Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розроблення автоматизованої системи для продажу готової продукції виробничого підприємства
(тема)

Виконав:
здобувач 2024 року навчання,
групи КІТПВм-24-2
Юрій ТИЩЕНКО
(власне ім'я, прізвище)

Спеціальність 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Комп'ютерно-інтегровані технологічні процеси і виробництва
(повна назва освітньої програми)

Керівник доц. Софія ХРУСТАЛЬОВА
(посада, ім'я, прізвище)

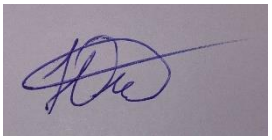
Допускається до захисту

Завідувач кафедри _____ Ігор НЕВЛЮДОВ
(підпис) (прізвище, ініціали)

2025 р.

Я, ПІБ, як здобувач вищої освіти ХНУРЕ, розумію і підтримую політику закладу з академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Дата

17 грудня 2025р  Юрій ТИЩЕНКО

Харківський національний університет радіоелектроніки

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти другий (магістерський)
Спеціальність 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка
(код і повна назва)
Тип програми освітньо-професійна
Освітня програма Комп'ютерно-інтегровані технологічні процеси і виробництва
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри КІТАР

_____
(підпис)
«_____»_____ 20 ___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Тищенку Юрію Олексійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення автоматизованої системи для продажу готової продукції виробничого підприємства
затверджена наказом університету від 10.11.2025 р. № 1029 Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 22.12.2025 р.
3. Вихідні дані до роботи Перелік використаних програмних засобів: AMPPS, Draw.io. Середовище розробки: Visual Studio Code, PhpMyAdmin. Мова програмування PHP, JavaScript. Технічне забезпечення: ПК з процесором Core i5 і вище.
Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ
4.2 Огляд та аналіз існуючих методів, засобів та автоматизованих систем для продажу готової продукції виробничого підприємства
4.3 Розроблення структурної схеми та алгоритму роботи автоматизованої системи для продажу готової продукції виробничого підприємства
4.4 Реалізація автоматизованої системи продажу готової продукції виробничого підприємства у вигляді програмного засобу
4.5 Охорона праці

1. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій демонстраційний матеріал, представлений у форматі презентації – 8 арк. ф. А4

2. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз існуючих методів, засобів та автоматизованих систем продажу готової продукції	01.09.25	виконано
2	Розроблення структурної схеми та алгоритму роботи автоматизованої системи продажу готової продукції	20.09.25	виконано
3	Реалізація автоматизованої системи продажу готової продукції виробничого підприємства у вигляді програмного засобу	20.10.25	виконано
4	Експериментальне дослідження та тестування системи	20.11.25	виконано
5	Висновки	01.12.25	виконано
6	Оформлення пояснювальної записки	14.12.25	виконано
7	Подання роботи на перевірку Інтернет-сервісом StrikePlagiarist	18.12.25	виконано
8	Подання роботи на рецензію	18.12.25	виконано
9	Подання роботи на підпис зав. кафедри	19.12.25	виконано
10	Подання кваліфікаційної роботи в ЕК	19.12.25	виконано

Дата видачі завдання 01.09.2025 р.

Здобувач _____ Юрій ТИЩЕНКО
(підпис) (прізвище, ініціали)

Керівник роботи _____ доц. Софія ХРУСТАЛЬОВА
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 89 с., 17 рис., 3 дод., 31 джерела.

БАЗА ДАНИХ, ОПТИМІЗАЦІЯ ПРОЦЕСІВ, CSS, HTML, JAVASCRIPT, PHP, АВТОМАТИЗОВАНА СИСТЕМА, ГОТОВА ПРОДУКЦІЯ.

Об’єктом дослідження є процес управління продажами готової продукції виробничого підприємства.

Предметом дослідження є програмний засіб для управління продажами готової продукції виробничого підприємства.

Мета дослідження полягає в підвищенні ефективності продажу готової продукції виробничого підприємства за рахунок розроблення автоматизованої системи.

В ході дослідження було проаналізовано існуючі системи для продажу готової продукції. Розглянуто актуальні питання електронної комерції та відбудови країни у післявоєнні роки, шляхом поєднання виробничих підприємств, постачальників та замовників готової продукції. На базі цього аналізу було розроблено автоматизовану систему для продажу готової продукції виробничого підприємства.

Результатом дослідження є автоматизована система реалізована у вигляді програмного засобу для продажу готової продукції виробничого підприємства.

Також отримані результати відповідають переліку Цілей сталого розвитку, зокрема Цілі 9 Промисловість, інновації та інфраструктура.

THE ABSTRACT

Explanatory note: 89 p., 63 fig., 3 add., 31 sources.

DATABASE, PROCESS OPTIMIZATION, CSS, HTML, JAVASCRIPT, PHP, AUTOMATED SYSTEM, FINISHED PRODUCTS.

The object of the development is the process of managing sales of finished products at a manufacturing enterprise.

The subject of the development is a software tool for managing the sales of finished products of a manufacturing enterprise.

The purpose of the work is to increase the efficiency of selling finished products by developing an automated system.

During the work, existing systems for selling finished products were analyzed. Relevant issues of e-commerce and the post-war reconstruction of the country were considered through the integration of manufacturing companies, suppliers, and customers of finished products. Based on this analysis, an automated system for selling finished products of a manufacturing enterprise was developed.

The result of this work is a software tool for an automated system designed to manage the sales of finished products at a manufacturing enterprise.

The obtained results also correspond to the list of Sustainable Development Goals, in particular Goal 9: Industry, Innovation and Infrastructure.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ДСТУ – державні стандарти України;

ER-модель (Entity-relationship model) – семантична модель даних;

E-commerce – електронна комерція;

SQL (Structured Query Language) – мова структурованих запитів;

UML (Unified Modeling Language) – уніфікована мова моделювання;

VS Code (Visual Studio Code) – середовище розробки;

БД – база даних;

СУБД – система управління базами даних.

ВСТУП

Сьогодні важко знайти успішне підприємство, яке б ігнорувало онлайн-продажі. Ринок диктує свої умови: навіть великі виробництва, що роками працювали за старими схемами збуту, тепер змушені йти в інтернет. Це не просто данина моді, а питання виживання та економії ресурсів. Автоматизація продажів прибирає зайву рутину і дозволяє менеджменту зосередитися на головному.

Ось чому розробка власної системи для продажу продукції зараз на часі. Клієнт змінився – він не хоче чекати на дзвінок менеджера чи їхати в офіс, щоб дізнатися ціну. Йому потрібно зробити замовлення тут і зараз, у кілька кліків. Тому створення зрозумілого та швидкого веб-додатку – це прямий шлях до того, щоб не програти конкурентам.

Мета дослідження – підвищення ефективності продажу готової продукції виробничого підприємства.

Об’єкт дослідження – процес управління продажами готової продукції виробничого підприємства.

Предмет дослідження – програмний засіб для управління продажами готової продукції виробничого підприємства.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– провести огляд та аналіз існуючих методів, засобів та автоматизованих систем для продажу готової продукції виробничого підприємства;

– розробити структурні схеми та алгоритм роботи автоматизованої системи для продажу готової продукції виробничого підприємства;

– реалізувати автоматизовану системи продажу готової продукції виробничого підприємства у вигляді програмного засобу;

– провести тестування та випробувати автоматизовану систему для продажу готової продукції виробничого підприємства;

– оформити кваліфікаційну роботу згідно ДСТУ 3008:2015 [1], а також з методичними вказівками з підготовки й оформлення кваліфікаційної роботи

здобувачами другого (магістерського) рівня вищої освіти спеціальності 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка освітньої програми «Комп'ютерно-інтегровані технологічні процеси і виробництва» [2].

Результати досліджень за темою роботи опубліковані в [3].

1 ОГЛЯД ТА АНАЛІЗ ІСНУЮЧИХ МЕТОДІВ, ЗАСОБІВ ТА АВТОМАТИЗОВАНИХ СИСТЕМ ДЛЯ ПРОДАЖУ ГОТОВОЇ ПРОДУКЦІЇ ВИРОБНИЧОГО ПІДПРИЄМСТВА

У сучасних умовах ринкової економіки ефективна діяльність виробничого підприємства значною мірою залежить від організації процесів реалізації готової продукції. Традиційні методи продажу, які базуються на ручному обліку замовлень, телефонних зверненнях чи паперовій документації, є малоефективними, трудомісткими та схильними до помилок. Тому виникає потреба в автоматизації цих процесів шляхом впровадження інформаційних систем, що забезпечують зручність, швидкість та достовірність обробки даних.

Сервіс «Rebuilding Ukraine», створений для підтримки відбудови України шляхом поєднання виробничих підприємств, постачальників та замовників готової продукції.

Проект спрямований на автоматизацію процесів продажу, замовлення та управління готовою продукцією, щоб сприяти ефективному відновленню промисловості та економіки країни.

На сайті є головна сторінка з матеріалами, які користуються попитом, перевагами, методами виготовлення, ліцензіями, сертифікатами та потужностями виробничого підприємства. Каталог, де клієнт може відфільтрувати матеріали за категорією або використати пошук у каталозі, знайти необхідний товар, ознайомитись з описом, технічними характеристиками та зручно додати у кошик. Інформативні сторінки Про нас і Контакти. А також особистий кабінет, де зберігаються дані про клієнта, про кошик клієнта, про минулі замовлення та можливість оформити й сплатити замовлення. Якщо потреба змінилась – клієнт може редагувати кошик до оформлення замовлення. Адміністратор може додавати, редагувати або видаляти товари з каталогу. Також може додавати або видаляти сертифікати з головної сторінки. Є можливість для перегляду статистики продажів, щоб визначити популярні товари, а також контроль залишків на складі.

1.1 Цифрова економіка та електронна торгівля в Україні

Сучасний бізнес характеризується постійним зростанням можливостей компаній-постачальників, що у свою чергу призводить до підвищення глобальної конкуренції та поліпшення якості товарів та послуг. Змінюються способи організації та керування бізнесом, відбувається модернізація бізнес-процесів та впровадження систем автоматизації. Для успішного ведення бізнесу все частіше залучаються комп'ютерні системи та мережі. За визначенням комісії ООН з промислового розвитку існує чотири основні компоненти бізнесу, а саме: маркетинг; виробництво; продаж; платежі. Якщо дві або більше складових реалізуються із застосуванням інформаційних комп'ютерних технологій (систем та мереж), бізнес є електронним.

На сьогодні електронне ведення бізнесу охоплює три складові:

– електронний документообіг;

– електронну систему платежів;

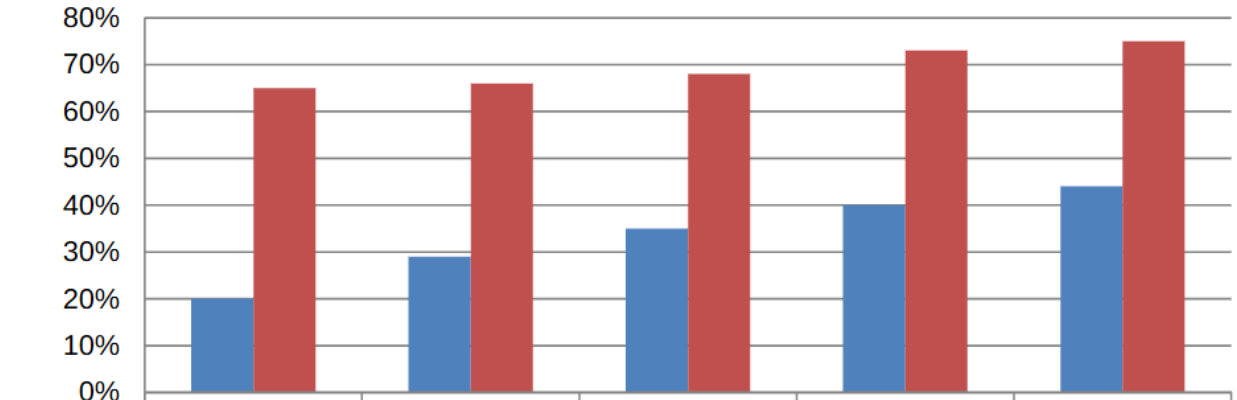
– електронну торгівлю.

Найважливішою складовою е-бізнесу є е-комерція, яка охоплює не тільки операції купівлі-продажу, а й супровід процесів створення попиту на продукцію і послуги, автоматизацію адміністративних функцій, пов'язаних з онлайновими продажами і обробленням замовлень, а також із вдосконаленням обміну інформацією між партнерами. Електронна комерція (е-комерція) – різновид бізнес-активності, в якій взаємодія суб'єктів бізнесу з купівлі-продажу товарів і послуг (як матеріальних, так й інформаційних) здійснюється з допомогою глобальної комп'ютерної мережі Internet або будь-якої іншої інформаційної мережі. Об'єктом електронної комерції є реалізація товарів і послуг за допомогою засобів електронного обміну даними. Суб'єктами електронної комерції є фізичні та юридичні особи. Поняття «електронна комерція» ширша, ніж інтернет-комерція, оскільки до нього входять усі види комерційної діяльності, здійснюваної електронним шляхом. Інтернет-комерція – електронна комерція, обмежена використанням тільки комп'ютерної мережі Інтернет.

Україна активно розвивається у сфері електронної комерції, що відкриває широкі перспективи для підприємств та споживачів. Зростання кількості інтернет-користувачів сприяє збільшенню обсягів онлайн-продажів, що створює нові можливості для бізнесу та сприяє розвитку економіки країни [4].

