Міністерство освіти і науки України

Луцький національний технічний університет

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ СЕРВІСНОГО ЦЕНТРУ НА БАЗІ МОВИ ПРОГРАМУВАННЯ C#

DEVELOPMENT OF AN INFORMATION SYSTEM FOR A SERVICE CENTER BASED ON THE C# PROGRAMMING LANGUAGE

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
Групи ІПЗс-21
Ткачук Святослав Сергійович
_____
(підпис)

Керівник:
к.т.н., доцент
Суринович Олена Миколаївна
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«5» 12 2024 р.
к.т.н., доцент
Гарант освітньої програми:
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *121 Інженерія програмного забезпечення*
Освітня програма: *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«2» 02 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Ткачуку Святославу Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *«Розробка інформаційної системи для сервісного центру на базі мови програмування C#»*

Керівник роботи: *к.т.н., доцент, Суринович Олена Миколаївна*
затверджені наказом закладу вищої освіти від «*30» грудня 2023 р.* № *477/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«4» грудня 2024 р.*
3. Вихідні дані до роботи: *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):
Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.

5. Перелік графічного матеріалу: *15 рисунків, з яких 9 схем*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Суринович О. М.*		
Специфікації вимог до розробленої системи	*Суринович О. М.*		
Розробка об'єкта проєктування	*Суринович О. М.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту		14%	
Академічна доброчесність	*Суринович О. М.*		

7. Дата видачі завдання *«2» лютого 2024 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*07.02.2024 р.*	вик.
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*27.02.2024 р.*	вик.
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*12.03.2024 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2024 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2024 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2024 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*04.12.2024 р.*	вик.

Здобувач вищої освіти _____ (підпис) *Ткачук С. С.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Суринович О. М.* (прізвище, ініціали)

АНОТАЦІЯ

Ткачук С. С. Розробка інформаційної системи для сервісного центру на базі мови програмування C#. Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2024.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків та списку використаних джерел.

У першому розділі здійснено аналіз предметної області і постановка завдань на кваліфікаційну роботу. В другому розділі визначено специфікацію вимог до розроблюваної системи. У третьому розділі описано етапи розробки інформаційної системи для сервісного центру. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробки демонструють можливості розробленої системи.

Ключові слова: інформаційна система, C#, база даних, MySQL, сервісний центр, бізнес-процеси.

ABSTRACT

Tkachuk S. Development of an Information System for a Service Center Based on the C# Programming Language. Manuscript.

Bachelor’s qualifying thesis of the educational program «Software Engineering». Lutsk National Technical University. Lutsk, 2024.

The bachelor’s qualifying thesis consists of an introduction, three sections, conclusions and references.

In the first section, the analysis of the subject area and the setting of tasks for the qualification work were carried out. The second section defines the specification of requirements for the developed system. The third section describes the stages of developing an information system for a service center. The conclusions summarize the information presented in the previous parts. The development results demonstrate the capabilities of the developed system.

Keywords: information system, C#, database, MySQL, service center, business processes.

ЗМІСТ

ВСТУП

Актуальність роботи. Розвиток інформаційних технологій та інтенсивне використання автоматизованих систем в управлінні бізнес-процесами створюють необхідність у розробці інноваційних рішень для забезпечення ефективної роботи підприємств. У сучасних умовах сервісні центри стикаються зі значними обсягами даних, що потребують систематизації, аналізу та зберігання. Створення інформаційної системи на основі мови програмування C# дозволяє оптимізувати обробку інформації, забезпечити її високу продуктивність та зручність використання для працівників сервісних центрів.

Об’єкт кваліфікаційної роботи бакалавра – автоматизована система управління, яка використовуватиметься у сервісних центрах для організації бізнес-процесів.

Предмет кваліфікаційної роботи бакалавра – методи, алгоритми та інструменти програмування для створення інформаційної системи, орієнтованої на автоматизацію операцій сервісного центру.

Метою роботи є розробка інформаційної системи для сервісного центру, яка дозволить автоматизувати ключові бізнес-процеси організації.

Для реалізації мети потрібно виконати наступні завдання:

– провести аналіз існуючих систем, якими користуються підприємства;

– обґрунтувати вибір мови програмування C# як інструменту для створення інформаційної системи;

– розробити базу даних;

– розробити алгоритми та інтерфейс;

– здійснити тестування системи на відповідність заявленим вимогам.

Новизна роботи. Розроблена інформаційна система базується на сучасних підходах до розробки програмного забезпечення та інтегрує інструменти, які забезпечують високий рівень автоматизації процесів для сервісного центру. В роботі запропоновані оптимізовані алгоритми обробки даних, що покращують швидкість виконання операцій.

Практична цінність кваліфікаційної роботи бакалавра полягає у тому, що інформаційна система для сервісного центру, розроблена в межах кваліфікаційної роботи, може бути використана як основа для впровадження у реальних підприємствах. Це дозволить зменшити час обробки клієнтських запитів, покращити контроль над ресурсами та підвищити продуктивність персоналу. Система має можливість адаптації під специфічні вимоги підприємства, що робить її універсальним інструментом для широкого спектра сервісних центрів.

Таким чином, створення інформаційної системи для сервісного центру на базі мови програмування C# є актуальним завданням, що спрямоване на підвищення ефективності роботи підприємств цієї сфери.

РОЗДІЛ 1
АНАЛІЗ ІНФОРМАЦІЙНИХ СИСТЕМ У СЕРВІСНИХ ЦЕНТРАХ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Ефективність роботи будь-якого підприємства значною мірою визначається правильним управлінням даними. Підвищення цієї ефективності досягається завдяки впровадженню автоматизованих систем, заснованих на базах даних.

Бази даних докорінно трансформують підходи до роботи організацій у різних сферах, автоматизуючи рутинні процеси, зокрема пошук інформації, який раніше вимагав перегляду численних файлів, паперових документів або довідників [1].

Використання баз даних стає необхідним завдяки їхній здатності виконувати різні операції з інформацією: зберігання, обробку та надання. Це дозволяє максимально задовольнити потреби користувачів. Щоб оптимізувати витрати часу і ресурсів на отримання потрібних даних, а також спростити доступ до них, застосовуються клієнт-серверні технології, які забезпечують централізоване зберігання й обробку інформації.

Програмне забезпечення для роботи з базами даних сприяє скороченню обсягу паперової документації й пришвидшує обробку даних, що вимагає оперативного виконання. Усе це сприяє автоматизації робочих процесів, що, своєю чергою, підвищує продуктивність праці працівників підприємства.

Сьогодні системи управління базами даних продовжують відігравати ключову роль у різних сферах людської діяльності, зокрема в сфері послуг.

Сфера послуг нині є ключовою складовою економіки, яка спрямована на задоволення індивідуальних потреб населення у різноманітних видах послуг. Як економічна галузь, вона об’єднує організації, які надають платні послуги, орієнтуючись на запити споживачів. До таких послуг належить, зокрема, сервісне обслуговування цифрової техніки.

На сьогодні існує безліч автоматизованих систем для підприємств, зокрема Oracle E-Business Suite, SAP та Microsoft Dynamics. Серед них можна виділити найбільш популярні та часто використовувані:

– Oracle E-Business Suite;

– Microsoft Dynamics.

Коротко опишемо ці системи, означивши їх переваги та недоліки.

Oracle E-Business Suite (Oracle EBS) – це інтегрований набір програм для управління бізнес-процесами, який охоплює такі сфери, як фінанси, виробництво, управління ланцюгами постачання, людські ресурси та багато інших. Ця система є одним із найбільш потужних рішень для підприємств, що прагнуть автоматизувати свої бізнес-процеси та забезпечити ефективне управління даними [2].

Oracle E-Business Suite складається з модулів, кожен з яких відповідає за певну бізнес-функцію. Наприклад:

– фінансовий модуль забезпечує управління рахунками, звітністю, бюджетами та іншими фінансовими операціями;

– модуль управління ланцюгами постачання автоматизує процеси закупівель, управління запасами та логістики;

– модуль управління людськими ресурсами (HCM) дозволяє вести облік персоналу, заробітної плати, робочого часу та навчання.

Система працює на основі централізованої бази даних Oracle, що забезпечує зберігання й обробку даних у реальному часі. Користувачі взаємодіють із системою через інтуїтивно зрозумілий веб-інтерфейс або спеціальні клієнтські додатки. Oracle EBS може бути розгорнута як на фізичних серверах компанії, так і в хмарному середовищі, що дає додаткову гнучкість у використанні.

Коротко опишемо переваги системи:

– широкий функціонал, де Oracle EBS надає інструменти для автоматизації практично всіх бізнес-процесів, що дозволяє замінити безліч окремих програм одним інтегрованим рішенням;

– масштабованість – система підходить для організацій будь-якого розміру, від середніх компаній до великих корпорацій. Її можна налаштувати відповідно до специфічних потреб бізнесу;

– інтеграція модулів, де усі модулі Oracle EBS взаємопов’язані, що дозволяє уникнути дублювання даних та зменшити ризик помилок при передачі інформації між відділами;

– безпека – завдяки використанню передових технологій Oracle забезпечує високий рівень захисту даних, включаючи шифрування, контроль доступу та аудит дій користувачів;

– можливості аналітики, де вбудовані інструменти для аналізу даних дозволяють отримувати глибокі аналітичні звіти, які сприяють прийняттю об’єктивних управлінських рішень.

До недоліків системи можемо віднести:

– висока вартість. Впровадження та ліцензування Oracle EBS може бути дорогим, що робить її недоступною для малих підприємств;

– складність налаштування. Через широку функціональність система потребує значних зусиль для налаштування, а також досвідчених спеціалістів для її впровадження й обслуговування;

– тривалий період впровадження. Залежно від масштабу бізнесу та кількості інтегрованих модулів, повноцінне впровадження Oracle EBS може зайняти декілька місяців або навіть років;

– потреба в навчанні персоналу. Для ефективного використання системи співробітникам необхідно пройти спеціалізоване навчання, що може збільшити витрати компанії на впровадження;

– залежність від технічної підтримки. Для забезпечення стабільної роботи системи потрібна постійна технічна підтримка, що може бути витратним.

Аналізуючи вищесказане, можна підсумувати, що Oracle E-Business Suite – це потужна система для управління бізнесом, яка здатна достатньо підвищити можливість роботи підприємства, забезпечивши інтеграцію та

автоматизацію основних процесів. Однак її висока вартість і складність впровадження можуть стати перешкодою для деяких компаній. Незважаючи на це, Oracle EBS залишається одним із провідних рішень на ринку, особливо для великих корпорацій, які прагнуть отримати максимальну ефективність від управління своїми ресурсами.

Також варто розглянути ще одну систему – Microsoft Dynamics.

Microsoft Dynamics – це набір програмних рішень для управління бізнес-процесами, який включає CRM (Customer Relationship Management) та ERP (Enterprise Resource Planning) системи [3]. Продукти Microsoft Dynamics, такі як Dynamics 365, об'єднують функції для управління фінансами, продажами, маркетингом, обслуговуванням клієнтів та відповідними операціями.

Microsoft Dynamics функціонує як інтегрована платформа, що підтримує хмарну архітектуру та модульний підхід. Користувачі можуть вибирати ті модулі, які відповідають їхнім бізнес-потребам, наприклад:

– ERP – забезпечує управління фінансовими потоками, виробництвом, запасами та логістикою;

– CRM – дає доступ до інструментів з автоматизації маркетингових кампаній, управління взаємодією з клієнтами та підтримки продажів;

– HR – модуль для управління людськими ресурсами, включаючи облік персоналу, навчання та оцінку продуктивності.

Дані зберігаються в централізованій базі, а доступ до них здійснюється через веб-інтерфейс, мобільні додатки або інтеграцію з іншими продуктами Microsoft, такими як Office 365 та Power BI. Завдяки вбудованим інструментам штучного інтелекту можна аналізувати дані в режимі реального часу та отримувати прогнози для ухвалення управлінських рішень.

