

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

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

на тему:

«Інтернет магазин спортивного харчування»

Виконав студент групи ІПЗ-22-4

ГЕЛЕВЕЙ Олексій Андрійович

Керівник роботи:

БАКАЛЮК Тетяна Анатоліївна

Рецензент:

ЄФРЕМОВ Юрій Миколайович

Житомир – 2026

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

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії програм-
ного забезпечення

Тетяна Вакалюк

«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ГЕЛЕВЕЙ Олексій Андрійович**

Керівник роботи: **ВАКАЛЮК Тетяна Анатоліївна**

Тема роботи: **«Інтернет магазин спортивного харчування»,**

затверджена Наказом закладу вищої освіти від **«27» лютого 2026 р., № 92**

Вихідні дані для роботи: **вимоги до вебплатформи харчових добавок, каталог товарів, ролі покупця та адміністратора.**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Вакалюк Т.А.	01.03.2026р.	01.03.2026р
2	Вакалюк Т.А.	01.03.2026р	01.03.2026р
3	Вакалюк Т.А.	01.03.2026р	01.03.2026р

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра містить 80 сторінку, 20 рисунків, 18 таблиць, 30 використаних джерел та 2 додатки.

Метою роботи є створення вебзастосунку платформи ATLANT, призначеного для спортивного харчування. Система забезпечує повний базовий цикл електронної торгівлі такі як пошук товару, перегляд характеристик та доступних варіантів, роботу з кошиком, а також швидке оформлення замовлення без обов'язкової реєстрації покупця.

Об'єктом дослідження є процес розроблення інформаційної системи для автоматизації продажу товарів спеціалізованого магазину харчових добавок. Предметом дослідження є методи серверної розробки на PHP, організації реляційної бази даних MySQL, керування сесіями [20], побудови адаптивного інтерфейсу та розмежування прав покупця й адміністратора [18].

У роботі виконано аналіз предметної області, сформовано функціональні та нефункціональні вимоги, спроектовано архітектуру і структуру бази даних, описано алгоритми автентифікації та транзакційного оформлення замовлення. Реалізовано каталог із пошуком і динамічними фільтрами, картки товарів із підтримкою різних фасувань та смаків, сесійний кошик, форму замовлення та захищену адміністративну панель для керування асортиментом і статусами замовлень.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ-МАГАЗИН, ХАРЧОВІ ДОБАВКИ, СПОРТИВНЕ ХАРЧУВАННЯ, PHP, MYSQL, PDO, ЕЛЕКТРОННА КОМЕРЦІЯ, КАТАЛОГ, КОШИК, ЗАМОВЛЕННЯ, АДМІНІСТРАТИВНА ПАНЕЛЬ.

					ІПЗ.КР.Б – 121 – 26 - ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Гелевей О.А.			Інтернет магазин спортивного харчування	Літ.	Арк.	Аркушів
Керівник		Вакалюк Т.А.					3	80
						Житомирська політехніка, група ІПЗ-22-4		
Рецензент		Євремів Ю.М.						
Зав.каф		Вакалюк Т.А.						

ABSTRACT

The bachelor's qualification thesis contains 80 pages, 20 figures, 18 tables, 35 references, and 2 appendices.

The purpose of the work is to develop the ATLANT web application intended for the sale sports nutrition. The system covers the essential e-commerce workflow: product discovery, viewing characteristics and available variants, shopping cart management, and fast guest checkout.

The object of research is the development process of an information system that automates sales operations for a specialized dietary supplements store. The subject includes PHP server-side development, MySQL relational data modeling, session management, responsive interface design, and role-based access separation between buyers and administrators.

The thesis analyzes the domain, defines functional and non-functional requirements, presents the architecture and database schema, and describes authentication and transactional checkout algorithms. The implemented application contains a searchable catalog with dynamic filters, product cards supporting various flavors and weights, a session-based cart, an order form, and a secure administrator dashboard for managing the assortment and order statuses.

KEYWORDS: ONLINE STORE, DIETARY SUPPLEMENTS, SPORTS NUTRITION, PHP, MYSQL, PDO, E-COMMERCE, PRODUCT CATALOG, SHOPPING CART, ORDER, ADMINISTRATION PANEL.

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

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CRUD – Create, Read, Update, Delete;

CSS – Cascading Style Sheets; каскадні таблиці стилів

DBMS / СУБД – система управління базами даних

HTML – HyperText Markup Language; мова розмітки гіпертексту

HTTP – HyperText Transfer Protocol; протокол обміну вебданими

PDO – PHP Data Objects; інтерфейс доступу до баз даних PHP

PHP – серверна мова програмування загального призначення

SQL – Structured Query Language; мова структурованих запитів

UI – User Interface; інтерфейс користувача

UX – User Experience; досвід взаємодії користувача

BCAA – Branched-Chain Amino Acids;

CSRF – Cross-Site Request Forgery; міжсайтова підробка запиту

SEO – Search Engine Optimization; пошукова оптимізація вебсайту

XSS – Cross-Site Scripting; міжсайтовий скриптинг

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		5

ЗМІСТ

РЕФЕРАТ.....	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	5
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ	9
1.1. Постановка задачі	9
1.2. Аналіз аналогів програмного продукту	11
1.3. Вибір архітектури платформи	15
1.4. Обґрунтування вибору інструментальних засобів	18
Висновки до першого розділу	22
РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ATLANT	24
2.1. Визначення варіантів використання та структури системи	24
2.2. Проєктування бази даних.....	29
2.3. Проєктування та реалізація алгоритмів роботи.....	36
2.4. Особливості програмної реалізації вебзастосунку.....	42
Висновки до другого розділу.....	45
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ	46
3.1. Порядок встановлення та налаштування системи.....	46
3.2. Структура інтерфейсу та порядок взаємодії з платформою.....	49
3.3. Тестування роботи програмного продукту	57
Висновки до третього розділу	63
ВИСНОВКИ	64
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	65
ДОДАТКИ.....	65

ВСТУП

Актуальність теми – Сучасний ринок харчових добавок в Україні демонструє стабільне зростання на 15–20% щороку, що зумовлено підвищенням культури здорового способу життя та формує високий попит на ефективні інструменти онлайн-торгівлі [11, 12, 22, 23]. Більшість нішевих магазинів розповсюджують продукцію через універсальні маркетплейси, втрачаючи тематичну ідентичність і гнучкість представлення специфічного асортименту [22]. Тому розробка власної спеціалізованої вебплатформи є практично виправданим рішенням для автоматизації бізнес-процесів підприємства. Використання перевіреного стеку технологій, що поєднує серверну мову PHP, реляційну СУБД MySQL та стандарти HTML5, CSS3 і JavaScript [15, 16, 29], дозволяє створити повноцінний адаптивний вебзастосунок без залучення складної серверної інфраструктури [7]. Локальні середовища OpenServer та XAMPP забезпечують швидке розгортання та демонстрацію системи без необхідності придбання хостингу [19]. Це робить розробку актуальною як з практичної точки зору, так і з позиції дослідження веборієнтованих систем електронної комерції.

Мета роботи – є проектування та розробка вебзастосунку ATLANT для автоматизації процесів продажу харчових добавок і спортивного харчування, керування каталогом продукції, оформлення замовлень та адміністрування діяльності спеціалізованого інтернет-магазину. Досягнення поставленої мети передбачає створення інформаційної системи, яка забезпечує централізоване зберігання даних, підтримку основних бізнес-процесів електронної торгівлі, розмежування прав доступу користувачів та зручний інтерфейс для покупців і адміністратора.

Для досягнення поставленої мети необхідно виконати наступні завдання:

1. Дослідити особливості організації онлайн-продажу харчових добавок та спортивного харчування.
2. Проаналізувати існуючі програмні рішення та аналоги на ринку.
3. Визначити функціональні та нефункціональні вимоги до розроблюваної системи.

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

4. Спроекувати клієнт-серверну архітектуру та реляційну структуру бази даних.
5. Реалізувати основні функціональні модулі вебзастосунку ATLANT.
6. Забезпечити механізми безпечної авторизації адміністратора та керування статусами замовлень.
7. Створити інструменти управління товарами та варіантами фасування.
8. Виконати комплексне функціональне тестування системи для перевірки її працездатності та відповідності поставленим вимогам.

Об'єкт дослідження – процес автоматизації онлайн-продажу харчових добавок і спортивного харчування із використанням сучасних вебтехнологій, що охоплює представлення товарного асортименту з варіантами смаку та ваги, взаємодію користувачів із каталогом продукції, оформлення та обробку замовлень, а також управління даними магазину в умовах електронної комерції.

Предмет дослідження – методи та засоби розробки веборієнтованих інформаційних систем електронної комерції, зокрема використання мови програмування PHP, системи керування базами даних MySQL, інтерфейсу PDO, механізмів авторизації через сесії, транзакційної обробки даних та технологій створення адаптивних вебінтерфейсів для продажу спортивного харчування.

Структура роботи – Випускна кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел із 30 найменувань та 3 додатків.

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

РОЗДІЛ 1. АНАЛІЗ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

1.1. Постановка задачі

Предметна область спортивного харчування та харчових добавок охоплює широкий та різномірний асортимент товарів, до якого входять протеїнові суміші, гейнери, амінокислотні комплекси, жироспалювачі, вітаміни і мінерали, передтренувальні комплекси та аксесуари для тренувань. У традиційній моделі організації продажів покупець отримує інформацію про наявність продукції шляхом відвідування магазину або телефонного звернення. Такий підхід має низку суттєвих недоліків, зокрема ускладнює порівняння товарів за складом і призначенням, не дозволяє самостійно перевіряти актуальні залишки та ціни, а також унеможливорює зручне повторне замовлення без участі продавця.

Використання інтернет-магазину як сучасного формату електронної комерції дозволяє ефективно вирішити зазначені проблеми. Проте значна частина спеціалізованих магазинів спортивного харчування або розміщує товари на великих маркетплейсах, втрачаючи власну тематичну ідентичність, або взагалі не має власного цифрового представництва. На універсальних торговельних платформах харчові добавки розміщуються серед тисяч інших категорій товарів, що значно ускладнює навігацію для цільової аудиторії. Крім того, продавці не мають повного контролю над структурою каталогу, варіантами товарів та процесами обробки замовлень.

Саме ці фактори стали основною передумовою для проектування та розробки власного вебзастосунку ATLANT. Розроблена система орієнтована на створення зручного цифрового середовища як для покупців, так і для адміністрації магазину. Платформа надає можливість здійснювати пошук і фільтрацію товарів за категорією, брендом, ціллю використання та ціновим діапазоном. Також реалізовано механізм відображення актуальних залишків товарів та їх варіантів (смак, вага) у режимі реального часу з автоматичним оновленням після кожного оформленого замовлення.

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

Окрім функціоналу для покупців, система передбачає повноцінний адміністративний модуль, призначений для ефективного управління діяльністю магазину. Для адміністратора створено окремий захищений інтерфейс, доступ до якого надається лише після проходження процедури авторизації через сесійний механізм РНР. За допомогою цього інтерфейсу адміністратор може здійснювати повний облік товарів, додавати нові позиції до каталогу разом із варіантами смаку і ваги, редагувати описи та ціни, контролювати складські залишки. Також передбачено можливість перегляду замовлень і зміни їх статусів на різних етапах виконання.

Адміністративна панель забезпечує централізований доступ до всієї інформації, необхідної для оперативного прийняття управлінських рішень. Система дозволяє відстежувати динаміку замовлень, переглядати графік продажів, контролювати залишки та виявляти популярні позиції. Крім того, інтерфейс адміністрування підтримує формування звітів у реальному часі, що дозволяє керівництву магазину миттєво реагувати на зміни попиту. Інтегровані інструменти аналітики допомагають прогнозувати потреби в закупівлях, мінімізуючи ризики дефіциту товарів, що суттєво зменшує кількість помилок під час виконання операцій з асортиментом.

Впровадження вебзастосунку ATLANT спрямоване на комплексну автоматизацію основних бізнес-процесів та підвищення якості обслуговування клієнтів. Створення незалежної цифрової платформи дозволяє гнучко оптимізувати облік товарних позицій, забезпечити високу надійність транзакцій та спростити щоденне адміністрування. Завдяки модульній архітектурі, система зберігає високий рівень масштабованості, що забезпечує стабільну роботу в умовах зростання навантаження та створює надійну основу для подальшої цифровізації торговельних процесів. Для наочного представлення функціональних вимог, основні задачі платформи ATLANT, їхній зміст та інструменти реалізації наведено у таблиці (табл. 1.1).

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

Основні задачі системи та їх реалізація

Задача	Зміст	Реалізація у ATLANT
Структурований каталог	Товари з варіантами смаку, ваги, форми випуску	Таблиці products, product_variants, сторінка catalog.php
Пошук і фільтрація	Вибір товарів за текстом, категорією, брендом, ціллю	Динамічний SQL WHERE, PDO bind params
Облік залишків	Актуальний stock з оновленням після замовлення	UPDATE product_variants у транзакції checkout.php
Кошик і замовлення	Збереження вибраних позицій та фіксація покупки	Таблиці orders, order_items, сесія \$_SESSION['cart']
Варіанти товарів	Різні смаки і ваги одного продукту	Таблиця product_variants, вибір на product.php
Адміністрування	Керування магазином із захищеним контуром доступу	Каталог admin/, перевірка \$_SESSION['admin_id']

Дані, наведені у таблиці 1.1, демонструють основні функціональні задачі вебзастосунку ATLANT та способи їх реалізації в системі. Реалізація структурованого каталогу з варіантами товарів, механізмів пошуку й фільтрації, обліку складських залишків, оформлення замовлень через сесійний кошик та адміністративного управління забезпечує комплексну автоматизацію діяльності магазину харчових добавок. Для обґрунтування доцільності розробки власного програмного продукту доцільно провести аналіз існуючих аналогів та визначити їхні переваги й недоліки.

1.2. Аналіз аналогів програмного продукту

Перед розробкою було розглянуто три типи майданчиків, якими фактично користуються покупці харчових добавок в Україні великий універсальний маркетплейс Rozetka, а також спеціалізовані нішеві інтернет-магазини спортивного харчування та нутрієнтів ВСАА.ua і Belok.ua. Порівняльний аналіз цих систем дозволив чітко визначити їхні переваги та недоліки, а також сформулювати перелік функціональних можливостей, необхідних для реалізації власного програмного продукту.

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

Rozetka — найбільший універсальний маркетплейс України з широким асортиментом, у тому числі розділом спортивного харчування. Сильні сторони: розвинений пошук і фільтри, система рейтингів і відгуків, впізнаваний процес оформлення замовлення та відстеження доставки. Проблема для нішевого продавця полягає в тому, що асортимент харчових добавок розпорошений між сотнями продавців, бренди конкурують на одній сторінці, тематична подача відсутня, а продавець не контролює власний адміністративний простір незалежно від платформи. Інтерфейс аналога наведено на рисунку (рис. 1.1).

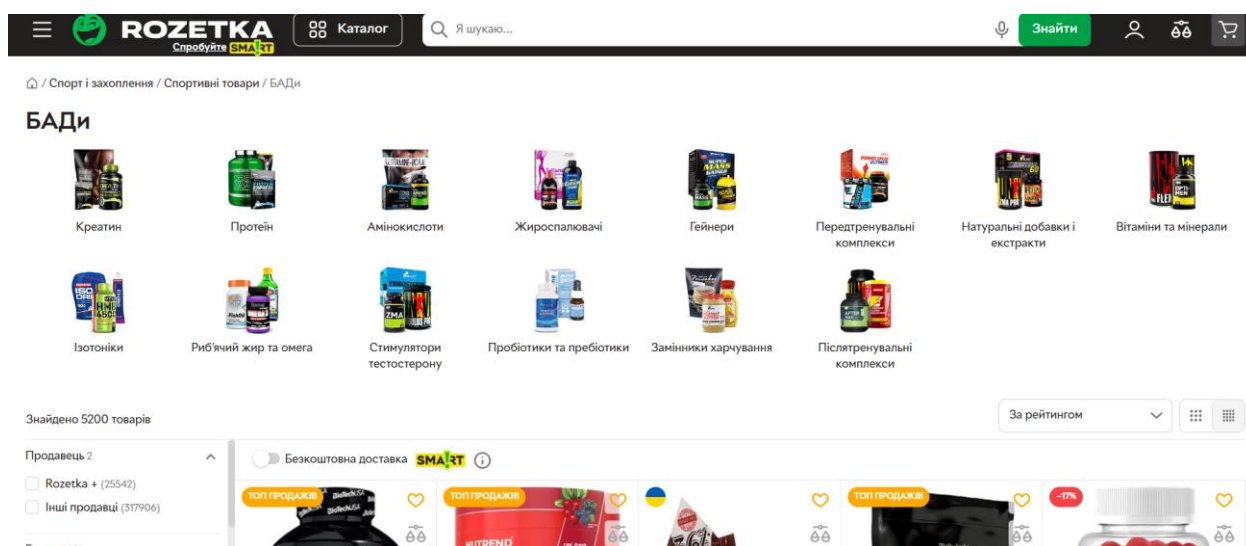


Рис. 1.1 – Платформа Rozetka (розділ спортивного харчування)

ВСАА.ua — спеціалізований інтернет-магазин вітамінів, амінокислот (ВСАА), протеїну та низькокалорійних солодощів без цукру, орієнтований на активну аудиторію спортсменів і прихильників здорового харчування. Його сильні сторони полягають у чіткій тематичній структурі категорій, зручній фільтрації за типом продукту та брендом, а також у вузькій спеціалізації, що забезпечує релевантність цільовій групі. Аналіз функціональних можливостей цього ресурсу показує, що навіть спеціалізовані магазини потребують подальшого розвитку для задоволення зростаючих вимог користувачів (рис. 1.2).

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

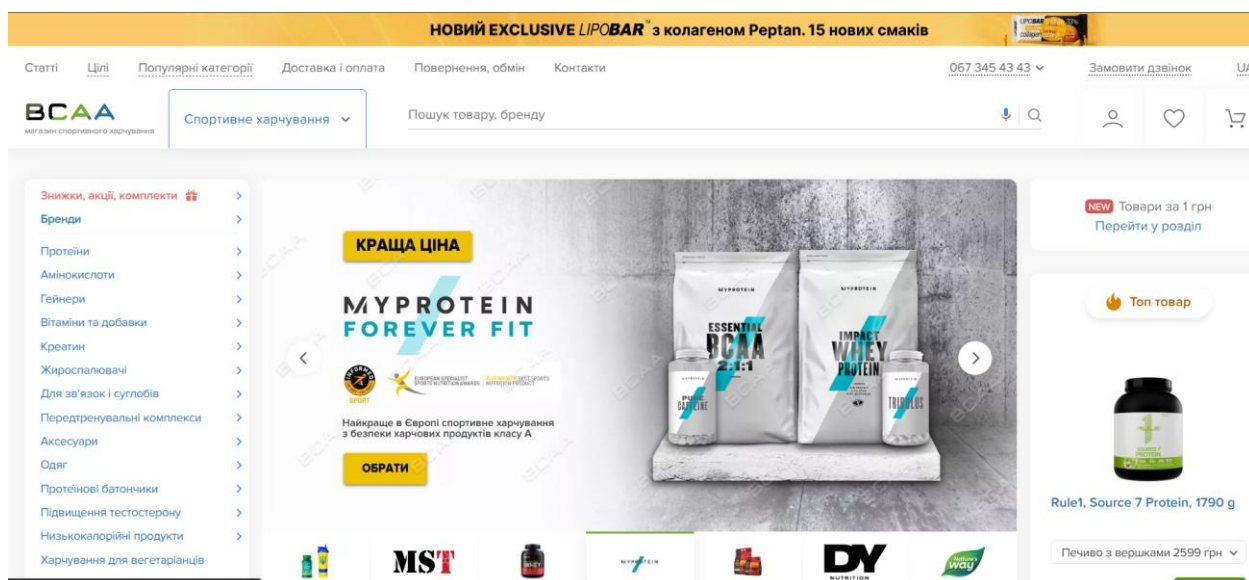


Рис. 1.2 – Спеціалізований магазин BCAA.ua

Belok.ua — один з найбільших спеціалізованих маркетплейсів спортивного харчування в Україні, чий каталог налічує понад 2000 найменувань товарів від провідних світових брендів [12]. До ключових сильних сторін цієї платформи належать широкий асортимент продукції, розгалужена та інформативна структура категорій, а також наявність товарів у різних цінових сегментах, що дозволяє охопити максимальну кількість покупців.

