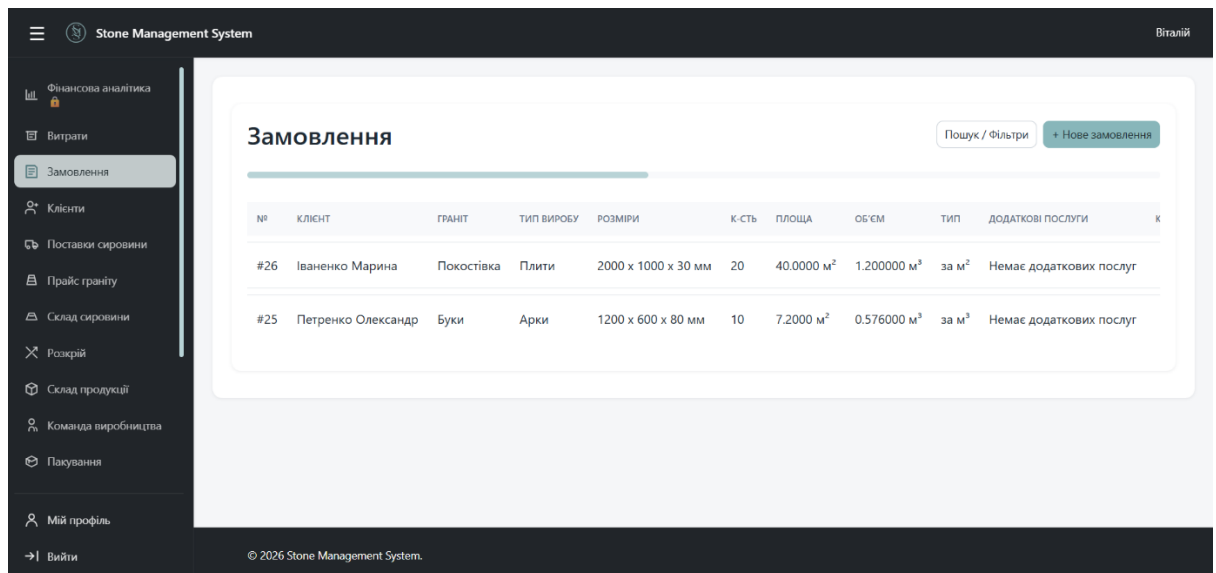


Рис. 3.4. Інтерфейс модуля керування замовленнями

Окремо відображаються статус замовлення, тип виробу, кількість деталей, об'єм, тип розрахунку, вартість однієї деталі, знижка та сума знижки. У нижній частині сторінки розміщено кнопки редагування та видалення. Такий вигляд сторінки дозволяє адміністратору або менеджеру швидко оцінити параметри замовлення та перевірити правильність розрахунку вартості.

Сторінка інтерфейсу перегляду детальної інформації про замовлення показана на рис. 3.5.

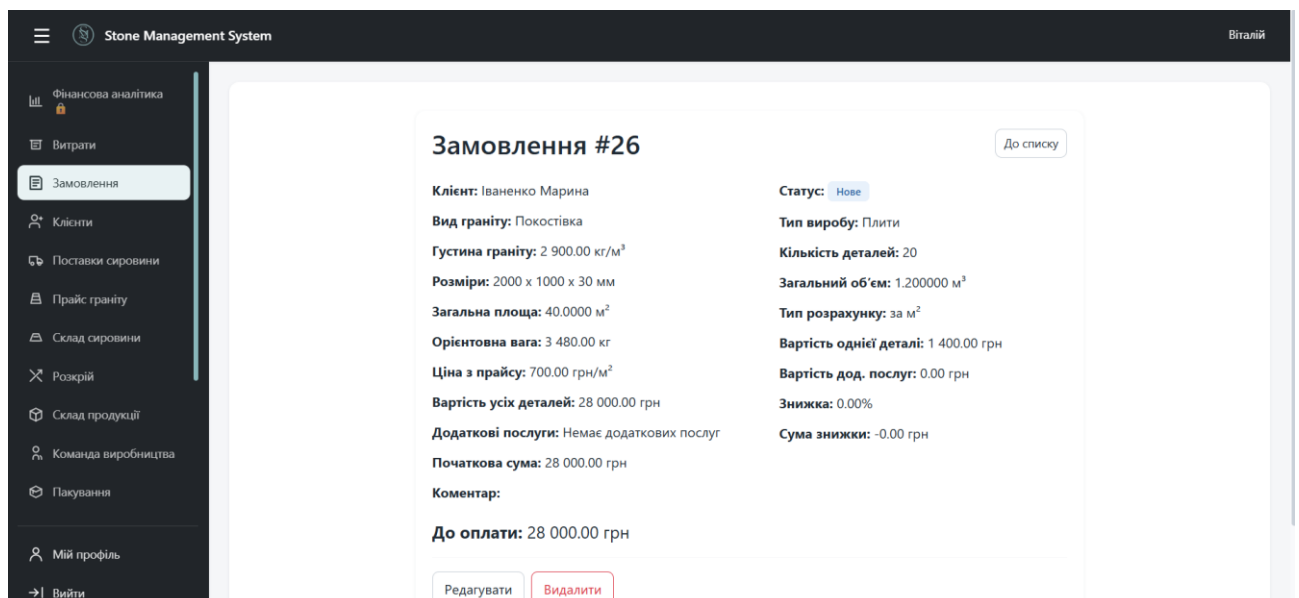


Рис. 3.5. Інтерфейс перегляду детальної інформації про замовлення

Модуль складу сировини призначений для обліку матеріалів, які перебувають на підприємстві. На сторінці складу сировини відображається

таблиця з інформацією про кожну одиницю матеріалу. Для сировини вказуються маркування, фотографія, назва граніту, тип матеріалу, статус, розміри, корисні розміри, наявність дефектів і джерело надходження. У верхній частині сторінки розміщено кнопки пошуку й фільтрації та додавання нового запису. Наявність фотографій у таблиці спрощує візуальну ідентифікацію матеріалу, а маркування дозволяє простежувати рух сировини в подальших виробничих операціях.

Сторінка інтерфейсу модуля обліку складу сировини показана на рис. 3.6.

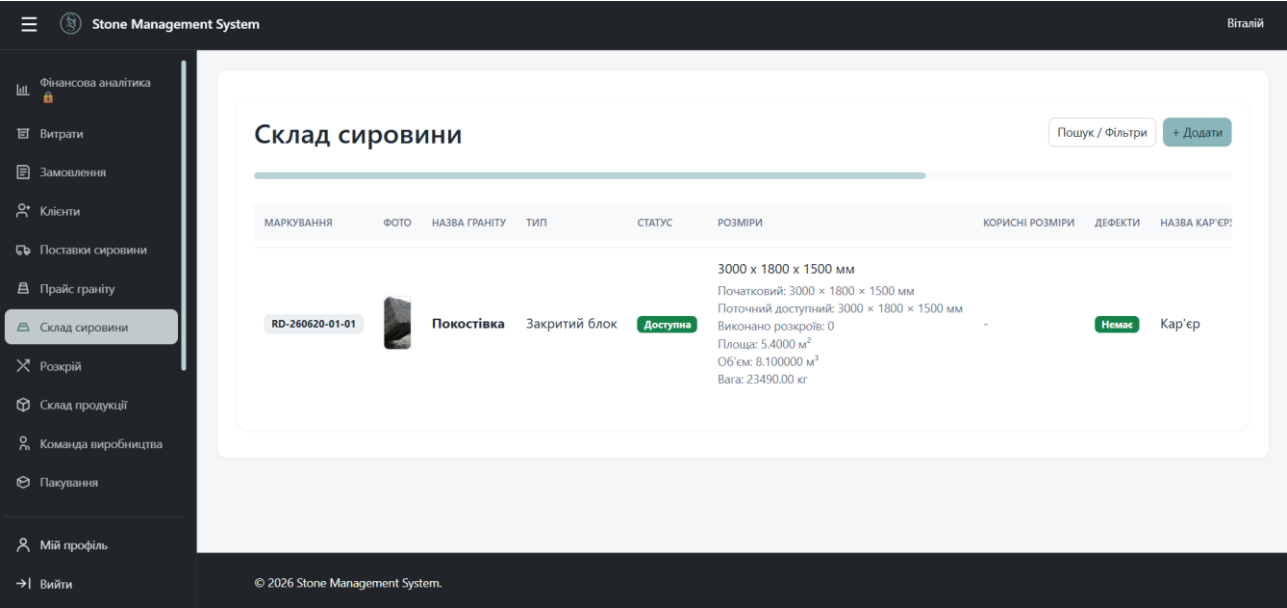


Рис. 3.6. Інтерфейс модуля обліку складу сировини

Модуль розкрою сировини використовується для фіксації виробничих операцій, під час яких блоки або заготовки перетворюються на деталі. На сторінці розкрою відображається список виконаних або запланованих операцій із зазначенням номера операції, дати, поставки, джерела матеріалу, розмірів джерела, початкового об’єму, кількості заготовок, об’єму заготовок і браку. У верхній частині сторінки передбачено фільтри та кнопку створення нового розкрою. Такий інтерфейс дозволяє контролювати використання сировини, обсяг отриманих деталей і виробничі втрати.

Сторінка інтерфейсу модуля розкрою сировини показана на рис. 3.7.

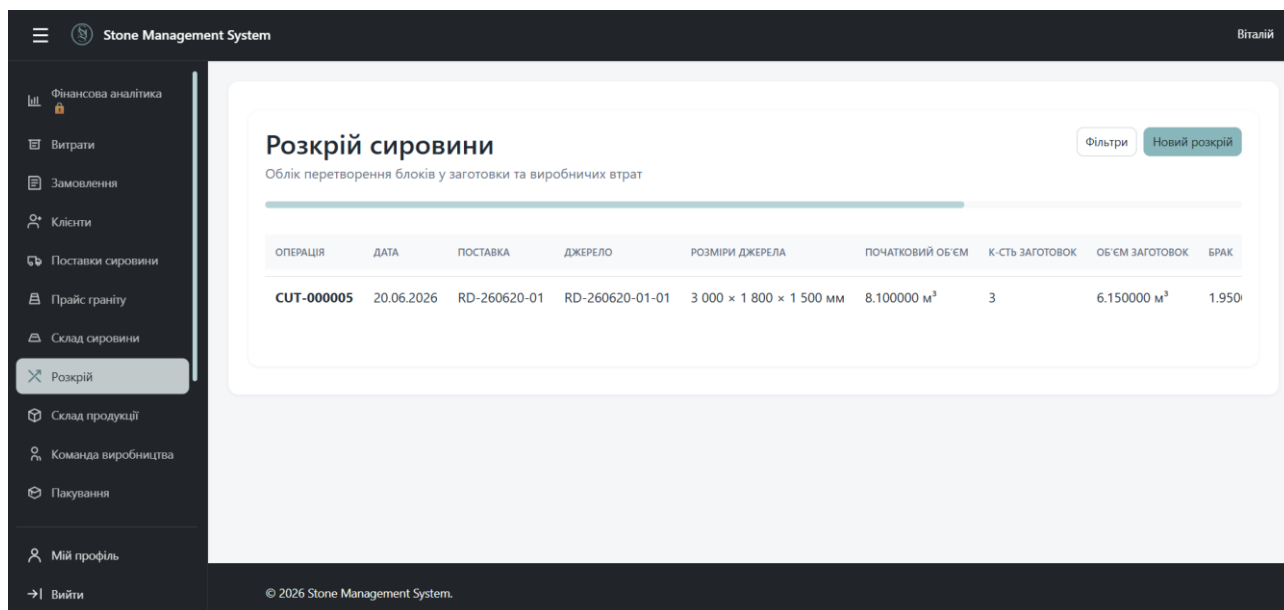


Рис. 3.7. Інтерфейс модуля розкрою сировини

Окрему частину інтерфейсу становлять робочі сторінки працівників. На прикладі сторінки полірування видно, що працівник має спрощений інтерфейс, який містить лише ті елементи, що потрібні для виконання конкретної виробничої операції. На сторінці відображаються дата виконання роботи, схема полірування, ставка за одиницю роботи, параметр повторного полірування, блок фіксації дефекту та список доступних деталей. Також передбачено пошук за маркуванням, назвою, гранітом або розмірами. Такий підхід дозволяє працівнику швидко знайти потрібну деталь і зафіксувати виконання роботи без доступу до зайвих адміністративних функцій.

