МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КОМОРНИЙ ЄВГЕН МИКОЛАЙОВИЧ

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій
д-р техн. наук, професор
_____Наталія ВЕСЕЛОВСЬКА
«____»_____ 2026 р.

ПРОГРАМНА РЕАЛІЗАЦІЯ АВТОМАТИЗАЦІЇ БІЗНЕС-ПРОЦЕСІВ ПІДПРИЄМСТВА

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (бакалаврська) робота

Керівник:
Ірина ФРИЗ, старший викладач кафедри
інформаційних технологій,
канд. фіз.-мат. наук
_____

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)
Голова ЕК: _____
(підпис)

Вінниця - 2026

АНОТАЦІЯ

Коморний Є. М. Програмна реалізація автоматизації бізнес-процесів підприємства. Кваліфікаційна робота за спеціальністю 122 «Комп'ютерні науки». Донецький національний університет імені Василя Стуса, Вінниця, 2026.

У кваліфікаційній (бакалаврській) роботі досліджено та проаналізовано автоматизацію бізнес-процесів підприємства, визначено основні принципи побудови CRM-систем для управління клієнтами та замовленнями. За допомогою мови програмування Python, системи керування базами даних SQLite та бібліотеки CustomTkinter розроблено CRM-систему, основною метою якої є автоматизація обліку клієнтів, управління замовленнями та формування звітності підприємства.

Ключові слова: CRM-система, автоматизація, Python, SQLite, інформаційна система.

66 ст. 25 рис., 5 табл., 40 джерел.

ANNOTATION

Komornyi Y. M. Software implementation of enterprise business process automation. Bachelor's thesis in specialty 122 «Computer Science». Vasyl' Stus Donetsk National University, Vinnytsia, 2026.

The qualification (bachelor's) thesis investigates and analyzes the automation of enterprise business processes and identifies the main principles of CRM system development for customer and order management. Using the Python programming language, the SQLite database management system, and the CustomTkinter library, a CRM system was developed. The primary purpose of the system is to automate customer record management, order processing, and enterprise reporting.

Keywords: CRM system, automation, Python, SQLite, information system.

66 pages, 25 figures, 5 tables, 40 references.

ЗМІСТ

ВСТУП

У сучасних умовах розвитку інформаційних технологій підприємства все активніше використовують програмні засоби для автоматизації основних бізнес-процесів. Використання інформаційних систем дозволяє підвищити ефективність обробки інформації, спростити управління даними та забезпечити підтримку процесів прийняття управлінських рішень. Одним із напрямів автоматизації є використання CRM-систем, призначених для організації роботи з клієнтами та замовленнями.

Актуальність теми дослідження зумовлена необхідністю створення ефективних програмних рішень для автоматизації процесів обліку клієнтів та замовлень, що дозволяє оптимізувати процеси обробки інформації та спростити контроль виконання основних операцій. Незважаючи на наявність великої кількості готових CRM-систем, багато з них є надмірно складними, дорогими або не адаптованими до потреб малих підприємств, що обумовлює доцільність розробки власного програмного продукту.

Метою кваліфікаційної роботи є розробка програмного забезпечення для автоматизації бізнес-процесів підприємства у вигляді CRM-системи, яка забезпечує ефективне управління клієнтами та замовленнями, а також формування звітності.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати предметну область автоматизації бізнес-процесів підприємства;

– дослідити існуючі програмні рішення та визначити їх переваги і недоліки;

– сформувати функціональні та нефункціональні вимоги до системи;

– розробити архітектуру програмного забезпечення;

– спроєктувати структуру бази даних;

– реалізувати програмний продукт і забезпечити функціонування основних модулів системи;

– провести тестування програмного забезпечення та проаналізувати результати його роботи.

Об’єктом дослідження є процеси автоматизації бізнес-процесів підприємства.

Предметом дослідження є методи та засоби розробки CRM-системи для управління клієнтами та замовленнями.

Практичне значення одержаних результатів полягає у створенні програмного продукту, який може бути використаний для автоматизації діяльності малих та середніх підприємств. Розроблена система має підвищити ефективність обробки інформації, забезпечити зручний інтерфейс користувача та сприяти покращенню контролю за виконанням замовлень.

Зв’язок роботи з науковими програмами, планами, темами. Наведені у бакалаврські роботі дослідження пов’язані з фундаментальною науково-дослідною роботою «Розвиток методів комп’ютерно-математичного моделювання складних систем та процесів у сфері освіти та діяльності IT-підприємств» (№ держреєстрації 0126U003221, 2026-2031 рр.).

Результати дослідження доповідалися на VII Всеукраїнській науково-практичній конференції здобувачів вищої освіти та молодих вчених «Прикладні інформаційні технології», 15 травня 2026 року. Донецький національний університет імені Василя Стуса, м. Вінниця, тема доповіді: «Програмна реалізація системи автоматизації бізнес-процесів підприємства».

У першому розділі виконано аналіз предметної області автоматизації бізнес-процесів підприємства, досліджено особливості CRM-систем, проведено аналіз існуючих програмних рішень, розглянуто основні підходи до реалізації CRM-систем, обґрунтовано вибір інструментальних засобів розробки та сформульовано функціональні й нефункціональні вимоги до програмного забезпечення. У другому розділі розроблено архітектуру CRM-системи,

спроєктовано структуру бази даних та інтерфейс користувача, а також описано алгоритми, застосовані у системі. У третьому розділі наведено реалізацію програмного забезпечення, зокрема структуру програмного проєкту, механізми роботи з базою даних, бізнес-логіку системи та графічний інтерфейс користувача, а також проведено тестування програмного забезпечення та аналіз результатів його роботи.

Список використаних джерел містить 40 найменувань. У роботі наведено 25 рисунків та 5 таблиць.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ CRM РІШЕНЬ

1.1 Сутність та особливості автоматизації бізнес-процесів підприємства

У сучасних умовах цифрової трансформації економіки автоматизація бізнес-процесів є невід'ємною складовою ефективного функціонування підприємств. Бізнес-процеси охоплюють сукупність взаємопов'язаних дій, спрямованих на досягнення певного результату, зокрема обслуговування клієнтів, обробку замовлень, управління ресурсами та аналіз діяльності підприємства [1].

У межах діяльності підприємства бізнес-процеси доцільно поділяти на основні, допоміжні та управлінські [2]. Основні бізнес-процеси безпосередньо пов'язані зі створенням цінності для клієнта та отриманням результату діяльності підприємства. До них можна віднести обробку замовлень, взаємодію з клієнтами, надання послуг або продаж товарів. Саме ці процеси найбільше впливають на якість обслуговування, швидкість виконання робіт та загальну ефективність підприємства.

Допоміжні бізнес-процеси забезпечують підтримку основної діяльності підприємства. Вони не створюють кінцевий продукт безпосередньо, однак є необхідними для стабільного функціонування організації. До таких процесів належать ведення внутрішньої документації, збереження інформації, технічна підтримка, контроль доступу до даних та облік виконаних дій.

Управлінські бізнес-процеси спрямовані на планування, контроль та аналіз діяльності підприємства. Вони забезпечують прийняття управлінських рішень на основі накопичених даних. Прикладами таких процесів є аналіз кількості замовлень, оцінка фінансових показників, контроль строків виконання задач та формування звітності.

У контексті розроблюваної CRM-системи найбільша увага приділяється автоматизації основних та управлінських бізнес-процесів. Основні процеси представлені управлінням клієнтами та замовленнями, а управлінські – формуванням звітності, контролем статусів замовлень і виявленням прострочених задач. Такий підхід дозволяє не лише зберігати інформацію, а й використовувати її для підвищення ефективності роботи підприємства.

Важливим аспектом дослідження є визначення рівнів автоматизації бізнес-процесів, що дозволяє оцінити ступінь використання інформаційних технологій у діяльності підприємства. Виділяють три основні рівні автоматизації: ручний, частково автоматизований та повністю автоматизований.

Ручний рівень характеризується виконанням усіх операцій без використання спеціалізованого програмного забезпечення. У цьому випадку облік клієнтів і замовлень здійснюється за допомогою паперових носіїв або простих інструментів, що значно ускладнює пошук інформації, підвищує ризик помилок та знижує ефективність роботи.

Частково автоматизований рівень передбачає використання окремих програмних засобів для виконання певних операцій. Наприклад, інформація може зберігатися у електронних таблицях або простих базах даних, проте відсутня єдина система управління, що призводить до розрізненості даних та ускладнює контроль за виконанням бізнес-процесів.

Повністю автоматизований рівень передбачає використання комплексних інформаційних систем, які забезпечують централізоване збереження даних, автоматичну обробку інформації та контроль виконання задач. На цьому рівні значно зменшується вплив людського фактора, підвищується швидкість обробки інформації та забезпечується можливість аналітики на основі накопичених даних.

До програмного продукту висуваються вимоги щодо забезпечення автоматизації основних бізнес-процесів підприємства, централізованого управління інформацією про клієнтів і замовлення, автоматичного контролю строків виконання замовлень та формування звітності. Система повинна сприяти

підвищенню ефективності роботи підприємства, оптимізації обробки інформації та спрощенню процесу прийняття управлінських рішень.

У сучасних інформаційних системах автоматизація бізнес-процесів реалізується за допомогою різних класів програмного забезпечення, серед яких найбільш поширеними є CRM-, ERP- та SCM-системи. Кожен із зазначених типів систем має своє призначення та використовується для вирішення певного кола задач.

CRM-системи (Customer Relationship Management) орієнтовані на управління взаємовідносинами з клієнтами. Основною метою таких систем є збереження інформації про клієнтів, історію взаємодії з ними, облік замовлень та забезпечення ефективного обслуговування. CRM-системи дозволяють централізувати дані, покращити комунікацію з клієнтами та підвищити рівень їх задоволеності [3].

ERP-системи (Enterprise Resource Planning) призначені для комплексного управління ресурсами підприємства. Вони охоплюють широкий спектр функцій, включаючи фінансовий облік, управління персоналом, планування виробництва та контроль ресурсів. ERP-системи інтегрують різні бізнес-процеси в єдину інформаційну систему, що дозволяє підвищити ефективність управління підприємством у цілому [4].

SCM-системи (Supply Chain Management) використовуються для управління ланцюгами постачання. Вони забезпечують планування, контроль та оптимізацію процесів постачання, зберігання та доставки продукції. Використання SCM-систем дозволяє зменшити витрати, оптимізувати логістичні процеси та підвищити ефективність взаємодії з постачальниками [5].

У межах даної роботи основна увага приділяється саме CRM-системам, оскільки вони безпосередньо пов’язані з управлінням клієнтами та замовленнями. Вибір цього класу систем обумовлений специфікою задачі, яка полягає в автоматизації процесів обліку клієнтів, контролю виконання замовлень та формування звітності. Інші класи систем, такі як ERP та SCM, мають ширший

функціонал і використовуються переважно великими підприємствами, що виходить за межі поставленої задачі.

Автоматизація бізнес-процесів передбачає використання інформаційних технологій для заміни або спрощення ручних операцій, що дозволяє зменшити витрати часу, підвищити точність виконання операцій та забезпечити контроль за виконанням задач [6, с. 87]. Основною метою автоматизації є підвищення ефективності діяльності підприємства, оптимізація внутрішніх процесів та покращення якості обслуговування клієнтів.

Впровадження CRM-систем є одним із ключових напрямів автоматизації, які забезпечують управління взаємовідносинами з клієнтами. Такі системи дозволяють централізувати зберігання інформації про клієнтів та переглядати історію взаємодії з ними. Використання CRM-систем сприяє підвищенню рівня обслуговування клієнтів та забезпечує більш ефективне управління підприємством.

До основних переваг автоматизації бізнес-процесів можна віднести:

– зменшення впливу людського фактора та кількості помилок;

– підвищення швидкості обробки інформації;

– зниження операційних витрат;

– покращення контролю за виконанням завдань;

– можливість аналізу діяльності підприємства на основі накопичених даних;

– масштабованість бізнесу за рахунок адаптації до збільшення обсягів операцій порівняно з ручними методами.

Разом з тим, автоматизація має певні особливості та обмеження. Зокрема, впровадження інформаційних систем потребує початкових витрат часу та ресурсів, а також адаптації персоналу до нових умов роботи. Крім того, важливим аспектом є правильний вибір програмного забезпечення, яке повинно відповідати специфіці діяльності підприємства та забезпечувати необхідний функціонал [7, с. 22].

Особливої актуальності набуває автоматизація для малих та середніх підприємств, які потребують доступних, гнучких та простих у використанні рішень. У таких випадках доцільним є створення спеціалізованих програмних продуктів, адаптованих до конкретних бізнес-процесів підприємства [8, с. 168; 9].

Таким чином, автоматизація бізнес-процесів є важливим інструментом підвищення ефективності діяльності підприємства, а використання CRM-систем дозволяє оптимізувати процеси управління клієнтами та замовленнями, забезпечуючи конкурентні переваги на ринку.

1.2 Аналіз існуючих CRM-систем та програмних аналогів

На сучасному ринку інформаційних технологій представлена значна кількість CRM-систем, які відрізняються функціональними можливостями, складністю використання, вартістю та сферою застосування. Аналіз існуючих рішень дозволяє визначити їх переваги та недоліки, а також обґрунтувати доцільність розробки власного програмного продукту.

Однією з найбільш відомих CRM-систем є Salesforce [10], яка належить до класу корпоративних рішень. Ця система характеризується широким функціоналом, можливістю інтеграції з іншими сервісами, високим рівнем масштабованості та підтримкою хмарних технологій. Водночас її основними недоліками є складність налаштування, висока вартість та надмірність функціоналу для малих підприємств.

Система Zoho CRM [11] орієнтована на малий та середній бізнес і забезпечує широкий набір інструментів для управління клієнтами, автоматизації продажів, аналітики та інтеграції з іншими сервісами. До її переваг належать гнучкість налаштування, підтримка хмарних технологій та відносно невисока вартість використання. Водночас велика кількість параметрів конфігурації може

ускладнювати початкове освоєння системи, а частина функціоналу доступна лише у платних тарифах.

HubSpot CRM [12] є більш простим рішенням, орієнтованим на швидке впровадження та зручність використання. Система надає базові можливості для управління контактами, відстеження угод і аналітики навіть у безкоштовній версії. Основними перевагами є інтуїтивно зрозумілий інтерфейс та легкість освоєння, однак розширений функціонал потребує переходу на платні тарифи.

Окрему увагу слід приділити локальним або самописним CRM-системам, які розробляються під конкретні потреби підприємства. Такі рішення характеризуються гнучкістю, можливістю повної адаптації до бізнес-процесів та відсутністю зайвого функціоналу. Водночас їх розробка потребує часу та відповідних технічних знань.

Аналіз зазначених CRM-систем наведений у таблиці 1.1. Він показує, що сучасні CRM-системи мають широкий функціонал, проте не завжди є оптимальними для використання у малому бізнесі. Зокрема, такі системи, як Salesforce характеризуються високою вартістю та складністю впровадження, що робить їх менш доступними для невеликих підприємств. У свою чергу, Zoho CRM забезпечує широкий набір функцій та можливостей інтеграції, проте для ефективного використання потребує додаткового налаштування системи.

Таблиця 1.1 – Порівняння CRM-систем

Система	Вартість використання	Складність впровадження	Зручність інтерфейсу	Функціональність	Придатність для малого бізнесу
Salesforce	Висока	Висока	Складний	Висока	Низька
Zoho CRM	Середня	Середня	Зручний	Висока	Середня
HubSpot CRM	Середня	Низька	Зручний	Середня	Висока

Система HubSpot CRM є більш простою у використанні, проте її функціональність у безкоштовній версії є обмеженою. На відміну від зазначених рішень, розроблена CRM-система орієнтована на конкретні потреби підприємства, що дозволяє забезпечити достатній функціонал при мінімальних витратах та високій зручності використання.

Порівняльний аналіз розглянутих систем дозволяє виділити основні проблеми існуючих рішень:

- висока вартість впровадження та використання;
- надмірність функціоналу для невеликих підприємств;
- складність налаштування та використання;
- залежність від зовнішніх сервісів та інтернет-з’єднання;
- обмеження безкоштовних версій.

З огляду на зазначені недоліки, доцільною є розробка власної CRM-системи, яка буде орієнтована на потреби малого підприємства. Така система повинна бути простою у використанні, мати інтуїтивно зрозумілий інтерфейс, забезпечувати необхідний функціонал для управління клієнтами та замовленнями, а також працювати автономно без необхідності постійного підключення до мережі Інтернет.

