Міністерство освіти і науки України
Криворізький національний університет
Факультет інформаційних технологій
Кафедра автоматизації, комп'ютерних наук і технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеню вищої освіти – магістр
за освітньо-професійною програмою
«Комп'ютерні науки»

зі спеціальності
122 – Комп'ютерні науки

тема роботи:

«*Інформаційна система обліку товарів для магазину побутової хімії***»**

Виконав ст. гр. КН-23м	_____	Тімофєєв В. І.
Керівник	_____	Тиханський М. П.
Нормоконтроль	_____	Маринич І. А.
Завідувач кафедри	_____	Рубан С. А.

Кривий Ріг – 2024

КРИВОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет: *інформаційних технологій*

Кафедра: *автоматизації, комп'ютерних наук і технологій*

Ступінь вищої освіти: *Магістр*

Спеціальність: *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ

Зав. кафедри: *к.т.н. Рубан С.А.*

_____

« *5* » *липня* 2024 р.

ЗАВДАННЯ

на кваліфікаційну роботу магістра

студентові групи КН-23м *Тімофєєву Владиславу Ігоровичу*

1. Тема кваліфікаційної роботи: *«Інформаційна система обліку товарів магазину побутової хімії»*

затверджено наказом по університету № *594с* від *04.07.2024 р.*

2. Термін здачі кваліфікаційної роботи: *01.12.2024 р.*

3. Склад кваліфікаційної роботи: *Пояснювальна записка обсягом 95с., додатки, презентація у Microsoft PowerPoint (15 слайдів) в електронному та друкованому вигляді*

4. Консультанти кваліфікаційної роботи:

Розділ 1-3 доц. Тиханський М. П.

_____

Нормоконтроль доц. Маринич І. А.

5. Календарний план:

№	Етапи роботи	Термін виконання
1	*Вступ*	*10.07.24*
2	*Розділ 1*	*15.07.24*
3	*Розділ 2*	*18.08.24*
4	*Розділ 3*	*19.09.24*
5	*Висновки*	*15.10.24*
6	*Оформлення кваліфікаційної роботи*	*20.11.24*
7	*Підготовка презентації та графічного матеріалу*	*28.11.24*
8	*Підготовка доповіді до захисту*	*01.12.23*

6. Дата видачі завдання: *28.06.2024р.*

Керівник _____ /Тиханський М. П./

7. Запевнення: *Я, Тімофєєв Влідислав Ігорович, запевняю, що ця кваліфікаційна робота виконана самостійно, не містить академічного плагіату, фабрикації, фальсифікації. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.*

Із чинним Положенням про академічну доброчесність Криворізького національного університету ознайомлений.

Чітко усвідомлюю, що в разі виявлення у кваліфікаційній роботі умисних порушень робота не допускається до захисту або оцінюється незадовільно.

Здобувач _____ /Тімофєєв В. І./

АНОТАЦІЯ

Тімофєєв В. І. «Інформаційна система обліку товарів магазину побутової хімії».

Кваліфікаційна робота на здобуття ступеню вищої освіти магістр за освітньо-професійною програмою «Комп'ютерні науки» зі спеціальності 122 – Комп'ютерні науки. – Криворізький національний університет, Кривий Ріг, 2024..

Метою роботи є аналіз проблем автоматизації процесу обробки інформації, пов’язаної з обліком товарів, і розробка пропозицій щодо вдосконалення діяльності підприємств торгівлі.

В першому розділі був проведений аналіз предметної області, поставлена задача розробки, була досліджена предметна область, загальні принципи застосунків для ведення обліку та принцип їх роботи. Розглянуті існуючі програми для порівняння та встановлення функціональності власної розроблювальної системи.

В другому розділі описано особливості створюваної програми, опис вимог, проектування її функціональності та задач, які повинна виконувати. Також важливими пунктами є вибір мови програмування та середовища розробки, бази даних та огляд інтерфейсу додатку. Вибір цих всіх складових впливає на роботу проекту і кінцевий результат.

В третьому розділі було розроблено систему, яка повністю відповідає описаним вимогам та особливостям, виконує всі поставлені завдання та автоматизує роботу. Розроблена система має перспективу для розвитку і розширення свого функціоналу, що дозволить оптимізувати і полегшити вирішення ще великої кількості задач обробки інформації у веденні обліку руху товарів в галузі електронної комерції.

Ключові слова:

АЛГОРИТМ, БАЗА ДАНИХ, ЕЛЕКТРОННА КОМЕРЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА, СИСТЕМА ОБЛІКУ ТОВАРІВ, CRM-СИСТЕМА, USE CASE DIAGRAM

ANNOTATION

Timofieiev V. I. «Information system for inventory management of household chemicals store».

Graduation master`s work for obtaining an educational degree «Master» for the educational and professional program «Computer science» in specialty 122 – «Computer science». – Kryvyi Rih National University, Kryvyi Rih, 2024.

The purpose of this qualification work is to analyze the issues of automating the process of information management related to inventory accounting and to develop proposals for improving the operations of trading enterprises.

In the first chapter, an analysis of the subject area was conducted, the development task was defined, and the subject area was explored. General principles of applications for inventory management and their operation were studied. Existing software solutions were reviewed to compare and determine the functionality of the system being developed.

The second chapter describes the features of the developed application, including the requirements, the design of its functionality, and the tasks it should perform. Important aspects also include the selection of the programming language, development environment, database, and an overview of the application interface. The choice of these components significantly affects the performance of the project and the final result.

In the third chapter, a system was developed that fully complies with the described requirements and features, fulfills all assigned tasks, and automates operations. The developed system has potential for growth and expansion of its functionality, which will enable further optimization and simplification of solving numerous information processing tasks in managing the tracking of goods movement in the field of e-commerce.

Keywords:

ALGORITHM, DATABASE, E-COMMERCE, INFORMATION SYSTEM, INVENTORY MANAGEMENT SYSTEM, CRM SYSTEM, USE CASE DIAGRAM

ЗМІСТ

ВСТУП

Сфера електронної комерції сьогодні демонструє стрімкий розвиток, що створює необхідність постійного вдосконалення процесів обслуговування користувачів і управління товарами. Для збереження конкурентних позицій на ринку торгівлі компанії повинні шукати інноваційні рішення, які дозволяють підвищувати якість обслуговування та ефективність роботи.

Якість обслуговування є ключовим чинником успіху будь-якого магазину. Сучасний клієнт вимагає не лише високої якості товарів, але й відмінного сервісу, зокрема швидкої обробки замовлень, чіткого контролю за товарними залишками та зручності оформлення покупок. Відповідно, «обслуговування на вищому рівні» стало обов'язковою умовою для підприємств, які прагнуть задовольнити потреби покупців і збільшити прибутки.

Розвиток інформаційних технологій і програмного забезпечення відкриває нові можливості для організації бізнес-процесів. Завдяки впровадженню сучасних інструментів автоматизації підприємства можуть:

- Оптимізувати процес управління товарообігом;
- Забезпечити швидкість і точність операцій, таких як облік, контроль залишків і обробка замовлень;
- Скоротити витрати часу і ресурсів на виконання рутинних завдань.

Інтеграція додаткових пристроїв (сканери штрих-кодів, POS-термінали), електронних ресурсів (онлайн-каталоги, CRM-системи) і програмних засобів (ERP-системи, аналітичні платформи) дозволяє створити комплексні рішення для управління торговельними процесами.

Основними напрямками для вдосконалення електронної комерції є:

– *Автоматизація обліку товарів.* Системи автоматизації дозволяють швидко обробляти дані, вести точний облік залишків та уникати помилок, пов'язаних із людським фактором.

– *Покращення клієнтського досвіду.* Це включає швидку обробку замовлень, персоналізацію сервісу, доступ до історії покупок і програм лояльності.

– *Інтеграція з цифровими платформами.* Підключення до онлайн-магазинів, платіжних систем і служб доставки забезпечує клієнтам максимальну зручність.

– *Аналіз даних.* Використання інструментів аналітики дозволяє краще розуміти поведінку клієнтів, прогнозувати попит і оптимізувати асортимент.

Таким чином, розвиток інформаційних технологій у сфері електронної комерції дозволяє підприємствам не лише ефективніше працювати, але й створювати конкурентні переваги, що сприяють їхньому довгостроковому успіху на ринку.

Актуальність теми. З розвитком торгівлі сфера обслуговування перетворилася на одну з ключових галузей, яка потребує постійного вдосконалення для забезпечення якісного надання послуг. У сучасному торговельному бізнесі висока конкуренція стимулює підприємства впроваджувати новітні технології, зокрема автоматизовані системи управління товарообігом.

Впровадження автоматизованих систем обліку товарів є важливим кроком для сучасних підприємств торгівлі. Такі рішення дозволяють підвищити якість надання послуг, оптимізувати внутрішні процеси та забезпечити довгостроковий розвиток бізнесу навіть у висококонкурентному середовищі. Успішна інтеграція цих систем дає компаніям можливість не лише втриматися на ринку, а й активно розвиватися, збільшуючи свою частку в торгівельній галузі.

Мета та завдання дослідження. Метою даної кваліфікаційної роботи є аналіз проблем автоматизації процесу обробки інформації, пов’язаної з

обліком товарів, і розробка пропозицій щодо вдосконалення діяльності підприємств торгівлі.

Для досягнення поставленої мети були визначені такі задачі:

– Вивчення сучасних напрямів впровадження нових технологій у торговельному бізнесі.

– Дослідження організації обліку товарів і надання послуг споживачам у магазинах.

– Визначення сутності автоматизації та її ролі в діяльності підприємств електронної комерції.

– Огляд програмного забезпечення, доступного на ринку торгівлі.

– Аналіз процесу автоматизації введення та обробки інформації про товари.

– Виявлення існуючих проблем у сфері комерції.

– Критичний аналіз програмних продуктів, що використовуються у магазинах.

– Розробка пропозицій щодо вдосконалення процесів автоматизації для покращення роботи користувачів.

РОЗДІЛ І
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПРЕДСТАВЛЕНИХ РІШЕНЬ НА РИНКУ

1.1 Розвиток та призначення електронної комерції

Електронна комерція, у широкому сенсі, передбачає використання комп'ютерних мереж для підвищення продуктивності та ефективності організаційної діяльності. Завдяки електронній торгівлі компанії отримують змогу досягати таких переваг, як зростання рентабельності, розширення ринкової частки, покращення обслуговування клієнтів і прискорення процесу постачання продукції. Вона охоплює не лише оформлення замовлень через онлайн-каталоги, але й взаємодію з усіма зацікавленими сторонами, включно з інвесторами, потенційними співробітниками та іншими партнерами. Таким чином, електронна комерція ґрунтується на застосуванні інформаційних технологій для забезпечення ефективної взаємодії та обміну інформацією з різними учасниками діяльності організації.

Електронна комерція є потужною концепцією та процесом, який суттєво трансформував сучасне життя. Вона виступає одним із ключових драйверів інформаційно-комунікаційної революції в економіці. Популярність цього способу торгівлі швидко зростає завдяки його значному позитивному впливу на суспільство. Електронна комерція ліквідує багато обмежень, притаманних традиційному бізнесу. Наприклад, зовнішній вигляд і формат ведення бізнесу суттєво змінюються, що створює основу для нових підходів до ухвалення економічних рішень.

Віртуальні ринки та інтернет-магазини, які не потребують фізичного простору, забезпечують легкий доступ до глобальних ринків і можливість здійснювати операції з будь-якої точки світу. Покупці можуть обирати та замовляти товари, представлені у віртуальних вітринах магазинів різних країн,

розміщувати рекламу в інтернеті та здійснювати платежі через електронні сервіси. Усе це стало реальністю завдяки електронній комерції, яку справедливо можна вважати дивом сучасної епохи. [1,2].

У 1970-х роках термін "електронна комерція" спочатку означав обмін електронними даними, що використовувався для передачі бізнес-документів, таких як замовлення та голосування, у цифровій формі. З розвитком технологій це поняття еволюціонувало і почало стосуватися торгівлі товарами та послугами через Інтернет. Поява Всесвітньої павутини в 1994 році привернула увагу багатьох дослідників до електронного бізнесу. Проте значущим явищем в економіці "інтернет-бізнес" став лише після широкого впровадження протоколів HTTP, що тривало близько чотирьох років [2].

Перші електронні комерційні рішення з’явилися у 1998 році в США та деяких європейських країнах. Ці ранні веб-сайти, хоча й були недосконалими, слугували основою для розвитку цього виду бізнесу. До 2005 року електронна комерція стрімко поширилася в більшості міст Америки, Європи та Східної Азії. Дехто вважає, що поява електронної комерції збігається з появою Інтернету, однак на початкових етапах цей вид бізнесу був доступний лише великим корпораціям та фінансовим установам через високі витрати. Зі зростанням доступності Інтернету серед широких верств населення та оптимізацією структури електронної комерції цей спосіб ведення бізнесу став надзвичайно популярним.[2, 3].

Основи електронної комерції базуються на трьох ключових складових, які є невід’ємними для її успішного функціонування:

1. *Інфраструктура*. Цей рівень включає апаратне забезпечення, програмне забезпечення, бази даних та комунікаційну інфраструктуру. Він забезпечує технічну основу для електронної торгівлі, функціонуючи через Інтернет або інші телекомунікаційні мережі. Без розвиненої інфраструктури ефективне здійснення електронної комерції є неможливим.

2. *Послуги.* Другий рівень охоплює набір функцій, які сприяють пошуку, представленню та обробці інформації. Це можуть бути послуги з пошуку торгових партнерів, проведення переговорів, укладення угод та підтримки зв’язку між учасниками. Цей рівень створює функціональні можливості для взаємодії всередині системи електронної комерції.

3. *Продукти та конструкції.* Ця складова включає процес передбачення та поставки товарів, послуг і інформації, отриманих від клієнтів та партнерів. Вона охоплює співпрацю, обмін даними, організацію електронних ринків, ланцюг постачання, а також підтримку всього процесу торгівлі.

Ці три рівні є взаємопов’язаними та утворюють основу, яка забезпечує ефективне функціонування електронної торгівлі, сприяючи її розвитку та адаптації до потреб сучасного бізнесу.

Основні типи електронної комерції поділяються на кілька категорій, які характеризують різні моделі взаємодії між учасниками ринку:

– *Бізнес-бізнес (B2B).* Ця модель охоплює транзакції між компаніями. Вона включає продаж послуг, товарів або матеріалів, що використовуються підприємствами у їхній діяльності. Прикладом є інтернет-платформи оптової торгівлі, де компанії реалізують свої продукти іншим компаніям через веб-сайти. Така модель забезпечує ефективність у постачанні та зниження витрат.

– *Бізнес-споживач (B2C).* У цій моделі підприємства взаємодіють безпосередньо з кінцевими споживачами. Вона передбачає створення електронних вітрин для надання інформації, продажу товарів та послуг через Інтернет. B2C включає роздрібну торгівлю, онлайн-продажі та фінансові транзакції між бізнесом і споживачем. Типовим прикладом є інтернет-магазини.

– *Споживач-бізнес (C2B).* Цей тип передбачає, що споживачі надають послуги, товари або інформацію компаніям. Наприклад, у цій моделі користувачі можуть пропонувати створений ними контент, послуги або товари,

які купують компанії. Прикладом є фріланс-платформи, де клієнти пропонують свої навички чи продукти бізнесу.

– *Споживач-споживач (C2C).* Ця модель стосується транзакцій між кінцевими споживачами, де обидві сторони взаємодіють безпосередньо. Типовими прикладами є онлайн-аукціони або маркетплейси, де користувачі продають і купують товари чи послуги один у одного. C2C забезпечує зручну платформу для прямих угод.

Ці типи електронної комерції охоплюють більшість бізнес-моделей, що використовуються у сучасному цифровому середовищі, та дозволяють задовольняти різні потреби учасників ринку.

1.2 Мобільна комерція – переваги та недоліки

Термін "Мобільна комерція" (M-Commerce) вперше з’явився у 1997 році для позначення процесу "купівлі та продажу продуктів, інформації та послуг" за допомогою бездротових портативних пристроїв, таких як мобільні телефони, ноутбуки та персональні цифрові помічники (PDA) [3,4]. Ці пристрої дозволяють користувачам взаємодіяти з комп’ютерними мережами, надаючи можливість здійснювати покупки онлайн з будь-якого місця, без потреби у стаціонарному підключенні до Інтернету.

Мобільна комерція відкриває доступ до Інтернету у будь-який час і в будь-якому місці, забезпечуючи зручність для споживачів. До операцій, які включає M-Commerce, належать:

- купівля та продаж товарів і послуг;
- онлайн-банкінг;
- оплата рахунків;
- доставка інформації;
- інші дії, що спрощують фінансові та інформаційні транзакції.

M-Commerce стала важливою частиною сучасної електронної комерції, враховуючи значне поширення мобільних пристроїв і розвиток технологій бездротового зв'язку.

Переваги електронної комерції для споживачів:

– *Нижчі ціни.* Товари в інтернет-магазинах зазвичай коштують дешевше, ніж у фізичних магазинах, завдяки скороченню витрат на оренду та персонал.

– *Доступність 24/7.* Магазини працюють безперервно, що дозволяє здійснювати покупки в будь-який час доби.

– *Порівняння товарів.* Споживачі можуть легко порівнювати ціни, характеристики товарів і послуг, що унеможливлює безпідставне завищення цін.

– *Доступ до унікальних товарів.* Можливість придбання елітних, рідкісних товарів чи послуг у зарубіжних магазинах, участь в онлайн-аукціонах, бронювання готелів та інші послуги від іноземних компаній.

– *Конфіденційність.* Покупки можна здійснювати без розголошення особистої інформації.

– *Швидка доставка цифрових продуктів.* Нематеріальні товари, такі як програмне забезпечення, музика чи електронні книги, доставляються миттєво через мережу.

– *Відгуки та рейтинги.* Можливість перегляду відгуків інших покупців, що допомагає зробити обґрунтований вибір.

– *Прозорість.* Компанії стають більш відкритими та інформативними для клієнтів.

Переваги електронної комерції для виробників:

– *Відсутність необхідності у великих торгових площах.* Скорочуються витрати на оренду приміщень та обладнання.

– *Зменшення витрат.* Зниження витрат на маркетинг, технічну підтримку та обслуговування клієнтів завдяки автоматизованим рішенням.

– *Скорочення потреби в персоналі.* Завдяки автоматизації процесів зменшуються витрати на оплату праці.

– *Доступ до світових ринків.* Можливість виходу на глобальний ринок без географічних обмежень.

– *Рівні умови.* Електронна комерція надає однакові можливості для розвитку як великим корпораціям, так і малим підприємствам.

Ці переваги роблять електронну комерцію привабливою як для споживачів, так і для бізнесу, сприяючи її стрімкому розвитку та популяризації у світі.

Недоліки електронної комерції для споживачів:

– *Процедура ідентифікації.* Споживачі змушені проходити ідентифікацію, що може викликати занепокоєння щодо конфіденційності їхніх даних та можливості контролю їхньої діяльності.

– *Неможливість перевірки якості.* Покупці не можуть фізично оцінити товар або послугу до моменту отримання, що може призвести до розчарувань.

– *Складність повернення.* Процедура повернення товару часто є складною та потребує додаткових зусиль, таких як оформлення заявок і пересилання товару.

Недоліки електронної комерції для виробників:

– *Втрата особистого контакту.* Відсутність прямого контакту між продавцем і покупцем обмежує можливість використовувати комунікативні навички продавця для впливу на вибір клієнта.

– *Посилення конкуренції.* Ринок стає більш конкурентним, оскільки споживачі мають доступ до численних альтернатив.

– *Високі інвестиції.* Потреба у постійному оновленні технологічної бази для підтримки конкурентоспроможності вимагає значних фінансових витрат.

– *Обмеження на завищення цін.* Прозорість цінової політики в Інтернеті ускладнює завищення цін без обґрунтованих причин.

– *Інформаційна безпека.* Ризики пов’язані з витоком даних, фінансовими шахрайствами та порушеннями прав інтелектуальної власності.

– *Потреба у кваліфікованих кадрах.* Для ефективного управління електронною комерцією потрібні висококваліфіковані фахівці, яких складно знайти та утримати.

Ці недоліки підкреслюють, що, незважаючи на численні переваги, електронна комерція також має свої виклики, які вимагають уваги як від споживачів, так і від бізнесу.

Розвиток електронної комерції в Україні тісно пов'язаний із станом телекомунікаційної інфраструктури. Держава відіграє ключову роль у сприянні цьому процесу через такі заходи:

– *Цифровізація ліній зв'язку та впровадження передових технологій.* Перехід від аналогових до цифрових мереж зв'язку є необхідним для підвищення якості та швидкості передачі даних. Впровадження сучасних технологій, таких як 4G та 5G, сприятиме покращенню доступу до Інтернету та мобільного зв'язку.

– *Підтримка конкуренції та створення умов для нових операторів.* Держава повинна забезпечувати рівні умови для всіх учасників ринку, сприяючи здоровій конкуренції та спрощуючи процедури входження нових операторів на ринок.

– *Розвиток телекомунікаційної галузі в сільських та гірських районах.* Особлива увага має приділятися цифровізації віддалених регіонів, забезпечуючи їх якісним зв'язком та доступом до Інтернету. Державна підтримка операторів у цих зонах є важливою для подолання цифрового розриву.

– *Розширення спектру послуг через новітні технології.* Впровадження інноваційних рішень у сфері телекомунікацій дозволить надавати споживачам широкий спектр послуг, підвищуючи їх якість та доступність.

Таким чином, комплексний підхід держави до розвитку телекомунікаційної інфраструктури сприятиме подальшому зростанню електронної комерції в Україні.

1.3 Огляд та аналіз представлених рішень для автоматизації обліку товарів

Програми для обліку товарів спеціалізуються на різних аспектах виробничої, торговельної та складської діяльності. Кожне підприємство обирає рішення, яке найбільш відповідає його специфіці, з огляду на функціональність, зручність і потреби. Перед розробкою власної системи обліку важливо вивчити існуючі аналоги, провести їх аналіз і врахувати сильні сторони, а також уникнути повторення слабкостей.

1.3.1 Сервіс обліку та управління малих підприємств ІС «LiteBox»

LiteBox — це програмний сервіс, спрямований на забезпечення обліку, торгівлі та фінансового управління для малих підприємств. Його функціонал охоплює шість ключових напрямків [5]:

Основні функції LiteBox:

– *Складський облік* — управління запасами товарів на складі.

– *Управління торгівлею* — організація та контроль торговельних процесів.

– *Аналітичні звіти* — автоматизація створення звітів для аналізу ефективності бізнесу.

– *Управління закупівлею* — моніторинг і контроль закупівельних операцій.

– *Документи* — ведення облікових документів, таких як ТОРГ-12, ТОРГ-16 та інші.

– *Маркетингові інструменти* — застосування стратегій для просування продукції.

Програма працює на базі хмарних технологій, що дозволяє користувачам отримувати доступ до сервісу з будь-якого пристрою, підключеного до Інтернету.

Переваги LiteBox:

– *Мультиплатформенність.* Доступність на різних пристроях (ПК, ноутбук, планшет).

– *Безкоштовний тариф.* Наявність базового безкоштовного тарифу та доступних цін на платні функції.

– *Пробний період.* Можливість тестування повного функціоналу протягом 14 днів.

– *Спеціальні функції.* Додаткові можливості для продавців алкогольної продукції.