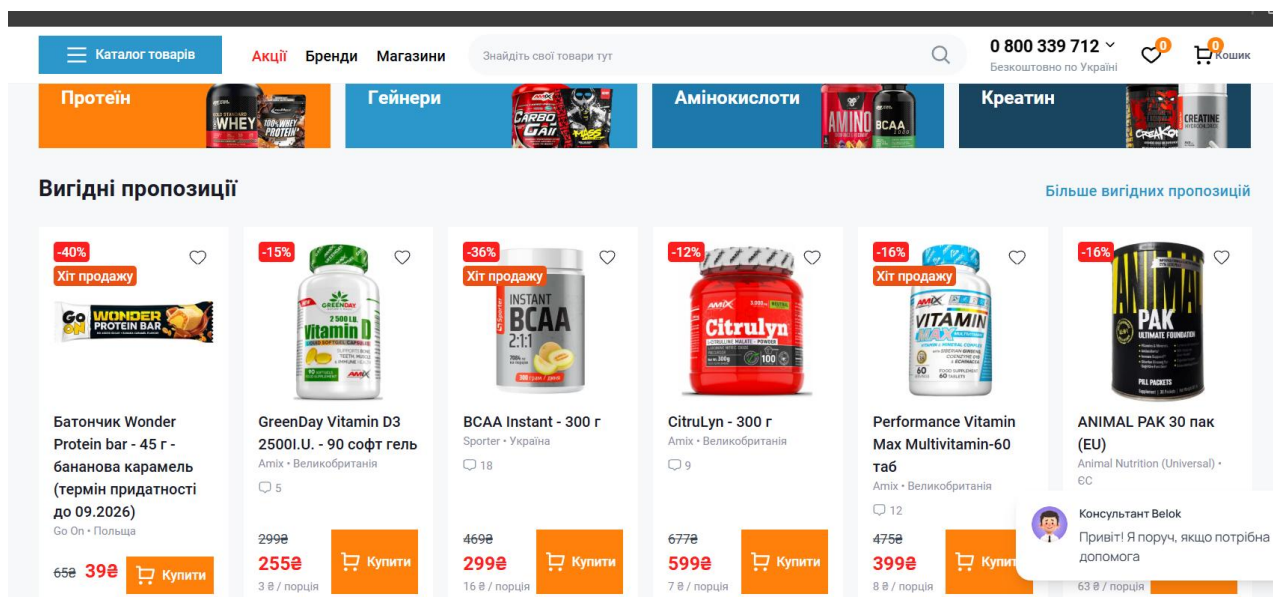


Рис. 1.3 – Маркетплейс спортивного харчування Belok.ua

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

У результаті аналізу від спеціалізованих магазинів запозичено підхід до тематичної структури каталогу та фільтрації за ціллю використання, від маркетплейсів — логіку широкого асортименту та прозорого процесу замовлення, а слабкі сторони всіх аналогів закриті власною структурованою базою даних, контролем залишків варіантів і повноцінною адміністративною панеллю — це наведено у таблиці (табл. 1.2).

Таблиця 1.2

Аналіз конкурентних рішень у сфері продажу харчових добавок

Критерій	Rozetka	BCAA.ua	Belok.ua	ATLANT
Тематичний каталог	Частково	Так	Так	Так
Пошук і фільтри	Розширені	Обмежені	Часткові	Так
Постійний кошик	Так	Так	Так	Так
Варіанти товарів (смак/вага)	Так	Частково	Частково	Так
Облік залишків	Так	Так	Так	Так
Адміністративна панель	Так	Обмежена	Обмежена	Так
Незалежність від платформи	Ні	Ні	Ні	Так

Аналіз, наведений у таблиці 1.2, дозволяє порівняти функціональні можливості існуючих рішень у сфері продажу харчових добавок та визначити їхні переваги й недоліки. Кожному з порівняних критеріїв відповідає окремий програмний модуль, реалізований у розділі 2: тематичний каталог відповідає сценарію UC-Catalog, пошук і фільтри — UC-Search, постійний кошик — UC-Cart, варіанти товарів сутності product_variants, а облік залишків — ERD-Products.

Великі маркетплейси забезпечують широкий набір функцій для покупців, проте не орієнтовані виключно на тематику харчових добавок і не надають продавцю повного контролю над власним магазином. Спеціалізовані магазини, такі як BCAA.ua та Belok.ua, мають більш релевантну структуру каталогу, однак їхня функціональність часто обмежується у частині управління варіантами товарів і

гнучкості адміністративної панелі. Платформа ATLANT розроблена з урахуванням виявлених недоліків і забезпечує повний контроль над усіма аспектами роботи магазину.

1.3. Вибір архітектури платформи

Вебзастосунок ATLANT побудований на основі класичної багаторівневої клієнт-серверної архітектури, яка забезпечує чітке розмежування між рівнем представлення даних, рівнем серверної бізнес-логіки та рівнем зберігання інформації. Використання такого підходу дозволяє зробити структуру програмного забезпечення більш зрозумілою, спрощує процес розробки та подальшого супроводу системи. Крім того, дана архітектура дає можливість простежити повний цикл обробки запиту в межах одного PHP-файлу: від отримання параметрів користувача та перевірки його прав доступу до виконання SQL-запитів і формування готової HTML-сторінки для відображення у браузері [1].

Клієнтський рівень відповідає за взаємодію користувача із системою та забезпечує відображення інформації у зручному вигляді. Для цього використовуються HTML-сторінки, стилізовані за допомогою CSS, а також мінімальний набір JavaScript-скриптів [17], необхідних для реалізації мобільного меню, підтвердження небезпечних дій, адаптивної сітки карток товарів та динамічного оновлення ціни при виборі варіанта [3, 4, 5, 6]. Серверний рівень реалізовано мовою програмування PHP, яка виконує обробку запитів користувачів, маршрутизацію між сторінками, перевірку сесій, валідацію введених даних та формування SQL-запитів через інтерфейс PDO [1, 27]. Рівень даних представлений реляційною системою управління базами даних MySQL, що забезпечує надійне зберігання інформації, підтримку зовнішніх ключів для збереження цілісності зв'язків між таблицями та можливість використання транзакцій під час виконання критично важливих операцій [2].

Використання серверного рендерингу HTML є доцільним і повністю виправданим рішенням для даного проєкту [30]. Такий підхід дозволяє значно

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						15
Змн.	Арк.	№ докум.	Підпис	Дата		

спростити архітектуру системи, оскільки відсутня необхідність у створенні окремого REST API, використанні сучасних frontend-фреймворків або налаштуванні додаткових інструментів збірки проєкту. Альтернативні підходи, зокрема розробка односторінкових застосунків (SPA) на основі React чи Vue з окремим серверним бекендом, потребують складнішої інфраструктури, більшої кількості ресурсів та додаткового часу на підтримку. Для невеликої системи, орієнтованої на роботу одного продавця та адміністратора, подібні рішення є надмірними. Натомість використання PHP із серверним формуванням сторінок забезпечує простоту розгортання, сумісність із більшістю хостинг-платформ та можливість швидкого запуску застосунку у локальному середовищі OpenServer без необхідності виконання додаткових налаштувань [13, 15, 16].

Публічна частина системи та адміністративна панель функціонують у межах єдиної програмної структури та використовують спільне підключення до бази даних. Для перевірки факту авторизації адміністратора використовується перевірка наявності сесійної змінної `$_SESSION['admin_id']` на початку кожного захищеного файлу. У разі її відсутності виконується негайне перенаправлення на сторінку входу через `header('Location: login.php')`. Такий підхід забезпечує централізоване управління доступом та спрощує контроль прав у системі [8].

Крім того, обрана клієнт-серверна архітектура забезпечує високий рівень продуктивності та стабільності для вебзастосунку даного типу навіть в умовах інтенсивного користувацького трафіку. Основна обробка бізнес-логіки, формування SQL-запитів до бази даних та генерація динамічного контенту виконуються безпосередньо на стороні сервера. Це дозволяє суттєво мінімізувати обчислювальне навантаження на клієнтські пристрої (браузери) та гарантувати високу швидкість відгуку інтерфейсу й коректну роботу системи навіть на малопотужних комп'ютерах або застарілих моделях мобільних пристроїв. Завдяки такому підходу вебзастосунок ATLANT є зручним, надійним, кросбраузерним та повністю придатним для подальшого масштабування, інтеграції нових модулів і впровадження в реальних умовах комерційної експлуатації.

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

З метою підвищення зручності адміністрування, супроводу та масштабування вихідного коду, всі спільні компоненти вебзастосунку були централізовано винесені до окремих структурних файлів. До них належать ізольовані модулі безпечного підключення до бази даних, конфігураційні параметри системи, а також повторно використовувані елементи інтерфейсу, такі як шапка (header) та підвал (footer) вебсторінок. Подібна організація структури проєкту базується на фундаментальному архітектурному принципі DRY (Don't Repeat Yourself), що дозволяє повністю уникнути дублювання коду, підвищити його читабельність для сторонніх розробників та значно спростити і прискорити внесення будь-яких функціональних змін у майбутньому [1]. Загальна схема взаємодії всіх програмних модулів, логічних рівнів та компонентів вебзастосунку ATLANT наведена на рисунку (рис. 1.4). Основні архітектурні компоненти системи, їхня конкретна роль у загальній структурі та розгорнуте обґрунтування вибору технологічного стеку наведено у таблиці (табл. 1.3).

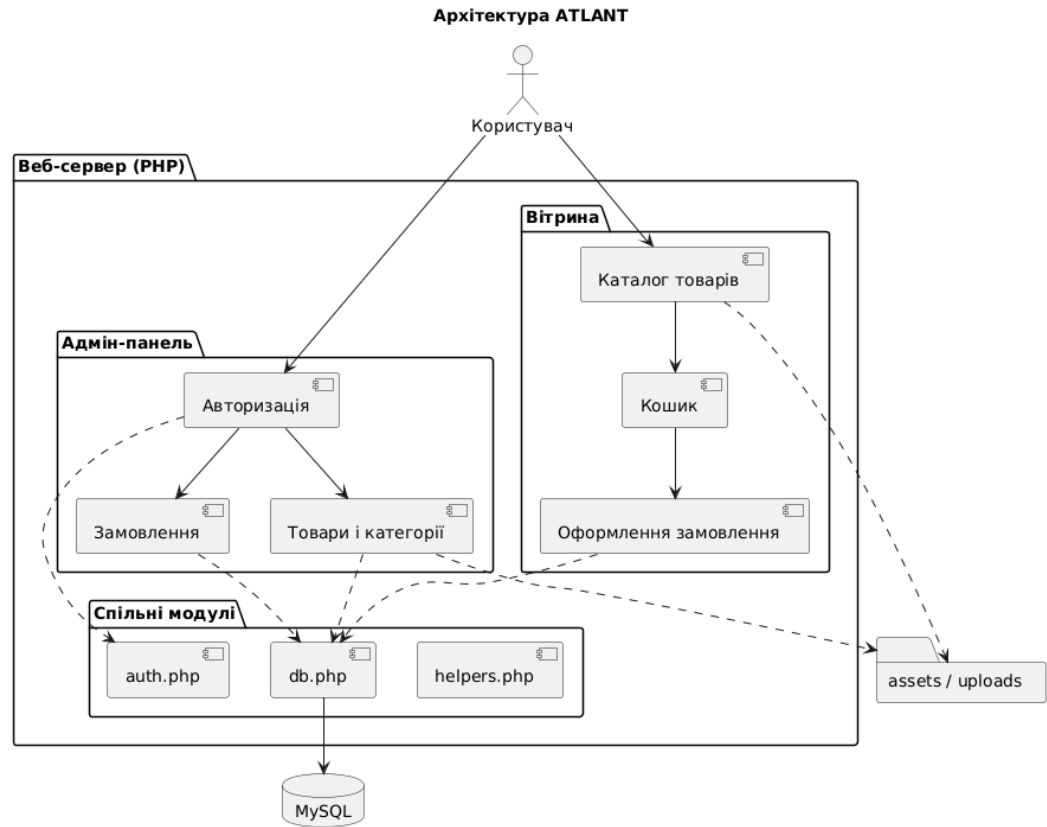


Рис. 1.4 – Загальна архітектура ATLANT

Архітектурні компоненти та їх обґрунтування

Компонент	Роль у системі	Причина вибору
HTML5	Структура сторінок і форм	Стандартна підтримка всіма браузерами
CSS3	Адаптивне оформлення інтерфейсу	Не потребує зовнішнього UI-фреймворку, повний контроль над стилем
JavaScript	Мобільне меню, вибір варіанту товару, підтвердження дій	Достатньо для невеликих клієнтських інтеракцій
PHP	Серверна бізнес-логіка, сесії, маршрутизація	Природна інтеграція з Apache, доступне середовище розгортання
PDO	Уніфікований доступ до MySQL	Підготовлені запити, єдиний інтерфейс, захист від SQL-ін'єкцій
MySQL	Реляційне зберігання всіх даних	Зовнішні ключі, транзакції, індекси, широка підтримка хостингів
OpenServer	Локальне середовище розробки та демонстрації	Вбудований Apache+MySQL+PHP, проста конфігурація доменів

Наведені в таблиці компоненти формують цілісну архітектуру вебзастосунку та забезпечують узгоджену взаємодію між усіма рівнями системи. Кожна технологія виконує чітко визначену роль, що дозволяє уникнути надлишкової складності та зберегти зрозумілу структуру програмного продукту.

1.4. Обґрунтування вибору інструментальних засобів

Головним критерієм під час вибору кожного програмного інструменту для розробки вебзастосунку ATLANT була його практична доцільність та здатність ефективно вирішувати конкретні завдання проєкту. При цьому перевага надавалася технологіям, які дозволяють реалізувати необхідний функціонал найпростішим та найбільш зрозумілим способом без використання зайвих компонентів або складних програмних рішень. Такий підхід дозволив зберегти простоту архітектури системи, забезпечити її стабільну роботу та спростити подальший супровід програмного продукту.

Мову програмування PHP 8 було обрано як основу серверної частини вебзастосунку завдяки її широкому поширенню, простоті використання та природній інтеграції з вебсервером Apache. Важливою перевагою також є наявність великої кількості вбудованих засобів для роботи з HTTP-запитами, формами користувачів, файлами cookie та механізмами сесій [1]. Крім того, PHP входить до складу популярних локальних середовищ розробки, зокрема XAMPP та OpenServer, що значно спрощує процес створення та тестування вебзастосунку [10]. Використання цієї мови дозволяє реалізувати повний цикл обробки запиту в межах одного програмного файлу, починаючи від отримання даних користувача і завершуючи формуванням готової HTML-сторінки. Завдяки цьому логіка роботи системи залишається зрозумілою та зручною для аналізу, що особливо важливо в межах кваліфікаційної роботи.

Для взаємодії із системою управління базами даних використовується розширення PDO, яке виступає єдиним інтерфейсом доступу до MySQL [1]. Однією з головних переваг PDO є підтримка підготовлених запитів через механізм `prepare()` та `execute()`, що дозволяє автоматично відокремлювати SQL-команди від користувацьких даних і значно підвищує рівень безпеки застосунку [1]. Такий підхід повністю виключає можливість успішного проведення SQL-ін'єкцій [8]. Додатково використовується режим `PDO::ERRMODE_EXCEPTION` та вмикається емуляція підготовлених запитів (`ATTR_EMULATE_PREPARES => false`), що забезпечує автоматичну генерацію винятків при виникненні помилок під час виконання запитів і гарантує сувору типізацію даних.

Як систему керування базами даних обрано MySQL 8, оскільки структура предметної області електронної торгівлі харчовими добавками природно відповідає реляційній моделі даних [2]. Основні сутності системи, такі як адміністратори, категорії, товари, варіанти товарів, замовлення та позиції замовлення, мають чітко визначені взаємозв'язки (One-to-Many, Many-to-Many), які ефективно реалізуються за допомогою зовнішніх ключів (Foreign Keys) рушія InnoDB [2]. Такий підхід дозволяє підтримувати посиальну цілісність (Referential Integrity)

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		19

даних, автоматизувати каскадне оновлення та видалення записів, а також мінімізувати ризик виникнення аномалій даних. Завдяки використанню ACID-сумісних транзакцій процес оформлення замовлення покупця групується в єдиний атомарний блок операцій за допомогою методів `beginTransaction()`, `rollback()` та `commit()`. Це гарантує, що замовлення не буде створено частково, навіть у разі системного збою, що є критично важливим для фінансових операцій [2]. Для забезпечення коректного відображення та зберігання розширеного набору символів (зокрема, Emojis та багатомовного кириличного тексту) встановлено кодування `utf8mb4_unicode_ci`, що є сучасним стандартом для підтримки міжнародних символів [2].

Для безпечного зберігання облікових даних адміністраторів використовується алгоритм хешування SHA-256 із застосуванням техніки «соління» (salting), що реалізовано за допомогою вбудованої функції `hash()` [1]. У результаті в базі даних зберігаються не паролі у відкритому вигляді, а їхні фіксовані криптографічні хеші, що унеможливорює відновлення паролів у разі витоку даних. Перевірка правильності введених даних під час автентифікації здійснюється шляхом повторної генерації хешу від введеного пароля з урахуванням збереженої «солі» та його суворого порівняння з наявним записом. Для підтримки стану авторизації використовується стандартний механізм сесій PHP, у якому надійно фіксуються ідентифікатори користувачів. Сесії захищені від типових атак шляхом встановлення відповідних параметрів Cookie (`HttpOnly`, `Secure`), що дозволяє гнучко контролювати права доступу до захищеної частини платформи та запобігати несанкціонованому перехопленню сесій [1, 8].

Користувацький інтерфейс вебзастосунку було реалізовано на базі сучасних стандартів HTML5 та CSS3 без використання важких сторонніх CSS-фреймворків, що значно покращує показники продуктивності та зменшує час завантаження сторінок (LCP – Largest Contentful Paint) [3, 4, 28]. Відмова від використання готових бібліотек надала можливість отримати повний контроль над структурою DOM-дерева, зовнішнім виглядом та поведінкою елементів. Для побудови гнучких та складних макетів інтерфейсу використовується технологія CSS

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

Grid, яка застосовується при створенні динамічної сітки каталогу товарів, забезпечуючи чітке вирівнювання блоків [6]. Адаптація інтерфейсу до мобільних пристроїв та забезпечення чуйності дизайну (Responsive Web Design) реалізована за допомогою медіазапитів (@media), які автоматично перераховують стилі залежно від ширини екрана користувача та типу пристрою [5]. Додатково для забезпечення безпеки клієнтської частини, зокрема для фільтрації виведених текстових даних та захисту від атак типу Cross-Site Scripting (XSS), розроблено спеціалізовану функцію-обгортку, яка базується на методі htmlspecialchars() [25]. Це забезпечує екранування небезпечних спеціальних символів HTML, перетворюючи їх на безпечні сутності, що виключає можливість виконання шкідливого коду в браузері користувача.