Отже, аналіз існуючих CRM-систем показав, що хоча на ринку представлено багато потужних рішень, вони не завжди відповідають потребам малих підприємств. Це обґрунтовує необхідність створення спеціалізованого програмного продукту, адаптованого до конкретних умов використання.

Таким чином, проведене порівняння підтверджує доцільність розробки власного програмного продукту, який поєднує необхідний функціонал, простоту використання та доступність для малого підприємства.

1.3 Аналіз підходів до реалізації CRM-систем та вибір інструментальних засобів

Розробка CRM-системи може здійснюватися з використанням різних підходів, кожен із яких має свої переваги, обмеження та сферу доцільного застосування. Вибір підходу залежить від масштабу підприємства, вимог до доступності системи, обсягу даних, рівня технічної підготовки користувачів та необхідності подальшого розширення функціоналу.

Одним із поширених підходів є використання хмарних CRM-систем – готових програмних сервісів, що надаються стороннім постачальником [13]. Такі рішення не потребують встановлення на локальний комп’ютер користувача та забезпечують доступ до даних через мережу Інтернет. Їх перевагами є швидке впровадження, можливість роботи з різних пристроїв та регулярне оновлення з боку постачальника. Водночас хмарні системи мають певні обмеження, зокрема залежність від стабільного інтернет-з’єднання, необхідність оплати підписки та менший рівень контролю над збереженням даних.

Іншим підходом є розробка власної веб-орієнтованої CRM-системи, яка розміщується на сервері та доступна користувачам через браузер. У цьому випадку підприємство самостійно розробляє та адмініструє програмне забезпечення. Такий варіант також є ефективним для підприємств, де кілька користувачів повинні працювати із системою одночасно з різних робочих місць. Перевагами веб-орієнтованого підходу є централізований доступ до системи, гнучкість, можливість адаптації функціоналу та вищий рівень контролю над даними, однак його реалізація потребує налаштування серверної частини, забезпечення захисту даних, підтримки хостингу та системного адміністрування [14].

Також можливим є створення десктопної CRM-системи, яка встановлюється та запускається безпосередньо на комп’ютері користувача. Таке рішення є зручним для малих підприємств або окремих робочих місць, де не потрібна складна серверна інфраструктура. Десктопна система може працювати

автономно, забезпечувати швидкий доступ до локальних даних та не залежати від інтернет-з'єднання. Саме цей спосіб реалізації є доцільним у межах даної роботи, оскільки розроблювана система орієнтована на базові потреби малого підприємства: облік клієнтів, управління замовленнями, контроль строків виконання та формування звітності.

Окремо можна виділити варіант використання електронних таблиць для обліку клієнтів і замовлень. Він є простим на початковому етапі, однак має суттєві обмеження. Зокрема, електронні таблиці не забезпечують повноцінного контролю цілісності даних, автоматизації бізнес-процесів, журналювання дій користувача та зручної роботи зі статусами замовлень. Тому такий підхід може використовуватися лише як тимчасове рішення.

У межах цієї кваліфікаційної роботи за результатами проведеного аналізу підходів до реалізації обрано варіант розробки локальної десктопної CRM-системи. Він є найбільш доцільним, оскільки дозволяє створити автономне програмне забезпечення з простим інтерфейсом, збереженням даних у локальній базі та мінімальними вимогами до апаратного забезпечення. Таке рішення забезпечує баланс між простотою використання, автономністю роботи, достатнім функціоналом і можливістю подальшого розширення системи. Запропонований варіант є оптимальним для невеликих підприємств, яким потрібен доступний інструмент для автоматизації основних бізнес-процесів.

Для реалізації CRM-системи необхідно обрати відповідні інструментальні засоби, які забезпечать ефективну розробку програмного продукту, зручність його використання та можливість подальшого розширення функціоналу.

Основною мовою програмування обрано Python. Такий вибір обумовлений простотою синтаксису, високою швидкістю розробки та наявністю великої кількості бібліотек для створення прикладних програм. Python широко використовується для розробки інформаційних систем, що робить його доцільним вибором для реалізації CRM-системи [15].

Альтернативними варіантами можуть бути мови програмування C# або Java, порівняння наведено в таблиці 1.2. Вони забезпечують високу

продуктивність та широкі можливості для створення складних систем, однак їх використання потребує більш складного налаштування середовища розробки та більших витрат часу на реалізацію. У межах даної роботи Python є оптимальним вибором, оскільки дозволяє швидше реалізувати необхідний функціонал.

Таблиця 1.2 – Порівняння мов програмування

Критерій	Python	C#	Java
Простота синтаксису	Висока	Середня	Середня
Швидкість розробки	Висока	Середня	Середня
Налаштування середовища	Просте	Середнє	Складніше
Продуктивність	Середня	Висока	Висока

Для зберігання даних використано вбудовану систему керування базами даних SQLite. Основною перевагою SQLite є її автономність – база даних зберігається у вигляді одного файлу та не потребує встановлення окремого серверного програмного забезпечення. Це робить її зручною для використання у десктопних застосунках [16, 17].

Альтернативними рішеннями є такі СКБД, як MySQL або PostgreSQL, які забезпечують вищу продуктивність та підтримку багатокористувацького доступу. Проте їх використання є доцільним для великих корпоративних систем. Для локальної десктопної CRM-системи використання SQLite є оптимальним рішенням завдяки простоті інтеграції, автономності та відсутності необхідності адміністрування серверної частини. У межах даної кваліфікаційної роботи пріоритетом є простота розгортання та використання програмного забезпечення.

Використання веб-технологій у межах даної роботи не є доцільним, оскільки розроблювана система орієнтована на локальне використання без необхідності розгортання серверної частини та забезпечення

багатокористувацького доступу через мережу Інтернет. Реалізація десктопного застосунку дозволяє спростити процес використання системи, зменшити вимоги до інфраструктури та забезпечити автономність роботи.

Для реалізації графічного інтерфейсу користувача використано бібліотеку CustomTkinter, яка є розширенням стандартного модуля Tkinter. Вибір бібліотеки обумовлений необхідністю створення сучасного графічного інтерфейсу для десктопного застосунку. Використання CustomTkinter забезпечує зручність взаємодії користувача із системою та підвищує її візуальну привабливість. На відміну від веб-інтерфейсів, такий спосіб реалізації не потребує використання браузера та забезпечує прямий запуск програмного забезпечення на локальному комп’ютері користувача [18, 19].

Стандартні засоби мови Python можуть використовуватися для організації логіки програми та взаємодії з базою даних, що дозволяє зменшити залежність від сторонніх бібліотек і спрощує підтримку програмного забезпечення.

1.4 Постановка задачі та формування вимог до програмного забезпечення

На основі проведеного аналізу предметної області та існуючих CRM-систем визначено необхідність розробки програмного забезпечення, призначеного для автоматизації основних бізнес-процесів малого підприємства, зокрема управління клієнтами та замовленнями.

Основною задачею є створення CRM-системи, яка дозволяє централізовано зберігати інформацію про клієнтів, вести облік замовлень, контролювати їх виконання, а також формувати звітність для аналізу діяльності підприємства. Система повинна бути простою у використанні, не вимагати складного налаштування та забезпечувати зручність і ефективність роботи користувача.

Відповідно до поставленої задачі сформовано функціональні та нефункціональні вимоги до програмного забезпечення. Необхідність

формування вимог на етапі проєктування програмних систем обґрунтовано в [20, с. 138].

Функціональні вимоги до розроблюваної CRM-системи включають:

– забезпечення авторизації користувача для обмеження доступу до системи;

– можливість додавання, редагування, пошуку та видалення інформації про клієнтів;

– збереження основних даних клієнтів, зокрема імені, контактної інформації (телефон, електронна пошта);

– можливість створення, редагування та видалення замовлень;

– збереження інформації про замовлення, включаючи опис, дату створення, кінцевий термін виконання, статус та вартість;

– реалізація механізму зміни статусів замовлень;

– автоматичне визначення прострочених замовлень на основі кінцевого терміну виконання;

– реалізація пошуку замовлень за ключовими параметрами (клієнт, опис);

– формування звітності щодо діяльності підприємства;

– ведення журналу дій користувача для фіксації змін у системі.

Нефункціональні вимоги вимоги до розроблюваної CRM-системи включають:

– простота та зручність користувацького інтерфейсу;

– швидкість обробки даних та виконання операцій;

– надійність зберігання інформації;

– автономність роботи системи без обов’язкового підключення до мережі Інтернет;

– можливість подальшого розширення функціоналу;

– мінімальні вимоги до апаратного забезпечення.

У межах розробки передбачається створення модульної структури програмного забезпечення, яка включає окремі компоненти для роботи з базою даних, реалізації бізнес-логіки та користувацького інтерфейсу. Такий підхід забезпечує зручність супроводу та подальшого розвитку системи [21, с. 21, 22].

Таким чином, сформульовані вимоги визначають основні напрями розробки CRM-системи та слугують основою для подальшого проєктування і реалізації програмного забезпечення.

Проведений аналіз предметної області дозволив визначити особливості автоматизації бізнес-процесів малого підприємства та обґрунтувати необхідність використання CRM-систем для управління клієнтами і замовленнями. У процесі дослідження розглянуто сутність автоматизації бізнес-процесів, визначено її основні переваги, а також досліджено роль інформаційних систем у підвищенні ефективності діяльності підприємств.

На основі аналізу предметної області та існуючих програмних рішень встановлено, що більшість сучасних CRM-рішень характеризуються складністю використання, надлишковим функціоналом або потребують додаткових фінансових витрат, що ускладнює їх використання у малому бізнесі. Окрім того, проаналізовано основні підходи до реалізації CRM-систем та обґрунтовано доцільність використання десктопного підходу для реалізації програмного продукту в межах даної роботи. За результатами аналізу сформовано функціональні і нефункціональні вимоги до розроблюваної CRM-системи.

Для забезпечення простоти розробки, автономності роботи системи та зручності використання програмного забезпечення вибір здійснено на користь мови програмування Python, системи керування базами даних SQLite та бібліотеки CustomTkinter для реалізації графічного інтерфейсу користувача.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Архітектура системи та обґрунтування вибору технологій

Розробка програмного забезпечення для автоматизації бізнес-процесів підприємства потребує чіткого визначення архітектури системи та обґрунтованого вибору технологій. Архітектура системи визначає структуру програмного продукту, взаємодію його компонентів та забезпечує можливість подальшого розвитку і супроводу [23, с. 3].

Структуру розроблюваної CRM-системи та взаємодію її основних компонентів наведено на рисунку 2.1.

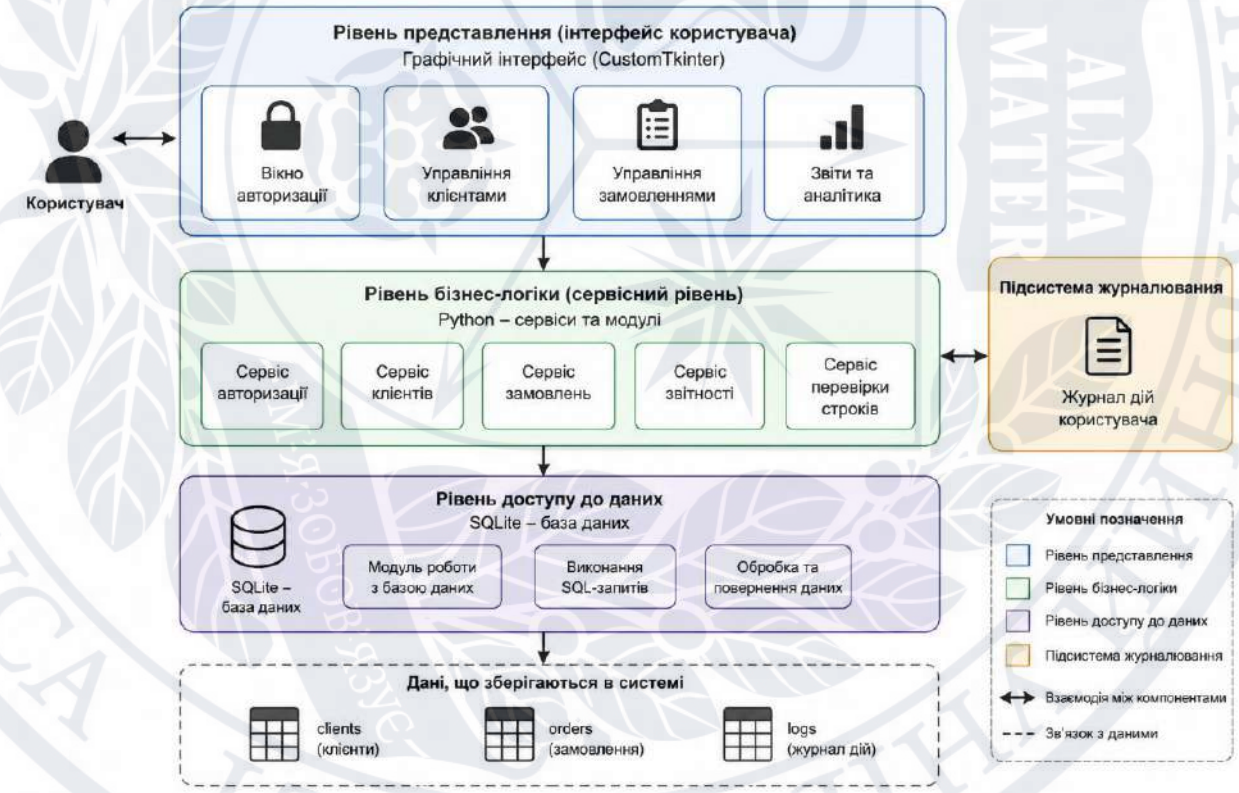

Рисунок 2.1 – Архітектура CRM-системи

Архітектура системи побудована за принципом багаторівневої організації, що передбачає розділення інтерфейсу користувача, бізнес-логіки та рівня

доступу до даних. Кожен із зазначених рівнів виконує окремі функції та взаємодіє з іншими компонентами через визначені інтерфейси [24, с.19; 25].

Рівень інтерфейсу користувача відповідає за відображення даних та обробку дій користувача. Він реалізований за допомогою бібліотеки CustomTkinter і включає всі графічні компоненти системи, зокрема вікна авторизації, управління клієнтами, замовленнями та звітами.

Рівень бізнес-логіки представлений сервісними модулями, які виконують обробку даних, перевірку введеної інформації та реалізацію основних операцій системи. Саме на цьому рівні реалізовано логіку створення замовлень, зміни їх статусів, контролю строків виконання та формування звітності.

Рівень доступу до даних забезпечує взаємодію з базою даних SQLite та реалізує функції підключення до бази, виконання SQL-запитів і отримання результатів. Важливою особливістю є використання універсальних функцій для роботи з даними, що дозволяє мінімізувати дублювання коду.

Окремим компонентом архітектури є підсистема журналювання, яка інтегрована з бізнес-логікою та забезпечує фіксацію дій користувача. Вона дозволяє здійснювати контроль та аналіз ключових операцій, що виконуються у системі.

Такий підхід до побудови архітектури забезпечує гнучкість та зручність супроводу програмного забезпечення. Розділення системи на окремі рівні дозволяє спростити розробку, тестування та подальше розширення функціоналу.

Структура програмного проєкту організована у вигляді окремих каталогів, що забезпечує логічний поділ функціональності:

ui/ – містить компоненти графічного інтерфейсу користувача;

services/ – реалізує бізнес-логіку системи;

database/ – забезпечує взаємодію з базою даних;

data/ – містить файл бази даних;

logs/ – використовується для збереження журналів дій користувача;

reports/ – зберігає сформовані звіти.

Така організація проєкту спрощує супровід системи та навігацію у коді, а також забезпечує можливість незалежного розвитку окремих компонентів програмного забезпечення.

Центральною точкою входу в систему є файл main.py, який ініціалізує базу даних та запускає інтерфейс авторизації. Такий підхід забезпечує чітке розмежування відповідальностей між компонентами системи.

Сервісні модулі реалізують основні операції системи, зокрема управління клієнтами, обробку замовлень, формування звітності та журналювання дій користувача.

Модуль журналювання забезпечує автоматичне збереження інформації про ключові дії користувача у текстових логах та базі даних.

Обрана архітектура та використані інструментальні засоби забезпечують ефективну роботу CRM-системи, зручність супроводу та можливість подальшого розширення функціоналу програмного забезпечення.

