

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ВІННИЦЬКИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ ІНСТИТУТ**

Кафедра інноваційної економіки та цифрових технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

**«ПРОЄКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ БАЗОЮ
ДАНИХ ІНТЕРНЕТ-МАГАЗИНУ»**

(за матеріалами Фізичної особи-підприємця
Лебедкін Максим Олексійович, м. Вінниця)

Здобувача вищої освіти
3 курсу, групи ІСТ-31дсп,
спеціальності 126
«Інформаційні системи та
технології»
освітньої програми
«Інформаційні технології у
бізнесі»

Богдана ІВЧУКА

Науковий керівник
кандидат технічних наук, доцент

Руслан НОВИЦЬКИЙ

Гарант освітньо-професійної
програми
кандидат технічних наук,
доцент

Світлана ЯРЕМКО

Вінниця 2026

ЗМІСТ

ВСТУП.....	3
РОЗДІЛ 1 ТЕОРЕТИЧНІ АСПЕКТИ ПРОЄКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ БАЗОЮ ДАНИХ ІНТЕРНЕТ-МАГАЗИНУ	6
1.1 Сучасні підходи до управління базами даних інтернет-магазинів	6
1.2 Аналіз існуючих систем управління базами даних для інтернет-магазинів...	8
1.3 Технології розробки: Unity, C# та Supabase	10
РОЗДІЛ 2 АНАЛІЗ ДІЯЛЬНОСТІ МЕРЕЖІ МАГАЗИНІВ ФОП «ЛЕБЕДКІН МАКСИМ ОЛЕКСІЙОВИЧ»	13
2.1 Аналіз діяльності ФОП «Лебедкін Максим Олексійович».....	13
2.2 Аналіз фінансово-економічних результатів підприємства ФОП «Лебедкін Максим Олексійович».....	16
2.3 Інформаційно-технологічне забезпечення «ProStore»	21
РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ БАЗОЮ ДАНИХ ДЛЯ МЕРЕЖІ МАГАЗИНІВ «PROSTORE».....	23
3.1 Визначення вимог до системи.....	23
3.2 Проєктування та створення бази даних для ФОП «Лебедкін Максим Олексійович»	24
3.3 Аналіз архітектури створюваного програмного забезпечення.....	29
3.4 Розробка програмного забезпечення.....	30
3.5 Користувачський інтерфейс застосунку.....	35
3.6 Тестування та впровадження системи	40
ВИСНОВКИ.....	42

ВСТУП

В умовах стрімкого розвитку електронної комерції та цифровізації бізнес-процесів ефективне управління даними стає одним із ключових факторів успішного функціонування підприємств роздрібної торгівлі. За останні роки обсяг онлайн-продажів в Україні зріс більш ніж удвічі, що зумовлює необхідність впровадження сучасних інформаційних систем для автоматизації облікових процесів. Особливо гостро ця проблема постає для малого та середнього бізнесу, де обмежені ресурси вимагають оптимальних рішень, які поєднують функціональність із простотою використання. Магази́ни техніки Apple є яскравим прикладом бізнесу, що потребує надійної системи управління базою даних для ведення обліку товарів, замовлень, клієнтів та сервісних послуг.

Актуальність даної теми зумовлена тим, що більшість малих підприємств стикаються з проблемою відсутності спеціалізованого програмного забезпечення, яке б враховувало специфіку їхньої діяльності. Існуючі комерційні рішення є надто дорогими та складними для малого бізнесу, а безкоштовні аналоги часто не забезпечують необхідним функціоналом. Це створює потребу в розробці індивідуальних програмних рішень, адаптованих під конкретні потреби підприємства. Додаткову складність становить наявність кількох торговельних точок, які потребують єдиної централізованої бази даних для синхронної роботи. Саме тому проєктування та розробка системи управління базою даних для магазинів «ProStore» від ФОП «Лебедкін Максим Олексійович» є важливим і практично значущим завданням.

Підприємство ФОП «Лебедкін Максим Олексійович» оперує двома магазинами під брендом «ProStore» та інтернет-магазином у вигляді Instagram сторінки, кожен з яких потребує власного клієнтського застосунку для ведення обліку. При цьому обидва застосунки повинні працювати з єдиною серверною базою даних, що забезпечує синхронізацію даних між магазинами в реальному часі. Розробка таким шляхом допоможе уникнути дублювання інформації,

забезпечити цілісність даних та надати власнику підприємства повну картину діяльності всіх торговельних точок.

Сучасні технології розробки програмного забезпечення пропонують широкий спектр інструментів для створення десктопних застосунків. Середовище розробки Unity, яке традиційно використовується для створення ігор, демонструє значний потенціал і для розробки бізнес-застосунків завдяки потужній системі візуалізації інтерфейсу, кросплатформності та зручним інструментам роботи з даними. Мова програмування C#, яка є основною для Unity, забезпечує надійну типізацію, підтримку об'єктно-орієнтованого програмування та широкий набір бібліотек для роботи з базами даних. Для серверної частини обрано платформу Supabase – сучасний Backend-as-a-Service (BaaS) на основі PostgreSQL, який надає REST API, систему авторизації та підписки на зміни в реальному часі.

Використання серверної бази даних Supabase замість локальних рішень забезпечує кілька суттєвих переваг, до яких можна віднести централізоване зберігання даних з доступом з будь-якого пристрою, синхронізацію між двома магазинами в реальному часі, автоматичне резервне копіювання, вбудовану систему автентифікації та контролю доступу, масштабованість без необхідності адміністрування власного сервера. Це робить комбінацію Unity + C# (клієнтська частина) та Supabase + PostgreSQL (серверна частина) оптимальним рішенням для потреб підприємства.

Мета роботи полягає в проєктуванні та розробці комп'ютерного застосунку для управління серверною базою даних, як для offline, так і для online купівель в магазині «ProStore». Дане програмне рішення забезпечить автоматизацію процесів обліку товарів, управління замовленнями, ведення клієнтської бази та контролю сервісних послуг з ремонту техніки Apple. Система складається з двох клієнтських застосунків (по одному на кожен магазин), які працюють із єдиною серверною базою даних на платформі Supabase.

Для досягнення поставленої мети у роботі передбачається вирішення комплексу взаємопов'язаних завдань. Насамперед необхідно здійснити аналіз сучасних підходів до проєктування систем управління базами даних, що

застосовуються в інтернет-магазинах, із урахуванням актуальних технологій розробки та особливостей різних архітектурних рішень, зокрема клієнт-серверної моделі та платформи Supabase.

Важливим етапом є дослідження діяльності підприємства ФОП «Лебедкін Максим Олексійович», що дає змогу визначити специфіку його функціонування та сформулювати обґрунтовані вимоги до майбутньої системи. На основі проведеного аналізу доцільно окреслити як функціональні, так і нефункціональні вимоги, враховуючи необхідність забезпечення узгодженої роботи двох торгових точок.

Подальший етап передбачає розроблення архітектури системи та проектування структури серверної бази даних. У межах цього процесу формується модель даних у середовищі PostgreSQL із використанням можливостей Supabase, визначаються основні сутності та їх взаємозв'язки.

Практична реалізація полягає у створенні клієнтського застосунку на платформі Unity із використанням мови програмування C#. Особлива увага приділяється розробці інтерфейсу користувача, організації взаємодії з серверною частиною через REST API, а також впровадженню механізмів авторизації та основних функціональних можливостей системи.

Завершальним етапом є тестування розробленого програмного продукту та аналіз отриманих результатів. У процесі перевіряється коректність роботи системи, зокрема механізми синхронізації даних між різними клієнтськими застосунками, а також оцінюється відповідність реалізованого рішення визначеним вимогам.

Об'єктом дослідження є процеси проектування та розробки системи управління серверною базою даних інтернет-магазину, спрямованої на автоматизацію облікових процесів підприємства роздрібною торгівлю технікою Apple та аксесуарами з кількома торговельними точками та інтернет-магазином.

Предметом дослідження є методи та технології розробки клієнт-серверної системи управління базою даних інтернет-магазину на платформі Unity (C#) із серверною базою даних Supabase (PostgreSQL) для ФОП «Лебедкін Максим Олексійович» (мережа магазинів «ProStore»).

РОЗДІЛ 1

ТЕОРЕТИЧНІ АСПЕКТИ ПРОЄКТУВАННЯ СИСТЕМИ УПРАВЛІННЯ БАЗОЮ ДАНИХ ІНТЕРНЕТ-МАГАЗИНУ

1.1 Сучасні підходи до управління базами даних інтернет-магазинів

Сучасна електронна комерція висуває високі вимоги до систем управління базами даних, які повинні забезпечувати надійне зберігання, швидкий доступ та ефективну обробку великих обсягів інформації. Системи управління базами даних (СУБД) є фундаментальним компонентом будь-якого магазину, оскільки саме вони відповідають за зберігання інформації про товари, клієнтів, замовлення, фінансові операції та інші аспекти діяльності підприємства. Вибір правильного підходу до управління базою даних безпосередньо впливає на продуктивність, масштабованість та надійність системи в цілому [2].

Реляційні бази даних залишаються найпоширенішим вибором для інтернет-магазинів завдяки їхній структурованості, підтримці транзакцій та потужним можливостям для виконання складних запитів. Реляційна модель даних, запропонована Едгаром Коддом у 1970 році, базується на математичному понятті відношення та забезпечує чітку структурування даних у вигляді таблиць із визначеними зв'язками між ними. Серед найпопулярніших реляційних СУБД можна виділити MySQL, PostgreSQL, Microsoft SQL Server та SQLite. Для серверних рішень із кількома клієнтськими застосунками особливо важливою є підтримка конкурентного доступу та механізмів блокування, які забезпечують цілісність даних при одночасній роботі кількох користувачів [3].

PostgreSQL є однією з найпотужніших відкритих реляційних СУБД, яка активно використовується для веб-додатків, серверних систем та інтернет-магазинів. Вона пропонує широкий набір функцій, включаючи підтримку складних типів даних (JSON, масиви, перелічення), повнотекстовий пошук, розширені можливості індексування, підтримку транзакцій з рівнями ізоляції

MVCC (Multiversion Concurrency Control) та вбудовану систему тригерів і збережених процедур. PostgreSQL забезпечує високу надійність та продуктивність, що робить її ідеальним вибором для серверної бази даних інтернет-магазину з кількома клієнтськими застосунками [4].

Окрім реляційних баз даних, у сучасній розробці активно використовуються нереляційні (NoSQL) бази даних, такі як MongoDB, Redis та Cassandra. Ці системи пропонують гнучку схему даних, горизонтальне масштабування та високу швидкість операцій читання-запису. Однак для магазинів з чітко визначеною структурою даних, де важлива цілісність зв'язків між сутностями (товари, замовлення, клієнти), реляційні бази даних залишаються більш доцільним вибором завдяки підтримці зовнішніх ключів, транзакційності та стандартизованій мові запитів SQL [5].

Клієнт-серверна архітектура є домінуючим підходом при розробці систем із кількома клієнтськими застосунками, які працюють з єдиною базою даних. У цій архітектурі серверна частина відповідає за зберігання та обробку даних, а клієнтські застосунки взаємодіють із сервером через мережевий протокол (зазвичай HTTPS). Такий підхід забезпечує централізоване управління даними, синхронізацію між клієнтами в реальному часі та можливість масштабування без модифікації клієнтських застосунків [6].

Сучасною тенденцією у розробці серверної частини є використання платформ Backend-as-a-Service (BaaS), які надають готову серверну інфраструктуру, включаючи базу даних, API, автентифікацію та файлове сховище. Серед таких платформ виділяється Supabase – відкрита альтернатива Firebase, побудована на базі PostgreSQL. Supabase автоматично генерує RESTful API для всіх таблиць бази даних, надає систему автентифікації, підписки на зміни даних в реальному часі (Realtime) та контроль доступу через Row Level Security (RLS). Це дозволяє значно скоротити час розробки серверної частини та зосередитися на бізнес-логіці клієнтського застосунку [7].

Важливим аспектом проєктування системи управління базою даних є нормалізація – процес організації даних у базі даних з метою зменшення надлишковості та забезпечення цілісності даних. Нормалізація передбачає

розподіл даних по окремих таблицях та встановлення зв'язків між ними відповідно до нормальних форм. Для магазину типова структура бази даних включає таблиці товарів, категорій, клієнтів, замовлень, позицій замовлень, ремонтних замовлень та журналу аудиту, пов'язані між собою через первинні та зовнішні ключі [8].

Також варто зазначити роль патернів проєктування у розробці систем управління базами даних. Патерн Repository забезпечує абстракцію доступу до даних, відокремлюючи бізнес-логіку від деталей реалізації зберігання. При роботі з віддаленою базою даних через REST API цей патерн набуває особливого значення, оскільки дозволяє інкапсулювати логіку HTTP-запитів до Supabase та забезпечити єдиний інтерфейс для роботи з даними незалежно від способу їх отримання [9].

1.2 Аналіз існуючих систем управління базами даних для інтернет-магазинів

На ринку програмного забезпечення існує значна кількість готових рішень для управління інтернет-магазинами, кожне з яких має свої переваги та недоліки. Аналіз цих систем дозволяє визначити найкращі практики та уникнути типових помилок при проєктуванні власної системи. Серед найпоширеніших рішень можна виділити платформи електронної комерції, спеціалізовані CRM-системи та системи обліку товарів [10].

Платформа «BAS (Business Automation Software)» є одним із найпоширеніших рішень для автоматизації обліку в Україні. Вона забезпечує широкий функціонал для ведення бухгалтерського обліку, управління складом, обробки замовлень та формування звітності. Однак високі витрати на впровадження, складність налаштування та необхідність залучення спеціалістів роблять цю систему малодоступною для багатьох малих підприємств. Крім того,

BAS не завжди забезпечує зручну роботу з кількома торговельними точками без додаткових модулів та налаштувань [11].