1.1.1 Розвиток електронної комерції в Україні та Європейському Союзі

Диджиталізація національної економіки характеризується впровадженням цифрових технологій, які направлені на оптимізацію та автоматизацію бізнес-процесів, покращення комунікаційних зв’язків із споживачами та підвищення ефективності господарської діяльності суб'єктів господарювання. Використання інформаційних технологій стає умовою забезпечення конкурентоспроможності як окремих підприємств і організацій, так і країн у цілому. Якість товарів і послуг покращується, а собівартість зменшується, що призводить до збільшення конкуренції між компаніями, що, своєю чергою, призводить до перебудови виробничих та економічних процесів. До того ж диджиталізація сприяє підвищенню рівня розвитку роздрібної торгівлі в Україні, збільшенню частки онлайн-продажів, зростанню прибутку суб'єктів господарювання, підвищенню рівня задоволеності покупців. Аналіз динаміки обсягів ринку електронної комерції в Україні показав, що до 2022 року Україна демонструвала значні успіхи на ринку електронної комерції (рис. 1.1). Найбільше зростання +48 % відповідно до попереднього періоду спостерігалось у 2020 році, що пов’язано також із пандемією COVID-19, яка стала акселератором розвитку цифровізації. У 2021 році український ринок електронної комерції значно зріс – на 27 %, порівняно зі світовим темпом зростання на 15%. Прогнозні статистичні дані, які аналізувались до 2022 року, свідчили про те, що ринок e-commerce в Україні повинен був продовжувати своє зростання у наступні роки. Але всі ці прогнози стали неактуальними після 24 лютого 2022 року [5].



	2017 р.	2018 р.	2019 р.	2020 р.	2021 р.
Україна	20%	29%	35%	40%	44%
ЄС	65%	66%	68%	73%	75%

Рисунок 1.1 – Динаміка електронної торгівлі в Україні та ЄС протягом 2017–2021 рр., %

На рисунку 1.1 можна помітити, що протягом останніх п’яти років український та європейський ринки електронної комерції нарощували свої обсяги та завойовували прихильність все більшої кіль-кості споживачів, проте військова агресія Росії призупинила стрімкий розвиток вітчизняної електронної комерції. Активні процеси реформування сфери електронної комерції в ЄС вимагають від українського уряду розробки державної регуляторної політики в цій сфері з метою подолання бар’єрів та перешкод для транскордонної електронної комерції з урахуванням нових викликів, ризиків та можливостей розвитку цієї сфери. Україна виконала свої зобов’язання щодо імплементації положень Директиви №2000/31/ЄС у національне законодавство, проте ще залишаються негармонізованими з нормами права ЄС деякі нормативні акти, що стосуються електронної комерції. Потребують врегулювання питання щодо створення єдиного державного органу в галузі електронної комерції, з метою реалізації цілісної державної політики в цій сфері, та детінізації діяльності продавців товарів, робіт, послуг в системах електронної комерції. В повоєнний період електронна комерція здатна стати одним з наймогутніших сегментів національної економіки за умови економічного стимулювання внутрішнього спожив-чого ринку та залучення клієнтів з інших країн ЄС після приєднання до Єдиного цифрового ринку Європейського Союзу [6].

1.2 Тенденції попиту на будівельні матеріали

Повномасштабне вторгнення кардинальним чином змінило географію та обсяг будівельного ринку, спричинило кадрові зміни, вплинуло на нормативну базу, а також відкрило нові напрямки діяльності для українських будівельних компаній. На сьогодні загальні втрати становлять більше $150 млрд, в тому числі найбільші збитки зазнав житловий фонд ($56 млрд), інфраструктура ($37 млрд) та промисловість ($12 млрд). За два роки повномасштабного вторгнення близько 15% виробничих потужностей будівельних матеріалів зазнали руйнувань. При цьому найбільших втрат зазнали сегменти металопрокату і сухих гіпсових сумішей.

У 2022 році обсяг будівельного ринку України зменшився приблизно на 65%. У 2023 році спостерігається тенденція до збільшення споживання будівельної продукції та послуг і за результатами року очікується зростання ринку на 25%. При цьому обсяг ринку житлової нерухомості у гривневому еквіваленті залишиться на рівні минулого року, нежитлової нерухомості – зросте на 15%, а інженерних споруд – покаже приріст 40%.

Кардинально змінилася структура попиту на нові об’єкти житлової нерухомості України. Прифронтові регіони зазнали найбільшого падіння обсягів будівництва до майже 90%, центральна частина – зменшення до 70%, а на заході будівництво зросло на 15%, що пов’язано з релокацією бізнесу та внутрішньо переміщених осіб, а також активним розвитком курортної нерухомості у Карпатах.

Ринок первинної нерухомості переорієнтувався переважно на захід України. Девелопери в інших регіонах зосереджені здебільшого на закінченні проєктів, розпочатих до березня 2022 року. Наразі більшість інвесторів не наважуються починати нові будівельні проєкти і займають вичікувальну позицію.

В центрі, на півночі та сході України зріс попит на послуги з відбудови зруйнованих будівель і споруд. Бізнес, що зазнав руйнувань об’єктів нерухомості, потребує реконструкції будівель задля відновлення функціонування.

Найбільш швидкозростаючим сегментом будівництва наразі є відбудова інфраструктури, в першу чергу мостів і об’єктів соціального призначення, за рахунок бюджетних та донорських коштів.

У відповідь на загрози воєнного часу з’явився нові сегменти будівельного ринку: конструкції для захисту об’єктів критичної інфраструктури та модульні залізобетонні укриття, призначені для захисту людей під час повітряних тривог, артилерійських обстрілів тощо.

Наразі більшість українських інвесторів готуються до відбудови і активно прораховують вартість будівництва, але очікують закінчення воєнних дій для старту нових проєктів.Міжнародні фінансові інституції планують залучатися до відбудови і поступово заходять на ринок України. Ключовими критеріями фінансування повоєнного будівництва в Україні будуть прозорість та швидкість реалізації проєктів, тому міжнародні організації вже сьогодні починають налагоджувати партнерство з надійними українськими будівельниками. При цьому у пріоритеті стають компанії, які працюють з європейськими матеріалами і технологіями, що дає змогу створювати сучасні архітектурно привабливі та енергоефективні будівлі [7].

1.3 Використання електронної комерції для онлайн-торгівлі

Електронна комерція – це сфера економіки, де з використанням Інтернету можливе проведення комерційних операцій між підприємствами або між підприємством та споживачами.

Сучасний розвиток цифрових технологій зумовив перехід більшості бізнес-процесів у віртуальне середовище, де одним із ключових напрямів є електронна комерція. Електронна комерція виступає не лише як інструмент збуту, а як повноцінна бізнес-модель, що забезпечує підприємствам можливість автоматизувати процеси продажу, спілкування з клієнтами та управління логістикою. Вона базується на використанні мережевих інформаційних технологій,

які дозволяють здійснювати комерційні операції через Інтернет без фізичної присутності сторін.

Електронна комерція стала закономірним етапом еволюції світового ринку, на який вплинуло поєднання розвитку інформаційних систем, глобалізації торгівлі та зростання попиту на швидкі й зручні форми придбання товарів. В Україні активне впровадження електронної комерції пов'язане з розширенням доступу до мережі Інтернет, цифровізацією державних сервісів і зміною поведінки споживачів, які все частіше віддають перевагу онлайн-покупкам.

Використання електронної комерції для онлайн-торгівлі передбачає побудову цілісної системи, яка об'єднує веб-платформу, систему управління замовленнями, платіжну інфраструктуру та базу даних клієнтів. Ефективність електронної комерції залежить від рівня інтеграції цих елементів між собою. Онлайн-платформа стає не лише вітриною для демонстрації продукції, а й інструментом управління маркетингом, аналітикою продажів, контролем залишків на складі та підтримкою клієнтів у режимі реального часу

Отже, електронна комерція сьогодні розглядається не лише як інноваційна форма торгівлі, а як комплексний механізм підвищення ефективності діяльності підприємства. Її використання забезпечує оптимізацію витрат, прискорення бізнес-процесів і гнучке реагування на зміни ринкової кон'юнктури. Електронна комерція стає невід'ємною частиною цифрової економіки, сприяючи формуванню нової моделі взаємодії між виробником і споживачем, що ґрунтується на прозорості, швидкості та довірі [8].

1.3.1 Типи бізнес-моделей електронної комерції

Електронна комерція охоплює різні форми взаємодії між учасниками ринку, що відрізняються за напрямом комунікації, структурою відносин та характером продажів. Залежно від того, хто виступає продавцем і хто – покупцем, виокремлюють декілька ключових моделей.

Модель B2C передбачає, що компанія продає товари або послуги безпосередньо кінцевому споживачу через Інтернет. Типовими прикладами є

онлайн-магазини, платформи бронювання, сервіси цифрового контенту та ін. У цій моделі компанія формує власний інтернет-майданчик, залучає клієнтів за допомогою диджитал-маркетингу, здійснює оплату та доставку товарів, а також надає онлайн-підтримку покупцям. Особливостями B2C є великий масштаб аудиторії, різноманітність товарів і висока конкуренція.

Модель B2B характеризується тим, що компанія продає свої продукти або послуги не кінцевому споживачу, а іншому бізнесу. Тут угоди укладаються між юридичними особами, часто на великих обсягах, з тривалішими циклами, складною логістикою та необхідністю довгострокового партнерства. Маркетинг у моделі B2B більш персоналізований, акцентовано на утриманні клієнтів, а не лише на залученні нових.

У моделі C2C продавцем і покупцем виступають самі споживачі, а роль платформи полягає у посередництві їхньої взаємодії. Користувачі реєструються, розміщують оголошення, узгоджують умови і здійснюють оплату й доставку – або з підтримкою платформи, або самостійно. Перевагою цієї моделі є широкий вибір і демократичність, а недоліками – ризики шахрайства або відсутність гарантій для покупця.

Модель C2B – це зворотний до поширених: споживач пропонує компанії свої товари або послуги. Часто така модель використовується у фрилансі або креативних індустріях, коли користувач має навички, ідеї або контент, які компанія готова придбати. Для бізнесу це можливість залучити нові таланти чи рішення, для виконавців – отримати оплату за свої навички без трудового договору [9].

1.3.2 Переваги ведення електронної комерції

Електронна комерція поступово стала одним із ключових напрямів розвитку сучасного бізнесу, оскільки поєднує зручність, мобільність та гнучкість у взаємодії між продавцем і покупцем. Однією з головних переваг є суттєве зменшення витрат на утримання бізнесу. На відміну від традиційних магазинів, які потребують оренди приміщення, оплати комунальних послуг та витрат на персонал, онлайн-бізнес дозволяє уникнути більшості цих накладних витрат. Відсутність фізичної

вітрини знижує фінансове навантаження на підприємця, а отже, дає змогу спрямовувати більше ресурсів на розвиток товарного асортименту та маркетинг.

Ще однією значною перевагою є цілодобова доступність. Інтернет-магазин не має обмежень у часі – він може працювати 24/7, приймаючи замовлення тоді, коли це зручно клієнту. Це забезпечує стабільний потік продажів навіть без постійної присутності персоналу. Сучасні системи автоматизації дозволяють обробляти замовлення, формувати накладні та надсилати повідомлення клієнтам автоматично, що значно оптимізує робочі процеси.

Крім того, електронна комерція відкриває широкі можливості для масштабування. Якщо у звичайного магазину є обмеження у вигляді площі, то онлайн-бізнес може легко розширювати асортимент, додавати нові категорії товарів або змінювати цінову політику практично миттєво. Це дозволяє оперативно реагувати на зміни попиту та гнучко адаптуватися до ринку.

Важливою перевагою є і географічна незалежність. Онлайн-торгівля дає змогу продавати товари не лише у своєму місті, а й у будь-якому регіоні країни або навіть за її межами. Підприємство може працювати з дому чи офісу, а клієнт – отримувати товар у зручний спосіб через службу доставки. Це створює умови для розвитку малого бізнесу, який раніше не мав змоги конкурувати з великими мережами [10].

Ще однією сильною стороною електронної комерції є прозорість і контроль процесів. Підприємці мають можливість у реальному часі відстежувати продажі, обробку замовлень та відправлення товарів. Такі інструменти дають змогу швидко виявляти помилки або затримки й підвищувати якість обслуговування.

Окрім цього, цифрова природа електронної комерції дозволяє збирати та аналізувати дані про клієнтів. Це не лише контактна інформація, а й дані про їхні купівельні звички, уподобання, частоту покупок. Завдяки цьому бізнес може розробляти персоналізовані пропозиції, створювати програми лояльності та ефективніше взаємодіяти з цільовою аудиторією.

Нарешті, електронна комерція показала свою стійкість у період глобальних криз, зокрема пандемії COVID-19. У той час, коли традиційні магазини змушені

були закриватися, онлайн-бізнес залишався доступним і продовжував функціонувати, що дозволило багатьом компаніям не лише зберегти прибуток, а й розширити клієнтську базу.

Отже, електронна комерція поєднує у собі економічну ефективність, доступність, масштабованість і технологічну гнучкість, що робить її одним із найперспективніших напрямів розвитку сучасного підприємництва [10].

1.3.3 Недоліки ведення електронної комерції

Незважаючи на численні переваги, електронна комерція має й певні виклики, які підприємству необхідно враховувати. Один із важливих недоліків – це відсутність можливості оцінити товар фізично перед покупкою, що може призводити до розчарування клієнта, якщо реальний товар не відповідатиме очікуванням.

Крім того, існує підвищений ризик шахрайства та витоку даних, адже інтернет-операції вимагають надійного захисту особистої інформації покупців. Логістика – ще одна складова, що може стати слабким місцем: залежність від сторонніх компаній доставки може призвести до затримок, пошкоджень товару або втрати замовлення, що негативно впливає на репутацію бізнесу.

В умовах електронної комерції діє жорстка конкуренція, адже низький бар’єр входу дозволяє багатьом легко створити онлайн-магазин, а це ускладнює просування нового бренду та збільшує витрати на маркетинг. Технічні проблеми – ще один виклик: збій у роботі сайту, повільне завантаження або недоступність платіжної системи можуть швидко вплинути на втрату клієнтів.

Також варто зазначити, що повернення товарів в e-commerce може бути складнішим, ніж у фізичному магазині – процес включає витрати на доставку назад, додатковий час та складність для споживача, що впливає на його лояльність.

Усе це вимагає від бізнесу постійного вдосконалення платформи, впровадження нових технологій та реагування на зміни ринку, що може потребувати значних ресурсів [11].

1.4 Аналіз існуючих рішень та систем-аналогів

Наразі в сегменті e-commerce представлено багато рішень, починаючи від універсальних платформ і закінчуючи інтернет-магазинами окремих торговельних мереж. Більшість із них закривають базові потреби: дають змогу переглянути каталог, додати товар у кошик і оформити покупку. Однак зростаючі вимоги клієнтів до зручності інтерфейсів виявляють архітектурні слабкості цих систем. Найпоширеніші недоліки – це інформаційний шум на сторінках, відсутність інтелектуальних інструментів пошуку та застарілі підходи до консультування покупців.