Сторінка інтерфейсу робочого місця працівника для полірування деталей показана на рис. 3.8.

Після завершення виробничих операцій готові деталі відображаються в модулі складу продукції. Сторінка складу продукції містить таблицю з фотографіями, описом деталі, кількістю, маркуванням, типом обробки, станом поверхні та поточним статусом. Для кожної позиції можуть відображатися окремі маркування фізичних деталей, що дає змогу простежувати рух конкретних виробів у межах системи.

Замовлення

Склад сировини

Склад продукції

Пакування

Відправки

Маркування

Облік інструментів

Мое полірування

Моя зарплата

Мое полірування

Полірування конкретних промаркованих деталей

Дата

06/20/2026

Схема полірування

Одне лице

Ставка: **150.00 грн / м².** Нарахування буде створено автоматично за фактичною площею або годинами.

☐ Підтверджую повторне полірування, якщо у вибраних деталях уже є оброблені поверхні

Виявлено дефект

Ні

Дефект фіксується тільки для однієї вибраної фізичної деталі за її маркуванням.

Доступні деталі

Пошук за маркуванням, назвою, грантом або розмірами

Знайти

МАРКУВАННЯ	ТИП	ГРАНТ	РОЗМІРИ	ОБ'ЄКТА	СТАТУС	КЛІЄНТ / СКЛАД
------------	-----	-------	---------	---------	--------	----------------

Рис. 3.8. Інтерфейс робочого місця працівника для полірування деталей

У верхній частині сторінки розміщено кнопки фільтрації, додавання продукції на склад і створення нового пакування. Така структура інтерфейсу поєднує складський облік, виробничу інформацію та підготовку продукції до логістичних операцій.

Сторінка інтерфейсу модуля складу готової продукції показана на рис. 3.9.

Замовлення

Склад сировини

Склад продукції

Пакування

Відправки

Маркування

Облік інструментів

Мас. полірування

Моя зарплата

Мій профіль

Вийти

Stone Management System

Андрій

Склад продукції

Фізичні деталі лішаються окремими записами, групування тільки в інтерфейсі.

Фільтри

+ Додати на склад

Нове пакування

ФОТО	ДЕТАЛЬ	КІЛЬКІСТЬ	МАРКУВАННЯ	ОБРОБКА	ПОВЕРХНЯ	СТАТУС
	Деталь Граніт: Покостівка Розмір: 2000x1000x30 мм	10 шт. рахується за фізичними записами	RD-260620-01-01-01-21	Піляна	Піляна	На складі
			RD-260620-01-01-01-22			На складі
			RD-260620-01-01-01-23			На складі
			Показати всі			
	Деталь Граніт: Покостівка Розмір: 2000x1000x30 мм	20 шт. рахується за фізичними записами	RD-260620-01-01-01-01	Поліроване лице	Частково полірована	На складі
			RD-260620-01-01-01-02			На складі
			RD-260620-01-01-01-03			На складі
			Показати всі			

Рис. 3.9. Інтерфейс модуля складу готової продукції

Модуль пакування призначений для формування пачок або піддонів із готовою продукцією перед її подальшою відправкою клієнту. На сторінці пакування відображається таблиця, у якій для кожного запису зазначаються номер пакування, клієнт, тип піддона, габарити пачки, кількість деталей, вага та

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						80
Змн.	Арк.	№ докум.	Підпис	Дата		

поточний статус. У верхній частині сторінки розміщено кнопки пошуку й фільтрації, переходу до списку відправлень і створення нового пакування. Такий інтерфейс дозволяє контролювати комплектність пачок, вагу продукції та готовність сформованих пакувань до відправлення.

Сторінка інтерфейсу модуля пакування готової продукції показана на рис. 3.10.

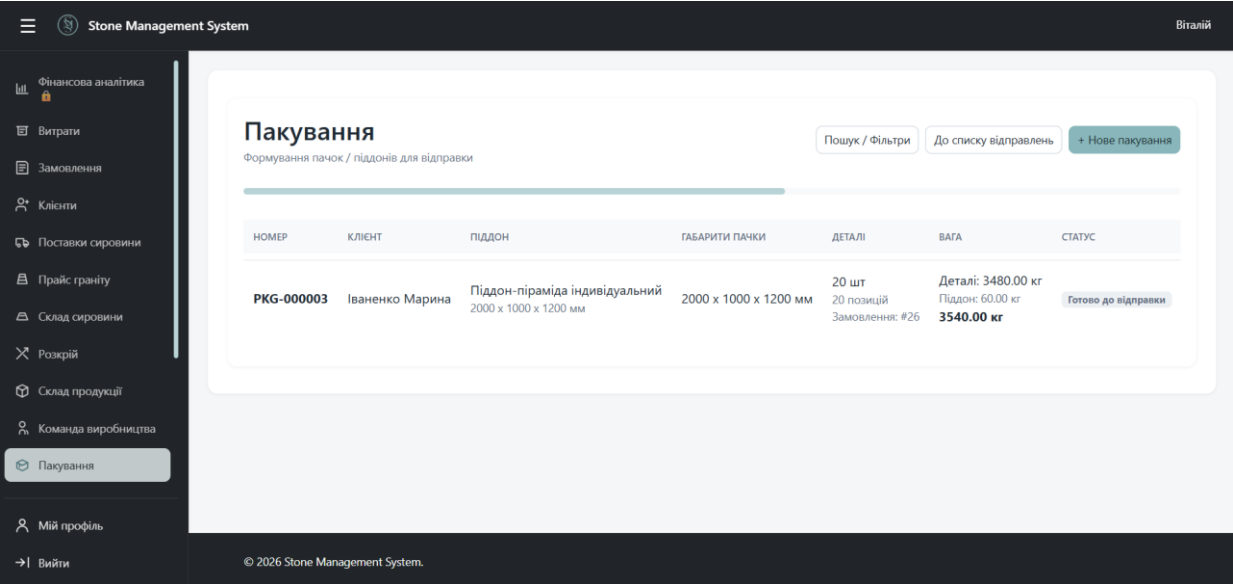


Рис. 3.10. Інтерфейс модуля пакування готової продукції

Після формування пачок або піддонів продукція передається до модуля відправок. Модуль відправок використовується для обліку передавання готової продукції клієнту. На сторінці відправок відображається таблиця з номером відправки, клієнтом, способом доставки, адресою, номером накладної, кількістю пачок, вагою, параметрами автомобіля, відсотком завантаження, статусом і датою створення запису. У верхній частині сторінки розміщено кнопки пошуку й фільтрації, переходу до пакування та створення нової відправки. Такий інтерфейс дозволяє контролювати завершальний етап роботи із замовленням – підготовку продукції до доставки та її фактичне відправлення клієнту.

Сторінка інтерфейсу модуля відправок показана на рис. 3.11.

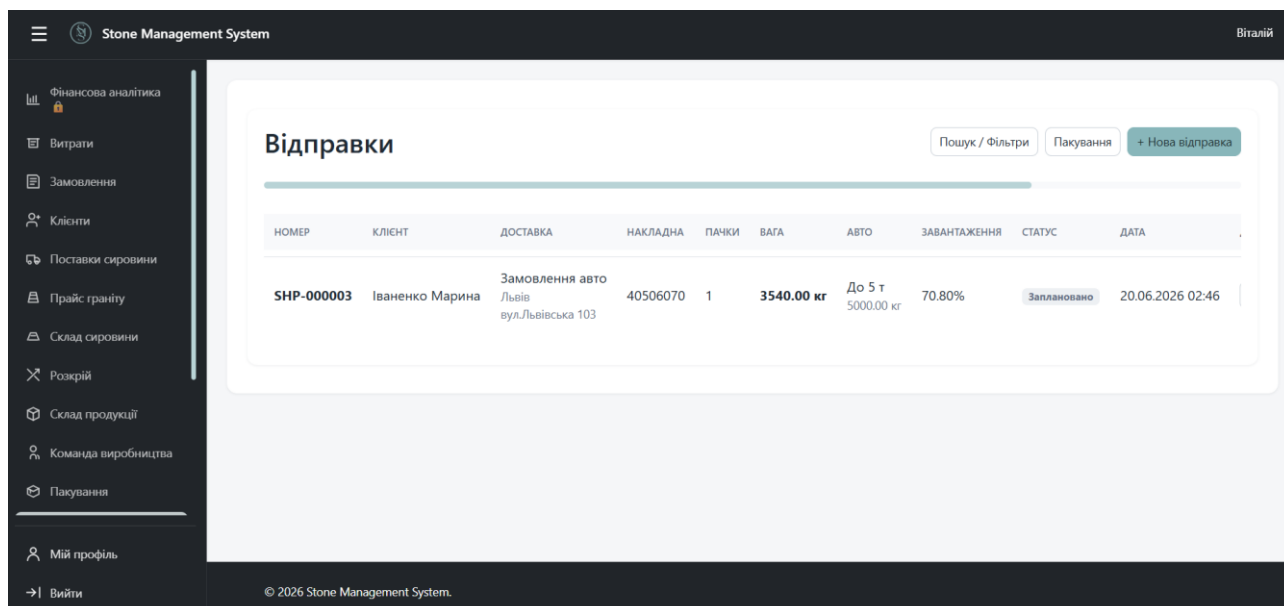


Рис. 3.11. Інтерфейс модуля відправок

Інтерфейс програмного комплексу адаптовано для використання на різних типах пристроїв [39]. На екранах персональних комп'ютерів основні елементи розміщуються у ширшій робочій області з боковим меню, таблицями та картками. На мобільних пристроях сторінки переходять до компактного вертикального подання, у якому основна інформація розміщується в одну колонку, а навігація доступна через кнопку меню. На прикладі мобільного вигляду профілю користувача видно, що сторінка зберігає читабельність: кнопки керування розміщені у верхній частині картки, а особисті дані користувача подаються послідовним списком. Це підтверджує можливість роботи із системою не лише з комп'ютера, а й з мобільного пристрою.

Мобільний вигляд сторінки профілю користувача показано на рис. 3.12.

Окрім наведених сторінок, у системі також реалізовано інші функціональні розділи: клієнти, прайс граніту, поставки сировини, команда виробництва, маркування, витрати, інструмент, зарплати, підписка та журнал дій.