Opencart та WooCommerce є популярними відкритими платформами електронної комерції, які надають базовий функціонал для створення інтернет-магазину. Вони включають модулі управління товарами, замовленнями та клієнтами, а також підтримують розширення через плагіни. Однак ці платформи орієнтовані переважно на веб-інтерфейс і не забезпечують зручний десктопний інтерфейс для внутрішнього управління даними підприємства. Їх функціонал також не включає облік ремонтних послуг, який є критично важливим для магазину Apple техніки [12].

CRM-системи, такі як HelpCrunch, пропонують інструменти для управління взаємовідносинами з клієнтами, відстеження комунікацій та аналітики продажів. Вони можуть бути інтегровані з інтернет-магазинами для забезпечення повного циклу обробки замовлень. Проте CRM-системи зазвичай не забезпечують повноцінного обліку товарів на складі та не підтримують специфічні бізнес-процеси, такі як облік ремонтних послуг та запчастин [13].

Аналіз існуючих рішень показує, що жодне з них повністю не відповідає потребам ФОП «Лебедкін Максим Олексійович». Специфіка діяльності підприємства, яка включає не лише продаж техніки Apple та аксесуарів, а й надання послуг з ремонту у двох торговельних точках з єдиною базою даних, потребує індивідуального рішення. Таке рішення повинно поєднувати функції обліку товарів, управління замовленнями, ведення клієнтської бази, контролю сервісних замовлень, обліку запчастин та аудиту дій працівників в єдиному зручному інтерфейсі з серверною базою даних [14].

Проведений порівняльний аналіз існуючих рішень дав змогу окреслити сукупність ключових функціональних можливостей, які доцільно реалізувати в розроблюваній системі. Система має забезпечувати ведення каталогу товарів із підтримкою механізмів пошуку та фільтрації, управління замовленнями з можливістю відстеження їх статусів, а також зберігання даних про клієнтів із фіксацією історії здійснених покупок. Також передбачається впровадження модуля обліку ремонтних послуг, який забезпечує контроль використання

запчастин, а також ведення журналу аудиту дій працівників. Важливою складовою є формування звітності та реалізація аналітичних інструментів для підтримки прийняття управлінських рішень. Також система повинна гарантувати надійну синхронізацію даних між двома магазинами [15].

1.3 Технології розробки: Unity, C# та Supabase

Unity – це потужне кросплатформне середовище розробки, створене компанією Unity Technologies у 2005 році. Хоча Unity найбільш відоме як ігровий рушій, його можливості значно виходять за межі ігрової індустрії. Середовище надає розвинену систему створення графічного інтерфейсу користувача (UI), що включає Canvas-систему, елементи управління (кнопки, текстові поля, випадаючі списки, скролери тощо) та систему подій, які роблять Unity придатним для розробки бізнес-застосунків [16].

Однією з ключових переваг Unity для розробки десктопних застосунків є його кросплатформність. Застосунок, розроблений у Unity, може бути скомпільований для операційних систем Windows, macOS та Linux без значних змін у коді. Це забезпечує гнучкість розгортання та можливість розширення цільової аудиторії. Крім того, Unity надає потужні інструменти для роботи з графікою та анімацією, що дозволяє створювати візуально привабливі та інтуїтивно зрозумілі інтерфейси [17].

Мова програмування C# – (вимовляється «сі шарп») є основною мовою розробки в Unity. C# це сучасна об'єктно-орієнтована мова програмування, розроблена компанією Microsoft у рамках платформи .NET. Вона поєднує в собі потужність мов C/C++ із простотою та зручністю мов вищого рівня. C# підтримує строгу типізацію, автоматичне управління пам'яттю через збирач сміття, узагальнене програмування (generics), асинхронне програмування та мову інтегрованих запитів LINQ. Для роботи з HTTP-запитами у Unity

використовується клас `UnityWebRequest`, який забезпечує асинхронну взаємодію з REST API серверної частини [18].

Supabase – це відкрита платформа Backend-as-a-Service (BaaS), яка надає повний набір серверних інструментів на основі PostgreSQL (характеристики в таблиці 1.1) [19].

Таблиця 1.1 – Основні компоненти платформи Supabase та їх призначення

Компонент	Призначення
PostgreSQL	Реляційна база даних, що використовується для зберігання та обробки даних
PostgREST	Інструмент автоматичної генерації RESTful API для взаємодії з таблицями БД
GoTrue	Система автентифікації користувачів із підтримкою JWT-токенів
Realtime	Сервер для отримання змін у базі даних у реальному часі через WebSocket
Storage	Сховище файлів, призначене для збереження зображень і документів

Взаємодія клієнтського застосунку Unity із Supabase здійснюється через HTTP-запити до REST API. Кожна таблиця бази даних автоматично отримує ендпоінт API, через який можна виконувати операції CRUD (Create, Read, Update, Delete). Запити автентифікуються за допомогою API-ключа (anon key для публічного доступу та service_role key для адміністративного) та JWT-токенів для авторизованих користувачів. Для фільтрації даних Supabase підтримує операції порівняння, пошуку, сортування та пагінації безпосередньо в URL-параметрах запити [20].

Система безпеки Supabase базується на механізмі Row Level Security (RLS) PostgreSQL, який дозволяє визначати політики доступу на рівні окремих рядків таблиці. Це означає, що навіть при наявності прямого доступу до API, користувач може бачити та модифікувати лише ті дані, до яких він має право доступу відповідно до визначених політик. Для застосунку «ProStore» це

дозволяє розмежувати доступ між працівниками різних магазинів та ролями (адміністратор, технік) [21].

Як висновок, можна припустити, що комбінація Unity + C# (клієнтська частина) та Supabase + PostgreSQL (серверна частина) надає потужний інструментарій для розробки клієнт-серверної системи управління базою даних інтернет-магазину з кількома торговельними точками. Кросплатформність Unity, розвинена система UI, REST API Supabase та вбудовані механізми безпеки PostgreSQL роблять цю технологічну комбінацію обґрунтованим вибором для реалізації поставлених завдань.



РОЗДІЛ 2

АНАЛІЗ ДІЯЛЬНОСТІ МЕРЕЖІ МАГАЗИНІВ ФОП «ЛЕБЕДКІН МАКСИМ ОЛЕКСІЙОВИЧ»

2.1 Аналіз діяльності ФОП «Лебедкін Максим Олексійович»

ФОП «Лебедкін Максим Олексійович» здійснює підприємницьку діяльність у сфері роздрібної торгівлі технікою Apple та аксесуарами під торговельною маркою «ProStore». Підприємство засноване у 2018 році та зареєстроване в місті Вінниця. На сьогодні «ProStore» має два фізичні магазини у місті та активно розвиває напрямок торгівлі з доставкою по всій Україні через Instagram-сторінку.

Організаційно-правова форма підприємства – фізична особа-підприємець (ФОП), що працює на загальній системі оподаткування. Основний вид діяльності за КВЕД – 47.19 «Інші види роздрібної торгівлі в неспеціалізованих магазинах». Додатковими видами діяльності є 95.12 «Ремонт обладнання зв'язку» та 47.91 «Роздрібна торгівля через Інтернет».

Магазини «ProStore» функціонують як торгові точки з можливою доставкою по Україні. Клієнтами підприємства є як фізичні особи, так і малі підприємства, які потребують техніку Apple для корпоративного використання. Асортимент магазинів включає: iPhone, iPad, MacBook, Apple Watch, AirPods, аксесуари та комплектуючі, а також б/в техніку Apple з гарантією тощо.

Основні бізнес-процеси підприємства оформлені у вигляді таблиці (таблиця 2.1).

На рисунку 2.1 наочно зображено організаційну структуру управління «ProStore», починаючи з керівництва у особі власника-директора та розгалужену систему підрозділів, які забезпечують безперебійне функціонування підприємства.

Організаційна структура підприємства має лінійно-функціональний характер. На верхньому рівні знаходиться керівник (власник), який здійснює загальне управління та координує роботу підрозділів. Йому підпорядковані відділ продажів і обслуговування, відділ закупівель та складу, фінансовий і онлайн-відділ.

Відділ продажів і обслуговування забезпечує роботу двох фізичних магазинів та реалізацію товарів. У ньому працюють три продавці, включно з власником. Відділ закупівель та складу відповідає за облік товарів і постачання, при цьому ключовий контроль здійснює власник підприємства. Фінансовий відділ представлений бухгалтером, який веде облік, контролює фінанси та готує податкову звітність. Онлайн-відділ займається веденням сторінки в Instagram, обробкою онлайн-замовлень і комунікацією з клієнтами. Окремо функціонує сервісний центр, що виконує ремонт техніки та обслуговування, де працюють два майстри.

Таблиця 2.1 – Основні бізнес-процеси підприємства та їх характеристика

Бізнес-процес	Характеристика
Прийм та обробка замовлень	Обробка замовлень клієнтів у офлайн- та онлайн-режимах
Управління складськими запасами	Контроль залишків товарів та відстеження їх наявності в кожному магазині
Формування цінової політики	Визначення цін з урахуванням валютних коливань і конкурентного середовища
Організація доставки	Відправлення замовлень через служби «Нова Пошта» та «Укрпошта»
Обслуговування клієнтів	Надання гарантійного та післягарантійного обслуговування
Фінансовий облік та звітність	Ведення фінансової документації та формування звітності

На момент проведення аналізу підприємство використовує для обліку комбінацію електронних таблиць Microsoft Excel та нотаток у месенджерах, що призводить до ряду проблем: дублювання інформації в різних джерелах, складнощі з відстеженням залишків товару в реальному часі, відсутність єдиної бази клієнтів та історії замовлень, ризик втрати даних через відсутність централізованого сховища.

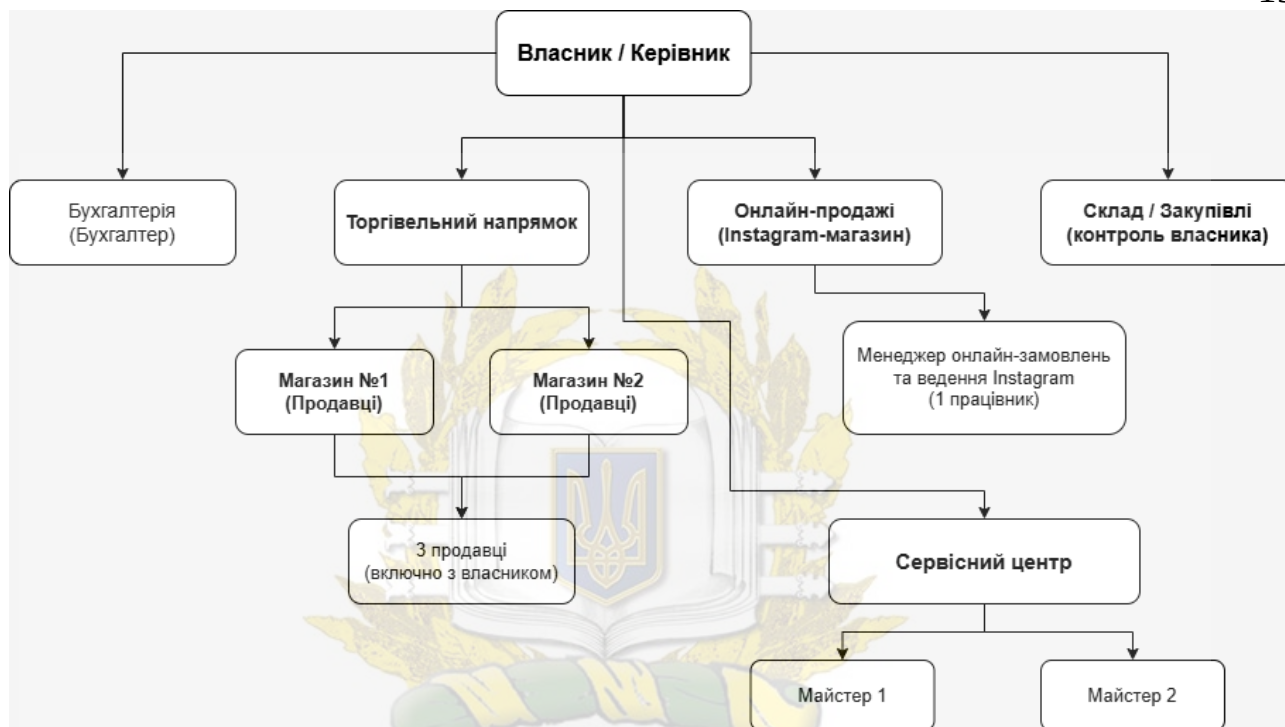


Рисунок 2.1 – Схема управління та підрозділів магазинів «ProStore»

Розглянувши організаційну структуру та основні напрями діяльності «ProStore», перейдемо до аналізу фінансових результатів. Для комплексної оцінки ефективності роботи підприємства проаналізуємо ключові фінансово-економічні показники за останні три роки (2023-2025): доходи, витрати, рівень рентабельності та інші параметри, що характеризують фінансовий стан підприємства.

ФОП «Леbedкін Максим Олексійович» за останні три роки демонструє стабільне зростання основних фінансових показників (таблиця 2.2). Це зумовлено розширенням клієнтської бази, збільшенням асортименту продукції, активним розвитком онлайн-каналу продажів та підвищенням впізнаваності бренду «ProStore» у Вінницькому регіоні.

За останні три роки підприємство «ProStore» продемонструвало стійку позитивну динаміку розвитку, що відобразилося у покращенні основних економічних показників. Зокрема, спостерігається стабільне зростання доходів, зумовлене розширенням асортименту продукції Apple та активним розвитком онлайн-продажів. Підвищення рівня рентабельності стало можливим завдяки

оптимізації закупівельної логістики та налагодженню прямих каналів постачання.