2.2 Проєктування бази даних

Ефективне функціонування CRM-системи значною мірою залежить від правильно спроєктованої бази даних [26, с. 20], яка забезпечує збереження, обробку та швидкий доступ до інформації. У межах даної роботи для зберігання даних використано реляційну базу даних SQLite, що є оптимальним рішенням для невеликих програмних систем.

База даних системи складається з трьох основних таблиць: clients, orders та logs, які використовуються для роботи з інформацією про клієнтів, замовлення та дії користувача відповідно.

Для наочного представлення структури бази даних CRM-системи використано ER-діаграму (рис. 2.2), яка відображає основні сутності системи, їх атрибути та зв’язки між ними [27; 28].

ER-діаграма бази даних відображає три основні сутності: клієнти, замовлення та журнал дій. Центральною сутністю є таблиця orders, оскільки саме вона пов’язує інформацію про клієнтів із процесом виконання замовлень. Таблиця clients зберігає основні контактні дані клієнтів, а таблиця logs використовується для фіксації дій користувача в системі.

Зв’язок між таблицями clients та orders реалізовано за допомогою зовнішнього ключа client_id, який зберігається у таблиці замовлень. Один клієнт може мати декілька замовлень, проте кожне замовлення належить лише одному клієнту. Такий зв’язок відповідає типу «один до багатьох» і забезпечує цілісність даних у системі.

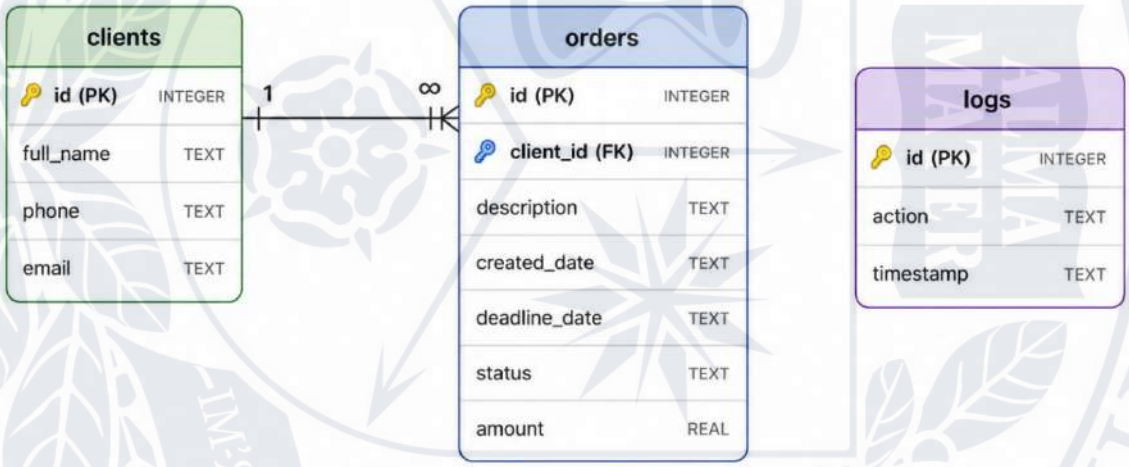

Рисунок 2.2 – ER-діаграма бази даних CRM-системи

Таблиця logs призначена для збереження інформації про виконані дії користувача у системі. Вона дозволяє реєструвати ключові операції, зокрема створення, редагування та видалення записів. Таблиця не є безпосередньо пов’язаною із конкретним клієнтом або замовленням через зовнішній ключ, однак логічно пов’язана з роботою всієї системи, оскільки містить інформацію про зміни, що вносяться до бази даних.

ER-діаграма наочно відображає організацію збереження даних у системі та взаємозв’язки між основними таблицями бази даних. Такий підхід спрощує подальшу реалізацію SQL-запитів і забезпечує структуроване зберігання інформації.

Для детальнішого опису структури бази даних подано характеристику кожної таблиці із зазначенням назв полів, типів даних та їх призначення:

Таблиця clients (таблиця 2.1) використовується для збереження основної інформації про клієнтів підприємства. Поле id є первинним ключем і забезпечує унікальність кожного запису. Поле full_name є обов’язковим, оскільки ідентифікує клієнта у системі. Поля phone та email використовуються для збереження контактної інформації та можуть застосовуватися для подальшої комунікації з клієнтом.

Таблиця 2.1 – Структура таблиці clients

Поле	Тип даних	Призначення
id	INTEGER	Унікальний ідентифікатор клієнта, первинний ключ
full_name	TEXT	Повне ім’я клієнта
phone	TEXT	Контактний номер телефону клієнта
email	TEXT	Електронна адреса клієнта

Таблиця orders (таблиця 2.2) є центральною таблицею бази даних, оскільки вона зберігає інформацію про замовлення підприємства. Поле client_id забезпечує зв’язок із таблицею clients, що дозволяє визначити, якому клієнту належить конкретне замовлення. Поля created_date та deadline_date використовуються для контролю строків виконання. Поле status відображає поточний стан замовлення, а поле amount використовується для збереження його вартості та подальшого формування фінансової звітності.

Таблиця 2.2 – Структура таблиці orders

Поле	Тип даних	Призначення
id	INTEGER	Унікальний ідентифікатор замовлення, первинний ключ
client_id	INTEGER	Ідентифікатор клієнта, зовнішній ключ
description	TEXT	Опис замовлення
created_date	TEXT	Дата створення замовлення
deadline_date	TEXT	Кінцевий термін виконання замовлення
status	TEXT	Поточний статус замовлення
amount	REAL	Вартість замовлення

Окремим елементом бази даних є таблиця logs, структура якої наведена у таблиці 2.3. Поле action містить опис дії, виконаної користувачем, а поле timestamp використовується для збереження дати та часу її виконання.

Таблиця 2.3 – Структура таблиці logs

Поле	Тип даних	Призначення
id	INTEGER	Унікальний ідентифікатор запису журналу, первинний ключ
action	TEXT	Опис виконаної дії користувача
timestamp	TEXT	Дата та час виконання дії

Таким чином, спроєктована структура бази даних є логічною, цілісною та достатньою для реалізації функціональних можливостей CRM-системи. Таблиці clients, orders та logs забезпечують ефективну роботу з даними клієнтів, замовлень і журналу дій користувача.

Усі таблиці створюються автоматично під час запуску програми за допомогою відповідних SQL-запитів. Крім того, у процесі ініціалізації бази даних передбачена перевірка наявності полів у таблиці замовлень, що забезпечує

можливість подальшого розширення структури бази даних без втрати існуючих даних.

2.3 Проєктування інтерфейсу користувача

Інтерфейс користувача є важливою складовою програмного забезпечення, оскільки саме через нього здійснюється взаємодія користувача із системою. Основною метою проєктування інтерфейсу є забезпечення зручності використання, інтуїтивної зрозумілості та ефективності виконання основних операцій [29, с. 227; 30].

На рисунку 2.3 зображено вікно авторизації, яке є першим елементом взаємодії користувача із системою. Воно містить поле введення пароля та кнопку підтвердження входу до системи. Основною функцією цього вікна є перевірка прав доступу та обмеження несанкціонованого використання системи.

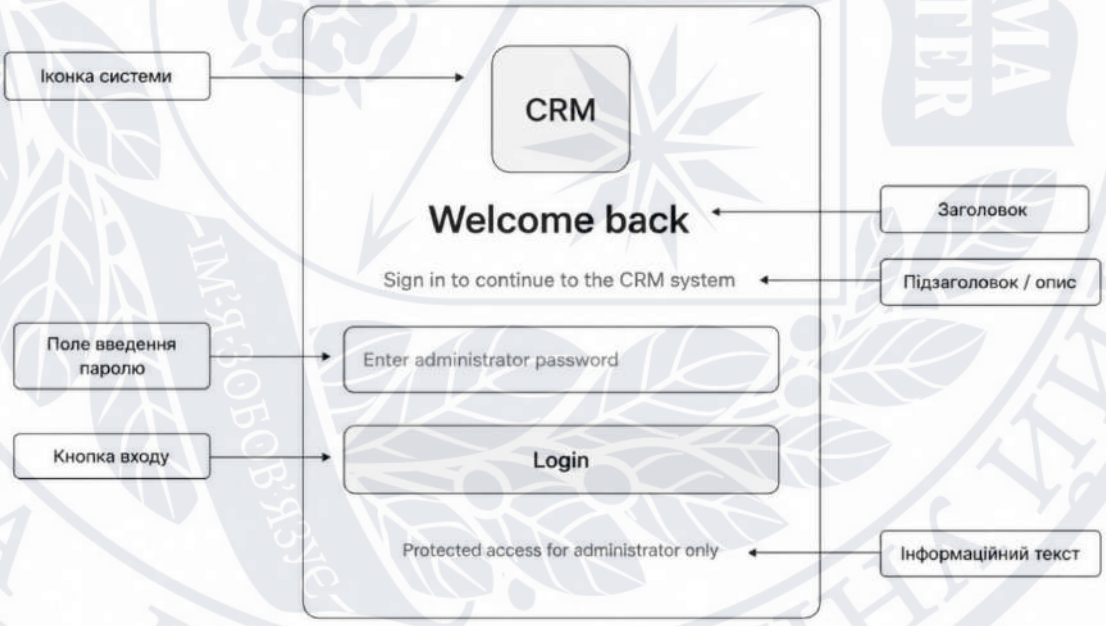

Рисунок 2.3 – Вікно авторизації CRM-системи

Після успішної авторизації відкривається головне вікно системи, яке виконує роль центральної панелі керування. Воно складається з двох основних частин:

– бічної панелі навігації;
– робочої області для відображення вмісту.

Бічна панель містить кнопки переходу між основними модулями системи: управління клієнтами, управління замовленнями та формування звітів, що забезпечує швидкий доступ до функціональних можливостей системи.

Вікно управління клієнтами (рис. 2.4) призначене для роботи з інформацією про клієнтів. Воно містить форми введення даних, кнопки для виконання основних операцій (додавання, редагування, очищення, видалення, пошуку та відображення всіх записів), а також таблицю для відображення списку клієнтів. Поля введення дозволяють користувачу вносити основну інформацію про клієнта, зокрема ім’я, номер телефону та електронну адресу. Табличне представлення даних забезпечує зручний перегляд усіх записів та швидкий доступ до їх редагування або видалення. Розташування елементів інтерфейсу дозволяє виконувати всі основні операції в межах одного вікна, що підвищує зручність роботи користувача та спрощує взаємодію із системою.

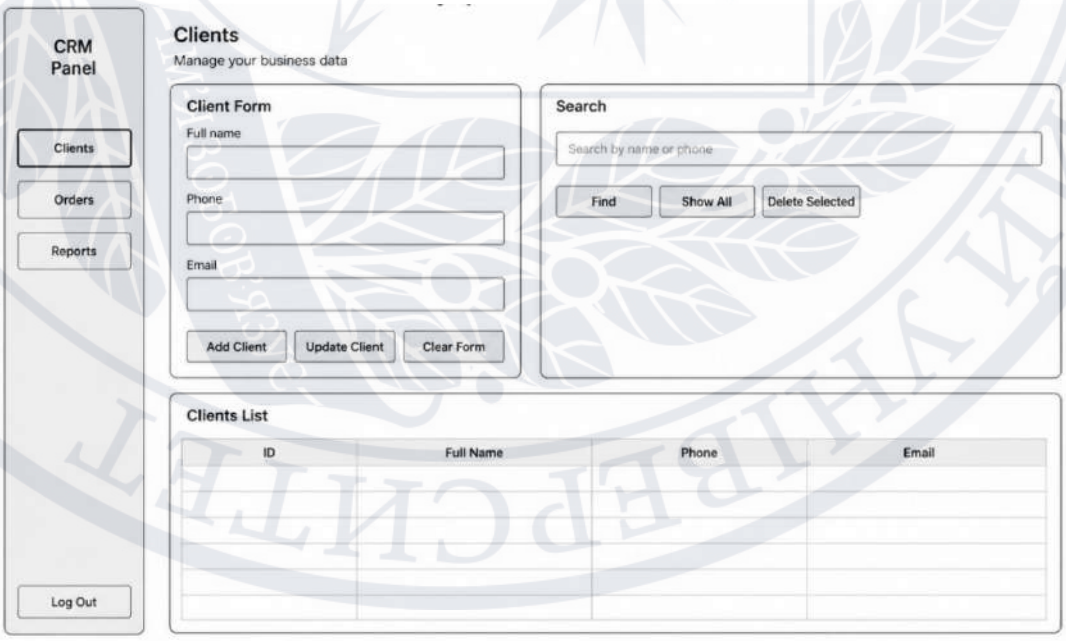

Рисунок 2.4 – Вікно управління клієнтами

Вікно управління замовленнями (рис. 2.5) реалізовано у вигляді двоколонкової структури, де ліва частина містить форму введення та пошуку, а права – список та детальну інформацію про замовлення. Такий підхід дозволяє одночасно переглядати інформацію та виконувати операції над даними.

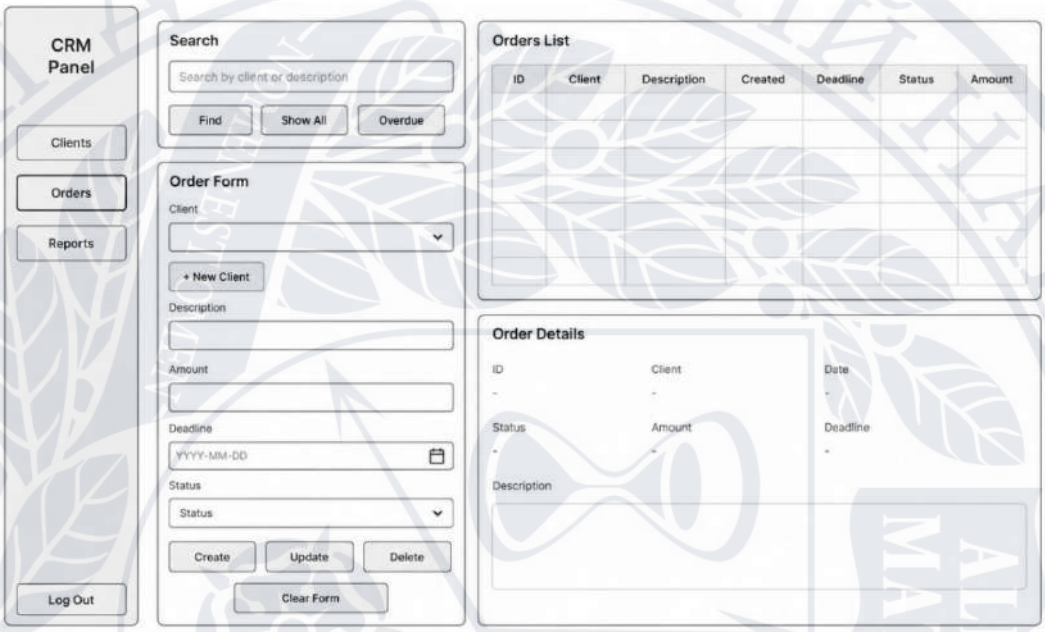

Рисунок 2.5 – Вікно управління замовленнями

Особливістю цього вікна є наявність механізму автодоповнення для вибору клієнта, можливість вибору статусу замовлення, введення вартості та опису, а також встановлення кінцевого строку виконання замовлення. У даному вікні розташовані кнопки для виконання операцій над замовленнями (пошук, відображення всіх замовлень, перевірка на простроченість, створення, оновлення інформації про замовлення, видалення, очищення форми) та кнопка, що забезпечує можливість додавання клієнта в разі його відсутності в базі даних.

Вікно звітності (рис. 2.6) призначене для відображення узагальненої інформації про діяльність підприємства. Воно містить елементи для вибору періоду, відображення статистичних даних та формування звітів. Реалізація цього вікна дозволяє користувачу аналізувати результати роботи та приймати управлінські рішення. У вікні розташовані елементи вибору періоду, кнопки

керування звітами (генерація звіту, експорт у формати TXT та CSV), блоки статистичної інформації щодо статусів замовлень, графічне відображення фінансових показників та область відображення сформованого звіту.