– *Цілодобова підтримка.* Постійна технічна допомога для користувачів.

– *Безпека даних.* Зберігання інформації в хмарі з сертифікатом Tier3.

– *Автоматизація документів.* Генерація первинних облікових документів для підприємців.

Недоліки LiteBox:

– *Обмежена інтеграція з інтернет-магазинами.* Відсутність підтримки онлайн-продажів.

– *Складний інтерфейс.* Вимагає навчання та звикання.

– *Початкове налаштування.* Програма має складний стартовий етап конфігурації.

– *Обмеження клієнтської бази.* Відсутня функціональність повноцінного обліку клієнтів із дисконтними картками.

– *Недостатня гнучкість.* Неможливість додавання довільних властивостей у товарній картці чи налаштування функцій під потреби користувача.

Висновки:

LiteBox пропонує необхідний функціонал для малих підприємств у сфері торгівлі, зокрема для управління складом, торгівлею та фінансами. Проте програма потребує вдосконалення, особливо в аспектах інтерфейсу, інтеграції з онлайн-торгівлею та гнучкості налаштувань. Вона є привабливим рішенням для підприємців, які лише починають використовувати облікові системи, але її повний потенціал може бути реалізований за умови подальших покращень.

1.3.2 Сервіс обліку роздрібної торгівлі ІС «CloudShop»

CloudShop — це програмний продукт, призначений для онлайн-обліку в роздрібній торгівлі, що підтримує широкий спектр операцій і підходить для малого та середнього бізнесу [6].

Ключові функції CloudShop:

- Оформлення продажів та повернень.
- Проведення закупівель.
- Завантаження номенклатури з табличних файлів.
- Імпорт і експорт баз даних постачальників та клієнтів.
- Облік приходу та витрат коштів.
- Налаштування системи знижок.
- Формування статистики та аналітичних звітів щодо продажів.

CloudShop є універсальним додатком, який не прив’язаний до специфічного національного законодавства, що робить його придатним для використання у різних країнах.

Переваги CloudShop:

- *Мультиплатформенність.* Програма доступна для використання як на комп’ютерах, так і на мобільних пристроях.
- *Інтеграція з інтернет-магазинами.* Забезпечується підтримка підключення до онлайн-торгівлі.

– *Зручний інтерфейс.* Простий у використанні, що полегшує початкове налаштування.

– *Гнучкі тарифні плани.* Орієнтовані на малі підприємства, що дозволяє підібрати оптимальне рішення.

– *Модуль клієнтської бази.* Полегшує управління клієнтами та їхніми даними.

– *Пробний період.* Можливість тестування всього функціоналу протягом 14 днів.

– *Підтримка багатопрофільної торгівлі.* Дозволяє підключати кілька магазинів та складів.

Недоліки CloudShop:

– *Обмежена підтримка касового обладнання.* Не всі моделі касових апаратів підтримуються.

– *Відсутність технічної підтримки.* Немає доступу до професійного обслуговування у разі виникнення проблем.

– *Базовий тариф.* Непрактичний для магазинів, які працюють лише з одним постачальником.

– *Відсутність кастомізації.* Неможливість налаштувати програму під конкретні потреби користувача.

– *Недостатній функціонал для ресторанів.* Відсутність можливостей для прийому замовлень або створення індивідуальних налаштувань.

Висновки:

CloudShop добре підходить для малих магазинів, які працюють за простою моделлю "купи-продай". Проте програма може бути обмеженою для більш складних сфер, таких як ресторанний бізнес чи роздрібна торгівля зі складною логістикою. Розробникам варто звернути увагу на розширення функціоналу для кращої адаптації до потреб користувачів, включаючи вдосконалення технічної підтримки та підтримку різноманітного обладнання.

1.3.3 Сервіс складського обліку ІС «Мій Склад»

"Мій Склад" — це популярна програма для складського обліку, яка забезпечує стабільну роботу та широкий функціонал для малого, середнього і навіть великого бізнесу. Її можливості охоплюють як торговельні, так і виробничі процеси [7].

Переваги програми "Мій Склад":

- *Широкий функціонал.* Підтримка оптової та роздрібної торгівлі, малих виробництв і навіть сфери громадського харчування.
- *Стабільна робота.* Програма демонструє високу надійність у використанні.
- *Дружній інтерфейс.* Зручний для користувачів, легко освоюється новими співробітниками.
- *Демо-версія.* Можливість протестувати всі функції програми перед покупкою.
- *Мультиплатформенність.* Програма доступна для операційних систем Windows, MacOS, Android, Linux та iOS.
- *Відкрите API.* Дозволяє інтеграцію з іншими системами та гнучке налаштування під потреби бізнесу.

Недоліки програми "Мій Склад":

- *Висока абонентська плата.* Щомісячна вартість використання вища за середньоринкові показники.
- *Відсутність шаблонів для продажу.* Це може ускладнити роботу підприємствам із великим асортиментом товарів.

Висновки та рекомендації:

"Мій Склад" був розроблений для малих підприємств і корпорацій, проте отримав популярність і серед великих компаній завдяки своєму універсальному функціоналу. Водночас, відсутність регулярних оновлень і покращень викликає незадоволення частини користувачів.

– Знизити абонентську плату або створити більш доступний тариф для малого бізнесу.

– Розробити шаблони для продажу, що спростять використання програми для підприємств із великим асортиментом товарів.

– Активніше впроваджувати оновлення з новими функціями, враховуючи відгуки користувачів.

1.3.4 Хмарна POS-система - Poster POS

Poster POS призначена для автоматизації бізнес-процесів у закладах громадського харчування та роздрібної торгівлі. Вона допомагає впорядкувати операції, оптимізувати витрати, мінімізувати крадіжки та підвищити продажі. Система надає можливість здійснювати продажі, вести фінансовий, бухгалтерський та складський облік, керувати персоналом і працювати з клієнтською базою. [8]

Poster POS підходить для різних форматів бізнесу, включаючи ресторани, кафе, бари, пекарні, магазини та франшизи. Вона адаптується під потреби як невеликих кав'ярень, так і мереж закладів. Система працює на звичайних планшетах і ноутбуках, підтримує всі типи замовлень — у закладі, на виніс та доставку.

Основні функції Poster POS включають:

– *Каса:* зручне робоче місце офіціанта або касира з інтуїтивно зрозумілим інтерфейсом, що дозволяє швидко приймати замовлення та обробляти платежі.

– *Адмін-панель:* інструменти для управління меню, складом, фінансами, аналітикою та маркетингом.

– *Аналітика:* вбудовані звіти допомагають аналізувати продажі, ефективність персоналу, витрати та інші ключові показники.

– *Складський облік:* контроль запасів, автоматичний розрахунок

собівартості страв, управління постачаннями та інвентаризація.

– *Програми лояльності:* робота з базою клієнтів, налаштування системи знижок та бонусів для залучення та утримання гостей.

– *Інтеграції:* підтримка інтеграцій з різними сервісами, такими як онлайн-замовлення, доставка, платіжні системи та інші.

Приклади підприємств зображено на рисунку 1.1, а на рисунку 1.2 наведено приклад функцій програми.

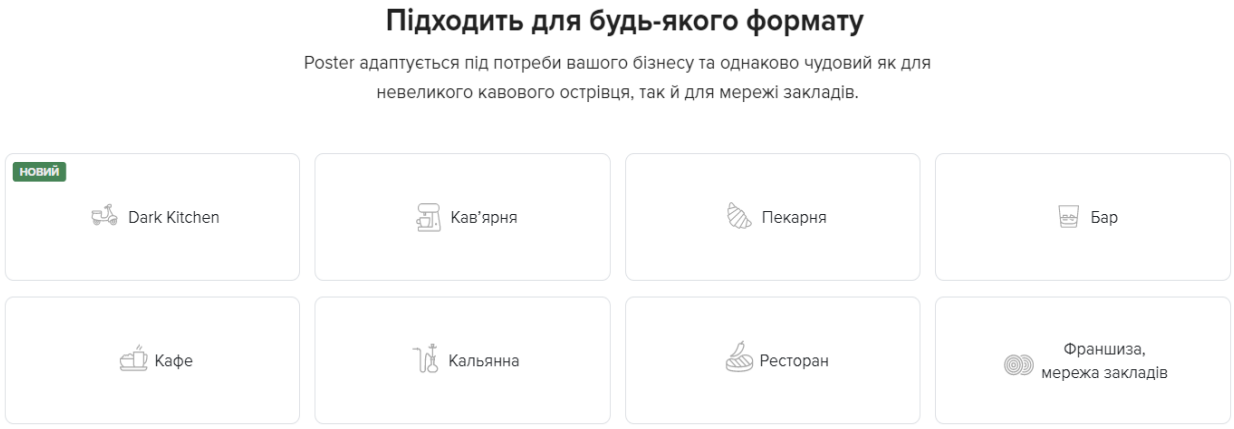
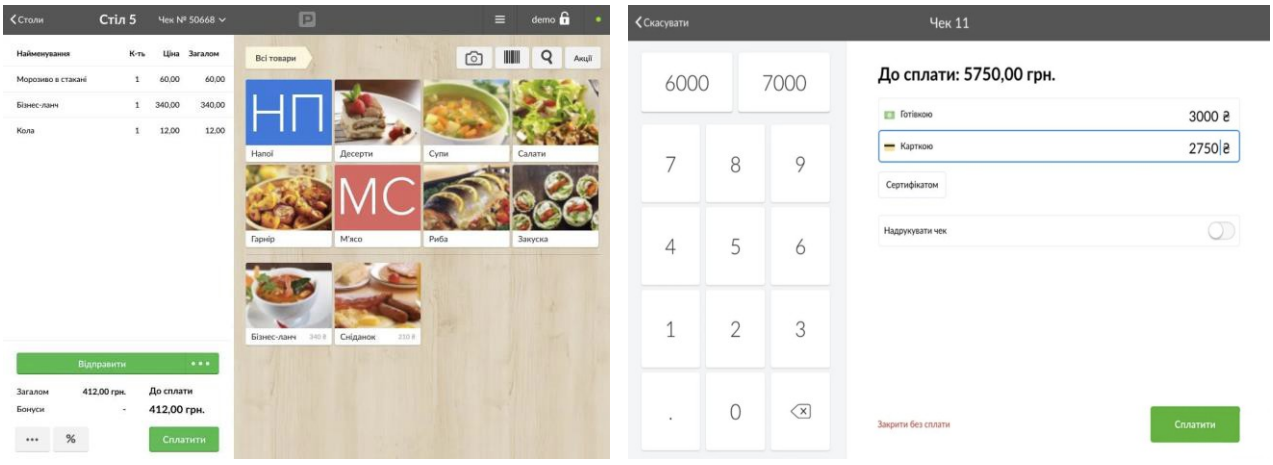


Рисунок 1.1 - Перелік підприємств для застосування програми Poster POS



а) б)

а) оформлення замовлень; б) розрахунок

Рисунок 1.2 – Функції Poster POS

Poster POS працює в хмарі, що забезпечує доступ до даних з будь-якої точки світу через інтернет. Система продовжує функціонувати навіть при відсутності інтернет-з'єднання, а дані синхронізуються автоматично після відновлення підключення. Вартість використання Poster POS починається від 24 євро на місяць, з можливістю безкоштовного тестового періоду протягом 15 днів.

Висновки:

Загалом, Poster POS є ефективним інструментом для автоматизації та оптимізації роботи закладів громадського харчування та роздрібної торгівлі, надаючи широкий спектр функцій для управління бізнесом.

1.3.5 Сервіс бухгалтерського та податкового обліку BAS Бухгалтерія

Бухгалтерія — це сучасне програмне забезпечення, призначене для автоматизації бухгалтерського та податкового обліку, а також підготовки регламентованої звітності. Воно підходить для підприємств різних масштабів і видів діяльності, включаючи оптову та роздрібну торгівлю, надання послуг і виробництво [9].

Основні можливості BAS Бухгалтерії:

– *Складський облік:* Ведення обліку товарно-матеріальних цінностей, контроль залишків на складах, облік руху товарів.

– *Облік торговельних операцій:* Реєстрація операцій купівлі-продажу, управління ціноутворенням, облік знижок та націнок.

– *Облік розрахунків з контрагентами:* Ведення взаєморозрахунків з постачальниками та покупцями, контроль дебіторської та кредиторської заборгованості.

– *Ведення обліку діяльності кількох організацій:* Можливість обліку господарської діяльності декількох юридичних осіб в одній інформаційній

базі.

– *Облік комісійної торгівлі:* Підтримка операцій комісійної торгівлі, включаючи субкомісію.

– *Завершальні операції періоду:* Автоматизація процесів закриття періоду, розрахунок фінансових результатів.

– *Експрес-перевірка обліку:* Інструменти для швидкої перевірки правильності ведення обліку та виявлення можливих помилок.

– *Регламентована звітність:* Формування та подання обов'язкової звітності відповідно до чинного законодавства України.

На рисунку 1.3 зображено приклад застосування BAS Бухгалтерії.

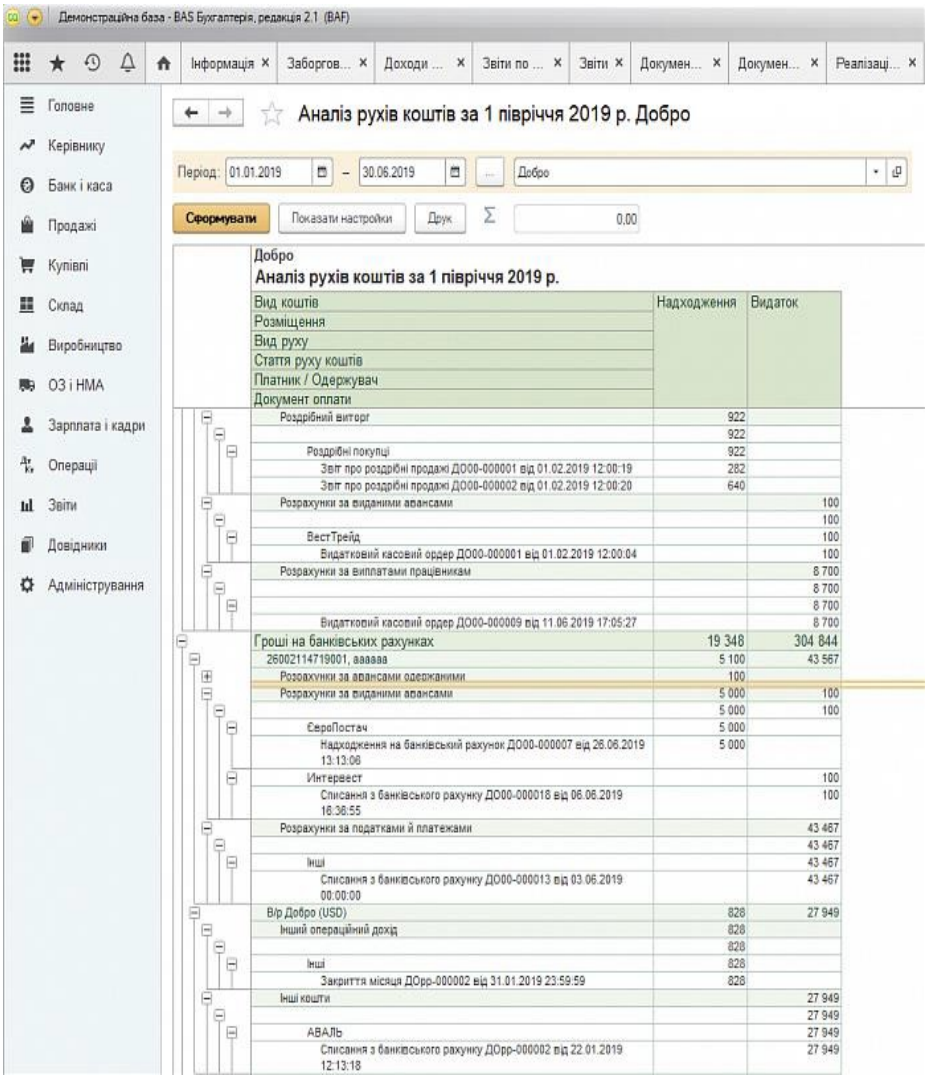


Рисунок 1.3 – Аналіз руху коштів в BAS Бухгалтерія

Висновки:

– BAS Бухгалтерія є якісною альтернативою російському продукту "1С:Підприємство", пропонуючи локалізоване рішення, адаптоване до українських стандартів обліку та законодавства.

– Програма забезпечує зручний інтерфейс, регулярні оновлення та підтримку, що робить її надійним інструментом для бухгалтерів та фінансових спеціалістів.

Для детальнішого ознайомлення з можливостями BAS Бухгалтерії можна скористатися демонстраційними базами, доступними онлайн.

1.3.6 Сервіс обліку товарів RemOnline

RemOnline — це програма для обліку товарів, яка підходить для різних типів бізнесу [10].

Вона надає такі можливості:

– *Контроль закупівель та оплати постачальників:* відстежуйте процеси закупівель і своєчасно здійснюйте розрахунки з постачальниками.

– *Відстеження стану та руху товарів:* контролюйте залишки на складі, переміщення товарів між складами та їх продаж.

– *Контроль роботи персоналу та складський облік:* аналізуйте ефективність роботи співробітників і ведіть точний облік товарів на складі.

– *Підключення онлайн-каси та інших інструментів продажу:* інтегруйте програму з онлайн-касами та іншими системами для спрощення процесу продажу.

– *Додатково:* дозволяє швидко наповнити торговий склад товарами та сформувати перелік послуг за допомогою імпорту з Excel-файлу, створювати категорії та підкатегорії для різних номенклатур, щоб стежити за продажами певних категорій товарів.

– *Мобільний додаток для обробки замовлень*: дозволяє додавати товари

та послуги, скануючи їх штрих-коди прямо з програми, а також сканувати QR-коди з документів для швидкого пошуку потрібних замовлень у базі.

– *Надає можливість додавати різні типи цін* (наприклад, роздрібну, ремонтну, акційну) та налаштовувати для них націнки, які автоматично розраховуються від закупівельної вартості товару. Також є можливість надавати клієнтам разові або постійні знижки, що допомагає підвищити рівень лояльності клієнтів.

Деякі можливості програми зображено на рисунку 1.4

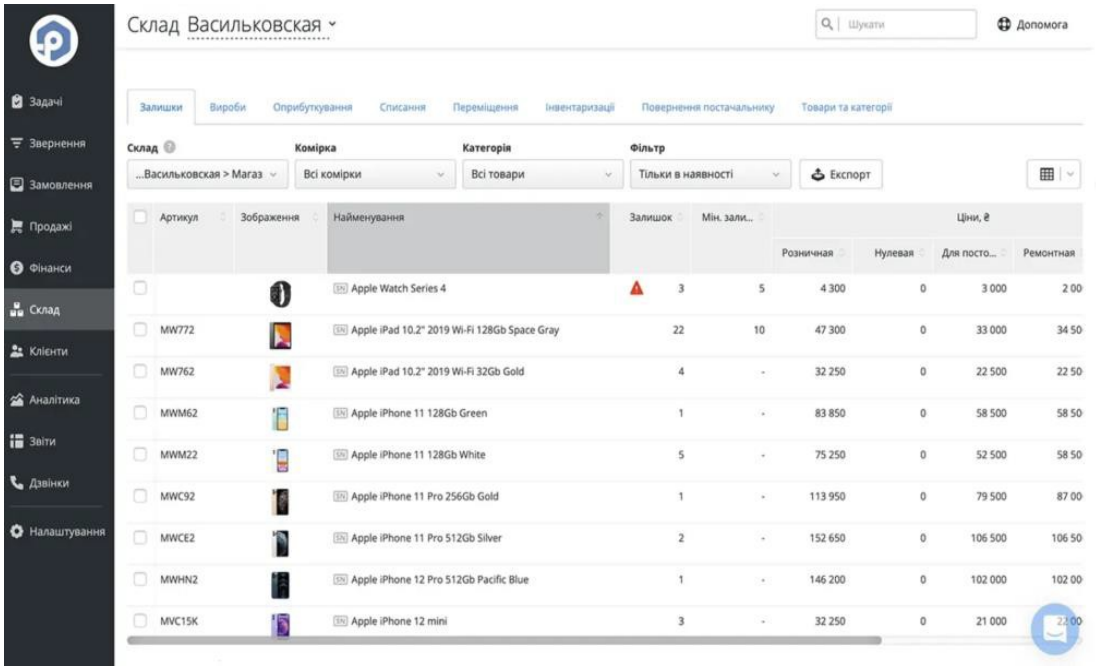


Рисунок 1.4 – Функціонал RemOnline

Висновки:

Є можливість створити загальну базу покупців і постачальників для всіх ваших філій, що зручно при переведенні замовлення в іншу майстерню або магазин. Таким чином, RemOnline є потужним інструментом для автоматизації та оптимізації бізнес-процесів, пов'язаних з обліком товарів, управлінням продажами та взаємодією з клієнтами.

1.4 Вимоги до систем обліку товарів

Розробка програми для обліку товарів повинна відповідати сучасним потребам підприємств і забезпечувати повний контроль за товарообігом. *Основні функціональні вимоги включають:*

– *Контроль залишків товарів.* Система повинна автоматично контролювати залишки кожного товару на складі. У разі розбіжностей між фактичними та зареєстрованими даними система має попереджати користувача про можливі помилки.

– *Управління інформацією про товари:* Реєстрація, зберігання та оновлення інформації про товари.

– Унікальна ідентифікація кожного товару для уникнення плутанини.

– Можливість додавання нових товарів та редагування існуючих.

– Пошук і перегляд товарів за кодом, назвою чи іншими атрибутами.

– Відображення детальної інформації про обраний товар.

– *Управління помилками:* У разі допущення помилки користувач повинен мати можливість її виправити. Всі зміни фіксуються в історії, що дозволяє відслідковувати попередні дії.

– *Робота з базою даних:* Система повинна бути інтегрована з базою даних для зберігання та обробки інформації. База даних повинна підтримувати роботу з великими обсягами інформації, забезпечуючи швидкий доступ і стабільність. Необхідно використовувати реляційну або нереляційну базу даних залежно від потреб проекту (наприклад, MySQL, PostgreSQL або MongoDB).

– *Інтеграція з касовою системою:* Система товарообліку повинна автоматично синхронізуватися з касовою системою магазину. При продажі

товару інформація про продані одиниці та залишки оновлюється в режимі реального часу.

– *Управління продажами:* Кожна транзакція продажу фіксується в системі. Автоматичний розрахунок загальної вартості продажу. Зберігання історії операцій для подальшого аналізу.

– *Звітування:* Формування звітів про: Залишки товарів. Продажі за період. Взаєморозрахунки з постачальниками. Експорт звітів у зручні формати (Excel, PDF).

– *Масштабованість і зручність використання:* Інтуїтивно зрозумілий інтерфейс для швидкого освоєння користувачами. Можливість масштабування системи для роботи з кількома магазинами чи складами.

1.5 Особливості існуючих програм обліку та постановка задачі дослідження

Програми для складського обліку:

– Основна функція — управління запасами товарів на складі.

– Підтримка обліку залишків, надходжень, списань і переміщень між складами.

– Використовуються підприємствами з великим обсягом товарів.

Програми для обліку продажів:

– Орієнтовані на ведення обліку реалізації товарів.

– Автоматизація касових операцій, формування чеків, управління ціноутворенням.

– Підходять для роздрібних магазинів або сфери послуг.