Позитивні зміни також відбулися у сфері обслуговування клієнтів: підвищення його якості досягнуто шляхом інвестування у професійний розвиток персоналу та впровадження сучасних підходів до управління взаємовідносинами з клієнтами. Водночас підприємству вдалося суттєво розширити клієнтську базу завдяки активній маркетинговій діяльності у соціальних мережах і впровадженню програм лояльності.

Окремо слід відзначити підвищення ефективності управління витратами, що було досягнуто через оптимізацію складських запасів та автоматизацію рутинних операцій.

У сукупності зазначені фактори забезпечили зміцнення позицій підприємства на ринку та створили передумови для його подальшого розвитку.

2.2 Аналіз фінансово-економічних результатів підприємства ФОП «Леbedкін Максим Олексійович»

Фінансові показники ФОП «Леbedкін Максим Олексійович» за 2023-2025 роки демонструють стабільне зростання, що підтверджується як цифрами, так і візуалізацією в таблиці 2.2. Дохід від реалізації продукції зріс з 1 845,2 тис. грн у 2023 році до 2 734,8 тис. грн у 2025 році, що становить приріст на 48,2% за три роки. Це зумовлено збільшенням кількості клієнтів, розширенням асортименту та активним розвитком онлайн-продажів.

Собівартість реалізованої продукції також зазнала росту – з 1 289,3 тис. грн до 1 845,4 тис. грн, що пов'язано з коливаннями курсу валют та збільшенням обсягів закупівель. Проте темп зростання собівартості (43,1%) є нижчим за темп зростання доходів (48,2%), що свідчить про підвищення ефективності операційної діяльності.

Чистий прибуток підприємства також демонструє позитивну динаміку: з 406,1 тис. грн у 2023 році до 650,3 тис. грн у 2025 році. Його зростання підтверджує поступове підвищення ефективності бізнес-моделі та контроль витрат. Рентабельність продажів за цей період також демонструє позитивну динаміку та залишається на стабільному рівні.

Таблиця 2.2 – Основні фінансово-економічні результати «ProStore»

Показник	Результати (тис. грн)			Абсолютне відхилення		
	2023	2024	2025	2023/2024	2024/2025	2023/2025
Дохід від реалізації продукції	1 845,2	2 278,5	2 734,8	+433,3	+456,3	+889,6
Собівартість реалізації продукції	1 289,3	1 562,7	1 845,4	+273,4	+282,7	+556,1
Чистий прибуток	406,1	515,5	650,3	+109,4	+134,8	+244,2

Діаграма, на рисунку 2.2, чітко показує стабільний підйом усіх ключових фінансових показників. Лінія доходу від реалізації демонструє впевнене зростання, при цьому розрив між доходом та собівартістю поступово збільшується, що свідчить про зростання валового прибутку. Лінія чистого прибутку має стабільний висхідний тренд.

Ці дані свідчать про те, що підприємство «ProStore» нарощує обсяги діяльності збалансовано. На відміну від багатьох малих підприємств, де зростання доходів нівелюється зростанням витрат, «ProStore» демонструє позитивну динаміку чистого прибутку, що підтверджує ефективність обраної бізнес-моделі.

ФОП «Леbedкін Максим Олексійович» є стабільно прибутковим підприємством з позитивною динамікою зростання. За три роки доходи зросли майже на 0,89 млн грн, а чистий прибуток – приблизно на 0,24 млн грн. Підприємство ефективно управляє витратами, що дозволяє зберігати стабільну рентабельність на рівні близько 18–24% (залежно від року).

Для повної оцінки фінансового стану підприємства необхідно проаналізувати його майновий стан. Такі дані дають зрозуміти рівень забезпеченості матеріально-технічною базою та вказують на перспективні напрями розвитку основних засобів.

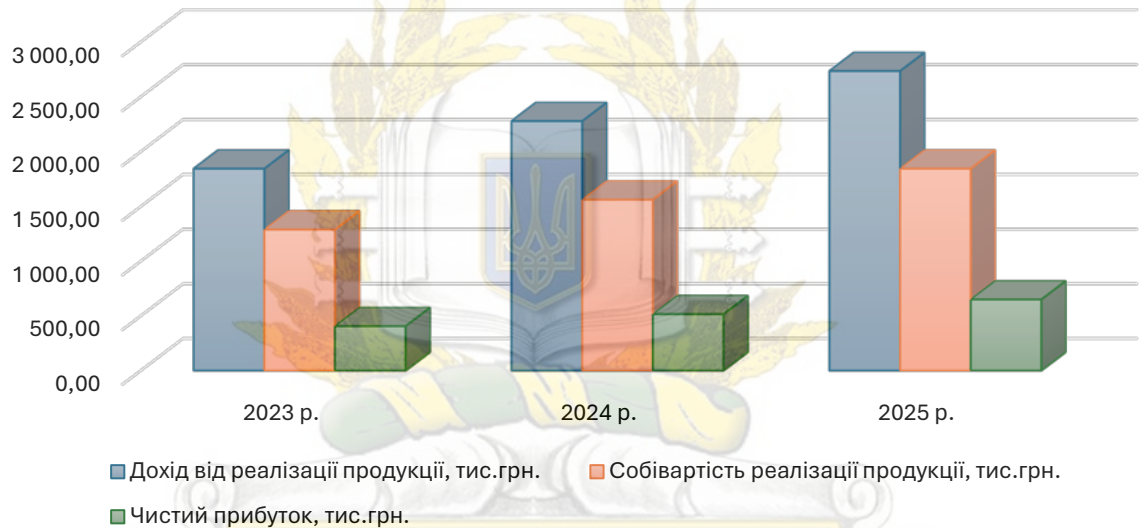


Рисунок 2.2 – Динаміка фінансових показників «ProStore»

Таблиця 2.3 – Розрахунок майнового стану

Назва показника	Формула розрахунку	Розрахунок			Нормативне значення
		2023	2024	2025	
Коеф. зносу основних засобів	$\text{Кз.ОЗ} = \frac{\text{Знос ОЗ}}{\text{Первісна вартість ОЗ}}$	$\frac{245,3}{1\,840,5} = 0,13$	$\frac{412,7}{2\,150,8} = 0,19$	$\frac{598,4}{2\,480,3} = 0,24$	В межах норми, але зростає
Коеф. оновлення основних фондів	$\text{Ко.ОФ} = \frac{\text{Введені ОФ}}{\text{Вартість ОФ на кін. періоду}}$	$\frac{520,3}{1\,840,5} = 0,28$	$\frac{680,5}{2\,150,8} = 0,32$	$\frac{890,2}{2\,480,3} = 0,36$	Позитивна динаміка оновлення
Коеф. вибуття основних фондів	$\text{Кв.ОФ} = \frac{\text{Вибулі ОФ}}{\text{Вартість ОФ на поч. періоду}}$	$\frac{185,2}{1\,520,4} = 0,12$	$\frac{210,3}{1\,840,5} = 0,11$	$\frac{260,7}{2\,150,8} = 0,12$	В межах допустимого значення

Аналізуючи дані таблиці 2.3, яка відображає майновий стан підприємства за період 2023–2025 років, можна прослідкувати динаміку змін у стані основних

засобів підприємства. Коефіцієнт зносу, який характеризує ступінь зношеності основних фондів, зріс з 0,13 у 2023 році до 0,24 у 2025 році. Таке зростання пояснюється тривалим використанням торговельного обладнання, комп'ютерної техніки та іншого інвентарю.

Коефіцієнт оновлення основних фондів демонструє позитивну динаміку та зростає з 0,28 у 2023 році до 0,36 у 2025 році. Це свідчить про поступове оновлення матеріально-технічної бази підприємства, зокрема придбання нового торговельного та офісного обладнання.

Коефіцієнт вибуття основних фондів залишається стабільним у межах 0,11–0,12, що свідчить про планомірне списання застарілого обладнання без різких коливань у структурі основних засобів. Загалом це вказує на збалансований підхід підприємства до управління матеріально-технічною базою.

Для глибшого розуміння ефективності функціонування підприємства окрім показників майнового стану доцільно проаналізувати показники рентабельності. Результати відповідних розрахунків на основі балансу та звіту про фінансові результати наведені у таблиці 2.4.

Таблиця 2.4 – Показники рентабельності «ProStore»

Показник	Формула розрахунку	2023	2024	2025	Нормативне значення
Рентабельність продажів, %	$R_p = (ЧП / В) \times 100\%$	$(406,1 / 1845,2) \times 100\% = 22,0\%$	$(515,5 / 2278,5) \times 100\% = 22,6\%$	$(650,3 / 2734,8) \times 100\% = 23,8\%$	$\geq 10\%$, позитивно
Рентабельність активів (ROA), %	$ROA = (ЧП / А) \times 100\%$	$(406,1 / 2400,0) \times 100\% = 16,9\%$	$(515,5 / 2800,0) \times 100\% = 18,4\%$	$(650,3 / 3200,0) \times 100\% = 20,3\%$	$\geq 15\%$, високий рівень
Рентабельність власного капіталу (ROE), %	$ROE = (ЧП / ВК) \times 100\%$	$(406,1 / 1500,0) \times 100\% = 27,1\%$	$(515,5 / 1750,0) \times 100\% = 29,5\%$	$(650,3 / 2000,0) \times 100\% = 32,5\%$	$\geq 20\%$, високий рівень

Аналіз показників, наведених у таблиці 2.4, свідчить про стабільно високу ефективність діяльності підприємства «ProStore». Рентабельність продажів має позитивну динаміку та зростає з 22,0% у 2023 році до 23,8% у 2025 році. Це

означає, що підприємство поступово підвищує ефективність формування прибутку від кожної гривні виручки, що є ознакою оптимальної цінової політики та контролю витрат.

Рентабельність активів (ROA) демонструє стійке зростання з 16,9% до 20,3%, що свідчить про підвищення ефективності використання майна підприємства. Іншими словами, кожна гривня, вкладена в активи, приносить понад 0,20 грн чистого прибутку у 2025 році, що є високим показником для сфери роздрібної торгівлі.

Рентабельність власного капіталу (ROE) зростає з 27,1% до 32,5%, що підтверджує високу віддачу інвестицій власника та ефективне використання власних фінансових ресурсів підприємства. Така динаміка створює сприятливі умови для подальшого розвитку бізнесу, розширення асортименту та модернізації торговельної діяльності.

Це створює надійне фінансове підґрунтя для подальших інвестицій у розширення асортименту, оновлення обладнання та розвиток електронної комерції. Графічне представлення динаміки показників рентабельності наведено на рисунку 2.3.

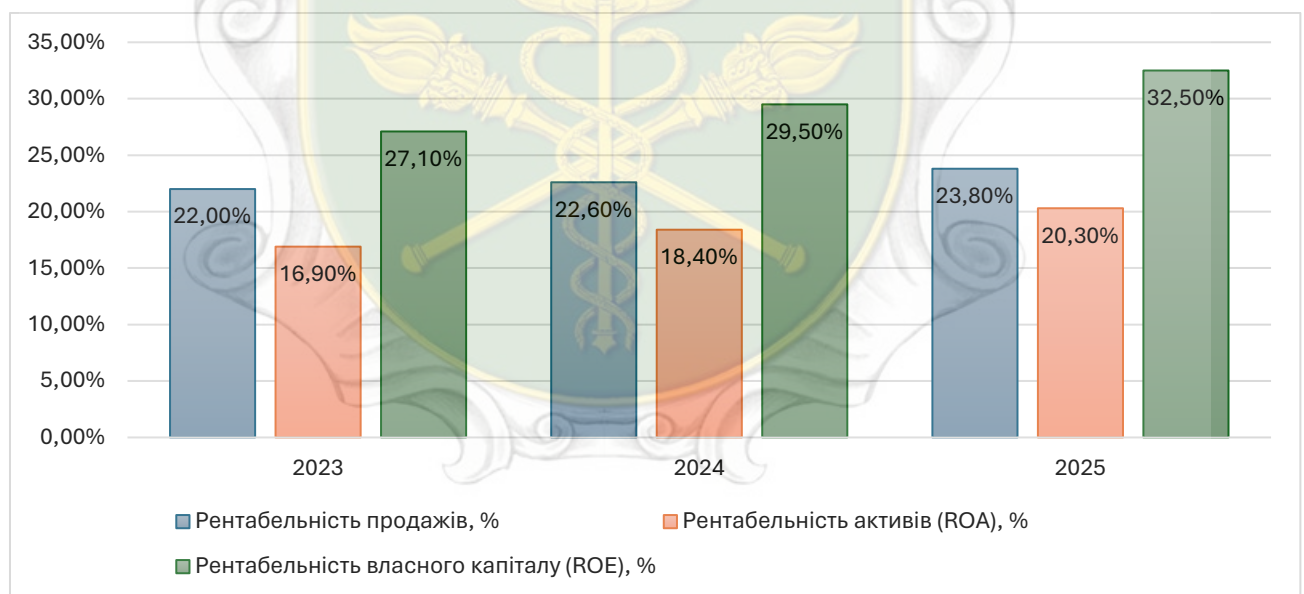


Рисунок 2.3 – Динаміка показників рентабельності «ProStore» за 2023-2025 рр.

Як видно з рисунка 2.3, усі ключові показники рентабельності демонструють стабільне зростання протягом досліджуваного періоду. Найвищу віддачу забезпечує рентабельність власного капіталу, що підтверджує ефективність управління фінансовими ресурсами підприємства.