Таблиця 1.4

Порівняння серверних технологій для реалізації вебзастосунку

Критерій	PHP + MySQL	Node.js + PostgreSQL	Django + PostgreSQL
Складність запуску	Низька	Середня	Середня
Серверний рендеринг	Природний, без збірника	Потребує шаблонізатора	Вбудовані шаблони
Навчальна прозорість	Висока, один файл = один сценарій	Середня	Середня
Локальне розгортання	XAMPP / OpenServer	Потребує npm, збірки	Потребує virtualenv, pip
Підсумок	Обрано	Не обрано	Не обрано

Детальне практичне застосування кожного елемента обраного стеку технологій, їхня безпосередня роль у програмній реалізації вебзастосунку ATLANT, а

також обґрунтування доцільності їх використання для вирішення конкретних завдань проєкту наведено у підсумковій таблиці (табл. 1.5).

Таблиця 1.5

Використані інструменти та їх практичне обґрунтування

Інструмент	Де використано	Практичне обґрунтування
PHP	Уся серверна логіка, сесії, форми	Природна інтеграція з Apache, доступне середовище розгортання
PDO	Усі SQL-запити до MySQL	Підготовлені запити, захист від SQL-ін'єкцій, винятки PDO::ERRMODE_EXCEPTION
MySQL / InnoDB	Зберігання всіх даних системи	Зовнішні ключі, підтримка транзакцій (commit)
hash('sha256')	Реєстрація та автентифікація	Хешування без зберігання відкритих паролів у базі даних
CSS Grid & @media	Каталог товарів, форми	Створення адаптивної сітки без використання зовнішніх фреймворків
XAMPP / OpenServer	Локальне розгортання	Пакет Apache, PHP і MySQL без необхідності ручного налаштування

Важливою перевагою обраного технологічного стеку є відсутність жорсткої залежності від надскладної інфраструктури, дорогих хмарних сервісів чи великої кількості сторонніх залежностей (vendor lock-in), що робить застосунок легким для розгортання та супроводу. Кожен компонент системи було обрано з урахуванням критеріїв стабільності, передбачуваності та легкості налагодження

(troubleshooting). Використані інструменти зокрема PHP та MySQL мають вичерпну офіційну документацію, потужну підтримку LTS (Long Term Support) версій, активну спільноту розробників та десятиліття успішного практичного застосування у корпоративному секторі веброботи [14].

Висновки до першого розділу

У першому розділі проведено комплексний аналіз предметної області ринку харчових добавок та спортивного харчування в Україні. Встановлено, що цей ринок характеризується широким асортиментом товарів із варіантами смаків і фасувань, різноманітністю цільових аудиторій та специфічними вимогами до каталогізації і фільтрації. Сформоване дослідження дозволило чітко визначити основні бізнес-процеси магазину та потреба адміністратора в управлінні залишками, що безпосередньо вплинуло на формування функціональних вимог до платформи ATLANT.

Аналіз конкурентних рішень показав, що жоден з розглянутих аналогів ні великий маркетплейс Rozetka, ні спеціалізовані магазини спортивного харчування ВСАА.ua та Belok.ua — не забезпечує оптимального балансу між тематичним каталогом, зручністю покупки і повноцінним контролем продавця. Вебплатформа ATLANT розроблена саме для усунення виявлених недоліків і надання власнику магазину незалежного, функціонального та зручного інструменту онлайн-торгівлі без сторонніх обмежень.

Обґрунтовано вибір клієнт-серверної архітектури з серверним рендерингом PHP як оптимальної для даного проєкту. В ході роботи було детально показано переваги цього підходу над SPA-рішеннями з точки зору простоти розгортання, кросбраузерності інтерфейсу, гарантованої сумісності з більшістю хостингів і прозорості вихідного коду для навчальних цілей.

Визначено та обґрунтовано стек технологій: PHP 8, MySQL 8, HTML5, CSS3, JavaScript, phpMyAdmin та OpenServer.

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

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ATLANT

2.1. Визначення варіантів використання та структури системи

Перш ніж проєктувати базу даних і реалізовувати модулі, необхідно чітко визначити, хто користується системою і що саме кожен із них може робити. У платформі ATLANT передбачено два рівні взаємодії: покупець (неzareєстрований відвідувач) та адміністратор. Межі між ролями закладені не лише на рівні інтерфейсу, а безпосередньо в серверній логіці через перевірки сесії та бази даних.

Покупець — це будь-яка людина, яка відкрила сайт. Йому доступний повний перегляд каталогу харчових добавок, пошук і фільтрація товарів за категорією, брендом і ціллю використання (набір маси, схуднення, витривалість), перегляд детальної картки товару з описом, вибір конкретного варіанту (смак та вага фасування), а також додавання позицій до кошика. Оскільки система орієнтована на швидкі продажі, реєстрація покупця не вимагається: всі дані для доставки та оплати вводяться безпосередньо під час оформлення замовлення (checkout), а кошик зберігається тимчасово в межах активної сесії користувача [1].

Адміністратор — це обліковий запис у таблиці admins. Він отримує доступ до окремого захищеного каталогу /admin/, недоступного для звичайних покупців. Його функції охоплюють повний спектр CRUD-операцій для товарів та їхніх варіантів, керування категоріями, перегляд усіх оформлених замовлень і зміну їхніх статусів, а також перегляд аналітики продажів. Для забезпечення високого рівня безпеки всі паролі адміністраторів зберігаються в базі даних у хешованому вигляді із використанням сучасних алгоритмів захисту. Крім того, інтерфейс панелі керування є повністю адаптивним, що дозволяє власнику магазину оперативно обробляти замовлення як з комп'ютера, так і з мобільного пристрою. Будь-яка спроба відкрити адміністративну сторінку без відповідної активної сесії завершується автоматичним перенаправленням на сторінку входу [8]. Розподіл ролей наведено у таблиці (табл. 2.1).

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

Ролі та їхні можливості в системі

Актор	Права та можливості	Технічна реалізація
Покупець	Каталог, пошук, фільтри, картка товару з варіантами, сесійний кошик, оформлення замовлення.	Доступ без авторизації, \$_SESSION['cart']
Адміністратор	Керування товарами, варіантами, категоріями, замовленнями, перегляд аналітики.	Перевірка \$_SESSION['admin_id']

Після визначення ролей необхідно описати конкретні сценарії взаємодії. Кожен такий сценарій складається з трьох елементів: актор, який виконує дію, обов'язкова передумова для початку та результат, який система зберігає або повертає.

Вхід в акаунт відбувається через форму на сторінці login.php. Адміністратор вводить логін і пароль. Сервер знаходить запис за логіном через підготовлений PDO-запит і перевіряє пароль за допомогою алгоритму хешування hash('sha256'). У разі збігу згенерованого хешу з тим, що зберігається в базі даних, до сесії записується admin_id, який визначає доступні функції та відкриває панель керування [1].

Додавання товару до кошика виконується через взаємодію з масивом \$_SESSION['cart']. Перед збереженням сервер фіксує ідентифікатор товару, обраний варіант (смак/вага), ціну та кількість. Якщо така позиція вже є в кошику, її кількість збільшується; інакше створюється новий запис у межах поточної сесії користувача [1].

Оформлення замовлення найбільш критичний сценарій системи. Файл checkout.php виконує валідацію введених контактних даних (ПІБ, телефон, місто, адреса, спосіб доставки та оплати), після чого обробляє збереження інформації у

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

межах транзакції PDO (beginTransaction() та commit()). Система створює запис замовлення в таблиці orders, додає всі позиції з кошика до таблиці order_items та повністю очищає кошик користувача після успішного збереження [1, 2].

Керування асортиментом виконується адміністратором і передбачає додавання нових харчових добавок, завантаження їхніх зображень, встановлення маркетингових позначок («Новинка», «Хіт») та управління статусом видимості (активний/прихований), що дозволяє прибрати товар з вітрини без його фізичного видалення з бази. Перелік усіх варіантів використання наведено у таблиці (табл. 2.2).

Діаграму варіантів використання платформи ATLANT наведено на рисунку (рис. 2.1).

Таблиця 2.2

Варіанти використання системи ATLANT

Варіант використання	Актор	Передумова	Результат
Перегляд каталогу	Покупець	Відкрита сторінка каталогу	Відображено сітку товарів
Пошук і фільтрація	Покупець	Введено GET-параметри (ціль, бренд, категорія)	Сформовано динамічний SQL-запит, показано результат
Вибір варіанту товару	Покупець	Відкрито картку товару	Оновлено ціну відповідно до обраного смаку або ваги

Додавання до кошика	Покупець	Наявність товару	Створено або оновлено запис у \$_SESSION['cart']
Оформлення замовлення	Покупець	Непорожній кошик, заповнені поля форми	Створено записи в orders та order_items, очищено кошик
Вхід адміна	Адміністратор	Правильний логін та пароль	Звірено sha256 хеш, встановлено сесію admin_id
Керування товарами (CRUD)	Адміністратор	Наявність ролі адміністратора	Додано/оновлено/видалено записи в products
Обробка замовлень	Адміністратор	Наявність ролі адміністратора	Змінено статус замовлення (нове, в обробці, виконано)

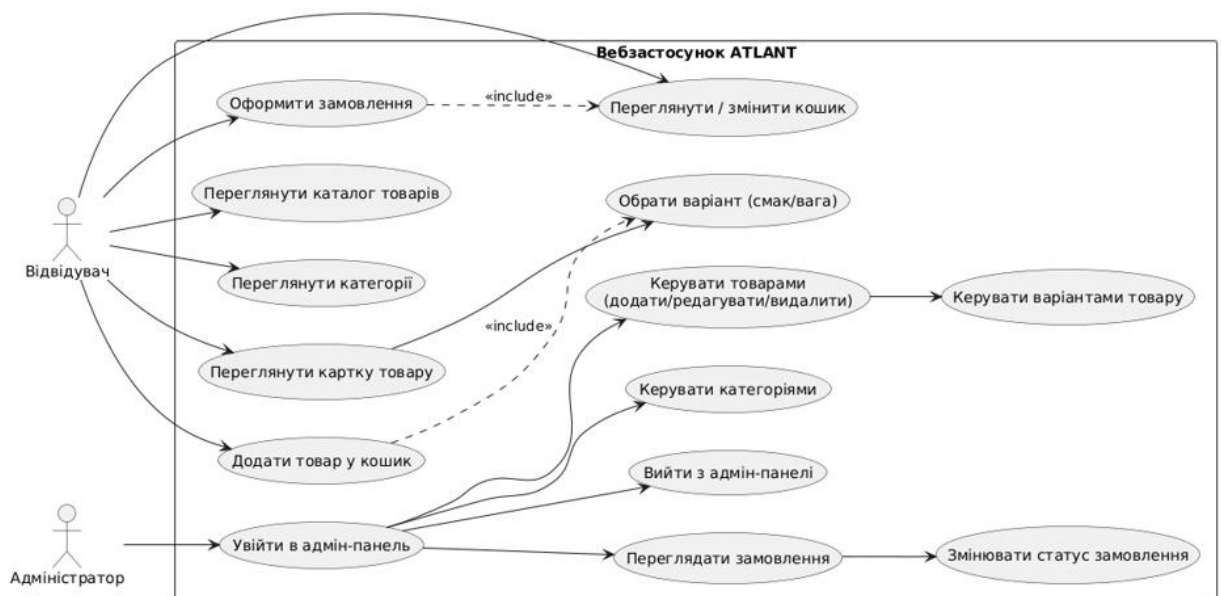


Рис. 2.1 – Діаграма варіантів використання

Діаграма варіантів використання відображає основні сценарії взаємодії користувачів із системою та демонструє функціональні можливості ATLANT. За допомогою даної діаграми було сформовано загальне уявлення про структуру майбутньої системи, визначено основних учасників процесів та перелік операцій,

які вони можуть виконувати відповідно до своїх повноважень. Крім того, діаграма дозволяє наочно представити межі системи, взаємозв'язки між окремими функціями та особливості реалізації бізнес-логіки інтернет-магазину. Її використання на етапі проектування сприяє кращому розумінню вимог замовника та забезпечує узгодженість між функціональними потребами покупців і технічними рішеннями, що впроваджуються в програмному продукті. Для подальшого посилення основним сценаріям покупця присвоєно такі ідентифікатори: UC-Catalog (перегляд тематичного каталогу), UC-Search (пошук і фільтрація товарів), UC-Cart (керування кошиком), UC-Checkout (оформлення замовлення); ключовій сутності бази даних товарів відповідає позначення ERD-Products. Нефункціональні вимоги наведено у таблиці (табл. 2.3).

Таблиця 2.3

Нефункціональні вимоги до системи ATLANT

Критерій	Вимога	Очікуваний результат
Сприйняття	Інтерфейс зрозумілий без додаткового навчання	Освоєння основних функцій покупцем до 5 хв
Надійність	Стабільна обробка замовлень і операцій із кошиком	Атомарність checkout через PDO транзакції, відсутність втрат даних
Гнучкість	Зручна робота з товарами, що мають різні параметри	Підтримка варіантів (смак, вага) без дублювання карток товарів
Безпека	Захист від несанкціонованого доступу та ін'єкцій	Підготовлені запити PDO, перевірка сесії на кожній сторінці /admin/
Розгортання	Запуск без складного середовища збірки	Копіювання файлів + імпорт SQL достатньо для старту в XAMPP / OpenServer

Наведені в таблиці 2.3 нефункціональні вимоги визначають ключові характеристики якості вебзастосунку ATLANT та впливають на вибір архітектурних і технологічних рішень під час його розробки. Забезпечення зрозумілого інтерфейсу спрямоване на швидке освоєння системи покупцями без необхідності обов'язкової реєстрації. Вимоги до надійності даних реалізуються за рахунок використання транзакційного механізму обробки замовлень. Безпека системи досягається шляхом застосування підготовлених SQL-запитів, надійного алгоритму хешування паролів адміністратора та розмежування прав доступу.

2.2. Проєктування бази даних

Реляційна база даних є центральним елементом платформи ATLANT, оскільки вся бізнес-логіка системи — від формування каталогу до оформлення замовлення базується на узгодженому стані даних у взаємопов'язаних таблицях. При проєктуванні структури ставилася мета відобразити реальні предметні зв'язки між сутностями електронної торгівлі харчовими добавками, забезпечити посила- льну цілісність даних на рівні СУБД і мінімізувати надлишковість за правилами третьої нормальної форми (3НФ) [2].

База даних містить шість таблиць. Центральними сутностями є *admins* та *products*. Перша описує облікові записи адміністраторів із захищеними хешами паролів для доступу до панелі керування, друга — повний каталог товарів із усіма атрибутами, необхідними для відображення, фільтрації та контролю залишків. Усі інші таблиці утворюють зв'язки навколо сутності товарів та процесу покупки.

Таблиці *categories* та *product_variants* реалізують логічну структуру тематичного каталогу. Таблиця *categories* групує товари за їхнім призначенням, а *product_variants* забезпечує гнучке зберігання різних смаків та фасувань (ваги) одного товару. Це дозволяє уникнути дублювання основного опису, фотографії та бренду добавки, якщо вона продається у кількох варіантах розміру чи смаку.

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

Замовлення свідомо розділено на дві таблиці: заголовок orders і позиції order_items. Така структура є загальноприйнятою практикою проєктування систем електронної торгівлі, оскільки одне замовлення може містити довільну кількість різних товарів [15]. У таблиці orders зберігаються контактні дані покупця, адреса, поточний статус та загальна сума. У таблиці order_items кожен рядок відповідає одній товарній позиції з фіксованою назвою варіанту, кількістю та ціною на момент оформлення. Копіювання ціни та текстової назви до order_items є принциповим рішенням: подальша зміна ціни або навіть видалення товару в каталозі не вплине на історичну цілісність уже оформлених замовлень, що забезпечує коректність звітності.

Усі залежні дочірні таблиці проєкту використовують зовнішні ключі (Foreign Keys) з обов'язковою директивою каскадного оновлення та видалення ON DELETE CASCADE [2]. Практично це означає, що у разі видалення певної категорії з бази даних автоматично будуть видалені всі пов'язані з нею товари, а при видаленні конкретного товару — безповоротно очищаються всі його доступні торгові варіанти (product_variants), що містили інформацію про вагу та смаки. Такий архітектурний підхід повністю забезпечує автоматичну референційну цілісність і несуперечність даних безпосередньо на рівні СУБД MySQL, усуваючи потребу в реалізації додаткової складної програмної логіки для контролю та пошуку «сирітських» (orphan) записів на стороні сервера мовою PHP [2, 26].

Вся спроектована схема реляційної бази даних разом із її початковим наповненням створюється та розгортається за допомогою єдиного консолідованого файлу структури database.sql. Цей SQL-скрипт спочатку послідовно ініціалізує створення структури всіх необхідних таблиць та індексів, а потім автоматично додає дефолтний демонстраційний обліковий запис адміністратора, набір базових категорій харчових добавок та первинні товари для комплексного тестування працездатності системи. Використання сучасного порівняння та кодування utf8mb4_unicode_ci для абсолютно всіх таблиць, текстових полів та зв'язків гарантує коректне відображення й надійне зберігання українського тексту, специфічних символів і смайлів без ризику спотворення чи втрати даних при копіюванні

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

бази [2]. Повну структуру розробленої бази даних (ER-діаграму) наведено на рисунку (рис. 2.2).