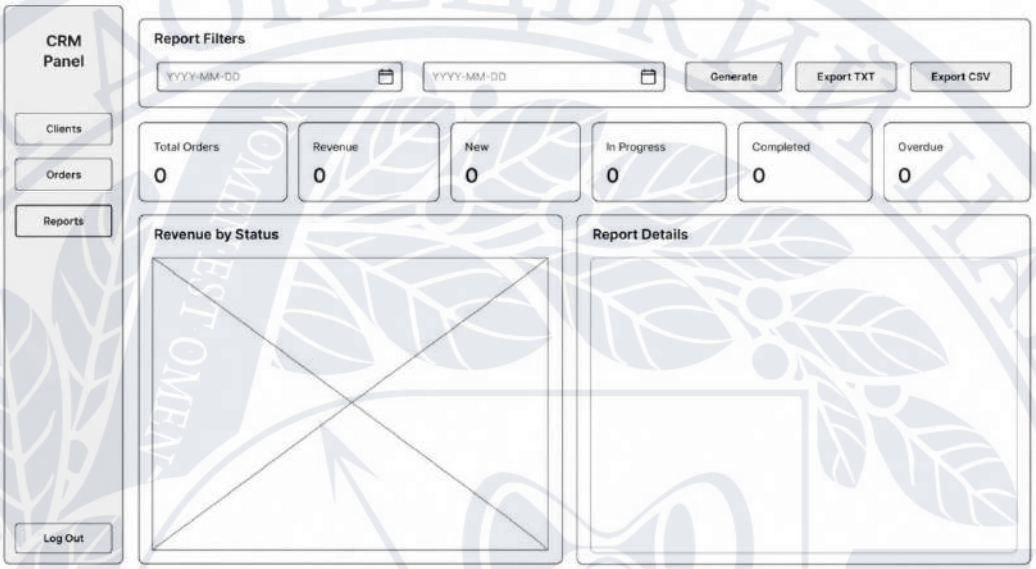

Рисунок 2.6 – Вікно звітності

Використання графічних макетів інтерфейсу дозволяє не лише наочно представити структуру системи, але й обґрунтувати вибір розташування елементів керування з точки зору зручності користувача. Запропонований інтерфейс забезпечує логічну організацію елементів керування системи, що сприяє ефективній взаємодії користувача з програмним забезпеченням.

Під час проектування інтерфейсу особливу увагу приділено:

– використанню єдиного стилю оформлення;

– розташуванню елементів управління;

– забезпеченню зручності взаємодії користувача із системою;

– зменшенню кількості дій, необхідних для виконання операцій [31, с. 562].

Таким чином, розроблений інтерфейс користувача забезпечує ефективну взаємодію з CRM-системою, є зручним у використанні та відповідає сучасним вимогам до програмного забезпечення.

2.4 Моделювання основних бізнес-процесів та алгоритмів системи

У процесі розробки CRM-системи важливим етапом є моделювання основних бізнес-процесів, які визначають логіку взаємодії користувача із системою та послідовність виконання операцій. У розробленій CRM-системі виділено основні бізнес-процеси, що забезпечують її функціонування, зокрема: авторизація користувача, управління клієнтами, управління замовленнями, контроль строків виконання та формування звітності. Для наочного представлення процесів використано блок-схеми, які відображають структуру та послідовність дій [32].

Процес авторизації (рис. 2.7) починається з відкриття вікна входу в систему та введення користувачем пароля у відповідне поле. Після цього система виконує перевірку правильності введених даних. Якщо введений пароль відповідає встановленому значенню, відбувається успішна авторизація та відкриття головного вікна системи. У разі введення неправильного пароля система відображає повідомлення про помилку та надає можливість повторного введення даних. Такий підхід забезпечує базовий контроль доступу до функціональних можливостей CRM-системи.

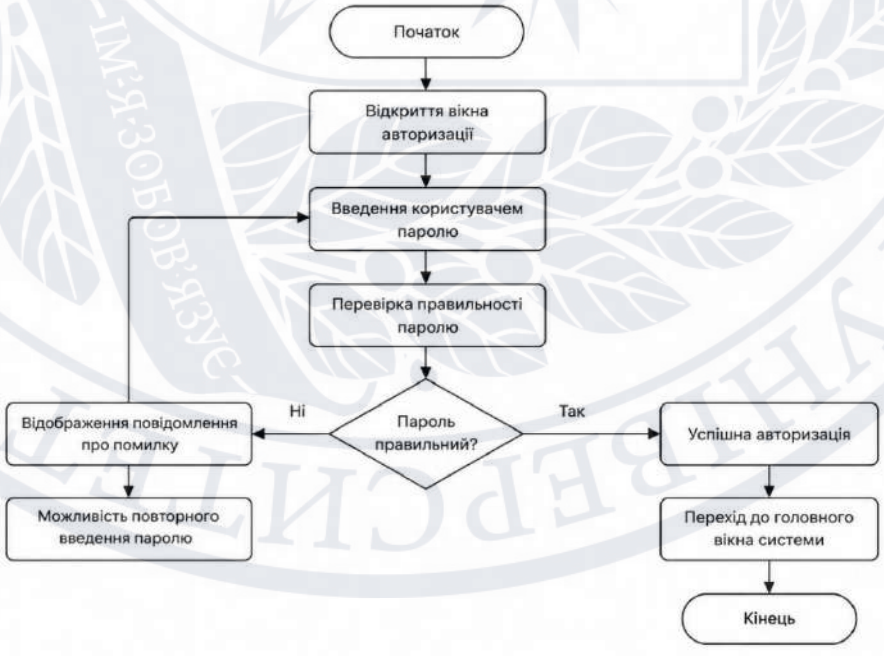

Рисунок 2.7 – Бізнес-процес авторизації користувача

Процес управління клієнтами (рис. 2.8) починається з відкриття відповідного модуля та відображення списку клієнтів. Після цього користувач обирає необхідну дію: додавання нового клієнта, редагування наявного запису або видалення клієнта. У разі додавання чи редагування запису система отримує введені дані та виконує перевірку їх коректності. Якщо дані відповідають встановленим вимогам, зміни зберігаються у базі даних, а список клієнтів автоматично оновлюється. У випадку виявлення помилки відображається відповідне повідомлення, після чого користувачу надається можливість повторного введення даних. Така організація процесу забезпечує цілісність даних та спрощує роботу з клієнтською інформацією.

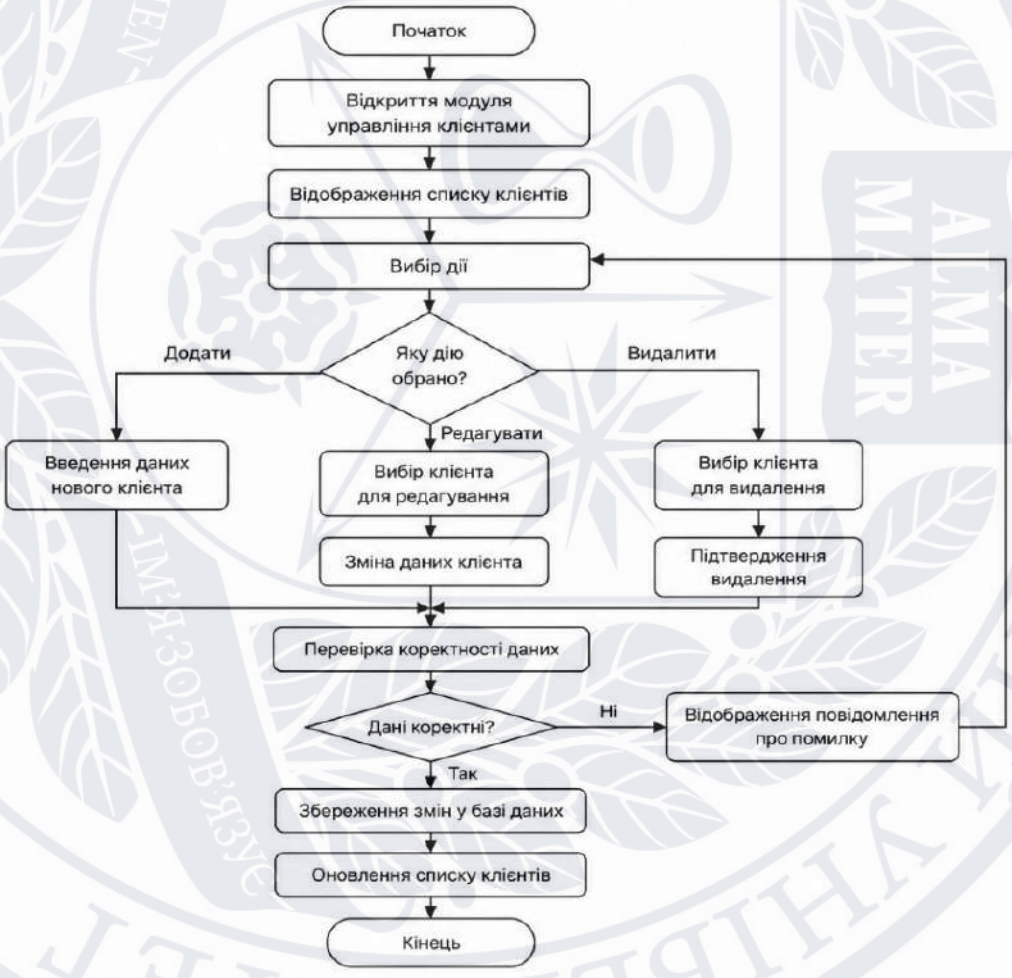

Рисунок 2.8 – Бізнес-процес управління клієнтами

Процес управління замовленнями (рис. 2.9) передбачає відкриття відповідного модуля та відображення списку наявних записів. Користувач має

можливість створювати нові замовлення, редагувати інформацію про існуючі замовлення або змінювати їх статус. Під час створення нового замовлення вводяться відомості про клієнта, опис послуги, вартість та строк виконання. У разі зміни статусу користувач обирає необхідний варіант із переліку доступних станів: «Нове», «В роботі», «Виконано» або «Прострочено». Після перевірки коректності введених даних система зберігає зміни у базі та оновлює список замовлень для відображення актуальної інформації. Якщо дані введено некоректно, система відображає повідомлення про помилку.

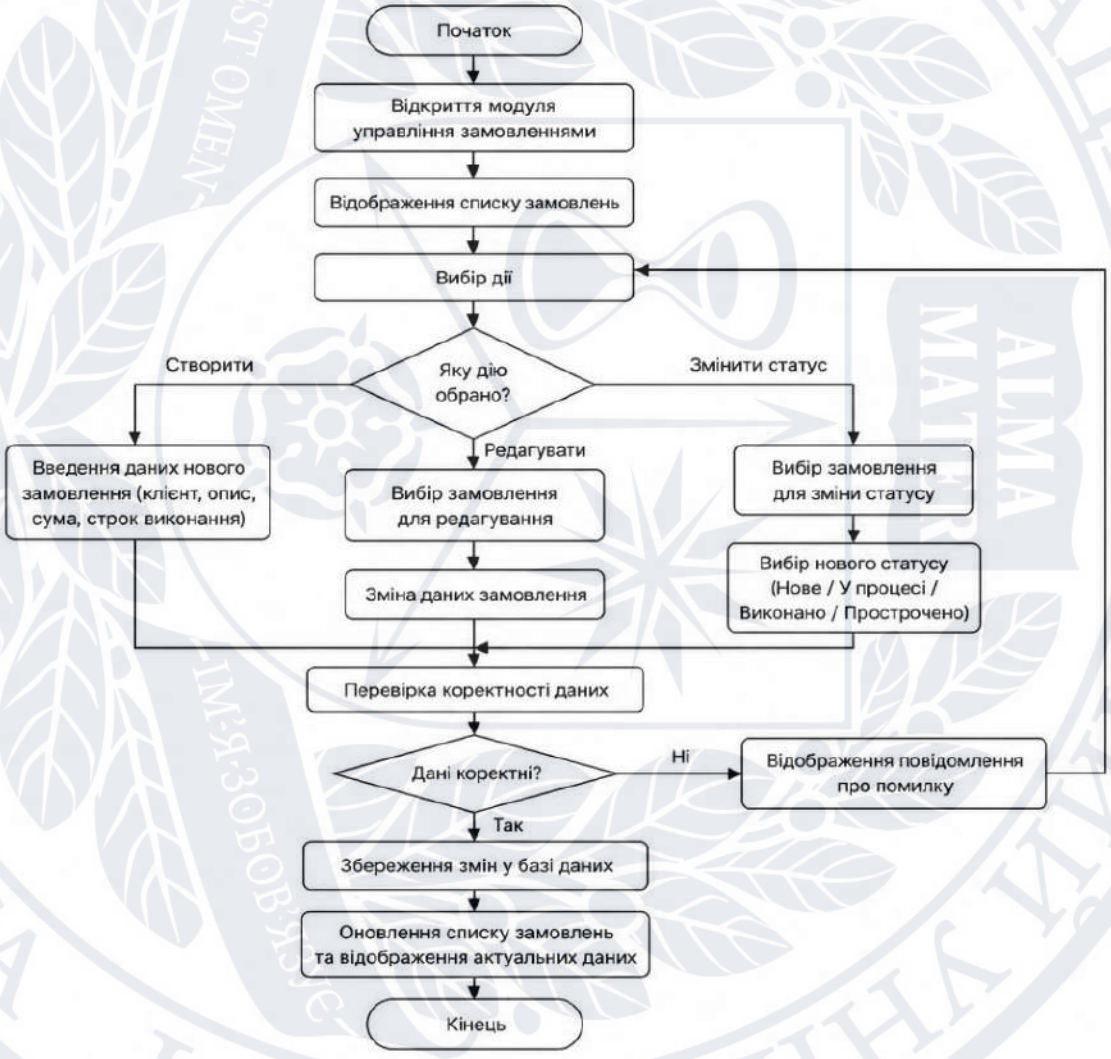

Рисунок 2.9 – Бізнес-процес управління замовленнями

Підсистема контролю строків виконання (рис. 2.10) забезпечує автоматичну перевірку актуальності замовлень. Після запуску перевірки система отримує список замовлень із бази даних та послідовно виконує їх аналіз. Для

кожного запису здійснюється порівняння поточної дати з кінцевим строком виконання. Якщо термін виконання перевищено, а замовлення не має статусу «Виконано», система автоматично змінює його статус на «Прострочено» та додає відповідний запис до переліку прострочених замовлень. Після завершення перевірки внесені зміни зберігаються у базі даних, а за потреби формується звіт щодо прострочених замовлень. Такий механізм дозволяє автоматизувати контроль виконання замовлень та своєчасно виявляти проблемні записи.

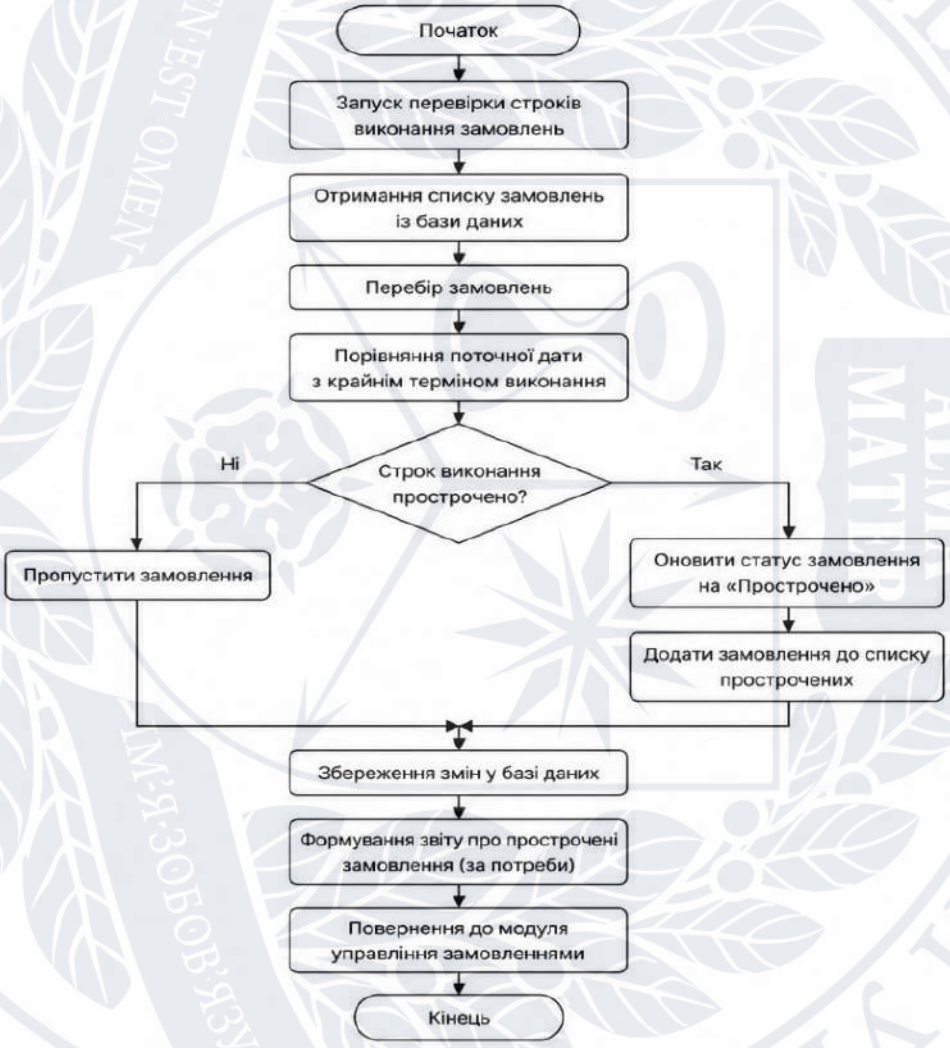

Рисунок 2.10 – Підсистема контролю строків виконання

Процес формування звітності (рис. 2.11) починається з відкриття модуля звітності та вибору параметрів формування звіту, зокрема періоду аналізу. На основі заданих параметрів система формує SQL-запит до бази даних та отримує необхідні дані. Після успішного отримання даних система формує звіт та

відображає його на екрані. Користувач має можливість експортувати сформований звіт у формати TXT або CSV для подальшого використання. У разі виникнення помилки під час отримання даних система відображає відповідне повідомлення та надає можливість повторного вибору параметрів звіту. Такий підхід забезпечує гнучкість формування звітності та спрощує аналіз діяльності підприємства.