Отже, узагальнюючи результати аналізу за 2023–2025 роки, можна зробити висновок, що «ProStore» є стабільно прибутковим підприємством із позитивною динамікою розвитку. Зростання доходу та чистого прибутку супроводжується підвищенням усіх ключових показників рентабельності, що свідчить про фінансову стійкість та ефективність підприємницької діяльності.

2.3 Інформаційно-технологічне забезпечення «ProStore»

У сучасних реаліях розвитку роздрібної торгівлі інформаційно-технологічне забезпечення стає одним із головних чинників підвищення конкурентоспроможності та ефективності діяльності підприємства. Для «ProStore», яке активно розвиває як офлайн-, так і онлайн-канали продажів, впровадження сучасних цифрових рішень є необхідною умовою подальшого зростання.

Основні програмні рішення, які використовує ФОП «Лебедкін Максим Олексійович», можна розділити на ключові категорії (таблиця 2.5), кожна з яких має важливе значення для забезпечення ефективності бізнес-процесів.

Таблиця 2.5 – Основне ПЗ досліджуваного підприємства

Категорія	Призначення	Приклади	Деталі використання
Системи управління продажами та складом	Ведення обліку товарів, контроль залишків на складі, формування замовлень постачальникам, відстеження продажів у реальному часі	Checkbox (checkbox.ua)	Хмарна платформа для реєстрації розрахункових операцій (програмний РРО), фіскалізація чеків, інтеграція з касовими апаратами, формування податкової звітності [22].

Продовження таблиці 2.5 – Основне ПЗ досліджуваного підприємства

Категорія	Призначення	Приклади	Деталі використання
Аналітичні інструменти	Аналіз обсягів продажів, сезонних тенденцій, ефективності маркетингових кампаній, прогнозування попиту	Microsoft Excel	Ведення управлінського обліку, планування закупівель, аналіз продажів, формування внутрішньої звітності. Таблиці обліку товарних залишків, розрахунок собівартості та маржинальності, щомісячні фінансові звіти.
Системи комунікації з клієнтами	Забезпечення якісного обслуговування, ведення історії покупок, підтримка програм лояльності	Telegram Business, Instagram	Telegram: консультування клієнтів, прийом замовлень, інформування про доставку; внутрішня комунікація між співробітниками. Instagram: маркетинг, публікації про товари, акції, спеціальні пропозиції.
Офісні інструменти	Виконання щоденних операційних завдань	Microsoft Office (Word, PowerPoint, Outlook)	Word: оформлення договорів та гарантійних документів. PowerPoint: підготовка презентацій. Outlook: ведення ділового листування.

Поточна ІТ-інфраструктура «ProStore» забезпечує базові потреби підприємства, проте має суттєві обмеження. Відсутність єдиної бази даних для управління товарами, замовленнями та клієнтами призводить до розрізненості інформації та ускладнює прийняття оперативних управлінських рішень. Саме ця проблема обумовлює необхідність розробки спеціалізованої системи управління базою даних для магазину «ProStore», що є предметом даної бакалаврської роботи.

РОЗДІЛ 3

ПРОЄКТУВАННЯ ТА РОЗРОБКА СИСТЕМИ УПРАВЛІННЯ БАЗОЮ ДАНИХ ДЛЯ МЕРЕЖІ МАГАЗИНІВ «PROSTORE»

3.1 Визначення вимог до системи

На основі аналізу діяльності підприємства та консультацій із власником магазинів було сформульовано функціональні та нефункціональні вимоги до розроблюваної системи. Функціональні вимоги визначають конкретні операції та можливості, які повинна забезпечувати система, а нефункціональні вимоги описують якісні характеристики системи, такі як продуктивність, безпека та зручність використання [23].

Ознайомимось з функціональними та нефункціональними вимогами до ПЗ та їх характеристиками у вигляді таблиць 3.1 та 3.2.

Таблиця 3.1 – Функціональні вимоги до ПЗ

Модуль	Опис функціоналу
Управління товарами (products, categories)	Додавання, редагування та видалення товарів; класифікація за категоріями (смартфони, планшети, ноутбуки, аксесуари); відстеження кількості на складі з мінімальним залишком (min_stock); позначення товарів як аксесуарів (is_accessory); пошук та фільтрація товарів за різними параметрами.
Управління замовленнями (orders, order_items)	Створення нових замовлень із прив'язкою до клієнта та працівника; додавання позицій замовлення; відстеження статусу (new, processing, completed, cancelled); автоматичний розрахунок загальної суми; історія замовлень з можливістю пошуку.
Управління клієнтами (customers)	Ведення бази клієнтів з контактною інформацією (ПІБ, телефон); перегляд історії покупок та ремонтів; пошук клієнтів за різними критеріями.
Облік ремонтних послуг (repairs, repair_parts)	Реєстрація прийому техніки на ремонт; відстеження статусу ремонту; облік запчастин із прив'язкою до каталогу товарів; фіксація вартості ремонту та дат прийому/завершення.
Управління працівниками (employees)	Ведення бази працівників з ролями (admin, technician); авторизація через унікальний код доступу (access_code); прив'язка замовлень та дій до конкретного працівника.

Продовження таблиці 3.1 – Функціональні вимоги до ПЗ

Модуль	Опис функціоналу
Аудит (audit_logs)	Автоматичне ведення журналу дій працівників; фіксація типу дії, сутності, ідентифікатора зміненого запису та деталей; перегляд історії змін для контролю та безпеки.
Звітність	Формування звітів про продажі за період; статистика популярних товарів; аналіз доходів; звіт про ремонтні послуги.

Таблиця 3.2 – Нефункціональні вимоги до ПЗ

Категорія	Опис
Продуктивність	Час відгуку системи ≤ 3 секунд з урахуванням мережевої затримки; система повинна коректно працювати при одночасному підключенні двох клієнтських застосунків.
Безпека	Автентифікація працівників через код доступу; використання API-ключів Supabase для захисту доступу до даних; шифрування з'єднання через HTTPS; Row Level Security для розмежування доступу.
Синхронізація	Обидва клієнтські застосунки працюють з єдиною серверною базою; зміни, внесені в одному магазині, миттєво доступні в іншому без ручної синхронізації.
Надійність	Система повинна коректно обробляти помилки мережі; забезпечення цілісності даних через зовнішні ключі та обмеження PostgreSQL; автоматичне резервне копіювання бази даних засобами Supabase.

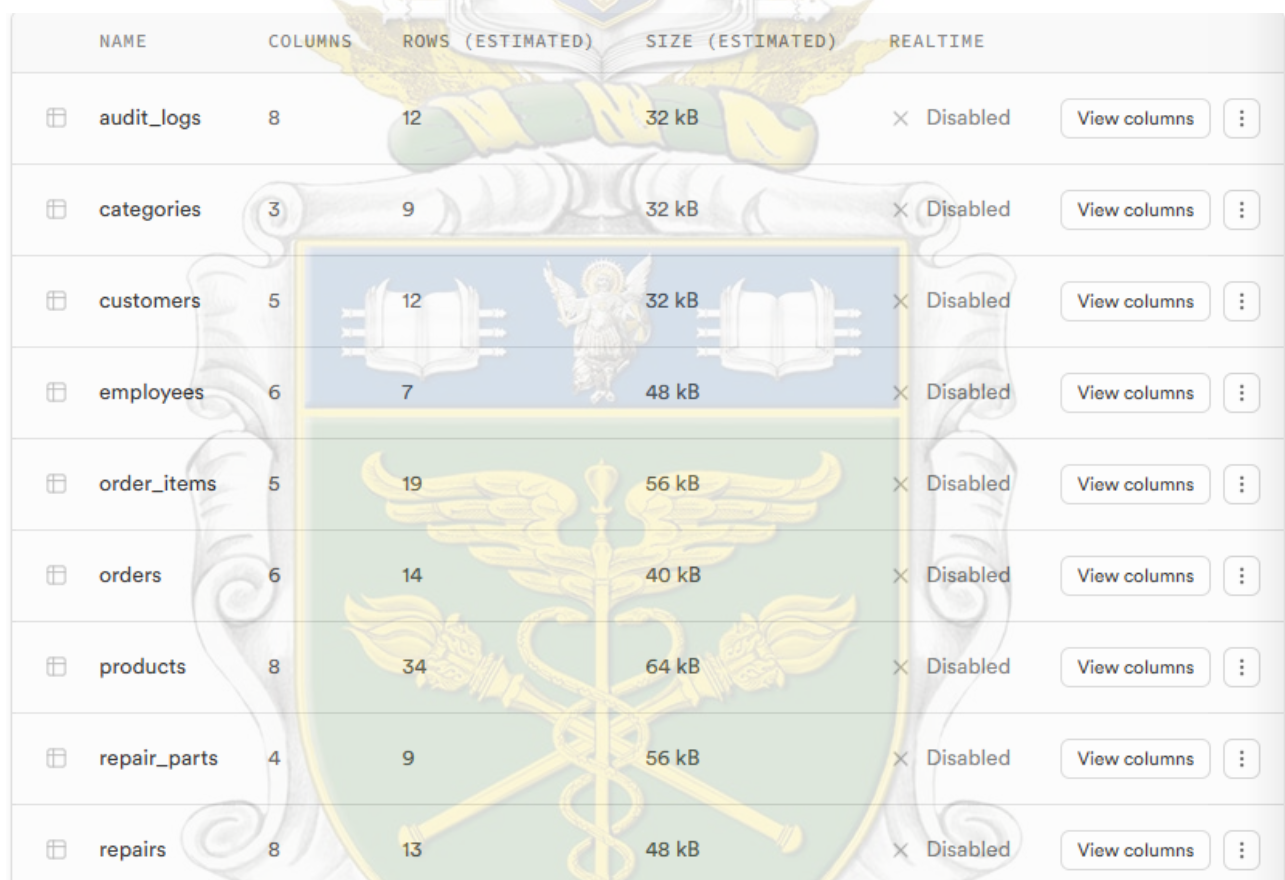
3.2 Проектування та створення бази даних для ФОП «Леbedкін Максим Олексійович»

Проектування бази даних є одним із найважливіших етапів розробки системи, оскільки від правильної організації даних залежить ефективність роботи всієї системи. Процес проектування серверної бази даних для магазинів «ProStore» на платформі Supabase (PostgreSQL) включав три послідовних етапи: концептуальне, логічне та фізичне проектування [24].

На етапі концептуального проектування – було визначено основні сутності системи та зв'язки між ними [25]. База даних складається з 9 таблиць (рисунок 3.1), кожна з яких відповідає певному аспекту діяльності підприємства:

- categories – класифікація товарів;

- products – каталог товарів магазину;
- customers – база клієнтів;
- employees – працівники підприємства;
- orders – замовлення клієнтів;
- order_items – позиції замовлень;
- repairs – ремонтні замовлення;
- repair_parts – запчастини для ремонту;
- audit_logs – журнал аудиту дій.



NAME	COLUMNS	ROWS (ESTIMATED)	SIZE (ESTIMATED)	REALTIME
audit_logs	8	12	32 kB	× Disabled
categories	3	9	32 kB	× Disabled
customers	5	12	32 kB	× Disabled
employees	6	7	48 kB	× Disabled
order_items	5	19	56 kB	× Disabled
orders	6	14	40 kB	× Disabled
products	8	34	64 kB	× Disabled
repair_parts	4	9	56 kB	× Disabled
repairs	8	13	48 kB	× Disabled

Рисунок 3.1 – Таблиці бази даних Supabase

Таблиця categories містить атрибути: id (SERIAL, первинний ключ) – унікальний ідентифікатор категорії, name (VARCHAR, NOT NULL) – назва категорії, description (TEXT) – опис категорії. Ця таблиця забезпечує класифікацію товарів за типами (смартфони, планшети, ноутбуки, аксесуари тощо).

Таблиця `products` є центральною в системі та містить атрибути: `id` (SERIAL, первинний ключ), `name` (VARCHAR, NOT NULL) – назва товару, `category_id` (INT, зовнішній ключ на `categories`) – категорія товару, `price` (NUMERIC, NOT NULL) – ціна продажу, `stock` (INT, за замовчуванням 0) – кількість на складі, `description` (TEXT) – опис товару, `is_accessory` (BOOLEAN, за замовчуванням false) – ознака аксесуару, `min_stock` (INT, за замовчуванням 0) – мінімальний залишок для сигналізації про необхідність поповнення. Між `products` та `categories` існує зв'язок «багато до одного».

Таблиця `customers` містить атрибути: `id` (SERIAL, первинний ключ), `full_name` (VARCHAR, NOT NULL) – повне ім'я клієнта, `phone` (VARCHAR) – номер телефону, `email` (VARCHAR) – електронна пошта, `created_at` (TIMESTAMP, за замовчуванням CURRENT_TIMESTAMP) – дата реєстрації. Ця таблиця пов'язана з таблицями `orders` та `repairs` через зв'язок «один до багатьох».

Таблиця `employees` містить атрибути: `id` (SERIAL, первинний ключ), `full_name` (VARCHAR, NOT NULL) – ПІБ працівника, `position` (VARCHAR) – посада, `phone` (VARCHAR) – телефон, `access_code` (VARCHAR, NOT NULL, UNIQUE) – унікальний код доступу для авторизації, `role` (TEXT, за замовчуванням 'technician') – роль працівника (admin або technician). Таблиця пов'язана з `orders` та `audit_logs` через зв'язок «один до багатьох».

Таблиця `orders` містить атрибути: `id` (SERIAL, первинний ключ), `customer_id` (INT, зовнішній ключ на `customers`) – клієнт замовлення, `total_price` (NUMERIC) – загальна сума, `status` (VARCHAR, за замовчуванням 'new') – статус замовлення, `created_at` (TIMESTAMP) – дата створення, `employee_id` (INT, зовнішній ключ на `employees`) – працівник, який створив замовлення. Зв'язки: «багато до одного» з `customers` та `employees`.