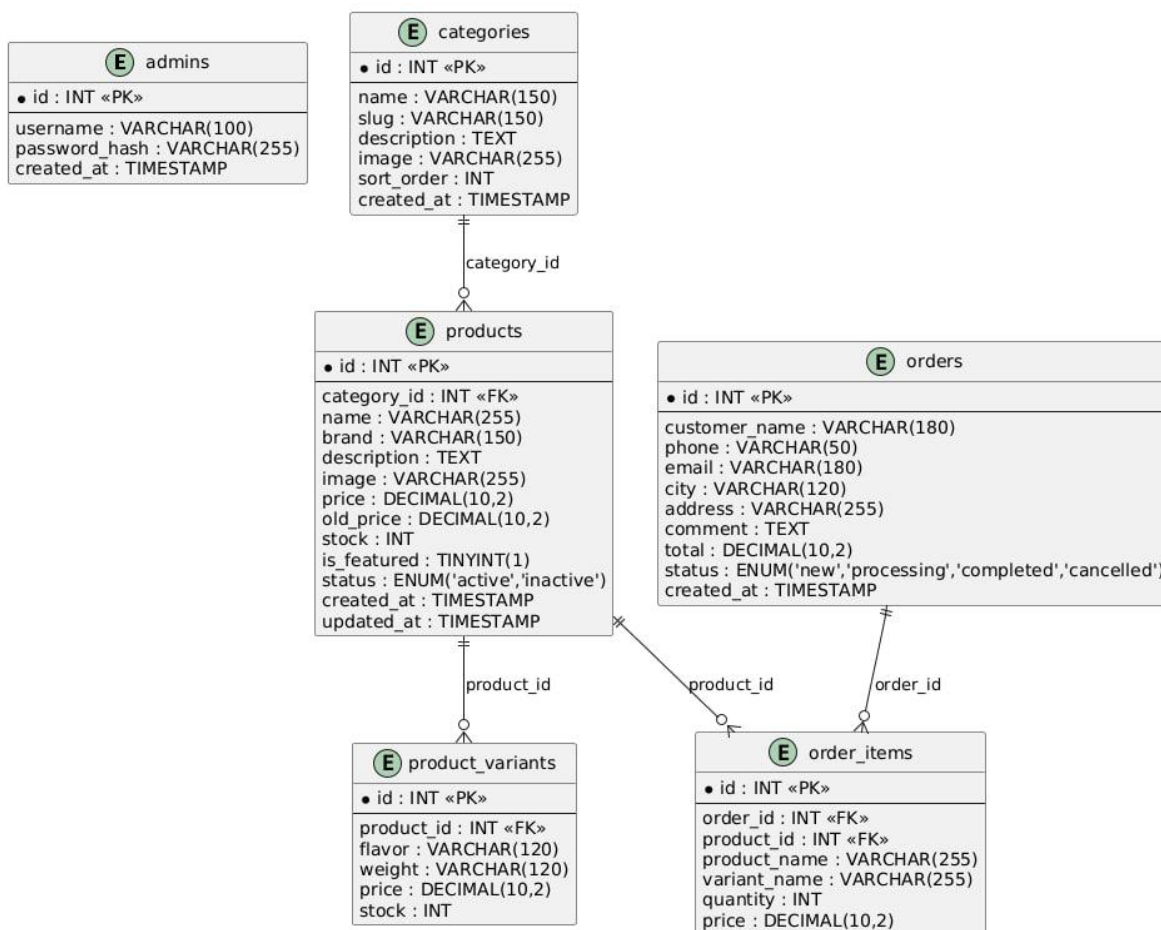


Рис. 2.2 – ER-модель бази даних ATLANT

Повний склад розроблених таблиць реляційної бази даних, їх безпосереднє функціональне призначення у системі, а також основні типи логічних зв'язків між ними детально наведено у таблиці (табл. 2.4). Спроектowana реляційна структура побудована таким чином, щоб максимально точно й ефективно відобразити всі ключові сутності предметної області інтернет-магазину харчових добавок. Це дозволяє забезпечити коректну, стабільну та швидку взаємодію між каталогом товарного асортименту, механізмом обліку динамічних варіантів продукції та логічно завершеним процесом оформлення замовлень покупцями. Оптимізація структури на етапі проєктування гарантує мінімальну надмірність даних та виключає появу аномалій при модифікації записів.

Таблиці бази даних та їх призначення

Таблиця	Призначення	Ключові зв'язки
admins	Облікові записи співробітників магазину	Незалежна таблиця
categories	Ієрархія та групування товарів	1:N з таблицею products
products	Основна інформація про товари в каталозі	N:1 з categories, 1:N з product_variants
product_variants	Варіанти товару (смак, вага, ціна)	N:1 з таблицею products (FK product_id)
orders	Загальні контактні дані за замовлення	1:N з таблицею order_items
order_items	Позиції замовлення з фіксованою ціною	N:1 з таблицею orders (FK order_id)

Детальна структура таблиці products наведена у таблиці (табл. 2.5). Вона виступає центральним ядром та основою інформаційного каталогу вебплатформи ATLANT. Дана таблиця містить повний набір суттєвих атрибутів, необхідних для коректного відображення кожної товарної позиції в користувацькому інтерфейсі, її точної фільтрації за категоріями, гнучкого сортування за вартістю чи новинками, а також для оперативного контролю поточних складських залишків. Крім того, окремо виділені поля забезпечують підтримку важливих візуальних та комерційних функцій, таких як виведення акційних цін для стимулювання продажів, відображення маркетингових міток та автоматичне приховання позицій, яких тимчасово немає в наявності.

Структура таблиці products

Поле	Тип	Призначення
id	INT, PK, AUTO_INCREMENT	Унікальний ідентифікатор товару
category_id	INT, FK	Посилання на категорію товару
name	VARCHAR(255), NOT NULL	Назва добавки для відображення і пошуку
brand	VARCHAR(150)	Бренд виробника товару
description	TEXT	Детальний опис для картки товару
price	DECIMAL(10,2), NOT NULL	Базова ціна продажу
old_price	DECIMAL(10,2), NULL	Попередня ціна для відображення знижки
stock	INT, NOT NULL	Кількість одиниць на складі
image	VARCHAR(255)	Шлях до файлу зображення товару
status	ENUM('active','inactive')	Статус видимості товару в каталозі

Таблиця product_variants, наведена у таблиці (табл. 2.6), суттєво розширює функціональні можливості обліку товарів і є критично важливою для ніші спортивного харчування, де один продукт майже завжди має декілька різних смаків або варіантів ваги пакування. Таке архітектурне рішення дозволяє динамічно прив'язувати унікальну ціну та точну кількість залишків до кожного окремого фасування в межах однієї сторінки продукту. Завдяки цьому вдається повністю

уникнути критичного дублювання інформації в базі даних та забезпечити покупцеві зручне перемикання між характеристиками без перезавантаження вебсторінки.

Таблиця 2.6

Структура таблиці product_variants

Поле	Тип	Призначення
id	INT, PK, AUTO_INCREMENT	Унікальний ідентифікатор варіанту
product_id	INT, FK	Посилання на основний товар
flavor	VARCHAR(120)	Назва смаку (наприклад, "Шоколад")
weight	VARCHAR(120)	Вага або кількість порцій
price	DECIMAL(10,2), NOT NULL	Специфічна ціна для даного об'єму
stock	INT, NOT NULL	Фактичний залишок конкретного варіанту

Структура таблиці orders наведена у (табл. 2.7). Вона є центральною ланкою системи, що відповідає за збереження даних про оформлені замовлення та реалізацію фінального етапу покупки. Таблиця фіксує контактні дані покупця на момент створення замовлення, а також містить поля для логістики та коментарів, що забезпечує адміністратору вичерпну інформацію для швидкої обробки замовлень. Використання типів даних DECIMAL для фінансових показників та ENUM для статусів обробки гарантує точність розрахунків і цілісність логіки життєвого циклу замовлення.

Структура таблиці orders

Поле	Тип	Призначення
id	INT, PK, AUTO_INCREMENT	Унікальний номер замовлення
customer_name	VARCHAR(180)	Ім'я отримувача на момент оформлення
phone	VARCHAR(50)	Контактний телефон для зв'язку
email	VARCHAR(180)	Електронна пошта для сповіщень
city	VARCHAR(120)	Місто доставки
address	VARCHAR(255)	Точна адреса або номер відділення пошти
comment	TEXT	Довільний коментар покупця до замовлення
total	DECIMAL(10,2)	Загальна розрахована сума замовлення
status	ENUM('new', 'processing', ...)	Поточний статус обробки

Наведені в таблицях 2.4–2.7 характеристики демонструють структуру основних сутностей бази даних ATLANT та їхнє призначення в межах функціонування системи. Центральне місце займають таблиці products, product_variants та orders, які забезпечують гнучке зберігання інформації про асортимент харчових добавок і процес реєстрації замовлень. Допоміжні таблиці admins, categories та order_items реалізують додаткові функціональні можливості вебзастосунку та забезпечують цілісність бізнес-процесів магазину.

2.3. Проектування та реалізація алгоритмів роботи

Логіка роботи платформи ATLANT не зосереджена в одному громіздкому модулі, а розподілена між окремими PHP-файлами відповідно до їхньої відповідальності. Алгоритм автентифікації реалізовано в `admin/login.php`, логіка кошика опрацьовується через масив сесії у відповідних обробниках, а критичний алгоритм оформлення замовлення повністю інкапсульований у файлі `checkout.php`. Такий підхід дозволяє змінювати код одного сценарію без ризику вплинути на інші, а також робить кожен файл зрозумілим для самостійного аналізу та розширення [1].

Алгоритм автентифікації адміністратора починається з отримання введеного логіна (`username`) та пароля із надісланого POST-запиту форми авторизації. Далі виконується безпечний пошук відповідного запису в таблиці `admins` через підготовлений PDO-запит із використанням параметризованих маркерів, що повністю виключає ризик уразливостей типу SQL-ін'єкцій [1]. Якщо запис успішно знайдено, система генерує хеш від введеного користувачем пароля за допомогою алгоритму `hash('sha256')` та суворо порівнює його з оригінальним хешем, що статично зберігається в базі даних [1]. У разі успішного збігу та автентифікації, до глобального масиву сесії записується унікальне значення `admin_id`. Саме цей ідентифікатор у подальшому визначає легітимне право доступу: після його встановлення адміністратор автоматично потрапляє до захищеної панелі керування. Якщо пошук у базі даних не повернув запису або згенерований хеш пароля не збігся, система видає загальне повідомлення про помилку входу без уточнення конкретної причини (неправильний логін чи пароль), що значно ускладнює зловмисникам визначення існуючих акаунтів методом перебору [8].

Оскільки розроблювана платформа ATLANT орієнтована на максимально швидкі продажі харчових добавок, складний алгоритм обов'язкової реєстрації покупців був навмисно виключений для спрощення клієнтського шляху та підвищення конверсії магазину. Натомість усю увагу зосереджено на захисті панелі управління, де реалізовано жорсткий алгоритм перевірки прав доступу до адміністративної частини. На початку виконання кожного скрипта і файлу в каталозі

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		36

/admin/ викликається вбудована перевірка наявності активної сесії \$_SESSION['admin_id'], яка підтверджує успішний статус входу [1]. Будь-яка не-санкціонована спроба відкрити або викликати ці сторінки без попередньої авторизації миттєво переривається системою та завершується автоматичним перенаправленням користувача на сторінку входу. Діаграму активності процесу автентифікації та захисту доступу наведено на рисунку (рис. 2.3).

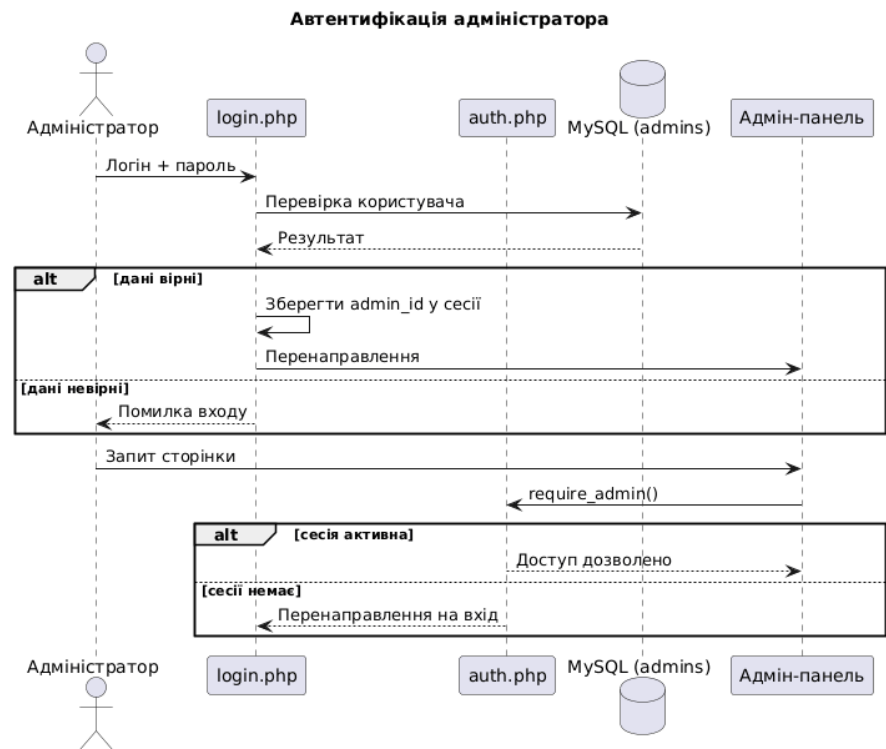


Рис. 2.3 – Схема процесу автентифікації та перевірки доступу адміністратора

Алгоритм формування каталогу та пошуку товарів динамічно будує SQL-запит залежно від переданих GET-параметрів: пошуковий рядок, категорія, бренд, ціль використання (набір маси, схуднення тощо) та ціновий діапазон. Для кожного непорожнього параметра до масиву умов WHERE додається відповідний вираз, а значення передається до масиву параметрів для безпечного виконання через prepare/execute [1].

Текстовий пошук за назвою або брендом реалізується через конструкцію LIKE із символом підстановки. Завдяки такому підходу всі вхідні дані надійно

екрануються, що виступає ключовим захистом від SQL-ін'єкцій у динамічних вибірках [8]. Діаграму активності алгоритму пошуку та фільтрації каталогу наведено на рисунку (рис. 2.4).

Алгоритм пошуку і фільтрації товарів

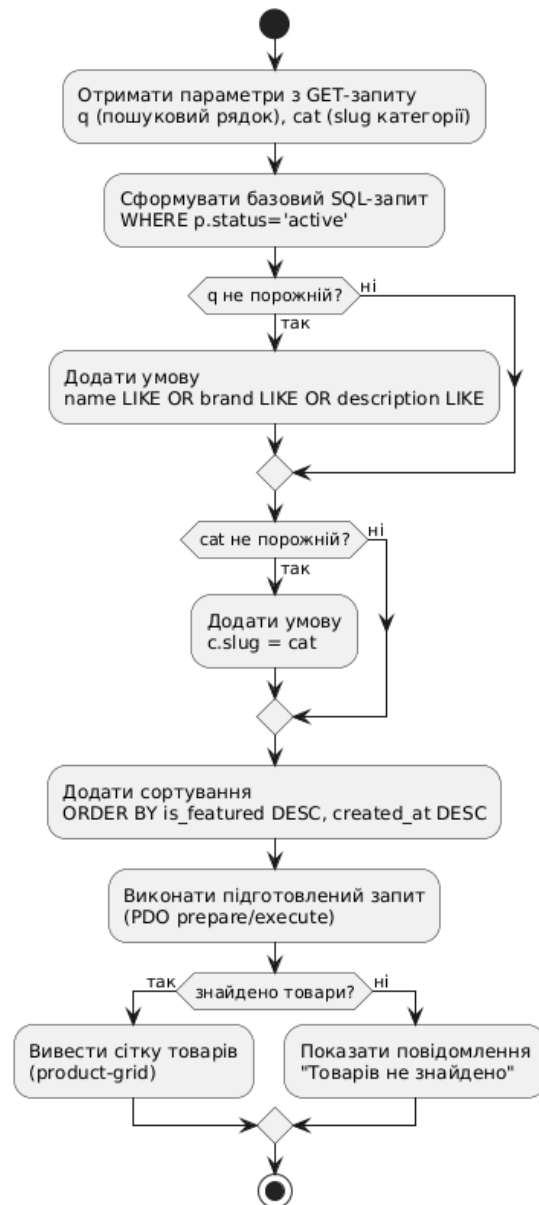


Рис. 2.4 – Схема алгоритму пошуку і фільтрації товарів

Алгоритм роботи з кошиком базується виключно на використанні PHP-сесій без проміжних збережень у базі даних, що зменшує навантаження на сервер. Структура \$_SESSION['cart'] являє собою асоціативний масив, де унікальним ключем виступає комбінація ідентифікатора товару та ідентифікатора обраного

варіанту (наприклад, `product_id_variant_id`). Під час додавання товару сервер перевіряє наявність позиції та актуальний залишок у таблиці `products` або `product_variants` [2]. Якщо вибраний товар у заданому смаку чи фасуванні вже присутній у кошику, система не створює новий запис, а лише збільшує кількість наявної позиції в межах доступного складського залишку. Загальна сума кошика перераховується динамічно при будь-якій зміні кількості або видаленні позиції.

Найбільш критичним з точки зору цілісності та узгодженості даних в архітектурі платформи ATLANT є алгоритм транзакційного оформлення замовлення. Він повинен одночасно надійно зафіксувати загальні контактні дані клієнта, зберегти всі обрані товарні позиції з урахуванням обраних смаків і фасувань, а також коректно очистити поточний кошик таким чином, щоб повністю унеможливити ризик часткового збереження інформації у разі збою. Після надходження POST-запиту з клієнтської форми `checkout.php` обов'язково виконується сувора серверна валідація всіх обов'язкових полів, включаючи перевірку ПІБ, номера телефону, міста та адреси доставки. Далі всі послідовні операції запису та модифікації об'єднуються в єдину атомарну транзакцію PDO через виклик методу `beginTransaction()`, що дозволяє захистити базу даних від аномалій [2].

У межах відкритої транзакції система спочатку виконує створення нового головного запису в таблиці `orders` та миттєво отримує згенерований ідентифікатор замовлення за допомогою функції `lastInsertId()`. Далі в циклі відбувається послідовне додавання всіх окремих позицій із сесійного кошика користувача до залежної таблиці `order_items`. На цьому етапі назва товару та його конкретного варіанту (смак/вага) копіюється і зберігається у текстовому вигляді, а ціна фіксується безпосередньо на момент покупки, що гарантує незмінність історичної вартості замовлення у разі майбутнього редагування каталогу. Після успішного запису всіх дочірніх позицій та коригування складських залишків виконується повне очищення кошика користувача через `unset($_SESSION['cart'])` та успішне завершення транзакції за допомогою методу `commit()` [1]. Якщо на будь-якому з етапів виникає системна помилка, виконується автоматичний відкат усіх змін че-

					ПІЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		39

рез метод rollBack(), що гарантує повернення бази даних до початкового стабільного стану. Повну діаграму активності процесу оформлення замовлення наведено на рисунку (рис. 2.5).

Алгоритм оформлення замовлення



Рис. 2.5 – Схема алгоритму оформлення замовлення

Для забезпечення цілісності даних процес оформлення замовлення розділено на послідовні кроки. Основні етапи розробленого алгоритму наведено у (табл. 2.8).

Таблиця 2.8

Етапи алгоритму оформлення замовлення та захисні механізми

Етап	Операція	Захисний механізм
1	Завантаження позицій кошика	Перевірка масиву \$_SESSION['cart']
2	Перевірка непорожнього кошика	Якщо порожній — перенаправлення до каталогу
3	Валідація контактних даних	Очищення через trim() і перевірка обов'язкових полів
4	beginTransaction()	Початок атомарної транзакції бази даних
5	INSERT до orders	Підготовлений запит із прив'язкою параметрів клієнта
6	INSERT до order_items у циклі	Фіксація текстової назви та ціни на момент оформлення
7	Очищення сесії	Видалення даних кошика з оперативної пам'яті сервера
8	commit()	Завершення транзакції та остаточне збереження

Окрім базових алгоритмів, система реалізує додаткові адміністративні функції. Алгоритм зміни статусу замовлення дозволяє адміністратору перевести замовлення на наступний етап обробки. PHP-скрипт перевіряє, чи є переданий ідентифікатор числовим, а новий статус допустимим (new, processing, completed, cancelled), після чого виконує параметризований UPDATE.

2.4. Особливості програмної реалізації вебзастосунку

У цьому підрозділі детально розглянуто архітектурну логіку роботи та специфіку програмного коду, який безпосередньо відповідає за забезпечення головних функцій серверної частини вебзастосунку ATLANT. Оскільки повний вихідний код розробленого проєкту наводиться у додатках до кваліфікаційної роботи, у цьому розділі зосереджено увагу виключно на найбільш важливих технічних рішеннях. До них належать модульна організація файлової структури, концепція централізованого підключення до бази даних, реалізація спільних системних компонентів, модуль інтерактивного каталогу, алгоритм транзакційного оформлення замовлень та комплексний захист адміністративного контуру. Програмна реалізація організована за модульним принципом, що забезпечує максимальне перевикористання коду і спрощує внесення змін у майбутньому.