Для того щоб довести доцільність створення власного вебзастосунку та визначити його функціональний вектор, необхідно проаналізувати провідні системи-аналоги на українському ринку. Виявлення недоліків у конкурентів дозволить уникнути їх у власному проєкті, зробивши покращений користувацький досвід та оптимізовану бізнес-логіку ключовими перевагами нової системи.

Для аналізу було обрано декілька із провідних ресурсів на ринку продажу готової продукції виробничого підприємства, інтерфейс яких зображено на рисунках 1.2-1.4.

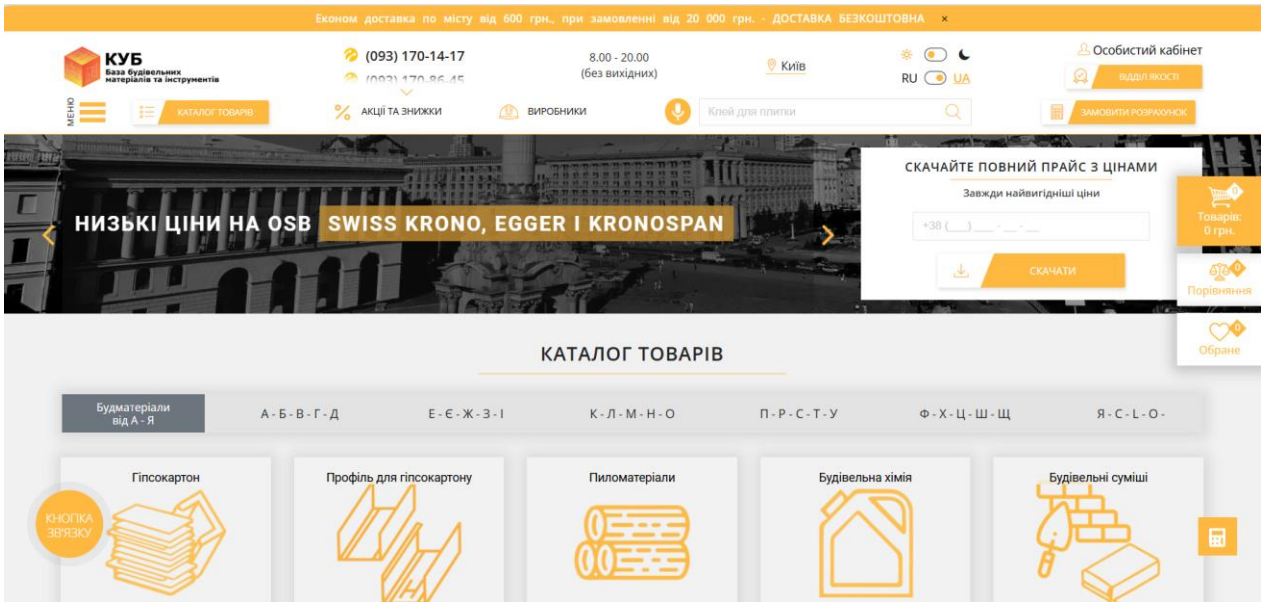


Рисунок 1.2 – інтернет-магазин «КУБ»

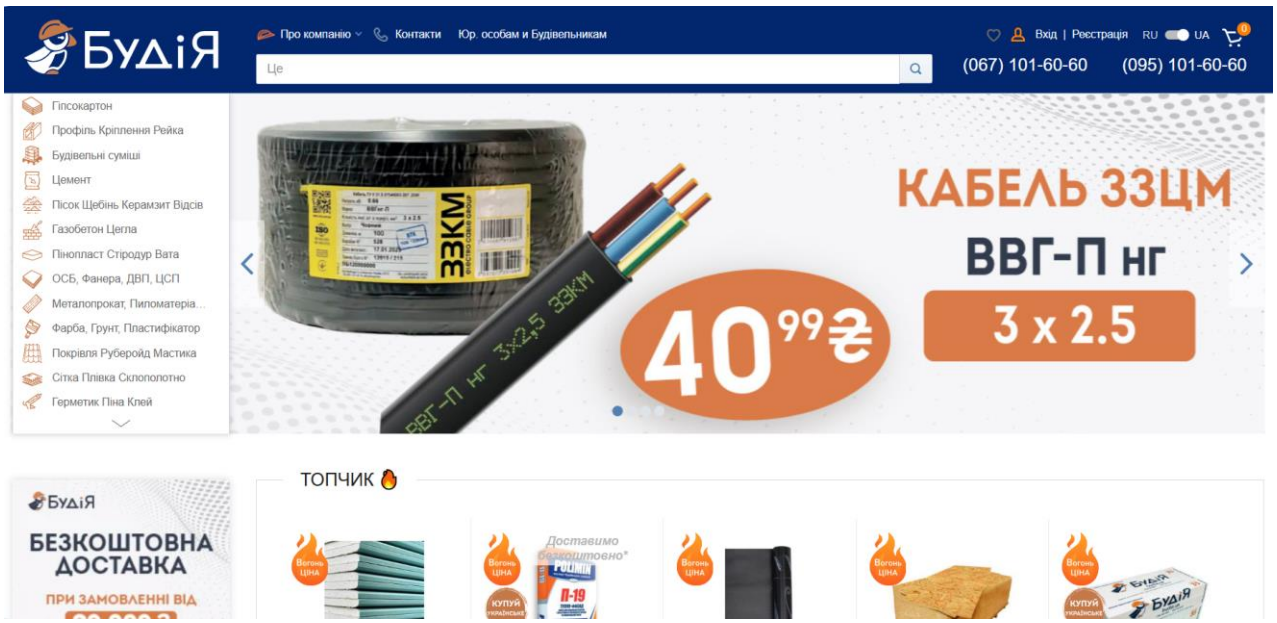


Рисунок 1.3 – інтернет-магазин «БудіЯ»

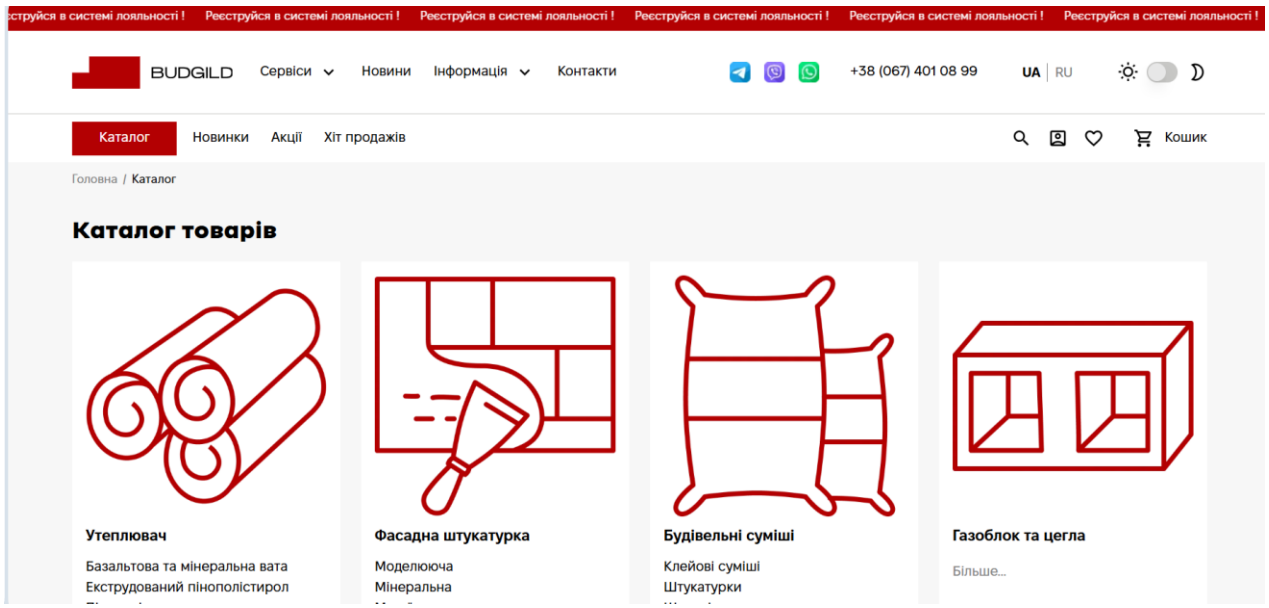


Рисунок 1.4 – інтернет-магазин «BUDGILD»

По-перше, організація простору на екрані залишає бажати кращого. Конкуренти впадають у крайнощі: сайт «КУБ» намагається вмістити все й одразу, через що користувач губиться в десятках кнопок, а «БудіЯ», навпаки, віддає найцінніше місце під гігантські банери, змушуючи покупця скролити до товарів.

По-друге, навігація не відповідає сучасним стандартам. Пошук за алфавітом – це архаїзм, який незручний для більшості користувачів. Статичні меню забирають

місце, а пошук працює примітивно: якщо не вгадав з точним словом – товар не знайдеш. Жоден із сайтів не використовує AI для розуміння суті запиту.

По-третє, є проблеми з позиціонуванням. «КУБ» дратує вимогою залишити номер телефону заради прайсу (що ріже конверсію), а «БудіЯ» намагається загравати з аудиторією через сленг типу «ТОПЧИК». Для серйозних забудовників це виглядає непрофесійно.

І нарешті, відсутня допомога у виборі. Жодна система не допоможе, скільки матеріалу треба на 20 квадратів стіни. Сайти працюють просто як електронні каталоги, ігноруючи потребу клієнта в автоматизованій консультації.

Аналіз показав, що існуючі рішення добре справляються з показом товарів, але не вміють якісно консультувати клієнта. Щоб виправити цей дисбаланс, доцільно розробити систему з іншою філософією: замість складних меню – простий інтерфейс, а замість очікування дзвінка менеджера – миттєва допомога AI-консультанта.

1.5 Опис сфери діяльності

Сфера діяльності, пов’язана з розробленням автоматизованої системи для продажу готової продукції виробничого підприємства, стосується створення сучасних цифрових рішень, які допомагають підприємствам ефективно організувати процес реалізації своєї продукції. Основна мета таких систем – зробити процес продажу більш зручним, швидким і контрольованим як для компанії, так і для її клієнтів.

Подібні системи забезпечують автоматизацію ключових бізнес-процесів: від обліку товарів і формування замовлень до здійснення оплати та контролю доставки. Завдяки цьому підприємство може зменшити кількість ручної роботи, уникнути помилок при оформленні замовлень і забезпечити оперативну обробку запитів клієнтів.

Для покупців така система створює зручний інструмент для взаємодії з компанією – через особистий кабінет вони можуть переглядати доступні товари,

оформлювати замовлення, стежити за їхнім статусом і отримувати актуальну інформацію щодо продукції.

З боку керівництва підприємства система відкриває можливості для аналізу продажів, формування звітів, контролю залишків і прогнозування попиту. Усе це сприяє підвищенню ефективності роботи компанії, покращенню обслуговування клієнтів і створенню конкурентних переваг на ринку.

Таким чином, розроблення автоматизованої системи для продажу готової продукції – це не лише технічне завдання, а й важливий етап цифрової трансформації підприємства, що поєднує виробництво, логістику та електронну комерцію в єдиний узгоджений механізм.

Розглянемо процес розробки компонентів для системи підтримки користувачів на прикладі системи для продажу будівельних матеріалів.

Цей веб-застосунок може включати такі ключові функції:

– вибір будівельних матеріалів в каталозі. Користувач може вибрати матеріали за різними категоріями, такими як гідроізоляція, теплоізоляція, фундамент, сухі суміші, матеріали для покрівлі;

– перегляд деталей замовлення. Користувач може детально ознайомитися з кожним товаром, включаючи опис, характеристику, ціну, сферу застосування, кількість в наявності на складах тощо;

– оформлення замовлення. Користувач може обрати необхідні матеріали та оформити замовлення, вказавши особисті дані та іншу необхідну інформацію;

– зв'язок зі службою підтримки. Користувач може звернутися до служби підтримки, якщо у нього виникли питання або проблеми з оформленням замовлення.

Одним із ключових аспектів під час створення такої системи є усунення недоліків, що можуть ускладнити роботу користувачів із веб-застосунком. Зокрема, це може стосуватися перевантаженого або незрозумілого інтерфейсу, а також браку детальної інформації про будівельні матеріали. Тому під час розробки важливо зосередитись на тому, щоб система була зручною, логічно побудованою та зрозумілою, а всі її елементи працювали швидко й стабільно.

2 РОЗРОБЛЕННЯ СТРУКТУРНОЇ СХЕМИ ТА АЛГОРИТМУ РОБОТИ АВТОМАТИЗОВАНОЇ СИСТЕМИ ДЛЯ ПРОДАЖУ ГОТОВОЇ ПРОДУКЦІЇ ВИРОБНИЧОГО ПІДПРИЄМСТВА

2.1 Розроблення структурної схеми автоматизованої системи для продажу готової продукції виробничого підприємства

Універсальна мова моделювання (Unified Modelling Language або UML) – це мова позначень або побудови діаграм, призначена для визначення, візуалізації і документування моделей зорієнтованих на об’єкти систем програмного забезпечення. UML не є методом розробки, іншими словами, у конструкціях цієї мови не повідомляється про те, що робити першим, а що останнім, і не надається інструкцій щодо побудови вашої системи, але ця мова допомагає вам наочно переглядати компонування системи і полегшує співпрацю з іншими її розробниками. Розробкою UML керує Object Management Group. Ця мова є загальноприйнятим стандартом графічного опису програмного забезпечення.

UML розроблено для розробки структури зорієнтованого на об’єкти програмного забезпечення, ця мова має дуже обмежену користь для програмування на основі інших парадигм.

Конструкції UML створюються з багатьох модельних елементів, які позначають різні частини системи програмного забезпечення. Елементи UML використовуються для побудови діаграм, які відповідають певній частині системи або точці зору на систему.