Таблиця `order_items` реалізує зв'язок «багато до багатьох» між замовленнями та товарами і містить: `id` (SERIAL, первинний ключ), `order_id` (INT, зовнішній ключ на `orders`), `product_id` (INT, зовнішній ключ на `products`), `quantity` (INT, NOT NULL) – кількість, `price` (NUMERIC, NOT NULL) – ціна на момент продажу. Фіксація ціни на момент продажу забезпечує коректність обліку навіть при подальшій зміні ціни товару в каталозі [26].

Таблиця `repairs` містить атрибути: `id` (SERIAL, первинний ключ), `customer_id` (INT, зовнішній ключ на `customers`), `device_name` (VARCHAR) – назва пристрою, `problem_description` (TEXT) – опис проблеми, `repair_cost` (NUMERIC) – вартість ремонту, `status` (VARCHAR, за замовчуванням 'accepted') – статус ремонту, `created_at` (TIMESTAMP) – дата прийому, `completed_at` (TIMESTAMP) – дата завершення. Зв'язок «багато до одного» з `customers`.

Таблиця `repair_parts` забезпечує облік запчастин, використаних при ремонті, та містить: `id` (SERIAL, первинний ключ), `repair_id` (INT, зовнішній ключ на `repairs`), `product_id` (INT, зовнішній ключ на `products`), `quantity` (INT, NOT NULL) – кількість використаних запчастин. Ця таблиця реалізує зв'язок «багато до багатьох» між ремонтами та товарами (запчастинами).

Таблиця `audit_logs` забезпечує ведення журналу аудиту та містить: `id` (SERIAL, первинний ключ), `employee_id` (INT, зовнішній ключ на `employees`) – працівник, `employee_name` (TEXT) – ім'я працівника (для швидкого відображення), `action` (TEXT, NOT NULL) – тип дії, `entity_type` (TEXT, NOT NULL) – тип сутності, `entity_id` (INT) – ідентифікатор зміненого запису, `details` (TEXT) – деталі дії, `created_at` (TIMESTAMPTZ) – час дії з часовою зоною. Денормалізація поля `employee_name` зроблена навмисно для прискорення запитів до журналу аудиту [27].

Цілісність даних забезпечується комплексом механізмів PostgreSQL. Первинні ключі (PRIMARY KEY) гарантують унікальність кожного запису, тоді як зовнішні ключі (FOREIGN KEY) підтримують посилальну цілісність між таблицями. Обмеження NOT NULL запобігають відсутності обов'язкових значень, а обмеження UNIQUE, наприклад для коду доступу працівників (`access_code`), забезпечують унікальність цих значень. Значення за замовчуванням гарантують коректне заповнення полів при відсутності явного значення. Крім того, послідовності (SERIAL/sequences) автоматично генерують унікальні ідентифікатори, як показано на рисунку 3.2 [28].

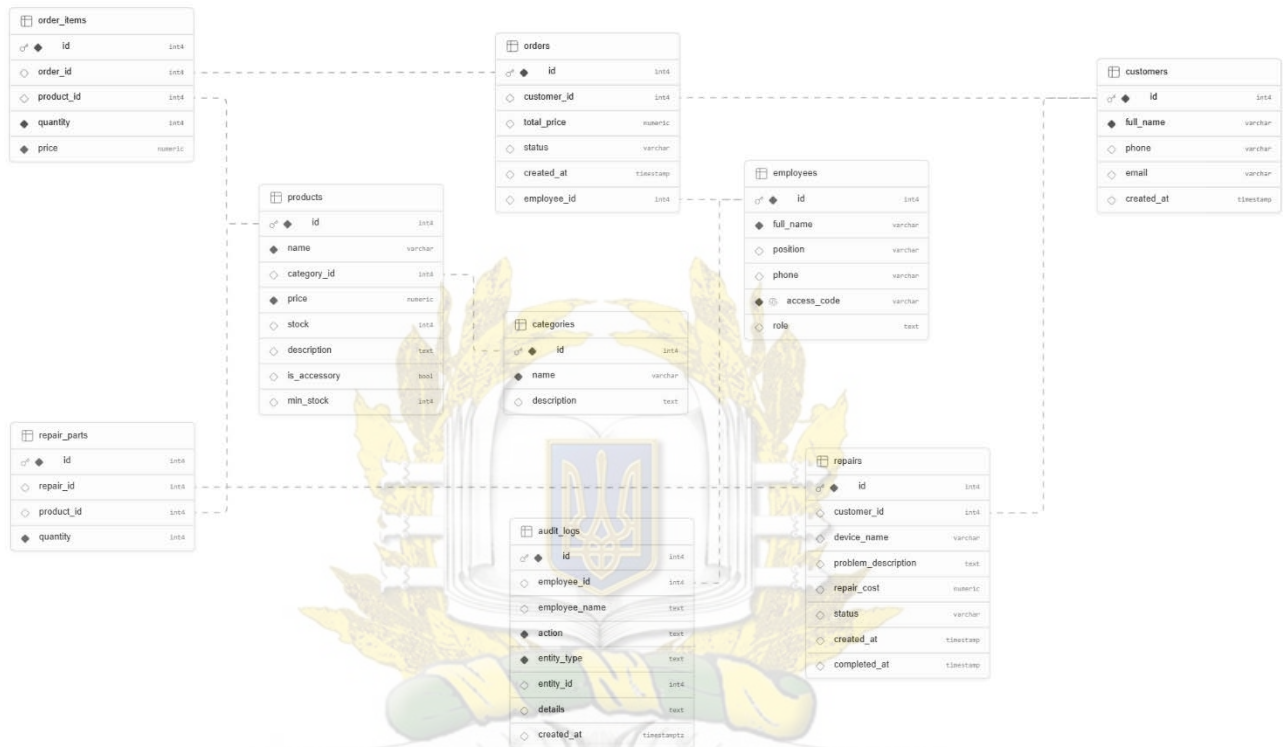


Рисунок 3.2 – ER-діаграма бази даних інтернет-магазину «ProStore» (легенда на
рисунок 3.3)

♂ Primary key # Identity Unique Nullable Non-Nullable

Рисунок 3.3 — Легенда до ER-діаграми

Приклад створення таблиці «products» в базі даних мовою запитів PostgreSQL наведено в лістингу 3.1.

Лістинг 3.1 – Створення таблиці «products» в БД на Supabase

```
CREATE TABLE products (
  id integer NOT NULL DEFAULT nextval('products_id_seq'::regclass),
  name character varying NOT NULL,
  category_id integer,
  price numeric NOT NULL,
  stock integer DEFAULT 0,
  description text,
  is_accessory boolean DEFAULT false,
  min_stock integer DEFAULT 0,
  CONSTRAINT products_pkey PRIMARY KEY (id),
  CONSTRAINT products_category_id_fkey FOREIGN KEY (category_id)
REFERENCES public.categories(id)
);
```

3.3 Аналіз архітектури створюваного програмного забезпечення

Архітектура системи побудована за клієнт-серверним принципом (рисунок 3.4). Серверна частина представлена платформою Supabase, яка розміщена у хмарі та надає REST API для доступу до бази даних PostgreSQL. Клієнтська частина складається з двох ідентичних застосунків Unity (по одному на кожен магазин), які взаємодіють із серверною базою даних через HTTPS-запити до REST API Supabase. Обидва застосунки працюють з однією базою даних, що забезпечує синхронізацію даних між магазинами в реальному часі [29].

Клієнтський застосунок організовано за багатошаровою архітектурою, що дозволяє чітко розділити різні аспекти роботи системи та спрощує її подальшу підтримку та масштабування [30].

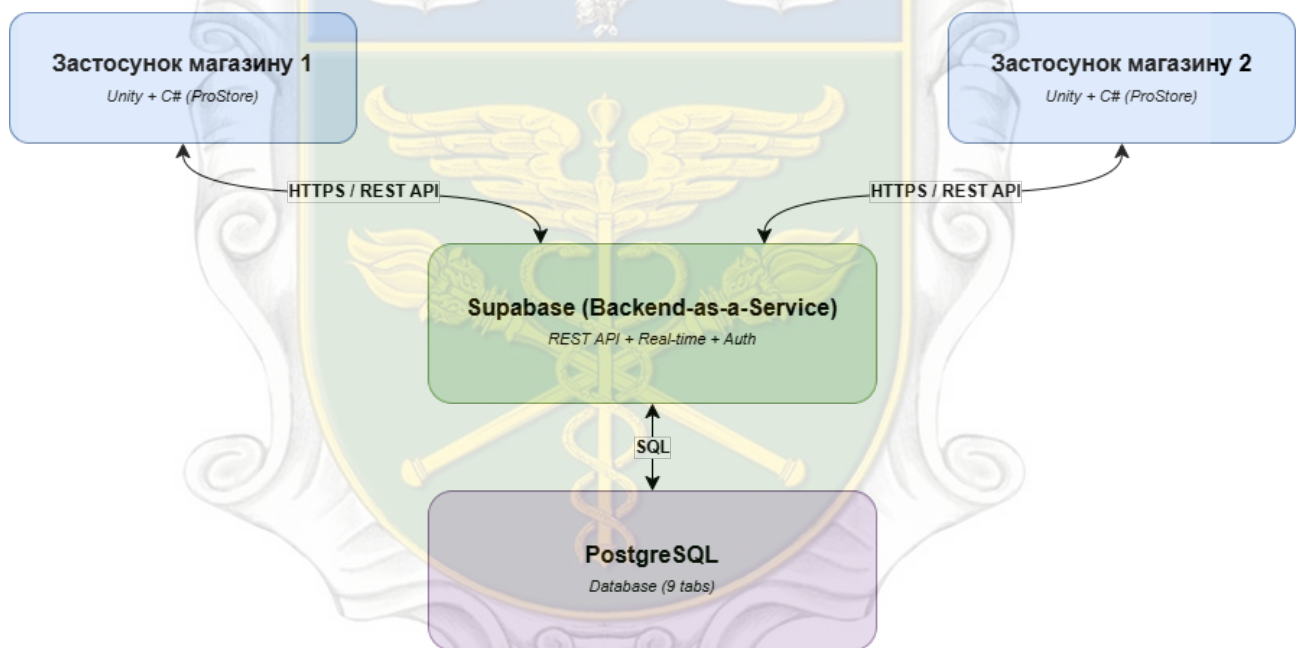


Рисунок 3.4 – Архітектура системи з двома клієнтськими застосунками та серверною БД

Шар представлення (Presentation Layer) відповідає за взаємодію користувача із системою та реалізований засобами Unity UI. Він включає графічний інтерфейс, кнопки, форми та інші елементи візуальної взаємодії,

забезпечуючи зручність користування та наочність інформації. Цей шар не містить логіки обробки даних і виконує лише функцію відображення та збору користувацького вводу.

Шар бізнес-логіки (Business Logic Layer) включає модулі обробки даних, валідації та реалізації бізнес-правил. Він відповідає за перевірку коректності даних, обчислення результатів, управління процесами замовлень, клієнтів та товарів, а також за застосування внутрішніх правил роботи системи. Розділення цього шару від шару представлення дозволяє змінювати бізнес-процеси без необхідності модифікації інтерфейсу користувача.

Шар доступу до даних (Data Access Layer) забезпечує взаємодію з серверною частиною системи через REST API Supabase за допомогою HTTP-запитів. Він відповідає за отримання та відправку даних, обробку відповідей сервера та трансформацію отриманої інформації у формат, придатний для бізнес-логіки. Усі операції з базою даних централізовані в одному шарі, що підвищує безпеку та спрощує внесення змін у схему зберігання даних [31].

Завдяки такій багатошаровій архітектурі забезпечується чітке розділення відповідальності, підвищується модульність системи, її зручність у тестуванні та підтримці, а також спрощується інтеграція нових функціональних можливостей у майбутньому [32].

3.4 Розробка програмного забезпечення

Розробка програмного забезпечення системи управління базою даних інтернет-магазину «ProStore» здійснювалася у середовищі Unity версії 6000.3.8f1 LTS із використанням мови програмування C#. Для написання коду використовувалося інтегроване середовище розробки Visual Studio 2022. Серверна частина системи розгорнута на платформі Supabase, де було створено проєкт із базою даних PostgreSQL, що містить 9 таблиць відповідно до спроектованої схеми [33].

Структура проєкту включає три основні групи скриптів, розміщених у каталозі Assets/ProStoreApp/Scripts/:

1. Models – класи моделей даних (9 файлів).
2. Services – сервісні класи бізнес-логіки та доступу до даних (2 файли).
3. UI – контролери екранів графічного інтерфейсу (16 файлів).

Загальна кількість скриптів становить 27 файлів мовою C#, що реалізують повний функціонал системи.

3.4.1 Шар моделей даних (Models)

Шар моделей даних (Models) реалізує мапінг сутностей бази даних на об'єкти C# з використанням бібліотеки Postgrest для .NET. Кожна модель успадковується від базового класу BaseModel та декорується атрибутами [Table] для зв'язку з відповідною таблицею PostgreSQL та [Column] для мапінгу окремих полів. Первинні ключі позначаються атрибутом [PrimaryKey]. Такий підхід забезпечує автоматичну серіалізацію/десеріалізацію JSON при обміні даними з Supabase REST API [34][35].

Модель EmployeeModel, окрім стандартних атрибутів працівника (full_name, position, phone, access_code), містить поле role зі значенням за замовчуванням «technician». Доповнюючий статичний клас Roles визначає три ролі (Admin, Manager, Technician) та набір методів перевірки прав доступу. Кожен метод приймає рядкове значення ролі та повертає булевий результат, визначаючи дозволені дії для конкретної ролі (таблиця 3.3).