Існуючі системи здебільшого спеціалізуються або на складському обліку, або на продажах. Проте для більшості підприємств, особливо малого та середнього бізнесу, потрібне інтегроване рішення, яке дозволяє:

– Здійснювати повний контроль товарообігу: від надходження товарів на склад до їх продажу кінцевому споживачеві.

– Забезпечувати прозорість усіх операцій і взаємозв'язок між етапами обліку.

– Скоротити витрати часу і зусиль на перемикання між різними програмами для різних видів обліку.

Сильні сторони існуючих програм:

– Простота в освоєнні для користувачів без спеціальної підготовки.

– Наявність готових шаблонів для звітів та автоматизації рутинних задач.

– Можливість інтеграції з додатковим обладнанням, таким як сканери штрих-кодів або POS-термінали.

– Хмарне зберігання даних, що забезпечує доступ до системи з будь-якого місця.

Слабкі сторони:

– Відсутність гнучкості для адаптації до унікальних бізнес-процесів.

– Неінтегрованість складських і торговельних функцій в одній системі.

– Висока вартість для малого бізнесу через нав’язування додаткових функцій, які не завжди використовуються.

– Складність оновлення системи та адаптації до нових вимог.

Рекомендації для розробки інтегрованої системи:

– *Поєднання функціоналу:*

  – Об’єднати складський і торговельний облік у єдиній системі.

  – Забезпечити взаємозв'язок між етапами обліку: від надходження товарів до їх реалізації.

– *Гнучкість налаштувань:*

– Надати можливість налаштовувати програму під специфіку кожного підприємства.

– Підтримка створення власних звітів, типів операцій і параметрів обліку.

– *Масштабованість:*

– Система повинна бути придатною для малого бізнесу, але також легко масштабуватися для великих підприємств.

– *Доступність:*

– Включити хмарне зберігання даних із можливістю доступу з будь-якого пристрою.

– Забезпечити підтримку різних операційних систем і пристроїв.

– *Зручність використання:*

– Інтуїтивний інтерфейс, який дозволяє швидко навчитися працювати з програмою.

– Автоматизація рутинних процесів, таких як формування документів і звітів.

Висновки до розділу:

Був проведений аналіз предметної області, поставлена задача розробки, була досліджена предметна область, загальні принципи застосунків для ведення обліку та принцип їх роботи. Розглянуті існуючі програми для порівняння та встановлення функціональності власної системи обліку.

Об’єднання функцій складського і торговельного обліку в одній системі створить унікальний інструмент, який буде затребуваним серед підприємств різного масштабу. Така система дозволить оптимізувати управління товарообігом, зменшити помилки, спричинені людським фактором, і підвищити ефективність роботи бізнесу.

РОЗДІЛ ІІ
ВИБІР ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ СИСТЕМИ ОБЛІКУ ІНТЕРНЕТ-МАГАЗИНУ ТА ЇЇ СТРУКТУРА

2.1 Аналіз технологій для розробки систем обліку товарів

Розробка системи обліку для інтернет-магазину потребує ретельного вибору технологій, щоб забезпечити її функціональність, надійність, масштабованість і відповідність бізнес-потребам. Розглянемо ключові технології, які можуть бути використані для створення такої системи:

- *Основи веб-розробки: HTML/CSS:*
  - *HTML (HyperText Markup Language):* Використовується для створення структури веб-сторінок, таких як форми для введення даних, каталоги товарів тощо.
  - *CSS (Cascading Style Sheets):* Забезпечує оформлення сторінок, включаючи адаптивний дизайн для різних пристроїв (десктоп, планшет, смартфон).

Обґрунтування: HTML і CSS є стандартом для розробки інтерфейсів користувача, які є легкими у використанні та забезпечують привабливий вигляд.

- *JavaScript та фронтенд-фреймворки*
  - *JavaScript:* Використовується для інтерактивності, таких як динамічне оновлення сторінок, без перезавантаження.
  - *Фреймворки (React, Angular, Vue.js):* Полегшують розробку складних інтерфейсів за допомогою компонентів та дозволяють легко масштабувати систему.

Обґрунтування: Використання JavaScript і сучасних фреймворків значно спрощує процес розробки інтерактивних та зручних для користувача веб-інтерфейсів.

– *Серверні технології*

–*Node.js з фреймворком Express:* Висока швидкість обробки запитів завдяки асинхронній моделі.

–*Python з Django:* Забезпечує швидку розробку, зручний ORM для роботи з базами даних та вбудовані функції безпеки.

–*PHP з Laravel:* Добре підходить для побудови систем середньої складності завдяки простоті та широкій підтримці.

–*ASP.NET:* Оптимальний вибір для великих корпоративних рішень.

Обґрунтування: Вибір залежить від масштабу проекту та специфіки облікової системи. Наприклад, для малих рішень Node.js або Laravel можуть бути ідеальними, тоді як для великих корпоративних систем доцільно використовувати ASP.NET.

– *Робота з базами даних*

– *Реляційні бази даних* (*MySQL, PostgreSQL):* Для систем із чіткою структурою даних (товари, клієнти, замовлення).

– *Нереляційні бази даних (MongoDB):* Для гнучкого зберігання даних, наприклад, з динамічною структурою товарів.

Обґрунтування: Реляційні бази підходять для більшості систем обліку завдяки структурованості, тоді як MongoDB може бути використана для специфічних проєктів із потребою в швидкому масштабуванні.

– *REST ful API*

–Забезпечує зв'язок між фронтендом та бекендом, дозволяючи обмінюватися даними через HTTP-запити (GET, POST, PUT, DELETE).

Обґрунтування: RESTful API дозволяє інтегрувати систему обліку з іншими сервісами, такими як CRM, ERP чи платіжні системи.

– *Хмарні платформи*

– *Amazon Web Services (AWS), Google Cloud, Microsoft Azure:* Забезпечують надійне зберігання даних, легке масштабування та високу доступність.

Обґрунтування: Хмарні платформи дозволяють економити на інфраструктурі та забезпечують швидке розгортання системи.

- *Інструменти для автоматизації та тестування*
  - *Docker:* Для контейнеризації додатка та зручного розгортання.
  - *Jenkins:* Для автоматизації процесів розробки та деплойменту.
  - *Postman:* Для тестування RESTful API.

Обґрунтування: Ці інструменти забезпечують стабільність системи та швидке впровадження оновлень.

Рекомендації щодо вибору технологій:

- *Для малих проектів:* Node.js з Express або Laravel (PHP), реляційна база даних (MySQL), мінімалістичний інтерфейс (HTML, CSS, JavaScript).
- *Для середніх проектів:* Django (Python) або React з Node.js, PostgreSQL або MongoDB, інтеграція RESTful API.
- *Для великих корпоративних рішень:* ASP.NET, мікросервісна архітектура, хмарні сервіси для масштабування.

Вибір технологій має базуватися на масштабі проєкту, вимогах до продуктивності та зручності використання, а також бюджеті на розробку.

2.2 Обґрунтування вибору технологій для розробки системи обліку інтернет-магазину

Для створення системи обліку товарів в інтернет-магазині обрано стек технологій: Laravel 9, Voyager 1.6, MySQL та PHP. Нижче наведемо аргументи на користь цих рішень:

Фреймворк Laravel 9 це потужний PHP-фреймворк, який забезпечує швидкий розвиток додатків завдяки широкому набору функцій.

Переваги:

- Модульність та масштабованість. Laravel дозволяє легко додавати нові функції завдяки використанню пакетів та модулів.

– Вбудовані механізми безпеки. Фреймворк підтримує захист від SQL-ін'єкцій, XSS-атак, CSRF та інші загрози.

– ORM Eloquent. Простий і зручний інструмент для роботи з базою даних, який пришвидшує розробку CRUD-операцій.

– Широка спільнота. Це гарантує доступ до численних ресурсів, бібліотек та готових рішень.

– Масштабованість. Laravel підходить як для невеликих проєктів, так і для масштабних систем.

Voyager 1.6 це пакет адміністрування, який інтегрується з Laravel, створюючи гнучкий інтерфейс для управління даними.

Переваги:

– Зручна панель адміністратора. Надає готовий бекенд-інтерфейс для управління товарами, категоріями, замовленнями та іншими даними.

– Можливість кастомізації. Легко адаптується до потреб проєкту завдяки інтеграції з Laravel.

– Швидке налаштування. Voyager значно скорочує час розробки системи адміністрування.

– Підтримка мультимовності. Актуально для інтернет-магазинів, орієнтованих на кілька ринків.

СУБД MySQL одна з найпопулярніших систем управління базами даних. Переваги:

– Швидкість роботи. MySQL забезпечує високу продуктивність навіть для великих обсягів даних.

– Надійність та перевірена часом архітектура. MySQL використовується в багатьох комерційних та open-source проєктах.

– Сумісність. Повністю інтегрується з Laravel та іншими PHP-рішеннями.

- Гнучкість у роботі з таблицями. MySQL підтримує реляційні структури даних, що зручно для моделювання зв’язків між товарами, категоріями, замовленнями та користувачами.

Мова програмування PHP це базова технологія для Laravel і Voyager. Переваги:

- Широке використання. PHP є стандартом для веб-розробки, підтримує велику кількість хостинг-платформ.
- Швидкість розробки. Велика кількість готових бібліотек та фреймворків спрощує процес створення системи.
- Активна підтримка. PHP постійно оновлюється, забезпечуючи сумісність з сучасними технологіями та безпеку.

Порівняємо обраний стек *Laravel 9 + Voyager 1.6 + MySQL + PHP* із популярними альтернативними рішеннями. Це допоможе оцінити їх переваги та недоліки для розробки системи обліку товарів в інтернет-магазині.

Laravel vs. Django (Python)

Django:

- Мова програмування: Python (популярна у машинному навчанні, аналізі даних і веб-розробці).
- Особливості:
  - Прискорена розробка завдяки "батареї в комплекті" (built-in tools).
  - Потужна ORM, схожа на Eloquent, для роботи з базами даних.
  - Більш суворе дотримання стандартів коду.
  - Прекрасно підходить для складних багатофункціональних додатків.
- Недоліки:
  - Складніший вхід для розробників, які не знайомі з Python.
  - Менше готових рішень для e-commerce.

Laravel:

- Швидше стартує для невеликих і середніх проєктів.
- Має багатшу екосистему пакетів, включно з Voyager.
- Простий для розробників, знайомих із PHP.

Висновок: *Laravel* краще підходить для швидкого створення інтернет-магазинів, тоді як *Django* підходить для складних проєктів із високими вимогами до стандартизації.

Voyager vs. WordPress (WooCommerce)

WordPress/WooCommerce:

- Переваги:
  - Величезна екосистема плагінів і шаблонів.
  - Швидке налаштування для простих інтернет-магазинів.
  - Підтримка безкодових рішень (для людей без технічних знань).
- Недоліки:
  - Низька продуктивність для великих каталогів товарів (без додаткової оптимізації).
  - Вразливість до атак через сторонні плагіни.
  - Обмежена гнучкість у кастомізації бізнес-логіки.

Voyager:

- Переваги:
  - Глибока інтеграція з Laravel дозволяє створювати кастомні рішення.
  - Проста реалізація специфічних вимог клієнта.
  - Легше масштабувати й підтримувати.
- Недоліки:
  - Більший час розробки, ніж у WordPress.
  - Вимагає технічних знань для налаштування та роботи.

Висновок: *Voyager* підходить для проєктів із нестандартними вимогами, тоді як *WordPress* ідеальний для швидкого старту простих магазинів.

MySQL vs. PostgreSQL

PostgreSQL:

- Переваги:
  - Потужніші можливості роботи з даними (наприклад, складні запити, JSONB, тригери).
  - Краща продуктивність при складних запитах.
  - Надійні транзакції для фінансових систем.
- Недоліки:
  - Складніша настройка та адміністрація.
  - Трохи менша популярність у PHP-розробників.

MySQL:

- Переваги:
  - Простота у використанні та висока швидкість для простих і середніх систем.
  - Широко підтримується в хостингах.
  - Легкість інтеграції з Laravel.
- Недоліки:
  - Менше можливостей для складних операцій із даними.
  - Не така потужна як PostgreSQL при роботі зі складними структурами.

Висновок: *MySQL* краще підходить для проєктів із простими або середньо складними запитами, тоді як *PostgreSQL* варто обрати для фінансових або аналітичних систем.

PHP (Laravel) vs. JavaScript (Node.js/Express)

Node.js/Express:

- Переваги:
  - Одна мова програмування для фронтенду та бекенду (JavaScript).
  - Швидкість роботи завдяки асинхронній архітектурі.

  - Підходить для додатків із реальним часом (чат, трекінг замовлень).
- Недоліки:
  - Складність роботи з реляційними базами даних (ORM типу Sequelize поступається Eloquent).
  - Менш стандартизована екосистема.

PHP (Laravel):

- Простота роботи з MySQL та реляційними базами.
- Більше готових рішень для класичних систем (інтернет-магазини, CRM).
- Підходить для серверних завдань із високим навантаженням.

Висновок: PHP із Laravel кращий для класичних e-commerce рішень, тоді як Node.js підходить для інтерактивних систем із високою динамікою.

Загальний висновок

Обраний стек технологій (Laravel 9 + Voyager 1.6 + MySQL + PHP) ідеально підходить для створення системи обліку товарів інтернет-магазину завдяки своїй гнучкості, надійності, швидкості розробки та можливості масштабування. Це рішення забезпечить ефективну роботу як для адміністративного персоналу, так і для кінцевих користувачів.

Альтернативні рішення, такі як Django або Node.js, підходять для специфічних сценаріїв, але вимагають більше ресурсів і часу на реалізацію. WordPress із WooCommerce підходить лише для простих проєктів і має обмежену гнучкість для кастомізації.

2.3 Проектування функціональних можливостей застосунку системи обліку товарів

Розроблюваний застосунок товарообліку забезпечує користувачам інтуїтивно зрозумілий інтерфейс для виконання завдань із управління товарообігом. Основні функції програми включають автоматизацію

операцій, контроль залишків і створення необхідних документів, що полегшує роботу персоналу та мінімізує помилки.

Загальні функціональні можливості застосунку:

- Інтуїтивний інтерфейс:
  - Використання кнопок та головного меню для навігації.
  - Зрозумілий дизайн для виконання операцій видатку, замовлення, внесення та створення товару.
- Підтримка офлайн режиму:
  - Можливість роботи без підключення до Інтернету.
  - Сумісність із версіями Windows, починаючи з Windows 7.
- Детальні повідомлення про помилки:
  - Оповіщення користувача з інструкціями щодо виправлення некоректних дій.

Функціональні модулі програми:

- Каталог (можливості):
  - Перегляд всього товару компанії.
  - Коригування існуючих товарів, створення нових із зазначенням параметрів (код, назва, характеристики, ціна тощо).
  - Видалення товарів, які більше не використовуються.
  - Пошук товарів за назвою або кодом постачальника.
- Постачальники (можливості):
  - Ведення бази постачальників із зазначенням їх контактних даних.
  - Створення нових постачальників із перевіркою унікальності.
  - Видалення постачальників за їх кодом.
  - Пошук за прізвищем або іншими критеріями.

– Замовлення (можливості):

– Перегляд списку створених замовлень.

– Пошук замовлень за кодом постачальника, кодом замовлення або датою.

– Створення нових замовлень.

– Формування документів надходження товару на основі обраного замовлення.

– Надходження товарів (можливості):

– Перегляд усіх документів надходжень.

– Пошук за номером документа, кодом замовлення, товару або постачальника.

– Створення нових документів надходжень із автоматичним оновленням залишків товару.

– Продажі (можливості):

– Перегляд документів продажів із можливістю фільтрації за датою чи кодом продажу.

– Створення документів повернення товару.

– Внесення продажів із автоматичною перевіркою залишків товару.

– Розрахунок суми продажу автоматично.

– Реалізація заброньованих товарів через кнопку «Реалізувати бронь».

– Бронювання (можливості):

– Створення документів бронювання товару для клієнтів.

– Вибір товару через спеціальне вікно, що виключає помилки.

– Списання товару при реалізації документа бронювання через

модуль "Продажі".

Завдяки своїй функціональності програма дозволяє автоматизувати всі основні процеси обліку товарів, значно підвищуючи ефективність роботи компанії.

На рисунку 2.1 представлена діаграма варіантів використання (Use Case Diagram), яка відображає взаємодію між користувачем, застосунком системи обліку товарів та базою даних. Вона дає загальне уявлення про взаємодію користувача із системою та сприяє виявленню ключових функцій, необхідних для реалізації програми.

Рисунок 2.1 - Use Case Diagram системи

Опис елементів діаграми:

Ролі (Actors):

– Ліворуч зображено користувача, який взаємодіє із системою через інтерфейс програми.

– Праворуч відображено базу даних, яка слугує для збереження, оновлення та обробки інформації про товари, постачальників, замовлення тощо.

Варіанти використання (Use Cases):

– *Створення картки товару*

– Функція, що дозволяє додати новий товар у систему.

– Користувач вводить такі параметри, як код товару, назва, характеристики (колір, розмір тощо), ціна закупівлі та продажу.

– Ця функція забезпечує унікальність ідентифікатора товару, щоб уникнути помилок у базі даних.

– *Створення картки постачальника*

– Ця дія дає змогу зареєструвати нового постачальника в базі даних.

– Адміністратор вводить контактні дані, назву, код постачальника тощо.

– Система перевіряє унікальність даних постачальника, щоб запобігти дублюванню записів.

– *Створення документу замовлення постачальнику*

– Використовується для формування замовлення на товари, які потрібно придбати у постачальника.

– Містить інформацію про товари, їх кількість, ціну, а також дату замовлення.

– Система зберігає замовлення з унікальним кодом для подальшої обробки.

– *Створення документу надходження від постачальника*

– Ця функція реалізує реєстрацію факту отримання товарів на склад.

– Документ автоматично заповнюється на основі попередньо створеного замовлення.

– При підтвердженні надходження оновлюються залишки товарів у базі даних.

– *Створення документу продажу товару*

– Функція, яка фіксує операції продажу товарів.

– Користувач обирає товари з каталогу, вказує кількість і ціни, після чого система автоматично розраховує підсумкову суму.

– При створенні документа продажу залишки товарів зменшуються.

– *Створення документу бронювання товару*

– Дозволяє зарезервувати товари для конкретного клієнта.

– Під час бронювання товари тимчасово виключаються із доступних залишків.

– Реалізація бронювання може бути завершена через документ продажу.

Зв'язки між компонентами:

– *Користувач ↔ Застосунок:*

– Користувач викликає функції системи через головне меню або кнопки інтерфейсу.

– Система перевіряє введені дані, генерує документи та опрацьовує запити.

– *Застосунок ↔ База даних:*

– Усі дані (товари, постачальники, замовлення, продажі) зберігаються в базі даних.

– При створенні чи оновленні інформації застосунок взаємодіє з БД, забезпечуючи актуальність і цілісність даних.

Алгоритм роботи застосунку зображено на рисунку 2.2.

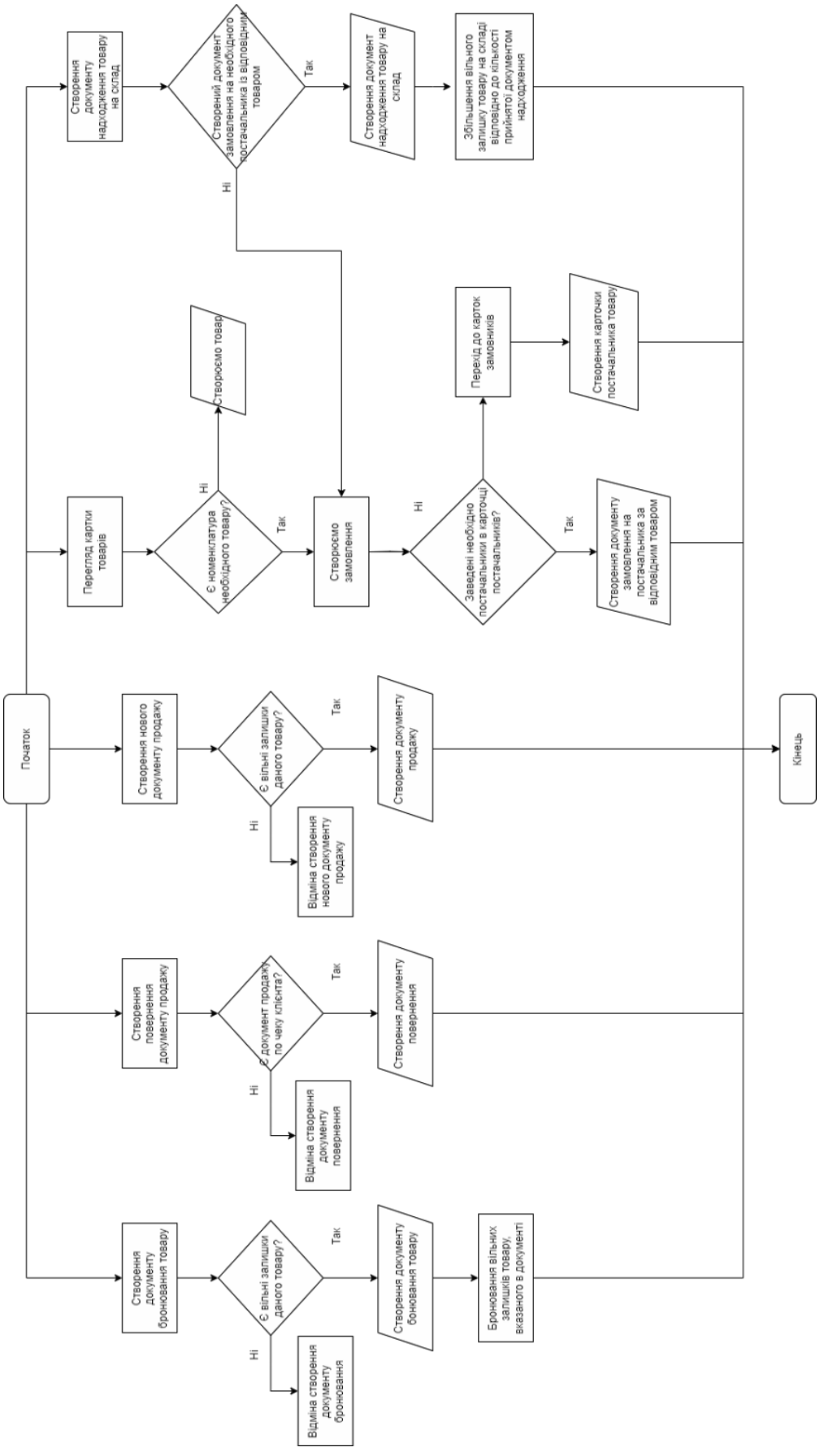


Рисунок 2.2 – Блок-схема системи

Основна функціональність системи:

– Користувач має доступ до функцій створення, редагування та перегляду інформації через застосунок.

– База даних забезпечує збереження та актуалізацію даних, необхідних для обліку товарів, роботи з постачальниками, замовленнями, продажами та бронюванням.

2.4 Проектування складових інформаційної системи обліку товарів магазину побутової хімії

Як було зазначено при обґрунтуванні засобів розробки інформаційної системи обліку товарів магазину побутової хімії, було вирішено використати фреймворк Laravel з розширенням Voyager для зручного проектування та розробки адміністративних панелей. Пакет Voyager реалізує надбудову над стандартним класом контролерів Laravel, та використовує концепцію BREAD, що дозволяє шаблонно опрацьовувати типові операції керування сутностями, такими як Browse (перегляд), Read (читання), Edit (редагування), Add (додавання), Delete (видалення), і контролер Voyager зазвичай реалізує ці дії.