Діаграми UML класифікуються на такі види:

– структурні – містять діаграми класів (відображає структуру класів у системі та їхні взаємозв’язки), компонентів (дає змогу уявити структуру високорівневих компонентів системи та їхні залежності) і об’єктів (фокусується на конкретних об’єктах у системі та їхніх взаємодіях);

– поведінкові – включають діаграми випадків використання (описує взаємодію між системою та її користувачем), активностей (моделює потоки управління або процесів у системі) і послідовності (показує взаємодію між різними об’єктами в системі в певній послідовності подій) [12].

2.1.1 Процес розроблення UML діаграм

Процес створення UML-діаграм проходить у кілька етапів:

– спочатку потрібно визначити функціональні та нефункціональні вимоги до системи, а також зрозуміти, як вона має поводитися. Потім аналізують основні сценарії використання, щоб краще усвідомити, що саме потрібно користувачам і які можливості повинна мати система;

– далі обирають типи UML-діаграм, які найкраще підходять для відображення архітектури майбутньої системи. Для їх побудови зручно користуватися такими програмами, як Lucidchart, Visual Paradigm чи іншими аналогами;

– після створення діаграм їх обов’язково перевіряють – чи відповідають вони вимогам проєкту, чи немає помилок або неточностей. Якщо під час роботи змінюються вимоги або структура системи, діаграми оновлюють.

Готові UML-діаграми допомагають краще зрозуміти, як побудована система та як вона працює. Вони також забезпечують спільне бачення проєкту всіма учасниками команди. Для опису структури системи та взаємозв’язків між її частинами зазвичай використовують діаграми класів, компонентів і об’єктів. А щоб показати процеси та взаємодію в системі, створюють діаграми прецедентів, активностей і послідовності [13].

UML-діаграми сьогодні є невід’ємною частиною процесу розробки програмного забезпечення. Вони дають змогу краще зрозуміти вимоги до системи, спроєктувати її архітектуру та забезпечити ефективну комунікацію всередині команди розробників. Їхня цінність зростає завдяки здатності зменшувати кількість помилок, підвищувати ефективність роботи та покращувати якість кінцевого

продукту. У майбутньому, зі стрімким розвитком технологій, роль UML-діаграм лише посилюватиметься. Вони будуть еволюціонувати й адаптуватися до нових підходів у проєктуванні, допомагаючи командам створювати ще складніші й надійніші програмні системи.

2.1.2 Структурні діаграми

Структурні діаграми призначені для візуалізації статичної структури системи, тобто її складових елементів та їхніх взаємозв’язків.

Існують три основних види структурних діаграм UML:

– діаграма об'єктів – це зосередження на конкретних об'єктах у системі та їхніх взаємодіях, що представляє миттєвий стан системи. Наприклад, у системі управління замовленнями інтернет-магазину на такій діаграмі можна показати конкретні об'єкти, такі як замовлення, товари, користувачі, та їхні взаємозв'язки в певний момент часу [12];

– діаграма класів – це статична структурна діаграма, що демонструє властивості системи, класи, операції та зв’язки між об’єктами для опису структури системи. Ви можете певним чином моделювати системи за допомогою Уніфікованої мови моделювання (UML). Одним із найвідоміших видів UML є діаграма класів. Він використовується серед інженерів програмного забезпечення для документування архітектури програмного забезпечення. Діаграми класів є формою структурних діаграм, оскільки вони визначають, що має бути включено до модельованої системи [14];

– діаграма компонентів – діаграми компонентів UML надають концептуальну картину взаємодії між різними системами. Можуть бути присутні як аспекти логічного, так і фізичного моделювання. Крім того, компоненти автономні. Це модульний системний елемент в UML, який можна замінити на альтернативні. Вони містять конструкції будь-якої складності та є самодостатніми. Лише через інтерфейси вкладені частини взаємодіють з іншими компонентами. Крім того, компоненти мають свої інтерфейси, але вони також можуть отримати доступ до операцій і служб інших компонентів за допомогою їхніх інтерфейсів. На діаграмі

компонентів інтерфейси також показують зв’язки та залежності в архітектурі програмного забезпечення [15].

На рисунку 2.1 зображена діаграма компонентів для платформи продажу будівельних матеріалів.

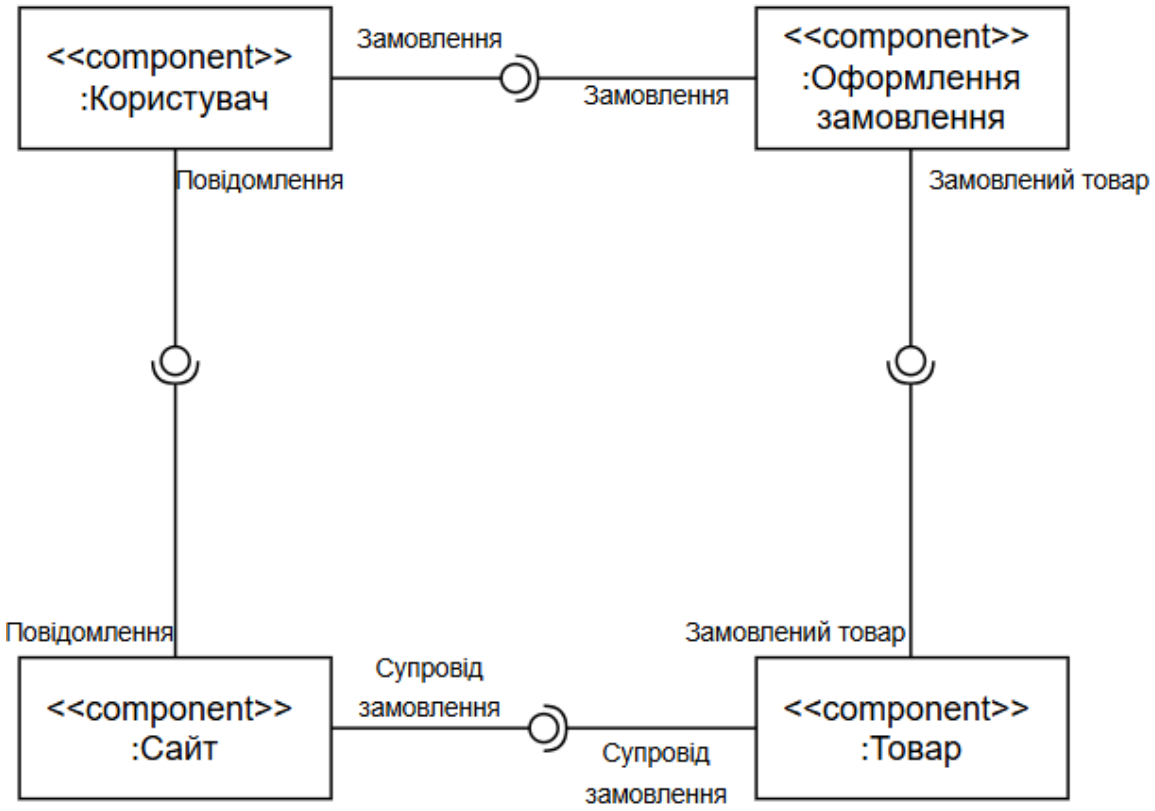


Рисунок 2.1 – UML. Діаграма компонентів

При плануванні архітектури програмного забезпечення можна використовувати ці діаграми у різних контекстах: якщо мова йде про базу даних, то діаграма класів допомагає визначити сутності та їх атрибути, а діаграма компонентів вказує на компоненти, такі як таблиці та збережені процедури.

Наприклад, при розробці мікросервісної архітектури діаграма компонентів ілюструє кожен сервіс як окремий компонент та їх взаємозв'язки.

2.1.3 Поведінкові діаграми

Поведінкові діаграми UML – це динамічні діаграми, які показують, як змінюються сутності програмного забезпечення під час виконання, як система

поводиться та взаємодіє з собою та з користувачами, іншими системами та іншими суб’єктами.

Розглянемо три основні типи поведінкових діаграм UML:

– діаграма прецендентів: Діаграма Прецедентів візуально зображає різноманітні сценарії взаємодії між акторами (користувачами) і прецедентами (випадками використання); описує функціональні аспекти системи (бізнес логіку). Діаграми прецедентів відіграють важливу роль не тільки у комунікації між збирачами вимог до проекту і потенційними користувачами. Діаграми Прецедентів дописані бізнес логікою і детальними специфікаціями прецедентів, як джерельна інформація, успішно використовують учасники розробки проекту на всіх його фазах (зародження, дизайн, програмування, тестування, документування..) [16].

На рисунку 2.2 зображена діаграма прецедентів для системи продажу будівельних матеріалів, на якій можна побачити функціональність системи для користувача, а саме – користувач може увійти в особистий кабінет, переглядати товари в каталозі, додавати необхідні матеріали в корзину, оформити замовлення та сплатити його. Адміністратор може переглядати товари, змінювати параметри карток товарів, додавати та видаляти товари, переглядати статистику продажів та є можливість увійти в особистий кабінет;

Рисунок 2.2 – UML. Діаграма прецедентів

– діаграма активностей – це блок-схема, яка показує, як одна діяльність призводить до іншої. Цю дію можна назвати системною операцією. Одна операція веде до наступної в потоці керування. Цей потік може бути паралельним, одночасним або розгалуженим. Діаграми діяльності використовують багато функцій, таких як fork, join тощо, щоб справлятися з усіма типами керування потоком. Подібно до інших діаграм, діаграми діяльності служать схожим фундаментальним цілям. Він фіксує динамічну поведінку системи [17].

Поведінкові діаграми відіграють важливу роль у розумінні процесів і взаємодії в системі:

– для проектування інтерфейсу користувача діаграма випадків використання визначає функціональні можливості та потреби користувачів, забезпечуючи створення зручного інтерфейсу;

– для оптимізації бізнес-процесів діаграма активностей допомагає візуалізувати процеси і виявити області для поліпшення, сприяючи бізнес-аналітикам у покращенні ефективності бізнесу;

– під час тестування і налагодження програмного забезпечення діаграма послідовності моделює сценарії використання системи, спрощуючи процес тестування і налагодження [18].

Діаграма послідовності – це тип UML-діаграми, який ілюструє, як саме взаємодіють об’єкти між собою у певній логічній черзі. Вона дає змогу наочно показати порядок обміну повідомленнями чи викликів між різними елементами системи під час виконання певного сценарію. Наприклад, у випадку моделювання роботи онлайн-чату така діаграма може відображати процес надсилання повідомлення користувачем, його передачу через сервер і подальшу доставку адресату, а також дії, які виконує система після отримання повідомлення.

У моєму дослідженні була розроблена діаграма послідовності для веб-застосунку продажу будівельних матеріалів, на якій можна побачити взаємодію між об'єктами в системі в певному порядку, а саме: користувач вибирає необхідний товар в каталозі, заповнює особисті дані для оформлення замовлення, цей запит відсилається на сайт, він перевіряє правильність внесення даних, замикає на себе

запит та відправляє його у базу даних, вони вносяться у БД, або знаходиться абсолютний збіг, БД відправляє сигнал на сайт, сайт відправляє повідомлення користувачу про правильність внесених даних; Користувач купує товар, цей запит відсилається на сайт, він перевіряє правильність внесення даних, замикає на себе запит та відправляє його у базу даних, БД перевіряє наявність товару, резервує його та коригує кількість товарів на складі, передає інформацію на сайт, сайт візуально змінює кількість залишків на складі та повідомляє користувача про успішне оформлення замовлення; Користувач вносить дані для авторизації або реєстрації, цей запит відсилається на сайт, він перевіряє правильність внесення даних, замикає на себе запит та відправляє його у базу даних, у БД знаходиться абсолютний збіг, або вносяться дані користувача. Передає інформацію на сайт, що в свою чергу повідомляє користувача про успішний вхід або реєстрацію; Адміністратор перевіряє статистику продажів, що відображаються в його особистому кабінеті, обираючи необхідні фільтри, цей запит відсилається на сайт, він замикає на себе запит та відправляє його у базу даних, у БД перевіряються дані та відправляється запит на сайт, дані відображаються на сайті, а адміністратор отримує дашборд продажів; Адміністратор вносить нові дані про товар, або редагує існуючі, цей запит відсилається на сайт, він замикає на себе запит та відправляє його у базу даних, у БД дані перевіряються або записуються нові, вона відправляє запит на сайт, дані відображаються на сайті, а адміністратор отримує повідомлення про успіх; Адміністратор видаляє дані про товар, цей запит відсилається на сайт, він замикає на себе запит та відправляє його у базу даних, дані видаляються з БД та відправляється запит на сайт, дані про товар видаляються зі сторінки, а адміністратор отримує повідомлення про успіх. На рисунку 2.3 зображена діаграма послідовності для веб-застосунку продажу будівельних матеріалів.