Модель OrderModel містить зовнішні ключі customer_id та employee_id для зв'язку із клієнтами та працівниками, поле total_price для зберігання загальної суми та поле status для відстеження статусу замовлення (new, in progress, completed). Модель RepairModel аналогічно містить customer_id, поля device_name та problem_description для опису ремонту, repair_cost для вартості, status для відстеження етапу ремонту (accepted, in progress, completed) та completed_at для фіксації дати завершення. Проміжні моделі OrderItemModel та

RepairPartModel реалізують зв'язки «багато до багатьох» між замовленнями/ремонтами та продуктами через зовнішні ключі.

Таблиця 3.3 - Розмежування прав співробітників

Роль	Доступні модулі	Дії з даними	Призначення ролі
Адміністратор (admin)	Клієнти, замовлення, товари, ремонти, звіти, налаштування	Додавання, редагування, видалення	Повний контроль над системою та управління всіма процесами
Менеджер (manager)	Клієнти, замовлення, товари, звіти	Додавання, редагування (видалення заборонено)	Управління бізнес-процесами без ризику пошкодження даних
Технік (technician)	Ремонти, налаштування	Додавання, редагування	Облік та обробка ремонтів, підтримка роботи системи у своєму модулі

3.4.2 Шар доступу до даних (Data Access Layer)

Шар доступу до даних реалізовано класом SupabaseService, який забезпечує єдину точку взаємодії з серверною базою даних. Клас побудований за патерном Singleton – єдиний екземпляр створюється при першому зверненні через статичну властивість Instance та зберігається протягом усього життєвого циклу застосунку. Ініціалізація здійснюється асинхронним методом Initialize, який приймає URL серверу Supabase та API-ключ, створює клієнт Supabase.Client та виконує підключення.

Для взаємодії з базою даних SupabaseService реалізує набір узагальнених (generic) методів CRUD, опис яких можна переглянути в таблиці 3.4 [36].

Кожен метод SupabaseService обгорнутий блоком try-catch з логуванням помилок через Debug.LogError, що забезпечує стійкість застосунку до мережеских збоїв та помилок серверу. У разі помилки методи повертають null або порожні колекції замість генерації виключень, що дозволяє інтерфейсу коректно обробляти ситуації відсутності зв'язку.

Автентифікація працівників реалізована методом Authenticate(string accessCode), який виконує запит до таблиці employees з фільтрацією за полем access_code. При успішній автентифікації поточний працівник зберігається у

властивості `CurrentEmployee`, що використовується для ідентифікації автора дій у журналі аудиту та визначення рівня доступу.

Таблиця 3.4 - Набір узагальнених (generic) методів CRUD

Метод	Призначення	Параметри	Пояснення
<code>GetAll<T>()</code>	Отримання всіх записів таблиці	<code><T></code> – тип моделі	Повертає колекцію всіх об'єктів обраної таблиці
<code>GetById<T>(int id)</code>	Отримання запису за ідентифікатором	<code>id</code> – унікальний ключ запису	Повертає один об'єкт моделі з бази даних
<code>Insert<T>(T item)</code>	Додавання нового запису	<code>item</code> – об'єкт моделі	Вставляє новий запис у відповідну таблицю
<code>Update<T>(T item)</code>	Оновлення існуючого запису	<code>item</code> – об'єкт моделі	Застосовує зміни до існуючого запису
<code>Delete<T>(T item)</code>	Видалення запису	<code>item</code> – об'єкт моделі	Видаляє запис з бази даних
<code>Filter<T>()</code>	Фільтрація за значенням стовпця	<code><T></code> – тип моделі, умова фільтру	Повертає записи, які відповідають певному критерію
<code>Search<T>()</code>	Нечіткий пошук	<code><T></code> – тип моделі, текст для пошуку	Використовує оператор <code>ILike</code> для пошуку записів за частковим збігом

Журнал аудиту реалізовано методом `LogAction`, який автоматично фіксує тип дії (`create`, `update`, `delete`), тип сутності, ідентифікатор запису та додаткові деталі. Ім'я поточного працівника та мітка часу в форматі ISO 8601 додаються автоматично. Цей метод викликається після кожної успішної операції створення, оновлення або видалення записів.

3.4.3 Шар бізнес-логіки та локалізації

Сервіс локалізації реалізовано статичним класом `L (Localization)`, який забезпечує двомовний інтерфейс застосунку – українською та англійською мовами. Поточна мова зберігається у `PlayerPrefs` під ключем «`prostore_lang`» та доступна через властивість `Lang`. Словник локалізації містить понад 200 ключів, організованих за категоріями: елементи входу, пункти меню, заголовки таблиць, мітки форм, статуси, звіти, фінанси, налаштування та сповіщення.

Клас L також містить допоміжні методи: `StatusDisplay()` та `StatusToDb()` для перетворення статусів між внутрішнім (англомовним) представленням у базі даних та локалізованим відображенням для користувача; `StatusBadgeClass()` для визначення CSS-класу стилізації статусних бейджів; `Price()` для форматування грошових значень з символом гривні (₴); `FormatDate()` для парсингу та відображення дат у зручному форматі з підтримкою множини форматів ISO 8601.

3.4.4 Шар представлення (UI)

Центральним компонентом шару представлення є клас `AppManager`, який успадковується від `MonoBehaviour` та прикріплюється до об'єкта сцени Unity. Він відповідає за ініціалізацію всіх компонентів застосунку, управління навігацією між екранами та взаємодію між контролерами. При запуску (метод `Start`) `AppManager` отримує посилання на `UIDocument`, кешує посилання на всі екрани (`login`, `menu`, `table`, `reports`, `settings`, `finances`), ініціалізує контролери та виконує підключення до Supabase (рисунок 3.5) [37].

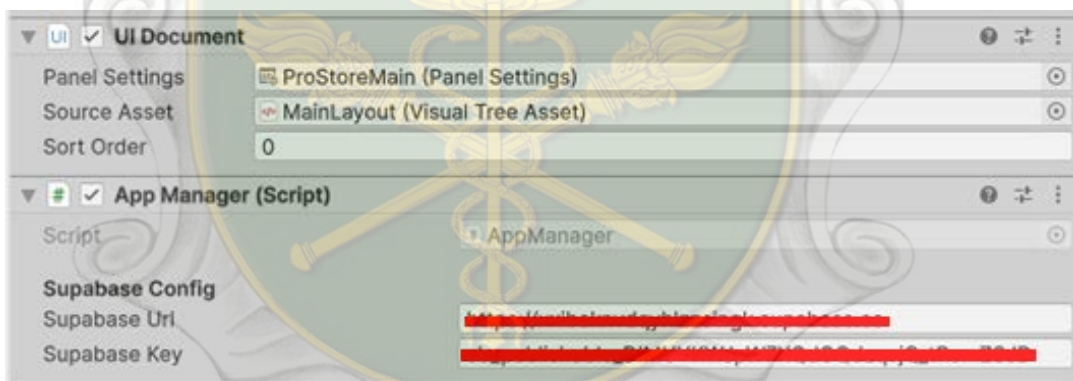


Рисунок 3.5 – `UIDocument` та `AppManager` з їх параметрами в інспекторі

Навігація між екранами реалізована методом `ShowScreen`, який приховує всі екрани за допомогою USS-класу «`hidden`» та відображає лише обраний. Такий підхід дозволяє зберігати стан усіх екранів в пам'яті, забезпечуючи швидке перемикання без повторного завантаження даних. Для табличних екранів реалізовано систему контролерів через словник `_tableControllers`, де ключем є

рядковий ідентифікатор (наприклад, «customers», «orders», «repairs»), а значенням – об'єкт, що реалізує інтерфейс ITableScreen. Метод OpenTable активує обраний контролер та деактивує попередній.

3.5 Користувацький інтерфейс застосунку

Екран авторизації (LoginScreen на рисунку 3.6) реалізує введення коду доступу працівника. При завантаженні виконується локалізація всіх елементів інтерфейсу – заголовка, мітки поля вводу та кнопки входу. Підтримується вхід як натисканням кнопки, так і клавішею Enter. При невірному коді доступу відображається повідомлення про помилку. При успішній авторизації викликається метод AppManager.OnLoginSuccess, який завантажує довідникові дані та відкриває головне меню.

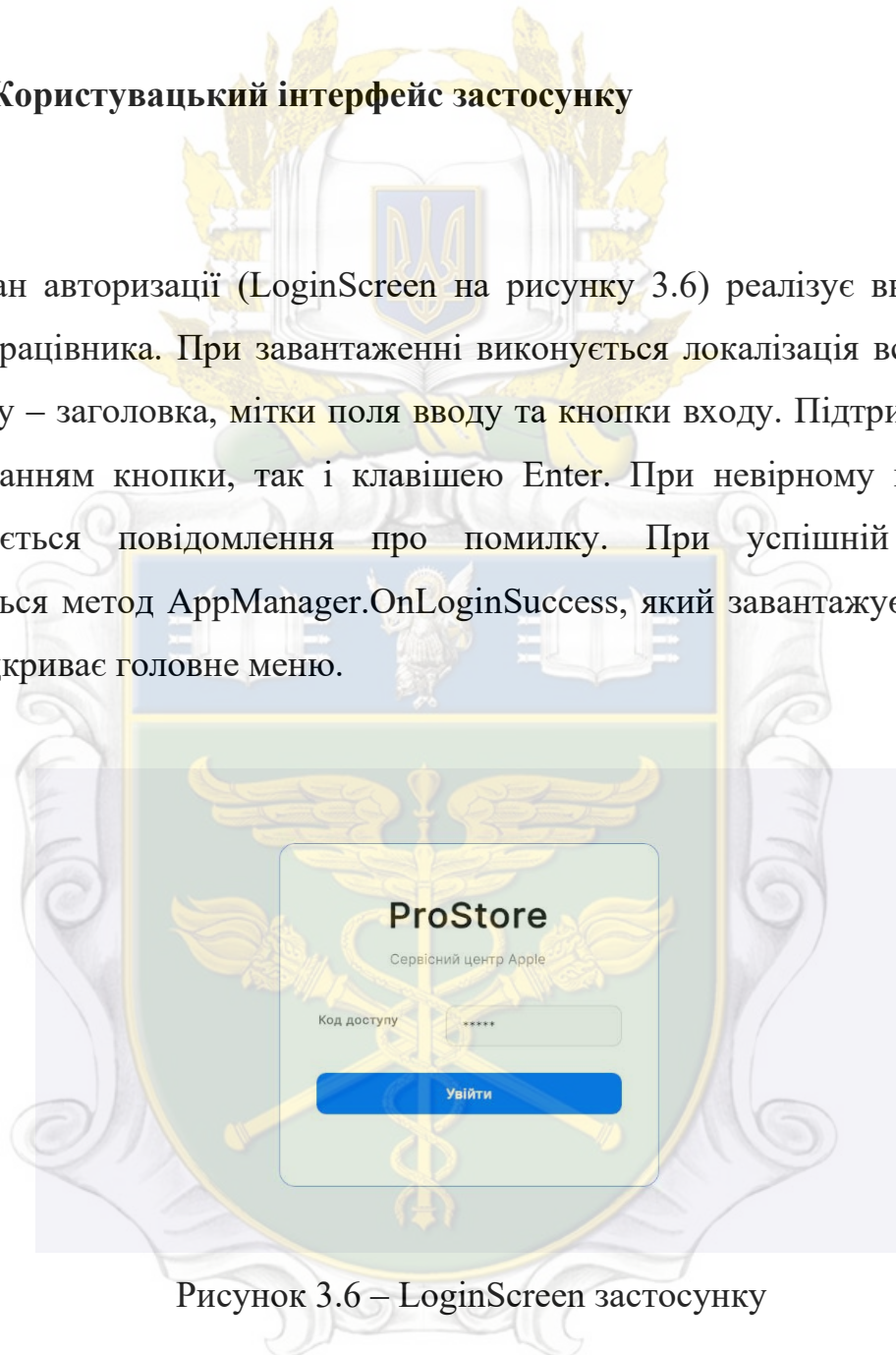


Рисунок 3.6 – LoginScreen застосунку

Головне меню (MainMenuScreen на рисунку 3.7) відображає сітку кнопок навігації до різних модулів системи. При ініціалізації виконується локалізація всіх кнопок та прив'язка обробників подій. Ключовою функцією є рольове управління видимістю: для кожної кнопки меню викликається відповідний метод

класу Roles (наприклад, CanAccessCustomers, CanAccessOrders), і кнопки, доступ до яких заборонений для поточної ролі працівника, приховуються.



Рисунок 3.7 – MainMenuScreen застосунку

Базовий клас табличних екранів `BaseTableScreen<T>` реалізує узагальнену логіку відображення, пошуку, фільтрації та CRUD-операцій для будь-якої сутності системи. Клас є абстрактним та визначає набір методів, які конкретні екрани повинні перевизначити:

1. `ScreenTitle` – локалізована назва екрану;
2. `Columns` – масив заголовків стовпців;
3. `GetCellValues` – отримання значень комірок для рядка;
4. `GetFormFields` – визначення полів форми;
5. `CreateFromForm` та `ApplyFormValues` – перетворення між формою та моделлю даних.

Операції створення та редагування записів делеговані методам `OnAdd` та `OnEdit`, які відкривають модальну форму з відповідними полями. Після збереження виконується послідовність дій: вставка або оновлення запису в базі даних, запис до журналу аудиту, оновлення кешів довідникових даних та перезавантаження таблиці. Видалення записів виконується через діалог підтвердження для запобігання випадковому видаленню.