На рис. 2.3 наведено діаграму класів для опису взаємозв’язків між класом контролера та сутністю Product, що описує товар.

На наведеній діаграмі класів елементи виконують наступні задачі:

– BreadController: наслідує базовий клас VoyagerBaseController, та реалізує методи для дій BREAD: browse, read, edit, update, add, store, delete.

– VoyagerBaseController: Базовий контролер, що надає загальні методи для всіх контролерів у Voyager.

– BaseModel є базовою моделлю для всіх моделей ORM-фреймворку Eloquent.

– Product є прикладом моделі (у даному випадку «Товар», з якою працює BreadController.

– Зв’язки: BreadController працює з моделями (наприклад, Product). Моделі наслідують базову модель BaseModel.

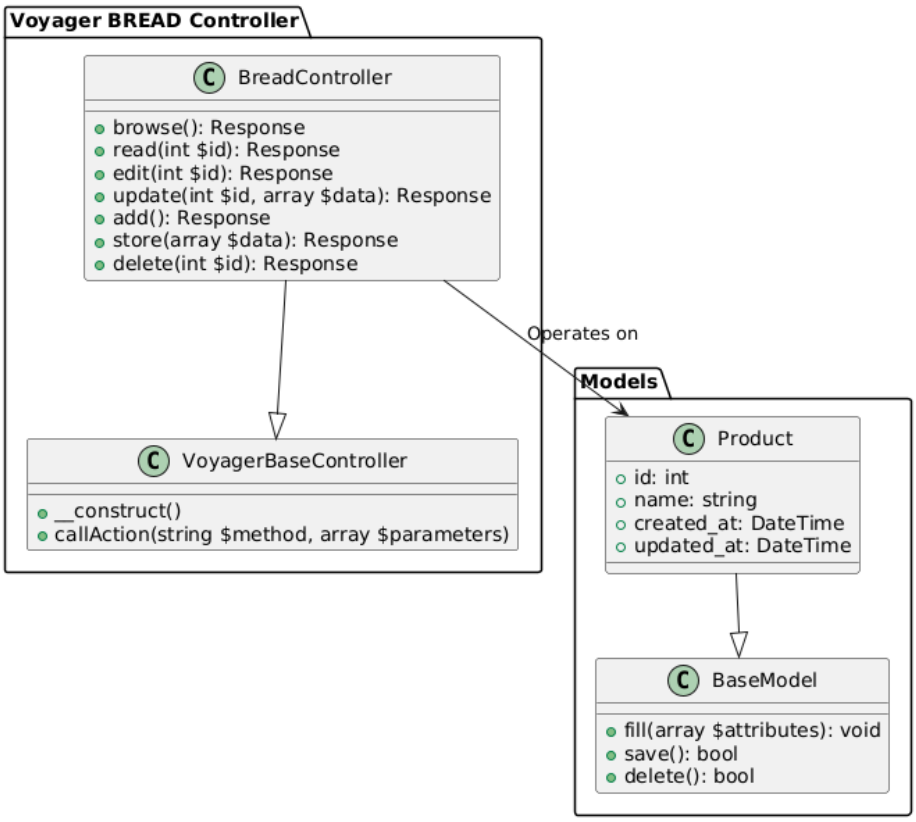


Рисунок 2.3 – Діаграма класів для опису взаємодії між контролером BreadController та класами предметної області

Для опису взаємодії між класами предметної області, ORM-фреймворком Eloquent, базою даних та контролером BREAD та представленнями побудуємо діаграму послідовності (рис. 2.4).

Опис дій:

– Користувач (User) – відправляє HTTP-запити для роботи з ресурсами через BREAD-контролер.

– BREAD Controller – обробляє запити, викликає методи Eloquent ORM для взаємодії з даними.

– Eloquent ORM – відповідає за зручну роботу з базою даних (отримання, збереження, оновлення, видалення).

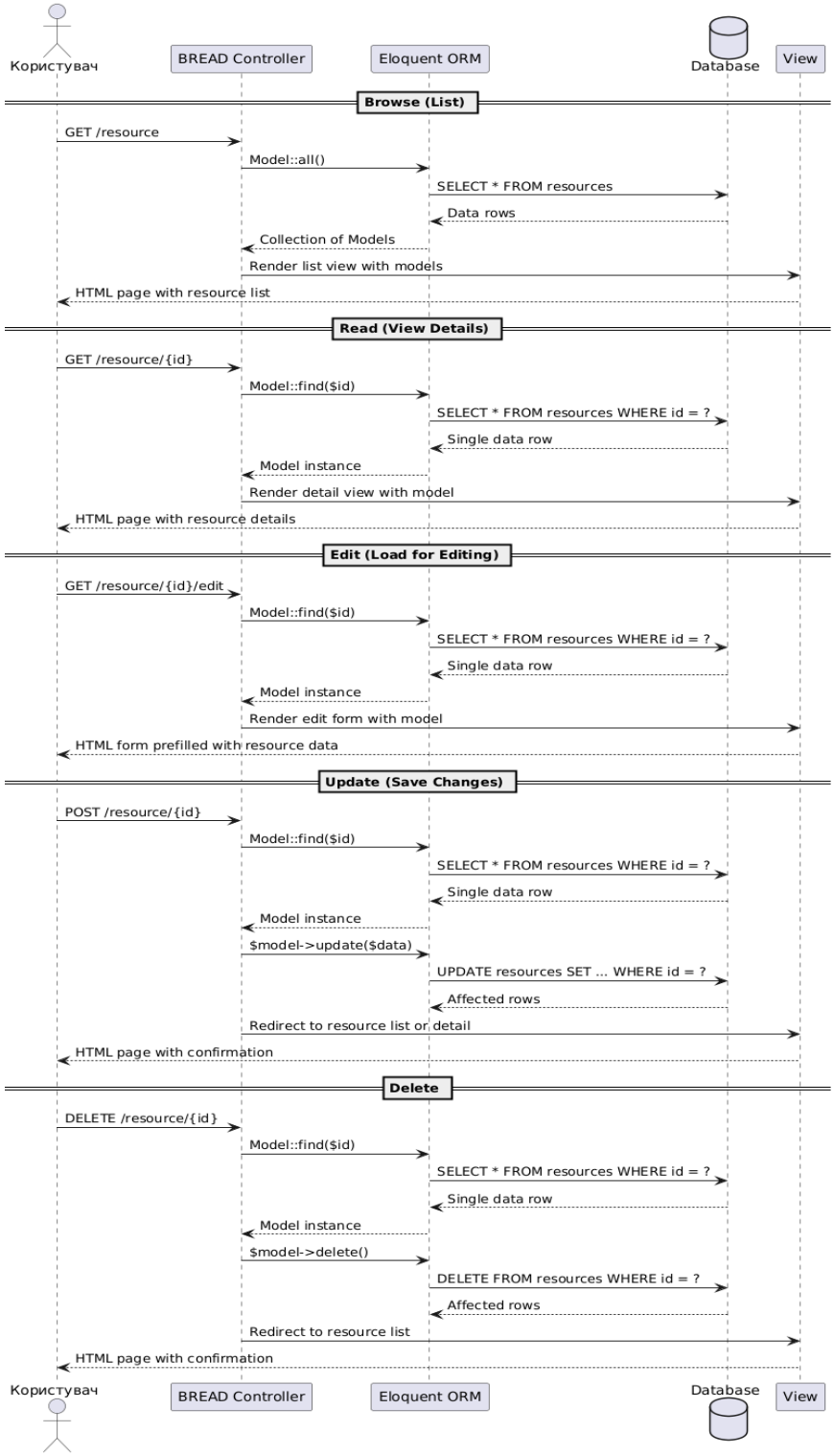


Рисунок 2.4 – Діаграма послідовності UML, що описує взаємодію між BREAD-контролером, ORM Eloquent, базою даних і представленнями

– База даних – берігає дані ресурсів і виконує SQL-запити.

– Представлення (View) – відповідає за рендеринг HTML-сторінок для відображення користувачеві.

Описана взаємодія реалізується наступним чином. Запити користувача (HTTP-методи GET, POST, DELETE) обробляються контролером. Контролер викликає методи моделі (через ORM) для доступу до бази даних. ORM працює з базою даних для виконання SQL-запитів.

Дані передаються назад у контролер, який використовує їх для формування представлення.

Представлення повертається користувачеві як HTML-сторінка.

Висновки до розділу:

У другому розділі було проведено всебічний аналіз і обґрунтування вибору технологій для створення системи обліку товарів інтернет-магазину, а також спроектовано її основні структурні складові. У підрозділах розглянуто такі ключові аспекти:

1. *Аналіз технологій для розробки систем обліку товарів* – здійснено огляд сучасних технологій, які використовуються для розробки інформаційних систем. Визначено їх переваги, недоліки та доцільність застосування в межах поставленого завдання.

2. *Обґрунтування вибору технологій* – на основі аналізу проведено обґрунтування вибору оптимальних технологій для розробки системи обліку інтернет-магазину. Зроблено акцент на тих рішеннях, які забезпечують високу продуктивність, масштабованість і легкість інтеграції.

3. *Проектування функціональних можливостей* – сформовано функціональні вимоги до системи, що включають облік товарів, управління запасами, моніторинг замовлень і автоматизацію ключових бізнес-процесів. Розроблено модель взаємодії між основними компонентами застосунку.

4. *Проектування інформаційної системи* – створено структуру інформаційної системи, що включає базу даних, інтерфейси користувача, модулі для управління даними, а також механізми забезпечення захисту та резервування інформації.

Таким чином, у межах розділу визначено концептуальні основи розробки системи обліку товарів для інтернет-магазину, які сприяють її ефективному впровадженню та подальшій експлуатації.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ ОБЛІКУ ТОВАРІВ МАГАЗИНУ ПОБУТОВОЇ ХІМІЇ

3.1 Розробка бази даних для інформаційної системи обліку товарів магазину побутової хімії

Оскільки проєкт інформаційної системи обліку товарів магазину побутової хімії було вирішено реалізовувати з використанням пакету Voyager на базі PHP фреймворку Laravel 9, то типовим рішенням є використання СУБД MySQL/MariaDB разом з пакетом адміністрування на базі phpMySQL.

Варто відзначити, що пакет Voyager при розгортанні застосовує цілий набір міграцій (рис. 3.1), які необхідні для створення таблиць, які забезпечують аутентифікацію та авторизацію користувачів.

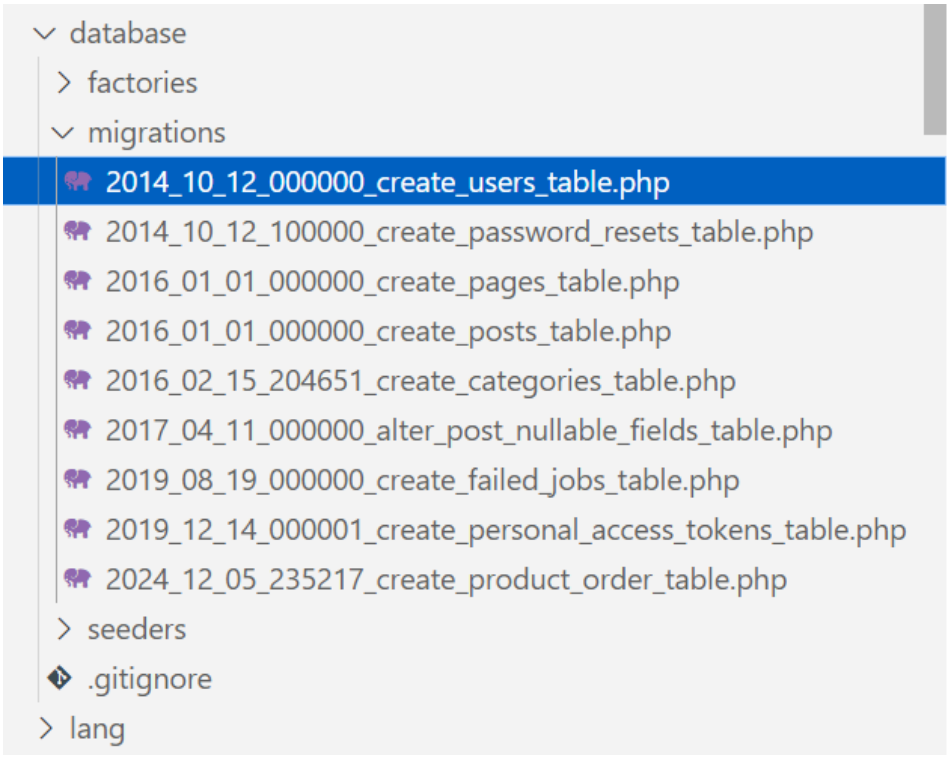


Рисунок 3.1 – Структура папки з підготовленими міграціями

Основними таблицями, відповідальними за аутентифікацію та авторизацію, є:

– roles – містить інформацію про зареєстровані ролі користувачів;

– users – містить інформацію про зареєстрованих користувачів;

– permissions – містить інформацію про зареєстровані дозволи на доступ до певних ресурсів програми;

– permission_role – зведена таблиця, що містить інформацію про зареєстровані для певної ролі дозволи на доступ до певних ресурсів програми;

– user_roles – зведена таблиця, що містить інформацію про участь певного користувача у існуючих ролях.

На рис. 3.2 наведено структуру та зв’язки між основними таблицями, що забезпечують аутентифікацію та авторизацію користувачів.

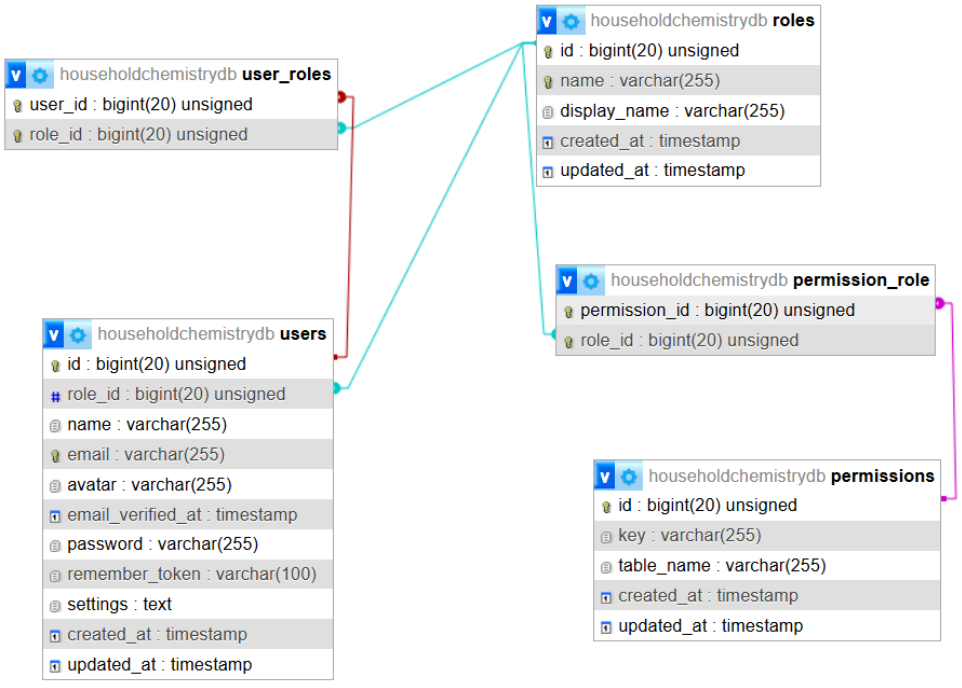


Рисунок 3.2 – Структура та зв’язки між основними таблицями, що забезпечують аутентифікацію та авторизацію користувачів

На рис. 3.3 наведено структуру та зв’язки між основними сутностями, які приймають участь у варіантах використання інформаційної системи обліку товарів магазину побутової хімії, пов’язаних із замовленням товарів у постачальників. Зокрема, відображено таблиці «Бренди» (brands), «Постачальники» (suppliers), «Товари» (products), «Замовлення» (orders), та таблиця-зв’язок product_order.

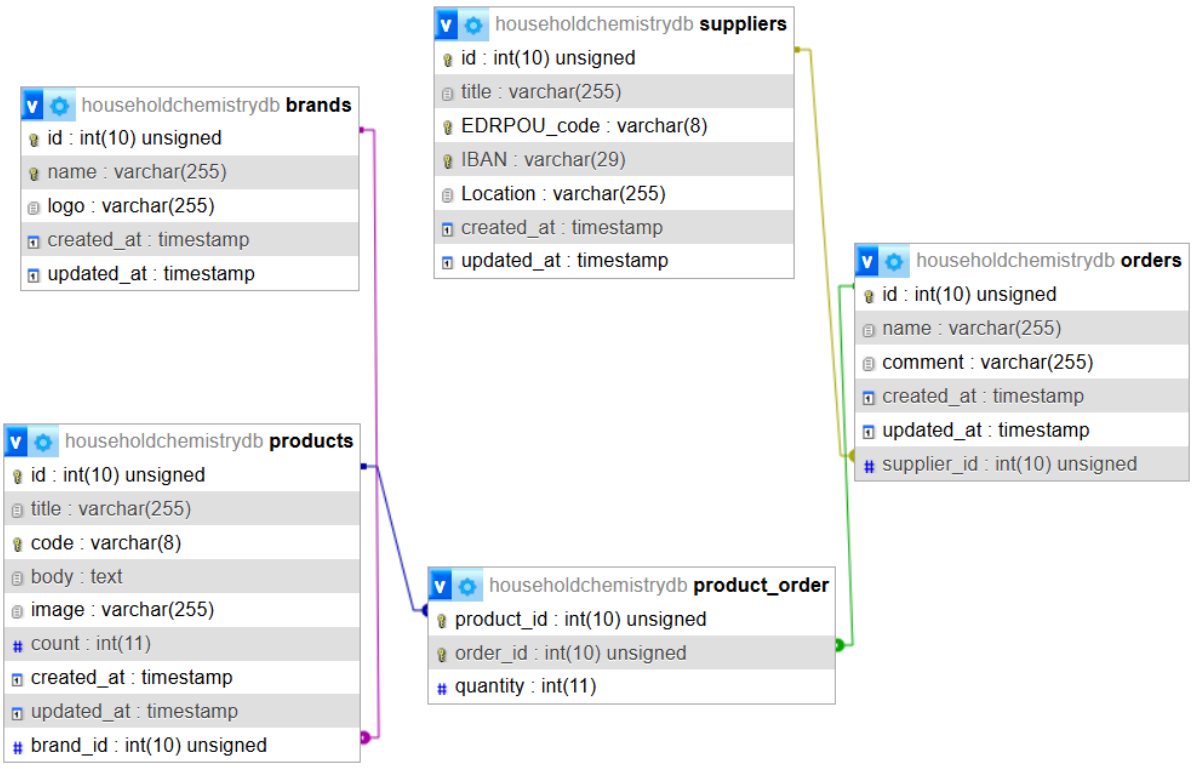


Рисунок 3.3 – Структура та зв’язки між основними таблицями, що забезпечують збереження інформації про замовлення товарів у постачальників

На рис. 3.4 наведено перелік таблиць бази даних householdchemistrydb для розробленого проєкту інформаційної системи обліку товарів магазину побутової хімії, що відображається у пакету адміністрування phpMyAdmin.

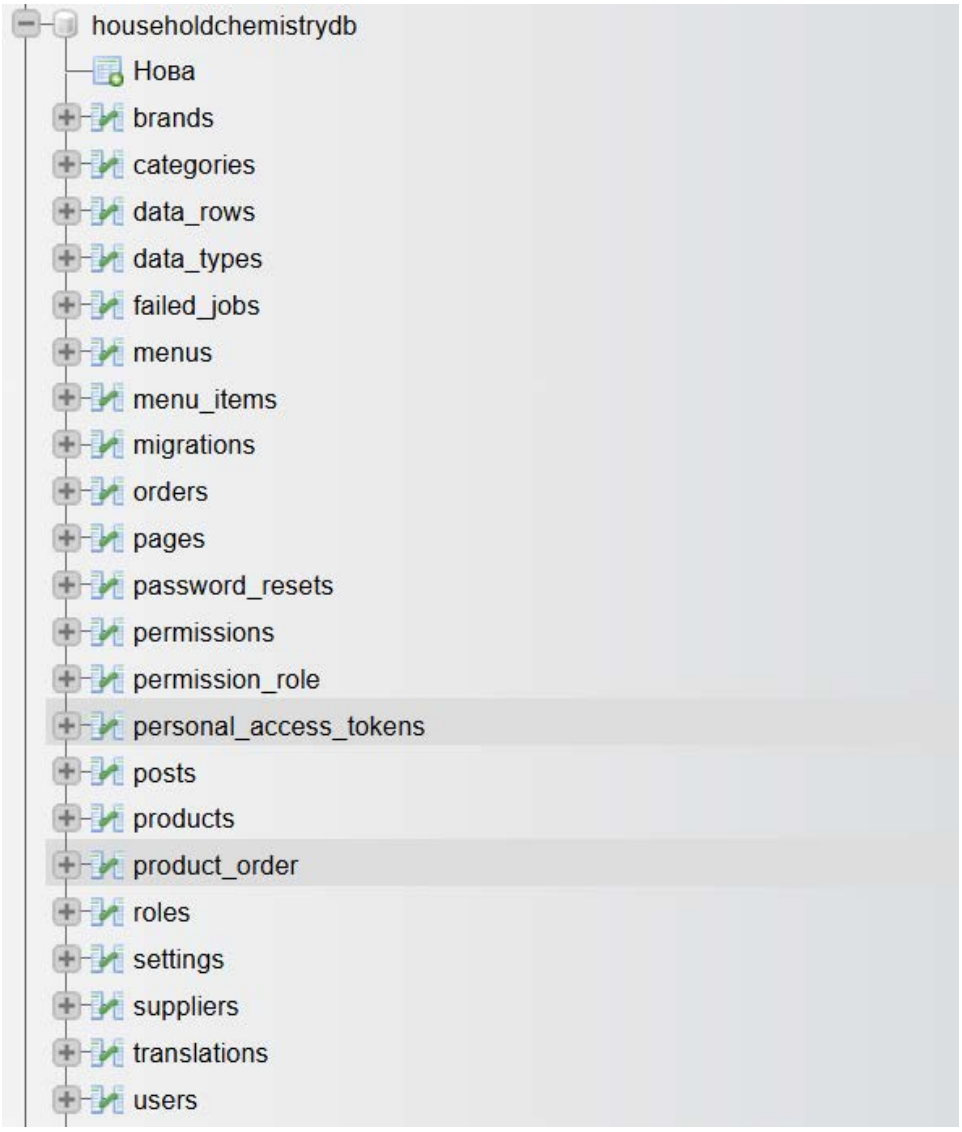


Рисунок 3.4 – Перелік таблиць бази даних householdchemistrydb для розробленого проєкту інформаційної системи обліку товарів магазину побутової хімії

Таким чином, для реалізації збереження інформації було вирішено використати СУБД MySQL. На першому етапі розробка відбувалась локально і використовувався локальний сервер MySQL, що входить у пакет XAMPP для розробки на PHP.