До переваг системи можемо віднести:

– інтеграція з екосистемою Microsoft, де Microsoft Dynamics легко інтегрується з іншими продуктами Microsoft, такими як Excel, Outlook та Teams, що спрощує робочі процеси та обмін інформацією між командами;

– модульна структура. Платформа дозволяє компаніям вибирати тільки ті функціональні модулі, які їм необхідні, що знижує витрати на впровадження;

– хмарна архітектура, де Dynamics 365 дозволяє розгортання в хмарі, що забезпечує доступ до системи з будь-якого місця, підвищує її гнучкість і зменшує витрати на підтримку локальної інфраструктури;

– можливості аналітики. Вбудовані інструменти аналітики та інтеграція з Power BI дозволяють отримувати детальні звіти, аналізувати дані та робити прогнози на основі штучного інтелекту;

– підтримка малого та середнього бізнесу, де Microsoft Dynamics доступна для компаній різного масштабу, включаючи малі та середні підприємства, завдяки різним конфігураціям та ціновим моделям.

До недоліків системи слід віднести:

– висока вартість для великих організацій. Незважаючи на доступність для малого бізнесу, для великих корпорацій повноцінне впровадження та підтримка всіх модулів можуть стати значними витратами;

– складність налаштування. Як і будь-яка потужна ERP/CRM система, Microsoft Dynamics потребує належного налаштування та адаптації під потреби бізнесу, що може зайняти багато часу;

– потреба в навчанні. Співробітникам необхідно освоїти нові інструменти, а для складніших задач потрібні спеціалісти, що збільшує витрати на навчання;

– залежність від інтернету. Хмарна архітектура залежить від стабільного інтернет-з’єднання, що може бути проблемою для компаній у регіонах із поганою інфраструктурою;

– залежність від Microsoft. Система тісно інтегрована з іншими продуктами Microsoft, що може стати обмеженням для компаній, які вже використовують альтернативні програмні рішення.

Здійснюючи аналіз системи, можемо зробити висновок, що Microsoft Dynamics – це гнучка та потужна система для управління бізнесом, яка підходить для компаній різного розміру. Її ключовими перевагами є інтеграція з

екосистемою Microsoft, хмарна архітектура та можливості аналітики. Водночас висока вартість для великих організацій та залежність від технічної підтримки можуть стати викликами при впровадженні. Незважаючи на це, Microsoft Dynamics є одним із провідних рішень на ринку, яке активно використовується в різних галузях для підвищення роботи бізнес-процесів.

Підсумовуючи, можна зазначити, що існуючі на сьогодні рішення є універсальними, але вони мають свої обмеження: вимагають сплати ліцензійних внесків, навчання персоналу, а також можуть бути перевантаженими зайвими функціями або мати суттєві недоліки. У зв’язку з цим було вирішено розробити власний програмний продукт, який повністю відповідатиме потребам сервісного центру з обслуговування цифрової техніки та забезпечуватиме зручний і продуктивний інтерфейс.

Це передбачає розробку бази даних і прикладного програмного забезпечення для управління обліком клієнтів, пристроїв, послуг, постачання, контролю розрахунків за замовлені послуги, нарахування заробітної плати та кадрового контролю. Реалізація такого продукту значно підвищить продуктивність організації.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Головною метою при розробці системи є спрощення та автоматизація бізнес-процесів сервісного центру з обслуговування цифрової техніки. Така система дозволить автоматизувати ключові бізнес-процеси даної організації.

Для реалізації автоматизованої системи необхідно виконати наступні завдання:

– провести аналіз існуючих систем, якими користуються підприємства;

– обґрунтувати вибір мови програмування C# як інструменту для створення інформаційної системи;

– розробити базу даних;

– розробити алгоритми та інтерфейс;

– здійснити тестування системи на відповідність заявленим вимогам.

Висновки до розділу 1

Ефективне управління даними на підприємствах значно залежить від автоматизованих систем, що працюють на основі баз даних. Вони не лише прискорюють рутинні процеси, але й дозволяють централізувати зберігання, обробку та доступ до інформації. Завдяки цьому підприємства оптимізують витрати ресурсів, покращують якість роботи персоналу та підвищують продуктивність. Використання сучасних програмних рішень, таких як клієнт-серверні технології, є ключовим для забезпечення швидкого доступу до даних і їх аналітичної обробки.

Розглянуті системи Oracle E-Business Suite та Microsoft Dynamics демонструють високий потенціал для автоматизації бізнес-процесів. Oracle EBS пропонує інтеграцію модулів, потужну аналітику та масштабованість, що робить її оптимальним вибором для великих корпорацій. Водночас Microsoft Dynamics, завдяки модульній структурі, хмарній архітектурі та інтеграції з іншими продуктами Microsoft, підходить як для малих, так і великих компаній. Однак обидві системи мають суттєві недоліки, такі як висока вартість впровадження, тривалий період налаштування та залежність від навчання персоналу.

Зважаючи на універсальність, але водночас обмеження існуючих рішень, постає необхідність розробки власного програмного продукту. Така система дозволить оптимізувати специфічні бізнес-процеси сервісного центру, включаючи облік клієнтів, пристроїв, послуг і кадрове управління. Використання бази даних, розробленої на сучасних платформах, та прикладного програмного забезпечення, створеного на мові C#, сприятиме ефективній автоматизації роботи організації та підвищить якість обслуговування клієнтів.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

Для розробки автоматизованої системи сервісного центру потрібно обрати найоптимальнішу систему управління базами даних (СУБД), беручи до уваги її функціональність, надійність, вартість і здатність інтегруватися з іншими елементами інформаційної системи. Розгляньмо основні варіанти СУБД.

Firebird – це безкоштовна СУБД з відкритим кодом, яка підтримує мови SQL і PSQL [4]. Коротко опишемо переваги та недоліки відповідної системи. До переваг можемо віднести відкритий вихідний код, невисокі вимоги до апаратного забезпечення, можливість обробки транзакцій і підтримка процедур. До недоліків відносимо обмежену масштабованість для великих проектів, обмежену підтримка комерційних рішень і менше інструментів для інтеграції.

Oracle – одна з найпотужніших СУБД, широко використовується у великих корпораціях [1]. Коротко опишемо переваги та недоліки відповідної системи. До переваг можемо віднести потужну аналітику і підтримку великих обсягів даних, високу безпеку і підтримку транзакцій. До недоліків відносимо високу вартість ліцензування, складність налаштування та підтримки.

Sybase Adaptive Server Enterprise (ASE) – СУБД, розроблена для корпоративного середовища [5]. Коротко опишемо переваги та недоліки відповідної системи. До переваг можемо віднести підтримку великих обсягів транзакцій, хорошу інтеграція з системами SAP. До недоліків відносимо обмежену спільнота розробників, високі вимоги до апаратного забезпечення.

InterBase – компактна СУБД, що підходить для вбудованих рішень. Коротко опишемо переваги та недоліки відповідної системи [6]. До переваг можемо віднести підтримку транзакцій і процедур, високу продуктивність на

невеликих проєктах. До недоліків відносимо обмежену масштабованість, платний комерційний варіант для повноцінного використання.

PostgreSQL – це високопродуктивна система управління базами даних з відкритим кодом, яка відзначається надійністю [7]. Опишемо переваги та недоліки відповідної системи. До переваг можемо віднести високу функціональність, підтримку JSON і складних запитів, безкоштовність і активна спільнота. До недоліків відносимо високу вимогливість до налаштувань і ресурсів сервера, не завжди підходящість для простих проєктів.

Microsoft SQL Server – популярна комерційна СУБД із підтримкою SQL. Коротко опишемо переваги та недоліки відповідної системи [8]. До переваг можемо віднести глибоку інтеграцію з продуктами Microsoft, високу продуктивність і масштабованість. До недоліків відносимо платність та вимогливість до апаратного забезпечення.

Mini SQL (mSQL) – легка СУБД для невеликих проєктів. Коротко опишемо переваги та недоліки відповідної системи [9]. До переваг можемо віднести простоту налаштування та невеликі системні вимоги. До недоліків відносимо обмежений функціонал та відсутність підтримки великих баз даних.

MySQL – популярна СУБД з відкритим кодом, що використовується в різних проєктах [10]. Коротко опишемо переваги та недоліки відповідної системи. До переваг можемо віднести: безкоштовність і відкритий вихідний код, широка підтримка веб-середовищ і модульної архітектури, простота у використанні й доступність документації. До недоліків відносимо наступне: обмежена підтримка транзакцій у старих версіях та потреба у налаштуванні для високого навантаження.

Для розробки автоматизованої системи для сервісного центру обрано MySQL через її універсальність, безкоштовність, активну спільноту та простоту інтеграції. MySQL дозволяє створювати бази даних для середніх за масштабом проєктів із мінімальними витратами на налаштування та підтримку.

Для створення автоматизованої системи розглянемо три мови програмування: Java, C++, C#.

Java – популярна мова з підтримкою об’єктно-орієнтованого програмування. Коротко опишемо переваги та недоліки відповідної мови. До переваг можемо віднести кросплатформеність, велику бібліотеку готових рішень, високу продуктивність для багатопоточних застосунків. До недоліків відносимо відносно високу складність у вивченні для початківців, потребу у більших ресурсах, ніж C++ [11].

C++ – потужна мова для створення високопродуктивного програмного забезпечення [12]. Коротко опишемо переваги та недоліки відповідної мови. До переваг можемо віднести: максимальну продуктивність і контроль над апаратним забезпеченням, підходящість для створення складних застосунків. До недоліків відносимо наступне: високу складність розробки і триваліший цикл створення продукту, відсутність стандартних інструментів для інтеграції з базами даних.

C# – сучасна мова програмування від Microsoft, оптимізована для розробки бізнес-додатків. Коротко опишемо переваги та недоліки відповідної мови. До переваг можемо віднести: висока інтеграція з Windows і продуктами Microsoft, простота вивчення та використання завдяки дружньому синтаксису, потужна підтримка інтеграції з базами даних, включаючи MySQL, через Entity Framework, наявність інструментів для розробки графічного інтерфейсу. До недоліків відносимо наступне: прив’язка до платформи .NET [13].

C# обрано для розробки автоматизованої системи завдяки її функціональності, зручності інтеграції з MySQL та можливостям створення графічного інтерфейсу. Використання цієї мови забезпечить ефективну реалізацію системи для сервісного центру, враховуючи її потреби та доступні ресурси [13].

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Розробка прикладного програмного забезпечення для ІС сервісу із обслуговування цифрової техніки вимагає розробки наступних алгоритмів:

– авторизації користувача для обраного сервера;

– вибору функції обробки даних, яка включає додавання, редагування і пошук;

– формування звітності;

– нарахування заробітної плати.

Виходячи з того, що окремих вимог до алгоритму не пред’являлося, був розроблений алгоритм авторизації користувачів, що складається з наступного: відображення вікна авторизації, вибір адреси серверу, що містить базу даних, введення логіну і пароля, читання даних з бази даних, порівняння даних з даними з бази даних, запуск основної програми з урахуванням поділу прав користувачів.

На етапі відображення вікна авторизації програми – програма чекає поки користувач вибере сервер, введе логін і пароль, після їх введення відбувається читання логіну і пароля з бази даних; ці дані порівнюються і в разі, якщо вони коректні, – відбудеться відкриття основної програми з елементами доступними для користувачів, в іншому випадку – виводиться повідомлення про те, що дані не вірні.

Схема алгоритму авторизації користувача представлена на рисунку 2.1.

Обробки даних у програмі повинна здійснюватися, спираючись на розроблений алгоритм, що містить такі функції як додавання, редагування і пошук. Робота алгоритму полягає в наступному: при запуску програми необхідно пройти авторизацію і в разі успішної авторизації, відображається головне вікно програми, в якому користувач вибирає потрібне йому вікно, наприклад, клієнт для необхідної операції; після цього настає відкриття

вибраного вікна і здійснюється необхідна дія, а саме – додавання, редагування або пошук.