Вони мають спільну структуру інтерфейсу: заголовок сторінки, кнопки основних дій, фільтри або пошук, таблиці записів, статуси та службові повідомлення. Така уніфікація спрощує роботу користувача, оскільки різні модулі побудовані за однаковими принципами навігації та взаємодії.

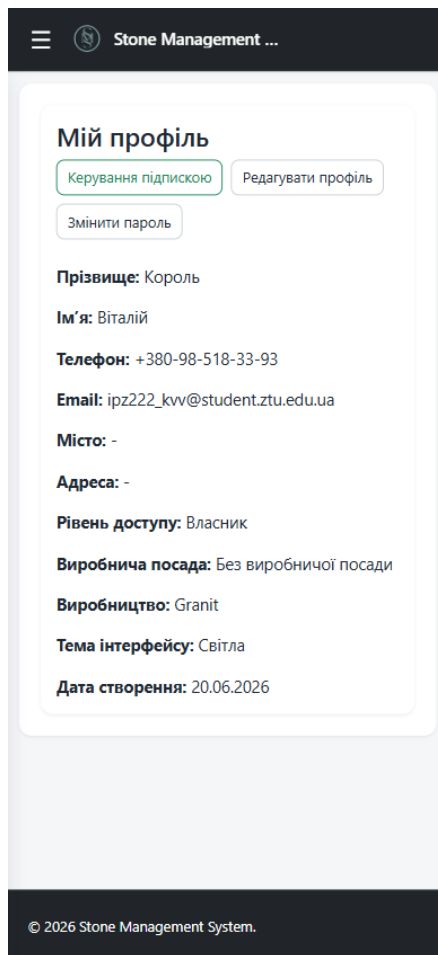


Рис. 3.12. Мобільний вигляд сторінки профілю користувача

Отже, інтерфейс системи автоматизації діяльності малого підприємства каменеобробної галузі забезпечує виконання основних сценаріїв роботи: авторизацію користувача, перегляд і створення замовлень, облік сировини, фіксацію розкрою, виконання виробничих операцій працівниками, облік готової продукції, формування пакувань, створення відправок, доступ до фінансової аналітики та роботу з мобільного пристрою. Побудова інтерфейсу на основі єдиного шаблону, бокового меню, таблиць, карток і фільтрів забезпечує цілісність програмного комплексу та зручність його використання різними категоріями користувачів.

3.3. Тестування роботи системи

Тестування системи автоматизації діяльності малого підприємства каменеобробної галузі виконувалося з метою перевірки коректності роботи основних функціональних модулів, відповідності програмного комплексу

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						83
Змн.	Арк.	№ докум.	Підпис	Дата		

визначеним сценаріям використання та виявлення помилок, які можуть виникати під час взаємодії користувача з інтерфейсом [40]. Основну увагу приділено сценаріям, які формують практичну цінність системи: авторизації користувачів, роботі з клієнтами, створенню замовлень, розрахунку вартості, веденню прайсу граніту, обліку сировини, розкрою, роботі працівників, складу готової продукції, пакуванню, відправкам, витратам, зарплатам, фінансовій аналітиці та журналюванню дій.

Перевірка проводилася методом функціонального тестування. Для кожного сценарію визначалися передумови, дії користувача, очікуваний результат і фактичний результат виконання. Такий підхід дозволяє перевірити не лише наявність окремої функції, а й правильність її роботи в межах загальної логіки програмного комплексу.

Під час тестування враховано рольову модель системи. Для адміністратора перевірялися сценарії повного керування модулями: клієнтами, замовленнями, сировиною, продукцією, прайсом граніту, пакуванням, відправками, працівниками, витратами, зарплатами та фінансовою аналітикою. Для працівника перевірялися виробничі сценарії, зокрема доступ до робочих сторінок, фіксація виконаної роботи та перегляд власних нарахунків. Для клієнта перевірявся обмежений доступ до інформації, що стосується його замовлень.

Результати функціонального тестування основних сценаріїв роботи програмного комплексу наведено в табл. 3.3.

Таблиця 3.3

Результати тестування основних функціональних сценаріїв системи

№	Сценарій тестування	Передумови та вхідні дані	Очікуваний результат	Фактичний результат	Статус
1	Авторизація користувача	Введено коректний email або телефон і пароль	Користувач входить у систему та отримує доступ відповідно до ролі	Вхід виконано, відкрито внутрішню частину системи	Пройдено

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		84

Продовження табл. 3.3

2	Невдала авторизація	Введено неправильний пароль	Система не виконує вхід і показує повідомлення про помилку	Авторизацію заблоковано, користувач отримує повідомлення про помилку	Пройдено
3	Перехід між модулями через бокове меню	Користувач авторизований у системі	Вибраний модуль відкривається в основній робочій області	Перехід між розділами виконується коректно	Пройдено
4	Перевірка обмеження доступу до фінансової аналітики	Аналітичний модуль не активовано через підписку	Система не відкриває аналітику та пропонує керування підпискою	Відображено повідомлення про недоступність модуля	Пройдено
5	Створення клієнта	Заповнено форму клієнта валідними даними	Дані клієнта зберігаються в базі та відображаються у списку	Новий клієнт створений і доступний для вибору в замовленні	Пройдено
6	Редагування клієнта	Відкрито існуючий запис клієнта	Змінені дані зберігаються та оновлюються в таблиці	Дані клієнта оновлено	Пройдено
7	Додавання позиції до прайсу граніту	Вказано назву граніту, ціну, густину та активність	Позиція зберігається в довіднику та доступна під час створення замовлення	Позиція прайсу додана й відображається в системі	Пройдено

8	Створення замовлення	Обрано клієнта, граніт, тип виробу, розміри, кількість і додаткові параметри	Система створює замовлення та розраховує основні показники	Замовлення збережено, параметри відображено в таблиці	Пройдено
9	Розрахунок вартості замовлення	У замовленні задано розміри, товщину, кількість і ціну з прайсу	Система визначає площу, об'єм, вагу, тип розрахунку і суму до оплати	Розрахункові дані сформовано та показано в деталях замовлення	Пройдено
10	Перегляд детальної інформації про замовлення	Відкрито існуюче замовлення	Користувач бачить клієнта, матеріал, розміри, статус, ціну, знижку і підсумкову суму	Детальна сторінка замовлення відображає повну інформацію	Пройдено
11	Додавання поставки сировини	Заповнено дані про постачальника, граніт, вартість, вагу та дату	Поставка створюється та може бути пов'язана із сировиною	Поставка збережена в системі	Пройдено
12	Додавання сировини на склад	Вказано маркування, тип матеріалу, розміри, статус, дефекти та фото	Запис сировини зберігається і відображається на складі	Сировина додана до складу, дані показано в таблиці	Пройдено

13	Завантаження фотографії сировини	До запису сировини додано зображення допустимого формату	Фото зберігається та відображається в інтерфейсі	Фото показано в таблиці складу сировини	Пройдено
14	Пошук і фільтрація складу сировини	Задано параметри пошуку або фільтрів	Система відображає записи, що відповідають умовам	Дані в таблиці оновлюються відповідно до фільтрів	Пройдено
15	Створення операції розкрою	Обрано вихідну сировину та задано параметри отриманих деталей	Система фіксує операцію, обсяг заготовок і виробничі втрати	Операцію розкрою створено, дані відображено в модулі	Пройдено
16	Перегляд списку операцій розкрою	У системі є створені операції розкрою	Користувач бачить номер операції, дату, джерело, об'єми, кількість заготовок і брак	Таблиця операцій розкрою відображається коректно	Пройдено
17	Робота працівника з виробничою сторінкою	Працівник авторизований і має відповідну виробничу роль	Працівник бачить лише доступні йому виробничі дії	Відкрито робочу сторінку полірування з потрібними елементами	Пройдено
18	Фіксація параметрів полірування	На сторінці працівника обрано дату, схему полірування та деталі	Система дозволяє зафіксувати виконану виробничу операцію	Параметри роботи доступні для внесення та подальшого обліку	Пройдено

19	Перегляд складу готової продукції	У системі є готові деталі після виробничих операцій	Сторінка показує фото, деталі, кількість, маркування, обробку, поверхню і статус	Склад продукції відображає готові вироби та їхні параметри	Пройдено
20	Створення пакування	Обрано продукцію для пакування та задано параметри пачки або піддона	Система формує запис пакування з деталями, вагою і статусом	Пакування створено та відображено в таблиці	Пройдено
21	Перегляд пакування	Відкрито список пакувань	Користувач бачить номер, клієнта, піддон, габарити, кількість деталей, вагу і статус	Дані пакування відображено коректно	Пройдено
22	Створення відправки	Обрано клієнта, пакування, спосіб доставки та транспортні параметри	Система створює запис відправки і прив'язує до нього пакування	Відправку створено, статус і вага відображаються в таблиці	Пройдено

23	Перегляд списку відправок	У системі є сформовані відправки	Користувач бачить номер, клієнта, доставку, накладну, вагу, завантаження, статус і дату	Таблиця відправок відображає логістичні дані	Пройдено
24	Додавання витрати	Заповнено категорію, суму, дату й опис витрати	Витрата зберігається і враховується у фінансовому обліку	Запис витрати створено	Пройдено
25	Перегляд зарплатного модуля	У системі є працівники, ставки або записи виконаних робіт	Система відображає дані для розрахунку оплати праці	Зарплатний модуль доступний і показує облікові дані	Пройдено
26	Перевірка журналу дій	Користувач виконує важливу дію в системі	Дія фіксується в журналі з датою, користувачем і типом операції	Запис події доступний у журналі дій	Пройдено
27	Перевірка рольового доступу працівника	Авторизовано користувача з роллю працівника	Працівнику доступні лише відповідні виробничі модулі	Адміністративні розділи обмежено, робочі сторінки доступні	Пройдено
28	Перевірка адаптивності інтерфейсу	Сторінку відкрито на мобільному екрані або через DevTools	Інтерфейс переходить до компактного подання, дані залишаються читабельними	Мобільний вигляд сторінки відображається коректно	Пройдено

Окремо було перевірено обмеження функціональності, які важливі для коректної роботи системи. Система не тестувалась як інтернет-магазин із кошиком, онлайн-оплатою товарів або автоматичною доставкою через API, оскільки така модель не є основою розробленого програмного продукту. Платіжна логіка перевірялася саме в межах доступу до платного аналітичного функціоналу, тобто фінансової аналітики за підпискою.

Під час тестування особливу увагу приділено сценаріям, які залежать від серверного середовища та зовнішніх сервісів. До таких сценаріїв належать надсилання листів через SMTP, авторизація через Google OAuth, перевірка доступу до платного функціоналу, завантаження фотографій, робота сесій користувачів і коректне відображення адаптивного інтерфейсу. Для стабільної роботи цих сценаріїв необхідні коректні конфігураційні параметри, доступність зовнішніх сервісів, права запису до директорій завантажень і коректне підключення до бази даних.

Окрім функціонального тестування, було виконано додаткову технічну перевірку вебзастосунку в середовищі розгортання. Її метою було підтвердити, що система працює не лише на рівні окремих функцій, а й як вебсистема, доступна користувачу через браузер. Додаткова перевірка охоплювала доступність сторінок, роботу маршрутизації, підключення до бази даних, сесії користувачів, клієнтські сценарії, завантаження файлів і адаптивність інтерфейсу. Результати додаткової технічної перевірки наведено в табл. 3.4.