На наступних етапах розробки, оскільки використовувалась система версій Git та розробка відбувалась на декількох комп’ютерах, то було вирішено розгорнути БД на хостінгу.

Аналіз наявних варіантів показав, що найкращим варіантом є сервіс Aiven.io, який надає безкоштовний тестовий аккаунт з великим обсягом місця на сервері БД, а також пропонує зручний пакет aiven-laravel для інтеграції у проекти на фреймворку Laravel.

3.2 Проєктування інформаційної системи обліку товарів магазину побутової хімії

Для реалізації інформаційної системи обліку товарів для магазину побутової хімії було вирішено використати фреймворк Laravel з пакетом Voyager для реалізації системи керування контентом (CMS).

Першим етапом реалізації системи є розробка бази даних. Для цього у шаблоні проєкту на Voyager є пункт меню «Tools», яке містить підпункт «Database», при виборі якогоVoyager дозволяє налаштувати все необхідне для створення та визначення полів таблиці БД (рис. 3.5), зокрема:

– назву поля (Name);

– тип даних (Type);

– довжину поля (Length);

– задання обмеження на те, що поле не може містити значення null (Not null);

– визначення, що поле зберігає дані цілого типу без знаку (Unsigned);

– визначення, що поле повинно автоматично збільшуватися (інкрементуватися) при вставці (Auto Increment);

– тип індексу (Index);

– значення за замовчуванням (Default)

Великою перевагою пакету Voyager можна вважати підтримку концепції так званих BREAD, яка полягає у підтримці наступних операцій:

– Browse – відображення даних під час перегляду;

– Read – читання даних;

– Edit – відображення полів з даними та можливість їх редагування;

– Add – відображення полів при додаванні нових даних;

– Delete – можливість видалення.

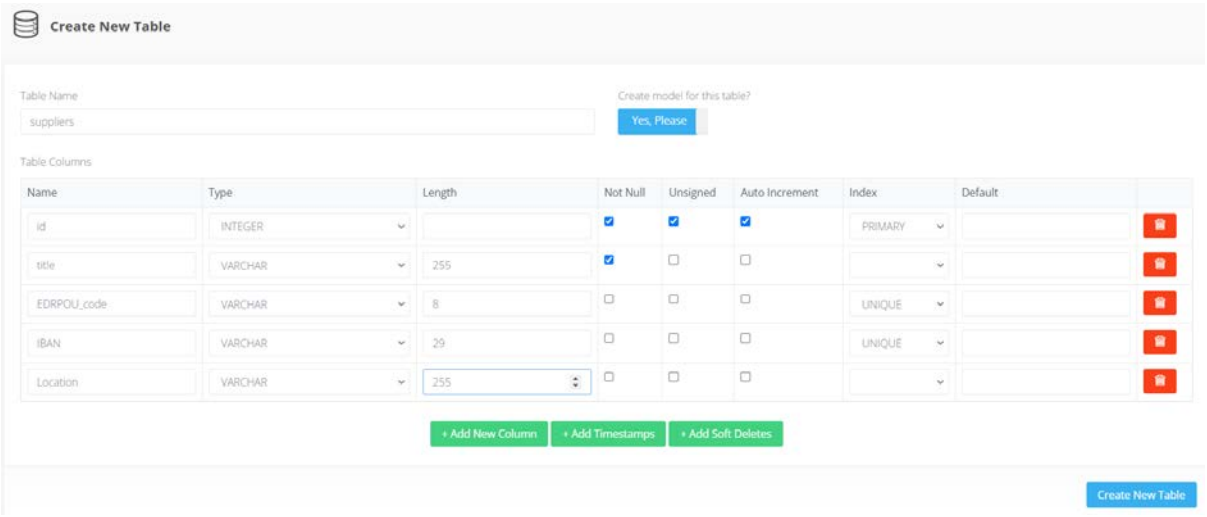


Рисунок 3.5 – Сторінка створення таблиці «Постачальники»

У шаблоні проєкту на Voyager є спеціальний інструмент BREAD (рис. 3.5) у меню «Tools», з використанням якого можна додавати шаблони BREAD для конкретних таблиць (або сутностей предметної області).

Рисунок 3.6 – Сторінка відображення інструменту BREAD для проєкту

На рис. 3.7 наведено сторінка додавання нового BREAD для таблиці постачальників.

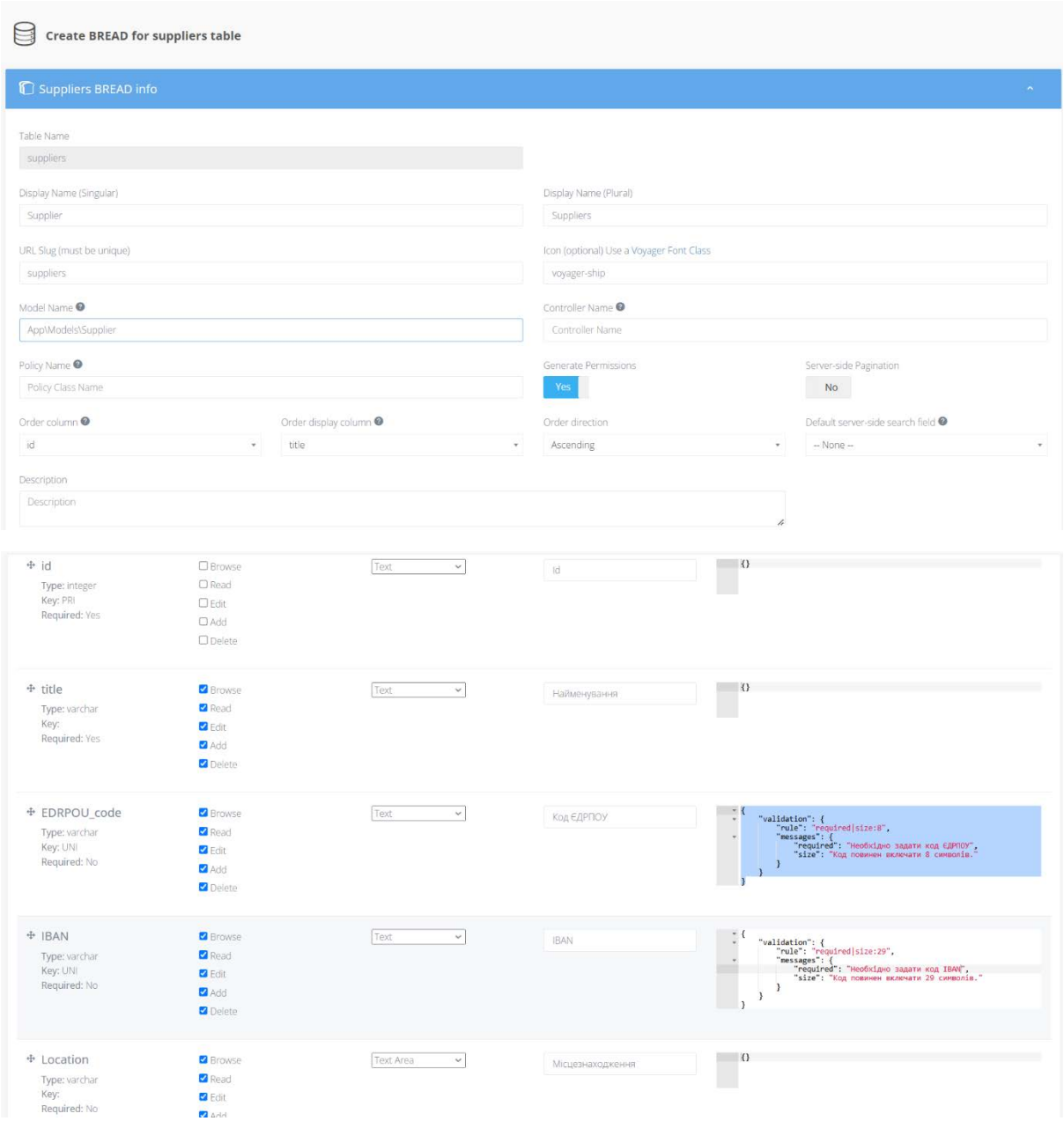


Рисунок 3.7 – Сторінка налаштування шаблонів для CRUD-операцій над об’єктами у таблиці «Постачальники»

Як видно з рисунку, при для створення сторінок керування сутністю «Постачальник» задаються наступні основні параметри:

– назва таблиці (Table name);

– назва класу моделі (Model name), вказується разом з простором імен у проєкті Laravel;

– назва поля для сортування (Order column);

– назва поля для відображення (Order display column);

– напрямок сортування (Order direction);

– іконка для відображення у меню та на сторінці керування сутностями (Icon).

Друга частина вікна містить на рис. 3.7 містить секцію «Edit the rows for the table…», де можна налаштувати: доступність для різних типів CRUD-операцій (Visibility); типи полів введення для форм створення та редагування (Input type); мітку для підпису відповідного поля (Display Name); додаткові налаштування (Optional Details). Додаткові налаштування визначаються у форматі JSON. Так, зокрема, на рис. 3.7 для таблиці «Постачальники» вказано додаткові налаштування для двох полів – коду ЄДРПОУ (EDRPOU_code) та міжнародного банківського номеру (IBAN).

Для обох полів задано правило валідації (rule) у вигляді required|size:8. Дане правило визначає, що поле є обов’язковим для заповнення та повинно складатися з 8 символів. У полі messages об’єкту validation визначено об’єкт, поля якого відповідають правилам валідації (required, size), а у якості значень визначено повідомлення, що будуть відображатися при виникненні помилок валідації даних у відповідних полях.

На рис. 3.8 наведено результати тестування створеної з використанням майстру BREAD сторінки для додавання нового постачальника. Як видно, оскільки для поля «Місцезнаходження» (Location) при створенні шаблону сторінки було вказано тип поля введення «Text area», то на формі дане поле відображається у вигляд елементу textarea HTML, і дозволяє вводити багаторядковий текст.

При введенні некоректних даних у поля, для яких визначені правила валідації, повідомлення з описом помилки виводиться як у блоці підсумку

вгорі сторінки, так і під конкретним полем. Так, на рис. 3.9 показано помилку валідації, викликану невідповідністю довжини введеного міжнародного банківського коду IBAN потрібній довжини (29 символів).

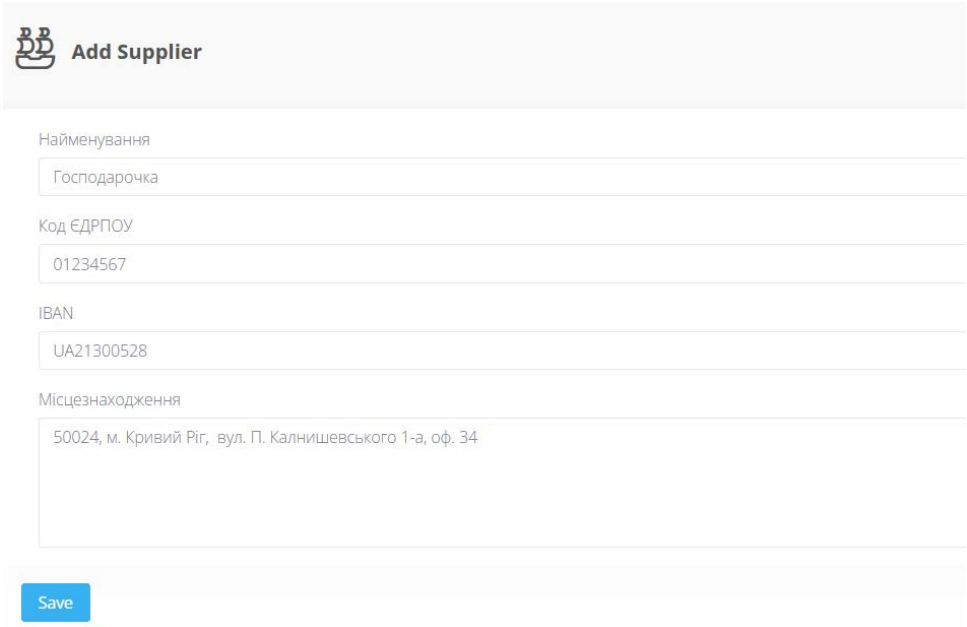


Рисунок 3.8 – Сторінка додавання нового постачальника

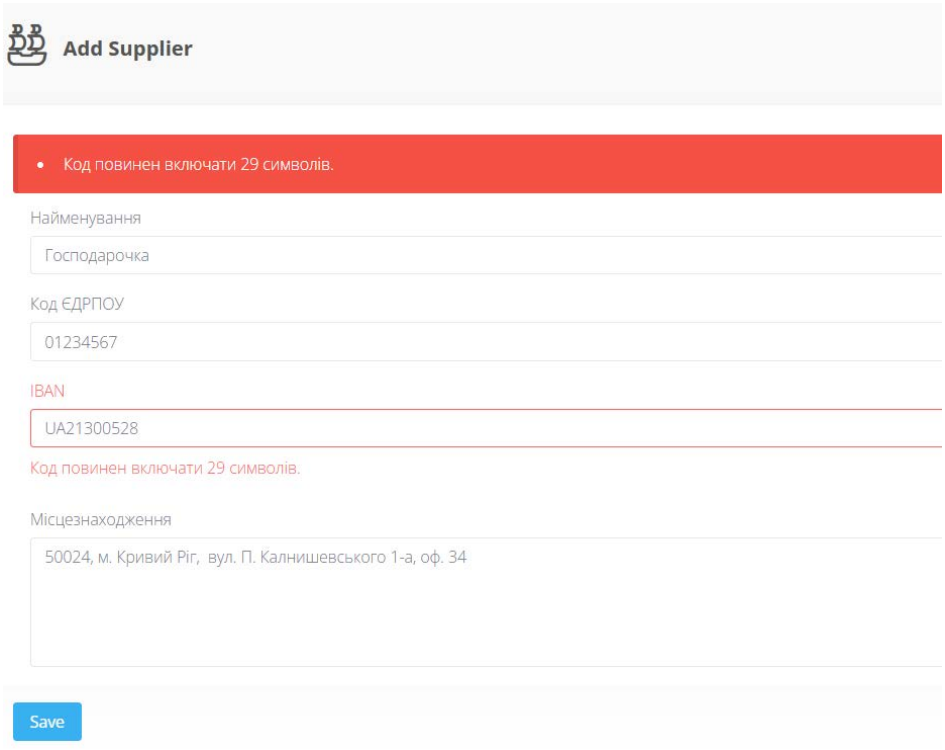


Рисунок 3.9 – Відображення помилок валідації на сторінці додавання нового постачальника

На рис. 3.10 наведено сторінку відображення доданих постачальників. Як видно, на сторінці можна переглянути загальну інформацію по постачальникам, переглянути детальну інформацію по кожному постачальнику (кнопка View), відредагувати (кнопка Edit) та, у разі необхідності, видалити конкретного постачальника (кнопка Delete). Крім того, можна додати нового постачальника (кнопка Add New), виконати операцію групового видалення (кнопка Bulk Delete), та змінити порядок сортування (кнопка Order).

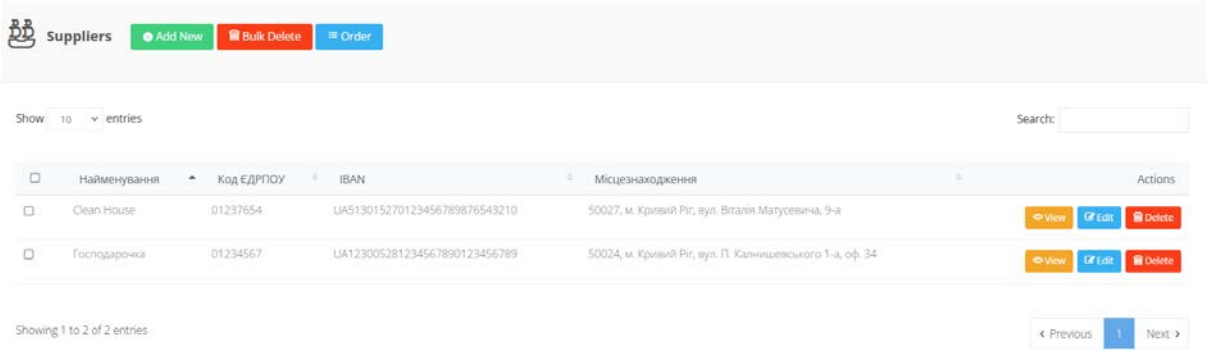


Рисунок 3.10 – Сторінка відображення списку постачальників

На наступному етапі з використанням вбудованих інструментальних сторінок пакету Voyager було додано таблицю «Бренди» (brands) (рис. 3.11).

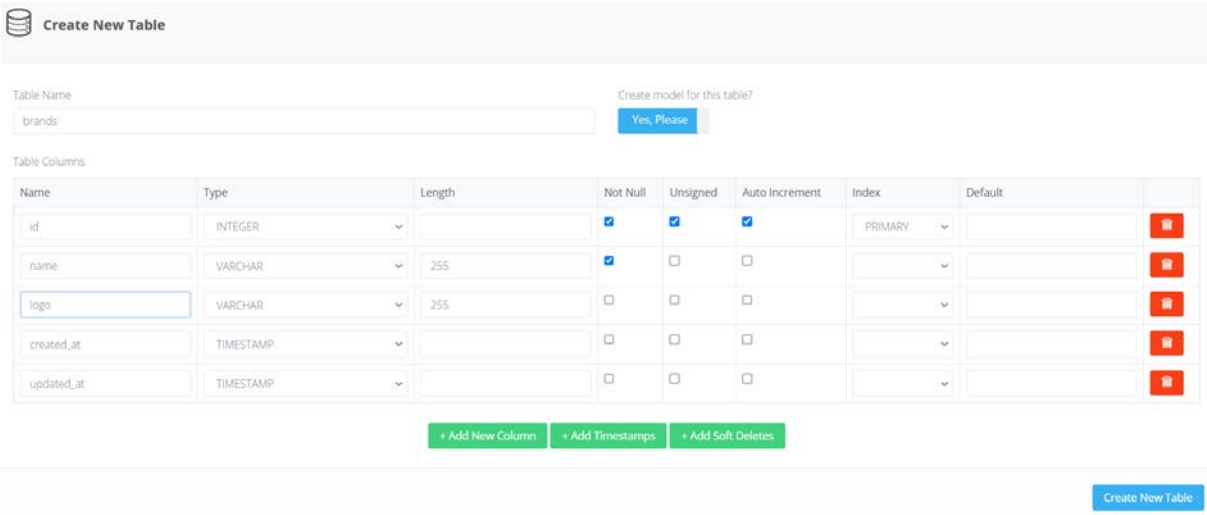


Рисунок 3.11 – Сторінка створення таблиці «Бренди»

Для зручного візуального пошуку бренду було вирішено додати можливість зберігати його логотип. Тому при створенні таблиці додано поле «logo» з типом даних Varchar та довжиною 255 символів. Даний тип використано тому, що Voyager зберігає ассети (статичні ресурси, такі, як картинки, іконки, відео) у каталозі на сервері, тож достатньо зберігати відносний шлях до відповідного ассета.

При додаванні шаблонів сторінок для виконання CRUD-операцій над записами про бренди з використанням інструменту BREAD також було виконано налаштування основних полів, з використанням яких буде здійснюватися додавання та редагування даних в таблиці «Бренди» (рис. 3.12). Для поля первинного ключа id встановлено в якості типу поля (Input type) значення «hidden», що призведе до створення на формі елементу типу <input type=`hidden`/>, що буде зберігати значення первинного ключа. Але для користувача це значення буде приховано.

Для поля «logo» встановлено тип (Input type) «image», що призведе до створення на формі елементу типу <input type=`file`/>, яке буде відкривати діалог вибору файлів і дозволить відправляти обраний файл на сервер.

У даному випадку не задавалися правила валідації, оскільки для поля name вони були встановлені при створенні таблиці БД.

На рис. 3.13 наведено створену форму додавання нового бренду, яка містить поле для введення назви бренду та поле для вибору файлу логотипу бренду. Варто відзначити, що на формі відображається іконка, яка була вказана у полі Icon при налаштуванні шаблонів BREAD. Варто відзначити, що пакет Voyager містить достатньо великий набір готових іконок, які можна використовувати за їх коротким ім’ям. Дана іконка також відображається поряд з відповідним пунктом головного меню адміністратора, що створюється автоматичного при додаванні нового BREAD.

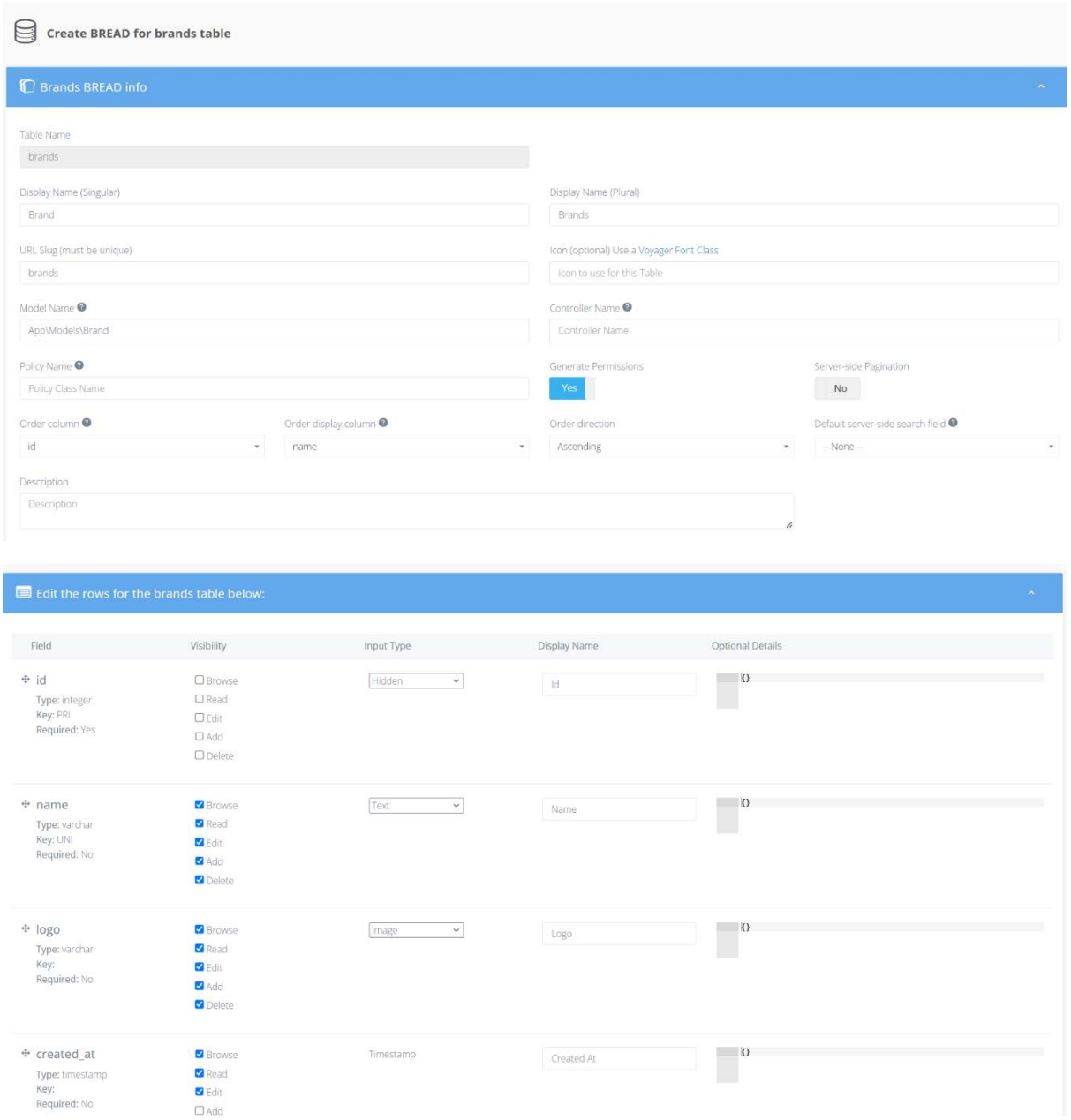


Рисунок 3.12 – Сторінка налаштування шаблонів для CRUD-операцій над об’єктами у таблиці «Бренди»

При використанні пакету Voyager у вбудованій панелі керування є спеціальний менеджер мультимедіа (Media Manager), який дозволяє зручно керувати зображеннями, що завантажуються на сервер. Зокрема, можна налаштувати бажані розміри зображень, і при завантаженні вони будуть автоматично масштабуватися до потрібних розмірів, що є дуже ефективним з точки зору використання дискового простору на сервері.