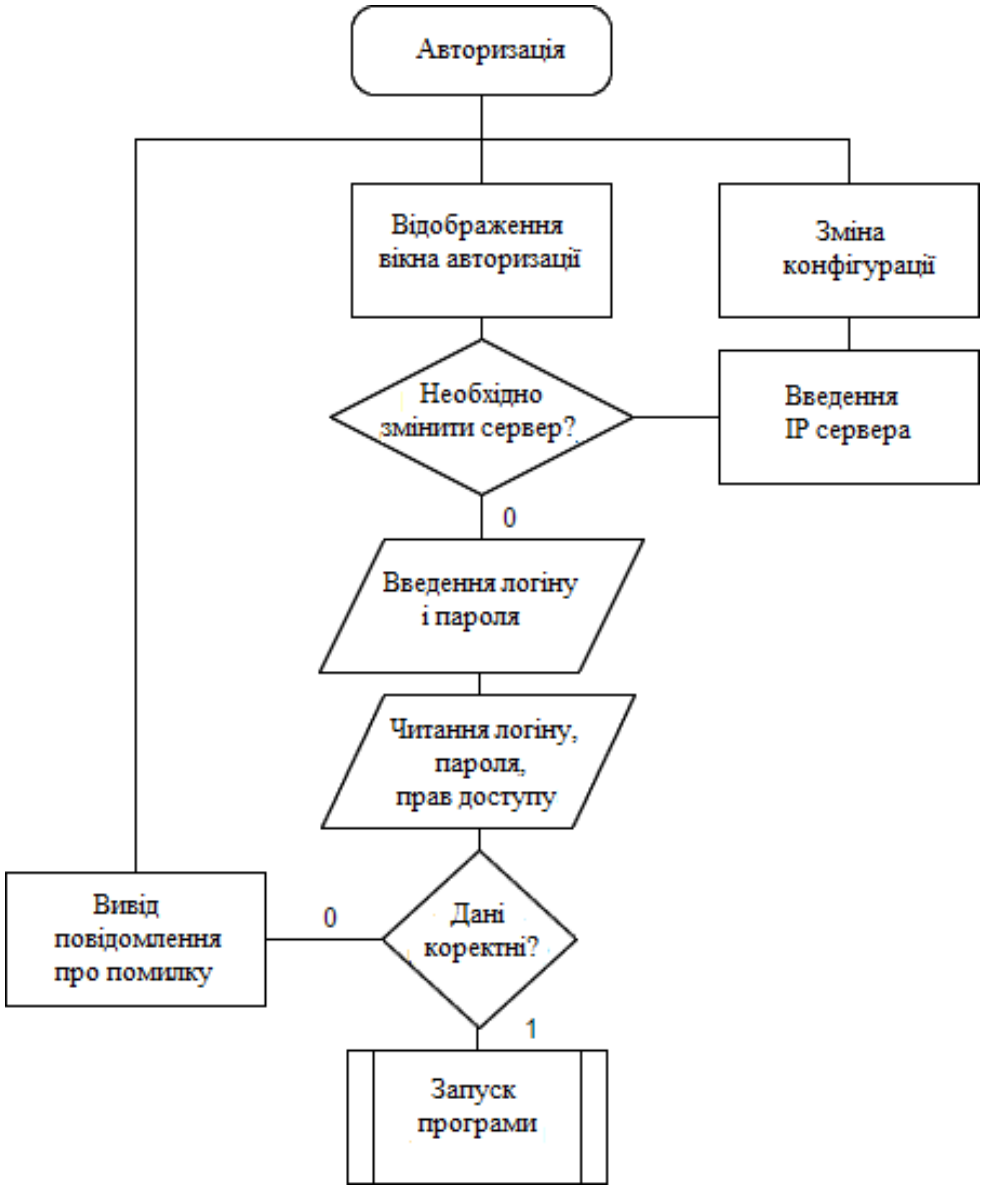

Рисунок 2.1 – Візуалізація алгоритму авторизації користувача

Після того як всі необхідні дії будуть виконані, можна або вийти з системи, або знову вибрати необхідну операцію і продовжити роботу.

Система алгоритму вибору функції обробки даних представлена на рисунку 2.2.

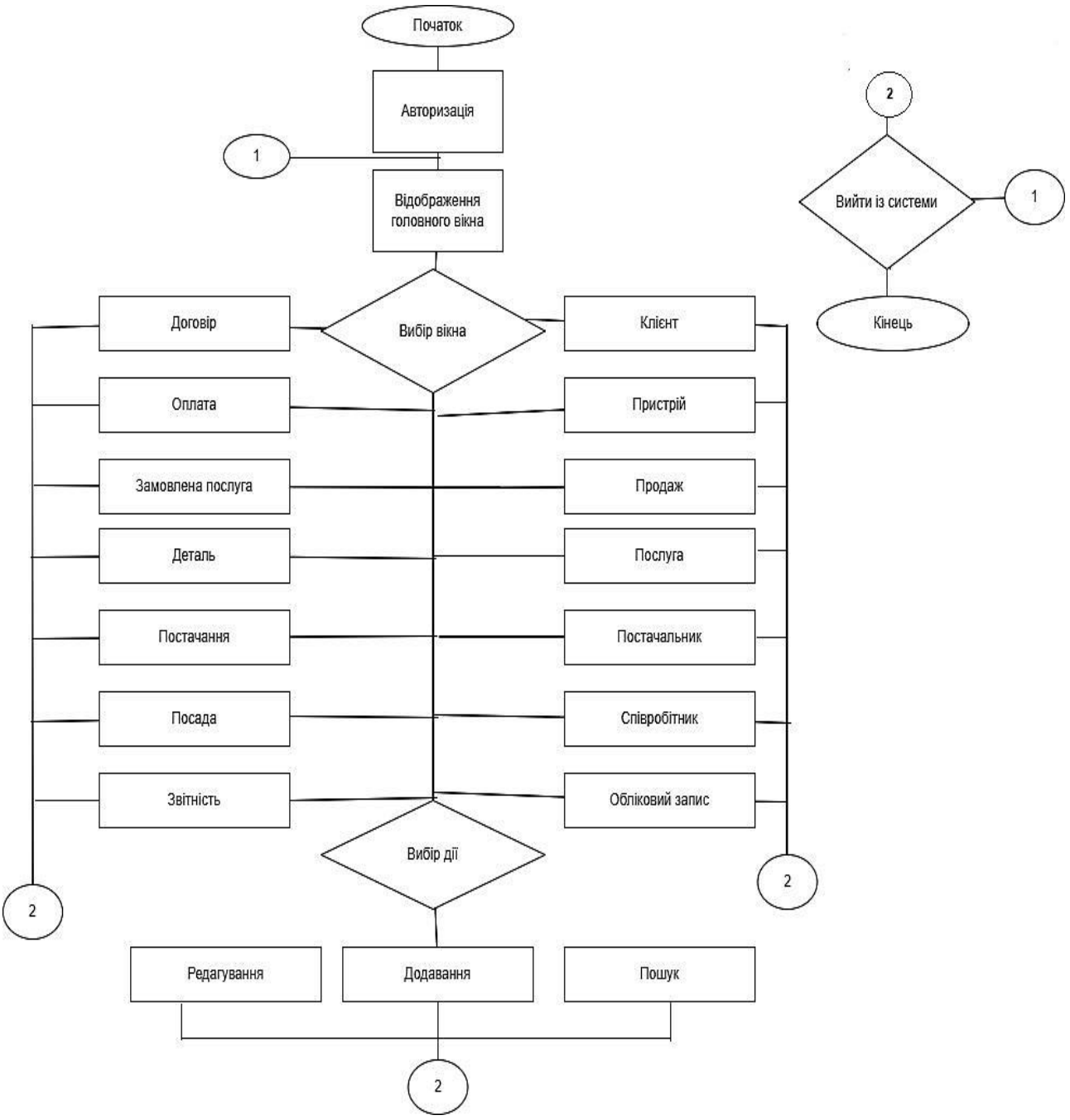

Рисунок 2.2 – Візуалізація алгоритму вибору функції обробки даних

Алгоритм формування звітності, включає наступні етапи: після відображення вікна звітності, слід встановити період звітності, а потім слід вибрати категорію звітності, встановити фільтри, сформувати і вивантажити звіт в Excel. Потім після формування звіту, у разі необхідності – виходимо з системи.

Розроблена схема алгоритму формування звітності представлена на рисунку 2.3.

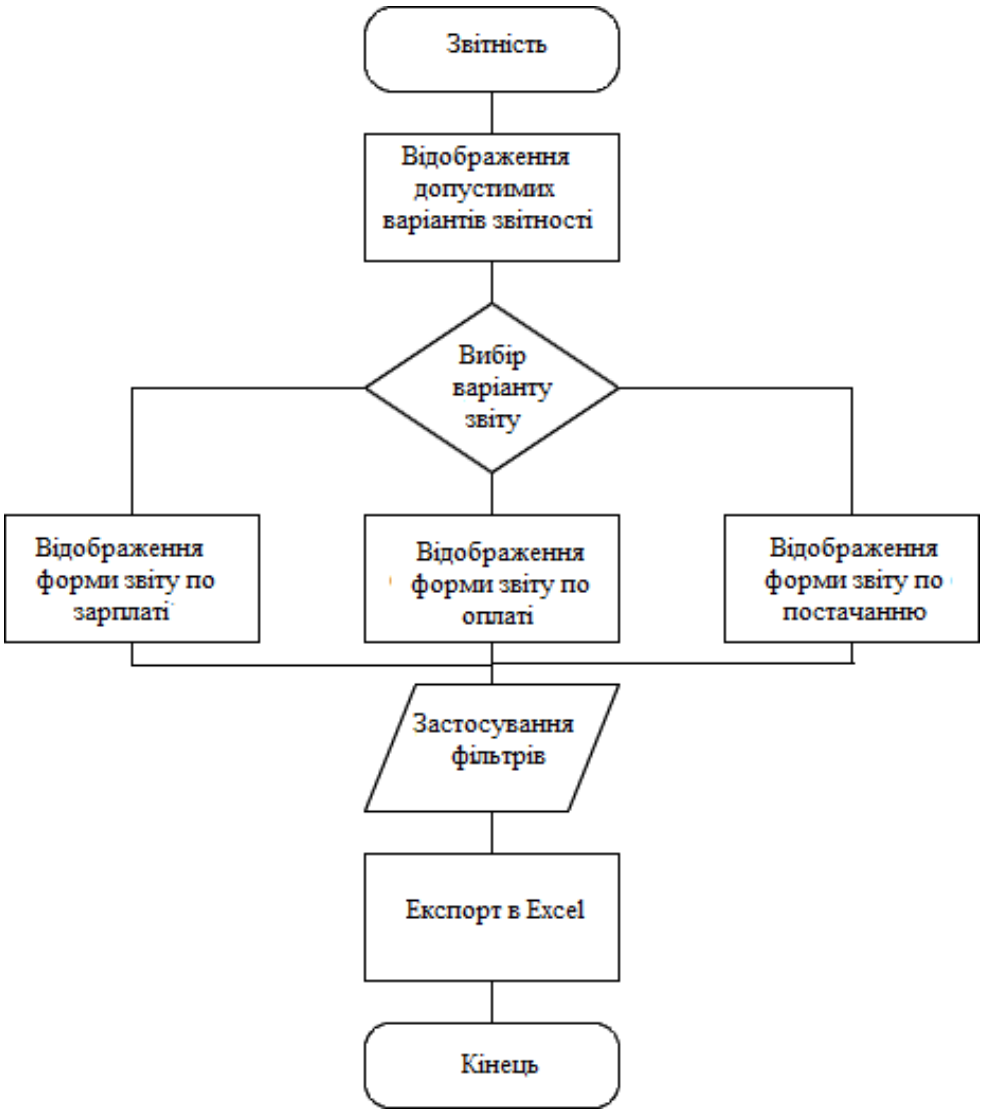

Рисунок 2.3 – Візуалізація алгоритму формування звітності

Для додавання запису в БД розроблений алгоритм, який заключається у наступному: після відображення вікна, що дозволяє додавання, здійснюється введення даних, після чого здійснюється перевірка на коректність і заповненість всіх обов’язкових полів, в тому випадку, якщо дані проходять перевірку успішно, то відбувається запит до сервера, що зберігає записи в БД, в іншому випадку виводитися повідомлення про помилку. Далі, якщо більше немає необхідності додавати дані – здійснюється вихід із системи.