Таблиця 3.4

Додаткова технічна перевірка вебзастосунку

№	Вид перевірки	Інструмент або спосіб перевірки	Очікуваний результат	Статус
1	Доступність сторінок системи	Відкриття основних маршрутів у браузері	Основні сторінки системи відкриваються без критичних помилок	Пройдено

Продовження табл. 3.4

2	Робота маршрутизації	Перехід між модулями через бокове меню	Кожен пункт меню відкриває відповідний модуль	Пройдено
3	Підключення до бази даних	Виконання операцій створення, перегляду та редагування записів	Дані зберігаються й відображаються з MySQL	Пройдено
4	Робота сесій	Авторизація, перехід між сторінками, вихід із системи	Сесія користувача зберігається до виходу з системи	Пройдено
5	Клієнтські помилки	Перевірка консолі браузера	Відсутні критичні JavaScript-помилки, які блокують роботу сторінок	Пройдено
6	Завантаження фотографій	Додавання фото до сировини або продукції	Зображення зберігаються та відображаються в інтерфейсі	Пройдено
7	Адаптивність інтерфейсу	Перевірка сторінок через DevTools або мобільний пристрій	Основні сторінки коректно відображаються на екранах різної ширини	Пройдено
8	Перевірка прав доступу	Вхід під адміністратором і працівником	Набір доступних модулів змінюється відповідно до ролі користувача	Пройдено
9	Доступ до платного модуля	Відкриття фінансової аналітики без активної підписки	Система показує повідомлення про обмежений доступ	Пройдено
10	Робота файлової підсистеми	Завантаження та перегляд файлів у відповідних модулях	Файли зберігаються у визначених директоріях і доступні в інтерфейсі	Пройдено

Результати додаткової технічної перевірки показали, що система може використовуватися як повноцінна вебсистема у локальному або серверному середовищі. Перевірка підтвердила доступність основних сторінок, коректну

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		91

роботу маршрутизації, підключення до бази даних, сесій користувачів, файлової підсистеми, адаптивного інтерфейсу та рольового доступу.

За результатами функціонального й технічного тестування встановлено, що основні сценарії системи автоматизації діяльності малого каменеобробного підприємства працюють відповідно до визначених вимог. Реалізовані модулі забезпечують авторизацію користувачів, керування клієнтами, створення й супровід замовлень, розрахунок вартості, облік сировини, фіксацію операцій розкрою, роботу працівників із виробничими сторінками, облік готової продукції, формування пакувань, створення відправок, облік витрат, розрахунок зарплат, перевірку доступу до фінансової аналітики, журналювання дій та адаптивне відображення інтерфейсу. Отримані результати підтверджують працездатність програмного комплексу та можливість його використання для автоматизації основних процесів малого підприємства каменеобробної галузі.

Висновки до розділу 3

У третьому розділі було розглянуто порядок розгортання, структуру інтерфейсу та результати тестування системи автоматизації діяльності малого підприємства каменеобробної галузі. Описано основні вимоги до середовища виконання програмного комплексу, зокрема використання PHP, вебсервера, СКБД MySQL або MariaDB, файлової підсистеми, конфігураційних файлів і зовнішніх сервісів.

У межах підрозділу, присвяченого розгортанню системи, визначено основні етапи підготовки програмного комплексу до роботи. До них належать розміщення файлів проєкту, створення та налаштування бази даних, імпорт структури таблиць, налаштування підключення до MySQL, перевірка прав доступу до директорій завантажень, налаштування поштового сервісу, зовнішньої авторизації та платного аналітичного функціоналу. Також було наведено основні параметри конфігурації системи та схему ресурсів програмного комплексу.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						92
Змн.	Арк.	№ докум.	Підпис	Дата		

Під час розгляду інтерфейсу системи було показано, що система має єдину структуру взаємодії з користувачем, яка складається з верхньої панелі, бокового навігаційного меню та основної робочої області. Описано сторінки авторизації, фінансової аналітики з обмеженням доступу, модуля замовлень, детального перегляду замовлення, складу сировини, розкрою, робочого місця працівника, складу готової продукції, пакування, відправок і мобільного вигляду профілю користувача. Такий набір інтерфейсних екранів демонструє основний виробничий цикл підприємства: від створення замовлення та обліку сировини до виготовлення продукції, пакування і відправлення клієнту.

У процесі тестування було перевірено основні функціональні сценарії роботи програмного комплексу. Зокрема, перевірено авторизацію користувачів, рольовий доступ, роботу з клієнтами, прайсом граніту, замовленнями, поставками сировини, складом сировини, розкромом, виробничими операціями працівників, складом продукції, пакуванням, відправками, витратами, зарплатами, фінансовою аналітикою та журналом дій. Для кожного сценарію було визначено передумови, очікуваний результат, фактичний результат і статус виконання.

Окрім функціонального тестування, було виконано додаткову технічну перевірку вебзастосунку. Було перевірено доступність основних сторінок, роботу маршрутизації, підключення до бази даних, сесії користувачів, файлову підсистему, адаптивність інтерфейсу, права доступу та обмеження платного функціоналу. Результати перевірки підтвердили, що система коректно працює як вебсистема і може використовуватися в локальному або серверному середовищі.

Отже, у третьому розділі підтверджено практичну працездатність розробленої системи. Вебсистема має зрозумілий інтерфейс, підтримує рольовий доступ, забезпечує виконання основних бізнес-процесів каміньоробного підприємства та проходить функціональну й технічну перевірку за ключовими сценаріями.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						93
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

У результаті виконання випускної кваліфікаційної роботи було розроблено вебсистему автоматизації діяльності малого підприємства каменеобробної галузі. Розроблена система призначена для централізованого обліку клієнтів, замовлень, сировини, готової продукції, працівників, прайсу граніту, пакування, відправок, витрат, зарплат і фінансових показників діяльності підприємства.

У першому розділі було виконано аналіз предметної області та визначено основні проблеми, які виникають під час ведення обліку на малому каменеобробному підприємстві за допомогою розрізаних файлів, таблиць або паперових записів. Встановлено, що для такої предметної області недостатньо лише класичного CRM-функціоналу, оскільки підприємство потребує автоматизації не тільки взаємодії з клієнтами, а й виробничих, складських, логістичних та фінансово-облікових процесів. Було проаналізовано програмні аналоги, зокрема CRM-рішення, орієнтовані на роботу з клієнтами, угодами, задачами та продажами. За результатами аналізу визначено, що розглянуті системи не забезпечують повної підтримки специфічних процесів каменеобробного підприємства, таких як облік гранітної сировини, готової продукції, фізичних параметрів матеріалів, розкрою, пакування, відправок і розрахунку заробітної плати працівників.

У процесі обґрунтування технічних рішень було обрано клієнт-серверну веборієнтовану архітектуру з використанням архітектурного підходу MVC. Такий підхід дав змогу розділити систему на логічні частини: моделі, контролери, представлення, сервіси, клієнтські сценарії та стилі. Для реалізації серверної частини використано PHP, для роботи з базою даних –MySQL, для формування інтерфейсу – HTML, CSS і JavaScript. Обрані інструментальні засоби забезпечують можливість створення вебзастосунку з централізованим збереженням даних, рольовим доступом і подальшим розширенням функціональності.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						94
Змн.	Арк.	№ докум.	Підпис	Дата		

У другому розділі було виконано проектування програмного комплексу. Визначено основні категорії користувачів системи: адміністратора, працівника та клієнта. Для кожної категорії встановлено основні права доступу та сценарії взаємодії із системою. Центральною сутністю програмного комплексу визначено замовлення, оскільки саме воно поєднує клієнта, матеріал, параметри виробу, розрахунок вартості, статус виконання, виробничі операції та подальшу відправку готової продукції.

Було спроектовано структуру бази даних, яка забезпечує збереження інформації про користувачів, клієнтів, замовлення, прайс граніту, поставки сировини, склад сировини, готову продукцію, фотографії, маркування, пакування, відправки, зарплати, витрати, підписки та журнал дій. Використання реляційної моделі даних дозволило впорядкувати основні сутності предметної області та забезпечити логічні зв'язки між модулями системи.

Також було розроблено алгоритми роботи системи, які відображають основний цикл діяльності каменюобробного підприємства: авторизацію користувача, створення клієнта й замовлення, надходження сировини, класифікацію та маркування матеріалу, розкрій, виробничу обробку деталей, переміщення готової продукції на склад, формування пакування та створення відправки клієнту. Окремо враховано алгоритми обліку витрат, фіксації виконаних робіт, розрахунку заробітної плати та перевірки доступу до фінансової аналітики.

У межах реалізації функціональних модулів було створено складові, які забезпечують роботу основних процесів підприємства. До них належать модулі авторизації, клієнтів, команди виробництва, замовлень, прайсу граніту, поставок сировини, складу сировини, розкрою, робочих сторінок працівників, складу готової продукції, фотографій, маркування, пакування, відправок, зарплат, витрат, інструменту, фінансової аналітики, підписки на платний функціонал і журналювання дій. Використання окремих сервісів дозволило винести складну бізнес-логіку з контролерів і спростити супровід програмного коду.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						95
Змн.	Арк.	№ докум.	Підпис	Дата		

У третьому розділі було описано порядок розгортання, структуру інтерфейсу та результати тестування програмного комплексу. Для запуску системи визначено необхідні компоненти середовища: вебсервер, PHP, MySQL або MariaDB, конфігураційні файли, файлову підсистему, поштовий сервіс, зовнішню авторизацію та платіжний сервіс для доступу до платної фінансової аналітики. Також описано основні параметри конфігурації та порядок налаштування прав доступу.

Інтерфейс системи реалізовано як веборієнтований інтерфейс із верхньою панеллю, боковим навігаційним меню та основною робочою областю. Було показано сторінки авторизації, фінансової аналітики з обмеженим доступом, модуля замовлень, детального перегляду замовлення, складу сировини, розкрою, робочого місця працівника, складу готової продукції, пакування, відправок і мобільного вигляду профілю користувача. Це підтверджує, що система має цілісну структуру інтерфейсу та підтримує виконання основних сценаріїв роботи підприємства.

Під час тестування було перевірено основні функціональні сценарії системи: авторизацію користувачів, рольовий доступ, роботу з клієнтами, прайсом граніту, замовленнями, поставками сировини, складом сировини, розкромом, виробничими сторінками працівників, складом продукції, пакуванням, відправками, витратами, зарплатами, фінансовою аналітикою та журналом дій. Також було виконано додаткову технічну перевірку вебзастосунку, зокрема доступність сторінок, роботу маршрутизації, підключення до бази даних, сесії користувачів, файлову підсистему, адаптивність інтерфейсу та обмеження доступу до платного функціоналу.

Отримані результати підтверджують, що розроблена система відповідає поставленій меті та забезпечує автоматизацію основних процесів малого підприємства каміньогрунтової галузі. Вебсистема дозволяє централізовано зберігати дані, зменшити кількість ручних операцій, підвищити точність обліку, спростити контроль виконання замовлень, забезпечити простежуваність

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						96
Змн.	Арк.	№ докум.	Підпис	Дата		

сировини та готової продукції, а також підтримати облік витрат, зарплат і відправок.