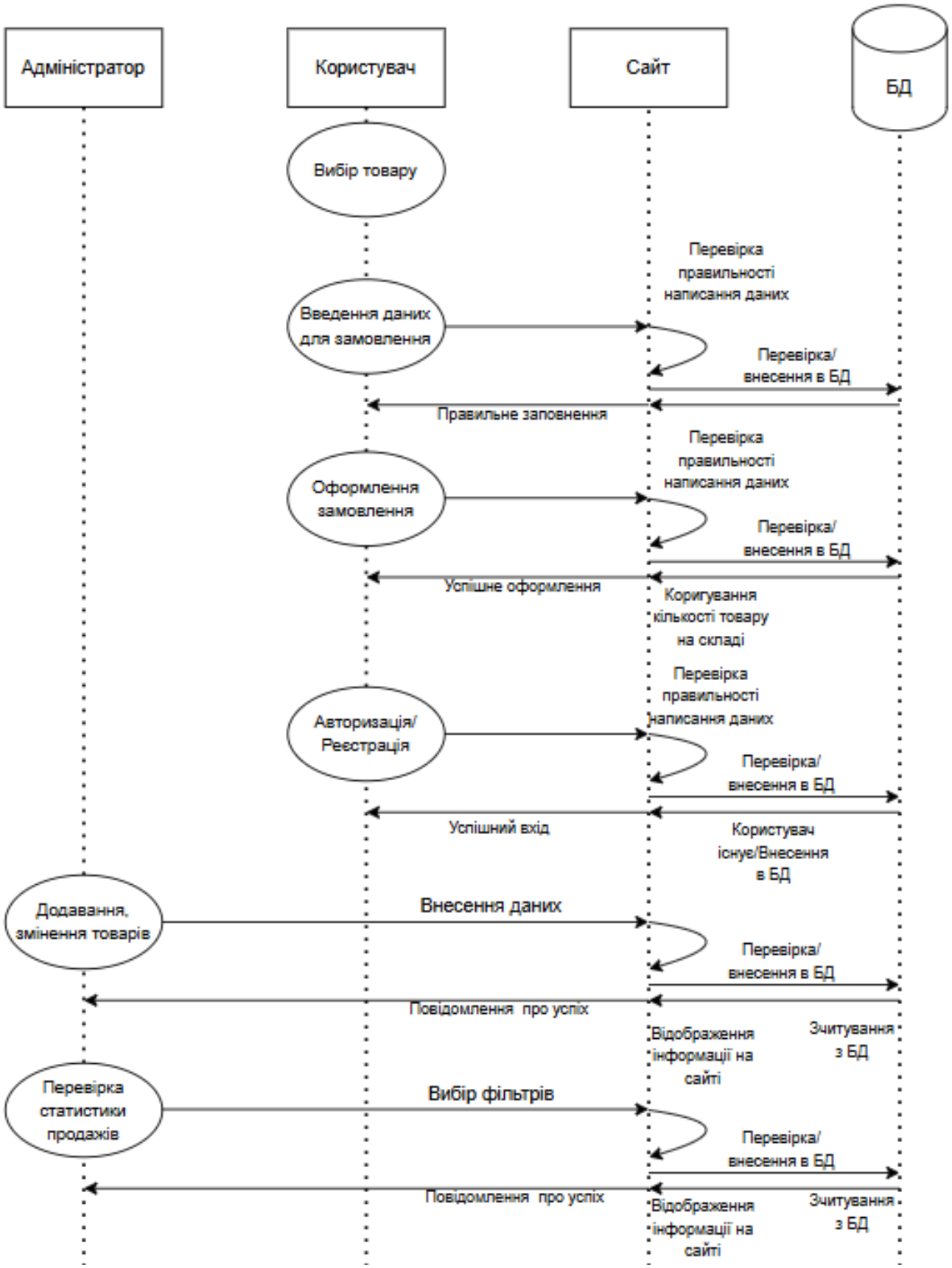


Рисунок 2.3 – UML. Діаграма послідовності

2.2 Вибір компонентів автоматизованої системи продажу готової продукції виробничого підприємства

Підбір компонентів системи диктувався потребою створити відмовостійкий інструмент для прямих продажів продукції зі складів виробництва. За основу взято класичну трирівневу архітектуру, що дозволило чітко розділити інтерфейс, бізнес-логіку та базу даних.

Клієнтська частина реалізована на стандартному стеку HTML5, CSS3, JavaScript. Це свідомий вибір: він забезпечує кросплатформність, тож менеджери можуть формувати замовлення як з офісу, так і зі смартфона прямо на об'єкті. Завдяки AJAX дані про ціни та залишки підтягуються миттєво, без оновлення сторінки.

За серверну логіку відповідає PHP. Цей модуль бере на себе всі «важкі» операції: перевіряє реальну наявність товару на складі, рахує вартість і тримає сесію користувача. Також сюди «вшито» шлюз для оплати через LiqPay та AI-модуль на базі Google Gemini, який розвантажує менеджерів, відповідаючи на питання клієнтів природною мовою.

Щодо бази даних, то було використано MySQL, але з цікавим нюансом: склад замовлення зберігається як JSON-зліпок. Це гарантує, що ціна і номенклатура в архіві точно відповідатимуть моменту покупки, що критично для бухгалтерії. Такий модульний підхід робить систему гнучкою і готовою до майбутнього підключення ERP.

2.3 Характеристика системи керування комерційною діяльністю підприємства

У контексті теорії автоматичного управління (ТАУ) розроблений програмний засіб доцільно розглядати як замкнену автоматизовану систему керування комерційними процесами підприємства. Об’єктом керування в даній моделі виступає динамічний процес реалізації будівельних матеріалів та ведення

складського обліку, тоді як функції регулятора (керуючого пристрою) покладено на програмний код серверної частини. Архітектура системи побудована на фундаментальному принципі зворотного зв'язку, що забезпечує безперервну циркуляцію інформації в замкненому контурі управління.

Робота системи ініціюється вхідним сигналом – діями користувача при оформленні замовлення через веб-інтерфейс. З точки зору ТАУ, це є збурюючим впливом, що виводить систему зі стану рівноваги. У відповідь програмний алгоритм генерує керуючу дію: відбувається обробка запиту, перевірка доступності товарних залишків у базі даних, валідація введених реквізитів та формування фінансової транзакції через платіжний шлюз LiqPay.

Ключову роль у забезпеченні стійкості та адекватності реакції системи відіграє контур зворотного зв’язку, реалізований за допомогою технології Webhook. Отримання відповіді від банківського сервера про статус платежу дозволяє системі адаптувати свою поведінку. У разі успішної транзакції спрацьовує алгоритм позитивного результату: автоматично коригуються параметри об’єкта керування (зменшується кількість товару на складі, змінюється статус замовлення). Якщо ж оплата не відбулася, система стабілізується через механізм негативного зворотного зв’язку, блокуючи подальшу обробку замовлення та повертаючи процес на попередній етап.

Варто також зазначити, що впровадження такого підходу дозволило суттєво знизити роль людського фактора, перевівши управління продажами з ручного режиму в напівавтоматичний. Додатковим елементом стабілізації виступає підсистема безпеки (сесії та розмежування прав доступу), яка діє як фільтр для відсіювання несанкціонованих впливів. Таким чином, розроблений вебзастосунок трансформує дискретні процеси продажу в цілісну адаптивну систему, здатну автономно реагувати на зовнішні запити в режимі реального часу [19].

3 РЕАЛІЗАЦІЯ АВТОМАТИЗОВАНОЇ СИСТЕМИ ПРОДАЖУ ГОТОВОЇ ПРОДУКЦІЇ ВИРОБНИЧОГО ПІДПРИЄМСТВА У ВИГЛЯДІ ПРОГРАМНОГО ЗАСОБУ

3.1 Вибір середовища розробки та мови програмування

Інтегроване середовище розробки – це програмний комплекс, призначений для створення програмного забезпечення. Воно зазвичай містить редактор вихідного коду, засоби для автоматизації збирання проєкту та налагодження програм. Більшість сучасних середовищ розробки підтримують функцію автодоповнення коду, що значно полегшує роботу розробника.

Деякі ІСР, наприклад Microsoft Visual Studio або Eclipse, мають у своєму складі компілятор чи інтерпретатор (а іноді й обидва), тоді як інші, такі як VS Code або Atom, не містять цих компонентів і використовують зовнішні інструменти для запуску програм. Часто середовища розробки включають також системи керування версіями або засоби для створення графічного інтерфейсу користувача – прикладами можуть бути XCode та PyCharm.

Багато сучасних інтегрованих середовищ оснащені інспекторами класів та об’єктів, а також відображенням ієрархії класів, що робить процес об’єктно-орієнтованої розробки більш наочним і зручним.

У моєму дослідженні було використано такі інтегровані середовища, як Visual Studio Code та PhpMyAdmin.

3.1.1 PhpMyAdmin

PHPMyAdmin – це широко використовувана веб-програма, яка надає графічний інтерфейс користувача (GUI) для керування базами даних MySQL. Він написаний на PHP і дозволяє користувачам виконувати різноманітні завдання, пов’язані з адмініструванням баз даних, наприклад створювати бази даних, таблиці та запити, а також імпортувати й експортувати дані.

За допомогою PHPMyAdmin користувачі можуть легко взаємодіяти зі своїми базами даних MySQL без необхідності писати складні команди SQL вручну. Інтерфейс забезпечує зручне середовище, де користувачі можуть переміщатися по своїх базах даних, таблицях і даних, що робить його важливим інструментом для веб-розробників і адміністраторів баз даних.

Однією з основних функцій PHPMyAdmin є можливість створювати бази даних і керувати ними. Користувачі можуть створювати нові бази даних лише кількома клацаннями, вказуючи назву та набір символів. Після створення бази даних користувачі можуть створювати таблиці в базі даних, визначати їх структуру та встановлювати зв’язки між ними. PHPMyAdmin також дозволяє користувачам змінювати існуючі таблиці, додаючи або видаляючи стовпці, змінюючи типи даних або встановлюючи первинні ключі та індекси.

Ще одна важлива функція PHPMyAdmin – це можливість виконувати SQL-запити. Користувачі можуть писати та виконувати оператори SQL безпосередньо в програмі, дозволяючи їм отримувати, змінювати та видаляти дані зі своїх баз даних. Ця функція особливо корисна для розробників, які хочуть перевірити свої запити або виконати складні операції зі своїми даними [20].

PHPMyAdmin також пропонує ряд інструментів для імпорту та експорту даних. Користувачі можуть імпортувати дані з файлів різних форматів, наприклад файлів CSV або SQL, у свої бази даних. Ця функція зручна під час міграції даних з однієї бази даних до іншої або під час заповнення нової бази даних наявними даними. З іншого боку, користувачі можуть експортувати дані зі своїх баз даних у різні формати, такі як CSV, SQL або XML, для резервного копіювання або для подальшого аналізу за допомогою інших інструментів.

Крім того, PHPMyAdmin надає різні інструменти для керування привілеями користувачів і параметрами безпеки. Користувачі можуть створювати облікові записи користувачів і керувати ними, призначати різні рівні привілеїв для кожного облікового запису та контролювати доступ до певних баз даних або таблиць. Ця функція гарантує, що лише авторизовані особи можуть отримати доступ до даних, що зберігаються в базах даних, і змінити їх.

PHPMyAdmin – це потужна веб-програма, яка спрощує керування базами даних MySQL. Він пропонує зручний інтерфейс для створення та керування базами даних, виконання SQL-запитів, імпорту та експорту даних і керування привілеями користувачів. Його повний набір функцій робить його незамінним інструментом для веб-розробників і адміністраторів баз даних [21].

Головну сторінку PhpMyAdmin можна побачити на рисунку 3.1.

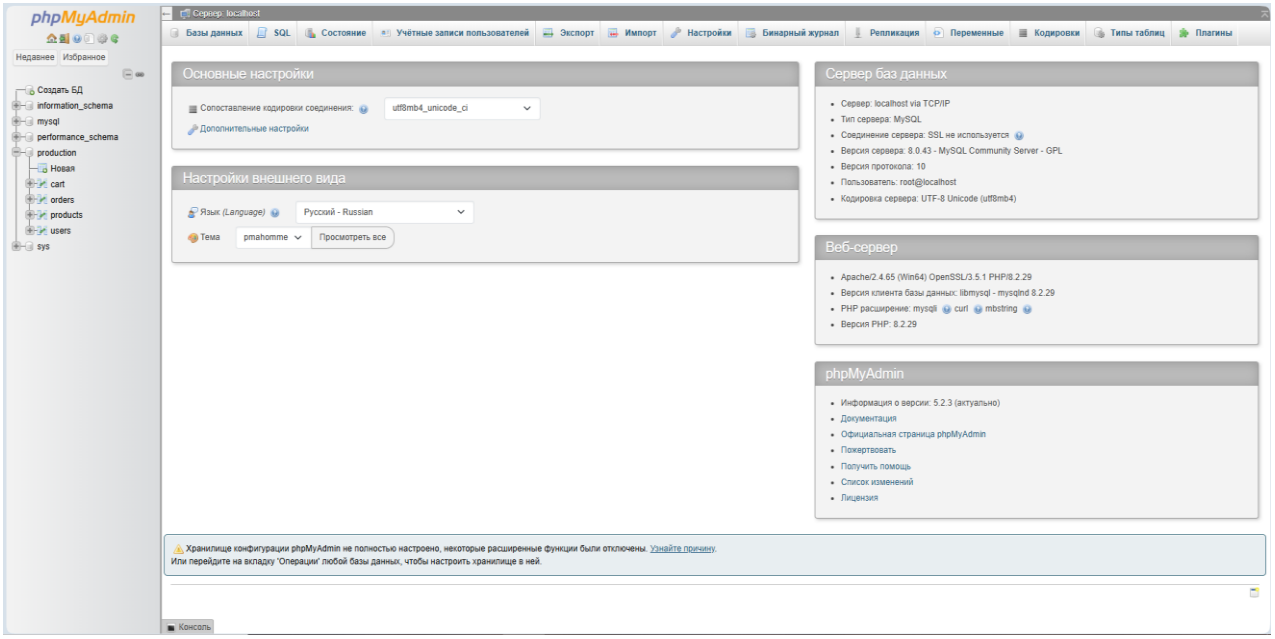


Рисунок 3.1 – Головна сторінка PhpMyAdmin

3.1.2 Visual Studio Code

Visual Studio Code (або VS Code) – це редактор вихідного коду, створений компанією Microsoft із використанням Electron Framework для операційних систем Windows, Linux та macOS. Він підтримує налагодження, підсвічування синтаксису, інтелектуальне завершення коду, рефакторинг, фрагменти коду, а також інтеграцію з Git. Користувач може налаштовувати теми, поєднання клавіш, параметри середовища та встановлювати розширення для додавання нових можливостей.

Visual Studio Code був уперше представлений 29 квітня 2015 року на конференції Build 2015, а вже 18 листопада 2015 року його вихідний код було опубліковано на GitHub під ліцензією MIT. Тоді ж з’явилася підтримка розширень. 14 квітня 2016 року завершився етап попередньої версії, після чого редактор став

офіційно доступним користувачам. Microsoft зробила більшість коду відкритою, проте офіційні збірки залишаються безкоштовним пропрієтарним продуктом.

Згідно з опитуванням розробників Stack Overflow 2022 року, у якому взяли участь понад 71 тисяча фахівців, Visual Studio Code став найпопулярнішим інструментом розробки – його використовують 74,48 % опитаних [22].

VS Code базується на структурі Electron, що дозволяє створювати вебзастосунки на Node.js із використанням рушія Blink. У редакторі застосовується компонент під назвою Monaco, який також використовується в Azure DevOps (раніше – Visual Studio Online).