Схема алгоритму додавання запису в БД представлена на рисунку 2.4.

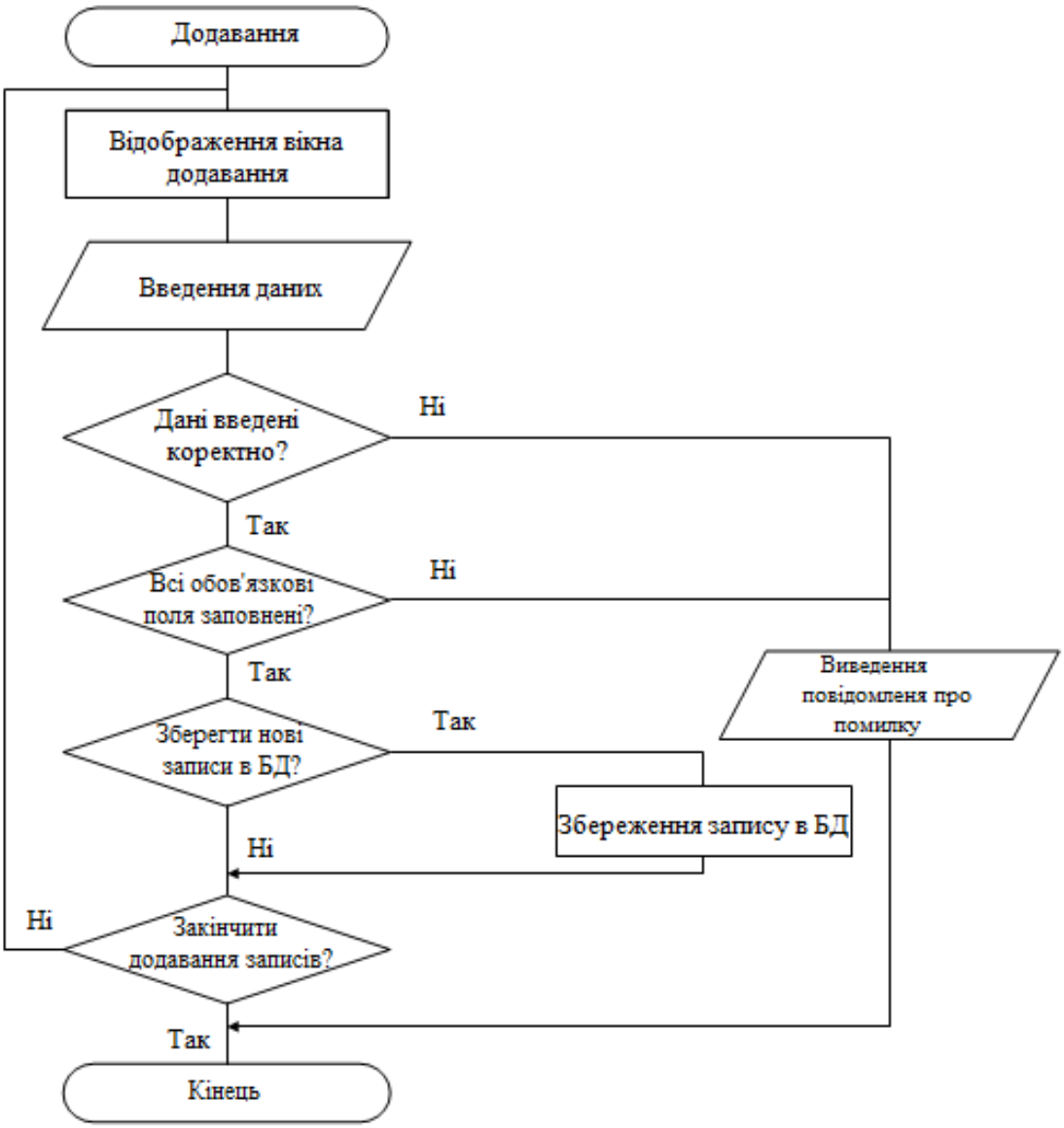

Рисунок 2.4 – Візуалізація алгоритму додавання запису

З метою пошуку запису в БД розроблений алгоритм. Пошук можна здійснювати за такими критеріями, як ПІБ клієнта, модель і тип пристрою.

Алгоритм пошуку запису в БД, включає наступні етапи: відображення наповненої форми, завдання пошукових критеріїв, перевірка чи знайдено запис; якщо запис знайдена, то вона виводиться на екран.

Схема алгоритму пошуку запису в БД представлена на рисунку 2.5.

Рисунок 2.5 – Візуалізація алгоритму пошуку запису в БД

Алгоритм редагування запису в БД полягає в наступному: після відображення форми, що допускає редагування записів, користувачу необхідно спочатку знайти і вибрати запис, який він хоче відредагувати, після того, як він буде обраний потрібно змінити запис і зберегти новий запис в БД.

Розроблена схема алгоритму редагування запису в БД представлена на рисунку 2.6.

Коректне нарахування заробітної плати здійснюється на основі розробленого алгоритму, що складається з наступних кроків: після відображення вікна зарплати, проводиться вибір співробітника, далі перевіряється, чи надана йому премія, і якщо так, вираховується заробітна плата з премією; якщо премія не нараховується, то здійснюється обчислення

заробітної плати без неї. Далі можна повторити процедуру, або ж вийти з системи.

Схема алгоритму нарахування заплати представлена на рисунку 2.7.

Рисунок 2.6 – Візуалізація алгоритму редагування запису

Рисунок 2.7 – Візуалізація алгоритму нарахування зарплати (частина 1)

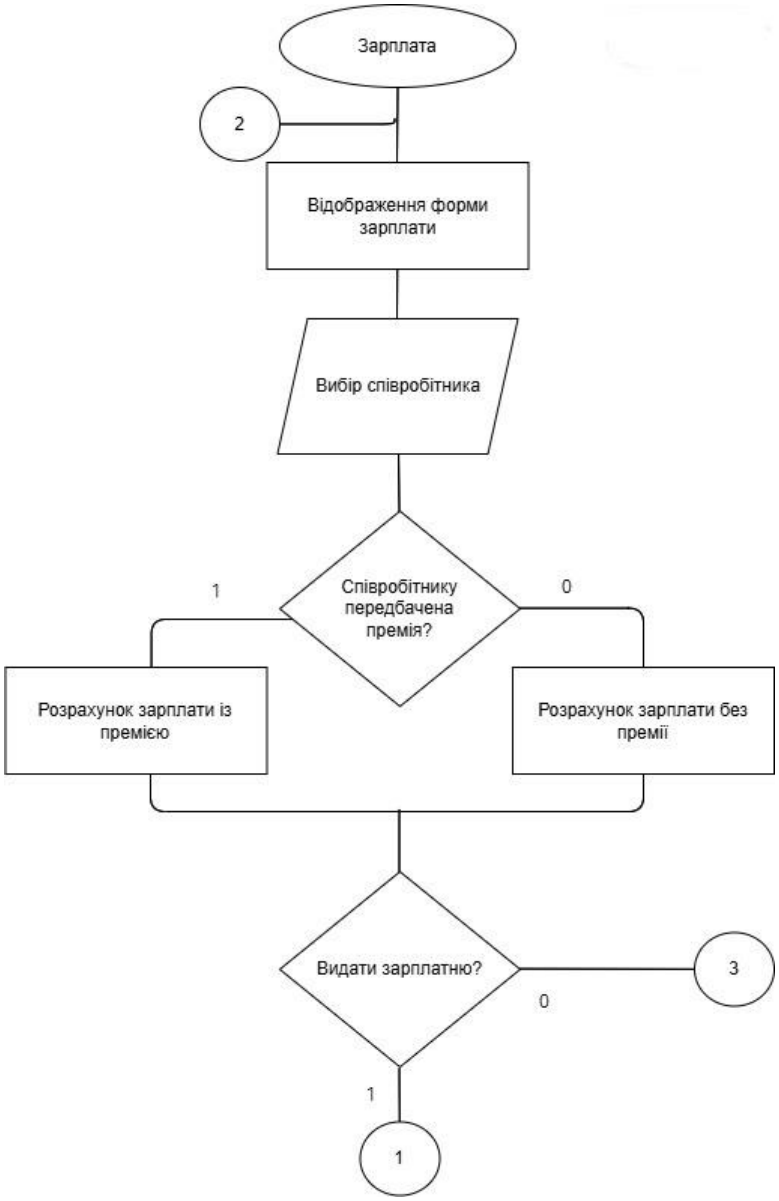

Рисунок 2.7 – Візуалізація алгоритму нарахування зарплати (частина 2)

Отож, для ІС сервісу із обслуговування цифрової техніки були розроблені наступні алгоритми:

– авторизації користувача для обраного сервера;

– вибору функції обробки даних, яка включає додавання, редагування і пошук;

– формування звітності;

– нарахування заробітної плати.

Висновки до розділу 2

У процесі аналізу різних систем управління базами даних (СУБД) і мов програмування було обрано MySQL та C# як оптимальні інструменти для створення автоматизованої системи для сервісного центру. MySQL забезпечує універсальність, простоту інтеграції, безкоштовність і активну підтримку спільноти, що робить її ідеальною для середніх проєктів. Своєю чергою, C# пропонує зручний синтаксис, інтеграцію з платформою .NET, а також ефективні інструменти для створення графічного інтерфейсу та взаємодії з базою даних через Entity Framework. Цей вибір забезпечує баланс між функціональністю, продуктивністю та економічною ефективністю.

Для автоматизації роботи сервісного центру були розроблені алгоритми, які охоплюють ключові функції: авторизацію користувачів, обробку даних (додавання, редагування, пошук), формування звітності та нарахування заробітної плати. Особлива увага приділена коректній взаємодії з базою даних, що забезпечує збереження, обробку й доступ до даних із врахуванням рівнів доступу користувачів. Кожен алгоритм реалізує послідовність операцій із виведенням відповідних повідомлень про помилки або успішні дії, що підвищує зручність роботи з програмою.

Розробка цих алгоритмів формує основу для створення автоматизованої інформаційної системи, яка спрощує управління даними, підвищує точність розрахунків та забезпечує зручність для користувачів. Візуалізація алгоритмів надає можливість їх адаптації до змінних вимог, що робить систему гнучкою та готовою до подальшого розвитку. Впровадження цієї системи сприятиме підвищенню ефективності роботи сервісного центру, покращенню якості обслуговування клієнтів та оптимізації витрат.

РОЗДІЛ 3
РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ДЛЯ СЕРВІСНОГО ЦЕНТРУ НА БАЗІ МОВИ ПРОГРАМУВАННЯ C#

3.1 Практична реалізація об’єкта проектування

Після обґрунтування вимог до програмного забезпечення, його проектування, а також детального опису засобів, методів і алгоритмів для вирішення поставленого завдання, переходимо до опису етапів роботи розробленої автоматизованої системи.

Для візуалізації розробленої системи буде спроектовано інтерфейс користувача.

Інтерфейс користувача (User Interface) є ключовим елементом будь-якої інформаційної автоматизованої системи, оскільки саме через нього користувачі взаємодіють із програмним забезпеченням. Добре спроєктований інтерфейс забезпечує зручність, ефективність і задоволення від роботи з системою, а також сприяє підвищенню продуктивності та мінімізації помилок під час роботи.

До основних вимог до інтерфейсу користувача слід віднести:

– зручність у використанні – інтерфейс має бути інтуїтивно зрозумілим, щоб навіть новачки могли легко взаємодіяти з системою;

– функціональність – інтерфейс повинен надавати доступ до всіх необхідних функцій та інструментів у максимально зручному форматі.

– естетичний дизайн – зовнішній вигляд інтерфейсу повинен бути привабливим, але без зайвих графічних елементів, які можуть відволікати від роботи;

– продуктивність – мінімізація часу на виконання завдань через зрозумілі й швидкі робочі процеси;

– гнучкість – можливість налаштування інтерфейсу відповідно до індивідуальних потреб користувача.