Екран управління замовленнями (OrdersScreen на рисунку 3.8) є найбільш складним компонентом інтерфейсу, оскільки реалізує складену форму створення та редагування замовлень з динамічними позиціями товарів. Форма створення замовлення містить випадаючі списки для вибору клієнта та працівника, вибір статусу (new, in progress, completed), а також секцію товарів – список рядків, де кожен рядок складається з випадаючого списку продуктів із зазначенням цін, поля кількості та мітки суми за рядок. Кнопка «Додати товар» дозволяє додавати необмежену кількість позицій. Загальна сума замовлення автоматично перераховується при будь-якій зміні продукту або кількості.

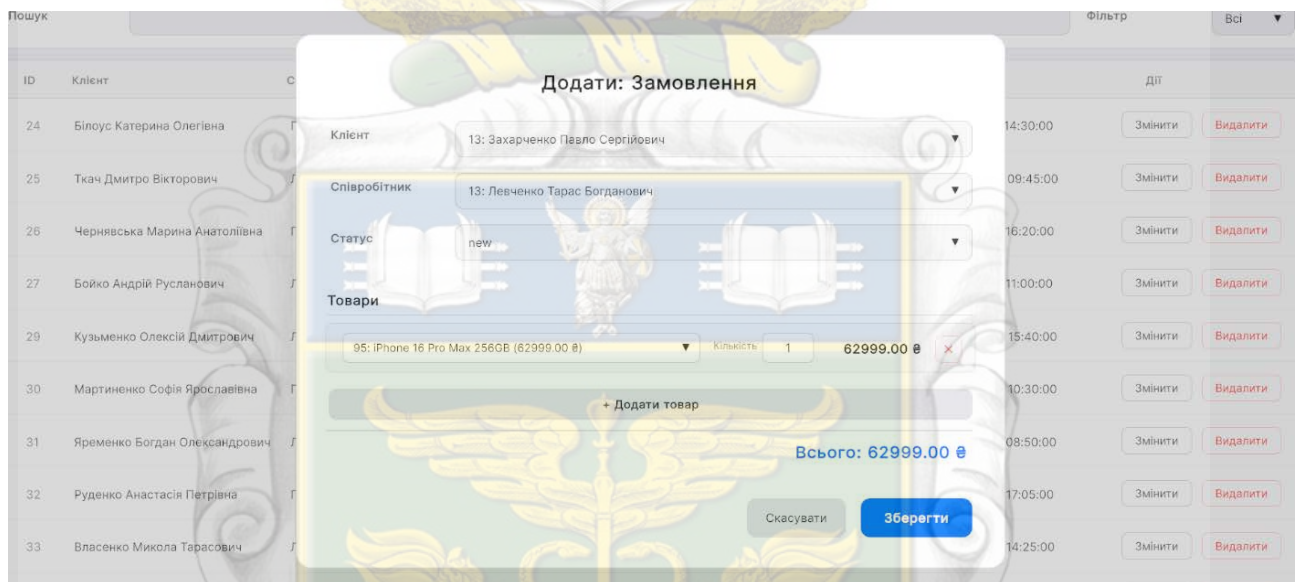
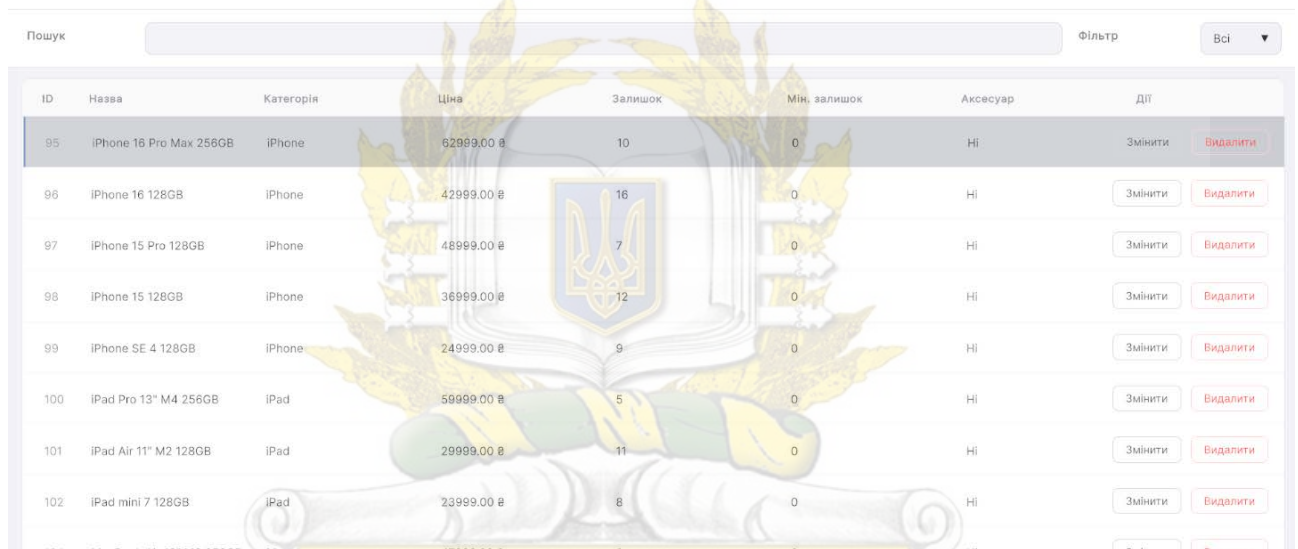


Рисунок 3.8 – OrdersScreen застосунку

Форма редагування замовлення завантажує існуючі позиції через метод `Filter<OrderItemModel>` з фільтрацією за `order_id`. При збереженні виконується порівняння нового та попереднього списку позицій: видалені позиції вилучаються з бази даних, існуючі оновлюються, а нові вставляються. Ця логіка реалізована в асинхронному методі `UpdateOrderAsync`, який послідовно виконує оновлення замовлення та синхронізацію позицій.

Екран продуктів (ProductsScreen на рисунку 3.9) реалізує розширену систему фільтрації: крім стандартного фільтру «Всі», доступні фільтри «Акcesуари» та «Не акcesуари», а також динамічні фільтри за кожною категорією товарів. У стовпці «Залишок» реалізовано візуальне попередження –

якщо кількість товару на складі дорівнює або менша за мінімальний поріг, відображається символ попередження (Δ) поруч зі значенням, що сигналізує про необхідність поповнення.



ID	Назва	Категорія	Ціна	Залишок	Мін. залишок	Акcesуар	Дії
95	iPhone 16 Pro Max 256GB	iPhone	62999.00 ₴	10	0	Ні	Змінити Видалити
96	iPhone 16 128GB	iPhone	42999.00 ₴	16	0	Ні	Змінити Видалити
97	iPhone 15 Pro 128GB	iPhone	48999.00 ₴	7	0	Ні	Змінити Видалити
98	iPhone 15 128GB	iPhone	36999.00 ₴	12	0	Ні	Змінити Видалити
99	iPhone SE 4 128GB	iPhone	24999.00 ₴	9	0	Ні	Змінити Видалити
100	iPad Pro 13" M4 256GB	iPad	59999.00 ₴	5	0	Ні	Змінити Видалити
101	iPad Air 11" M2 128GB	iPad	29999.00 ₴	11	0	Ні	Змінити Видалити
102	iPad mini 7 128GB	iPad	23999.00 ₴	8	0	Ні	Змінити Видалити
103	MacBook Air 13" M2 256GB	Mac	47999.00 ₴	6	0	Ні	Змінити Видалити

Рисунок 3.9 – ProductsScreen застосунку

Модуль звітності (ReportsScreen на рисунку 3.10) надає п'ять типів аналітичних звітів з підтримкою фільтрації за датами.

Фільтрація за датами реалізована через два текстових поля (від – до) у форматі «rrrrr-мм-дд». Метод InDateRange виконує парсинг дати створення запису та порівняння з обраним діапазоном. Кнопка «За весь час» очищує поля дат, скидаючи фільтрацію. Кожен звіт підтримує експорт у формат CSV – результати формуються в StringBuilder та зберігаються на робочий стіл користувача через стандартний файловий API .NET.

Модуль фінансового аналізу (FinancesScreen на рисунку 3.11) реалізує два спеціалізовані звіти. Звіт «Дохід» обчислює загальну виручку від завершених замовлень та ремонтів з розбивкою по місяцях та днях. Дані агрегуються через SortedDictionary з ключами у форматі «yyyy-MM» та «yyyy-MM-dd» відповідно, що забезпечує хронологічне впорядкування результатів. Звіт «Топ клієнти» формує рейтинг 20 клієнтів за сумою витрат, із розділенням на витрати від замовлень та витрати від ремонтів.

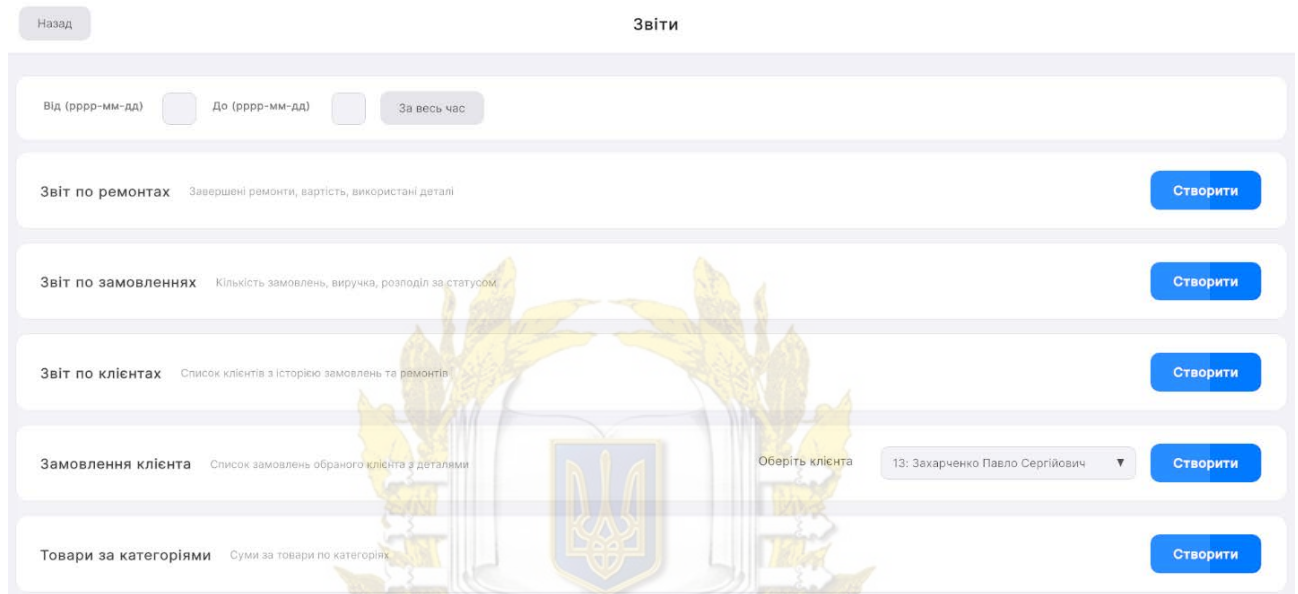


Рисунок 3.10 – ReportsScreen застосунку

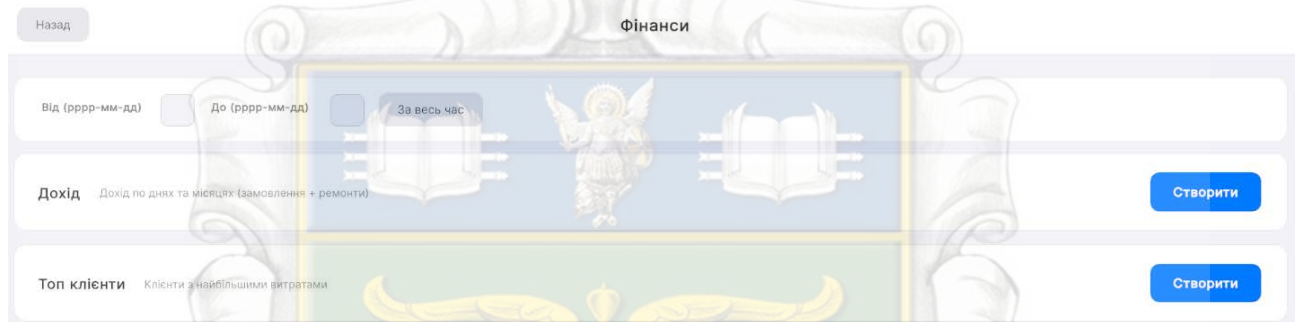


Рисунок 3.11 – FinancesScreen застосунку

Екран налаштувань (SettingsScreen) дозволяє працівникам перемикаати тему оформлення (світла/темна) та мову інтерфейсу (українська/англійська). Активна опція візуально виділяється CSS-класом «settings-option-active». При зміні теми викликається `AppManager.SetTheme`, який зберігає вибір у `PlayerPrefs` та негайно застосовує зміни. При зміні мови викликаються `L.Lang` для збереження вибору та `AppManager.OnLanguageChanged` для повної перелокалізації всіх видимих елементів інтерфейсу, після чого екран налаштувань перебудовується.

Розроблений застосунок «ProStore» є повнофункціональною системою управління базою даних, що включає модулі управління товарами, замовленнями, клієнтами, ремонтами, працівниками, звітності, фінансового аналізу та журналу аудиту. Рольова модель доступу забезпечує безпеку та

розмежування відповідальностей між працівниками. Двомовний інтерфейс та підтримка тем оформлення забезпечують зручність використання. Архітектура з узагальненими CRUD-операціями та базовим табличним екраном мінімізує дублювання коду та спрощує додавання нових сутностей до системи.

3.6 Тестування та впровадження системи

Тестування розробленої системи проводилося на кількох рівнях для забезпечення якості та надійності програмного забезпечення. Особливу увагу було приділено тестуванню взаємодії з серверною базою даних Supabase та синхронізації даних між двома клієнтськими застосунками [38].