Конфігураційний файл `config/db.php` виступає головною точкою входу для підключення до бази даних та виконує роль ядра застосунку. Програмне з'єднання створюється з обов'язковим атрибутом `PDO::ERRMODE_EXCEPTION`, що дозволяє ефективно перехоплювати будь-які помилки MySQL у блоці `try/catch`. Усі інші сторінки системи підключають цей файл через конструкцію `require_once` і використовують єдиний глобальний об'єкт `$pdo` [1]. Для виведення користувацьких даних у HTML-розмітку систематично використовується вбудована функція `htmlspecialchars()`, що формує надійний захист від міжсайтового скриптингу (XSS) у всіх шаблонах системи [1, 8].

Спільні структурні шаблони `header.php` та `footer.php`, розміщені в каталозі `includes/`, виконують значно ширший спектр задач, ніж просте виведення статичної розмітки сайту. Шаблон шапки (`header`) при кожному завантаженні сторінки аналізує стан масиву `$_SESSION['cart']` та динамічно виводить актуальну кількість товарів у лічильнику кошика. Крім того, він містить адаптивне навігаційне меню, яке на мобільних пристроях замінюється інтерактивною гамбургер-кнопкою. Шаблон підвалу (`footer`) логічно завершує загальну HTML-структуру документів та здійснює підключення базових клієнтських скриптів. Структуру основних файлів проєкту наведено у таблиці (табл. 2.9).

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						42
Змн.	Арк.	№ докум.	Підпис	Дата		

Структура файлів проєкту ATLANT та їх призначення

Файл / Каталог	Призначення
index.php	Головна сторінка, каталог товарів із пошуком та фільтрацією за категорією
product.php	Картка товару з можливістю вибору варіанту (смак/вага)
cart.php	Кошик (перегляд позицій, зміна кількості, підрахунок суми)
checkout.php	Форма оформлення замовлення та її серверна обробка
config/db.php	Налаштування параметрів підключення до MySQL (PDO)
includes/	Каталог спільних компонентів (header.php, footer.php)
admin/index.php	Захищений дашборд адміністратора зі статистикою
admin/products.php, product-form.php	Керування номенклатурою товарів (CRUD-операції)
admin/orders.php	Перегляд та зміна статусів оформлених замовлень
assets/css/style.css	Каскадна таблиця стилів із набором CSS-змінних у :root
assets/js/main.js	Клієнтський скрипт (динамічне додавання варіантів товару в формі)
uploads/products/	Директорія для збереження завантажених зображень

Модуль інтерактивного каталогу реалізовано в межах двох файлів. Головна сторінка index.php виконує єдиний параметризований SQL-запит до бази да-

них, який одночасно відповідає за пошук, фільтрацію за категорією та сортування товарів за маркетинговою позначкою (`is_featured`) і датою додавання. Усі вхідні дані передаються виключно через параметризований метод `execute()`, що повністю виключає можливість проведення ін'єкцій [8]. Сторінка детального перегляду `product.php` завантажує інформацію про конкретний товар та виводить усі доступні варіанти з таблиці `product_variants`.

Адаптивний користувацький інтерфейс реалізовано за допомогою сучасного стандарту CSS3 (модулі Flexbox та CSS Grid) із використанням системи CSS-змінних, оголошених у блоці `:root` файлу `style.css`, що дозволяє швидко змінювати кольорову палітру без редагування окремих правил. Мова JavaScript у проєкті використовується мінімально і лише для покращення користувацького досвіду: єдиний клієнтський скрипт `main.js` відповідає за динамічне додавання й видалення полів варіанта товару (смак, вага, ціна, кількість) у формі адміністративної панелі без перезавантаження сторінки [3, 4].

Найбільш відповідальним у системі є модуль оформлення замовлення `checkout.php`. При GET-запиті він відображає форму для введення даних клієнта, а при POST-запиті запускає їх валідацію. Збереження замовлення відбувається в межах транзакції PDO (`beginTransaction/commit`), що гарантує запис усіх позицій кошика у таблицю `order_items` як єдину узгоджену операцію [1, 2].

Усі адміністративні файли ізольовані в каталозі `admin/` і реалізують класичний CRUD-цикл із можливістю завантаження зображень. Під час завантаження фотографій PHP-скрипт перевіряє MIME-тип файлу через `mime_content_type()` [21] за білим списком (`jpeg, png, webp, gif`) і генерує унікальне ім'я на основі випадкових байтів через `random_bytes()`, що запобігає завантаженню шкідливого коду. Аутентифікація адміністратора базується на алгоритмі хешування `hash('sha256')`, що виключає зберігання паролів у відкритому вигляді [1].

Висновки до другого розділу

Було визначено ролі користувачів платформи ATLANT та описано основні сценарії їхньої взаємодії з вебзастосунком. Для незареєстрованого покупця та авторизованого адміністратора було встановлено чіткий перелік доступних функцій. Також сформовано діаграму варіантів використання системи, що дозволило формалізувати бізнес-логіку електронної торгівлі, включаючи перегляд каталогу, управління сесійним кошиком, транзакційне оформлення замовлення та виконання CRUD-операцій з товарами в панелі керування.

Було виконано проєктування реляційної бази даних вебзастосунку ATLANT. Розроблена схема складається із шести взаємопов'язаних таблиць: `admins`, `categories`, `products`, `product_variants`, `orders` та `order_items`. Використання зовнішніх ключів забезпечило цілісність даних і коректну взаємодію між усіма сутностями системи: для `product_variants` і `order_items` застосовано каскадне видалення (ON DELETE CASCADE) при видаленні батьківського товару чи замовлення, а для зв'язку `products` із `categories` встановлення NULL (ON DELETE SET NULL), що дозволяє видаляти категорію без втрати самих товарів. Окрема таблиця варіантів дозволила зручно реалізувати облік різних смаків і фасувань харчових добавок.

Розроблено та оптимізовано основні алгоритми роботи вебзастосунку. Описано логіку динамічної фільтрації каталогу через підготовлені запити PDO, алгоритм роботи сесійного кошика, транзакційний механізм збереження інформації про замовлення, а також алгоритм аутентифікації адміністратора з використанням алгоритму хешування SHA-256. Враховано сценарії перевірки вхідних даних для захисту від вразливостей.

Розглянуто особливості програмної реалізації вебзастосунку ATLANT. Описано модульну файлову структуру проєкту, принципи організації спільного коду через підключення каталогу `includes/`, централізоване налаштування бази даних та механізми роботи адміністративної панелі.

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						45
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З СИСТЕМОЮ

3.1. Порядок встановлення та налаштування системи

Платформу ATLANT реалізовано на основі класичної клієнт-серверної архітектури з використанням вебсервера Apache, серверної мови PHP та СУБД MySQL. Розгортання системи не потребує складних налаштувань серверної інфраструктури архітектура розрахована на роботу у локальному середовищі OpenServer Panel або XAMPP, які надають усі необхідні програмні компоненти. На відміну від проєктів із мікросервісною архітектурою, система не потребує контейнеризації, складного процесу збірки чи налаштування кількох паралельних процесів [10]. Процес встановлення займає мінімальний час за наявності готових файлів проєкту та SQL-скрипту бази даних.

Діаграму розгортання, яка відображає взаємодію між компонентами локального середовища, наведено на рисунку (рис. 3.1).

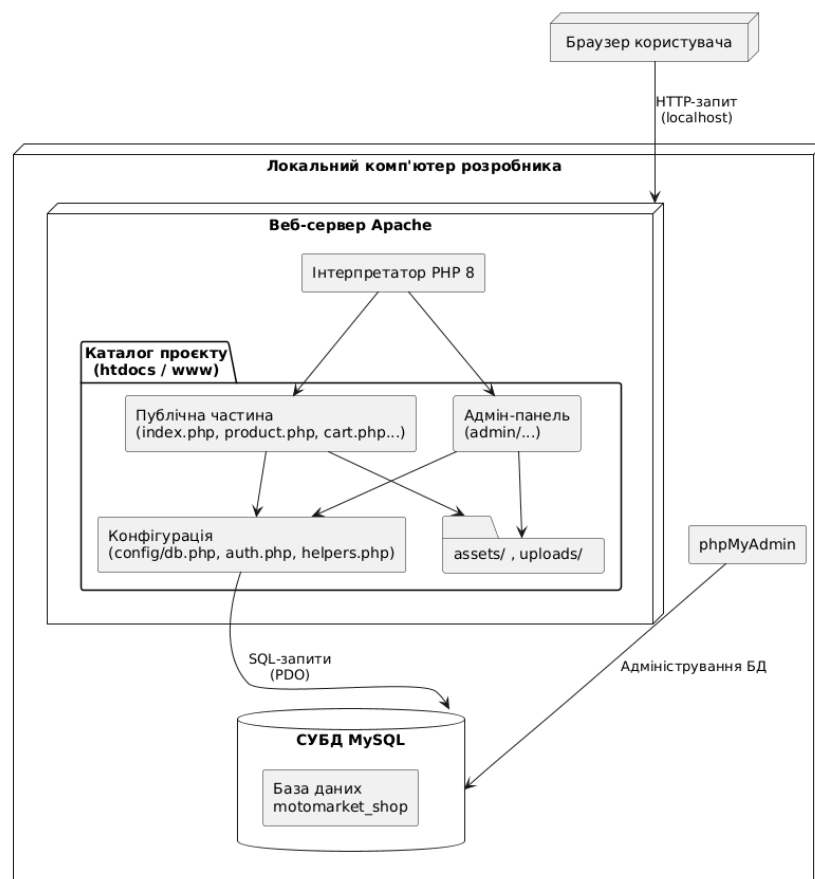


Рис. 3.1 – Схема локального розгортання платформи ATLANT

Для забезпечення стабільної роботи вебзастосунку використовується класичний стек вебтехнологій. Основні компоненти програмного середовища, необхідні для розгортання та роботи платформи, наведено у (табл. 3.1).

Таблиця 3.1

Компоненти середовища платформи ATLANT

Компонент	Технологія	Призначення
Вебсервер	Apache HTTP Server	Обробка HTTP-запитів, передача керування PHP-файлам
Серверна мова	PHP 8.0+	Бізнес-логіка, сесії, формування HTML-відповідей
СУБД	MySQL / InnoDB	Зберігання всіх даних системи, транзакції, зовнішні ключі
Локальний пакет	OpenServer / XAMPP	Об'єднує Apache, PHP і MySQL в єдиному середовищі
Управління БД	phpMyAdmin	Візуальний інтерфейс для імпорту SQL-схеми та перегляду даних

Перенесення вебзастосунку на нове робоче місце або сервер вимагає точного відтворення його програмної інфраструктури. Для коректної роботи системи недостатньо простого копіювання вихідного коду — необхідно виконати комплексну базову ініціалізацію, яка включає налаштування віртуального домену вебсервера, створення відповідної бази даних та імпорт початкового набору даних.

Детальна послідовність кроків для розгортання у середовищі OpenServer виглядає наступним чином: Завантаження та запуск середовища. Необхідно завантажити OpenServer Panel і запустити його. Слід переконатися, що у системному треї активні іконки Apache і MySQL (зелений прапорець). Якщо сервери не запускаються, слід перевірити зайнятість портів 80 і 3306 іншими програмами.

Розміщення файлів проєкту. Необхідно зупинити сервер і скопіювати папку проєкту (наприклад, atlant/) до каталогу доменів OpenServer\domains\. Після копіювання папка має містити головні PHP-файли та каталоги admin/, assets/, includes/, uploads/. Далі сервер потрібно перезапустити, щоб Apache ідентифікував новий локальний домен.

Налаштування бази даних. Через браузер необхідно відкрити панель phpMyAdmin і створити нову базу даних із назвою atlant та кодуванням utf8mb4_unicode_ci [2]. На вкладці «Імпорт» необхідно завантажити файл database.sql із кореня проєкту. Після успішного імпорту в базі буде створено шість таблиць із демонстраційними даними.

Налаштування конфігурації. У текстовому редакторі слід відкрити файл config/db.php і переконатися, що константа назви бази відповідає створеній (atlant), користувач встановлений як root, а пароль є порожнім рядком або дорівнює root (залежно від версії локального сервера). Перезавантаження вебсервера після зміни цього файлу не вимагається [1].

Для розгортання на платформі XAMPP процедура є абсолютно аналогічною: файли розміщуються у каталозі htdocs/atlant/, а сайт відкривається за адресою http://localhost/atlant/.

Після успішного виконання цих кроків система повністю готова до тестування. У браузері необхідно перейти за адресою локального домену (наприклад, http://atlant/). Відображення головної сторінки з каталогом добавок свідчить про успішне розгортання.

Для доступу до адміністративної частини необхідно перейти за адресою /admin/ і використати демонстраційні дані з файлу імпорту: логін admin та пароль 2311. При спробі авторизації система хешує введений пароль алгоритмом SHA-256 і порівнює його з наявним записом у базі даних [1]. Після успішного входу відкривається панель керування платформою. Типові помилки, що можуть виникнути під час встановлення, та способи їх вирішення наведено у (табл. 3.2).

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		48

Таблиця 3.2

Типові помилки встановлення платформи ATLANT

Помилка	Причина	Спосіб усунення
Access denied for user root	Неправильний пароль або користувач MySQL	Перевірити та змінити константи DB_USER / DB_PASS у config/db.php
Unknown database atlant	Базу не створено або вказано іншу назву	Створити базу даних atlant у панелі phpMyAdmin
Сторінка не відкривається (404)	Сервер не запущено або відсутній домен	Перезапустити OpenServer, перевірити правильність розміщення файлів
Не відображаються фотографії	Неправильний шлях або відсутня папка	Перевірити наявність та права доступу до каталогу uploads/products/
Помилки синтаксису PHP	Версія інтерпретатора PHP нижче 8.0	Змінити налаштування сервера на використання PHP 8.0+
Знак питання замість тексту	Некоректне кодування таблиць	Створювати базу з кодуванням utf8mb4_unicode_ci

Додатково після розгортання рекомендується перевірити працездатність критичних бізнес-процесів: додати товар до кошика та оформити тестове замовлення. У разі успішної роботи транзакційного механізму кошик буде очищено, а замовлення з'явиться в адміністративній панелі, що підтверджує 100% готовність застосунку.

3.2. Структура інтерфейсу та порядок взаємодії з платформою

Головна сторінка виконує презентаційну й навігаційну функції. У верхній частині розміщено контакти, основне меню, пошук, лічильник кошика та кнопку

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

кабінету. Головну сторінку вебплатформи ATLANT зображено на рисунку (рис. 3.2).

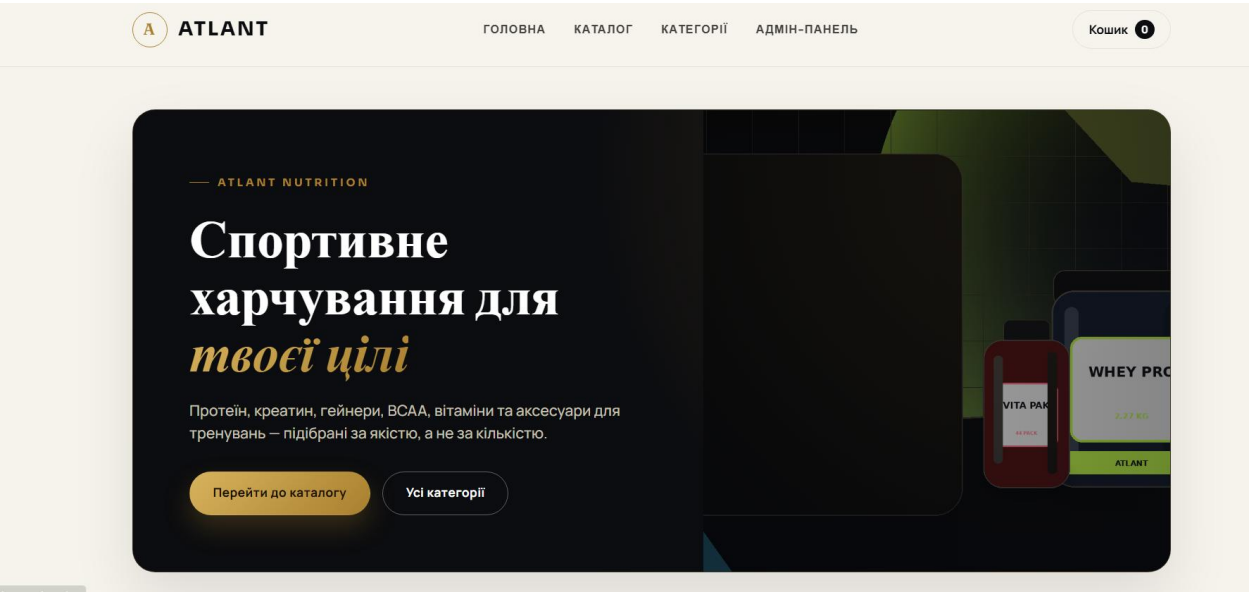


Рис. 3.2 – Головна сторінка ATLANT

Структура головної сторінки спроектована з урахуванням сучасних вимог до зручності користувацького інтерфейсу. Для швидкого залучення уваги клієнта одразу під навігаційним блоком розміщено секції, що демонструють ключові категорії спортивного харчування (протеїн, гейнери, креатин, ВСАА, вітаміни, передтренувальні комплекси, жироспалювачі, аксесуари), а також спеціальні добірки акційних і популярних товарів. Базовим елементом представлення асортименту є стандартизована картка товару, яка містить усю необхідну для прийняття рішення інформацію: якісне зображення, назву, категорію, бренд, а також стару та актуальну ціну. Важливим архітектурним рішенням є використання єдиного стилю (компонентного підходу) для відображення карток товарів у загальному каталозі та блоках рекомендацій, що забезпечує візуальну цілісність усього застосунку.

Розділ каталогу починається з інформаційного банера, під яким інтегровано панель пошуку та фільтрації за категорією. Зовнішній вигляд сторінки каталогу товарів із реалізованими інструментами пошуку та фільтрації наведено на рисунку (рис. 3.3).