До етапів розробки інтерфейсу користувача можемо віднести:

– аналіз вимог. Перший етап розробки інтерфейсу включає вивчення потреб кінцевих користувачів, визначення основних задач системи та типів операцій, які вони виконуватимуть. Для цього можуть використовуватися опитування, інтерв'ю або спостереження за роботою користувачів. Результатом цього етапу є список функціональних вимог до інтерфейсу;

– розробка концепції дизайну. На основі зібраних вимог створюється концепція інтерфейсу. Вона включає:

1) прототипи – ескізи екранів системи, які демонструють розташування основних елементів. Для створення прототипів використовуються спеціальні інструменти, такі як Figma, Sketch, Adobe XD тощо;

2) архітектура інтерфейсу – визначення структури інтерфейсу, переходів між вікнами та розташування функціональних елементів;

– вибір технологій та інструментів. Для реалізації інтерфейсу обираються технології та інструменти залежно від платформи, на якій працюватиме система. Для веб-додатків найчастіше використовуються HTML, CSS, JavaScript, а для настільних застосунків – фреймворки, як-от WPF (C#) або JavaFX (Java);

– реалізація дизайну**.** На цьому етапі програмісти втілюють дизайн у код, додаючи інтерактивність до графічних елементів. Важливим є дотримання принципів адаптивності для коректного відображення на різних пристроях і екранах.

– тестування інтерфейсу. Після створення інтерфейсу його обов'язково тестують на зручність (usability testing). Користувачі перевіряють роботу системи на практиці, виявляють проблеми або незручності, після чого інтерфейс доопрацьовується;

– впровадження та підтримка. На фінальному етапі інтерфейс інтегрується в інформаційну систему, а користувачам надається навчання або документація для ефективного використання. Додатково відбувається моніторинг роботи інтерфейсу для подальших покращень.

Коротко опишемо принципи дизайну інтерфейсу для автоматизованих систем.

Мінімалізм – уникання зайвих елементів на екрані для забезпечення чіткої навігації та концентрації на важливих функціях.

Спрямування на задачі – інтерфейс має відповідати конкретним задачам користувачів, що спрощує роботу в системі.

Послідовність – використання однакових шрифтів, кольорів, кнопок і меню в усіх розділах системи для спрощення навігації.

Зворотний зв’язок – надання користувачам інформації про їхні дії через спливаючі повідомлення, підказки або індикатори завантаження.

Інтерфейс автоматизованих систем зазвичай включає такі компоненти:

– форми введення даних – для створення, редагування чи пошуку інформації в базі даних;

– меню та навігація – зручні панелі або вкладки для переходу між модулями системи;

– панелі звітності – для виводу результатів обробки даних у вигляді таблиць, графіків або звітів;

– інтерактивні кнопки та фільтри – для швидкого доступу до функцій та управління даними.

Таким чином, інтерфейс користувача є важливою складовою інформаційних автоматизованих систем, оскільки він визначає ефективність роботи користувачів. Успішна розробка інтерфейсу вимагає комплексного підходу, що поєднує в собі аналіз потреб, проєктування, реалізацію та тестування для забезпечення найкращого досвіду користувача.

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

Для інформаційної системи потрібно було створити прикладне програмне забезпечення з інтуїтивно зрозумілим інтерфейсом. Передбачалося реалізувати такі функції обробки даних, як:

– поповнення бази даних;
– зміна (редагування та видалення) даних;
– нарахування заробітної плати;
– пошук за заданими критеріями;
– формування звітності.

У відповідності до вимог розроблено прикладне програмне забезпечення для сервісу з обслуговування цифрової техніки, яка реалізована на основі алгоритмів, розроблених в розділі 2.

При запуску програми на екрані з’являється вікно вибору сервера і авторизації користувача, в якому необхідно ввести логін і пароль, а також, при необхідності, адресу сервера, на відмінний від адресу за замовченням. Внесення змін до облікових записів доступно користувачеві, що володіє правами адміністратора ІС, з головного меню програми. Інтерфейс системи спроектований з урахуванням підтримки роботи недосвідчених користувачів без необхідності додаткового навчання. У випадку якщо дані вірні, відбувається відкритті основної форми з елементами доступними для користувача, у відповідності з його рівнем доступу.

Форма вибору сервера та авторизації користувача продставлена нижче на рисунку 3.1.

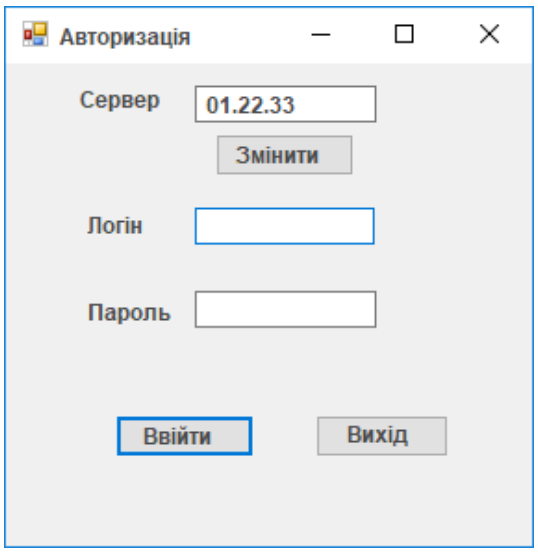

Рисунок 3.1 – Форма вибору сервера і авторизації користувача

Після входу користувач попадає в головне вікно програми, що містить список всіх доступних йому таблиць.

Також на основному вікні користувач здатний ознайомитися з тим, які операції, коли і в який час були виконані в цьому випадку. Користувач може внести зміни в базу даних напряму, таблиці представлені саме в тому вигляді, в якому вони зберігаються на сервері. Основне вікно для користувача з повним доступом подано на рисунку 3.2.

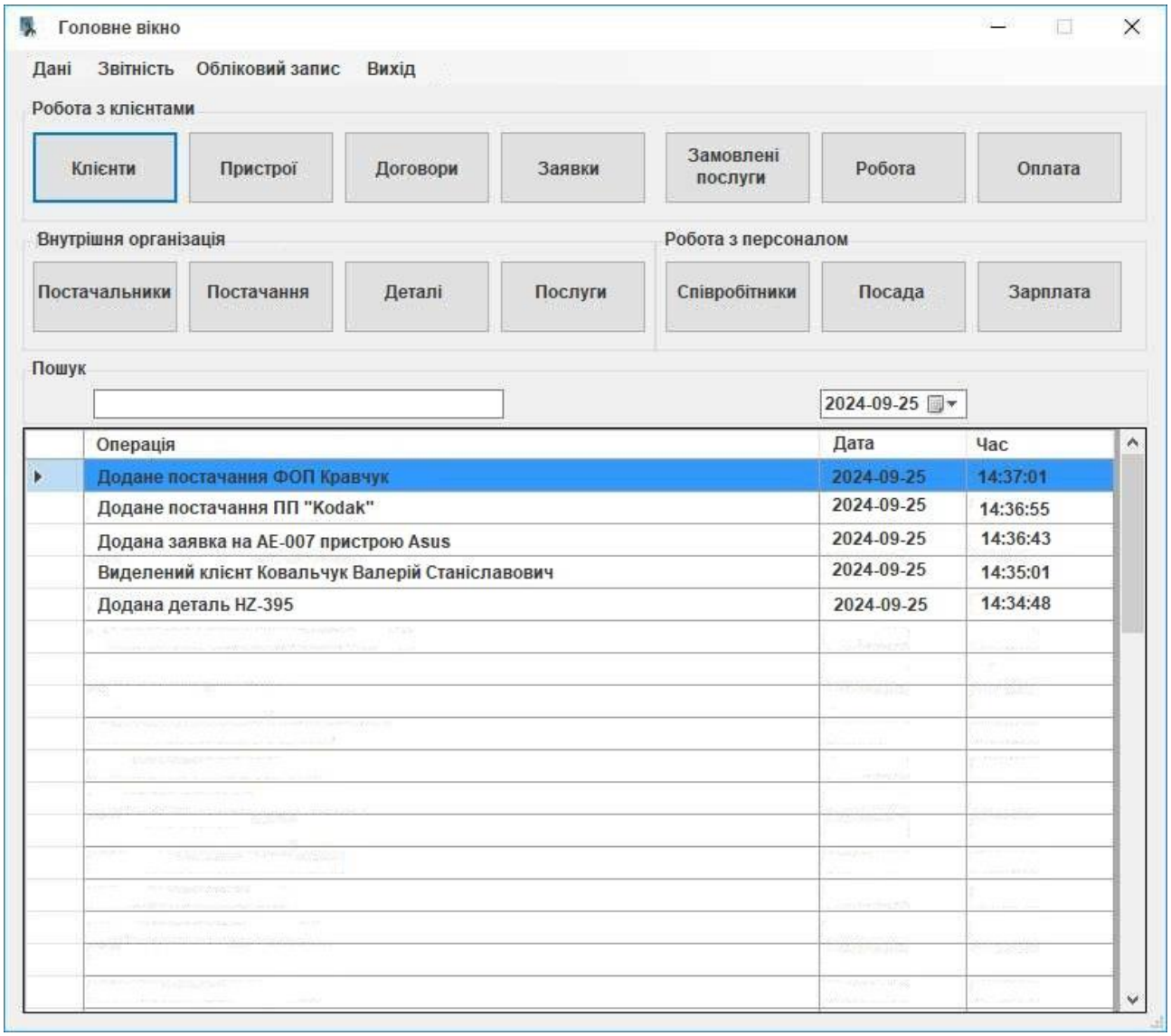

Рисунок 3.2 – Основне вікно програми

Історію виконаних операцій можна відтворювати за датою, певному клієнту або співробітнику, що виконав роботи, а також за будь-якими видами

дій. Користувач може подивитися всю необхідну йому інформацію про те, що сталося за останній час і які завдання йому потрібно виконати (рис. 3.3).