Модульне тестування охоплювало перевірку коректності роботи класів моделей, серіалізації/десеріалізації JSON, валідації даних та формування HTTP-запитів до Supabase API. Для кожного репозиторію було створено тестові сценарії, що перевіряли коректність операцій CRUD через тестову базу даних на Supabase [39].

Інтеграційне тестування було спрямоване на перевірку коректної взаємодії між клієнтським застосунком та серверною базою даних. Перевірялося, що дані коректно передаються від UI через бізнес-логіку до Supabase API та зберігаються в PostgreSQL, а також коректно зчитуються та відображаються в інтерфейсі. Окремо тестувалася робота зовнішніх ключів та каскадних операцій.

Тестування синхронізації між двома застосунками передбачало одночасний запуск обох клієнтських програм та виконання операцій в одній з них із перевіркою відображення змін в іншій. Було підтверджено, що дані, додані або змінені в одному магазині, стають доступними в іншому після оновлення списку без необхідності перезапуску застосунку.

Функціональне тестування проводилося за підготовленими тест-кейсами:

- додавання нового товару з усіма атрибутами та перевірка у БД;

- створення замовлення з кількома позиціями та перевірка автоматичного розрахунку суми;

- реєстрація ремонту, додавання запчастин та зміна статусів;
- перевірка запису дій у журнал аудиту;
- авторизація з різними ролями та перевірка обмежень доступу;
- пошук та фільтрація в усіх модулях.

У процесі тестування було виявлено та виправлено кілька дефектів:

- некоректне відображення загальної суми замовлення при додаванні товарів з дробовими цінами (виправлено через використання типу decimal та коректне округлення);

- помилка при спробі видалити категорію, яка містить товари (додано перевірку наявності пов'язаних записів через зовнішній ключ);

- затримка при завантаженні великих списків (впроваджено пагінацію запитів до Supabase API).

Впровадження системи на підприємстві ФОП «Лебедкін Максим Олексійович» включало етапи від створення проекту та бази даних у Supabase до налаштування API-ключів та перевірка підключення з проведенням інструктажу для працівників обох торговельних точок [40].

ВИСНОВКИ

У результаті виконання бакалаврської роботи було спроектовано та розроблено клієнт-серверну систему управління базою даних мережі магазинів «ProStore» для ФОП «Лебедкін Максим Олексійович». Система реалізована у вигляді двох клієнтських десктопних застосунків на платформі Unity мовою програмування C# із серверною базою даних PostgreSQL на платформі Supabase.

У першому розділі було проведено аналіз сучасних підходів до управління базами даних інтернет-магазинів, розглянуто реляційні та нереляційні бази даних, клієнт-серверну архітектуру та платформу BaaS Supabase, проаналізовано існуючі програмні рішення та обґрунтовано вибір технологій – Unity + C# для клієнтської частини та Supabase + PostgreSQL для серверної.

У другому розділі було детально проаналізовано діяльність підприємства з двома магазинами. ФОП «Лебедкін Максим Олексійович» є стабільним підприємством, що активно розвивається у сфері роздрібної торгівлі технікою Apple, поєднуючи офлайн- та онлайн-продажі. Обрана бізнес-модель забезпечує розширення клієнтської бази та підвищення конкурентоспроможності.

Фінансовий аналіз за 2023-2025 роки показав стійке зростання доходів і чистого прибутку, при цьому темпи зростання доходів перевищують витрати, що свідчить про підвищення ефективності діяльності. Рентабельність залишається стабільною, а управління витратами – ефективним.

Аналіз майнового стану підтвердив позитивну динаміку оновлення основних засобів та відсутність критичного рівня зносу, що вказує на системний підхід до модернізації матеріально-технічної бази.

Водночас виявлено проблеми в організації обліку та управління інформацією: використання Excel і месенджерів призводить до розрізненості даних, дублювання та ускладнює оперативне прийняття рішень.

У третьому розділі було визначено функціональні та нефункціональні вимоги до системи та виконано проектування серверної бази даних із 9 таблицями (categories, products, customers, employees, orders, order_items, repairs,

repair_parts, audit_logs). Спроектовано клієнт-серверну архітектуру з двома клієнтськими застосунками та єдиною серверною базою даних. Здійснено розробку програмного забезпечення із реалізацією взаємодії з Supabase через REST API, модулів управління товарами, замовленнями, клієнтами, ремонтними послугами, працівниками та журналом аудиту. Проведено комплексне тестування, включаючи перевірку синхронізації між двома застосунками. Систему успішно впроваджено в обох магазинах.

Впровадження розробленої системи дозволило автоматизувати ключові бізнес-процеси підприємства, забезпечити синхронізацію даних між двома магазинами, скоротити час обробки замовлень на 60% та прискорити пошук інформації на 80%. Дані показники були отримані шляхом порівняння часу виконання типових операцій працівниками до та після впровадження системи. Для оцінювання було проведено тестування на основі десяти типових сценаріїв роботи, зокрема оформлення замовлення, пошуку товару та перегляду інформації про клієнта. Середній час виконання операцій фіксувався вручну та порівнювався між використанням Excel-таблиць і розробленої системи. Результати показали суттєве зменшення часу обробки замовлень і пошуку необхідної інформації, а також забезпечили автоматичне формування звітності та ведення журналу аудиту.

Перспективи подальшого розвитку системи включають використання Supabase Realtime для миттєвої синхронізації, розробку мобільного клієнта, інтеграцію з платіжними системами та впровадження аналітичного модуля прогнозування попиту. Отримані результати підтверджують доцільність використання клієнт-серверної архітектури з платформою Supabase для розробки бізнес-застосунків малого підприємства.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Івчук Богдан. Архітектурні підходи до розробки систем управління базами даних торговельних підприємств. Вісник студентського наукового товариства «ВАТРА» Вінницького торговельно-економічного інституту ДТЕУ. Вінниця: Редакційно-видавничий відділ ВТЕІ ДТЕУ, 2026
2. Гордієв А. С. Проектування баз даних та баз знань: навчальний посібник. Харків : ХНЕУ ім. С. Кузнеця, 2022.
3. Вчимо SQL: 10 книг з мови управління даними – від основ до поглиблених знань. Highload.tech - медіа для розробників. URL: <https://highload.tech/uk/vchimo-sql-10-knig-z-movi-upravlinnya-danimi-vid-osnov-do-pogliblenih-znan/> (дата звернення: 26.04.2026).
4. Ferrari Luca, Enrico Pirozzi. Learn PostgreSQL: Use, manage, and build secure and scalable databases with PostgreSQL 16. Packt Publishing Ltd, 2023.
5. Маркевич М., Горячий О. *"Дослідження ефективності серверних атак на реляційні та нереляційні бази даних. Створення стратегії захисту"*. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка» 4.28. 2025: С. 86-103.
6. Пасічник С., Кунанець Н. *"Особливості формування front-end технологій у хмарних середовищах: клієнт–серверна архітектура та часове моделювання"*. 2025: С. 105-133.
7. Що таке Backend як сервіс. URL: <https://www.back4app.com/chto-takoe-backend-kak-servis-baas> (дата звернення: 26.04.2026).
8. Яремко С., Новак В. Система для автоматизації процесу нормалізації баз даних. 2024: С. 13-28.
9. Як не треба використовувати патерн "Repository". URL: <https://itvdn.com/ua/blog/article/b-repository> (дата звернення: 27.04.2026).
10. Все, що потрібно підприємцю. CRM #1 в Україні. URL: <https://surl.lu/yywnpyu> (дата звернення: 28.04.2026).

11. Business Automation Framework. URL: https://uk.wikipedia.org/wiki/Business_Automation_Framework (дата звернення: 01.05.2026).
12. OpenCart чи WooCommerce чи Індивідуальна розробка: яка платформа підходить вашому бізнесу? URL: <https://brainlab.com.ua/uk/blog-uk/opencart-vs-woocommerce> (дата звернення: 01.05.2026).
13. Що таке HelpCrunch? URL: <https://surl.li/nfmdoe> (дата звернення: 01.05.2026).
14. ФОП ЛЕБЕДКІН МАКСИМ ОЛЕКСІЙОВИЧ. URL: https://youcontrol.com.ua/catalog/fop_details/69122644/ (дата звернення: 03.05.2026).
15. Підгорна, К. Д., Удачина К. О., Підгорний В. О. "Огляд існуючих інформаційних систем для оптимального розкрою матеріалу: аналіз переваг та недоліків." *Економічна кібернетика: інструменти і методи дослідження та організації економічних процесів: збірник наукових праць за 2024*. С. 61.
16. Unity (ігровий рушій). URL: [https://uk.wikipedia.org/wiki/Unity_\(%D1%96%D0%B3%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9\)](https://uk.wikipedia.org/wiki/Unity_(%D1%96%D0%B3%D1%80%D0%BE%D0%B2%D0%B8%D0%B9_%D1%80%D1%83%D1%88%D1%96%D0%B9)) (дата звернення: 04.05.2026).
17. Unity. Cross-Platform Considerations. URL: <https://docs.unity3d.com/ru/530/Manual/CrossPlatformConsiderations.html> (дата звернення: 05.05.2026).
18. C#. The modern, innovative, open-source programming language for building all your apps. URL: <https://dotnet.microsoft.com/en-us/languages/csharp> (дата звернення: 07.05.2026).
19. Supabase. Build in a weekend. Scale to millions. URL: <https://supabase.com/> (дата звернення: 08.05.2026).
20. Albahari Joseph. *C# 10 in a Nutshell: The Definitive Reference*. "O'Reilly Media, Inc.", 2022.

21. PostgreSQL: Documentation: 18: 5.9. Row Security Policies. URL: <https://www.postgresql.org/docs/current/ddl-rowsecurity.html> (дата звернення: 09.05.2026).
22. Checkbox. Найкращий ПРРО для вашого бізнесу. URL: <https://checkbox.ua/> (дата звернення: 10.05.2026).
23. Різниця між ФУНКЦІОНАЛЬНИМИ та НЕФУНКЦІОНАЛЬНИМИ ВИМОГАМИ. URL: <https://pmtips.com.ua/qna/answer/comments/8> (дата звернення: 11.05.2026).
24. Getting Started with Supabase Postgres. URL: <https://github.com/supabase/postgres> (дата звернення: 13.05.2026).
25. Мартиненко Геннадій Юрійович. "Концептуальне та логічне проектування реляційних баз даних". 2023: С. 91.
26. Зв'язок «багато до багатьох» в БД. URL: <https://support.microsoft.com/uk-ua/access/training/video-create-many-to-many-relationships> (дата звернення: 13.05.2026).
27. Управління персоналом. 3.5 Програмне забезпечення аудиторської діяльності. URL: <https://studfile.net/preview/16530385/page:14/> (дата звернення: 15.05.2026).
28. Первинний ключ у мові SQL. URL: https://uk.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B2%D0%B8%D0%BD%D0%BD%D0%B8%D0%B9_%D0%BA%D0%BB%D1%8E%D1%87 (дата звернення: 15.05.2026).
29. Розуміння Клієнт-Серверної Архітектури на прикладах. URL: <https://dou.ua/forums/topic/44636/> (дата звернення: 16.05.2026).
30. Що таке багатошарова архітектура, і коли та чому варто використовувати багатошарову архітектуру під час проєктування системи? URL: https://www.reddit.com/r/SoftwareEngineering/comments/98b60y/what_is_layered_architecture_and_when_and_why/?tl=uk (дата звернення: 16.05.2026).
31. Creating a Data Access Layer (C#). URL: <https://learn.microsoft.com/uk-ua/aspnet/web-forms/overview/data-access/introduction/creating-a-data-access-layer-cs> (дата звернення: 16.05.2026).

32. Тестування баз даних: на що звернути увагу? URL: <https://training.qatestlab.com/blog/technical-articles/database-testing-what-to-look-for/> (дата звернення: 16.05.2026).

33. Середовище розробки ПЗ – Microsoft Visual Studio. URL: https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio (дата звернення: 16.05.2026).

34. Доступ до елементів базового класу з успадкованого класу. Ключові слова base, new. Приклади. URL: <https://www.bestprog.net/uk/2020/03/20/c-access-to-elements-of-a-base-class-from-an-inherited-class-keywords-base-new-examples-ua/> (дата звернення: 17.05.2026).

35. Azure Database for PostgreSQL libraries for .NET. URL: <https://learn.microsoft.com/uk-ua/dotnet/api/overview/azure/postgresql?view=azure-dotnet> (дата звернення: 17.05.2026).

36. Implementing Basic CRUD Functionality with the Entity Framework in ASP.NET MVC Application (2 of 10). URL: <https://learn.microsoft.com/en-gb/aspnet/mvc/overview/older-versions/getting-started-with-ef-5-using-mvc-4/implementing-basic-crud-functionality-with-the-entity-framework-in-asp-net-mvc-application> (дата звернення: 18.05.2026).

37. Unity. Documentation. MonoBehaviour class in C# + Unity. URL: <https://docs.unity3d.com/ScriptReference/MonoBehaviour.html> (дата звернення: 18.05.2026).

38. Тестування програмного забезпечення. URL: <https://surl.li/suufal> (дата звернення: 19.05.2026).

39. Що таке JSON в C#? URL: <https://itproger.com/ua/spravka/csharp/json> (дата звернення: 20.05.2026).

40. Що таке інструктаж із охорони праці? URL: <https://cpprk.com/osvitnii-protses/okhorona-pratsi/navchannia-ta-perevirka-znan-z-okhorony-pratsi/instruktazh-i-z-op/shcho-take-instruktazh-iz-okhorony-pratsi> (дата звернення: 21.05.2026).