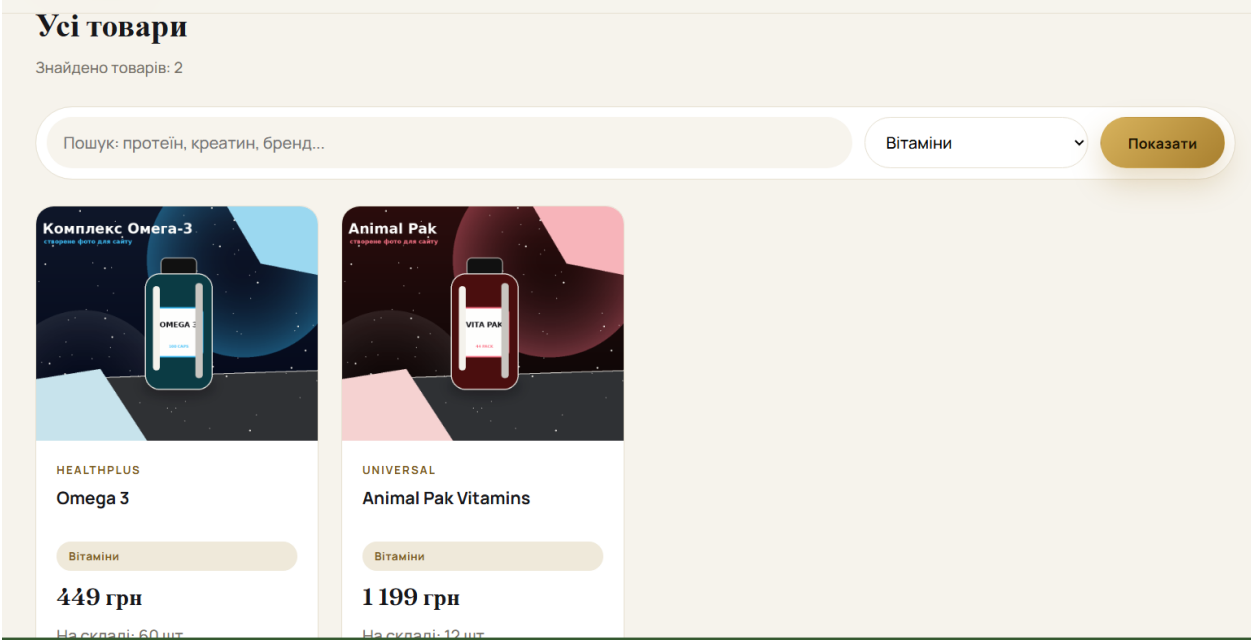


Рис. 3.3 – Каталог, пошук і фільтрація товарів

Сторінка детального перегляду товару побудована таким чином, щоб максимально зосередити увагу користувача на інформації про конкретну позицію. Візуальна частина із фотографією товару відокремлена від інформаційного блоку, що покращує сприйняття контенту та підвищує зручність користування сторінкою. Покупець має можливість переглянути категорію товару, його назву, бренд, актуальну ціну, детальний опис та інформацію про наявний залишок на складі. Для товарів із кількома варіантами фасування реалізовано вибір конкретного смаку та ваги добавки, ціна та залишок при цьому динамічно оновлюються відповідно до обраного варіанта. Сторінку детального перегляду товару та основні елементи її інтерфейсу показано на рисунку (рис. 3.4).

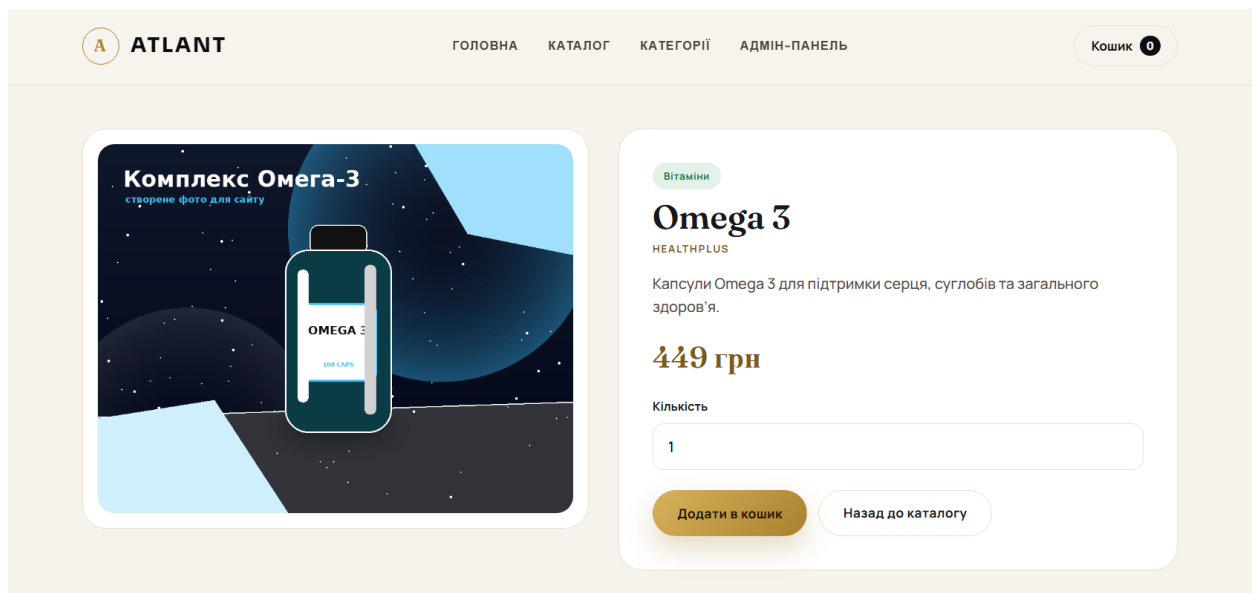


Рис. 3.4 – Сторінка детальної інформації про товар

Кошик реалізовано у вигляді таблиці, де для кожного товару показано назву, обраний варіант (смак/вага), ціну, кількість, суму та дію «видалити». Загальна вартість автоматично обчислюється сервером на основі актуальної ціни та кількості, тому користувач завжди бачить точну суму перед оформленням замовлення. Кошик покупця представлено на рисунку (рис. 3.5).

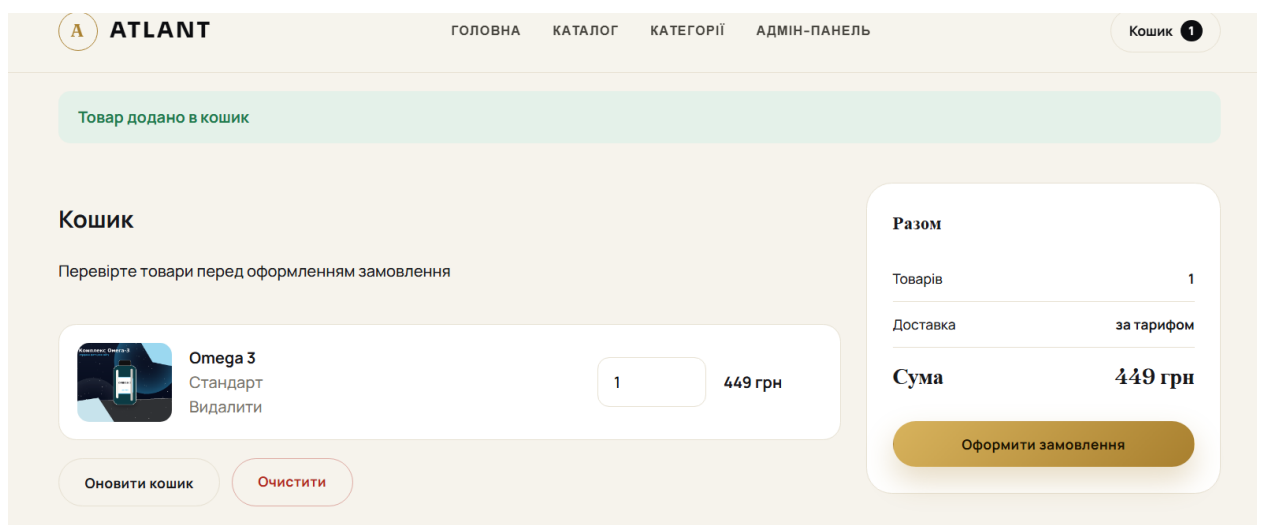


Рис. 3.5 – Кошик покупця

Під час оформлення замовлення покупець вказує контактні дані (ім'я, телефон, email), місто та адресу доставки, а також за потреби залишає коментар до

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

замовлення. У процесі він контролює склад замовлення: бачить перелік товарів, обрані варіанти, їх кількість та ціну.

Підсумкова сума автоматично обчислюється системою ще до підтвердження операції, що дозволяє покупцеві оцінити витрати та прийняти остаточне рішення. Збереження замовлення відбувається в межах однієї транзакції бази даних, що гарантує цілісність даних навіть у разі переривання процесу. Сторінку оформлення замовлення представлено на рисунку (рис. 3.6).

Рис. 3.6 – Сторінка оформлення замовлення

Після підтвердження покупки система автоматично створює новий запис у таблиці замовлень і відображає покупцеві сторінку підтвердження з номером замовлення. Сторінку підтвердження оформленого замовлення зображено на рисунку (рис. 3.7).

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		53

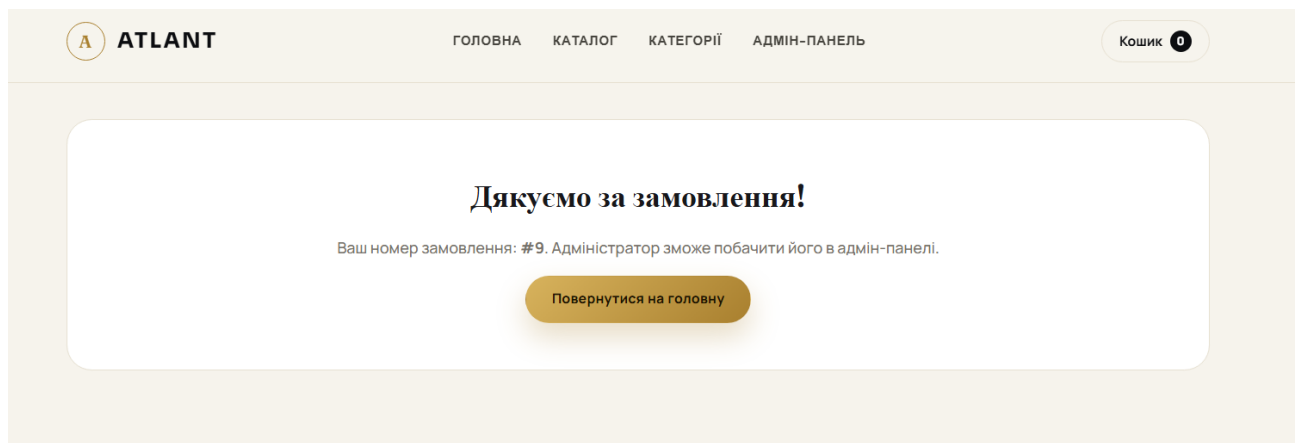


Рис. 3.7 – Сторінка підтвердження замовлення

Адміністративна панель зберігає той самий візуальний стиль, що й користувацька частина платформи, але має окреме меню з інструментами для керування. На першому екрані адміністратор бачить ключові показники: кількість товарів у системі та загальну кількість оформлених замовлень. Кнопки керування відкривають відповідні розділи: перелік товарів із можливістю редагування та завантаження зображень, керування категоріями, список замовлень із деталями та статусами. Це забезпечує адміністратору централізований доступ до всіх важливих даних і дозволяє оперативно реагувати на зміни. Головну сторінку адміністративної панелі ATLANT наведено на рисунку (рис. 3.8).

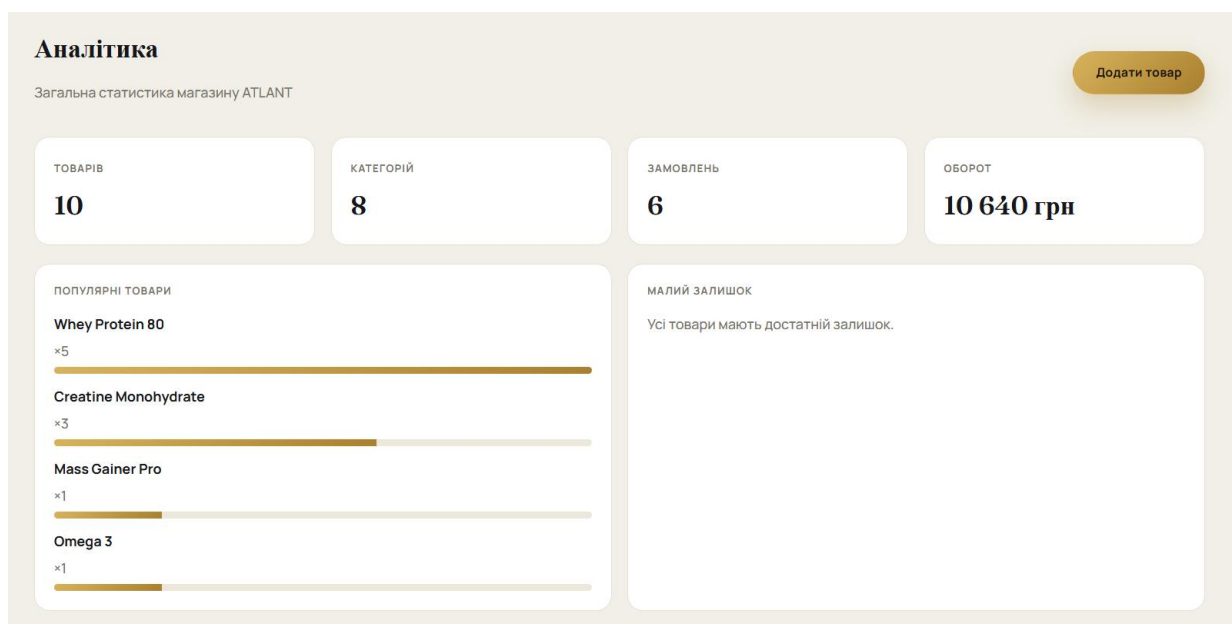


Рис. 3.8 – Головна сторінка адміністративної панелі

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

Сторінка товарів дозволяє додавати позицію або перейти до редагування. Форма містить назву, опис, ціну, стару ціну, категорію, бренд, залишок на складі, зображення (вибір із готових або завантаження власного файлу), позначку «рекомендований товар» та статус активності, а також перелік варіантів товару. Видалення товару супроводжується клієнтським підтвердженням і повторною серверною перевіркою прав доступу адміністратора. Сторінку керування товарами зображено на рисунку (рис. 3.9).

ФОТО	НАЗВА	КАТЕГОРІЯ	ЦІНА	ЗАЛИШОК	СТАТУС	ДІЇ
	Whey Protein 80 ATLANT Nutrition	Протеїн	1299 грн	34 шт.	active	Редагувати Видалити
	Mass Gainer Pro ATLANT Nutrition	Гейнери	1599 грн	22 шт.	active	Редагувати Видалити
	Creatine Monohydrate PowerLab	Креатин	699 грн	48 шт.	active	Редагувати Видалити
	BCAA 2:1:1 SportLife	BCAA / Амінокислоти	799 грн	28 шт.	active	Редагувати Видалити
	Omega 3 HealthPlus	Вітаміни	449 грн	59 шт.	active	Редагувати Видалити

Рис. 3.9 – Керування товарами в адміністративній панелі

Сторінка категорій в адміністративній панелі дозволяє здійснювати повний цикл керування структурою каталогу інтернет-магазину, зокрема додавати нові, редагувати наявні або безповоротно видаляти застарілі категорії товарів. Під час створення або модифікації категорії адміністратор має можливість детально вказати її назву, текстовий опис для покращення SEO-показників сайту, завантажити тематичне графічне зображення, а також налаштувати числовий індекс порядку відображення, що дозволяє гнучко сортувати категорії в головному меню платформи ATLANT. Крім того, на серверному рівні впроваджено захисні перевірки, які попереджають користувача про наявність прив'язаних товарів перед видаленням категорії, щоб уникнути порушення цілісності даних. Сторінку керування категоріями в реальних умовах функціонування системи зображено на рисунку (рис. 3.10).

Категорії

Керування розділами каталогу

Нова категорія

Назва

Slug

protein

ФОТО

ДАНІ

РЕДАГУВАННЯ

Протеїн

protein

category-protein-photo

Зберегти

Видалити

Гейнери

gainer

category-gainer-photo

Зберегти

Видалити

Креатин

creatine

category-creatine-photo

Зберегти

Видалити

Рис. 3.10 – Керування категоріями товарів

Для кожного замовлення адміністратор переглядає контактні дані покупця, адресу доставки, коментар та повний перелік позицій замовлення з кількістю та цінами. Статус замовлення («нове», «в обробці», «виконано», «скасовано») оновлюється окремою формою безпосередньо зі сторінки перегляду замовлення. Сторінку керування замовленнями зображено на рисунку (рис. 3.11).

Аналітика

Товари

Категорії

Замовлення

Перегляд сайту

Вийти

Замовлення

Перегляд клієнтів, складу замовлення та зміна статусу

ID	КЛІЄНТ	ДОСТАВКА	ТОВАРИ	СУМА	СТАТУС
#9 2026-06-22 22:59:10	ого 0682513633 j4294@gmail.com	Житомир Центральна	Omega 3 Стандарт *1	449 грн	Скасовано
#8 2026-06-22 22:30:44	ого 0682513633 j4294@gmail.com	Житомир Центральна	Whey Protein 80 Стандарт *1	1 299 грн	Виконано
#4 2026-06-22 22:15:56	уклукл уклукл ouliwd@gmail.com	Житомир Центральна	Whey Protein 80 Стандарт *1	1 299 грн	Нове
#3 2026-06-22 22:14:57	уклукл уклукл ouliwd@gmail.com	Житомир Центральна	Creatine Monohydrate Стандарт *2 Whey Protein 80 Стандарт *1 Mass Gainer Pro Стандарт *1	4 296 грн	Нове
#2 2026-06-21 11:04:22	Марія +380671234567 maria@example.com	Київ вул. Спортивна, 12	Whey Protein 80 Ваніль 1 кг *1	1 299 грн	Виконано

Рис. 3.11 – Керування замовленнями в адміністративній панелі

Таким чином, розроблений інтерфейс вебзастосунку ATLANT поєднує в собі інтуїтивну зрозумілість для цільового покупця та високу функціональність

для адміністратора магазину. Усі представлені сторінки спроектовано з дотриманням принципів адаптивного вебдизайну (Responsive Web Design). Це гарантує коректне та зручне відображення елементів навігації, карток товарів, форми кошика та адміністративних таблиць як на широкоформатних моніторах настільних комп'ютерів, так і на екранах мобільних пристроїв. Використання єдиної системи CSS-стилів та продуманої навігаційної архітектури дозволяє користувачам швидко знаходити та замовляти необхідні харчові добавки, а персоналу — оперативно обробляти замовлення та актуалізувати асортимент без залучення сторонніх фахівців.

3.3. Тестування роботи програмного продукту

Перевірку сформовано за сценарним підходом і охоплює п'ять функціональних областей системи: каталог і пошук товарів, сесійний кошик, оформлення замовлення, автентифікацію адміністратора та адміністративну панель керування (товари, категорії, замовлення). Для кожної функції визначено передумову, послідовність дій та результат, який безпосередньо впливає з реалізованої серверної логіки. Окремо перевіряються негативні випадки: доступ до адмінки без сесії, перевищення складського залишку, порожні обов'язкові поля форми замовлення. Такий підхід дозволяє перевірити не лише позитивні сценарії, а й коректність захисних механізмів системи [8, 9].

На відміну від проєктів із повноцінним кабінетом покупця, ATLANT реалізує модель гостьового оформлення замовлення: реєстрація та автентифікація передбачені лише для адміністратора, а покупець вказує контактні дані безпосередньо у формі checkout. Тому перевірка автентифікації стосується виключно адміністративного контуру і складається з двох сценаріїв. Успішний вхід із коректним логіном і паролем встановлює сесійну змінну `$_SESSION['admin_id']` і виконує перенаправлення на дашборд адмін-панелі. Введення неправильного пароля відхиляє вхід із єдиним узагальненим повідомленням «Невірний логін або пароль», без уточнення причини — це не дає зловмиснику зрозуміти, чи існує введений логін у системі [8].

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						57
Змн.	Арк.	№ докум.	Підпис	Дата		

Перевірка каталогу та пошуку підтверджує коректну роботу фільтрів. При виборі категорії система повертає виключно товари цієї категорії (умова `c.slug = ?`). При одночасному застосуванні пошуку й категорії результат відповідає обом умовам одночасно, оскільки обидві перевірки об'єднуються в одному запиті через AND. Текстовий пошук знаходить товари за збігом одразу у трьох полях — назві, бренді та описі (LIKE за кожним із них). Порожній результат при некоректному запиті відображає повідомлення «Товарів за таким запитом не знайдено» без помилки сервера.