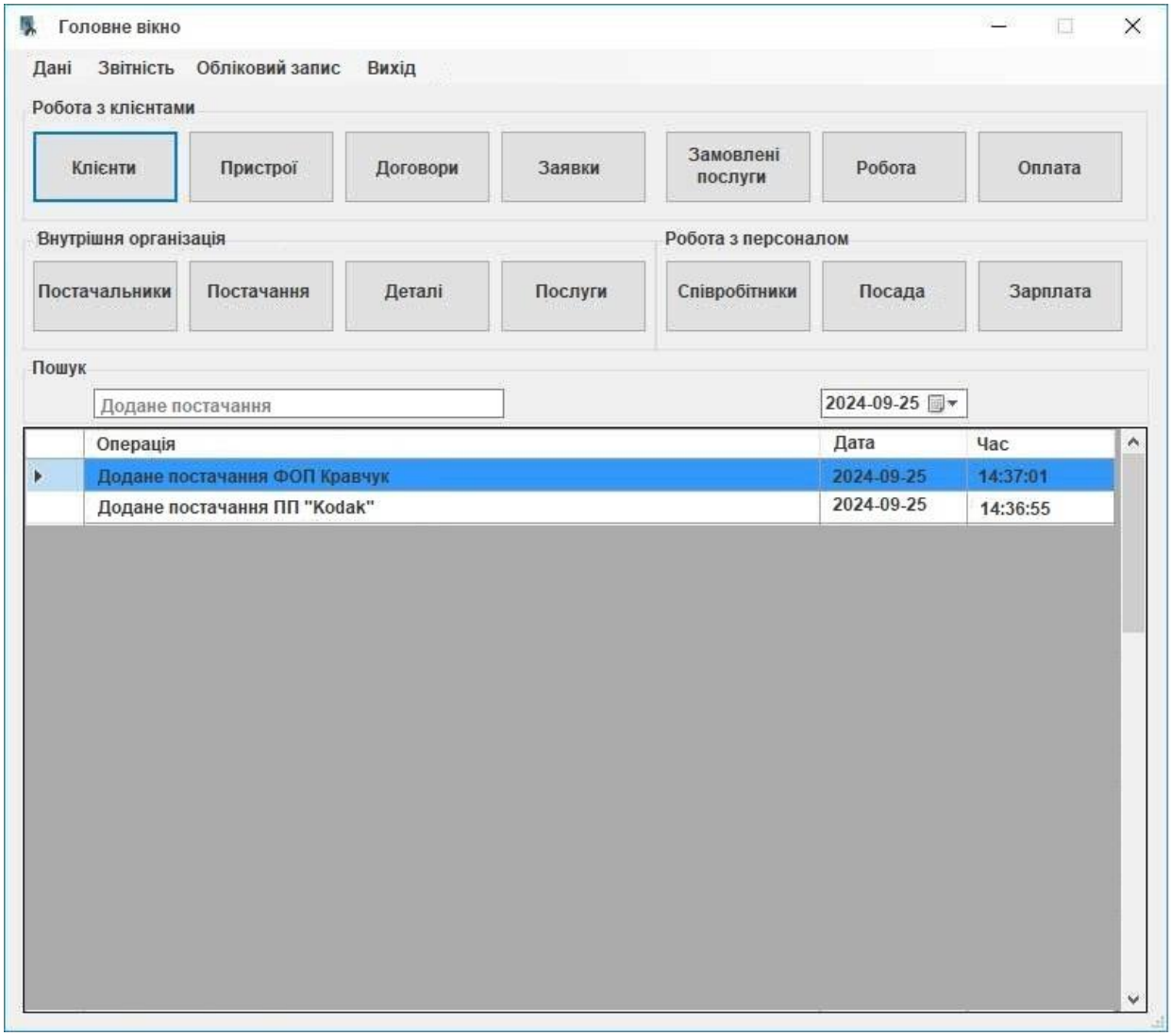

Рисунок 3.3 – Форма пошуку

Далі користувач може вибрати необхідну таблицю зі списку і продовжити роботу, виконавши необхідні дії, такі як додавання клієнтів, оформлення договорів та ін. Після додання клієнта необхідно оформити його пристрій, а потім договір. Після оформлення договору користувач може приступити до оформлення замовлення клієнта на деталі і послуги (рис. 3.4).

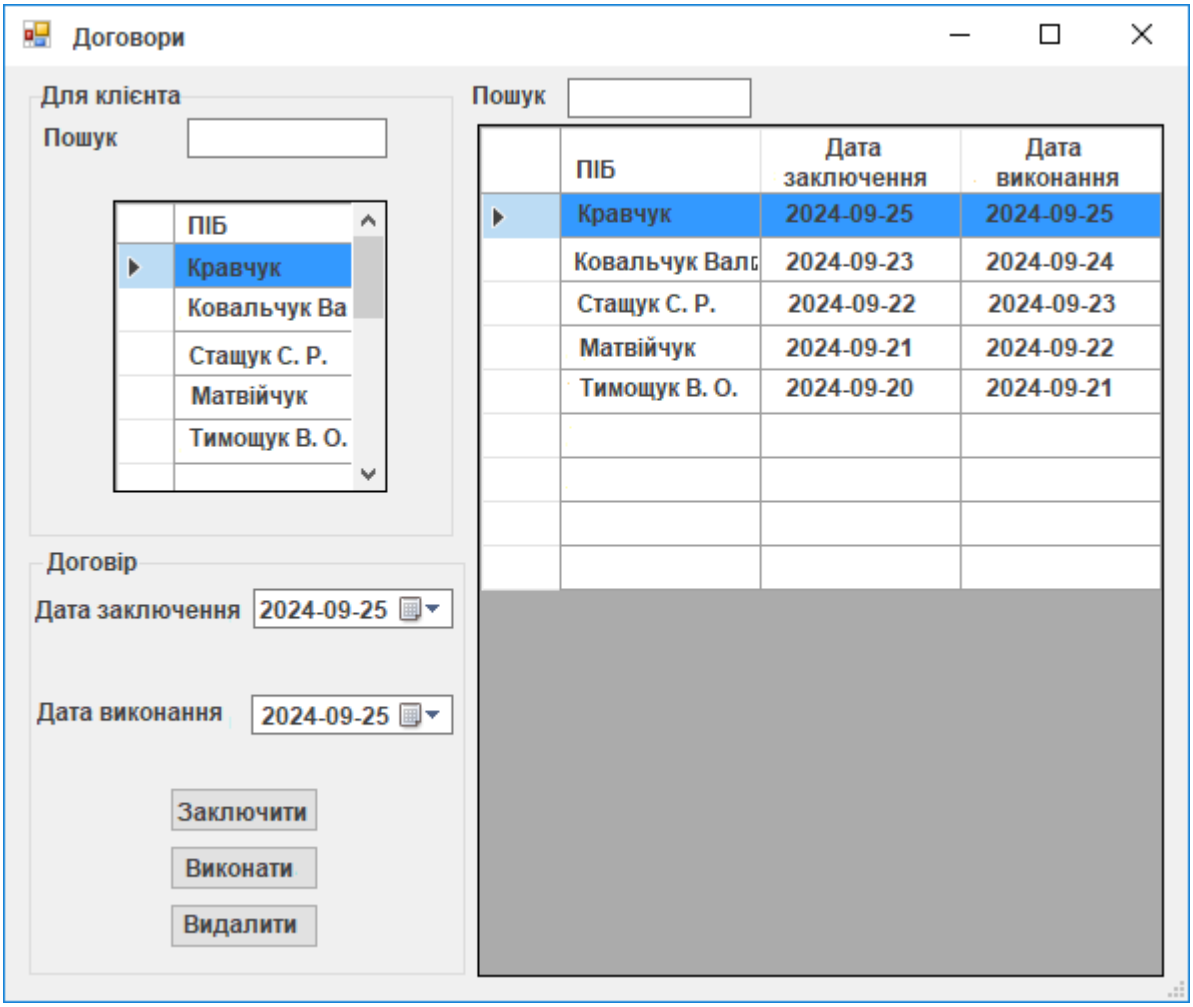

Рисунок 3.4 – Форма договору

Для оформлення роботи, замовленої клієнтом, користувачеві необхідно вибрати послугу і договір, для якого необхідно виконати.

Для контролю за виконанням замовлених послуг служить таблиця виконані роботи.

У дальшому потрібно приймати від клієнта оплати за всі замовлені послуги. У вікні оплати обрахунок проходить автоматично. Наприклад, щоб обрахувати суму замовлених товарів і послуг для клієнта необхідно вибрати клієнта, що формував замовлення зі списку договорів, повна вартість буде розраховано автоматично. Існує також можливість предоплати товару, готівковий і безготівковий рахунок. Після того, як оплата повністю виконана, і всі послуги, замовлені в договорі виконані, – договір необхідно закрити

При повторному зверненні клієнта необхідно вибрати його зі списку клієнтів, і оформити новий договір. Для зручності користувача по всіх таблицях передбачений пошук по кількох критеріях. Також у програмі передбачено додавання даних. За завданням було необхідно реалізувати функцію редагування даних. Спочатку необхідно за допомогою пошуку знайти потрібний запис, а потім, вибравши редагування, змінити дані (рис. 3.5).

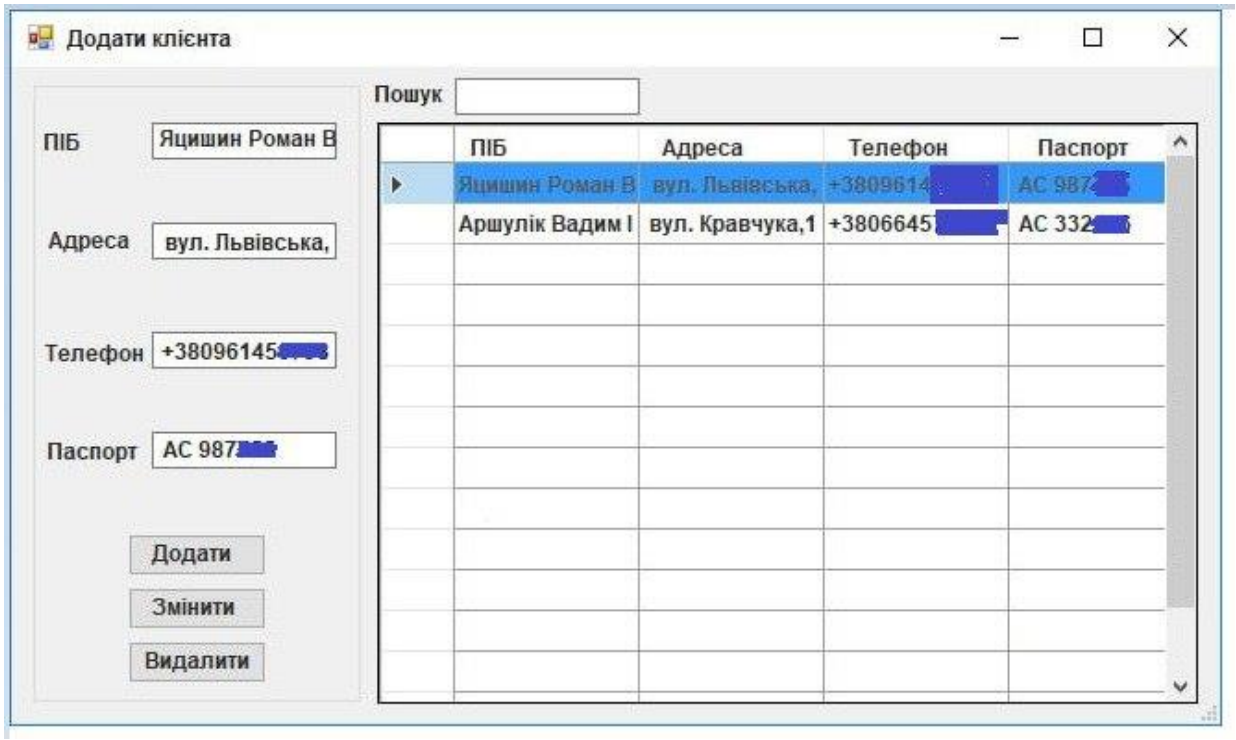

Рисунок 3.5 – Форма клієнта

При видачі співробітнику заробітної плати враховуються такі фактори, як оклад, що відповідає зайнятій ним посаді, а також премія.

Для обробки зібраної інформації та отримання зведених даних в зручному для перегляду і аналізу виді, передбачено формування звітів за критеріями, такими, як, зарплата, оплата і постачання. Оскільки звіти, в більшості випадків, використовуються для формування вихідних документів, вони можуть бути експортовані у формат Microsoft Excel і збережені у файл на диску для подальшого використання або перенесення на інші комп’ютери.

У програмі передбачена можливість додавання нових облікових записів для користувачів, а також передбачена зміна пароля і логіну для наявних. Форми механізму управління обліковими записами представлені на рисунку 3.6.

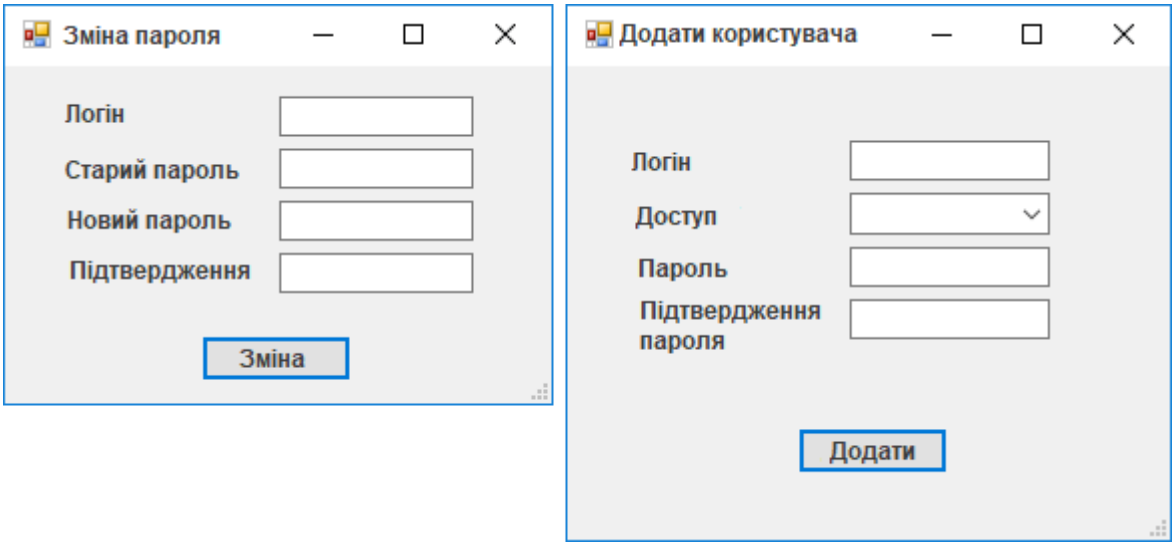

Рисунок 3.6 – Форми механізму облікових записів

Розроблений продукт пройшов тестування, де показало, що він реалізує такі функції як вибір сервера, авторизацію користувача, додавання, редагування. видалення даних, моніторинг проведених операцій, формування звітності і нарахування заробітної плати.