Редактор підтримує численні мови програмування, серед яких C, C#, C++, Java, Python, JavaScript, Go, Rust, Fortran та інші. Базова підтримка включає підсвічування синтаксису, згортання коду, зіставлення дужок та налаштовувані фрагменти. Також доступна технологія IntelliSense для мов JavaScript, TypeScript, JSON, CSS і HTML, а підтримку інших мов можна додати через безплатні розширення з VS Code Marketplace.

На відміну від традиційних IDE, VS Code не використовує систему проєктів – користувач може відкривати один або кілька каталогів, які зберігаються як робочі області. Це робить редактор універсальним і зручним для роботи з будь-якою мовою. Також користувач може виключати непотрібні файли з дерева проєкту та отримувати доступ до прихованих функцій через палітру команд.

Середовище має вбудовану систему керування версіями, що дозволяє створювати репозиторії, виконувати операції push і pull безпосередньо з інтерфейсу редактора. VS Code підтримує роботу з Git, Apache Subversion, Perforce та іншими системами контролю версій.

Додатково VS Code може використовуватись як зручний інструмент для веброзробки завдяки численним розширенням, зокрема для FTP. Це дозволяє синхронізувати файли між локальним редактором і сервером без необхідності встановлення додаткового ПЗ.

Редактор підтримує зміну кодової сторінки, символів нового рядка та вибір мови програмування для кожного документа, що забезпечує сумісність між різними платформами та середовищами.

Visual Studio Code збирає анонімні дані про використання для покращення продукту, проте користувач може вимкнути цю функцію. Завдяки відкритому вихідному коду, телеметрія програми є прозорою – будь-хто може переглянути, які саме дані передаються.

Під час розробки вебресурсів або перенесення сайтів VS Code може використовуватись для роботи з базами даних – як через командний рядок, так і за допомогою графічних інтерфейсів, які відображають структуру даних у зручному ієрархічному вигляді [23].

3.1.3 AMPPS

AMPPS – це повноцінний стек програмного забезпечення, який включає сервер Apache, системи керування базами даних MySQL і MongoDB, а також підтримку мов програмування PHP, Perl і Python. Додатково AMPPS містить інструмент Softaculous – систему автоматичного встановлення вебзастосунків (CMS), що дозволяє за кілька кліків розгорнути такі популярні платформи, як WordPress, Joomla, Drupal, Magento тощо.

AMPPS створений для використання на настільних комп’ютерах і серверах під керуванням Windows, macOS або Linux. Його основне призначення – надати розробникам готове середовище для створення, тестування й налагодження вебсайтів або вебдодатків без необхідності підключення до віддаленого сервера [24].

На рисунку 3.2 зображено інтерфейс програми AMPPS, на якому вже працює Apache Server та MySQL Server.

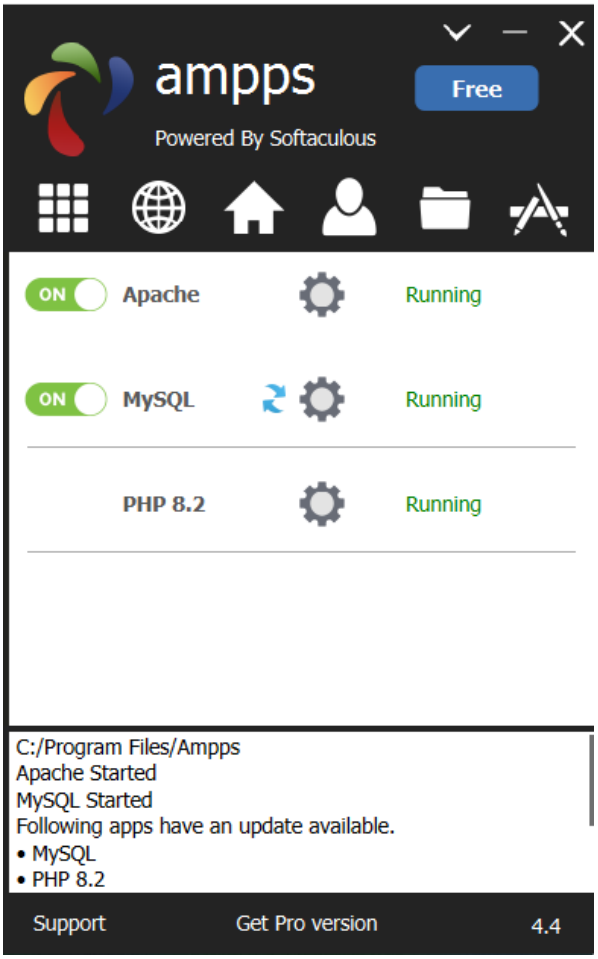


Рисунок 3.2 – Інтерфейс AMPPS

3.2 Мова програмування

Мова програмування – це створена людиною система символів і правил, призначена для спілкування з комп'ютером. Вона використовується для написання програм, які керують роботою пристроїв, а також для опису алгоритмів та структури даних. По суті, це формальна мова, за допомогою якої можна точно описати послідовність дій, що має виконати машина. Кожна мова програмування має власний набір лексичних, синтаксичних і семантичних правил, які визначають вигляд коду та поведінку програми під час її виконання.

Від часу появи перших обчислювальних машин було розроблено понад дві з половиною тисячі мов програмування, і їхня кількість продовжує зростати щороку. Деякі з них залишаються вузькоспеціалізованими та використовуються лише невеликими групами розробників, тоді як інші набувають широкої популярності у

світі. Професійні програмісти зазвичай володіють кількома мовами одночасно, обираючи кожну залежно від конкретних завдань і галузі застосування [25].

У моєму дослідженні було використано такі мови програмування як: Javascript, PHP, SQL та формальні мови декодування HTML, CSS.

3.2.1 Мова програмування JavaScript

JavaScript – динамічна, об'єктно-орієнтована прототипна мова програмування. Реалізація стандарту ECMAScript. Найчастіше використовується для створення сценаріїв вебсторінок, що надає можливість на боці клієнта (пристрої кінцевого користувача) взаємодіяти з користувачем, керувати браузером, асинхронно обмінюватися даними з сервером, змінювати структуру та зовнішній вигляд вебсторінки. JavaScript класифікують як прототипну (підмножина об'єктно-орієнтованої), скриптову мову програмування з динамічною типізацією. Окрім прототипної, JavaScript також частково підтримує інші парадигми програмування (імперативну та частково функційну) і деякі відповідні архітектурні властивості, зокрема: динамічна та слабка типізація, автоматичне керування пам'яттю, прототипне наслідування, функції як об'єкти першого класу.

JavaScript має низку властивостей об'єктно-орієнтованої мови, але завдяки концепції прототипів підтримка об'єктів в ній відрізняється від традиційних мов ООП. Крім того, JavaScript має кілька властивостей, притаманних функціональним мовам, – функції як об'єкти першого класу, об'єкти як списки, каррінг, анонімні функції, замикання – що додає мові додаткову гнучкість.

Мова JavaScript використовується для:

– написання сценаріїв вебсторінок для надання їм інтерактивності;

– створення вебзастосунків(React, AngularJS, Vue.js);

– програмування на боці сервера (Node.js (Express.js));

– стаціонарних застосунків (Electron, NW.js);

– мобільних застосунків (React Native, Cordova);

– сценаріїв в прикладних програмах (наприклад, в програмах зі складу Adobe Creative Suite чи Apache JMeter);

Попри схожість назв, мови Java та JavaScript є двома різними мовами, що мають відмінну семантику, хоча й мають схожі риси в стандартних бібліотеках та правилах іменування. Синтаксис обох мов отриманий «у спадок» від мови C, але семантика та дизайн JavaScript є результатом впливу мов Self та Scheme [26].

3.2.2 Мова програмування PHP

PHP – це серверна мова програмування з відкритим кодом, призначена для створення динамічних веб-сторінок і веб-додатків, яка інтегрується з HTML і працює на стороні сервера.

Основна концепція PHP полягає в тому, що код виконується на сервері, а результат у вигляді HTML надсилається браузеру користувача. Це дозволяє створювати динамічні сторінки, інтегровані з базами даних, API чи іншими веб-сервісами.

PHP легко інтегрується з HTML, що робить його простим у використанні навіть для початківців. Завдяки підтримці сучасних стандартів та регулярним оновленням мова зберігає свою актуальність і залишається одним із найпопулярніших інструментів для веб-розробки.

PHP залишається універсальним інструментом для веб-розробки завдяки своїй гнучкості, доступності та широкій підтримці спільноти. Серед основних сфер використання мови:

– створення динамічних сайтів (PHP є основою для багатьох сайтів, які генерують контент у реальному часі, зокрема блоги, форуми, новинні портали та інтернет-магазини);

– розробка систем управління контентом (Найпопулярніші системи управління контентом, такі як WordPress, Drupal та Joomla, створені на PHP. Вони дозволяють легко створювати, керувати та модифікувати сайти навіть без глибоких знань програмування);

– робота з базами даних (PHP чудово підходить для створення веб-додатків, які взаємодіють із базами даних, зокрема MySQL, PostgreSQL чи SQLite. Це робить

його популярним вибором для розробки CRM-систем, інвентаризаційних додатків та інших бізнес-інструментів);

– створення API та мікросервісів (Завдяки сучасним можливостям, таким як підтримка REST і GraphQL, PHP часто використовується для створення серверних API, які забезпечують обмін даними між клієнтськими додатками та серверами);

– інтеграція з іншими технологіями (PHP можна поєднувати з HTML, CSS, JavaScript та іншими мовами, створюючи інтерактивні та функціональні веб-додатки. Також він легко інтегрується із сторонніми бібліотеками чи сервісами, наприклад для платежів чи авторизації);

– аутсорсинг у веб-розробці (Завдяки своїй поширеності PHP часто використовується невеликими студіями та фрілансерами для створення бюджетних рішень, особливо для стартапів і малих підприємств).

Переваги PHP – це швидка, гнучка, з відкритим кодом мова програмування, ідеальна для створення динамічних сайтів.

Недоліки PHP – має обмеження в порівнянні з сучасними мовами програмування, такими як Python або JavaScript, і може бути менш ефективним для певних завдань, наприклад, розробки для мобільних додатків [27].

3.2.3 SQL

SQL – декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів, оновлення і керування реляційними БД, створення схеми бази даних і її модифікації, системи контролю за доступом до бази даних. Сам по собі SQL не є ні системою керування базами даних, ні окремим програмним продуктом. Не бувши мовою програмування в тому розумінні, як C або Pascal, SQL може формувати інтерактивні запити або, бувши вбудованою в прикладні програми, виступати в якості інструкцій для керування даними. Стандарт SQL, крім того, вміщує функції для визначення зміни, перевірки і захисту даних.

SQL – це діалогова мова програмування для здійснення запиту і внесення змін до бази даних, а також управління базами даних. Багато баз даних підтримує

SQL з розширеннями до стандартної мови. Ядро SQL формує командна мова, яка дозволяє здійснювати пошук, вставку, оновлення, і вилучення даних, використовуючи систему управління і адміністративні функції. SQL також включає CLI (Call Level Interface) для доступу і управління базами даних дистанційно.

Перша версія SQL була розроблена на початку 1970-х років у IBM. Ця версія мала назву SEQUEL і була призначена для обробки й пошуку даних, що містилися в реляційній базі даних IBM, System R . Мова SQL пізніше була стандартизована Американськими Держстандартами (ANSI) в 1986. Спочатку SQL розроблялась як мова запитів і управління даними, пізніші модифікації SQL створено продавцями системи управління базами даних, які додали процедурні конструкції, control-of-flowкоманд і розширення мов. З випуском стандарту SQL:1999 такі розширення були формально запозичені як частина мови SQL через Persistent Stored Modules (SQL/PSM).

SQL складається з наступних елементів мови:

– положення, які є складовими компонентами заяв і запитів;

– предикати для тризначної логіки (3VL) (так / ні / невідомо) або логічних значень, які використовуються для обмеження наслідків заяв та *запитів або змінити хід виконання програми;

– запити для видачі скалярних величин або таблиць, що складаються із стовпців і рядків даних;

– запити вилучення даних на основі певних критеріїв;

– заяви впливу на схеми і даних, зв'язків;

– заяви управлінням транзакціями, ходом виконання програми, завдань та діагностики.

За допомогою SQL програміст описує тільки те, які дані потрібно витягнути або модифікувати. Те, яким чином це зробити, вирішує СУБД безпосередньо при обробці SQL-запиту [28].

3.3 Розробка ER-моделі бази даних

ER-модель (Entity-relationship model або Entity-relationship diagram) – це семантична модель даних, яка призначена для спрощення процесу проектування бази даних. З ER-моделі можуть бути породжені всі види баз даних: реляційні, ієрархічні, мережні, об’єктні. В основі ER-моделі лежать поняття “сутність”, “зв’язок” та “атрибут”.

Для великих баз даних побудова ER-моделі дозволяє уникнути помилок проектування, які надзвичайно важко виправляти, особливо, якщо база даних вже експлуатується чи на стадії тестування. Помилки в розробці структури бази даних може призвести до перебудови коду програмного забезпечення, що керує цією базою даних. У результаті час, кошти та людські ресурси будуть використані неефективно.

ER-модель – це представлення бази даних у вигляді наочних графічних діаграм. ER-модель візуалізує процес, що визначає деяку предметну область. Діаграма “сутність-зв’язок” – це діаграма, яка представляє в графічному вигляді сутності, атрибути і зв’язки.

ER-модель – це тільки концептуальний рівень моделювання. ER-модель не містить деталей реалізації. Для тієї самої ER-моделі деталі її реалізації можуть відрізнятися [29].

У розробці автоматизованої системи для продажу готової продукції виробничого підприємства було створено ER-модель, яка використовується для реалізації бази даних. Використовуючи веб-додаток для адміністрування та управління Бази Даних PhpMyAdmin, який є простим та зручним для досягнення цієї мети. ER-модель представлено на рисунку 3.3.