Практичне значення розробленої системи полягає в тому, що вона може бути використана як основа для впровадження цифрового обліку на малому каменеобробному підприємстві. Подальший розвиток програмного комплексу може передбачати розширення аналітичного модуля, формування звітів, експорт даних, удосконалення клієнтського кабінету, інтеграцію з додатковими платіжними або логістичними сервісами та розширення функцій виробничого планування.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						97
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Король В. В., Кушнір Н. О., Локтікова Т. М. Побудова та проєктування вебсистеми управління взаємовідносинами з клієнтами для малого бізнесу : тези Всеукраїнської науково-практичної on-line конференції здобувачів вищої освіти і молодих учених, присвяченої Дню науки, м. Житомир, 11–15 травня 2026 р. Житомир : Державний університет «Житомирська політехніка», 2026. С. 90–91.
2. Buttle F., Maklan S. Customer Relationship Management: Concepts and Technologies. 4th ed. London : Routledge, 2019. 468 p.
3. Dumas M., La Rosa M., Mendling J., Reijers H. A. Fundamentals of Business Process Management. 2nd ed. Berlin : Springer, 2018. 527 p.
4. Weske M. Business Process Management: Concepts, Languages, Architectures. 3rd ed. Berlin : Springer, 2019. 426 p.
5. Uspacy. CRM для управління продажами та взаємодії з клієнтами. URL: <https://uspacy.ua/features/crm> (дата звернення: 20.06.2026).
6. Perfex CRM. Customer Relationship & Project Management. URL: <https://www.perfexcrm.com> (дата звернення: 20.06.2026).
7. Pipedrive. What is Pipedrive CRM. URL: <https://www.pipedrive.com/en/what-is-pipedrive-crm> (дата звернення: 20.06.2026).
8. Sommerville I. Software Engineering. 10th ed. Boston : Pearson, 2016. 796 p.
9. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner's Approach. 9th ed. New York : McGraw-Hill Education, 2019. 1408 p.
10. Bass L., Clements P., Kazman R. Software Architecture in Practice. 4th ed. Boston : Addison-Wesley, 2021. 464 p.
11. Richards M., Ford N. Fundamentals of Software Architecture: An Engineering Approach. Sebastopol : O'Reilly Media, 2020. 432 p.
12. Fowler M. Patterns of Enterprise Application Architecture. Boston : Addison-Wesley, 2002. 560 p.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						98
Змн.	Арк.	№ докум.	Підпис	Дата		

13. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Boston : Addison-Wesley, 1994. 416 p.
14. PHP Documentation. PHP Manual. URL: <https://www.php.net/manual/en> (дата звернення: 20.06.2026).
15. MySQL. MySQL Reference Manual. URL: <https://dev.mysql.com/doc/mysql/en> (дата звернення: 20.06.2026).
16. MDN Web Docs. Web technology for developers. URL: <https://developer.mozilla.org/en-US/docs/Web> (дата звернення: 20.06.2026).
17. Apache Software Foundation. Apache HTTP Server Version 2.4 Documentation. URL: <https://httpd.apache.org/docs/2.4> (дата звернення: 20.06.2026).
18. phpMyAdmin. Official phpMyAdmin Documentation. URL: <https://www.phpmyadmin.net/docs> (дата звернення: 20.06.2026).
19. JetBrains. PhpStorm Documentation. URL: <https://www.jetbrains.com/help/phpstorm/getting-started.html> (дата звернення: 20.06.2026).
20. ISO/IEC/IEEE 29148:2018. Systems and software engineering — Life cycle processes — Requirements engineering. Geneva : ISO, 2018. 92 p.
21. ISO/IEC 25010:2011. Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models. Geneva : ISO, 2011. 34 p.
22. Fowler M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3rd ed. Boston : Addison-Wesley, 2003. 208 p.
23. Larman C. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. 3rd ed. Upper Saddle River : Prentice Hall, 2004. 736 p.
24. Booch G., Rumbaugh J., Jacobson I. The Unified Modeling Language User Guide. 2nd ed. Boston : Addison-Wesley, 2005. 475 p.
25. Silberschatz A., Korth H. F., Sudarshan S. Database System Concepts. 7th ed. New York : McGraw-Hill Education, 2019. 1376 p.

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						99
Змн.	Арк.	№ докум.	Підпис	Дата		

- 26.Connolly T., Begg C. Database Systems: A Practical Approach to Design, Implementation, and Management. 6th ed. Harlow : Pearson, 2015. 1440 p.
- 27.MySQL Documentation. The InnoDB Storage Engine. URL: <https://dev.mysql.com/doc/refman/en/innodb-storage-engine.html> (дата звернення: 20.06.2026).
- 28.Beaulieu A. Learning SQL. 3rd ed. Sebastopol : O'Reilly Media, 2020. 380 p.
- 29.PHP Documentation. PDO — PHP Data Objects. URL: <https://www.php.net/manual/en/book.pdo.php> (дата звернення: 20.06.2026).
- 30.PHP Documentation. Sessions. URL: <https://www.php.net/manual/en/book.session.php> (дата звернення: 20.06.2026).
- 31.PHP Documentation. Handling file uploads. URL: <https://www.php.net/manual/en/features.file-upload.php> (дата звернення: 20.06.2026).
- 32.OWASP Foundation. OWASP Top Ten Web Application Security Risks. URL: <https://owasp.org/www-project-top-ten> (дата звернення: 20.06.2026).
- 33.OWASP Foundation. Cross-Site Request Forgery Prevention Cheat Sheet. URL: https://cheatsheetseries.owasp.org/cheatsheets/Cross-Site_Request_Forgery_Prevention_Cheat_Sheet.html (дата звернення: 20.06.2026).
- 34.Klensin J. Simple Mail Transfer Protocol : RFC 5321. Internet Engineering Task Force, 2008. URL: <https://www.rfc-editor.org/rfc/rfc5321> (дата звернення: 20.06.2026).
- 35.Google Developers. Using OAuth 2.0 for Web Server Applications. URL: <https://developers.google.com/identity/protocols/oauth2/web-server> (дата звернення: 20.06.2026).
- 36.monobank. Acquiring API Documentation. URL: <https://monobank.ua/api-docs/acquiring> (дата звернення: 20.06.2026).
- 37.Nielsen J. Usability Engineering. San Francisco : Morgan Kaufmann, 1993. 362 p.

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						100
Змн.	Арк.	№ докум.	Підпис	Дата		

- 38.Krug S. Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability. 3rd ed. Berkeley : New Riders, 2014. 200 p.
- 39.Marcotte E. Responsive Web Design. New York : A Book Apart, 2011. 153 p.
- 40.Myers G. J., Sandler C., Badgett T. The Art of Software Testing. 3rd ed. Hoboken : Wiley, 2011. 256 p.

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						101
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						102
Змн.	Арк.	№ докум.	Підпис	Дата		

ПРОГРАМНИЙ КОД ОСНОВНИХ КОМПОНЕНТІВ МОДУЛІВ СИСТЕМИ

А.1 Розширений лістинг програмного коду модуля замовлень

У цьому підрозділі наведено програмний код основних компонентів модуля замовлень, який відповідає за отримання детальної інформації про замовлення, розрахунок його вартості, застосування знижки та підготовку фінансових показників для відображення користувачу. Наведений код демонструє, як у системі централізовано реалізовано розрахункову логіку на рівні моделі Order.

```
<?php
namespace MVC\models;
use core\Core;
use core\Logger;
use core\Model;
use core\TenantContext;
class Order extends Model
{
    public static $tableName = 'orders';
    public static function getStatuses()
    {
        return [
            'pending_confirmation' => 'Очікує підтвердження',
            'new' => 'Нове',
            'in_work' => 'В роботі',
            'completed' => 'Виконане',
            'packed' => 'Запаковане',
            'in_stock' => 'На складі',
            'cancelled' => 'Скасоване',
        ];
    }
    public static function getAllowedStatuses()
    {

```

```

        return array_keys(self::getStatuses());
    }
    public static function isAllowedStatus($status)
    {
        return in_array($status, self::getAllowedStatuses(), true);
    }
    public static function getStatusAliases()
    {
        return [
            'pending_confirmation' => ['pending_confirmation', 'Очікує підтвердження'],
            'new' => ['new', 'Нове'],
            'in_work' => ['in_work', 'in_progress', 'active', 'В роботі'],
            'completed' => ['completed', 'done', 'finished', 'complete', 'Виконано', 'Виконане',
'Завершене'],
            'packed' => ['packed', 'packaged', 'Запаковане', 'Запаковано'],
            'in_stock' => ['in_stock', 'stock', 'in_storage', 'На складі'],
            'cancelled' => ['cancelled', 'canceled', 'Скасоване', 'Скасовано'],
        ];
    }
    public static function normalizeStatus($status, $default = 'new')
    {
        $status = trim((string)$status);
        if ($status === "") {
            return $default;
        }
        foreach (self::getStatusAliases() as $normalized => $aliases) {
            if (in_array($status, $aliases, true)) {
                return $normalized;
            }
        }
        return array_key_exists($status, self::getStatuses()) ? $status : $default;
    }
    public static function getStatusLabel($status)
    {
        $normalized = self::normalizeStatus($status, (string)$status);

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						104
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    $labels = self::getStatuses();
    return $labels[$normalized] ?? ($status ? : $labels['new']);
}

public static function getStatusClass($status)
{
    return 'status-' . str_replace('_', '-', self::normalizeStatus($status));
}

public static function getStatusBySearchQuery($query)
{
    $query = self::normalizeSearchText($query);
    if ($query === "") {
        return null;
    }
    foreach (self::getStatuses() as $status => $label) {
        if ($query === self::normalizeSearchText($status) || $query ===
self::normalizeSearchText($label)) {
            return $status;
        }
    }
}

$aliases = [
    'новий' => 'new',
    'нова' => 'new',
    'в роботі' => 'in_work',
    'у роботі' => 'in_work',
    'в роботі' => 'in_work',
    'у роботі' => 'in_work',
    'виконано' => 'completed',
    'завершене' => 'completed',
    'завершено' => 'completed',
    'запаковане' => 'packed',
    'запаковано' => 'packed',
    'склад' => 'in_stock',
    'на складі' => 'in_stock',
    'на складі' => 'in_stock',
    'скасовано' => 'cancelled',

```

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						105
Змн.	Арк.	№ докум.	Підпис	Дата		

```

];
return $aliases[$query] ?? null;
}
private static function normalizeSearchText($value)
{
    $value = trim((string)$value);

    return function_exists('mb_strtolower')
        ? mb_strtolower($value, 'UTF-8')
        : strtolower($value);
}
public static function getList($filters = [], $user = null)
{
    $where = [];
    $params = ['production_id' => TenantContext::requireProductionId()];
    $canSeeFinancials = Users::CanSeeFinancials($user);
    if (TenantContext::tableHasColumn('orders', 'production_id')) {
        $where[] = 'o.production_id = :production_id';
    } else {
        $where[] = 'c.production_id = :production_id';
    }
    if (!empty($filters['status'])) {
        $status = self::normalizeStatus($filters['status']);
        if (self::isAllowedStatus($status)) {
            $where[] = 'o.status = :status';
            $params[':status'] = $status;
        }
    }
    if (!empty($filters['client_id'])) {
        $where[] = 'o.client_id = :client_id';
        $params[':client_id'] = $filters['client_id'];
    }
    if (!empty($filters['granite_type_id'])) {
        $where[] = 'o.granite_type_id = :granite_type_id';
        $params[':granite_type_id'] = (int)$filters['granite_type_id'];
    }
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						106
Змн.	Арк.	№ докум.	Підпис	Дата		