3.3 Розробка бази даних

На етапі концептуального проєктування, спираючись на словесний опис предметної області, виконуються такі завдання:

– визначення об’єктів предметної області, їхніх атрибутів і первинних ключів;

– встановлення зв’язків між об’єктами та визначення їхньої кількості;

– побудова концептуальної моделі бази даних за допомогою підходу «Сутність-зв’язок».

Модель «сутність – зв’язок» являє собою графічне представлення об’єктів розглянутої предметної області, характеристик цих об’єктів та зв’язків між ними.

Для даної системи, як основні інформаційні об’єкти предметної області, виділяються: «договір», «клієнт», «пристрій», «замовлена послуга», «послуга», «оплата», «виконана робота», «постачальник», «постачання», «заявка», «деталь», «працівник», «посада», «заплата».

Між об’єктами встановлюємо відповідні логічні відношення (зв’язки).

Для зв’язку «клієнт – пристрій», на підставі того, що одному клієнту можуть належати кілька пристроїв, а один пристрій тільки клієнту, приймаємо відношення «один-до-багатьох». Для зв’язку «клієнт – договір», на підставі того, що один клієнт спроможний укласти кілька договорів, а один договір може бути укладений тільки на одного клієнта, приймаємо відношення «один-до-багатьох».

Для зв’язку «договір – оплата», на підставі того, що по одному договору можливе проведення кількох оплат, а одна оплата на один договір, приймаємо відношення «один-до-багатьох».

Для зв’язку «постачальник – постачання», на підставі того, що один постачальник спроможний здійснювати безліч постачань, а одне постачання може здійснюватися одним постачальником – приймаємо відношення «один-до-багатьох».

Для зв’язку «заявка – постачання», з огляду на те, що на підставі однієї заявки можуть здійснюватися кілька постачань, а одне постачання – для однієї конкретної заявки, приймаємо відношення «один-до-багатьох».

Для зв’язку «договір – заявка», на підставі того, що на один договір може бути оформлено безліч заявок, а одна заява пов’язана лише з одним договором – обираємо відношення «один-до-багатьох».

Для зв’язку «деталь – заявка», на підставі того, що одна деталь може згадуватися в великій кількості заявок, а одна заявка включає лише одну деталь, то обираємо відношення «один-до-багатьох».

Для зв’язку «договір – замовлена послуга», на підставі того, що на один договір можуть бути оформлені кілька замовлених послуг, але для одного замовлення може бути лише один договір, приймаємо відношення «один-до-багатьох».

Для зв’язку «замовлена послуга – послуга», на підставі того, що одна замовлена послуга здатна включати в себе декілька послуг, а одна послуга може включати одну замовлену послугу, приймаємо відношення «один-до-багатьох».

Для зв’язку «посада – співробітник», на підставі того, що на одній і тій же посаді можуть працювати кілька співробітників, а один працівник завжди обіймає лише одну посаду, приймаємо відношення «один-до-багатьох».

Для зв’язку «співробітник – заробітна плата», на підставі того, що один працівник може отримувати кілька заробітних плат, а конкретна заробітна плата належить одному працівнику, приймаємо відношення «один-до-багатьох».

Для зв’язку «співробітник – виконана робота», на підставі того, що один працівник виконує кілька робіт, а конкретна виконана робота може бути зроблена одним працівником – приймаємо відношення «один-до-багатьох».

Для зв'язку «замовлена послуга – виконана робота», на підставі того, що одна замовлена послуга може отримати статус однієї виконаної роботи, а конкретна виконана робота включає в себе одну замовлену послугу, приймаємо відношення «один-до-багатьох».

Модель «сутність – зв’язок» бази даних сервісу для обслуговування цифрової техніки представлена на рисунку 3.7.

На етапі логічного проектування концептуальна модель переходить в логічну, опираючись на правила перетворення, та враховуючи обрану реляційну модель.

Всі об’єкти концептуальної моделі відображаються в таблиці БД.

Рисунок 3.7 – Модель «сутність – зв’язки» бази даних

За причиною того, що в концептуальній моделі бази даних для ІС сервісу з обслуговування цифрової техніки зв’язки між об’єктами мають відношення «один-до-багатьох», то не потрібно створювати додаткові сполучення таблиць в логічну БД.

Проводимо перетворення на зовнішні ключі, зв’язків «один-до-багатьох». Перетворення проходить шляхом включення зв’язку в таблицю, що описує об’єкт зі сторони «багатьох», як зовнішній ключ ключового атрибуту об’єкта зі сторони «один».

У таблиці «оплата» виділяємо такі атрибути як: «код оплати», «код договору», «сума», «вид оплати», «дата», «грошовий внесок», «залишок», «готовність». Статус первинного ключа придаємо цілочисельному полю «код оплати». Статус зовнішнього ключа присвоюється полю «код договору».

У таблиці «договір» виділяємо такі атрибути як: «код договору», «код клієнта», «дата укладання», «дата виконання». Статус первинного ключа придаємо цілісному полю «код договору». Статус зовнішнього ключа присвоюється полю «код клієнта».

У таблиці «послуга» виділяємо такі атрибути як: «код послуги», «вид», «назва», «вартість». Статус первинного ключа придаємо цілочисельному полю «код послуги».

У таблиці «пристрій» виділяємо такі атрибути як: «код пристрою», «код клієнта», «тип», «модель», «рік випуску», «номер місця» «дата». Статус первинного ключа придаємо цілочисельному полю «код пристрою». Статус зовнішнього ключа присвоюється полю «код клієнта».

У таблиці «співробітник» виділяємо такі атрибути як: «код співробітника», «код посади», «ПІБ» і «дата народження». Статус первинного ключа придаємо цілочисельному полю «код співробітника». Статус зовнішнього ключа присвоюється полю «код посади».

У таблиці «постачання» виділяємо такі атрибути як: «код постачання», «код постачальника», «код заявки», «кількість», «дата», «вартість». Статус первинного ключа придаємо цілочисельному полю «код постачання». Статус зовнішнього ключа присвоюється полю «код заявки».

У таблиці «виконана робота» виділяємо такі атрибути як: «код виконаної роботи», «код замовленої послуги», «код співробітника», «кількість годин», «дата». Статус первинного ключа придаємо цілісному полю «код виконаної роботи». Статус зовнішнього ключа присвоюється полям «замовлена послуга» і «код співробітника».

У таблиці «заявка» виділяємо такі атрибути як: «код заявки», «код договору», «код деталі», «кількість» і «вартість». Статус первинного ключа придаємо цілочисельному полю «код заявки». Статус зовнішнього ключа присвоюється полям «код договору» і «код деталі».

У таблиці «деталь» виділяємо такі атрибути як: «код деталі», «назва», «фірма» і «характеристики». Статус первісного ключа прийдемо цілісному полю «код деталі».

У таблиці «клієнт» виділяємо такі атрибути як: «код клієнта», «ПІБ», «адреса», «телефон» і «паспорт». Статус первинного ключа придаємо цілочисельному полю «код клієнта».

У таблиці «постачальник» виділяємо такі атрибути як: «код постачальника», «назва», «телефон» і «адреса».

В таблиці «замовлена послуга» виділяємо такі атрибути як: «код замовленої послуги», «код договору», «код послуги», «кільеість» «вартість», «готовність». Статус зовнішнього ключа присвоюється полям «код договору» і «код послуги».

У таблиці «посада» виділяємо такі атрибути як: «код посади», «назва», «оклад» і «премія». Статус первинного ключа придаємо цілісному полю «код посади».

У таблиці «заплата» виділяємо такі атрибути як: «код заплати», «код співробітника», «сума» і «дати видачі». Статус первинного ключа придаємо цілочисельному полю «код зарпплати». Статус зовнішнього ключа присвоюється полю «код співробітника».

З метою запобігання від надмірності даних потрібно здійснити нормалізацію таблиць (відносин) БД. На практиці достатньо, щоб відношення знаходилося в 3 нормальній формі (3НФ). Відношення задовольняє 3 нормальній формі, тоді і тільки тоді, коли відношення знаходиться у другій нормальній формі, і відсутні транзитні функціональні залежності неключових атрибутів від ключових.

Спроектовані для БД таблиці перевірені на відповідність вимогам 3 нормальної форми.

Логічна схема бази даних для ІС сервісу з обслуговування цифрової техніки представлена на рисунку 3.8.

Етап фізичного проектування представляє собою третій і останній етап проектування БД, який підсумовує прийняття рішень про способи реалізації створеної бази даних.

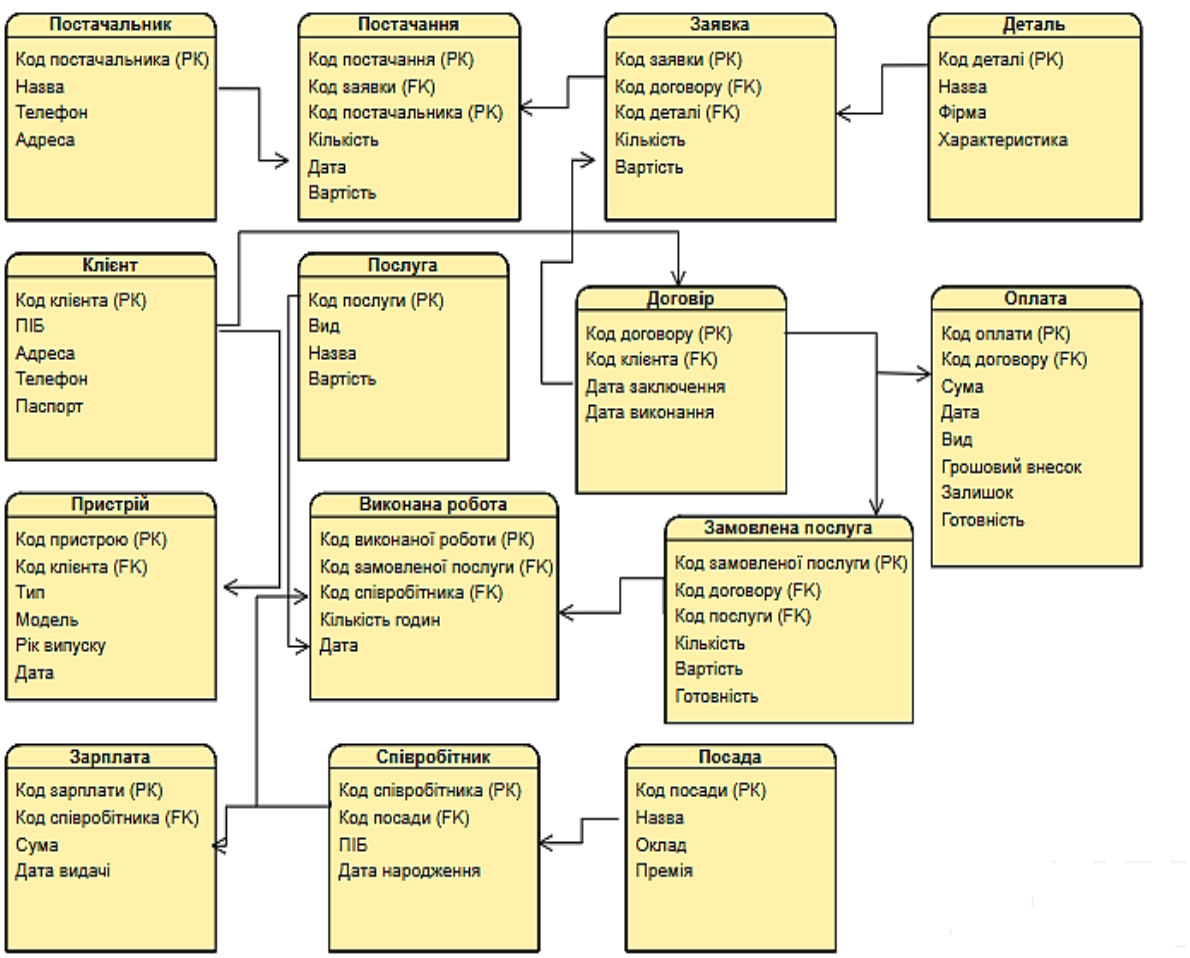

Рисунок 3.8 – Логічна схема бази даних сервісу

Між логічним і фізичним проєктуванням існує тісний зворотний зв’язок, оскільки рішення, прийняті на етапі фізичного проєктування для підвищення продуктивності системи, можуть впливати на структуру логічної моделі даних. Зазвичай головною метою фізичного проєктування бази даних є визначення способу фізичної реалізації логічного проєкту.