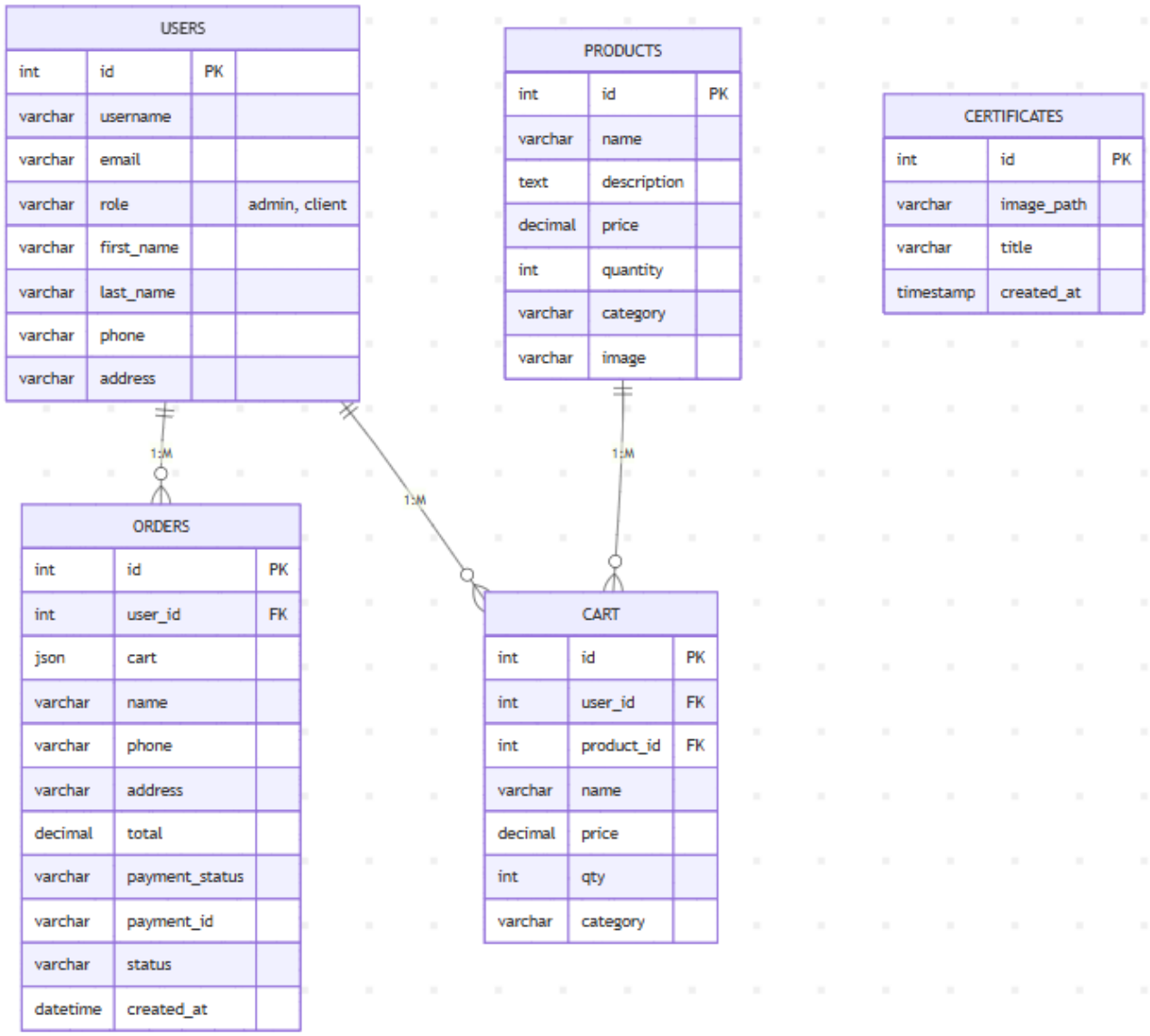


Рисунок 3.3 – ER-модель бази даних для автоматизованої системи для продажу готової продукції виробничого підприємства

Переглядаючи розроблену ER-модель, видно, що база даних складається з п’яти основних сутностей: «users» (користувачі), «products» (товари), «cart» (кошик), «orders» (замовлення) та «certificates» (сертифікати).

Сутність «users» має відношення 1:M (один-до-багатьох) із сутністю «orders». Це означає, що один зареєстрований користувач може мати історію з багатьох замовлень, але кожне конкретне замовлення належить лише одному користувачеві.

Також сутність «users» має зв’язок 1:M (один-до-багатьох) із сутністю «cart». Користувач може додати до свого кошика декілька різних позицій (товарів), кожна з яких зберігається як окремий запис у таблиці кошика.

Сутність «products» (товари) має відношення 1:M (один-до-багатьох) із сутністю «cart». Це обумовлено тим, що один і той самий товар може бути доданий до кошиків багатьох різних користувачів одночасно. При оформленні покупки дані з таблиці «cart» переносяться у поле json таблиці «orders», фіксуючи стан покупки на певний момент часу.

Сутність «certificates» є довідковою таблицею, яка не має прямих зовнішніх ключів (Foreign Keys) до процесів замовлення, але логічно пов’язана із загальним контентом магазину для підтвердження якості продукції.

Концепція розробленої моделі бази даних спрямована на оптимізацію швидкодії системи та спрощення архітектури шляхом використання гібридного підходу. Зберігання вмісту замовлення у форматі JSON («snapshot») замість створення додаткових таблиць дозволило уникнути надмірної нормалізації та зменшити кількість ресурсомістких запитів (JOIN) при обробці історії покупок. Такий підхід гарантує цілісність історичних даних про ціни та свідчить про те, що для повноцінної реалізації бізнес-процесів інтернет-магазину цілком достатньо п’яти виділених сутностей.

3.4 Розробка скриптів для роботи вебзастосунку продажу готової продукції виробничого підприємства

Під час розробки програмного забезпечення були створені скрипти для реєстрації нового користувача, для авторизації користувача та адміністратора у системі при вході в особистий кабінет, скрипт для виходу з особистого кабінету, скрипт для роботи з профілем користувача та кошиком у кабінеті користувача, скрипт для додавання нового товару, редагування та видалення товару адміністратором у каталозі, скрипт для коректної роботи каталогу, скрипт для сторінки статистики продажу, скрипт для моніторингу залишків на складі, скрипт для роботи AI консультанта, скрипт для використання платіжної системи.

3.4.1 Розробка скрипту для реєстрації користувача

Цей код написаний на PHP і реалізує реєстрацію користувача, включаючи ініціалізація сесії та підключення до бази даних, обробку даних із форми , перевірку існування користувача та реєстрація нового.

Програмний код для ініціалізація сесії та підключення до бази даних, наведений нижче.

```
// --- Налаштування сесії ---
ini_set('session.cookie_lifetime', 600); // кукі живуть 10 хвилин
ini_set('session.gc_maxlifetime', 600);  // і сама сесія також
session_save_path(__DIR__ . '/tmp_sessions');
if (!file_exists(__DIR__ . '/tmp_sessions')) mkdir(__DIR__ . '/tmp_sessions', 0777,
true);
session_start();
// Підключення до бази
$conn = new mysqli('localhost', 'root', 'mysql', 'production');
if ($conn->connect_error) {
    die("Помилка з'єднання: " . $conn->connect_error);
}
```

У цьому фрагменті здійснюється конфігурація параметрів безпеки сесії (cookie_lifetime, gc_maxlifetime), що дозволяє автоматично завершувати сеанс користувача при неактивності. Також ініціалізується об'єкт класу mysqli для взаємодії з базою даних production.

Наступним кроком є обробка даних, отриманих від користувача через форму реєстрації. Алгоритм валідації вхідних даних та створення нового запису в таблиці users, код наведений нижче.

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $username = trim($_POST['username'] ?? '');
    $email = trim($_POST['email'] ?? '');
    $password = trim($_POST['password'] ?? '');
    $confirm_password = trim($_POST['confirm_password'] ?? '');

    if ($username === '' || $email === '' || $password === '' || $confirm_password
=== '') {
        $error = "Заповніть усі поля.";
    } elseif (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
        $error = "Невірний формат email.";
    } elseif ($password !== $confirm_password) {
        $error = "Паролі не збігаються.";
    } else {
        // Перевіряємо, чи існує користувач
        $stmt = $conn->prepare("SELECT id FROM users WHERE email = ?");
        $stmt->bind_param("s", $email);
```

```
        $stmt->execute();
        $res = $stmt->get_result();

        if ($res->num_rows > 0) {
            $error = "Користувач з таким email вже існує.";
        } else {
            // Реєструємо нового користувача
            $hash = password_hash($password, PASSWORD_DEFAULT);
            $stmt = $conn->prepare("INSERT INTO users (username, email, password,
role) VALUES (?, ?, ?, 'client')");
            $stmt->bind_param("sss", $username, $email, $hash);

            if ($stmt->execute()) {
                $_SESSION['user_id'] = $conn->insert_id;
                $_SESSION['username'] = $username;
                $_SESSION['role'] = 'client';
                header("Location: client_dashboard.php");
                exit;
            } else {
                $error = "Помилка при реєстрації: " . $stmt->error;
            }
        }
    }
}
```

Система виконує перевірку на заповненість полів, валідність формату електронної пошти та співпадіння введених паролів. Для забезпечення безпеки паролі користувачів не зберігаються у відкритому вигляді, а піддаються хешуванню за допомогою функції password_hash(). Для запобігання SQL-ін’єкціям використано підготовлені запити (prepared statements). У разі успішного виконання запиту користувач автоматично авторизується та перенаправляється на сторінку client_dashboard.php.

3.4.2 Розробка скрипту для авторизації користувача

Цей PHP-скрипт реалізує повноцінну систему авторизації користувачів із підтримкою корзини покупок.

Для доступу до захищених ресурсів системи (особистого кабінету клієнта або адміністративної панелі) реалізовано механізм аутентифікації користувачів. Його програмна реалізація починається з налаштування сесії та перевірки наявного стану авторизації. Код наведений нижче.

```
// --- Налаштування сесії ---
session_save_path(__DIR__ . '/tmp_sessions');
if (!file_exists(__DIR__ . '/tmp_sessions')) mkdir(__DIR__ . '/tmp_sessions', 0777,
true);
ini_set('session.cookie_lifetime', 600);
ini_set('session.gc_maxlifetime', 600);
session_start();

// Підключення до БД
$conn = new mysqli('localhost', 'root', 'mysql', 'production');
if ($conn->connect_error) die("Помилка з'єднання: " . $conn->connect_error);
$conn->set_charset("utf8mb4");

// Якщо користувач уже увійшов
if (isset($_SESSION['user_id']) && isset($_SESSION['role'])) {
    if ($_SESSION['role'] === 'admin') {
        header("Location: admin_dashboard.php");
    } else {
        header("Location: client_dashboard.php");
    }
    exit;
}
```

Алгоритм, який запобігає повторному входу в систему. Якщо в сесії вже існують змінні user_id та role, скрипт автоматично перенаправляє користувача на відповідну сторінку згідно з його роллю, минаючи форму входу.

Основна логіка аутентифікації виконується після відправки форми методом POST. Система здійснює пошук користувача в базі даних та перевірку валідності введеного пароля. Код наведений нижче.

```
$error = '';

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $email = trim($_POST['email'] ?? '');
    $password = trim($_POST['password'] ?? '');

    if ($email === '' || $password === '') {
        $error = "Введіть email і пароль.";
    } else {
        // Пошук користувача за email
        $stmt = $conn->prepare("SELECT id, username, password, role FROM users WHERE
email = ?");
        $stmt->bind_param("s", $email);
        $stmt->execute();
        $res = $stmt->get_result();

        if ($res->num_rows === 1) {
            $user = $res->fetch_assoc();
```

```
            // Верифікація хешу пароля
            if (password_verify($password, $user['password'])) {

                // Збереження тимчасового кошика
                $anon_cart = $_SESSION['cart'] ?? [];

                // Регенерація сесії для безпеки
                session_unset();
                $_SESSION['user_id'] = $user['id'];
                $_SESSION['username'] = $user['username'];
                $_SESSION['role'] = strtolower(trim($user['role']));
                $_SESSION['login_time'] = time();

                // Маршрутизація користувача
                if ($_SESSION['role'] === 'admin') {
                    header("Location: admin_dashboard.php");
                } else {
                    header("Location: client_dashboard.php");
                }
                exit;
            } else {
                $error = "Невірний пароль.";
            }
        } else {
            $error = "Користувача не знайдено.";
        }
    }
}
```

Ключовою особливістю реалізованого механізму є функція збереження товарів, доданих до кошика до моменту входу в систему ($anon_cart). Після успішної авторизації ці товари автоматично переносяться до постійного кошика користувача в базі даних, що забезпечує безшовний досвід використання сервісу (User Experience). Безпека паролів гарантується використанням стандартної функції PHP password_verify()

3.4.3 Розробка скрипту для виходу з особистого кабінету

Завершення роботи користувача з системою (вихід з акаунта) потребує повного знищення даних поточної сесії на стороні сервера та видалення ідентифікатора сесії з браузера клієнта. Код наведений нижче.

```
session_start();

// Якщо користувач був авторизований, сесія просто завершується.
if (isset($_SESSION['user_id'])) {
```

```
}

// 1. Очищення масиву суперглобальної змінної $_SESSION
$_SESSION = [];

// 2. Видалення cookie сесії з браузера користувача
if (ini_get("session.use_cookies")) {
    $params = session_get_cookie_params();
    setcookie(session_name(), '', time() - 42000,
        $params["path"], $params["domain"],
        $params["secure"], $params["httponly"]
    );
}

// 3. Остаточне знищення сесії на сервері
session_destroy();
```

Продемонстровано трирівневу очистку даних:

– обнулення масиву $_SESSION, що видаляє всі змінні в оперативній пам'яті скрипта;

– примусове видалення cookie з ідентифікатором сесії (PHPSESSID) шляхом встановлення часу її життя у минуле (time() - 42000). Це гарантує, що зловмисник не зможе використати старий ідентифікатор сесії;

– виклик функції session_destroy(), яка фізично видаляє файл сесії на сервері.

3.4.4 Розробка скрипту для роботи з профілем користувача та кошиком

Даний PHP-код виконує основні операції після авторизації користувача: ініціалізацію сесії, підключення до бази даних, завантаження профілю, формування кошика з бази даних і обробку AJAX-запитів для оновлення даних користувача чи оформлення замовлення.

Програмний код для ініціалізації сесії та підключення до бази даних, поданий нижче.