Перевірка кошика охоплює додавання, оновлення та граничні випадки. При першому додаванні товару створюється новий запис у сесійному масиві `$_SESSION['cart']` з ключем `productId:variantId`. При повторному додаванні того самого товару (і того самого варіанту) кількість акумулюється, а не дублюється новим рядком. Окремо перевірено сценарій, де користувач змінює кількість прямо в полі кошика і одразу переходить до оформлення замовлення, не натискаючи окрему кнопку «Оновити кошик», — кнопка «Оформити замовлення» прив'язана до тієї самої форми (`form="cart-form"`), тому актуальна кількість зберігається коректно в обох випадках.

Перевірка оформлення замовлення є найбільш розгорнутою, оскільки цей сценарій зачіпає одразу три таблиці. Після заповнення форми та підтвердження система створює запис в `orders`, додає позиції до `order_items` із зафіксованою на момент покупки ціною, зменшує `stock` у `products` і, за наявності обраного варіанту, додатково в `product_variants` — усе в межах однієї транзакції PDO. Якщо на етапі списання залишку виявляється, що товару недостатньо, виконується `rollback()`: замовлення не створюється, складські залишки залишаються незмінними, а покупець бачить повідомлення «Недостатньо товару на складі» [1, 2].

Окремий блок перевірки стосується адміністративної панелі, оскільки саме через неї відбувається керування всім асортиментом і замовленнями. Додавання та редагування товару перевіряються разом із завантаженням зображення: сервер приймає файл лише після перевірки MIME-типу за білим списком (`jpeg, png,`

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

webp, gif), а варіанти товару (смак, вага, ціна, залишок) повністю перезаписуються при кожному збереженні форми. Видалення товару супроводжується клієнтським підтвердженням (confirm()) і призводить до каскадного видалення пов'язаних записів у product_variants. Зміна статусу замовлення перевіряється через випадючий список зі значеннями new/processing/completed/cancelled — вибір іншого статусу одразу надсилає форму та оновлює orders.status без додаткових дій адміністратора. Перелік усіх перевірених сценаріїв — за каталогом, кошиком, замовленнями та адмін-панеллю — наведено у таблиці (табл. 3.3).

Таблиця 3.3

Функціональні сценарії перевірки системи

Сценарій	Дії	Очікуваний результат	Стан
Перегляд каталогу	Відкрити головну сторінку	Показано лише товари зі status='active'	Реалізовано
Текстовий пошук	Ввести назву, бренд або фрагмент опису	Знайдено товари за збігом у трьох полях	Реалізовано
Фільтр за категорією	Обрати категорію зі списку	Показано лише товари обраної категорії	Реалізовано
Комбінований фільтр	Застосувати пошук і категорію одночасно	Результат відповідає обом умовам (AND)	Реалізовано
Порожній результат	Ввести неіснуючий запит	Повідомлення «Товарів не знайдено»	Реалізовано
Додавання в кошик	Натиснути «Купити» на товарі	Створено запис у сесійному кошику (qty=1)	Реалізовано
Повторне додавання	Додати той самий товар ще раз	Кількість у кошику акумулюється (1+1=2)	Реалізовано
Вибір варіанту	Додати товар з конкретним смаком/вагою	Окремий рядок із ціною варіанту	Реалізовано

продовж. табл. 3.3.

Зміна кількості	Змінити qty та одразу оформити замовлення	Нова кількість враховується без проміжного оновлення	Реалізовано
Очищення кошика	Натиснути «Очистити»	Сесійний кошик порожній	Реалізовано
Checkout (порожній)	Відкрити checkout.php без товарів	Redirect на cart.php з повідомленням	Реалізовано
Порожні поля checkout	Залишити обов'язкове поле пустим	Список помилок, замовлення не створюється	Реалізовано
Успішний checkout	Підтвердити замовлення	Створено orders, order_items, зменшено stock	Реалізовано
Перевищення залишку	Замовити кількість > stock	Транзакція відкочена (rollback)	Реалізовано
Вхід адміністратора	Ввести логін і пароль	Сесію встановлено, redirect на адмінку	Реалізовано
Неправильний пароль	Ввести невірний пароль	Повідомлення про помилку входу	Реалізовано
Admin-доступ без сесії	Відкрити admin/... на пряму	Redirect на admin/login.php	Реалізовано
CRUD товару	Додати/редагувати товар з фото	Запис у products, product_variants, перевірка MIME	Реалізовано
Видалення товару	Підтвердити видалення	Каскадне видалення записів та варіантів	Реалізовано
Зміна статусу	Обрати новий статус у списку	orders.status оновлено	Реалізовано

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

Проведений аналіз вихідного коду розробленої системи підтверджує надійність її архітектури та захищеність від поширених уразливостей. Статичний аналіз коду показує послідовне використання підготовлених запитів PDO (методи `prepare()` та `execute()`) для всіх значень і параметрів, що надходять від користувача [1]. Це повністю нівелює ризик успішного проведення SQL-ін'єкцій, оскільки дані клієнта відокремлюються від тіла самого запиту на етапі компіляції. Сортвання товарів у каталозі є чітко фіксованим на рівні самого SQL-запиту (`ORDER BY is_featured DESC, created_at DESC`) і не приймає жодних довільних параметрів від клієнта, що виключає можливість підстановки сторонніх команд у цю частину коду [8]. Весь динамічний HTML-контент перед виведенням на екран обов'язково проходить через допоміжну функцію `e()`, яка застосовує стандартну функцію `htmlspecialchars()` із прапором `ENT_QUOTES` [1, 8]. Такий підхід гарантує безпеку від XSS-атак, блокуючи виконання шкідливих скриптів у браузері користувача [24]. Крім того, надійна серверна перевірка прав доступу за допомогою функції `require_admin()` викликається на початку кожного файлу адміністративного каталогу, що унеможливорює несанкціонований доступ в обхід інтерфейсу [8].

Окремо в ході роботи було проведено детальний аналіз можливих безпечових ризиків для веборієнтованої системи електронної комерції. Для кожної потенційної загрози (такої як перехоплення сесій, підбір паролів чи збої при передачі даних) було чітко визначено діючий захисний механізм, реалізований на платформі ATLANT. Комплексний підхід до моделювання загроз дозволив мінімізувати критичні вразливості та забезпечити надійну роботу інтернет-магазину в агресивному мережевому середовищі. Також сформовано практичні рекомендації щодо подальшого посилення захисту системи перед її майбутнім промисловим розгортанням на реальному сервері хостинг-провайдера. Отримані результати порівняння та аналізу безпекових факторів наведено у таблиці (табл. 3.4).

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

Аналіз безпекових ризиків системи

Ризик	Наявний захист	Рекомендоване посилення
SQL-ін'єкція	PDO prepared statements для всіх user-input значень	Зберегти підхід для будь-яких нових динамічних запитів
XSS	Функція <code>е()</code> / <code>htmlspecialchars()</code> на всьому динамічному виводі	Впровадити Content Security Policy через HTTP-заголовки
Злом пароля адміністратора	Порівняння <code>hash('sha256')</code> під час входу	Перейти на <code>password_hash()/password_verify()</code> (bcrypt) [25] і додати rate limiting на вхід
Несанкціонований доступ до адмінки	<code>require_admin()</code> на початку кожної admin-сторінки	Додати журнал адміністративних операцій із timestamp
CSRF	Форми лише POST, без токенів	Додати CSRF-токени до всіх форм зміни стану [8]
Витік конфігурації	Параметри підключення у <code>config/db.php</code> у репозиторії	Винести секрети до .env-файлу поза репозиторієм
Негативний залишок	Атомарний UPDATE ... WHERE stock >= ? у межах транзакції з <code>rollBack()</code> при недостатці	Додатково розглянути блокування рядка при високому навантаженні
Перебір облікових записів	Єдине узагальнене повідомлення про помилку входу	Додати тимчасове блокування після N невдалих спроб

Основним напрямом подальшого посилення захисту є впровадження CSRF-токенів для операцій зміни даних кошика та оформлення замовлення, а також перехід автентифікації адміністратора з sha256 на bcrypt.

Перевірка адаптивності підтвердила коректну роботу інтерфейсу на різних розмірах екранів, зокрема мобільне меню коректно відкривається й закривається на вузьких екранах. У подальшому доцільно виконати аудит інтерфейсу на відповідність стандарту WCAG 2.2 [5, 11].

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

Висновки до третього розділу

Було розглянуто процес встановлення, налаштування та запуску вебзастосунку ATLANT у локальному середовищі. Описано архітектуру клієнт-серверної взаємодії, основні програмні компоненти системи (Apache, PHP, MySQL) та особливості їх роботи в рамках WAMP. Також визначено параметри конфігурації підключення до бази даних і наведено послідовність кроків розгортання системи, що забезпечує її коректний запуск і готовність до тестування.

Проаналізовано структуру користувацького та адміністративного інтерфейсів вебзастосунку ATLANT. Розглянуто основні функціональні екрани системи: головну сторінку, каталог товарів із функціями пошуку та фільтрації, сторінку товару з вибором варіантів (смак, вага), кошик та процес оформлення замовлення. Окрему увагу приділено адміністративній панелі, яка дозволяє зручно керувати товарами, категоріями та замовленнями, а також надає доступ до базової аналітики магазину, формуючи повноцінний інструмент управління системою.

Виконано тестування функціональних можливостей вебзастосунку за сценарним підходом, що охоплює як позитивні, так і негативні сценарії використання. Перевірено роботу основних модулів системи: каталогу, сесійного кошика, транзакційного оформлення замовлень зі списанням залишків та системи адміністративного доступу. У ході перевірки особливу увагу було приділено обробці помилкових дій користувача та валідації вхідних даних на стороні сервера. Додатково проведено аналіз безпекових ризиків і перевірено ефективність реалізованих механізмів захисту, що підтвердило коректність роботи системи, її стійкість до зовнішніх загроз та повну готовність до використання.

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

ВИСНОВКИ

У кваліфікаційній роботі розроблено веборієнтовану платформу ATLANT для спеціалізованого продажу харчових добавок та спортивного харчування. На основі аналізу предметної області обґрунтовано доцільність створення незалежного інтернет-магазину з фокусом на швидкі продажі без обов'язкової реєстрації покупців. Сформовано вимоги до структурованого каталогу, динамічної фільтрації, сесійного кошика, обліку замовлень та окремого адміністративного контуру.

Спроектовано клієнт-серверну архітектуру з серверним рендерингом сторінок. Мова програмування PHP 8 реалізує бізнес-логіку системи, механізм PDO забезпечує безпечний доступ до бази даних MySQL, а технології HTML5, CSS3 та JavaScript формують адаптивний користувацький інтерфейс. Структура проєкту оптимізована для швидкого розгортання в локальних середовищах XAMPP або OpenServer без використання складних систем збірки коду.

Реляційна база даних складається з шести взаємопов'язаних таблиць: `admins`, `categories`, `products`, `product_variants`, `orders` та `order_items`. Зовнішні ключі відображають логіку предметної області: каскадне видалення застосовано для залежних сутностей (варіантів товарів), а встановлення SET NULL для зв'язків із замовленнями, що дозволяє видаляти товари чи категорії без втрати фінансової історії продажів. Оформлення замовлення відбувається у межах атомарної транзакції PDO, яка узгоджено фіксує покупку, списує складські залишки та очищує кошик.

Програмно реалізовано інтерактивний каталог із комбінованими фільтрами, сторінки деталей товару з вибором смаку чи ваги, транзакційний checkout та захищену адмін-панель для керування номенклатурою і статусами замовлень. Комплексне тестування системи за 20 різноплановими сценаріями повністю підтвердило коректність роботи всіх заявлених модулів.

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		64

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. PHP Group. PHP Manual [Електронний ресурс]. — 2026. — Режим доступу: <https://www.php.net/manual/en/> (дата звернення: 18.06.2026).
2. Oracle. MySQL 8.4 Reference Manual [Електронний ресурс]. — 2026. — Режим доступу: <https://dev.mysql.com/doc/refman/8.4/en/> (дата звернення: 18.06.2026).
3. Mozilla Developer Network. HTML: HyperText Markup Language [Електронний ресурс]. — 2026. — Режим доступу: <https://developer.mozilla.org/en-US/docs/Web/HTML> (дата звернення: 23.05.2026).
4. Mozilla Developer Network. CSS: Cascading Style Sheets [Електронний ресурс]. — 2026. — Режим доступу: <https://developer.mozilla.org/en-US/docs/Web/CSS> (дата звернення: 23.05.2026).
5. Mozilla Developer Network. Responsive web design [Електронний ресурс]. — 2026. — Режим доступу: https://developer.mozilla.org/en-US/docs/Learn_web_development (дата звернення: 01.06.2026).
6. Mozilla Developer Network. CSS grid layout [Електронний ресурс]. — 2026. — Режим доступу: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_grid_layout (дата звернення: 01.06.2026).
7. Mozilla Developer Network. Using Fetch [Електронний ресурс]. — 2026. — Режим доступу: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch (дата звернення: 18.06.2026).
8. OWASP Foundation. Security Cheat Sheet Series [Електронний ресурс]. — 2026. — Режим доступу: <https://cheatsheetseries.owasp.org/> (дата звернення: 07.06.2026).
9. phpMyAdmin Development Team. phpMyAdmin documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://docs.phpmyadmin.net/en/latest/> (дата звернення: 13.06.2026).
10. Apache Friends. XAMPP [Електронний ресурс]. — 2026. — Режим доступу: <https://www.apachefriends.org/index.html> (дата звернення: 18.06.2026).

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

- 11.ВСАА.ua. Інтернет-магазин харчових добавок [Електронний ресурс]. — 2026. — Режим доступу: <https://bcaa.ua/> (дата звернення: 18.06.2026).
- 12.Belok.ua. Маркетплейс спортивного харчування [Електронний ресурс]. — 2026. — Режим доступу: <https://belok.ua/ua/> (дата звернення: 18.06.2026).
- 13.Rozetka. Розділ спортивного харчування [Електронний ресурс]. — 2026. — Режим доступу: <https://rozetka.com.ua/ua/sport-nutrition/c4628591/> (дата звернення: 18.06.2026).
- 14.Nixon R. Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML5. 7th ed. Sebastopol: O'Reilly Media, 2024. 832 p.
- 15.Lockhart J. Modern PHP: The Complete Guide. New York: Apress, 2023. 420 p.
- 16.Duckett J. HTML and CSS: Design and Build Websites. Updated Edition. Indianapolis: Wiley, 2021. 490 p.
- 17.Duckett J. JavaScript and jQuery: Interactive Front-End Web Development. Updated Edition. Indianapolis: Wiley, 2022. 640 p.
- 18.W3C Web Accessibility Initiative. WCAG 2.2 [Електронний ресурс]. — 2023. — Режим доступу: <https://www.w3.org/TR/WCAG22/> (дата звернення: 18.06.2026).
- 19.OpenServer Team. Open Server Panel Documentation [Електронний ресурс]. — 2026. — Режим доступу: <https://ospanel.io/docs/> (дата звернення: 10.06.2026).
- 20.IETF. HTTP State Management Mechanism: RFC 6265 [Електронний ресурс]. — 2021. — Режим доступу: <https://datatracker.ietf.org/doc/html/rfc6265> (дата звернення: 19.06.2026).
- 21.OWASP Foundation. File Upload Cheat Sheet [Електронний ресурс]. — 2026. — Режим доступу: https://cheatsheetseries.owasp.org/cheatsheets/File_Upload_Cheat_Sheet.html (дата звернення: 18.06.2026).
- 22.Laudon K. C., Traver C. G. E-Commerce 2023: Business, Technology, Society. 18th ed. Boston: Pearson, 2023. 912 p.

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

23. Turban E., Outland J., King D., Lee J. K., Liang T.-P. Electronic Commerce 2018: A Managerial and Social Networks Perspective. 9th ed. New York: Springer, 2018. 690 p.
24. Kolawole A. O., Olugbara O. O. Web Application Security Vulnerabilities: A Systematic Literature Review. Journal of Information Security. 2022. Vol. 13, No. 2. P. 88–105. DOI: 10.4236/jis.2022.132006.
25. Gajić Z., Đorđević B. Design Patterns for Secure PHP Web Applications. International Journal of Computer Science and Information Technology. 2021. Vol. 13, No. 4. P. 45–58. DOI: 10.5121/ijcsit.2021.13404.
26. W3Schools. PHP Tutorial [Электронный ресурс]. — 2026. — Режим доступа: <https://www.w3schools.com/php/> (дата звернення: 20.05.2026).
27. W3Schools. SQL Tutorial [Электронный ресурс]. — 2026. — Режим доступа: <https://www.w3schools.com/sql/> (дата звернення: 20.05.2026).
28. Google Developers. Core Web Vitals [Электронный ресурс]. — 2026. — Режим доступа: <https://web.dev/vitals/> (дата звернення: 15.05.2026).
29. Niederst Robbins J. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics. 5th ed. Sebastopol: O'Reilly Media, 2018. 800 p.
30. Welling L., Thomson L. PHP and MySQL Web Development. 5th ed. Boston: Addison-Wesley, 2016. 1080 p.

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

ДОДАТКИ

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

У додатку наведено основну частину SQL-сценарію створення таблиць ATLANT. Повний файл database.sql також містить демонстраційні облікові записи, товари, відгуки та контактне повідомлення.