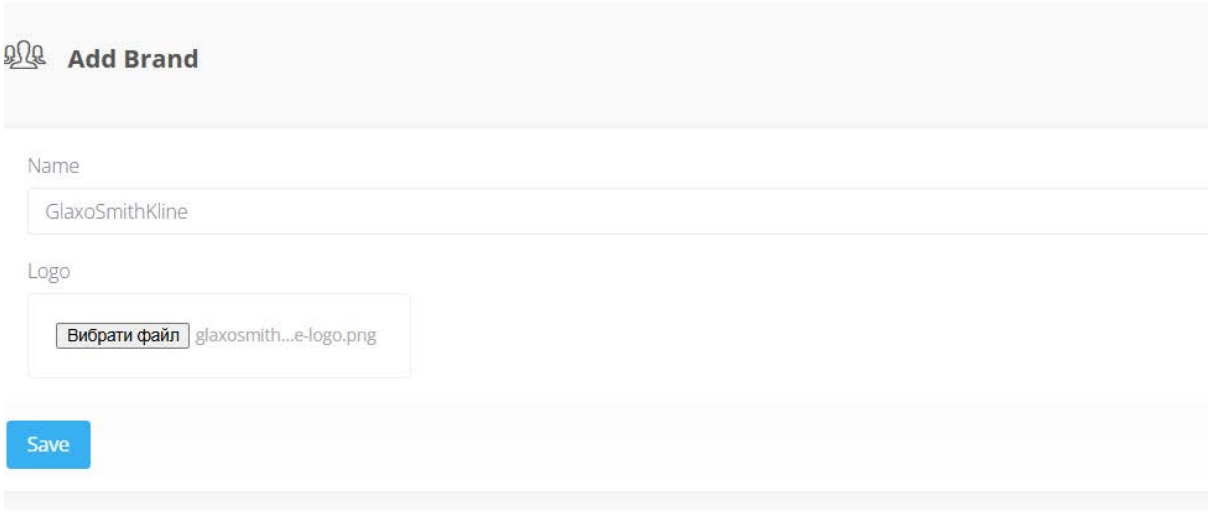


Рисунок 3.13 – Сторінка додавання нового бренду

На рис. 3.14 наведено сторінку відображення доданих брендів. Як видно, на сторінці можна переглянути загальну інформацію по брендам, включно з логотипом. Варто відзначити, що всі створені за допомогою інструментів BREAD за замовчуванням мають кнопки пагінації, яку доступна «з коробки» і не потребує окремого встановлення, налаштування та реалізації.

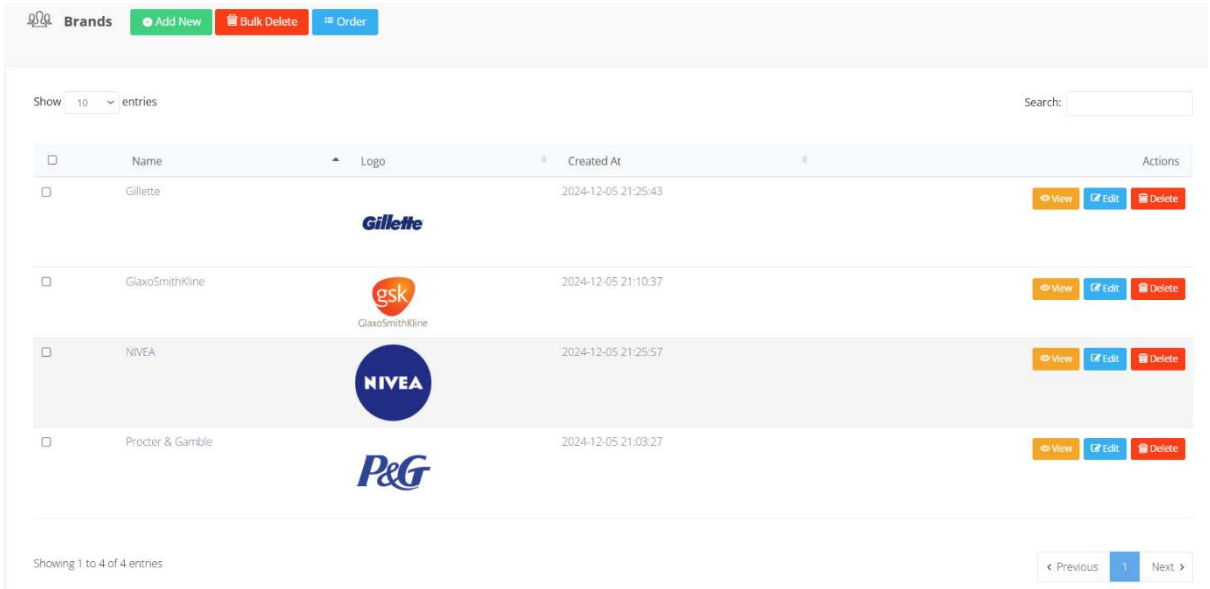


Рисунок 3.14 – Сторінка відображення списку брендів

На наступному кроці було розроблено та налаштовано структуру таблиці з описом товарів (products). Окрім назви товару (title), було додано наступні поля (рис. 3.15):

– code – код для відображення його користувачу, для якого також вказано індекс UNIQUE для забезпечення унікальності значень даного поля;

– body – поле для опису характеристик товару з типом TEXT, що може містити до 64 КБ текстових даних;

– image – поле для збереження відносного шляху до зображення товару з типом VARCHAR;

– count – поле для збереження кількості товарів, тип INTEGER;

– created_at, updated_at – так звані поля позначок часу (timestamps), які широко використовуються у Laravel, Eloquent та MySQL для збереження часу створення та останньої модифікації даних відповідного запису.

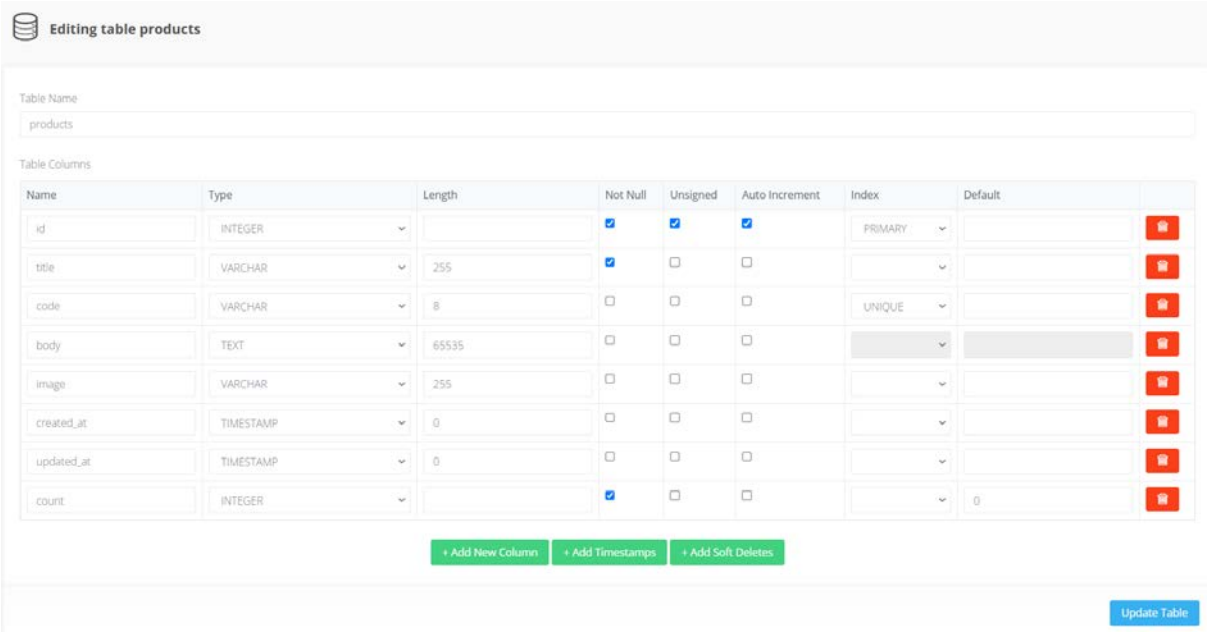


Рисунок 3.15 – Сторінка створення таблиці «Продукти»

Варто відзначити, що через особливості налаштування відношень між таблицями у Voyager, на цьому етапі не було додано поле зовнішнього ключа

brand_id для зв’язку з таблицею брендів. Даний крок буде виконано пізніше, після додавання шаблонів сторінок для виконання CRUD-операцій за допомогої інструментів BREAD.

На рис. 3.16 наведено форму для налаштування шаблонів сторінок для виконання CRUD-операцій над записами про товари з використанням інструменту BREAD.

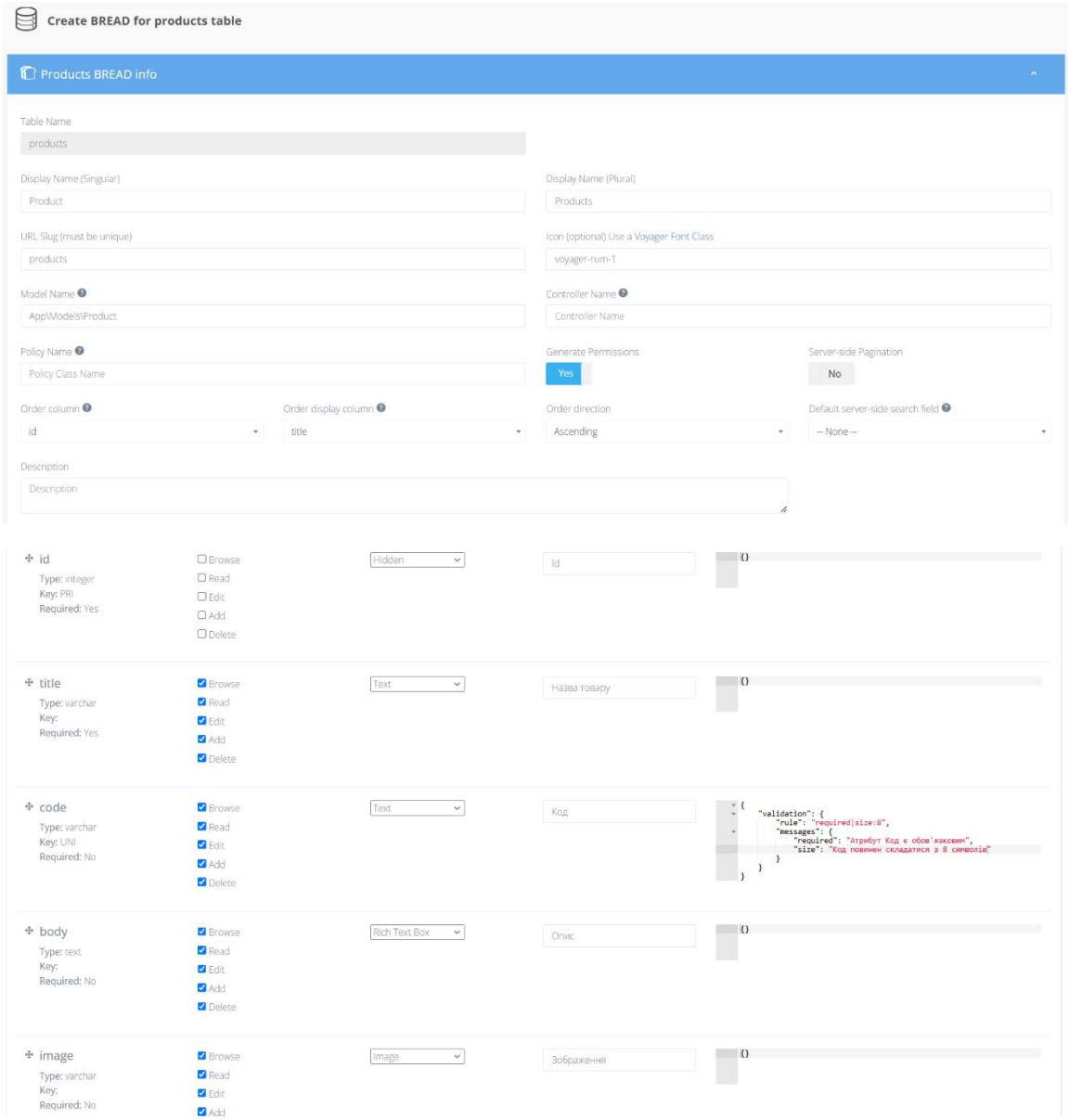


Рисунок 3.16 – Сторінка налаштування шаблонів для CRUD-операцій над об’єктами у таблиці «Продукти»

Для поля первинного ключа id встановлено в якості типу поля (Input type) значення «hidden», що призведе до створення на формі елементу типу <input type=`hidden`/>, що буде зберігати значення первинного ключа. Але для користувача це значення буде приховано.

Варто звернути увагу на те, що для поля «body», призначеного для опису основних характеристик товару, задано значення типу поля «Rich Text Box». При такому налаштуванні Voyager створює для відповідного поля дуже популярний у Web багаторядковий редактор TinyMCE, за допомогою якого можна налаштовувати контент, використовувати структурні елементи HTML, додавати картинки тощо. Фактично таке рішення з коробки дозволяє як завгодно ефектно оформлювати опис товарів.

Для поля «image» встановлено тип (Input type) «image», що призведе до створення на формі елементу типу <input type=`file`/>, яке буде відкривати діалог вибору файлів і дозволить відправляти обраний файл на сервер.

Для поля «code» задано правило валідації (rule) у вигляді required|size:8. Дане правило визначає, що поле є обов’язковим для заповнення та повинно складатися з 8 символів. У полі messages об’єкту validation визначено об’єкт, поля якого відповідають правилам валідації (required, size), а у якості значень визначено повідомлення, що будуть відображатися при виникненні помилок валідації даних у відповідних полях (у даному випадку «Атрибут Код є обов’язковим» для правила валідації «required», та «Код повинен складатися з 8 символів» для правила валідації «size:8»).

На наступному кроці необхідно було налаштувати зв’язок між таблицями «Бренди» та «Продукти». Для цього у таблицю products було додано поле зовнішнього ключа brand_id, яке має посилатися на первинний ключ таблиці brands (рис. 3.17).

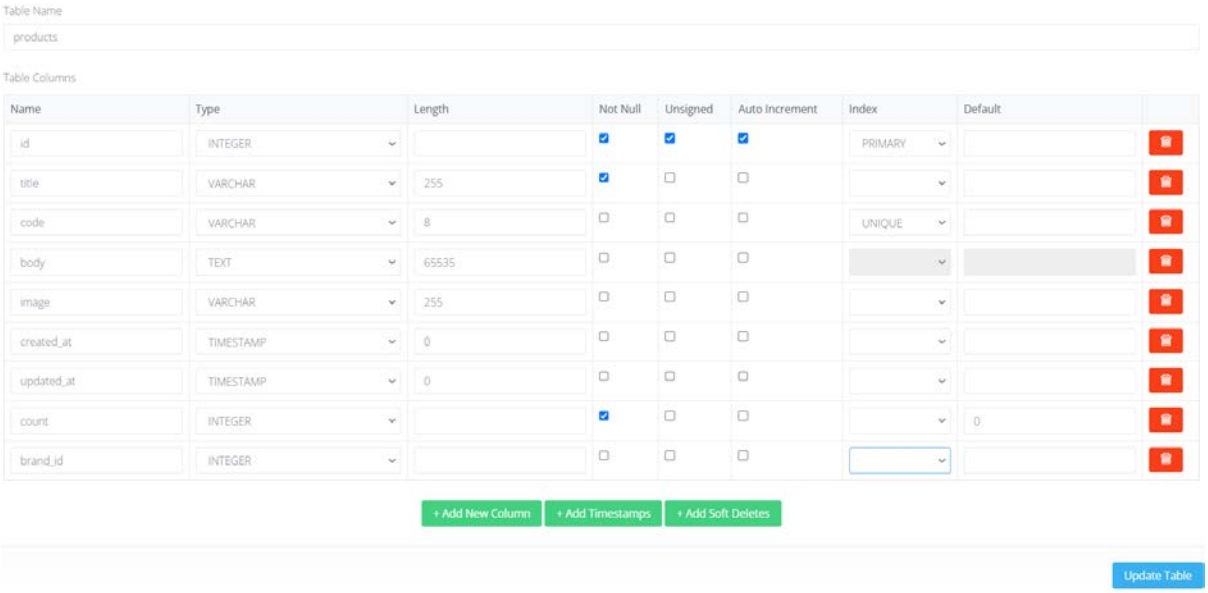


Рисунок 3.17 – Додавання нового поля до таблиці «Продукти»

Варто відзначити, що інструмент редагування таблиць баз даних пакету Voyager безпосередньо не надає можливості налаштовувати відносини між окремими таблицями та визначати поля як зовнішні ключі. Для цього у вікні раніше створеного макету BREAD для відповідної сутності скористатися кнопкою «Create Relationship» для створення нового відношення між таблицями. Причому на етапі створення макету BREAD виклик цієї кнопки призведе до помилки.

На рис. 3.18 наведено вигляд модального вікна для налаштування нового відношення між таблицями. Таблиця продуктів (products) зв’язана з таблицею брендів (brands) як «багато до одного». Тому, використовуючи термінологію ORM-системи Eloquent, потрібно визначити, що сутність «Продукт» (Product) «належить» («Belongs To») сутності «Brand». При визначенні пов’язаної сутності потрібно вказати точну назву класу відповідної незалежної сутності (з вказанням простору імен). Також у полі «Which column from the Product is used to reference the Brand?» необхідно вказати поле зовнішнього ключа в залежній сутності (у нашому випадку це поле brand_id класу Product).

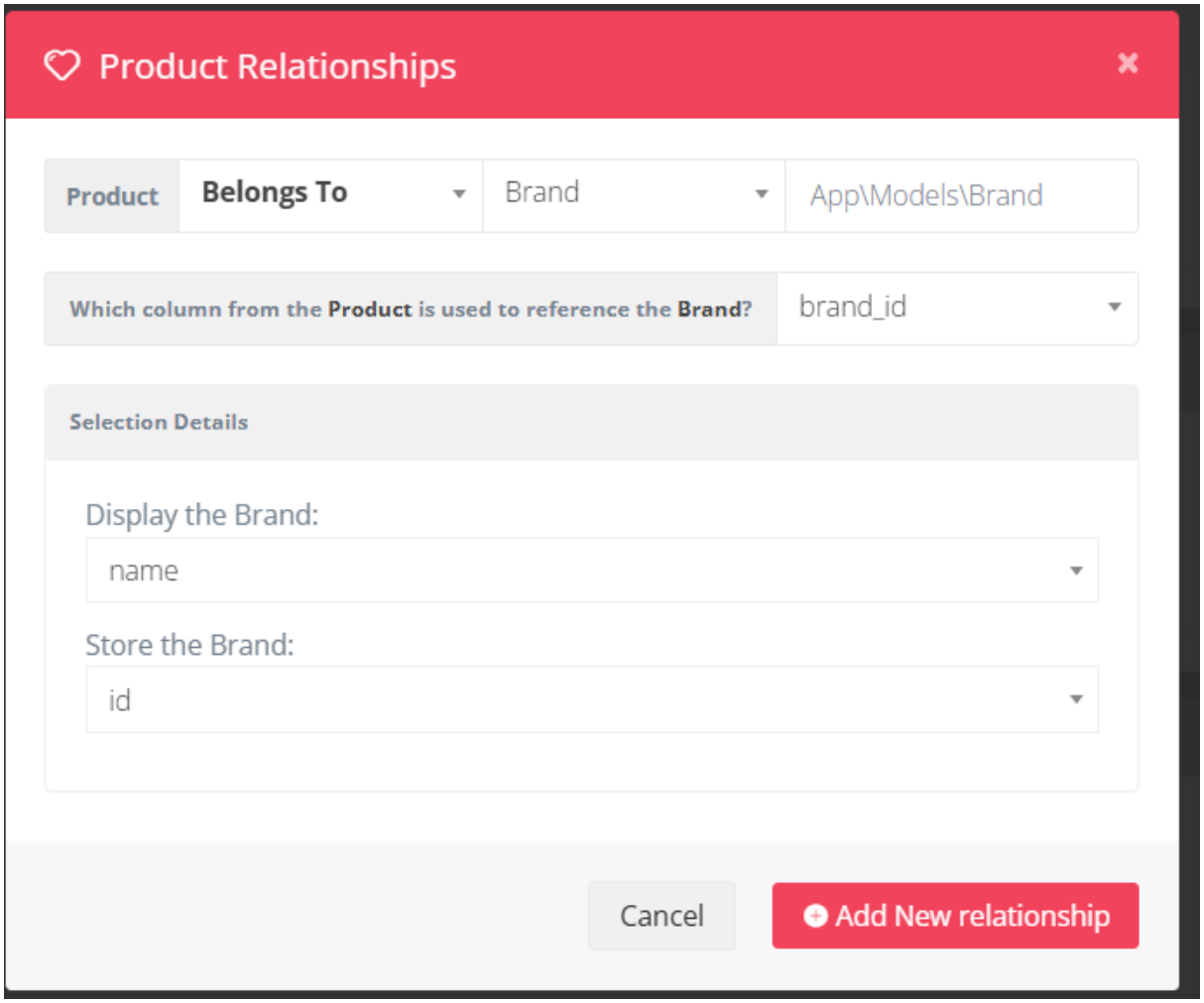


Рисунок 3.19 – Вигляд модального вікна для налаштування відношення між таблицями «Продукти» та «Бренди»

У секції Selection Details можна налаштувати те, як буде відображатися інформація про зв’язок у випадаючих списках (наприклад, у елементах <select/>). У полі «Display the Brand» необхідно вибрати поле, яке буде відображатися всередині елементів випадаючого списку <option/> (у нашому випадку це поле name таблиці brands), а в поле «Store the Brand» - поле, яке буде використовуватися в якості значення атрибуту value (у нашому випадку це поле id). Саме це значення буде передаватися при відправці форми на сервер.

Після виконаних налаштувань потрібно зберегти результати, натиснувши кнопку «Add New relationship».