```

}
if (isset($filters['length']) && trim((string)$filters['length']) !== "") {
    $where[] = 'o.length = :length';
    $params[':length'] = (float)$filters['length'];
}
if (isset($filters['width']) && trim((string)$filters['width']) !== "") {
    $where[] = 'o.width = :width';
    $params[':width'] = (float)$filters['width'];
}
if (isset($filters['thickness']) && trim((string)$filters['thickness']) !== "") {
    $where[] = 'o.thickness = :thickness';
    $params[':thickness'] = (float)$filters['thickness'];
}
if (self::isValidDate($filters['date_from'] ?? "")) {
    $where[] = 'o.created_at >= :date_from';
    $params[':date_from'] = $filters['date_from'] . ' 00:00:00';
}
if (self::isValidDate($filters['date_to'] ?? "")) {
    $where[] = 'o.created_at < DATE_ADD(:date_to, INTERVAL 1 DAY)';
    $params[':date_to'] = $filters['date_to'];
}
if (!empty($filters['q'])) {
    $qWhere = "(
        CAST(o.id AS CHAR) LIKE :q
        OR o.material LIKE :q
        OR o.extra_services LIKE :q
        OR o.notes LIKE :q
        OR g.name LIKE :q
        OR c.first_name LIKE :q
        OR c.last_name LIKE :q
        OR c.phone_number LIKE :q
        OR c.email LIKE :q
    ";
    if ($scanSeeFinancials) {
        $qWhere .= " OR CAST(o.total_price AS CHAR) LIKE :q";
    }
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						107
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
    $params[':q'] = '%' . trim($filters['q']) . '%';
    $qStatus = self::getStatusBySearchQuery($filters['q']);
    if ($qStatus !== null) {
        $qWhere .= " OR o.status = :q_status";
        $params[':q_status'] = $qStatus;
    }
    $where[] = $qWhere . ')';
}

if ($user && strtolower($user->role) === 'client') {
    $where[] = '(o.client_id = :current_client_id OR o.client_user_id =
:current_client_user_id)';
    $params[':current_client_id'] = (int)(Users::getCurrentClientId($user) ?? 0);
    $params[':current_client_user_id'] = (int)($user->id ?? 0);
}

$orderFields = $scanSeeFinancials
? 'o.*'
: 'o.id, o.client_id, o.client_user_id, o.material, o.granite_type_id,
o.length, o.width, o.thickness, o.quantity, o.area_m2, o.volume_m3,
o.product_type, o.extra_services, o.notes, o.single_detail_price,
o.detail_price, o.extra_services_price, o.discount_percent, o.total_price,
o.calculation_type, o.calculation_unit, o.status, o.created_at';

$sql = "SELECT {$orderFields}, c.first_name AS client_first_name, c.last_name AS
client_last_name,
        COALESCE(g.name, NULLIF(TRIM(o.material), '')) AS granite_name,
        g.density_kg_m3 AS granite_density_kg_m3,
        u.first_name AS employee_first_name, u.last_name AS employee_last_name
FROM orders o
LEFT JOIN clients c
    ON c.id = o.client_id
AND c.production_id = o.production_id
LEFT JOIN granite_types g
    ON g.id = o.granite_type_id
AND g.production_id = o.production_id
LEFT JOIN users u

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						108
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        ON u.id = o.employee_id
        AND u.production_id = o.production_id";

if (!empty($where)) {
    $sql .= " WHERE " . implode(' AND ', $where);
}

$sortColumns = [
    'id' => 'o.id',
    'client' => 'c.last_name',
    'granite' => "COALESCE(g.name, NULLIF(TRIM(o.material), ''))",
    'quantity' => 'o.quantity',
    'area' => 'o.area_m2',
    'volume' => 'o.volume_m3',
    'total_price' => 'o.total_price',
    'status' => 'o.status',
    'created_at' => 'o.created_at',
];

$sort = array_key_exists((string)($filters['sort'] ?? ''), $sortColumns)
    ? (string)$filters['sort']
    : 'created_at';

$direction = strtolower((string)($filters['direction'] ?? 'desc')) === 'asc' ? 'ASC' : 'DESC';
$sql .= " ORDER BY {"$sortColumns[$sort]} {"$direction}, o.id DESC";
self::debugFilters($sql, $params);
return Core::get()->db->fetchAll($sql, $params);
}

private static function isValidDate($value)
{
    $date = \DateTimeImmutable::createFromFormat('!Y-m-d', trim((string)$value));
    return $date !== false && $date->format('Y-m-d') === trim((string)$value);
}

private static function debugFilters($sql, $params)
{
    Logger::debug('Order list SQL', [
        'sql' => $sql,

```

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						109
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'params' => $params,
    );
}

public static function getDetails($id)
{
    $productionId = TenantContext::requireProductionId();
    $productionWhere = TenantContext::tableHasColumn('orders', 'production_id')
        ? 'o.production_id = :production_id'
        : 'c.production_id = :production_id';
    return Core::get()->db->fetch(
        "SELECT o.*, c.first_name AS client_first_name, c.last_name AS client_last_name,
c.email AS client_email,
        COALESCE(g.name, NULLIF(TRIM(o.material), '')) AS granite_name,
        g.density_kg_m3 AS granite_density_kg_m3,
        c.phone_number AS client_phone, u.first_name AS employee_first_name,
u.last_name AS employee_last_name
        FROM orders o
        LEFT JOIN clients c
            ON c.id = o.client_id
        AND c.production_id = o.production_id
        LEFT JOIN granite_types g
            ON g.id = o.granite_type_id
        AND g.production_id = o.production_id
        LEFT JOIN users u
            ON u.id = o.employee_id
        AND u.production_id = o.production_id
        WHERE o.id = :id AND {$productionWhere}",
        [':id' => $id, ':production_id' => $productionId]
    );
}

public static function getDeletionBlockers($order): array
{
    if (!$order) {
        return ['Замовлення не знайдено.'];
    }
}

```

					ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
						110
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$blockers = [];
$status = self::normalizeStatus($order->status ?? '');
if (!in_array($status, ['new', 'pending_confirmation'], true)) {
    $blockers[] = 'Фізично видаляти можна лише нове або непідтверджене
замовлення.';
}
$db = Core::get()->db;
if (TenantContext::tableHasColumn('orders', 'confirmed_at') && !empty($order-
>confirmed_at)) {
    $blockers[] = 'Замовлення вже підтверджено.';
} elseif (self::tableHasColumn('audit_logs', 'entity_type')
    && self::tableHasColumn('audit_logs', 'entity_id')
    && self::tableHasColumn('audit_logs', 'action')
) {
    $confirmed = $db->fetch(
        "SELECT id FROM audit_logs
        WHERE entity_type = 'order' AND entity_id = :order_id AND action = 'confirm'
        LIMIT 1",
        [':order_id' => (int)$order->id]
    );
    if ($confirmed) {
        $blockers[] = 'Замовлення вже підтверджено.';
    }
}
$dependencies = [
    'order_items' => 'створені позиції замовлення',
    'products' => 'створені деталі або складська продукція',
    'finished_products' => 'готова продукція',
    'cutting_outputs' => 'виробничі операції розкрою',
    'packing_batch_items' => 'позиції пакування',
    'packing_batches' => 'пакування',
    'shipments' => 'відправлення',
];
foreach ($dependencies as $table => $label) {
    if (!self::tableHasColumn($table, 'order_id')) {

```

					ІПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						111
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        continue;
    }
    $row = $db->fetch(
        "SELECT id FROM `{table}` WHERE order_id = :order_id LIMIT 1",
        [':order_id' => (int)$order->id]
    );
    if ($row) {
        $blockers[] = 'Є пов'язані дані: ' . $label . '!';
    }
}
return array_values(array_unique($blockers));
}

private static function tableHasColumn(string $table, string $column): bool
{
    static $cache = [];
    $key = $table . '.' . $column;
    if (!array_key_exists($key, $cache)) {
        try {
            $row = Core::get()->db->fetch(
                'SELECT 1
                FROM information_schema.COLUMNS
                WHERE TABLE_SCHEMA = DATABASE()
                AND TABLE_NAME = :table_name
                AND COLUMN_NAME = :column_name
                LIMIT 1',
                [':table_name' => $table, ':column_name' => $column]
            );
            $cache[$key] = $row !== null;
        } catch (\Throwable $e) {
            $cache[$key] = false;
        }
    }
    return $cache[$key];
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						112
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public static function calculatePrice($data)
{
    $length = max(0, (float)($data['length'] ?? 0));
    $width = max(0, (float)($data['width'] ?? 0));
    $thickness = max(0, (float)($data['thickness'] ?? 0));
    $quantity = max(1, (int)($data['quantity'] ?? 1));
    $area = max(0, ($length * $width) / 1000000);
    $volume = max(0, ($length * $width * $thickness) / 1000000000);
    $unitPrice = (float)($data['granite_unit_price'] ?? $data['material_price'] ?? 0);
    $singleDetailPrice = ($thickness >= 40 ? $volume : $area) * $unitPrice;
    $base = $singleDetailPrice * $quantity;
    $extras = max(0, (float)($data['extra_services_price'] ?? 0));
    $subtotal = $base + $extras;
    $discountPercent = min(100, max(0, (float)($data['discount_percent'] ?? 0)));
    return round($subtotal - ($subtotal * $discountPercent / 100), 2);
}

public static function getClientSingleDetailPrice($order)
{
    $singleDetailPrice = (float)($order->single_detail_price ?? 0);
    if ($singleDetailPrice <= 0) {
        $quantity = max(1, (int)($order->quantity ?? 1));
        $singleDetailPrice = (float)($order->detail_price ?? 0) / $quantity;
    }
    return self::applyClientDiscount($singleDetailPrice, $order);
}

public static function getClientDetailsTotalPrice($order)
{
    return self::applyClientDiscount($order->detail_price ?? 0, $order);
}

public static function getClientExtraServicesPrice($order)
{
    return self::applyClientDiscount($order->extra_services_price ?? 0, $order);
}

private static function applyClientDiscount($amount, $order)
{

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						113
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $discountPercent = min(100, max(0, (float)($order->discount_percent ?? 0)));
        return round((float)$amount * (1 - $discountPercent / 100), 2);
    }
}

```

А.2 Розширений лістинг програмного коду модуля доступу до фінансової аналітики

У цьому підрозділі наведено програмний код основних компонентів сервісу підписки, який використовується для перевірки доступу до платного функціонального модуля фінансової аналітики. Наведений код демонструє перевірку активної підписки, отримання даних про підписку, створення очікуваної підписки, активацію доступу після підтвердження оплати та використання цих даних для обмеження або відкриття доступу до аналітичного модуля.

```

<?php
namespace MVC\services;
use core\Core;
use DateTimeImmutable;
use DateTimeInterface;
use DateTimeZone;
class SubscriptionService
{
    public const FEATURE_FINANCE_ANALYTICS = 'finance_analytics';
    public const PROVIDER_MONOBANK = 'monobank';
    public const PROVIDER_MANUAL_TEST = 'manual_test';
    public const AMOUNT_MINOR = 5000;
    public const CURRENCY_CODE = 980;
    public const BILLING_INTERVAL = '1m';
    private SubscriptionRepository $subscriptions;
    public function __construct(?SubscriptionRepository $subscriptions = null)
    {
        $this->subscriptions = $subscriptions ?? new SubscriptionRepository();
    }
}