Обробка А.1 – Файл database.sql

```
CREATE DATABASE IF NOT EXISTS atlant_shop CHARACTER SET utf8mb4
COLLATE utf8mb4_unicode_ci;
```

```
USE atlant_shop;
```

```
SET FOREIGN_KEY_CHECKS=0;
```

```
DROP TABLE IF EXISTS order_items;
```

```
DROP TABLE IF EXISTS orders;
```

```
DROP TABLE IF EXISTS product_variants;
```

```
DROP TABLE IF EXISTS products;
```

```
DROP TABLE IF EXISTS categories;
```

```
DROP TABLE IF EXISTS admins;
```

```
SET FOREIGN_KEY_CHECKS=1;
```

```
CREATE TABLE admins (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    username VARCHAR(100) NOT NULL UNIQUE,
```

```
    password_hash VARCHAR(255) NOT NULL,
```

```
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
CREATE TABLE categories (
```

```
    id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    name VARCHAR(150) NOT NULL,
```

```
    slug VARCHAR(150) NOT NULL UNIQUE,
```

```
    description TEXT NULL,
```

```
    image VARCHAR(255) NULL,
```

					ІІЗ.КР.Б – 121 – 26 - ІЗ	Арк.
						69
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        sort_order INT DEFAULT 0,
        created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE products (
    id INT AUTO_INCREMENT PRIMARY KEY,
    category_id INT NULL,
    name VARCHAR(255) NOT NULL,
    brand VARCHAR(150) NOT NULL,
    description TEXT NULL,
    image VARCHAR(255) NULL,
    price DECIMAL(10,2) NOT NULL DEFAULT 0,
    old_price DECIMAL(10,2) NULL,
    stock INT NOT NULL DEFAULT 0,
    is_featured TINYINT(1) NOT NULL DEFAULT 0,
    status ENUM('active','inactive') NOT NULL DEFAULT 'active',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    CONSTRAINT fk_products_category FOREIGN KEY (category_id)
REFERENCES categories(id) ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE product_variants (
    id INT AUTO_INCREMENT PRIMARY KEY,
    product_id INT NOT NULL,
    flavor VARCHAR(120) NULL,
    weight VARCHAR(120) NULL,
    price DECIMAL(10,2) NOT NULL DEFAULT 0,
    stock INT NOT NULL DEFAULT 0,
    CONSTRAINT fk_variants_product FOREIGN KEY (product_id)
REFERENCES products(id) ON DELETE CASCADE

```

					ПІЗ.КР.Б – 121 – 26 - ПІЗ	Арк.
						70
Змн.	Арк.	№ докум.	Підпис	Дата		

```

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE orders (
    id INT AUTO_INCREMENT PRIMARY KEY,
    customer_name VARCHAR(180) NOT NULL,
    phone VARCHAR(50) NOT NULL,
    email VARCHAR(180) NULL,
    city VARCHAR(120) NOT NULL,
    address VARCHAR(255) NOT NULL,
    comment TEXT NULL,
    total DECIMAL(10,2) NOT NULL DEFAULT 0,
    status ENUM('new','processing','completed','cancelled') NOT
NULL DEFAULT 'new',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

CREATE TABLE order_items (
    id INT AUTO_INCREMENT PRIMARY KEY,
    order_id INT NOT NULL,
    product_id INT NULL,
    product_name VARCHAR(255) NOT NULL,
    variant_name VARCHAR(255) NULL,
    quantity INT NOT NULL DEFAULT 1,
    price DECIMAL(10,2) NOT NULL DEFAULT 0,
    CONSTRAINT fk_items_order FOREIGN KEY (order_id) REFERENCES
orders(id) ON DELETE CASCADE,
    CONSTRAINT fk_items_product FOREIGN KEY (product_id) REFERENCES
products(id) ON DELETE SET NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

INSERT INTO admins (username, password_hash) VALUES
('2311', '...');

```

					ІІЗ.КР.Б – 121 – 26 - ІІЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		71

```
INSERT INTO categories (id, name, slug, description, image,
sort_order) VALUES
```

```
(1, 'Протеїн', 'protein', 'Білкові суміші для набору та підтримки
м'язової маси.', 'category-protein-photo.png', 1),
```

```
(2, 'Гейнери', 'gainer', 'Калорійні комплекси для набору маси.',
'category-gainer-photo.png', 2),
```

```
(3, 'Креатин', 'creatine', 'Добавки для сили, вибуховості та про-
гресу у тренуваннях.', 'category-creatine-photo.png', 3),
```

```
(4, 'BCAA / Амінокислоти', 'bcaa', 'Амінокислоти для відновлення
після навантажень.', 'category-bcaa-photo.png', 4),
```

```
(5, 'Вітаміни', 'vitamins', 'Комплекси для здоров'я та підтримки
організму.', 'category-vitamins-photo.png', 5),
```

```
(6, 'Передтренувальні', 'preworkout', 'Енергія, концентрація та ви-
тривалість для тренувань.', 'category-preworkout-photo.png', 6),
```

```
(7, 'Жироспалювачі', 'fatburner', 'Підтримка контролю ваги та акти-
вного способу життя.', 'category-fatburner-photo.png', 7),
```

```
(8, 'Акcesуари', 'accessories', 'Шейкери, рукавички та інші товари
для залу.', 'category-accessories-photo.png', 8);
```

```
INSERT INTO products (id, category_id, name, brand, description,
image, price, old_price, stock, is_featured, status) VALUES
```

```
(1, 1, 'Whey Protein 80', 'ATLANT Nutrition', 'Сироватковий протеїн
з високим вмістом білка. Підходить для набору м'язової маси та віднов-
лення після тренування.', 'product-whey.png', 1299.00, 1499.00, 35, 1,
'active'),
```

```
(2, 2, 'Mass Gainer Pro', 'ATLANT Nutrition', 'Гейнер з вуглеводами
та білком для спортсменів, яким потрібно збільшити калорійність раці-
ону.', 'product-gainer.png', 1599.00, 1799.00, 22, 1, 'active'),
```

```
(3, 3, 'Creatine Monohydrate', 'PowerLab', 'Креатин моногідрат для
підвищення силових показників, витривалості та прогресу у тренуваннях.',
'product-creatine.png', 699.00, NULL, 48, 1, 'active'),
```

```
(4, 4, 'BCAA 2:1:1', 'SportLife', 'Амінокислоти з класичним спів-
відношенням 2:1:1 для підтримки відновлення.', 'product-bcaa.png',
799.00, 899.00, 28, 0, 'active'),
```

```
(5, 5, 'Omega 3', 'HealthPlus', 'Капсули Omega 3 для підтримки се-
рця, суглобів та загального здоров'я.', 'product-omega.png', 449.00,
NULL, 60, 0, 'active'),
```

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						72
Змн.	Арк.	№ докум.	Підпис	Дата		

(6, 6, 'Pre-Workout Energy', 'BlastFit', 'Передтрениувальний комплекс для енергії, концентрації та продуктивності на тренуванні.', 'product-preworkout.png', 999.00, 1199.00, 16, 1, 'active'),

(7, 5, 'Animal Pak Vitamins', 'Universal', 'Комплекс вітамінів і мінералів для активних людей та спортсменів.', 'product-animal-pak.png', 1199.00, NULL, 12, 0, 'active'),

(8, 7, 'L-Carnitine 3000', 'FitBurn', 'L-карнітин для підтримки активного способу життя та тренувального режиму.', 'product-carnitine.png', 649.00, 749.00, 25, 0, 'active'),

(9, 1, 'Casein Night Protein', 'ATLANT Nutrition', 'Казеїновий протеїн повільного засвоєння для вечірнього прийому.', 'product-casein.png', 1399.00, NULL, 18, 0, 'active'),

(10, 8, 'Shaker + Gloves Set', 'ATLANT Gear', 'Набір аксесуарів для тренування: шейкер і рукавички для залу.', 'product-accessories.png', 499.00, 599.00, 40, 0, 'active');

INSERT INTO product_variants (product_id, flavor, weight, price, stock) VALUES

(1, 'Шоколад', '1 кг', 1299.00, 16),
 (1, 'Ваніль', '1 кг', 1299.00, 19),
 (2, 'Банан', '3 кг', 1599.00, 12),
 (2, 'Шоколад', '3 кг', 1599.00, 10),
 (3, 'Без смаку', '300 г', 699.00, 48),
 (4, 'Кавун', '500 г', 799.00, 14),
 (4, 'Лимон', '500 г', 799.00, 14),
 (6, 'Ягода', '300 г', 999.00, 16),
 (9, 'Шоколад', '900 г', 1399.00, 18);

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
Змн.	Арк.	№ докум.	Підпис	Дата		73

Фрагмент checkout.php демонструє завантаження кошика, перевірку форми, створення заголовка й позицій замовлення, зміну складських залишків та очищення кошика у межах транзакції.

Обробка Б.1 – Серверний сценарій checkout.php

```
<?php
require_once __DIR__ . '/config/db.php';
require_once __DIR__ . '/config/helpers.php';

$items = get_cart_items($pdo);
if (!$items) {
    flash('Кошик порожній', 'error');
    header('Location: ' . url('cart.php'));
    exit;
}
$total = array_sum(array_column($items, 'subtotal'));
$errors = [];
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $name = trim($_POST['customer_name'] ?? '');
    $phone = trim($_POST['phone'] ?? '');
    $email = trim($_POST['email'] ?? '');
    $city = trim($_POST['city'] ?? '');
    $address = trim($_POST['address'] ?? '');
    $comment = trim($_POST['comment'] ?? '');
    if ($name === '') $errors[] = 'Вкажіть ім'я';
    if ($phone === '') $errors[] = 'Вкажіть телефон';
    if ($city === '') $errors[] = 'Вкажіть місто';
    if ($address === '') $errors[] = 'Вкажіть адресу';

    if (!$errors) {
        try {
            $pdo->beginTransaction();
            $stmt = $pdo->prepare("INSERT INTO orders (customer_name,
phone, email, city, address, comment, total, status) VALUES
(?,?,?,?,?,?,?, 'new')");
            $stmt->execute([$name, $phone, $email, $city, $address,
$comment, $total]);
            $orderId = (int)$pdo->lastInsertId();
            $itemStmt = $pdo->prepare("INSERT INTO order_items (or-
der_id, product_id, product_name, variant_name, quantity, price) VALUES
(?,?,?,?,?,?,?)");
            $productStockStmt = $pdo->prepare("UPDATE products SET
stock = stock - ? WHERE id = ? AND stock >= ?");
            $variantStockStmt = $pdo->prepare("UPDATE product_variants
SET stock = stock - ? WHERE id = ? AND stock >= ?");
            foreach ($items as $item) {
```

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

```

        $itemStmt->execute([$orderId, $item['product']['id'],
$item['product']['name'], $item['variant_name'], $item['qty'],
$item['price']]);

        $productStockStmt->execute([$item['qty'],
$item['product']['id'], $item['qty']]);
        if ($productStockStmt->rowCount() === 0) {
            throw new RuntimeException('Недостатньо товару на
складі: ' . $item['product']['name']);
        }
        if ($item['variant']) {
            $variantStockStmt->execute([$item['qty'],
$item['variant']['id'], $item['qty']]);
            if ($variantStockStmt->rowCount() === 0) {
                throw new RuntimeException('Недостатньо товару
на складі: ' . $item['product']['name']);
            }
        }
    }
    $pdo->commit();
    unset($_SESSION['cart']);
    header('Location: ' . url('order-success.php?id=' . $or-
derId));
    exit;
} catch (Exception $e) {
    $pdo->rollBack();
    $errors[] = $e instanceof RuntimeException ? $e->getMes-
sage() : 'Не вдалося оформити замовлення. Спробуйте ще раз.';
}
}
}
$pageTitle = 'Оформлення замовлення – ATLANT';
include __DIR__ . '/includes/header.php';
?>
<div class="container">
    <div class="checkout-box">
        <span class="eyebrow">Оформлення</span>
        <h2>Дані для доставки</h2>
        <?php if ($errors): ?><div class="alert alert-error"><? e(im-
plode(' ', $errors)) ?></div><?php endif; ?>
        <form method="post" class="form-grid">
            <div><label>Ім'я</label><input class="input" name="cus-
tomer_name" value="<? e($_POST['customer_name'] ?? '') ?>" re-
quired></div>
            <div><label>Телефон</label><input class="input"
name="phone" value="<? e($_POST['phone'] ?? '') ?>" required></div>
            <div><label>Email</label><input class="input" type="email"
name="email" value="<? e($_POST['email'] ?? '') ?>"></div>
            <div><label>Місто</label><input class="input" name="city"
value="<? e($_POST['city'] ?? '') ?>" required></div>

```

						Арк.
					ІПЗ.КР.Б – 121 – 26 - ПЗ	75
Змн.	Арк.	№ докум.	Підпис	Дата		

Обробка Б.2 – Серверний сценарій cart.php (сесійний кошик)

```
<?php
require_once __DIR__ . '/config/db.php';
require_once __DIR__ . '/config/helpers.php';

$action = $_POST['action'] ?? $_GET['action'] ?? '';
if ($action === 'add') {
    $productId = (int)($_POST['product_id'] ?? 0);
    $variantId = (int)($_POST['variant_id'] ?? 0);
    $qty = max(1, (int)($_POST['qty'] ?? 1));
    $stmt = $pdo->prepare("SELECT id FROM products WHERE id=? AND
status='active'");
    $stmt->execute([$productId]);
    if ($stmt->fetch()) {
        $key = cart_key($productId, $variantId);
        $_SESSION['cart'][$key] = [
            'product_id' => $productId,
            'variant_id' => $variantId,
            'qty' => ($_SESSION['cart'][$key]['qty'] ?? 0) + $qty,
        ];
        flash('Товар додано в кошик');
    }
    header('Location: ' . url('cart.php'));
    exit;
}
if ($action === 'update' || $action === 'checkout') {
    foreach ($_POST['qty'] ?? [] as $key => $qty) {
        if (isset($_SESSION['cart'][$key])) {
            $qty = (int)$qty;
            if ($qty <= 0) unset($_SESSION['cart'][$key]);
            else $_SESSION['cart'][$key]['qty'] = min($qty, 99);
        }
    }
    if ($action === 'checkout') {
        header('Location: ' . url('checkout.php'));
        exit;
    }
    flash('Кошик оновлено');
    header('Location: ' . url('cart.php'));
    exit;
}
if ($action === 'remove') {
    $key = $_GET['key'] ?? '';
    unset($_SESSION['cart'][$key]);
    flash('Товар видалено з кошика');
    header('Location: ' . url('cart.php'));
    exit;
}
if ($action === 'clear') {
    unset($_SESSION['cart']);
}
```

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

```

        flash('Кошик очищено', 'info');
        header('Location: ' . url('cart.php'));
        exit;
    }

    $pageTitle = 'Кошик – ATLANT';
    $items = get_cart_items($pdo);
    $total = array_sum(array_column($items, 'subtotal'));
    include __DIR__ . '/includes/header.php';
    ?>
    <div class="container cart-layout">
        <section>
            <div class="section-head">
                <div>
                    <h2>Кошик</h2>
                    <p class="lead">Перевірте товари перед оформленням замов-
лення</p>
                </div>
            </div>
            <?php if (!$items): ?>
                <div class="empty">
                    <h3>Кошик порожній</h3>
                    <p class="muted">Додайте товари з каталогу.</p>
                    <a class="btn" href="<?= url('index.php#catalog') ?>">До
каталогу</a>
                </div>
            <?php else: ?>
                <form method="post" id="cart-form">
                    <div class="cart-list">
                        <?php foreach ($items as $item): $p =
$item['product']; ?>
                            <div class="cart-item">
                                ">
                                <div>
                                    <b><?= e($p['name']) ?></b>
                                    <div class="muted"><?=
e($item['variant_name']) ?></div>
                                    <a class="muted" href="<?=
url('cart.php?action=remove&key=' . urlencode($item['key'])) ?>">Вида-
лити</a>
                                </div>
                                <input class="input" type="number"
name="qty[<?= e($item['key']) ?>]" value="<?= (int)$item['qty'] ?>"
min="0" max="99">
                                <b><?= money($item['subtotal']) ?></b>
                            </div>
                        <?php endforeach; ?>
                    </div>
                    <div class="cart-actions">

```

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						77
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        <button class="btn btn-outline" type="submit"
name="action" value="update">Оновити кошик</button>
        <a class="btn btn-red" href="<?=
url('cart.php?action=clear') ?>">Очистити</a>
    </div>
</form>
<?php endif; ?>
</section>
<aside class="summary">
    <h3>Разом</h3>
    <div class="summary-row"><span>Товарів</span><b><?= cart_count()
?></b></div>
    <div class="summary-row"><span>Доставка</span><b>за тари-
фом</b></div>
    <div class="summary-row total"><span>Сума</span><b><?=
money($total) ?></b></div>
    <?php if ($items): ?>
        <button class="btn" style="width:100%;margin-top:16px"
type="submit" name="action" value="checkout" form="cart-form">Оформити
замовлення</button>
    <?php endif; ?>
</aside>
</div>
<?php include __DIR__ . '/includes/footer.php'; ?>

```

Обробка Б.3 – Автентифікація адміністратора admin/login.php та config/auth.php

```

<?php
// config/auth.php
require_once __DIR__ . '/db.php';
require_once __DIR__ . '/helpers.php';

function admin_logged_in(): bool
{
    return !empty($_SESSION['admin_id']);
}

function require_admin(): void
{
    if (!admin_logged_in()) {
        header('Location: ' . url('admin/login.php'));
        exit;
    }
}
<?php
// admin/login.php
require_once __DIR__ . '/../config/db.php';
require_once __DIR__ . '/../config/helpers.php';
if (!empty($_SESSION['admin_id'])) {
    header('Location: ' . url('admin/index.php'));
}

```

					ІПЗ.КР.Б – 121 – 26 - ПЗ	Арк.
						78
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        exit;
    }
    $error = '';
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $username = trim($_POST['username'] ?? '');
        $password = trim($_POST['password'] ?? '');
        $stmt = $pdo->prepare("SELECT * FROM admins WHERE username=? LIMIT
1");
        $stmt->execute([$username]);
        $admin = $stmt->fetch();
        if ($admin && hash('sha256', $password) === $admin['password_hash'])
        {
            $_SESSION['admin_id'] = $admin['id'];
            $_SESSION['admin_name'] = $admin['username'];
            header('Location: ' . url('admin/index.php'));
            exit;
        }
        $error = 'Невірний логін або пароль';
    }
?>
<!doctype html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Вхід в адмін-панель</title>
    <link rel="stylesheet" href="<?= asset('css/style.css') ?>">
</head>
<body class="login-page">
    <form class="login-box" method="post">
        <a class="logo" href="<?= url('index.php') ?>"><span class="logo-
mark">A</span><span>ATLANT</span></a>
        <h1>Адмін-панель</h1>
        <?php if ($error): ?><div class="alert alert-error"><?= e($error)
?></div><?php endif; ?>
        <div class="field">
            <label>Логін</label>
            <input class="input" name="username" required autofocus>
        </div>
        <div class="field">
            <label>Пароль</label>
            <input class="input" type="password" name="password"
required>
        </div>
        <button class="btn" style="width:100%"
type="submit">Увійти</button>
    </form>
</body>
</html>

```

					ІІЗ.КР.Б – 121 – 26 - ІЗ	Арк.
						79
Змн.	Арк.	№ докум.	Підпис	Дата		

Обробка Б.4 – Керування замовленнями admin/orders.php

```
<?php
require_once __DIR__ . '/../config/auth.php';
require_admin();
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $id = (int)($_POST['id'] ?? 0);
    $status = $_POST['status'] ?? 'new';
    $allowed = ['new', 'processing', 'completed', 'cancelled'];
    if (in_array($status, $allowed, true)) {
        $pdo->prepare("UPDATE orders SET status=? WHERE id=?")-
>execute([$status, $id]);
        flash('Статус замовлення оновлено');
    }
    header('Location: ' . url('admin/orders.php'));
    exit;
}
$pageTitle = 'Замовлення – ATLANT';
$orders = $pdo->query("SELECT * FROM orders ORDER BY created_at DESC")-
>fetchAll();
include __DIR__ . '/partials/header.php';
?>
<div class="admin-top"><div><h1>Замовлення</h1><p class="muted">Перегляд
клієнтів, складу замовлення та зміна статусу</p></div></div>
<div class="table-wrap">
<table>
    <thead><tr><th>ID</th><th>Клієнт</th><th>Доставка</th><th>То-
вари</th><th>Сума</th><th>Статус</th></tr></thead>
    <tbody>
        <?php foreach ($orders as $order):
            $stmt = $pdo->prepare("SELECT * FROM order_items WHERE
order_id=?");
            $stmt->execute([(int)$order['id']]);
            $items = $stmt->fetchAll();
        ?>
        <tr>
            <td><b>#<?=(int)$order['id'] ?></b><div class="muted"><?=(int)$order['created_at'] ?></div></td>
            <td><b><?=(int)$order['customer_name'] ?></b><div
class="muted"><?=(int)$order['phone'] ?></div><div class="muted"><?=(int)$order['email'] ?></div></td>
            <td><?=(int)$order['city'] ?><br><span class="muted"><?=(int)$order['address'] ?></span></td>
            <td><?php foreach ($items as $i): ?><div><?=(int)$i['product_name'] ?> <span class="muted"><?=(int)$i['variant_name'] ?>
x<?=(int)$i['quantity'] ?></span></div><?php endforeach; ?></td>
            <td><b><?=(int)$order['total'] ?></b></td>
        </tr>
    </tbody>
</table>
</div>
```