Після збереження результатів у секції «Edit the rows for the table…» з’явиться опис доданого поля «brands», для якого у колонці «Input type» буде вказано значення «Relationship». У колонці «Display Name» змінимо назву на «Бренд».

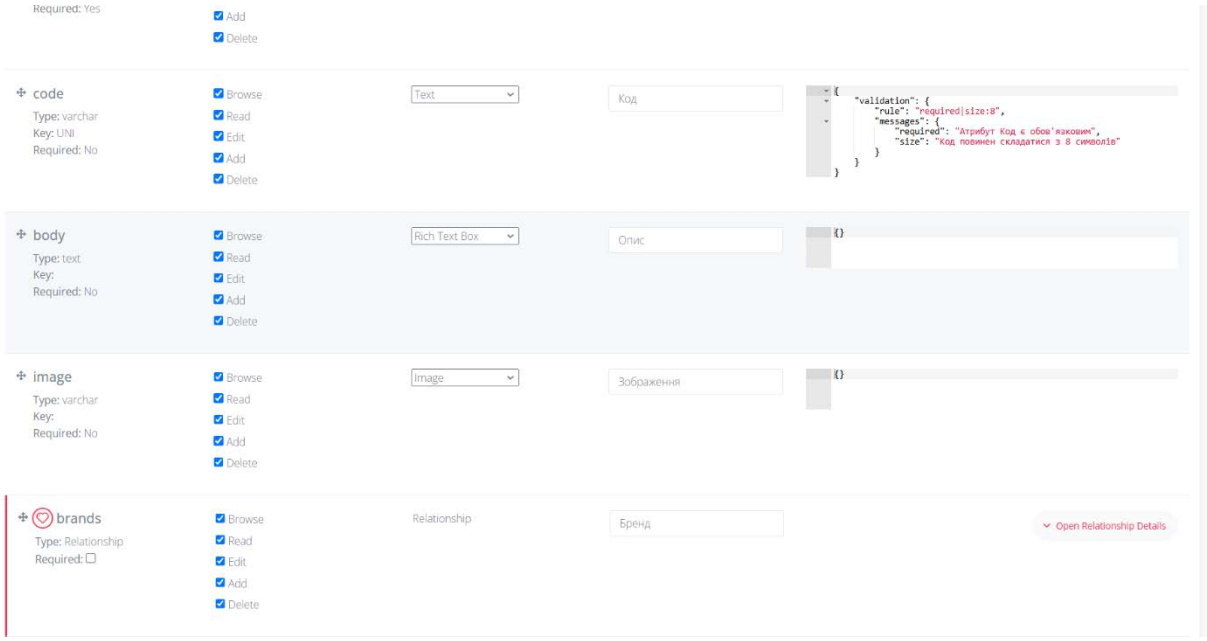


Рисунок 3.19 – Сторінка налаштування шаблонів для CRUD-операцій над об’єктами у таблиці «Продукти» після додавання зв’язку з таблицею «Бренди»

На рис. 3.20 наведено результати тестування створеної з використанням майстру BREAD сторінки для додавання нового товару. Варто звернути на багаторядкове поле «Опис», для побудови якого Voyager використовує плагін у вигляді редактору TinyMCE, який дозволяє форматувати текст та HTML, вставляти таблиці, зображення, посилання на зовнішні ресурси. Таким чином, за допомогою вказаного поля можна створювати описи товарів, розміщуючи фактично будь-який тип контенту, що може відображатися у веб.

У результаті налаштування зв’язку між таблицями «Товари» та «Бренди» на сторінці додавання товару з’явилось поле «Бренд», яке уявляє собою випадний список, що містить увесь перелік брендів.

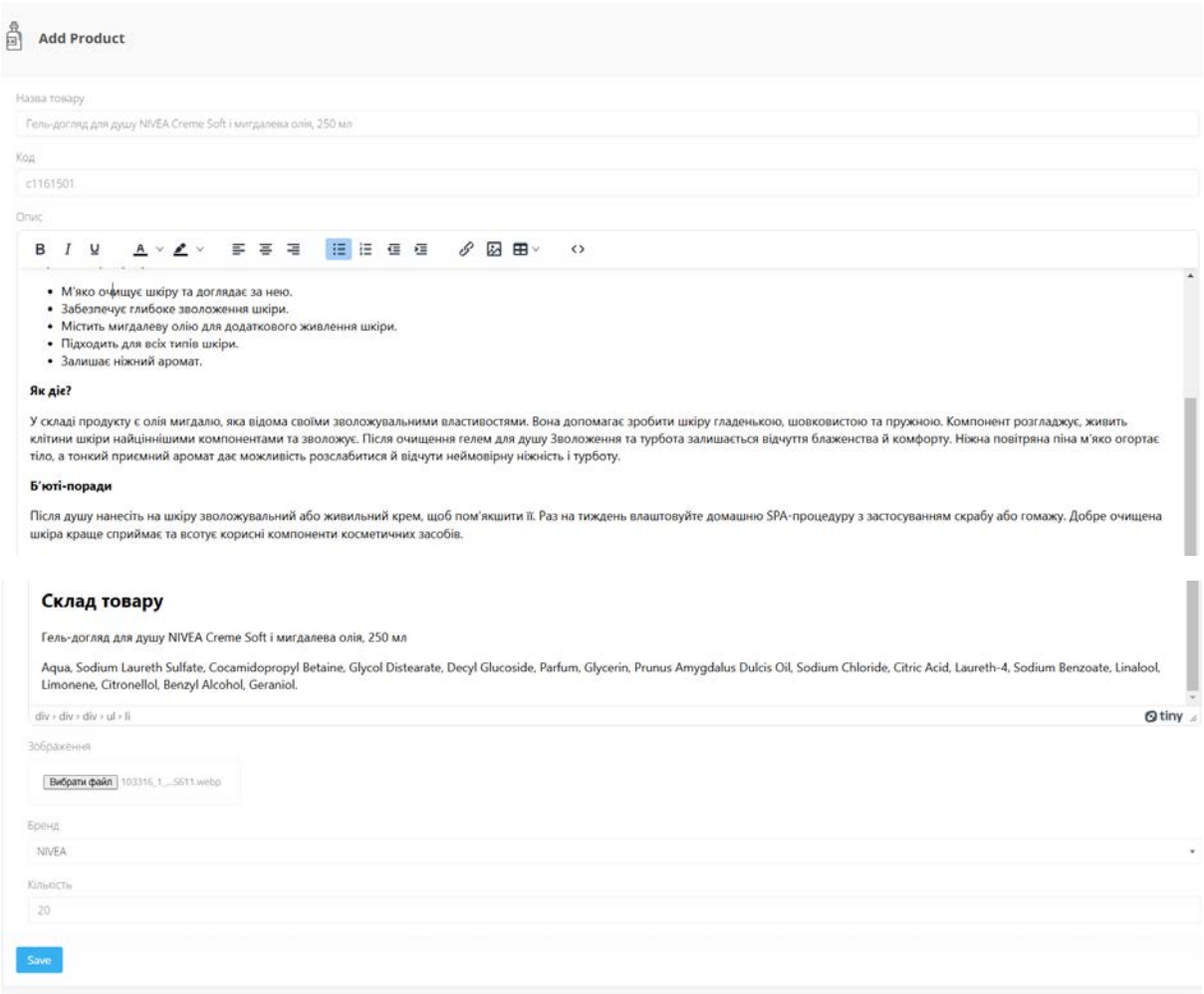


Рисунок 3.20 – Сторінка додавання нового товару

На рис. 3.21 наведено сторінку відображення доданих товарів. Як видно, за рахунок виконаних налаштувань розміру зображень у медіа-менеджері усі завантажені зображення товарів мають однакові розміри. З урахуванням того, що поле «Опис» має тип TEXT, при відображенні за допомогою табличного елементу у представленні відображається лише частина контенту.

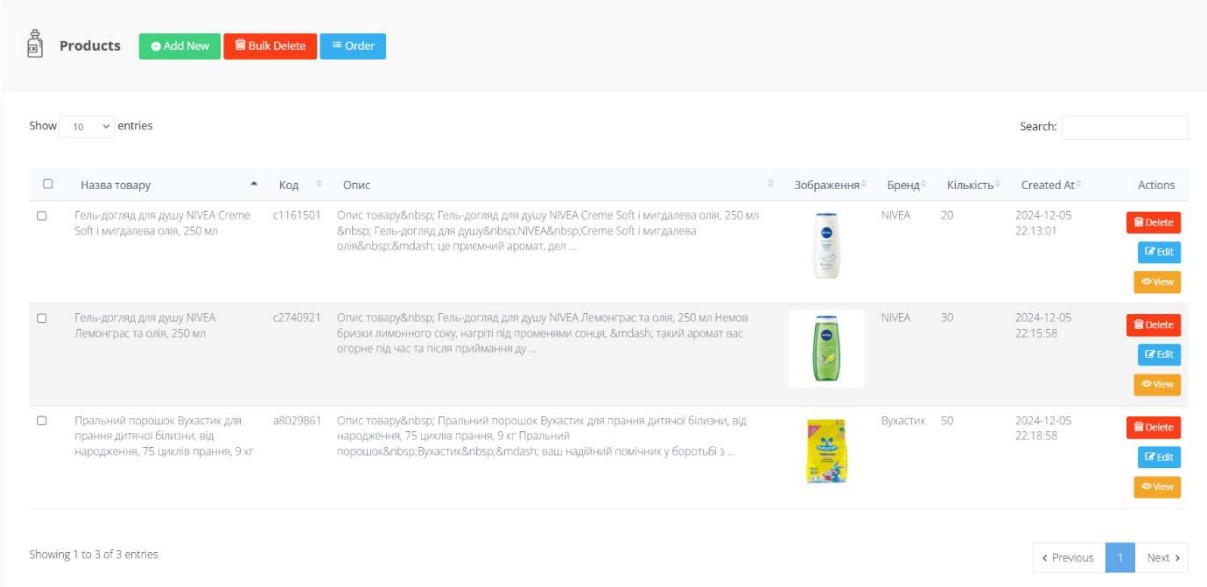


Рисунок 3.21 – Сторінка відображення списку доданих товарів

На наступному кроці потрібно було реалізувати функціонал для забезпечення можливості створювати замовлення у постачальників. Це завдання ускладнено тим, що використовується зв’язок «багато до багатьох» між продуктами та замовленнями, але зведена таблиця (або, як також називають такі таблиці – таблиця-зв’язок, pivot table) повинна мати додаткове поле – кількість товарів у замовленні. Для вирішення цього завдання перед виконанням операцій у вбудованій адміністративній панелі пакету Voyager необхідно також адаптувати класи моделей Laravel, налаштувати особливості зв’язку між сутностями «Замовлення» (order) та «Товар» (product).

Для цього на початковому етапі з використанням вбудованого редактора таблиць БД було розроблено та налаштовано структуру таблиці з описом замовлення (orders). Окрім первинного ключа id, до таблиці було додано наступні поля (рис. 3.18):

– name – коротка назва замовлення для спрощення пошуку оператором при необхідності;

– comment – поле для коментаря стосовно замовлення, типом Varchar

– created_at, updated_at – так звані поля позначок часу (timestamps), які широко використовуються у Laravel, Eloquent та MySQL для збереження часу створення та останньої модифікації даних відповідного запису.

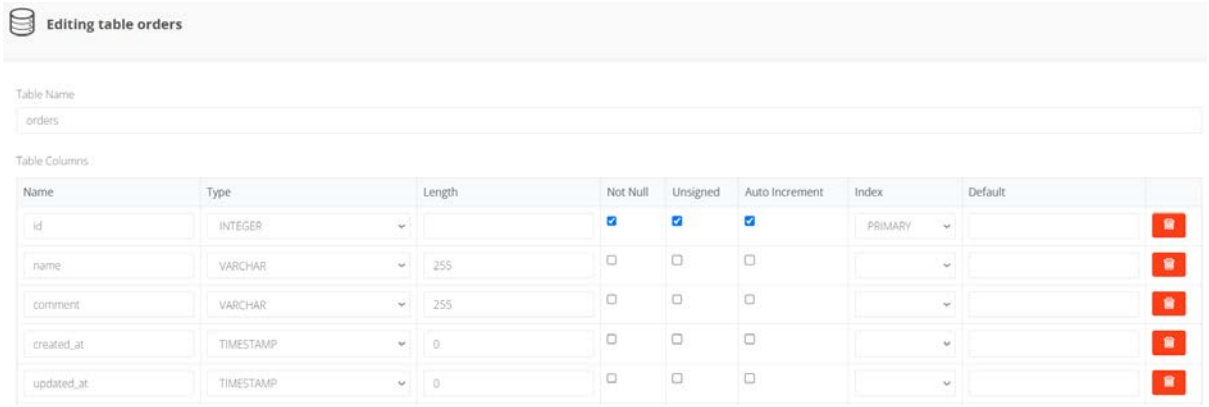


Рисунок 3.22 – Сторінка створення таблиці «Продукти»

На наступному кроці було створено шаблон BREAD для таблиці замовлень (orders). На даному кроці були використані всі стандартні параметри та налаштування.

Після додавання шаблону BREAD необхідно було реалізувати зв’язок замовлення з конкретним постачальником. Тому з використанням редактора таблиць БД було додано поле зовнішнього ключа supplier_id (рис. 3.23).

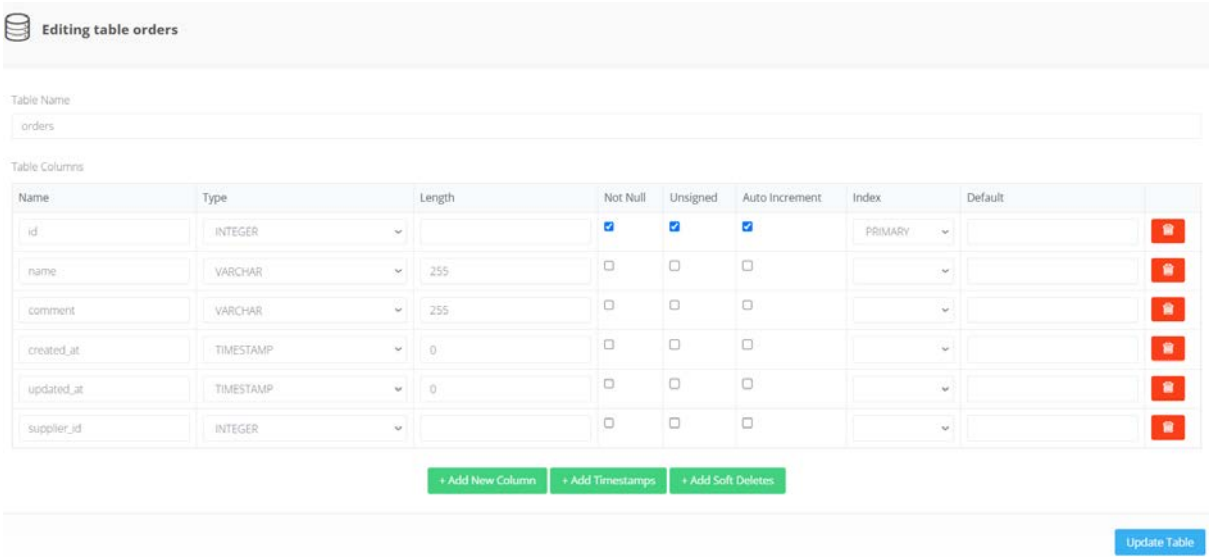


Рисунок 3.23 – Додавання нового поля до таблиці «Замовлення»

Після додавання поля зовнішнього ключа до таблиці «product_order» було відкрито раніше створений BREAD у режимі редагування, після чого додано нове відношення за допомогою кнопки «Create Relationship».

На рис. 3.24 наведено зовнішній вигляд налаштованого відношення між таблицями «Замовлення» (orders) та «Постачальники» (suppliers) на сторінці редагування шаблону BREAD. Таблиця «Замовлення» (orders) пов’язана з таблицею «Постачальники» (suppliers)як «багато до одного». Тому, використовуючи термінологію ORM-системи Eloquent, потрібно визначити, що сутність «Замовлення» (Order) «належить» («Belongs To») сутності «Supplier». При визначенні пов’язаної сутності потрібно вказати точну назву класу відповідної незалежної сутності (з вказанням простору імен). Також у полі «Which column from the Product is used to reference the Brand?» необхідно вказати поле зовнішнього ключа в залежній сутності (у нашому випадку це поле supplier_id класу Order).

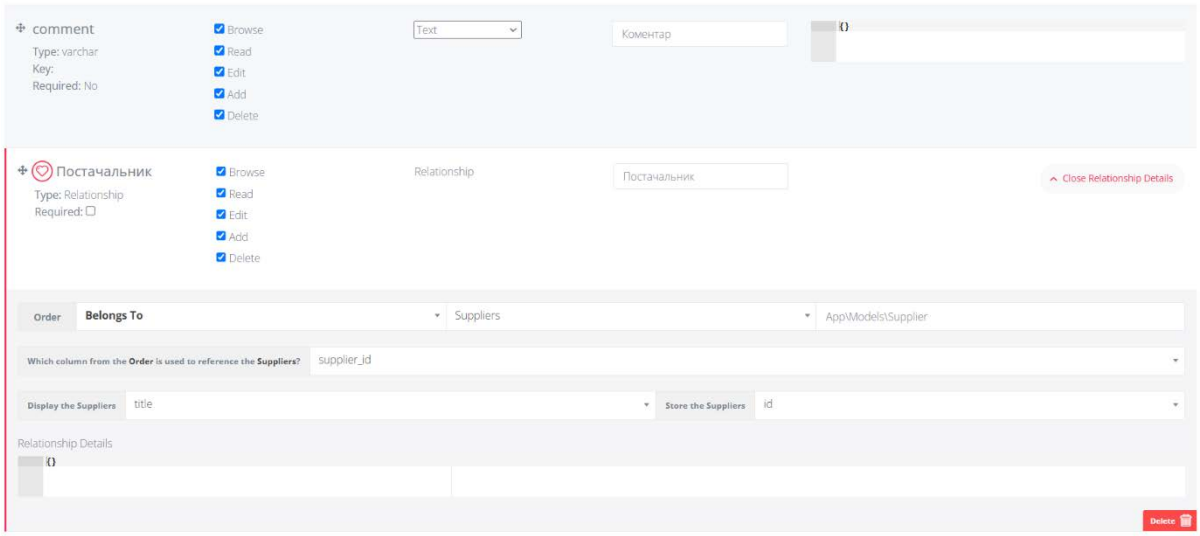


Рисунок 3.24 – Додавання нового відношення до таблиці «Замовлення»

На наступному кроці необхідно налаштувати програмні моделі сутностей предметної області у Laravel для реалізації зв’язку «багато до багатьох».

Для цього необхідно створити файл з підготовленою міграцією для додавання зведеної таблиці. Для об’єкту $table класу Blueprint за допомогою методу foreignId() налаштовано складений первинний ключ, що вказує на первинні ключі в таблицях products та orders. На рис. 3.25 наведено код підготовленого класу міграцій для створення зведеної таблиці «product_order».

database > migrations > 2024_12_05_235217_create_product_order_table.php > class > dc

```php
<?php
use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;
return new class extends Migration
{
    /**
     * Run the migrations.
     * @return void
     */
    public function up(): void
    {
        Schema::create(table: 'product_order',
        callback: function (Blueprint $table): void {
            $table->foreignId(column: "product_id");
            $table->foreignId(column: "order_id");
            $table->integer(column: "quantity")->default(value: 0);
        });
    }

    /**
     * Reverse the migrations.
     * @return void
     */
    public function down(): void
    {
        Schema::dropIfExists(table: 'product_order');
    }
};
```

Рисунок 3.25 – Код файлу міграції для створення зведеної таблиці, призначеної для збереження інформації про товари у замовленнях

У класах Product та Order у відповідності до підходів, прийнятих у Laravel для реалізації даного типу зв’язку, також необхідно додати функції, що будуть виступати у ролі полів, за допомогою яких буде отримуватися доступ до пов’язаних колекцій сутностей. Так, на рис. 3.26 до класу Product додано метод orders(), у коді якого викликається метод belongsToMany(), якому передається назва пов’язаного зв’язком «багато до багатьох» класу моделі Order. Варто також відзначити конструкцію wherePivot(column: “quantity”), за допомогою якої налаштовується наявність у зведеній таблиці додаткового поля quantity, яке буде містити кількість товарів у відповідному замовленні.

```
class Product extends Model
{
    0 references
    protected $fillable = [
        "title",
        "code",
        "body",
        "image",
        "count",
        "brand_id"
    ];
        /**
         * The roles that belong to the Product
         *
         * @return \Illuminate\Database\Eloquent\Relations\BelongsToMany
         */
        0 references | 0 overrides
        public function orders(): BelongsToMany
        {
            return $this->belongsToMany(related: Order::class)
            ->wherePivot(column: "quantity");
        }
}
```

Рисунок 3.26 – Код файлу моделі Product з функцією, що повертає посилання на всі замовлення цього товару

Відповідні зміни також необхідно зробити у класі Order для реалізації доступу до колекції товарів, що містяться у конкретному замовленні.

На останньому етапі реалізації логіки створення замовлень з можливістю додавання декількох товарів до замовлення необхідно відредагувати раніше створений BREAD для сутності «Замовлення» (Order) та додати відповідне відношення за допомогою кнопки «Create a Relationship» (рис. 3.23).