```

```

public function hasActiveFeature(int $productionId, string $featureKey): bool
{
    return $this->subscriptions->findActiveFeature($productionId, $featureKey, $this-
>now()) !== null;
}
public function getFeatureSubscription(int $productionId, string $featureKey): ?object
{
    return $this->subscriptions->findFeatureForProduction($productionId, $featureKey);
}
public function canAccessFeature(int $productionId, string $featureKey): bool
{
    return $this->hasActiveFeature($productionId, $featureKey);
}
public function createPendingFinanceSubscription(int $productionId, int $ownerUserId):
object
{
    $current = $this->getFeatureSubscription($productionId,
self::FEATURE_FINANCE_ANALYTICS);
    if (
        $current
        && (
            (string)$current->status === 'pending'
            || (
                in_array((string)$current->status, ['active', 'past_due', 'cancelling'], true)
                && $this->hasActiveFeature($productionId,
self::FEATURE_FINANCE_ANALYTICS)
            )
        )
    ) {
        throw new \RuntimeException('Підписка вже створена або активна.');
```

```

    }

    $reference = $this->generateReference($productionId);
    if ($current) {
        $this->subscriptions->updateProviderData((int)$current->id, [

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						115
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'provider' => self::PROVIDER_MONOBANK,
        'provider_reference' => $reference,
        'provider_subscription_id' => null,
        'status' => 'pending',
        'amount_minor' => self::AMOUNT_MINOR,
        'currency_code' => self::CURRENCY_CODE,
        'billing_interval' => self::BILLING_INTERVAL,
        'current_period_start' => null,
        'current_period_end' => null,
        'last_payment_at' => null,
        'next_payment_at' => null,
        'cancel_at_period_end' => 0,
        'cancelled_at' => null,
    ];

    AuditService::log('subscription_pending', 'production_subscription', (int)$current-
>id, 'Створено pending-підписку на фінансову аналітику.');
```

```

        return $this->subscriptions->findFeatureForProduction($productionId,
self::FEATURE_FINANCE_ANALYTICS);
    }

    $id = $this->subscriptions->createPending([
        'production_id' => $productionId,
        'feature_key' => self::FEATURE_FINANCE_ANALYTICS,
        'provider' => self::PROVIDER_MONOBANK,
        'provider_reference' => $reference,
        'amount_minor' => self::AMOUNT_MINOR,
        'currency_code' => self::CURRENCY_CODE,
        'billing_interval' => self::BILLING_INTERVAL,
        'created_by_user_id' => $ownerUserId,
    ]);

    AuditService::log('subscription_pending', 'production_subscription', $id, 'Створено
pending-підписку на фінансову аналітику.');
```

```

        return $this->subscriptions->findFeatureForProduction($productionId,
self::FEATURE_FINANCE_ANALYTICS);
    }

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						116
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public function activateSubscription(int $subscriptionId, array $providerData): void
{
    $paidAt = $this->dateOrNow($providerData['paid_at'] ??
$providerData['provider_modified_at'] ?? null);
    $periodStart = $this->dateOrNull($providerData['current_period_start'] ?? null) ??
$paidAt;
    $periodEnd = $this->dateOrNull($providerData['current_period_end'] ?? null)
    ?? $periodStart->modify('+1 month');

    $this->subscriptions->updateProviderData($subscriptionId, [
        'provider_subscription_id' => $providerData['provider_subscription_id'] ?? null,
        'status' => 'active',
        'current_period_start' => $periodStart->format('Y-m-d H:i:s'),
        'current_period_end' => $periodEnd->format('Y-m-d H:i:s'),
        'last_payment_at' => $paidAt->format('Y-m-d H:i:s'),
        'next_payment_at' => $this->dateOrNull($providerData['next_payment_at'] ?? null)?->format('Y-m-d H:i:s'),
        'cancel_at_period_end' => 0,
    ]);
    $this->logSystemAudit($subscriptionId, 'subscription_active', 'Підписку на фінансову
аналітику активовано.');
```

```

    }
    public function markPastDue(int $subscriptionId): void
    {
        $this->subscriptions->updateStatus($subscriptionId, 'past_due');
        $this->logSystemAudit($subscriptionId, 'subscription_past_due', 'Не вдалося
продовжити підписку на фінансову аналітику.');
```

```

    }
    public function cancelAtPeriodEnd(int $subscriptionId): void
    {
        $subscription = $this->subscriptions->findById($subscriptionId);
        if (!$subscription || !in_array((string)$subscription->status, ['active', 'past_due',
'cancelling'], true)) {
            return;
        }
    }

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						117
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$periodEnd = $this->dateOrNull($subscription->current_period_end ?? null);
if ($periodEnd === null || $periodEnd <= $this->now()) {
    $this->subscriptions->updateStatus($subscriptionId, 'expired', [
        'cancel_at_period_end' => 1,
        'cancelled_at' => $this->now()->format('Y-m-d H:i:s'),
    ]);
    return;
}
$this->subscriptions->updateStatus($subscriptionId, 'active', [
    'cancel_at_period_end' => 1,
    'cancelled_at' => $this->now()->format('Y-m-d H:i:s'),
]);
$this->logSystemAudit($subscriptionId, 'subscription_cancelled', 'Підписку на
фінансову аналітику скасовано в кінець періоду.');
}
public function expireFinishedSubscriptions(): void
{
    $now = $this->now()->format('Y-m-d H:i:s');
    Core::get()->db->query(
        "UPDATE production_subscriptions
        SET status = 'expired'
        WHERE status IN ('active', 'past_due', 'cancelling', 'cancelled')
        AND current_period_end IS NOT NULL
        AND current_period_end <= :now",
        [':now' => $now]
    );
}
public function reconcileMonobankSubscription(int $localSubscriptionId): array
{
    $subscription = $this->subscriptions->findById($localSubscriptionId);
    if (!$subscription || (string)$subscription->provider !==
self::PROVIDER_MONOBANK) {
        throw new \RuntimeException('Підписку monobank не знайдено.');
```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						118
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$providerSubscriptionId = trim((string)($subscription->provider_subscription_id ?? ''));
if ($providerSubscriptionId === '') {
    throw new \RuntimeException('У підписки відсутній provider subscription ID.');
```

```

}
$client = new MonobankAcquiringClient();
$providerStatus = $client->getSubscriptionStatus($providerSubscriptionId);
$expectedAmountMinor = (int)$subscription->amount_minor;
$expectedCurrencyCode = (int)$subscription->currency_code;
$expectedInterval = (string)$subscription->billing_interval;
$this->assertTariff($providerStatus, $expectedAmountMinor,
$expectedCurrencyCode);
if (
    $expectedAmountMinor <= 0
    || $expectedCurrencyCode !== self::CURRENCY_CODE
    || $expectedInterval !== self::BILLING_INTERVAL
) {
    throw new \RuntimeException('Локальний тариф підписки не відповідає тарифу
фінансової аналітики.');
```

```

}
$createdAt = $this->dateOrNull($subscription->created_at ?? null) ?? $this->now();
$paymentHistory = $client->getSubscriptionPayments(
    $providerSubscriptionId,
    $createdAt->modify('-1 day')->format(DATE_RFC3339),
    $this->now()->modify('+1 day')->format(DATE_RFC3339)
);
$payment = $this->findSuccessfulProviderPayment(
    $paymentHistory,
    $expectedAmountMinor,
    $expectedCurrencyCode
);
$providerPayments = $this->providerPayments($paymentHistory);
$providerStatusName = $this->firstString($providerStatus, ['status']);
$totalPaid = (int)($providerStatus['summary']['totalPaid'] ?? 0);
if ($payment === null || $totalPaid <= 0) {
    $processing = $this->hasRecentProcessingPayment($providerPayments);

```

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						119
Змн.	Арк.	№ докум.	Підпис	Дата		

```

return [
    'state' => $this->isFailureStatus($providerStatusName)
        ? 'failed'
        : ($processing ? 'processing' : 'unconfirmed'),
    'provider_status' => $providerStatusName,
    'total_paid' => $totalPaid,
    'payment_recorded' => false,
    'provider_payment_statuses' =>
array_values(array_unique(array_filter(array_map(
    fn(array $item): string => strtolower((string)$this->firstString($item, ['status',
'paymentStatus'])),
    $providerPayments
))),
];
}
$chargedAt = $this->firstString($payment, ['chargedAt']);
$paidAt = $this->dateOrNull($chargedAt);
if ($paidAt === null) {
    throw new \RuntimeException('monobank не повернув коректний chargedAt
успішної оплати.');
```

```

    }
    $periodStart = $this->dateOrNull($this->firstString($providerStatus, [
        'startDate',
        'currentPeriodStart',
        'current_period_start',
    ]));
    $periodEnd = $this->dateOrNull($this->firstString($providerStatus, [
        'nextChargeDate',
        'currentPeriodEnd',
        'current_period_end',
        'nextPaymentAt',
    ]));
    $nextPaymentAt = $this->dateOrNull($this->firstString($providerStatus, [
        'nextChargeDate',
        'nextPaymentAt',
    ]));

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						120
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'next_payment_at',
    ));
    if ($periodStart === null || $periodEnd === null || $nextPaymentAt === null) {
        throw new \RuntimeException('monobank не повернув повні дати періоду
підписки.');
```

```

    }
    $paidAtSql = $paidAt->format('Y-m-d H:i:s');
    $pdo = Core::get()->db->pdo;
    $ownsTransaction = !$pdo->inTransaction();
    if ($ownsTransaction) {
        $pdo->beginTransaction();
    }
    try {
        $lockedSubscription = $this->subscriptions-
>findByIdForUpdate($localSubscriptionId);
        if (!$lockedSubscription) {
            throw new \RuntimeException('Локальну підписку не знайдено під час
синхронізації.');
```