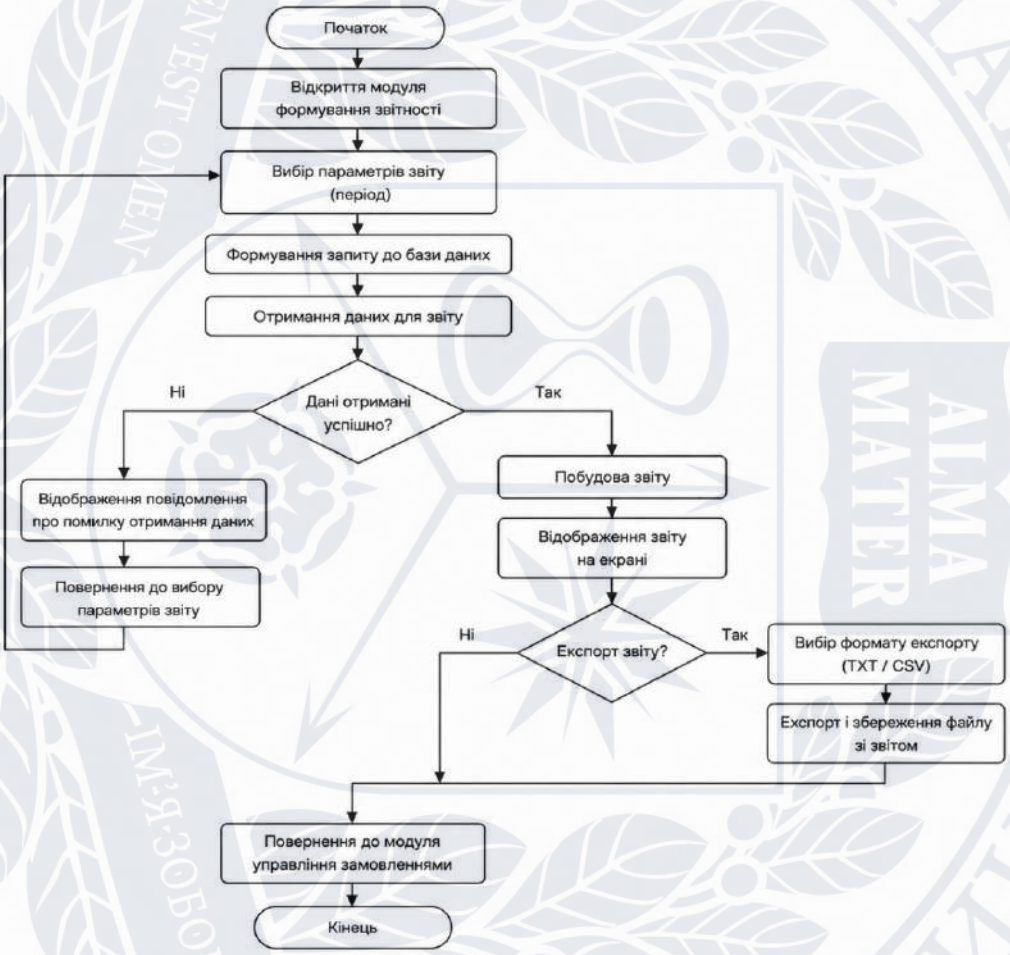

Рисунок 2.11 – Бізнес-процес формування звітності

Використання блок-схем дозволяє чітко представити основні бізнес-процеси системи, визначити послідовність виконання операцій та взаємодію між її компонентами. Це сприяє кращому розумінню логіки роботи CRM-системи та

спрощує подальшу реалізацію, тестування та супровід програмного забезпечення.

Алгоритмічне забезпечення є важливою складовою проєктування програмної системи, оскільки визначає логіку обробки даних, умови виконання операцій та послідовність взаємодії між окремими компонентами системи [33, 34]. У межах розробленої CRM-системи використовуються алгоритми для реалізації механізмів авторизації користувача, управління клієнтами та замовленнями, контролю строків виконання і формування звітності. Основні алгоритми системи базуються на перевірці вхідних даних, виконанні логічних умов та взаємодії з базою даних SQLite.

Алгоритм авторизації користувача реалізує механізм перевірки введеного пароля та надання доступу до функціональних можливостей системи лише у разі успішного проходження перевірки. Після отримання введеного значення система виконує його порівняння із встановленим паролем адміністратора. Якщо перевірка виконується успішно, користувач отримує доступ до головного вікна системи. У разі невідповідності введених даних система відображає повідомлення про помилку та надає можливість повторного введення пароля.

Основні етапи алгоритму авторизації:

- отримання введеного користувачем пароля;
- порівняння введеного значення з паролем адміністратора;
- надання або обмеження доступу до системи;
- обробка помилки у разі неправильного введення даних.

Умову авторизації можна подати у вигляді логічного виразу:

access = True, якщо entered_password = ADMIN_PASS

access = False, якщо entered_password ≠ ADMIN_PASS

Такий алгоритм забезпечує базовий контроль доступу до функціоналу CRM-системи та запобігає несанкціонованому використанню програми.

Алгоритм управління клієнтами забезпечує виконання операцій додавання, редагування, пошуку та видалення записів. Основною умовою виконання операцій є перевірка коректності введених даних перед їх збереженням у базі

даних. Після отримання інформації з форми система перевіряє наявність обов’язкового поля full_name. Якщо перевірка виконується успішно, формується відповідний SQL-запит та виконується оновлення таблиці clients. У разі виявлення помилки система повідомляє користувача про некоректність введених даних.

Основні етапи алгоритму управління клієнтами:

- отримання даних із форми введення;
- перевірка коректності введених даних;
- формування SQL-запиту;
- додавання, оновлення або видалення запису;
- оновлення списку клієнтів у системі.

Умова перевірки даних клієнта має вигляд:

valid_client = True, якщо full_name ≠ «»

valid_client = False, якщо full_name = «»

Це дозволяє уникнути створення порожніх або некоректних записів у базі даних.

Механізм управління замовленнями є одним із ключових компонентів системи, оскільки забезпечує створення, редагування, пошук, видалення та зміну статусів замовлень. Після отримання введених даних система виконує перевірку їх коректності, формує дату створення замовлення та виконує відповідний SQL-запит до таблиці orders. У разі успішного виконання операції список замовлень автоматично оновлюється в інтерфейсі користувача.

Основні етапи алгоритму управління замовленнями:

- отримання даних про замовлення;
- перевірка коректності введеної інформації;
- формування дати створення замовлення;
- збереження або оновлення запису в таблиці orders;
- оновлення списку замовлень.

Для перевірки коректності замовлення використовуються такі логічні умови:

valid_order = True, якщо client_id ≠ NULL і description ≠ «» і amount≥0

valid_order = False, якщо client_id = NULL або description = «» або amount<0

Початковий статус нового замовлення визначається таким чином:

status = «Нове»

У подальшому статус замовлення може змінюватися користувачем відповідно до етапу його виконання.

Алгоритм контролю строків виконання забезпечує автоматичне визначення прострочених замовлень шляхом порівняння поточної дати з кінцевим строком виконання. Система послідовно аналізує записи таблиці orders та перевіряє значення поля deadline_date. Якщо поточна дата перевищує встановлений строк виконання, а замовлення не має статусу «Виконано», система автоматично змінює його статус на «Прострочено».

Основні етапи алгоритму контролю строків виконання:

- отримання списку замовлень із бази даних;
- аналіз значення поля deadline_date;
- порівняння кінцевого строку виконання з поточною датою;
- автоматична зміна статусу замовлення;
- збереження оновлених даних у базі.

Логічну умову визначення простроченого замовлення можна подати так:

overdue = True, якщо current_date > deadline_date і status ≠ «Виконано»

overdue = False, якщо current_date ≤ deadline_date або status = «Виконано»

Такий алгоритм дозволяє автоматизувати контроль за виконанням замовлень та своєчасно виявляти проблемні записи.

Алгоритм формування звітності забезпечує отримання статистичної та фінансової інформації про діяльність підприємства за обраний період. На основі заданих параметрів система формує SQL-запит до бази даних, виконує обробку отриманих даних та формує підсумковий звіт. Після завершення обчислень результати відображаються у вікні звітності та можуть бути експортовані у файл.

Основні етапи алгоритму формування звітності:

- отримання параметрів формування звіту;
- виконання SQL-запиту до бази даних;
- обробка отриманих даних;
- обчислення статистичних показників;
- формування та відображення звіту;
- експорт результатів у файл.

Для підрахунку загальної суми замовлень використовується формула:

$$S = \sum_{i=1}^{n} amount_i$$

де S – загальна сума замовлень за вибраний період, $amount_i$ – вартість окремого замовлення.

Кількість замовлень за певним статусом визначається так:

$$N_{status} = count(orders\ where\ status = selected\_status)$$

де N_{status}– кількість замовлень із відповідним статусом.

Застосування наведених алгоритмів дозволяє формалізувати логіку роботи CRM-системи та забезпечити послідовну реалізацію її функціональних можливостей.

Виконане проєктування CRM-системи дозволило сформувати цілісну структуру програмного забезпечення для автоматизації бізнес-процесів підприємства. У межах розділу спроєктовано архітектуру системи, що базується на розподілі функціоналу між рівнями інтерфейсу користувача, бізнес-логіки та доступу до даних. Такий підхід забезпечує модульність програмного забезпечення, спрощує супровід системи та створює можливість подальшого розширення її функціональних можливостей.

У процесі проєктування бази даних визначено основні сутності системи, їх атрибути та взаємозв’язки. Розроблена структура бази даних забезпечує збереження інформації про клієнтів, замовлення та дії користувача, а також підтримує цілісність і впорядкованість даних у системі.

Окрему увагу приділено проєктуванню інтерфейсу користувача, який забезпечує зручну взаємодію із системою та швидкий доступ до її функціональних можливостей. Запропонована структура інтерфейсу дозволяє виконувати основні операції в межах окремих функціональних модулів, що підвищує ефективність роботи користувача.

Також у межах розділу змодельовано основні бізнес-процеси та алгоритми функціонування CRM-системи. Використання блок-схем дозволило формалізувати логіку роботи програмного забезпечення, визначити послідовність обробки даних та взаємодію між компонентами системи. Отримані результати створюють основу для подальшої програмної реалізації CRM-системи та її практичного використання.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ CRM-СИСТЕМИ

3.1 Реалізація структури програмного проєкту

Розроблена CRM-система реалізована у вигляді модульного програмного проєкту. Кожен модуль виконує окрему функцію та відповідає за визначену частину логіки роботи системи.

Відповідно до рекомендацій щодо проєктування програмного забезпечення структура проєкту повинна забезпечити розмежування компонентів користувацького інтерфейсу, бізнес-логіки, роботи з базою даних та службових функцій системи [35, с. 25]. З огляду на це структура проєкту наведена на рисунку 3.1.

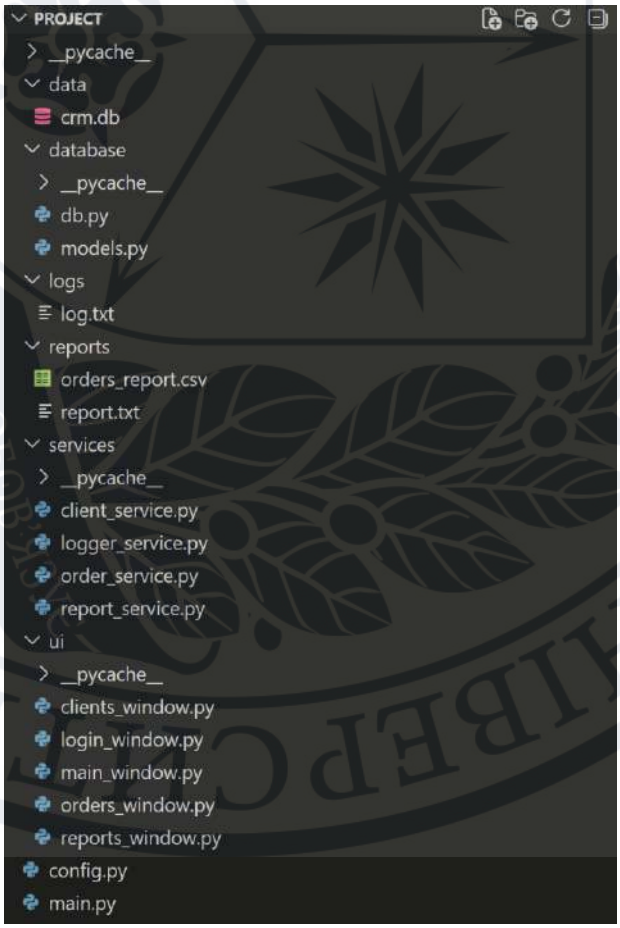

Рисунок 3.1 – Загальна структура проєкту

Початковою точкою запуску програмного забезпечення є файл main.py, який виконує ініціалізацію системи та запуск головного вікна програми. У процесі запуску відбувається підключення до бази даних, створення необхідних таблиць та ініціалізація основних програмних модулів.

Файл config.py містить основні конфігураційні параметри програмного забезпечення, зокрема шлях до файлу бази даних та службові налаштування системи. Використання окремого конфігураційного модуля дозволяє централізовано керувати параметрами програми та спрощує процес внесення змін до налаштувань.

Для роботи з базою даних у структурі проєкту використовується каталог database, який містить модулі db.py та models.py. Модуль db.py забезпечує підключення до бази даних SQLite та виконання SQL-запитів, тоді як models.py містить SQL-конструкції для створення таблиць та опису структури бази даних. Такий підхід дозволяє відокремити логіку роботи із базою даних від інших компонентів системи.

Каталог services містить модулі бізнес-логіки CRM-системи. Зокрема, client_service.py реалізує роботу з клієнтами, order_service.py – обробку замовлень, report_service.py – формування звітності, а logger_service.py – журналювання дій користувача.

Каталог ui містить модулі графічного інтерфейсу користувача, реалізовані за допомогою бібліотеки CustomTkinter. Кожне окреме вікно системи винесено у відповідний файл, що дозволяє забезпечити незалежність компонентів інтерфейсу та спрощує їх модифікацію. Зокрема, модуль login_window.py реалізує вікно авторизації користувача, main_window.py – головне вікно системи, clients_window.py – модуль управління клієнтами, orders_window.py – модуль роботи із замовленнями, а reports_window.py – модуль формування звітності.

Окремими каталогами у структурі проєкту є data, reports та logs. Каталог data використовується для збереження файлу бази даних crm.db, у якому міститься вся інформація про клієнтів, замовлення та журнал дій користувача.

Використання окремого каталогу для бази даних дозволяє централізовано організувати зберігання інформації та спрощує резервне копіювання даних.

Каталог reports призначений для збереження сформованих системою звітів у форматах TXT та CSV. У процесі роботи користувач має можливість експортувати результати аналізу діяльності підприємства у файли для подальшого перегляду або використання в інших програмних засобах. Такий підхід забезпечує зручність роботи зі звітною інформацією та дозволяє зберігати результати роботи системи поза межами інтерфейсу програми.