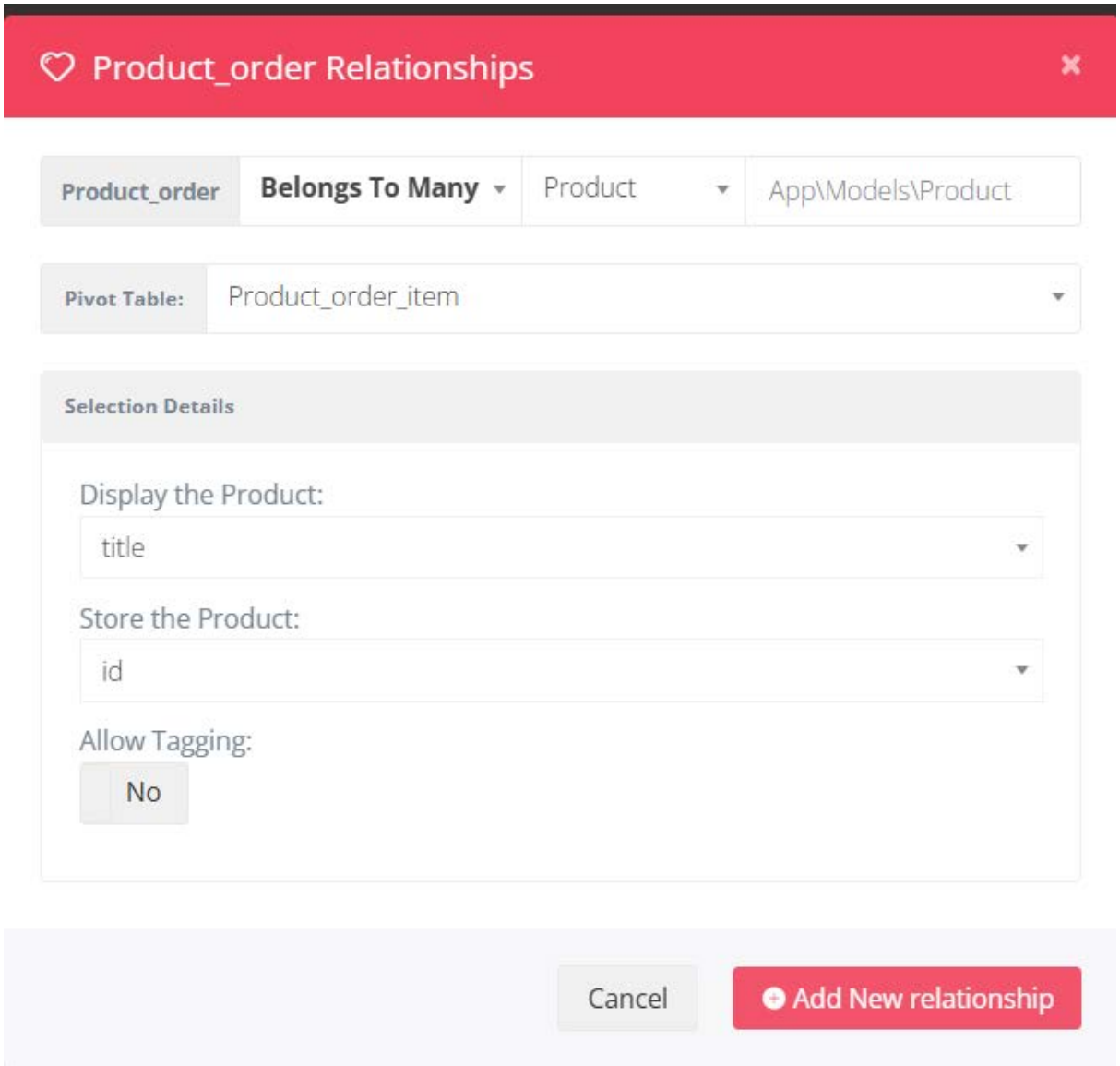


Рисунок 3.27 – Вигляд модального вікна для налаштування відношення між таблицями «Замовлення» та «Товари»

Як було визначено раніше, таблиця замовлень (orders) зв’язана з таблицею продуктів (products) як «багато до багатьох». Тому, використовуючи термінологію ORM-системи Eloquent, потрібно визначити, що сутність «Замовлення» (Order) «належить» («Belongs To Many») сутності «Product». При визначенні пов’язаної сутності потрібно вказати точну назву класу відповідної незалежної сутності (з вказанням простору імен). Також у полі «Pivot table» необхідно вказати назву зведеної таблиці.

У секції Selection Details можна налаштувати те, як буде відображатися інформація про зв’язок у випадаючих списках (наприклад, у елементах <select/>). У полі «Display the Product» необхідно вибрати поле, яке буде відображатися всередині елементів випадаючого списку <option/> (у нашому випадку це поле title таблиці products), а в поле «Store the Product» – поле, яке буде використовуватися в якості значення атрибуту value (у нашому випадку це поле id). Саме це значення буде передаватися при відправці форми на сервер.

На рис. 3.28 наведено зображення доданого зв’язка між таблицями «Замовлення» та «Товари» у редакторі шаблону BREAD.

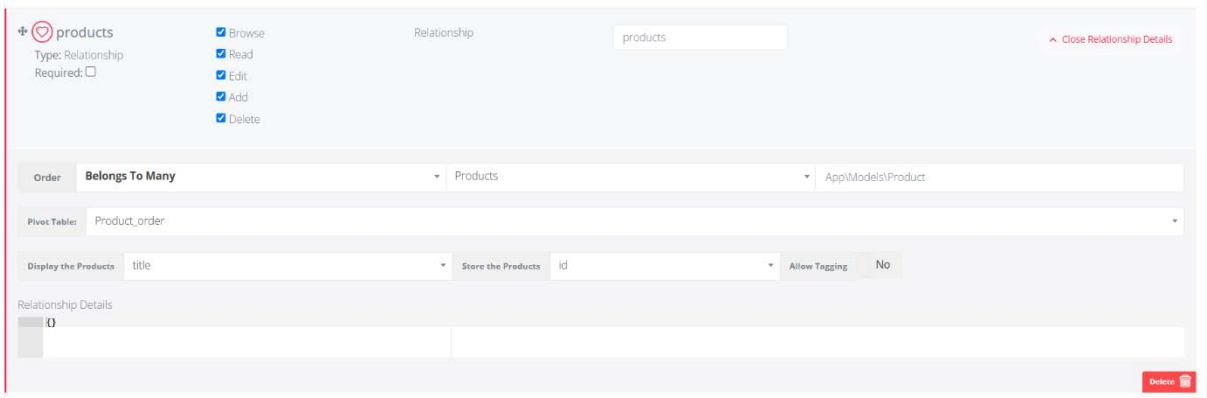


Рисунок 3.28 – Додане відношення між таблицями «Замовлення» та «Товари»

На рис. 3.29 наведено результати тестування створеної з використанням майстру BREAD сторінки для додавання нового замовлення. Варто звернути на поле «Товари», де у випадному списку можна додати всі необхідні товари до замовлення. Також форма містить випадний список «Постачальник», в якому можна обрати бажаного постачальника.

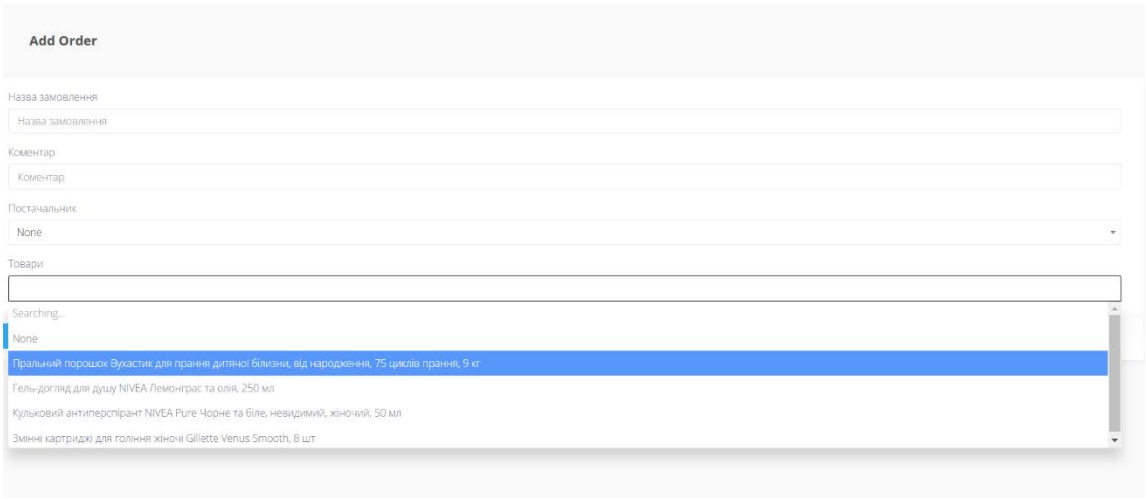


Рисунок 3.29 – Сторінка додавання нового замовлення

На рис. 3.30 наведено форму, підготовлену до збереження замовлення.

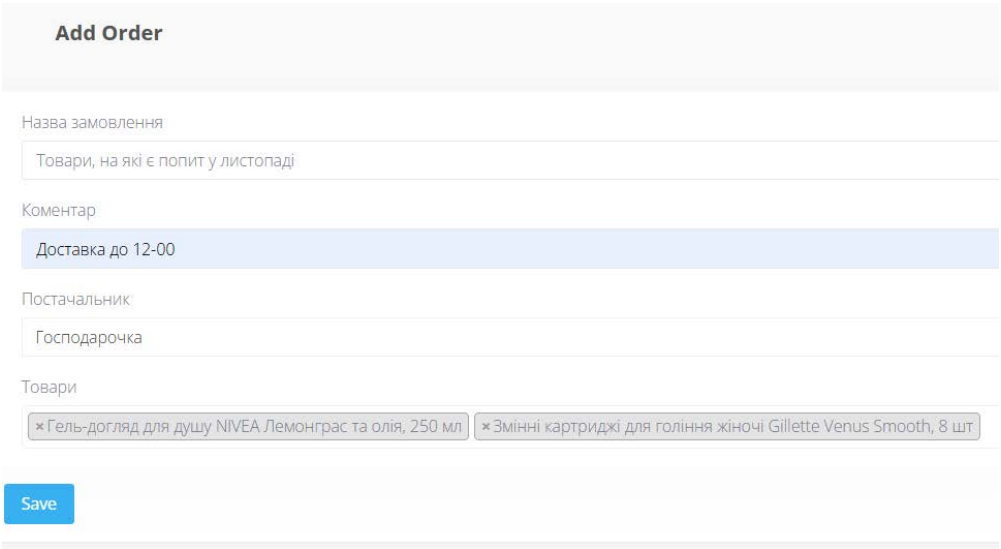


Рисунок 3.30 – Сторінка додавання нового замовлення з обраними товарами

На рис. 3.31 наведено сторінку зі списком сформованих замовлень. Як видно, на сторінці можна переглянути загальну інформацію по оформленим

замовленням, переглянути детальну інформацію по кожному замовленню (кнопка View), відредагувати (кнопка Edit) та, у разі необхідності, видалити конкретне замовлення (кнопка Delete). Крім того, можна додати нове замовлення (кнопка Add New), виконати операцію групового видалення (кнопка Bulk Delete), та змінити порядок сортування (кнопка Order).

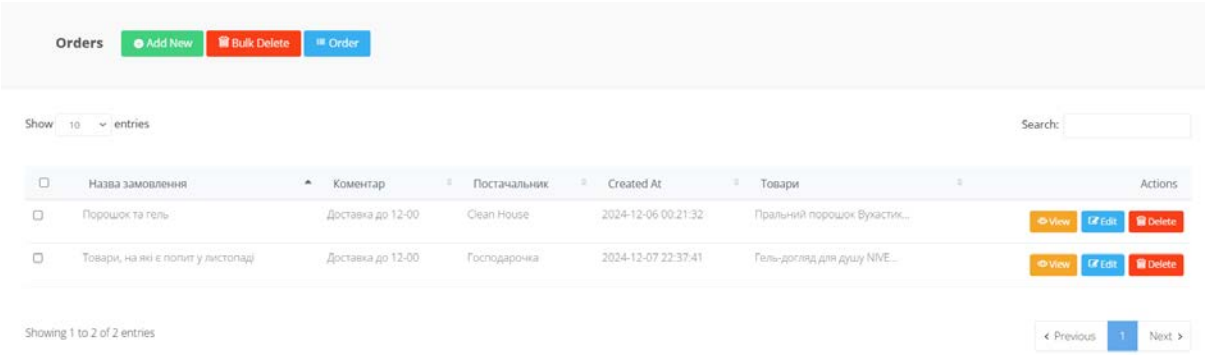


Рисунок 3.31 – Сторінка відображення списку оформлених замовлень

Для перевірки логічної цілісності даних при виконанні операції створення замовлення, яка впливає на вміст зведеної таблиці, доцільно розглянути, як була заповнена дана таблиця, та як було встановлено значення полів product_id та order_id, які утворюють складений первинний ключ таблиці Product_order. На рис. 3.32 наведено скрін зі сторінки «Переглянути» пакету для адміністрування баз даних MySQL/MariaDB для PHP, який називається phpMyAdmin.

Як видно, у наведеній таблиці містяться коректні дані, а поля product_id та order_id посилаються на відповідні записи у таблицях «Товари» (products) та «Замовлення» (orders).

Варто також звернути увагу, що на сторінці «Структура» - «Вид відносин» коректно відображається зв’язки зведеної таблиці Product_order з таблицями products та orders (рис. 3.33).

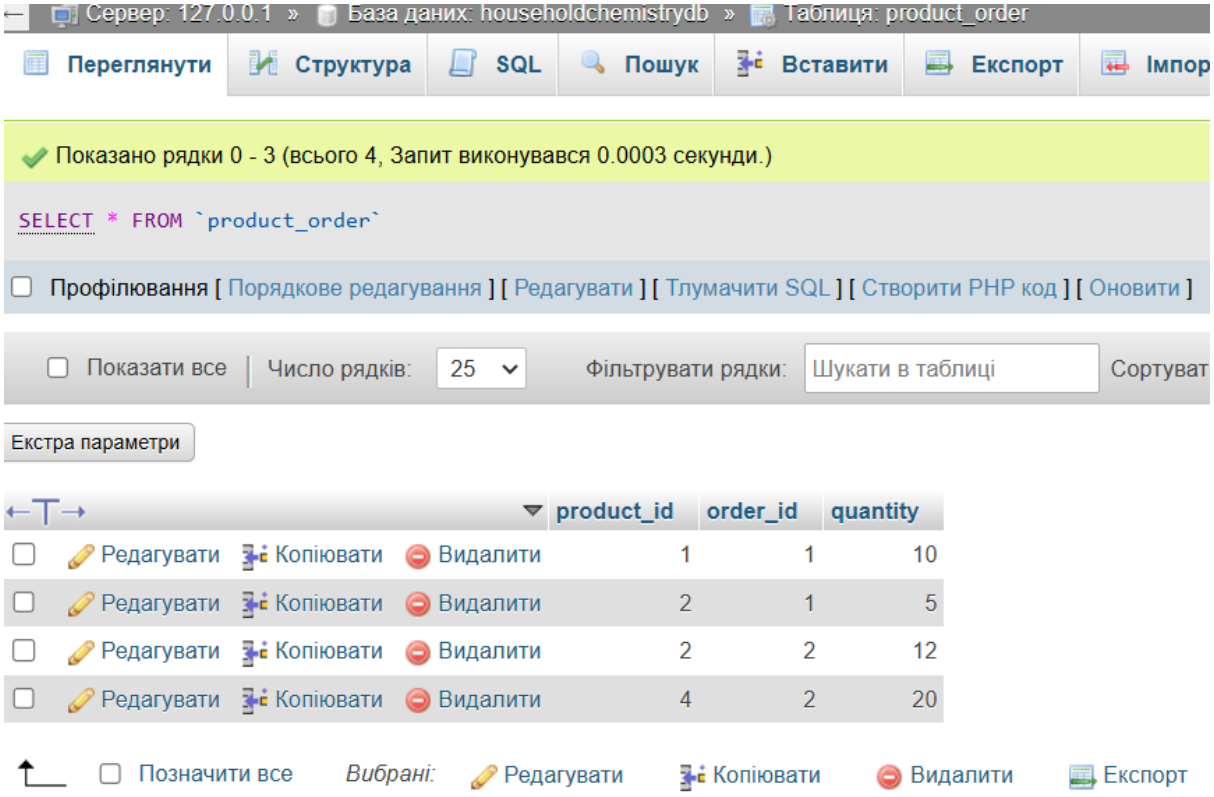


Рисунок 3.32 – Перегляд даних у зведеній таблиці product_order

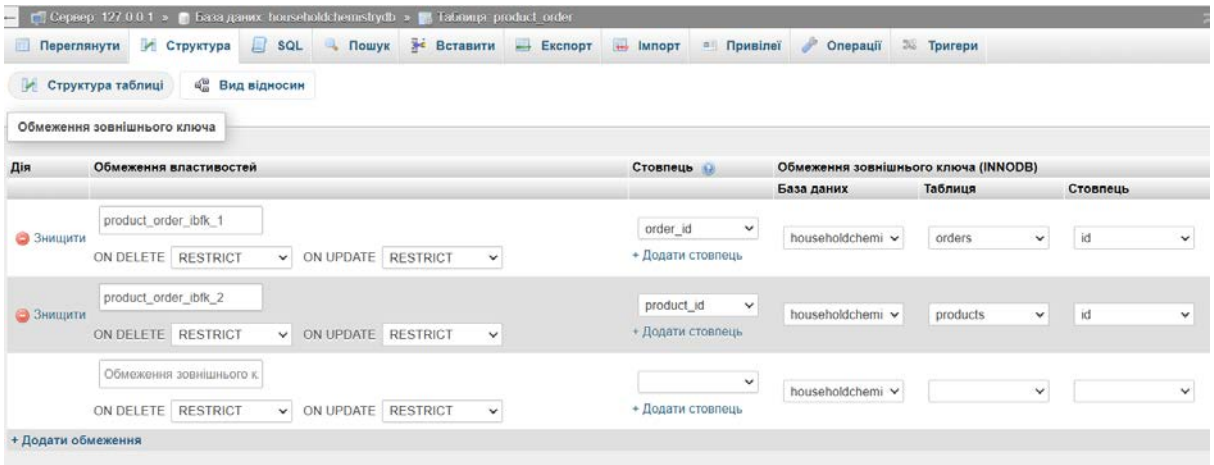


Рисунок 3.33 – Зв’язки типу «зовнішній ключ» у зведеній таблиці product_order з таблицями «Товари» та «Замовлення»

Аналогічним чином було виконано проектування структури таблиць баз даних, зв’язків між таблицями, та шаблонів для виконання CRUD-операцій для надходжень товарів від постачальників, бронювання та покупок від

користувачів системи. Використання пакету Voyager разом з фреймворком Laravel 9 дозволило відійти від рутинних операцій створення адміністративної панелі сервісу і сконцентруватися на реалізації бізнес-задач та сценаріїв роботи з даними при проєктуванні системи обліку товарів магазину побутової хімії.

Висновки за розділом:

З урахуванням обраних технологій розробки, зокрема фреймворку Laravel та пакету розширення для створення адмін-панелей Voyager, для збереження інформації про діяльність магазину побутової хімії було вирішено використати СУБД MySQL з розгортанням бази даних на безкоштовному тестовому сервері у хмарного провайдера Aiven.io.

Для реалізації інформаційної системи обліку товарів було розроблено необхідні файли міграцій Laravel та розгорнуто структуру БД.

З використанням інструментів пакету Voyager виконано проектування інформаційної структури інформаційної системи обліку товарів для магазину побутової хімії, розроблено основні шаблони сторінок для виконання типових CRUD-операцій над сутностями предметної області та реалізації визначених у попередньому розділі варіантів використання інформаційної системи. Використання пакету Voyager разом з фреймворком Laravel 9 дозволило сконцентруватися на реалізації бізнес-задач та сценаріїв роботи з даними при проектуванні системи обліку товарів магазину побутової хімії та ефективно реалізувати поставлені задачі.

ВИСНОВКИ

Був проведений аналіз предметної області, поставлена задача розробки, була досліджена предметна область, загальні принципи застосунків для ведення обліку та принцип їх роботи. Розглянуті існуючі програми для порівняння та встановлення функціональності власної системи обліку.

Об’єднання функцій складського і торговельного обліку в одній системі створить унікальний інструмент, який буде затребуваним серед підприємств різного масштабу. Така система дозволить оптимізувати управління товарообігом, зменшити помилки, спричинені людським фактором, і підвищити ефективність роботи бізнесу.

Було проведено всебічний аналіз і обґрунтування вибору технологій для створення системи обліку товарів інтернет-магазину, а також спроектовано її основні структурні складові. Таким чином, визначено концептуальні основи розробки системи обліку товарів для інтернет-магазину, які сприяють її ефективному впровадженню та подальшій експлуатації.

Сформовано функціональні вимоги до системи, що включають облік товарів, управління запасами, моніторинг замовлень і автоматизацію ключових бізнес-процесів. Розроблено модель взаємодії між основними компонентами застосунку.

Створено структуру інформаційної системи, що включає базу даних, інтерфейси користувача, модулі для управління даними, а також механізми забезпечення захисту та резервування інформації.

З урахуванням обраних технологій розробки, зокрема фреймворку Laravel та пакету розширення для створення адмін-панелей Voyager, для збереження інформації про діяльність магазину побутової хімії було вирішено використати СУБД MySQL з розгортанням бази даних на безкоштовному тестовому сервері у хмарного провайдера Aiven.io.

Для реалізації інформаційної системи обліку товарів було розроблено необхідні файли міграцій Laravel та розгорнуто структуру БД.

З використанням інструментів пакету Voyager виконано проектування інформаційної структури інформаційної системи обліку товарів для магазину побутової хімії, розроблено основні шаблони сторінок для виконання типових CRUD-операцій над сутностями предметної області та реалізації визначених у попередньому розділі варіантів використання інформаційної системи. Використання пакету Voyager разом з фреймворком Laravel 9 дозволило сконцентруватися на реалізації бізнес-задач та сценаріїв роботи з даними при проектуванні системи обліку товарів магазину побутової хімії та ефективно реалізувати поставлені задачі.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Кетков Ю., Кетков О. Практика програмування visual basic, C++ builder, delph. 2016.

2. ACode. Уроки програмування на C++. URL: https://acode.com.ua/uroki-po- cpp/

3. Besedin A. Secrets of access database development and programming. 2017.

4. C++ програмування. URL: http://cpp.dp.ua/

5. Організаціяя товарообігу. URL: https://library.if.ua/book/43/2972.html

6. Механізм управління товарооборотом. Ефективна економіка. 2019. 5. URL: http://www.economy.nayka.com.ua/?op=1&z=555

10.Про що розповідає роздрібний товарообіг. URL: http://edclub.com.ua/analityka/pro-shcho-rozpovidaye-rozdribnyy-tovaroobig

11. Head First C: A Brain-Friendly Guide. Beijing: O'Reilly, 2012. 591 с.

12. C++ Tutorial. URL: https://www.w3schools.com/cpp/default.asp

13. C++ Programing Language. URL: https://www.geeksforgeeks.org/c-plus-plus/

14. SQL Tutorial. URL: https://www.w3schools.com/sql/default.asp

15. What is Structured QueryLanguage (SQL). URL: https://www.techtarget.com/searchdatamanagement/definition/SQL

16. RAD Studio wiki. URL: https://docwiki.embarcadero.com/RADStudio/Alexandria/en/Main_Page

17. C++ builder. URL: https://www.bestprog.net/en/2016/02/25/008-c-builder-an- example-of-creating-and-calling-a-new-form-from-main-form-of-application/

18. Green D., Guntheroth K., Ross Mitchell S. The C++ Workshop: Learn to write clean, maintainable code in C++ and advance your career in software engineering : підручник. Packt Publishing, 2020. 606 p.

19. Bancila M. Modern C++ Programming Cookbook: Master C++ core language and standard library features, with over 100 recipes, updated to C++20, 2nd Edition : підручник. 2nd ed. Packt Publishing, 2020. 1141 p.

20. Roth S. Clean C++20: sustainable software development patterns and best practices : підручник. 2nd ed. Apress, 2021. 657 p.

21. Shields W. SQL quickstart guide: the simplified beginner's guide to managing, analyzing, and manipulating data with SQL (quickstart guides™ - technology). 2019. 242 p.

22. Moestl Vasilik S. SQL Practice Problems: 57 beginning, intermediate, and advanced challenges for you to solve using a “learn-by-doing” approach. 2017. 125 p.

23. Beighley L. Head first SQL: your brain on SQL : підручник. O'Reilly Media, 2007. 608 p.

24. SQL Querry Examples. URL: https://www.datacamp.com/tutorial/sql-query- examples-and-tutorial

25. ДСТУ 3008:2015. Звіти у сфері науки і техніки. Структура і правила оформлення. Київ, ДП «УкрННЦ», 2015. 26с. (Інформація та документація).

26. ДСТУ 8302:2015. Бібліографічне посилання. Загальні вимоги та правила складання Київ, ДП «УкрННЦ», 2016. 16 с. (Інформація та документація).

27. ДСТУ 3582:2013. Бібліографічний опис. Скорочення слів і словосполучень в українській мові. Загальні вимоги та правила. Київ, ДП «УкрННЦ», 2013. 23 с. (Інформація та документація)