При роботі з раляційною моделлю даних підсумовується наступне:

– визначення конкретних структур зберігання даних і методів доступу до них, що забезпечують оптимальну продуктивність СУБД;

– розробка комплекту реляційних таблиць і обмежень для них, на основі інформації, одержаної з повної логічної моделі даних.

З фізичної схемою бази даних, розробленою для ІС сервісу з обслуговування цифрової техніки, в графічному представленні можна ознайомитися на рисунку 3.9.

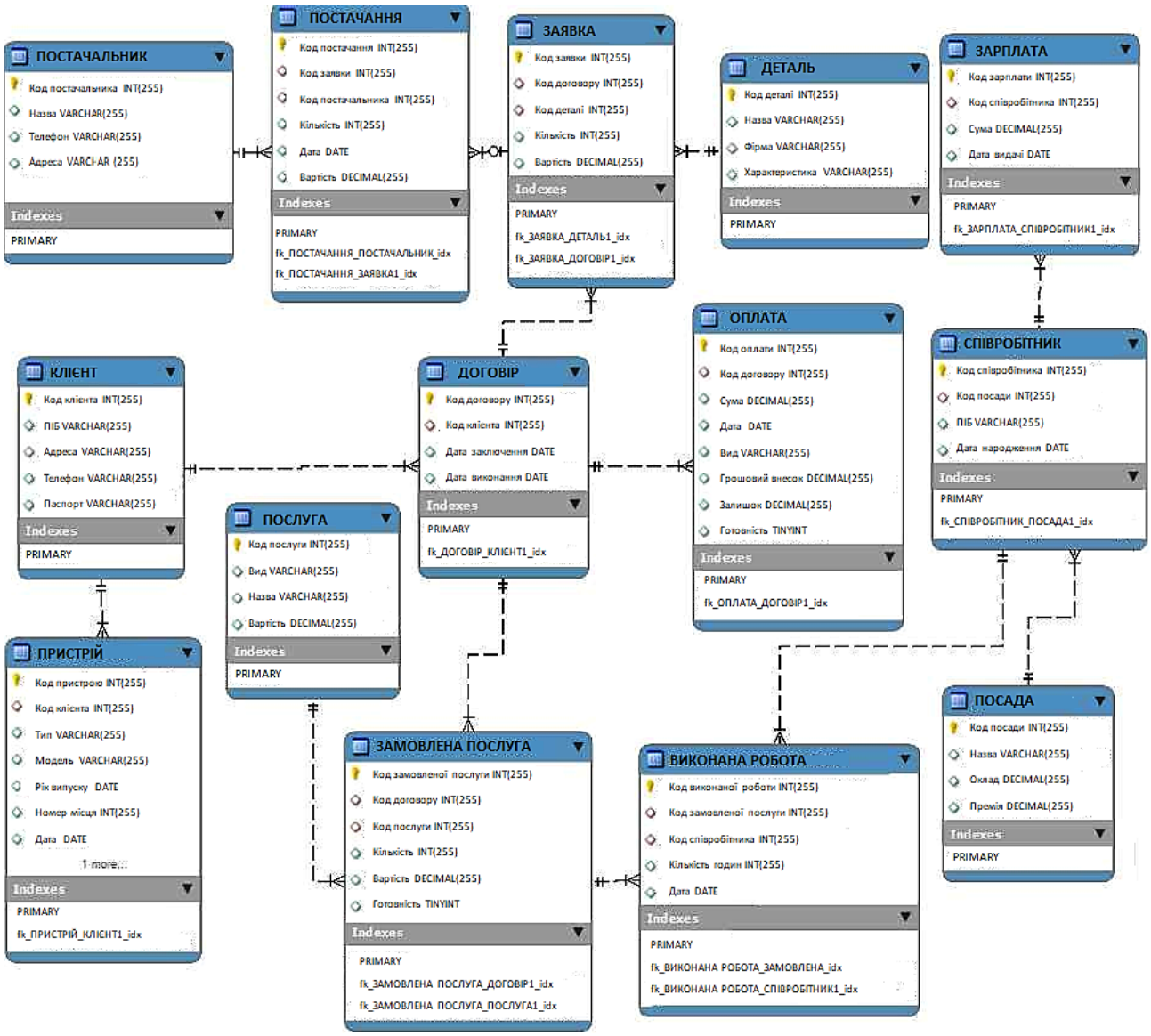

Рисунок 3.9 – Фізична схема БД сервісу

Підводячи підсумок, зазначимо, що була створена концептуальна модель даних, яка є графічним відтворенням об’єктів, розглянутої предметної області, характеристик цих об’єктів і відношень між об’єктами. Далі концептуальна модель була перетворена в логічну схему з урахуванням обраної реляційної моделі даних за правилами перетворення. Потім з урахуванням СУБД MSQL була побудована фізична модель БД. Спроектована база даних має мінімальну надмірність, так як кожна з таблиць задовольняє 3НФ.

Висновки до розділу 3

У третьому розділі було здійснено практичну реалізацію інформаційної системи для сервісного центру на основі мови програмування C#. Особливу увагу приділено розробці зручного, інтуїтивно зрозумілого інтерфейсу користувача. Завдяки аналізу потреб користувачів, створено ефективний та функціональний UI, що включає компоненти для внесення даних, звітності, а також навігації між модулями системи. Реалізовано основні принципи дизайну, такі як мінімалізм, послідовність та зворотний зв’язок, що дозволяють підвищити продуктивність та знизити ймовірність помилок під час використання системи.

Процес тестування та налагодження показав, що система успішно виконує функції, поставлені в рамках завдання: обробка даних, формування звітності, нарахування заробітної плати, а також контроль виконаних операцій. Реалізовано багаторівневу систему доступу для користувачів, що забезпечує гнучкість та безпеку в роботі. Завдяки автоматизації функціональних завдань, значно полегшується робота сервісного центру, а можливість інтеграції звітів у формат Excel підвищує зручність аналітичної обробки даних.

На етапі розробки бази даних побудовано концептуальну модель на основі підходу «Сутність-зв’язок», що дозволило визначити основні об'єкти предметної області, їхні атрибути та зв’язки. Впровадження логічної моделі БД забезпечило ефективне збереження та обробку даних, зокрема завдяки реляційній структурі та використанню зовнішніх ключів для встановлення відносин «один-до-багатьох». Таким чином, розроблена інформаційна система відповідає поставленим вимогам, є ефективною та готовою до впровадження для автоматизації сервісного обслуговування клієнтів.

ВИСНОВКИ

Проведений аналіз сучасних автоматизованих систем управління, таких як Oracle E-Business Suite, Microsoft Dynamics та інші, показав їх високий потенціал для автоматизації бізнес-процесів. Проте висока вартість впровадження, складність налаштування та залежність від технічної підтримки роблять їх недоступними для малих і середніх підприємств. На основі цього обґрунтовано необхідність створення власної інформаційної системи, що відповідала б специфічним потребам сервісного центру, спрощуючи управління ресурсами та підвищуючи ефективність бізнес-процесів.

Серед розглянутих мов програмування (Java, C++ і C#), C# було обрано завдяки її інтеграції з платформою .NET, зручному синтаксису, широким можливостям для створення графічного інтерфейсу та ефективності в роботі з базами даних. Ця мова дозволила забезпечити швидку розробку програмного забезпечення з урахуванням потреб сервісного центру, а також створити систему, адаптовану для користувачів із різним рівнем технічної підготовки.

Було створено реляційну базу даних із використанням MySQL, яка забезпечує збереження, обробку й ефективне управління даними. Для цього побудовано концептуальну модель на основі підходу «Сутність-зв’язок» із подальшим перетворенням у логічну й фізичну структури. База даних відповідає вимогам третьої нормальної форми (3НФ), що мінімізує надмірність даних і підвищує продуктивність системи.

У системі реалізовано алгоритми, що включають авторизацію користувачів, обробку даних (додавання, редагування, пошук), формування звітності та нарахування заробітної плати. Інтерфейс користувача розроблено з акцентом на мінімалізм, зручність та інтуїтивність, що забезпечує ефективну взаємодію з системою. Використано адаптивний дизайн для коректної роботи на різних пристроях.

Розроблене програмне забезпечення пройшло повне тестування на відповідність функціональним і нефункціональним вимогам. Система

забезпечує точність обробки даних, безпеку доступу та зручність використання. Тестування підтвердило, що система здатна автоматизувати ключові бізнес-процеси сервісного центру, оптимізувати облік ресурсів і підвищити якість обслуговування клієнтів.

Таким чином, виконані завдання підтверджують успішність розробки інформаційної системи, яка має високу практичну цінність для автоматизації роботи сервісних центрів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

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

2. Oracle E-Business Suite: Innovations in 2019. URL: https://www.oracle.com/a/ocom/docs/announcing-ebs-innovations-2019.pdf (дата звернення: 20.11.2024).

3. Newell E. Mastering Microsoft Dynamics 365 Implementations. 1st ed. Hoboken: Wiley, 2022. 416 p.

4. Simonov D. Detailed New Features of Firebird 5. URL: https://firebirdsql.org/en/community-news/free-book-detailed-features-of-firebird-5-is-published/ (дата звернення: 20.11.2024).

5. SAP Adaptive Server Enterprise: Configuration Guide for UNIX. URL: https://help.sap.com/docs/SAP_ASE (дата звернення: 21.11.2024).

6. Embarcadero Technologies. InterBase 2021 Update 3 Documentation. URL: https://docwiki.embarcadero.com/InterBase/2021/en/Main_Page (дата звернення: 21.11.2024).

7. Schönig H.-J. Mastering PostgreSQL 15: Advanced techniques to build and manage scalable, reliable, and fault-tolerant database applications. 5th ed. Birmingham: Packt Publishing, 2023. 522 p.

8. West R., Assaf W., Noble E., Longoria M., D'Antoni J., Davidson L. SQL Server 2022 Administration Inside Out. 1st ed. Redmond: Microsoft Press, 2023. 992 p.

9. Beaulieu A. Learning SQL: Generate, Manipulate, and Retrieve Data. 3rd ed. Sebastopol: O'Reilly Media, 2020. 384 p.

10. Nichter D. Efficient MySQL Performance: Best Practices and Techniques. 1st Ed. Sebastopol: O'Reilly Media, 2021. 275 с.

11. Урма Р.-Г., Фуско М., Майкрофт А. Сучасна Java в дії: функціональне програмування, лямбда-вирази, потоки API та сучасні розширення Java. 2 -ге вид. Нью-Йорк: Manning Publications, 2020. 592 с.

12. Іванов Є. О., Ліндер Я. М., Жереб К. А. Основи мови програмування C++: навчальний посібник. К.: Логос, 2020. 90 с.

13. Troelsen Andrew, Japikse Phil. Pro C# 10 with .NET 6: Foundational Principles and Practices in Programming. Eleventh Edition. New York: Apress, 2022. 1680 p.