Каталог logs використовується для ведення журналу дій користувача. У файлі log.txt зберігається інформація про виконані операції в системі, зокрема створення, редагування та видалення записів. Реалізація механізму журналювання дозволяє контролювати зміни у системі та підвищує надійність роботи програмного забезпечення.

Використання модульної структури програмного проєкту забезпечує розділення відповідальності між окремими компонентами системи. Графічний інтерфейс відповідає за взаємодію з користувачем, модулі бізнес-логіки виконують обробку даних та реалізацію функціональних можливостей системи, а модулі роботи з базою даних забезпечують збереження та отримання інформації. Такий підхід дозволяє спростити тестування окремих компонентів, підвищує зручність супроводу програмного забезпечення та забезпечує можливість подальшого розширення функціоналу CRM-системи.

Таким чином, реалізована структура програмного проєкту забезпечує логічну організацію компонентів системи, спрощує процес розробки та створює основу для ефективної реалізації функціональних можливостей CRM-системи.

3.2 Реалізація роботи з базою даних

Робота з базою даних у CRM-системі реалізована у модулі db.py, який містить функції підключення до бази даних, виконання SQL-запитів та

отримання результатів. Для підключення до SQLite використовується функція `connect()`, яка створює з'єднання з файлом бази даних відповідно до шляху, визначеного у файлі config.py.

Створення таблиць виконується функцією `create_tables()`, яка викликається під час запуску програмного забезпечення. У процесі роботи функція послідовно виконує SQL-запити створення таблиць clients, orders та logs, визначені у модулі models.py.

Особливістю реалізації є автоматична перевірка наявності поля deadline_date у таблиці orders. Для цього використовується SQL-команда `PRAGMA table_info(orders)`, яка дозволяє отримати інформацію про структуру таблиці. Якщо поле відсутнє, система автоматично виконує SQL-запит `ALTER TABLE` для його додавання. Такий підхід дозволяє змінювати структуру бази даних без втрати вже збережених даних.

Для виконання SQL-запитів типу `INSERT`, `UPDATE` та `DELETE` використовується універсальна функція `execute_query()`. Використання єдиної функції для виконання змін у базі даних дозволяє уникнути дублювання коду та спрощує реалізацію взаємодії програмних модулів із базою даних.

Отримання даних реалізовано за допомогою функцій `fetch_all()` та `fetch_one()`. Функція `fetch_all()` використовується для отримання списку записів відповідно до параметрів SQL-запиту, тоді як `fetch_one()` дозволяє отримати окремий запис із бази даних.

SQL-конструкції створення таблиць винесені до окремого модуля models.py. Для створення таблиць використовується команда `CREATE TABLE IF NOT EXISTS`, що дозволяє уникнути помилок при повторному запуску системи та забезпечує коректну ініціалізацію бази даних.

Після виконання кожної операції з базою даних підключення автоматично закривається, що дозволяє оптимізувати використання ресурсів системи та забезпечує стабільну роботу програмного забезпечення.

Реалізований механізм роботи з базою даних забезпечує централізовану взаємодію програмних модулів із SQLite та спрощує виконання основних операцій з даними у CRM-системі.

3.3 Реалізація основних функціональних можливостей системи

Реалізація основних функціональних можливостей CRM-системи здійснюється на рівні сервісних модулів, які відповідають за обробку даних та виконання операцій над основними сутностями системи. Використання окремих модулів для реалізації бізнес-логіки дозволяє відокремити обробку даних від графічного інтерфейсу користувача та спрощує взаємодію між компонентами програмного забезпечення.

Модуль client_service.py забезпечує реалізацію операцій роботи з клієнтами. У межах модуля реалізовано додавання, редагування, пошук та видалення інформації про клієнтів. Додавання нового клієнта виконується шляхом формування SQL-запиту `INSERT INTO clients`, після чого введені дані зберігаються у базі даних. Після успішного виконання операції система автоматично створює запис у журналі дій користувача (рис. 3.2).

```
2026-05-21 13:22:09 - Замовлення ID=9 автоматично позначено як 'Прострочено'
2026-05-22 13:52:18 - Оновлено клієнта ID=8:Анна Трінчер
2026-05-22 13:54:05 - Оновлено замовлення ID=7
```

Рисунок 3.2 – Записи у журналі дій користувача

Для пошуку клієнтів використовується SQL-оператор `LIKE`, який дозволяє здійснювати пошук за частковим збігом значень [36]. Пошук виконується за ім’ям клієнта або номером телефону, що забезпечує швидке отримання необхідної інформації навіть при неповному введенні даних. Оновлення інформації про клієнта здійснюється шляхом зміни відповідних полів запису за його

ідентифікатором, а видалення супроводжується попереднім отриманням інформації про клієнта для подальшого журналювання виконаної операції.

Модуль order_service.py реалізує функціонал роботи із замовленнями. Під час створення нового замовлення система автоматично формує дату створення та встановлює початковий статус «Нове». До бази даних зберігаються ідентифікатор клієнта, опис замовлення, дата створення, кінцевий строк виконання, статус та вартість замовлення.

Отримання списку замовлень виконується із використанням SQL-операції `JOIN`, яка дозволяє об’єднати дані таблиць orders та clients. Завдяки цьому у програмному інтерфейсі відображається не лише ідентифікатор клієнта, а і його повне ім’я, що підвищує зручність роботи користувача із системою.

```
SELECT orders.id,
       clients.full_name,
       orders.description,
       orders.created_date,
       orders.deadline_date,
       orders.status,
       orders.amount
FROM orders
JOIN clients ON orders.client_id = clients.id
```

Оновлення замовлень передбачає зміну інформації про клієнта, опису, вартості, кінцевого строку виконання та статусу замовлення. Для пошуку замовлень використовується фільтрація за ім’ям клієнта або описом замовлення.

Важливою частиною модуля є механізм автоматичного контролю строків виконання замовлень. Система отримує із бази даних записи зі статусом «В роботі» та перевіряє відповідність кінцевого строку виконання поточній даті. Якщо встановлений строк перевищено, статус замовлення автоматично змінюється на «Прострочено», після чого інформація про зміну записується до

журналу дій користувача. Такий підхід дозволяє автоматизувати контроль актуальності замовлень та зменшує ймовірність пропуску прострочених записів.

Модуль report_service.py забезпечує формування статистичних даних та звітності щодо діяльності підприємства. У межах модуля реалізовано підрахунок кількості замовлень за статусами, отримання інформації за вибраний період, обчислення фінансових показників та експорт сформованих звітів у зовнішні файли.

Для аналізу діяльності системи використовується обробка замовлень за визначений часовий проміжок. На основі отриманих даних система визначає загальну кількість замовлень, кількість нових, активних, виконаних та прострочених записів, а також обчислює сумарну вартість замовлень. Додатково здійснюється окремий підрахунок фінансових показників відповідно до статусу замовлення, що дозволяє оцінювати ефективність виконання робіт.

Формування звітності реалізовано у вигляді текстового звіту, який містить статистичну та фінансову інформацію за вибраний період. Для підвищення зручності сприйняття грошових значень використовується окремий механізм форматування числових даних, який забезпечує відображення сум із двома десятковими знаками та розділенням розрядів.

Окрім відображення інформації у програмному інтерфейсі, система підтримує експорт звітів у формати TXT та CSV. TXT-файли використовуються для збереження сформованої статистичної інформації, тоді як CSV-формат дозволяє виконувати подальшу обробку даних у сторонніх програмних засобах. Збереження звітів виконується у каталог reports, що забезпечує централізоване зберігання результатів роботи системи.

Модуль logger_service.py реалізує механізм журналювання дій користувача. Під час виконання операцій система автоматично формує запис, який містить опис виконаної дії та часову мітку (рис. 3.2). Отримана інформація одночасно зберігається у таблиці logs бази даних та у текстовому файлі logs/log.txt.

Журналювання використовується під час додавання, оновлення та видалення інформації про клієнтів і замовлення, а також під час автоматичної зміни статусів замовлень. Це дозволяє зберігати історію змін у системі та спрощує контроль виконаних операцій.

Використання окремих сервісних модулів для реалізації бізнес-логіки забезпечує структурованість програмного коду та спрощує супровід програмного забезпечення. Кожен модуль відповідає за окрему функціональну частину системи, що дозволяє зменшити залежність між компонентами та забезпечує можливість подальшого розширення функціональних можливостей CRM-системи.

Отже, реалізація основних функціональних можливостей системи забезпечує виконання операцій над клієнтами та замовленнями, автоматичний контроль строків виконання, формування звітності та журналювання дій користувача, що створює основу для стабільної та ефективної роботи CRM-системи.

3.4 Реалізація графічного інтерфейсу користувача

Графічний інтерфейс користувача CRM-системи реалізовано у вигляді окремих програмних модулів із використанням бібліотеки CustomTkinter. Основними модулями графічного інтерфейсу є login_window.py, main_window.py, clients_window.py, orders_window.py та reports_window.py. Для реалізації окремих елементів інтерфейсу також використовуються бібліотеки tkinter, ttk, matplotlib та tkcalendar.

Вікно авторизації (рис. 3.3) реалізовано у класі `LoginWindow`, який успадковує клас `CTk` бібліотеки CustomTkinter. Під час запуску програми виконується налаштування режиму відображення інтерфейсу за допомогою функцій `set_appearance_mode()` та `set_default_color_theme()`. У межах ініціалізації вікна задаються його розміри, кольорова схема та параметри

відображення. Для центрування вікна на екрані використовується окремий метод `center_window()`, який визначає координати розташування відповідно до роздільної здатності дисплея.

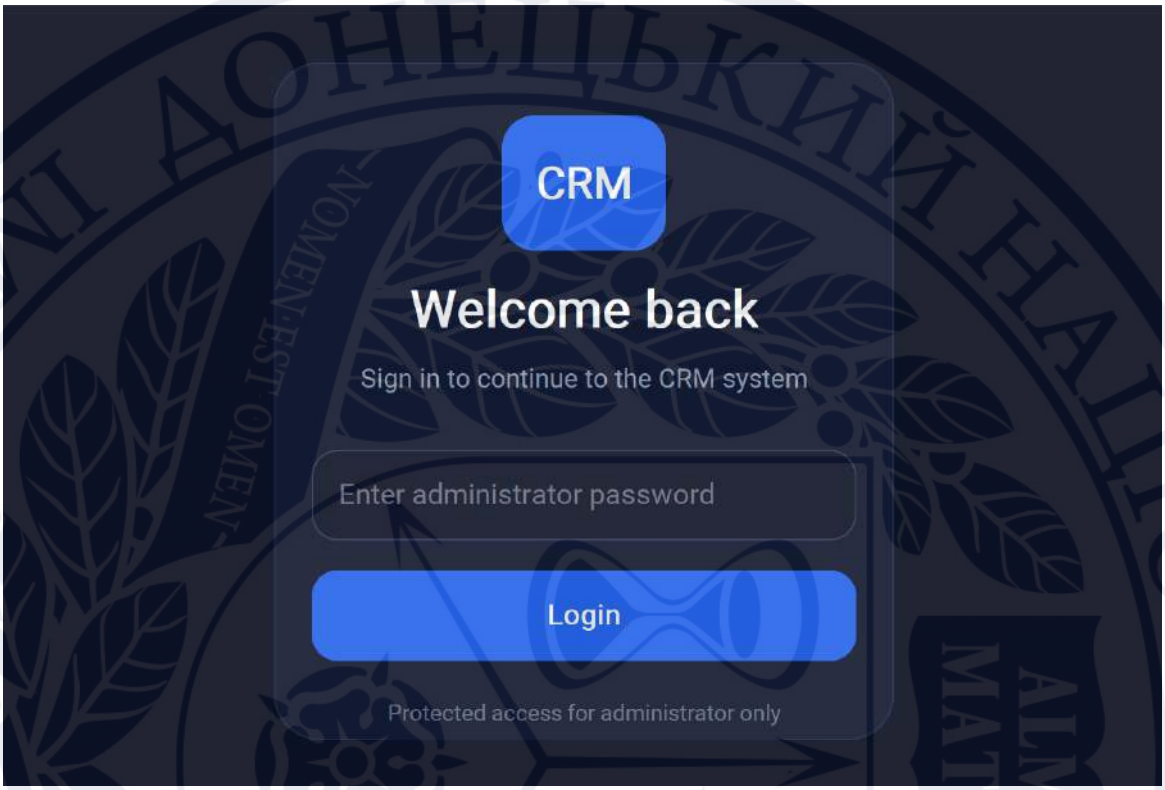

Рисунок 3.3 – Вікно авторизації

Побудова елементів інтерфейсу реалізована у методі `build_ui()`. Основні компоненти вікна розміщуються всередині контейнерів CTkFrame, що дозволяє групувати елементи інтерфейсу в окремі функціональні блоки. У центральній частині вікна розташована картка авторизації, яка містить логотип системи, текстові підказки, поле введення пароля та кнопку входу до системи.

Для введення пароля використовується компонент CTkEntry із прихованим відображенням символів за допомогою параметра `show=«*»`. Поле введення підтримує обробку клавіші Enter через механізм `bind()`, що дозволяє виконувати авторизацію без натискання кнопки входу. Перевірка правильності введених даних виконується у методі `check_password()`. Якщо введений пароль відповідає значенню `ADMIN_PASS`, відбувається закриття вікна авторизації та запуск головного вікна системи. У разі помилки система

відображає повідомлення через `messagebox.showerror()`, очищує поле введення та повертає фокус на компонент введення пароля.

Головне вікно CRM-системи реалізовано у класі `MainWindow`. Для побудови структури інтерфейсу використовується механізм `grid`, який дозволяє гнучко керувати розташуванням елементів та забезпечує адаптацію інтерфейсу до зміни розмірів вікна. Головне вікно складається з бічної панелі навігації та центральної області відображення вмісту, на рисунку 3.4 це зображено цифрами 1 та 2 відповідно.

Рисунок 3.4 – Схематичне зображення структури головного вікна

Бічна панель реалізована за допомогою компонента `CTkFrame` та містить кнопки переходу між основними модулями системи. Для створення кнопок навігації використовується окремий метод `create_sidebar_button()`, який дозволяє централізовано налаштовувати параметри відображення елементів керування. Перехід між модулями виконується методом `handle_nav()`, який оновлює заголовок сторінки та викликає відповідний метод завантаження інтерфейсу.

Центральна область інтерфейсу складається із заголовка сторінки та робочої області для відображення функціональних модулів. Перед відкриттям нового модуля виконується очищення поточного вмісту за допомогою методу `clear_body()`, що дозволяє уникнути накладання елементів інтерфейсу. Завантаження модулів здійснюється через виклик класів `ClientsView`, `OrdersView` та `ReportsView`.

Модуль управління клієнтами (рис. 3.5) реалізовано у класі `ClientsView`. Під час ініціалізації модуля створюється контейнер для розміщення елементів інтерфейсу, налаштовуються параметри таблиці та виконується завантаження списку клієнтів із бази даних. Інтерфейс даного модуля складається з трьох основних блоків: форми введення даних, блоку пошуку та таблиці відображення клієнтів.

Рисунок 3.5 – Модуль управління клієнтами

Для відображення списку клієнтів використовується компонент `ttk.Treeview`. Налаштування зовнішнього вигляду таблиці виконується у методі `setup_treeview_style()`, де визначаються кольори фону, параметри

шрифтів, висота рядків та оформлення вибраного елемента. Для покращення візуального сприйняття використовується окремий стиль `Custom.Treeview`.

Форма введення містить поля для введення імені клієнта, номера телефону та електронної адреси. Для виконання операцій над даними реалізовано кнопки додавання, оновлення та очищення форми. Після натискання кнопки Add Client викликається метод `handle_add_client()`, який виконує перевірку коректності введених даних, викликає функцію `add_client()` модуля бізнес-логіки та оновлює таблицю клієнтів.

Оновлення інформації про клієнта виконується методом `handle_update_client()`. Перед внесенням змін система перевіряє, чи вибраний відповідний запис у таблиці. Після успішного оновлення інформації виконується автоматичне очищення форми та повторне завантаження списку клієнтів.

Пошук клієнтів реалізовано через метод `handle_search_client()`, який отримує текст пошуку та виконує фільтрацію записів за допомогою функції `search_clients()`. Результати пошуку автоматично відображаються у таблиці. Для повернення повного списку записів використовується кнопка Show All.