```

        }
        $recorded = !$this->subscriptions->equivalentPaymentExists(
            (int)$lockedSubscription->id,
            self::PROVIDER_MONOBANK,
            $expectedAmountMinor,
            $expectedCurrencyCode,
            $paidAtSql
        );
        if ($recorded) {
            $this->subscriptions->recordPayment([
                'production_id' => (int)$lockedSubscription->production_id,
                'subscription_id' => (int)$lockedSubscription->id,
                'provider' => self::PROVIDER_MONOBANK,
                'provider_payment_id' => $this->firstString($payment, [
                    'paymentId',
                    'payment_id',
                    'transactionId',

```

					ІПЗ.КР.Б – 121 – 26 – ІЗ	Арк.
						121
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'operationId',
    ),
    'provider_invoice_id' => $this->firstString($payment, ['invoiceId', 'invoice_id']),
    'amount_minor' => $expectedAmountMinor,
    'currency_code' => $expectedCurrencyCode,
    'status' => 'paid',
    'paid_at' => $paidAtSql,
    'provider_modified_at' => $chargedAt !== null ? $paidAtSql : null,
    );
}

$providerCancellation = $this->providerCancellationState($providerStatus);
$cancelAtPeriodEnd = (int)($lockedSubscription->cancel_at_period_end ?? 0);
$cancelledAt = $lockedSubscription->cancelled_at ?? null;
if ($providerCancellation === true) {
    $cancelAtPeriodEnd = 1;
    $cancelledAt = $cancelledAt ?: $this->now()->format('Y-m-d H:i:s');
} elseif ($providerCancellation === false) {
    $cancelAtPeriodEnd = 0;
    $cancelledAt = null;
}

$this->subscriptions->updateProviderData((int)$lockedSubscription->id, [
    'provider_subscription_id' => $providerSubscriptionId,
    'status' => 'active',
    'current_period_start' => $periodStart->format('Y-m-d H:i:s'),
    'current_period_end' => $periodEnd->format('Y-m-d H:i:s'),
    'last_payment_at' => $paidAtSql,
    'next_payment_at' => $nextPaymentAt->format('Y-m-d H:i:s'),
    'cancel_at_period_end' => $cancelAtPeriodEnd,
    'cancelled_at' => $cancelledAt,
    );

if ($recorded) {
    $this->logSystemAudit(
        (int)$lockedSubscription->id,
        'subscription_payment',
        'Успішну оплату підписки на фінансову аналітику зафіксовано.',
    );
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						122
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        [
            'amount_minor' => $expectedAmountMinor,
            'currency_code' => $expectedCurrencyCode,
            'paid_at' => $paidAtSql,
        ]
    );
}

if ($ownsTransaction) {
    $pdo->commit();
}

} catch (\Throwable $exception) {
    if ($ownsTransaction && $pdo->inTransaction()) {
        $pdo->rollBack();
    }
    throw $exception;
}

return [
    'state' => 'active',
    'provider_status' => $providerStatusName,
    'total_paid' => $totalPaid,
    'payment_recorded' => $recorded,
    'paid_at' => $paidAtSql,
    'current_period_start' => $periodStart->format('Y-m-d H:i:s'),
    'current_period_end' => $periodEnd->format('Y-m-d H:i:s'),
    'next_payment_at' => $nextPaymentAt->format('Y-m-d H:i:s'),
];

}

public function processConfirmedPayment(object $subscription, array $payload): bool
{
    $paymentId = $this->firstString($payload, ['paymentId', 'payment_id', 'transactionId',
'operationId']);
    $invoiceId = $this->firstString($payload, ['invoiceId', 'invoice_id']);
    if ($this->subscriptions->paymentExists((string)$subscription->provider, $paymentId,
$invoiceId)) {
        return false;
    }
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						123
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
    $expectedAmountMinor = (int)$subscription->amount_minor;
    $expectedCurrencyCode = (int)$subscription->currency_code;
    $this->assertTariff($payload, $expectedAmountMinor, $expectedCurrencyCode);
    $paidAt = $this->dateOrNow($this->firstString($payload, [
        'paidAt',
        'paid_at',
        'chargedAt',
        'modifiedDate',
        'createdDate',
    ]));
    $modifiedAt = $this->dateOrNull($this->firstString($payload, [
        'modifiedDate',
        'modified_at',
        'chargedAt',
    ])) ?? $paidAt;
    $periodStart = $this->dateOrNull($this->firstString($payload, ['currentPeriodStart',
    'current_period_start'])) ?? $paidAt;
    $periodEnd = $this->dateOrNull($this->firstString($payload, ['currentPeriodEnd',
    'current_period_end']))
        ?? $periodStart->modify('+1 month');
    if ($this->subscriptions->equivalentPaymentExists(
        (int)$subscription->id,
        (string)$subscription->provider,
        $expectedAmountMinor,
        $expectedCurrencyCode,
        $paidAt->format('Y-m-d H:i:s')
    )) {
        return false;
    }
    $this->subscriptions->recordPayment([
        'production_id' => (int)$subscription->production_id,
        'subscription_id' => (int)$subscription->id,
        'provider' => (string)$subscription->provider,
        'provider_payment_id' => $paymentId,
    ]);
}

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						124
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'provider_invoice_id' => $invoiceId,
        'amount_minor' => $expectedAmountMinor,
        'currency_code' => $expectedCurrencyCode,
        'status' => 'paid',
        'paid_at' => $paidAt->format('Y-m-d H:i:s'),
        'provider_modified_at' => $modifiedAt->format('Y-m-d H:i:s'),
    ]);
    $this->activateSubscription((int)$subscription->id, [
        'provider_subscription_id' => $this->firstString($payload, ['subscriptionId',
'subscription_id']) ?: $subscription->provider_subscription_id,
        'paid_at' => $paidAt->format('Y-m-d H:i:s'),
        'provider_modified_at' => $modifiedAt->format('Y-m-d H:i:s'),
        'current_period_start' => $periodStart->format('Y-m-d H:i:s'),
        'current_period_end' => $periodEnd->format('Y-m-d H:i:s'),
        'next_payment_at' => $this->firstString($payload, ['nextPaymentAt',
'next_payment_at']),
    ]);
    $this->logSystemAudit((int)$subscription->id, 'subscription_payment', 'Успішну
оплату підписки на фінансову аналітику зафіксовано.', [
        'payment_id' => $paymentId,
        'invoice_id' => $invoiceId,
        'amount_minor' => $expectedAmountMinor,
        'currency_code' => $expectedCurrencyCode,
    ]);
    return true;
}

private function findSuccessfulProviderPayment(
    array $paymentHistory,
    int $expectedAmountMinor,
    int $expectedCurrencyCode
): ?array
{
    $payments = $paymentHistory['payments'] ?? $paymentHistory['items'] ??
$paymentHistory['data'] ?? [];
    if (is_array($payments)) {

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						125
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        return null;
    }
    foreach ($payments as $payment) {
        if (!is_array($payment)) {
            continue;
        }
        $status = strtolower((string)$this->firstString($payment, ['status', 'paymentStatus']));
        $amount = (int)($payment['amount'] ?? $payment['amount_minor'] ??
$payment['amountMinor'] ?? 0);
        $ccy = (int)($payment['ccy'] ?? $payment['currency_code'] ??
$payment['currencyCode'] ?? 0);
        if (
            $status === 'success'
            && $amount === $expectedAmountMinor
            && $ccy === $expectedCurrencyCode
        ) {
            return $payment;
        }
    }
    return null;
}

private function providerPayments(array $paymentHistory): array
{
    $payments = $paymentHistory['payments'] ?? $paymentHistory['items'] ??
$paymentHistory['data'] ?? [];
    if (is_array($payments)) {
        return [];
    }
    return array_values(array_filter($payments, 'is_array'));
}

private function hasRecentProcessingPayment(array $payments): bool
{
    $threshold = $this->now()->modify('-10 minutes');
    foreach ($payments as $payment) {
        $status = strtolower((string)$this->firstString($payment, ['status', 'paymentStatus']));
    }
}

```

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						126
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        if (!in_array($status, ['new', 'pending', 'processing', 'created'], true)) {
            continue;
        }
        $providerDate = $this->dateOrNull($this->firstString($payment, [
            'chargedAt',
            'createdDate',
            'modifiedDate',
            'paidAt',
        ]));
        if ($providerDate !== null && $providerDate > $threshold) {
            return true;
        }
    }
    return false;
}

private function assertTariff(array $payload, int $expectedAmountMinor, int
$expectedCurrencyCode): void
{
    $amount = (int)($payload['amount'] ?? $payload['amount_minor'] ??
$payload['amountMinor'] ?? 0);
    $ccy = (int)($payload['ccy'] ?? $payload['currency_code'] ?? $payload['currencyCode']
?? 0);
    if ($amount !== $expectedAmountMinor || $ccy !== $expectedCurrencyCode) {
        throw new \RuntimeException('monobank повернув неочікувану суму або
валюту.');
```

```

    }
}

private function providerCancellationState(array $payload): ?bool
{
    foreach (['cancelAtPeriodEnd', 'cancel_at_period_end', 'cancelled', 'canceled'] as $key) {
        if (array_key_exists($key, $payload)) {
            return filter_var($payload[$key], FILTER_VALIDATE_BOOLEAN,
FILTER_NULL_ON_FAILURE);
        }
    }
}

```

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						127
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        foreach (['autoRenew', 'auto_renew', 'recurring'] as $key) {
            if (array_key_exists($key, $payload)) {
                $value = filter_var($payload[$key], FILTER_VALIDATE_BOOLEAN,
FILTER_NULL_ON_FAILURE);
                return $value === null ? null : !$value;
            }
        }
        $status = strtolower((string)$this->firstString($payload, ['status', 'subscriptionStatus']));
        if (in_array($status, ['cancelled', 'canceled'], true)) {
            return true;
        }
        return null;
    }
    public function firstString(array $payload, array $keys): ?string
    {
        foreach ($keys as $key) {
            if (isset($payload[$key]) && trim((string)$payload[$key]) !== "") {
                return trim((string)$payload[$key]);
            }
        }
        return null;
    }
    public function isSuccessfulStatus(?string $status): bool
    {
        return in_array(strtolower((string)$status), ['success', 'paid', 'approved', 'hold',
'finalized'], true);
    }
    public function isFailureStatus(?string $status): bool
    {
        return in_array(strtolower((string)$status), ['failure', 'failed', 'declined', 'expired',
'reversed', 'cancelled'], true);
    }
    private function generateReference(int $productionId): string
    {
        return 'fin_' . $productionId . '_' . bin2hex(random_bytes(16));
    }

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						128
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    }
    private function now(): DateTimeImmutable
    {
        return new DateTimeImmutable('now', new DateTimeZone('Europe/Kyiv'));
    }
    private function dateOrNow($value): DateTimeImmutable
    {
        return $this->dateOrNull($value) ?? $this->now();
    }
    private function dateOrNull($value): ?DateTimeImmutable
    {
        $value = trim((string)$value);
        if ($value === "") {
            return null;
        }
        try {
            return (new DateTimeImmutable($value, new DateTimeZone('Europe/Kyiv')))
                ->setTimezone(new DateTimeZone('Europe/Kyiv'));
        } catch (\Throwable $exception) {
            return null;
        }
    }
    private function logSystemAudit(int $subscriptionId, string $action, string $description,
    array $context = []): void
    {
        try {
            $subscription = Core::get()->db->fetch(
                'SELECT production_id, feature_key, status
                FROM production_subscriptions
                WHERE id = :id
                LIMIT 1',
                [':id' => $subscriptionId]
            );
            Core::get()->db->insert('audit_logs', [
                'production_id' => (int)($subscription->production_id ?? 0),

```

					ІІЗ.КР.Б – 121 – 26 – ІІЗ	Арк.
						129
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        'user_id' => null,
        'user_name' => 'System',
        'user_role' => 'system',
        'action' => $action,
        'entity_type' => 'production_subscription',
        'entity_id' => $subscriptionId,
        'old_value' => null,
        'new_value' => $context ? json_encode($context,
JSON_UNESCAPED_UNICODE | JSON_UNESCAPED_SLASHES) : null,
        'description' => $description,
        'ip_address' => $_SERVER['REMOTE_ADDR'] ?? null,
        'user_agent' => $_SERVER['HTTP_USER_AGENT'] ?? null,
    });
} catch (\Throwable $exception) {
    // Subscription state must not depend on audit availability.
}
}
}

```

					ПЗ.КР.Б – 121 – 26 – ПЗ	Арк.
						130
Змн.	Арк.	№ докум.	Підпис	Дата		