Видалення клієнта здійснюється методом `handle_delete_client()`. Перед виконанням операції система відображає діалогове вікно підтвердження через `messagebox.askyesno()`, що дозволяє уникнути випадкового видалення даних. Після підтвердження операції виконується видалення запису та оновлення таблиці.

Важливою особливістю реалізації є механізм автоматичного заповнення форми редагування. Після вибору запису у таблиці генерується подія `<<TreeviewSelect>>`, яка викликає метод `on_client_select()`. Даний метод автоматично переносить інформацію вибраного клієнта у відповідні поля введення, що спрощує процес редагування інформації.

Модуль управління замовленнями (рис. 3.6) реалізовано у класі `OrdersView`. Інтерфейс побудований за двоколонковою структурою.

Для роботи із замовленнями використовується таблиця `ttk.Treeview` із налаштованими стилями оформлення. У таблиці відображаються ідентифікатор замовлення, ім’я клієнта, опис, дата створення, кінцевий строк виконання, статус та вартість замовлення. Завантаження даних виконується методом `load_orders()`, а оновлення відображення – методом `refresh_tree()`.

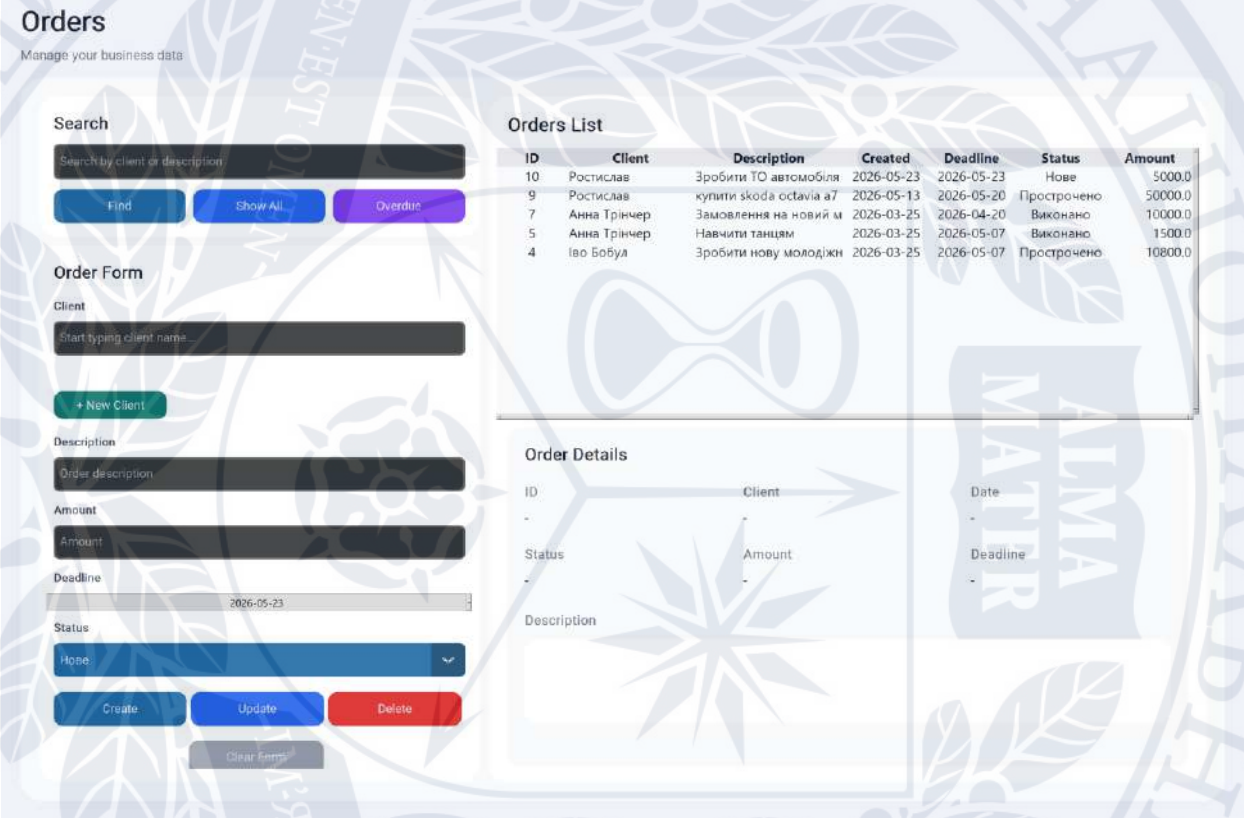

Рисунок 3.6 – Модуль управління замовленнями

Особливістю даного модуля є реалізація механізму автодоповнення для вибору клієнта. Після введення тексту у поле `client_entry` викликається метод `on_client_typing()`, який аналізує введені символи та формує список відповідних клієнтів. Варіанти відображаються у компоненті `Listbox`. Після вибору клієнта система автоматично встановлює ідентифікатор клієнта та заповнює поле введення відповідним значенням.

Для введення строку виконання замовлення використовується компонент `DateEntry` із бібліотеки tkcalendar [37]. Використання календаря спрощує вибір дати та зменшує ймовірність помилок введення. Статус замовлення обирається через компонент `CTkComboBox`, який містить перелік доступних станів замовлення.

Створення нового замовлення виконується методом `handle_create_order()`. Перед збереженням система перевіряє правильність введення числових значень, наявність вибраного клієнта та заповнення обов'язкових полів. У разі успішного створення запису виконується оновлення таблиці замовлень та очищення форми введення.

Редагування замовлень реалізовано методом `handle_update_order()`, який дозволяє змінювати інформацію про замовлення, включаючи опис, статус, вартість та кінцевий строк виконання. Видалення записів виконується методом `handle_delete_order()` із попереднім підтвердженням операції.

Для пошуку замовлень використовується метод `handle_search_orders()`, який виконує пошук за описом замовлення або ім'ям клієнта. Окремо реалізовано метод `handle_show_overdue()`, який дозволяє швидко відобразити список прострочених замовлень.

У нижній частині модуля розташовано блок детальної інформації про вибране замовлення. Оновлення даних у даному блоці виконується методом `update_order_details()`, який автоматично змінює текстові елементи та кольорове оформлення статусу відповідно до поточного стану замовлення.

Модуль формування звітності (рис. 3.7) реалізовано у класі `ReportsView`. Інтерфейс складається з блоку фільтрів, статистичних карток, області побудови графіків та області відображення текстового звіту. Для вибору періоду формування звітності використовуються компоненти `DateEntry`, які дозволяють задавати початкову та кінцеву дату аналізу.

Після натискання кнопки Generate викликається метод `handle_generate_report()`, який отримує статистичні дані за вибраний

період та оновлює елементи інтерфейсу. Якщо за заданий період відсутні записи, система відображає повідомлення про відсутність даних та очищує графічні елементи інтерфейсу.

Для відображення статистичної інформації реалізовано окремі інформаційні картки, які містять кількість замовлень, загальний дохід та статистику за статусами замовлень. Значення карток оновлюються автоматично після формування звіту.

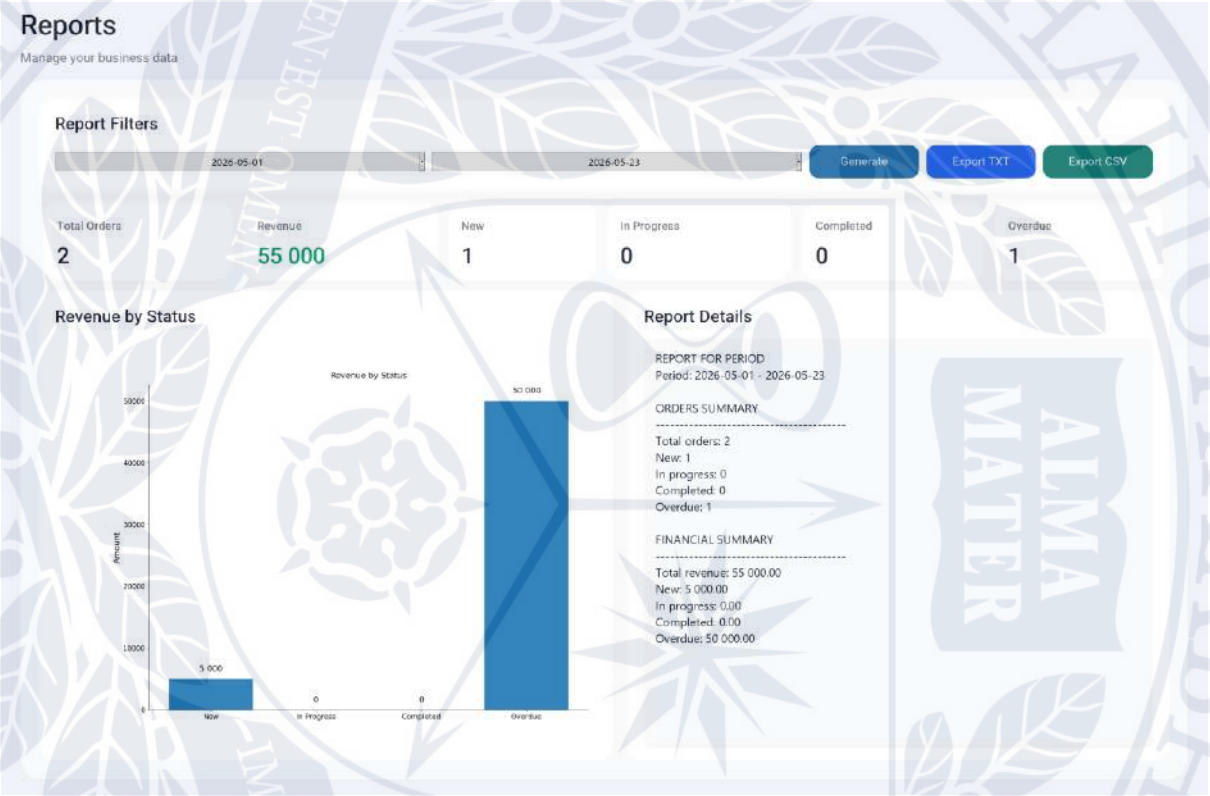

Рисунок 3.7 – Модуль формування звітності

Побудова графічного представлення фінансових показників виконується за допомогою бібліотеки matplotlib [38]. Метод `draw_chart()` формує стовпчикову діаграму, яка відображає розподіл доходу відповідно до статусів замовлень. Для інтеграції графіка у графічний інтерфейс використовується компонент `FigureCanvasTkAgg`.

Текстове представлення звіту відображається у компоненті `CTkTextbox`. Окрім перегляду інформації у програмному інтерфейсі, користувач має можливість експортувати звіти у формати TXT та CSV. Для цього реалізовано

методи `handle_export_txt()` та `handle_export_csv()`, які викликають відповідні функції модуля report_service.py та повідомляють користувача про результат виконання операції.

Реалізований графічний інтерфейс забезпечує зручну взаємодію користувача з функціональними можливостями CRM-системи, підтримує виконання операцій над клієнтами та замовленнями, формування звітності та відображення статистичних даних. Модульна організація інтерфейсу полегшує супровід програмного забезпечення та забезпечує можливість подальшої модифікації й розвитку системи.

3.5 Тестування програмного забезпечення

Після завершення реалізації CRM-системи з метою перевірки коректності роботи основних функціональних модулів, стабільності роботи системи та правильності обробки даних проведено тестування програмного забезпечення. У процесі тестування перевірялась працездатність механізмів авторизації, управління клієнтами та замовленнями, формування звітності, контролю строків виконання, експорту даних та журналювання дій користувача.

Основною метою тестування є підтвердження відповідності реалізованого функціоналу поставленим вимогам та перевірка правильності взаємодії між окремими компонентами системи [39, с. 5; 40]. Тестування виконувалось шляхом проведення функціональних перевірок основних сценаріїв роботи користувача із програмним забезпеченням.

Під час перевірки механізму авторизації виконувалось тестування введення правильного та неправильного пароля. У разі введення коректного значення система успішно відкривала головне вікно CRM-системи та надавала доступ до функціональних модулів. Після введення неправильного пароля система відображає повідомлення про помилку (рис. 3.8) та блокує доступ до

функціоналу програми. Також перевірено очищення поля введення після помилки та повторне встановлення фокусу на поле введення пароля.

Для перевірки модуля управління клієнтами виконувалось тестування операцій додавання, редагування, пошуку та видалення записів. Під час додавання нового клієнта перевірялась коректність збереження даних у базі та оновлення таблиці відображення клієнтів. Також виконувалась перевірка валідації введених даних, зокрема контролю обов'язкового введення імені клієнта. У разі спроби додавання запису без заповнення обов'язкових полів система відображала відповідне повідомлення про помилку (рис. 3.9).

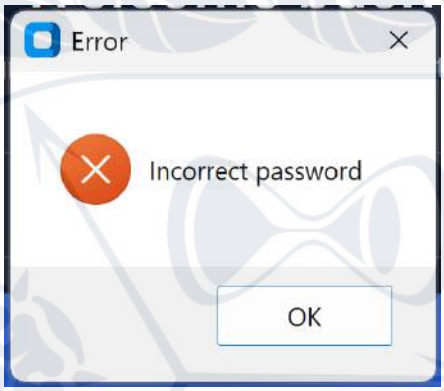

Рисунок 3.8 – Повідомлення про помилку "Неправильний пароль"

Редагування клієнтів тестувалось шляхом вибору запису у таблиці та зміни його параметрів. Після збереження змін система коректно оновлювала інформацію у таблиці та базі даних. Пошук клієнтів перевірявся за різними параметрами, зокрема ім'ям та номером телефону. Результати тестування підтвердили правильність роботи механізму фільтрації записів.

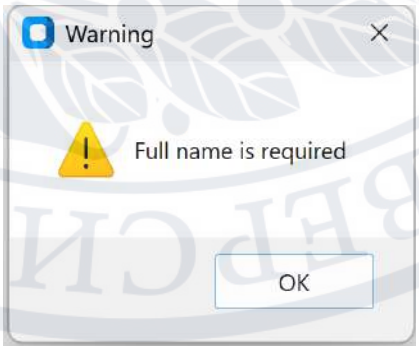

Рисунок 3.9 – Повідомлення про помилку «Запис без заповнення обов'язкових полів»

Під час тестування функції видалення клієнтів перевірялась робота механізму підтвердження операції. Перед видаленням запису система відображала діалогове вікно підтвердження, що дозволяло уникнути випадкового видалення інформації (рис. 3.10). Після підтвердження операції запис успішно видалявся із бази даних та зникав із таблиці відображення.

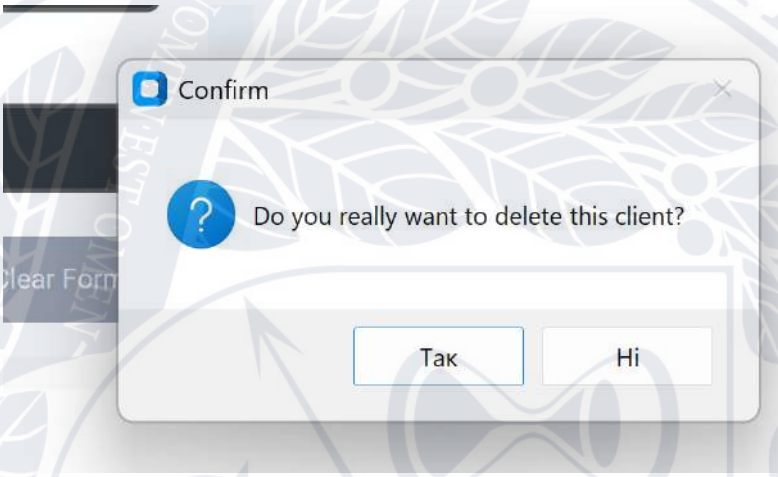

Рисунок 3.10 – Запит підтвердження на видалення

Тестування модуля управління замовленнями включало перевірку створення, редагування, пошуку та видалення замовлень. Під час створення нового замовлення перевірялась правильність збереження інформації про клієнта, опис послуги, вартість, дату створення, кінцевий строк виконання та статус замовлення. Також тестувалась перевірка введення числових значень для поля вартості замовлення (рис. 3.11).

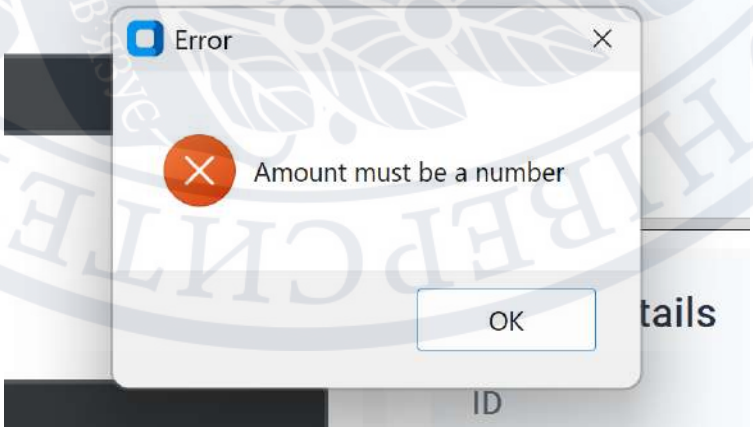

Рисунок 3.11 – Повідомлення про помилку «Числові дані для поля вартості»

Особливу увагу приділено перевірці механізму автодоповнення під час вибору клієнта. У процесі тестування система коректно відображала список клієнтів відповідно до введених символів та автоматично підставляла вибране значення у поле введення (рис. 3.12).

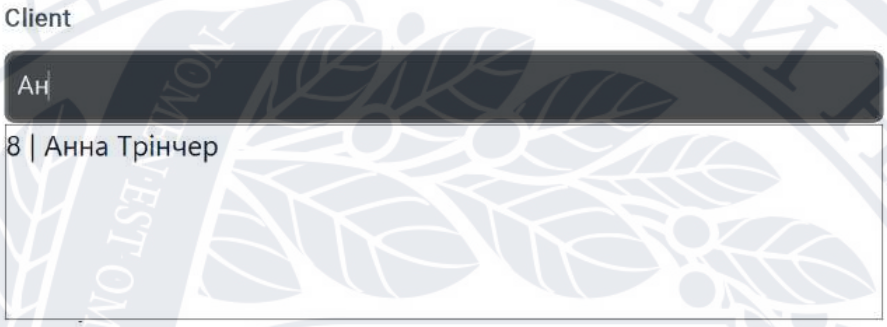

Рисунок 3.12 – Функція автодоповнення

Редагування замовлень перевірялось шляхом зміни статусів, опису та строків виконання. Після внесення змін система успішно оновлювала інформацію у таблиці та відображала актуальні дані у блоці детальної інформації про замовлення.

Окремо виконувалась перевірка підсистеми контролю строків виконання замовлень. У процесі тестування створювались замовлення із простроченим кінцевим строком виконання. Після запуску перевірки система автоматично змінювала статус відповідних записів на «Прострочено». Результати тестування підтвердили правильність роботи механізму автоматичного контролю дедлайнів.

Тестування модуля формування звітності включало перевірку генерації статистичних даних, побудови графіків та експорту результатів у файли. Під час формування звіту система коректно виконувала обчислення статистичних показників за вибраний період та відображала результати у текстовому та графічному вигляді.

Окремо перевірялась робота механізму експорту звітів у формати TXT та CSV. Після виконання експорту система успішно створювала відповідні файли у каталозі reports. Також тестувалась обробка ситуації відсутності даних за

вибраний період. У такому випадку система відображала повідомлення про відсутність інформації та очищувала графічні елементи інтерфейсу (рис. 3.13).

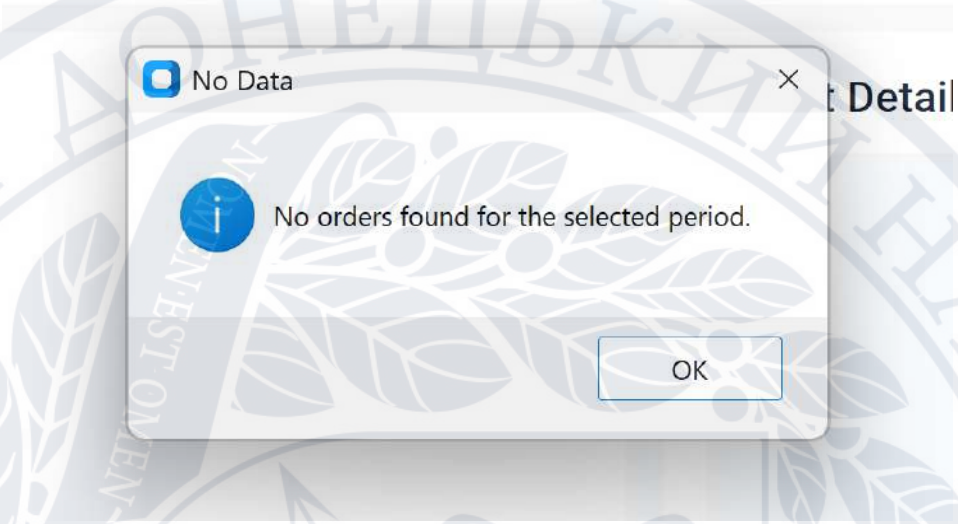

Рисунок 3.13 – Повідомлення про відсутність даних за обраний період

Для перевірки механізму журналювання виконувалось тестування створення, редагування та видалення записів у системі. Після виконання кожної операції система автоматично додавала відповідний запис до журналу дій користувача та файлу log.txt (рис. 3.14). Це дозволило підтвердити коректність роботи механізму фіксації подій у системі.

```
2026-05-07 18:40:15 - Оновлено клієнта ID=6:Іво Бобул
2026-05-07 18:40:27 - Оновлено клієнта ID=12:Катя Акутіна
2026-05-07 18:45:59 - Оновлено замовлення ID=7
2026-05-07 18:46:13 - Оновлено замовлення ID=5
2026-05-07 18:47:23 - Оновлено замовлення ID=4
2026-05-13 16:36:00 - Додано клієнта: Ростислав
2026-05-13 16:38:08 - Створено замовлення для клієнта ID=16
2026-05-13 16:38:18 - Оновлено замовлення ID=9
2026-05-21 13:22:06 - Оновлено замовлення ID=9
```

Рисунок 3.14 – Записи в журналі дій

Проведене тестування засвідчило стабільну роботу CRM-системи та коректність реалізації її основних функціональних можливостей. У процесі перевірки встановлено, що програмне забезпечення коректно обробляє введені

дані, забезпечує взаємодію між модулями системи та правильно виконує операції з базою даних. Реалізовані механізми авторизації, управління клієнтами та замовленнями, контролю строків виконання, формування звітності та журналювання функціонують відповідно до поставлених вимог.

У межах третього розділу реалізовано структуру програмного проєкту, яка забезпечує розмежування компонентів графічного інтерфейсу, бізнес-логіки та роботи з базою даних. Реалізований механізм взаємодії між модулями системи забезпечує централізовану обробку даних та спрощує супровід програмного забезпечення. Використання модульного підходу дозволило організувати структуру проєкту таким чином, щоб окремі компоненти системи могли функціонувати незалежно один від одного.

У процесі реалізації програмного забезпечення створено механізми роботи з базою даних SQLite, реалізовано основні функції управління клієнтами та замовленнями, а також забезпечено автоматичний контроль строків виконання замовлень. Окрему увагу приділено реалізації графічного інтерфейсу користувача, який забезпечує зручну взаємодію із системою та підтримує виконання основних операцій у межах єдиного програмного середовища.

Реалізований модуль формування звітності дозволяє виконувати аналіз діяльності підприємства, відображати статистичну інформацію та експортувати результати роботи у текстові формати. Додатково реалізовано механізм журналювання дій користувача, який забезпечує фіксацію основних операцій у системі та підвищує надійність роботи програмного забезпечення.

Таким чином, у межах третього розділу реалізовано програмну частину CRM-системи, забезпечено взаємодію між основними компонентами програмного забезпечення та підтверджено працездатність системи шляхом проведення функціонального тестування. Отримані результати підтверджують відповідність розробленого програмного забезпечення поставленим функціональним вимогам та можливість його використання для автоматизації основних бізнес-процесів підприємства.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи досліджено особливості автоматизації бізнес-процесів підприємства та проаналізовано існуючі CRM-системи. Встановлено, що більшість наявних рішень характеризуються надлишковим функціоналом, складністю використання або потребують додаткових фінансових витрат, що ускладнює їх застосування у малому бізнесі. На основі проведеного аналізу обґрунтовано доцільність розробки власної десктопної CRM-системи для управління клієнтами та замовленнями.

У процесі виконання роботи сформовано функціональні та нефункціональні вимоги до CRM-системи, спроєктовано структуру бази даних та змодельовано основні бізнес-процеси. Це дозволило визначити логіку взаємодії користувача із системою та реалізувати механізми управління клієнтами і замовленнями, контролю строків виконання та формування звітності.

Практичним результатом роботи стала реалізація CRM-системи засобами Python із використанням SQLite та CustomTkinter. Розроблене програмне забезпечення забезпечує управління клієнтами та замовленнями, автоматичний контроль строків виконання, формування звітності, експорт даних і журналювання дій користувача. Модульна структура проєкту спрощує супровід та подальший розвиток системи.

Розроблено зручний графічний інтерфейс користувача, який забезпечує доступ до основних функціональних можливостей системи.

Проведене функціональне тестування підтвердило працездатність програмного забезпечення та відповідність реалізованих функцій поставленим вимогам.

Таким чином, у результаті виконання кваліфікаційної роботи досягнуто поставленої мети – розроблено CRM-систему для автоматизації основних бізнес-процесів підприємства. Отримані результати можуть бути використані для підвищення ефективності роботи малого підприємства, спрощення управління

клієнтами та замовленнями, а також автоматизації процесів обробки й аналізу інформації.

Практичне значення отриманих результатів полягає у можливості впровадження розробленої CRM-системи у діяльність малих підприємств для зберігання інформації про клієнтів і замовлення, контролю строків виконання та формування звітності.

Перспективами подальшого розвитку розробленої CRM-системи є розширення функціональних можливостей програмного забезпечення, реалізація багатокористувацького режиму роботи, впровадження системи ролей та розмежування прав доступу, а також інтеграція із зовнішніми сервісами обміну даними. Додатково можливим є розширення аналітичного модуля шляхом реалізації більш складних механізмів статистичного аналізу та візуалізації інформації.

СПИСОК ВИКОРИСТАНИХ ПОСИЛАНЬ

1. Янчук Т., Боєнко О. Впровадження CRM-систем як засіб підвищення ефективності маркетингової діяльності. Економіка та суспільство. 2023. № 48. URL: https://economyandsociety.in.ua/index.php/journal/article/view/2269/2192 (дата звернення: 18.05.2026)

2. EDILO Перший в Україні онлайн-сервіс оплати частинами для бізнесу. URL: https://edilo.com.ua/blog/biznes-proczesy-ponyattya-vydy-ta-pryklady/ (дата звернення: 18.05.2026)

3. Дія Бізнес. Допомога підприємцям у створенні та розвитку бізнесу. URL: https://business.diia.gov.ua/entrepreneur-handbook/item/scho-take-crm-sistema-yak-obrati-y-pracyuvati-z-crm-scho-vazhlivo-zamiryuvati (дата звернення: 18.05.2026)

4. Oracle. Американська корпорація, один із найбільших у світі виробників програмного забезпечення для бізнесу та хмарних технологій. URL: https://www.oracle.com/ua/erp/what-is-erp/ (дата звернення: 18.05.2026)

5. IT Enterprise. Українська it-компанія, яка спеціалізується на розробці та впровадженні рішень цифрової трансформації бізнес-процесів. URL: https://www.it.ua/knowledge-base/technology-innovation/supply-chain-management-scm (дата звернення: 18.05.2026)

6. Hammer M., Champy J. Reengineering the Corporation: a Manifesto for Business Revolution. New York : Harper Business, 2006. 272 p.

7. Nethanani R., Matlombe L., Vuko S., Thango B. Customer Relationship Management (CRM) Systems and their Impact on SMEs Performance: A Systematic Review. Johannesburg, 2024. 52 p.

8. Гужва В. М. Інформаційні системи і технології на підприємствах: Навч. посібник. Київ, 2001. 400 с.

9. Гадецька З. М. CRM-системи як засіб автоматизації бізнес-процесів торговельного бізнесу.
URL: https://dees.iei.od.ua/index.php/journal/article/view/292/278 (дата звернення: 18.05.2026)

10. Salesforce. Хмарна платформа для автоматизації бізнесу та управління взаємовідносинами з клієнтами. URL: https://www.salesforce.com/ (дата звернення: 18.05.2026)

11. Zoho. Хмарна екосистема для управління бізнесом. URL: https://www.zoho.com/crm/ (дата звернення: 18.05.2026)

12. Hubspot. Платформа, яка об'єднує інструменти для маркетингу, продажів, обслуговування клієнтів та управління контентом. URL: https://www.hubspot.com/products/crms (дата звернення: 18.05.2026)

13. What is Cloud CRM?. URL: https://www.salesforce.com/crm/what-is-cloud-crm/ (дата звернення: 09.05.2026).

14. Web-based CRM Software. URL: https://www.zoho.com/crm/what-is-crm/ (дата звернення: 18.05.2026)

15. Офіційна сторінка мови програмування Python. URL: https://www.python.org/about/apps/ (дата звернення: 18.05.2026)

16. Офіційна сторінка SQLite. URL: https://sqlite.org/about.html (дата звернення: 18.05.2026)

17. Що таке SQlite? URL: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-sqlite/ (дата звернення: 18.05.2026)

18. CustomTkinter. URL: https://customtkinter.tomschimansky.com/ (дата звернення: 18.05.2026)

19. Linkedin. CustomTkinter - A Complete Tutorial. URL: https://www.linkedin.com/pulse/customtkinter-complete-tutorial-nuno-bispo-wuehe (дата звернення: 18.05.2026)

20. Sommerville I. Software Engineering. 10th ed. Boston, 2015. 816 p.

21. Bass L., Clements P., Kazman R. Software Architecture in Practice. 4th ed. Boston, 2021. 497 p.

22. Модульність у програмуванні: навіщо вона потрібна і як її досягти. URL: https://itproger.com/ua/news/modulnost-v-programmirovanii-zachem-ona-nuzhna-i-kak-eio-dobitsya (дата звернення: 18.05.2026)

23. Mark Richards, Neal Ford. Fundamentals of Software Architecture. 2021. 419 p.

24. Лясковець, Ю. Багаторівневі архітектури для досягнення масштабованості, безпеки та стійкості. Київ, 2024. 105 с.

25. Про архітектуру додатків. URL: https://foxminded.ua/arkhitektura-zastosunku/ (дата звернення: 18.05.2026)

26. Доценко С. І. Організація та системи керування базами даних: Навч. посібник. Харків, 2023. 117 с.

27. Chen P. P.-S. The Entity-Relationship Model: Toward a Unified View of Data. ACM Transactions on Database Systems. 1976. Vol. 1, No. 1. 36 p.

28. Що таке ER Modeling?. URL: https://www.guru99.com/uk/er-modeling.html (дата звернення: 18.05.2026)

29. Nielsen J. Usability Engineering. Boston, 1993. 362 p.

30. 10 принципів створення хороших інтерфейсів. URL: https://journal.gen.tech/post/desyat-principiv-stvorennya-horoshih-interfeisiv (дата звернення: 18.05.2026)

31. Shneiderman B., Plaisant C. Designing the User Interface: Strategies for Effective Human-Computer Interaction. 6th ed. Boston, 2016. 624 p.

32. FoxmindEd. Школа IT-професій. URL: https://foxminded.ua/blok-skhema/ (дата звернення: 18.05.2026)

33. Алгоритми та структури даних – від «десь чув» до «ефективно застосовую». DOU, 2022. URL: https://dou.ua/forums/topic/40645/ (дата звернення: 18.05.2026)

34. Важливість алгоритмів у сучасному програмуванні. URL: https://optima.college/news/vazhlivist-algoritmiv-u-suchasnomu-programuvanni-yak-voni-formuyut-maybutnye (дата звернення: 18.05.2026).

35. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Boston, 2017. 432 p.

36. SQLite SQL Language Expressions. URL: https://www.sqlite.org/lang_expr.html (дата звернення: 18.05.2026).

37. tkcalendar Documentation. URL: https://tkcalendar.readthedocs.io/ (дата звернення: 18.05.2026)

38. Matplotlib Documentation. URL: https://matplotlib.org/stable/index.html (дата звернення: 18.05.2026).

39. Myers G. J., Sandler C., Badgett T. The Art of Software Testing. 3rd ed. Hoboken, 2011. 248 p.

40. Dorothy Graham, Rex Black, Erik van Veenendaal. Foundations of Software Testing ISTQB Certification: 4th Edition. 2018. 275 p.