```
session_save_path(__DIR__ . '/tmp_sessions');
ini_set('session.cookie_lifetime', 600);
ini_set('session.gc_maxlifetime', 600);
session_start();

$conn = new mysqli("localhost", "root", "mysql", "production");
if ($conn->connect_error) die("DB connection failed");

if (!isset($_SESSION['user_id'])) {
```

```
    header('Location: login.php');
    exit;
}
```

У цьому фрагменті відбувається створення тимчасової директорії для збереження сесій, встановлення часу їхнього життя (10 хвилин) і підключення до бази даних. Якщо користувач не авторизований – його перенаправляють на сторінку входу.

Функціонування особистого кабінету користувача базується на взаємодії клієнтської частини з сервером через механізм асинхронних запитів (AJAX). При завантаженні сторінки система ініціалізує дані користувача та стан його кошика. Код ініціалізації наведено нижче.

```
$user_id = $_SESSION['user_id'];
$username = $_SESSION['username'] ?? '';

// Отримання актуальних даних профілю з БД
$stmt = $conn->prepare("SELECT first_name, last_name, email, phone, address FROM
users WHERE id=?");
$stmt->bind_param("i", $user_id);
$stmt->execute();
$userData = $stmt->get_result()->fetch_assoc();

// Синхронізація кошика: якщо сесія порожня, завантажуємо збережений кошик з БД
if (!isset($_SESSION['cart']) || empty($_SESSION['cart'])) {
    $_SESSION['cart'] = [];
    $res = $conn->query("SELECT * FROM cart WHERE user_id = $user_id");
    while ($row = $res->fetch_assoc()) {
        $_SESSION['cart'][$row['product_id']] = [
            'name' => $row['name'],
            'price' => $row['price'],
            'qty' => $row['qty'],
            'category' => $row['category'] ?? '',
            'description' => $row['description'] ?? ''
        ];
    }
}
// Розрахунок загальної суми для відображення
$cart_total = array_sum(array_map(fn($i) => $i['price'] * $i['qty'],
$_SESSION['cart']));
```

Наведено механізм відновлення стану кошика. Це забезпечує збереження вибраних товарів навіть після закриття браузера, оскільки дані дублюються в базі даних.

Програмний код для обробки AJAX-запитів: оновлення кошика, поданий нижче.

```
// Підключення конфігураційного файлу з ключами доступу
if (file_exists('config_pay.php')) {
    require_once 'config_pay.php';
}

if ($action === 'checkout_submit') {
    // Підготовка даних кошика у форматі JSON
    $cart_data = json_encode($_SESSION['cart'], JSON_UNESCAPED_UNICODE);

    // Розрахунок підсумкової суми
    $total = 0;
    foreach ($_SESSION['cart'] as $item) {
        $total += $item['price'] * $item['qty'];
    }

    // Збереження замовлення зі статусом оплати 'pending' (очікує)
    $stmt = $conn->prepare("INSERT INTO orders (user_id, cart, total, payment_status,
...) VALUES (?, ?, ?, 'pending', ...)");
    // ... (прив'язка параметрів) ...
    $stmt->execute();
    $order_id = $stmt->insert_id;

    // Формування масиву параметрів згідно з протоколом LiqPay API
    $liqpay_params = [
        'action'      => 'pay',
        'amount'      => $total,
        'currency'    => 'UAH',
        'description' => "Оплата замовлення #$order_id",
        'order_id'    => $order_id,
        'version'     => '3',
        'public_key'  => LIQPAY_PUBLIC_KEY,
        'server_url'  => 'http://your-site.com/webhook.php', // Callback-адреса
        'result_url'  => 'http://your-site.com/client_dashboard.php'
    ];
    // Кодування даних та генерація підпису (Signature)
    $data = base64_encode(json_encode($liqpay_params));
    $signature = base64_encode(sha1(LIQPAY_PRIVATE_KEY . $data . LIQPAY_PRIVATE_KEY,
1));

    // Повернення JSON-відповіді клієнту
    echo json_encode([
        'success' => true,
        'pay_data' => $data,
        'pay_signature' => $signature
    ]);
    exit;
}
```

Система формує об'єкт даних $liqpay_params, який містить суму, валюту та ID замовлення. Критично важливим етапом є створення цифрового підпису ($signature) за допомогою алгоритму SHA-1, що гарантує цілісність даних при передачі до платіжної системи.

Для забезпечення безшовної взаємодії з користувачем використовується JavaScript, який обробляє відповідь сервера та автоматично формує запит до банку. Реалізацію клієнтського сценарію наведено нижче.

```
document.getElementById('submitOrder').addEventListener('click', () => {
    // ... (Валідація полів форми) ...

    // Асинхронна відправка даних на сервер
    fetch('', {
        method: 'POST',
        headers: {'Content-Type': 'application/x-www-form-urlencoded'},
        body: `action=checkout_submit&...` // Параметри форми
    }).then(r => r.json()).then(data => {

        if (data.success) {
            // Динамічне створення форми для відправки на LiqPay
            const form = document.createElement('form');
            form.method = 'POST';
            form.action = 'https://www.liqpay.ua/api/3/checkout';
            form.target = '_blank'; // Відкриття у новій вкладці

            // Додавання прихованих полів з даними та підписом
            const inputData = document.createElement('input');
            inputData.type = 'hidden';
            inputData.name = 'data';
            inputData.value = data.pay_data;

            const inputSignature = document.createElement('input');
            inputSignature.type = 'hidden';
            inputSignature.name = 'signature';
            inputSignature.value = data.pay_signature;

            form.appendChild(inputData);
            form.appendChild(inputSignature);
            document.body.appendChild(form);

            // Запит підтвердження та відправка форми
            if(confirm('🎉 Замовлення створено! Перейти до оплати зараз?')) {
                form.submit();
            }
        }
    });
});
```

Продемонстровано динамічне створення HTML-форми засобами DOM API. Це дозволяє уникнути перезавантаження сторінки та миттєво перенаправити користувача на захищений платіжний шлюз LiqPay, використовуючи дані, отримані від сервера у попередньому кроці.

3.4.5 Розробка скрипту для відображення товарів і роботи з кошиком

Даний PHP-скрипт забезпечує виведення товарів, роботу користувацького кошика та обробку AJAX-запитів для додавання або видалення товарів.

Програмний код для ініціалізації сесії та підключення до бази даних, поданий нижче.

```php
session_save_path(__DIR__ . '/tmp_sessions');
ini_set('session.cookie_lifetime', 600);
ini_set('session.gc_maxlifetime', 600);
session_start();

$conn = new mysqli("localhost", "root", "mysql", "production");
if ($conn->connect_error) die("DB connection failed");

if (!isset($_SESSION['user_id'])) {
    header('Location: login.php');
    exit;
}
```

У цьому фрагменті виконується налаштування параметрів сесії користувача створюється спеціальна папка для збереження сесійних файлів, встановлюється час життя сесії (10 хвилин) та ініціалізується PHP-сесія.

Сторінка каталогу товарів є ключовим елементом інтерфейсу користувача, що забезпечує пошук, перегляд та вибір продукції. Програмна логіка сторінки починається з ініціалізації сесії, підключення до бази даних та обробки вхідних параметрів для фільтрації товарів. Фрагмент коду, що реалізує цей функціонал, наведено нижче.

```php
// Отримання параметрів фільтрації з URL
$category_filter = $_GET['category'] ?? '';
$search_query = $_GET['search'] ?? '';
$product_id = $_GET['id'] ?? null;

$conditions = [];
$is_single_product = false;
```

```
// Формування умов SQL-запиту
if ($product_id) {
    // Режим перегляду одного товару
    $conditions[] = "id=" . intval($product_id);
    $is_single_product = true;
} else {
    // Режим списку товарів
    if ($category_filter) {
        $conditions[] = "category='" . $conn->real_escape_string($category_filter) .
"'";
    }
    if ($search_query) {
        $safe_search = $conn->real_escape_string($search_query);
        // Пошук за назвою або описом
        $conditions[] = "(name LIKE '%$safe_search%' OR description LIKE
'%$safe_search%')";
    }
}

// Виконання запиту до БД
$sql = "SELECT *, IFNULL(full_description, description) as full_description FROM
products";
if (count($conditions) > 0) {
    $sql .= " WHERE " . implode(' AND ', $conditions);
}
```

Продемонстровано динамічне формування SQL-запиту залежно від дій користувача. Використання функції real_escape_string забезпечує базовий захист від SQL-ін’єкцій при обробці пошукових запитів.

Для забезпечення інтерактивності інтерфейсу без перезавантаження сторінки реалізовано механізм додавання товарів до кошика через AJAX-запити. Серверна частина обробки цих запитів наведена нижче.

```
if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_POST['action'])) {
    header('Content-Type: application/json'); // Встановлення заголовка JSON
    $action = $_POST['action'];
    $id = intval($_POST['product_id']);

    if (!isset($_SESSION['cart'])) $_SESSION['cart'] = [];

    // Додавання товару
    if ($action === 'add') {
        $name = $_POST['product_name'];
        $price = $_POST['product_price'];
        $qtyToAdd = isset($_POST['qty']) ? intval($_POST['qty']) : 1;

        // Оновлення сесії
        if (isset($_SESSION['cart'][$id])) {
```

```
            $_SESSION['cart'][$id]['qty'] += $qtyToAdd;
        } else {
            $_SESSION['cart'][$id] = ['name' => $name, 'price' => $price, 'qty' =>
$qtyToAdd];
        }

        // Синхронізація з БД для авторизованих користувачів
        if (isset($_SESSION['user_id'])) {
            $user_id = intval($_SESSION['user_id']);
            // ... (SQL запити на INSERT/UPDATE таблиці cart) ...
        }
    }
    // Видалення товару
    elseif ($action === 'remove') {
        if (isset($_SESSION['cart'][$id])) unset($_SESSION['cart'][$id]);
        // ... (SQL запит на DELETE з таблиці cart) ...
    }

    // Повернення актуального стану кошика
    $cart_count = array_sum(array_column($_SESSION['cart'], 'qty'));
    $cart_total = array_sum(array_map(fn($i) => $i['price'] * $i['qty'],
$_SESSION['cart']));
    echo json_encode(['count' => $cart_count, 'total' => $cart_total, 'cart' =>
$_SESSION['cart']]);
    exit;
}
```

Алгоритм дозволяє синхронізувати стан кошика між сесією браузера та базою даних, що забезпечує збереження вибору користувача навіть при зміні пристрою (за умови авторизації).

Клієнтська частина реалізована за допомогою мови JavaScript і використовує Fetch API для асинхронної взаємодії з сервером. Код обробника події додавання товару до кошика представлено нижче.

```
document.querySelectorAll('.add-btn').forEach(btn => {
    btn.addEventListener('click', () => {
        // Отримання даних товару з атрибутів data-*
        const id = btn.dataset.id;
        const name = btn.dataset.name;
        const price = btn.dataset.price;
        const qtyInput = document.getElementById('qty-' + id);
        const quantity = qtyInput ? parseInt(qtyInput.value) : 1;

        // Відправка запиту на сервер
        fetch('', {
            method: 'POST',
            headers: {
```

```
                'Content-Type': 'application/x-www-form-urlencoded',
                'X-Requested-With': 'XMLHttpRequest'
            },
            body:
`action=add&product_id=${id}&product_name=${encodeURIComponent(name)}&product_price=$
{price}&qty=${quantity}`
        }).then(res => res.json()).then(data => {
            // Оновлення лічильника кошика
            document.getElementById('cartCount').textContent = data.count;
            updateCartTable(data.cart, data.total);

            // Візуальне підтвердження дії
            btn.textContent = "Додано! ✓";
            btn.style.background = "#10b981";
            setTimeout(() => {
                btn.textContent = "У кошик";
                btn.style.background = "";
            }, 1000);
        });
    });
});
```

Показано реалізацію інтерактивного елементу інтерфейсу, який забезпечує миттєвий зворотний зв'язок для користувача (зміна тексту та кольору кнопки) без перезавантаження сторінки, що значно покращує UX (User Experience).

Важливою функціональною особливістю системи є автоматичне визначення популярних товарів на основі історії продажів. Алгоритм аналізу замовлень та формування рейтингу «ТОП-5». Програмний код наведено нижче.

```
// Ініціалізація масиву продажів
$sales_data = [];

// 1. Отримання історії всіх замовлень
$orders_res = $conn->query("SELECT cart FROM orders");

if ($orders_res) {
    while ($order = $orders_res->fetch_assoc()) {
        $cart_items = json_decode($order['cart'], true);

        // 2. Агрегація кількості проданих одиниць по кожному товару
        if (is_array($cart_items)) {
            foreach ($cart_items as $item) {
                $p_name = $item['name'] ?? '';
                $p_qty = intval($item['qty'] ?? 0);

                if (!isset($sales_data[$p_name])) {
                    $sales_data[$p_name] = 0;
                }
```

```
                $sales_data[$p_name] += $p_qty;
            }
        }
    }
}

// 3. Сортування масиву від найбільшого до найменшого
arsort($sales_data);

// 4. Виділення топ-5 найменувань
$top_5_names = array_slice(array_keys($sales_data), 0, 5);
?>
```

Реалізовано аналітичний модуль, який сканує JSON-дані в таблиці замовлень, підсумовує продажі для кожної товарної позиції та формує масив ідентифікаторів найпопулярніших товарів. Це дозволяє динамічно маркувати продукцію бейджем «ТОП Продажів» у каталозі.

Окрім списку товарів, система підтримує режим детального перегляду окремої позиції. Логіка перемикання режимів відображення (список/картка) реалізована через перевірку вхідних GET-параметрів. Програмний код наведено нижче.

```
// Перевірка режиму відображення: один товар чи каталог
if ($is_single_product && $single_product) {
    // РЕЖИМ ДЕТАЛЬНОЇ КАРТКИ ТОВАРУ
    // Відображення зображення, ціни та кнопок управління
    // ...

    // Блок вкладок (Tabs) для опису та характеристик
    echo '<div class="product-details-full">';
    echo '<div class="tabs">';
    echo '<div class="tab-button active" onclick="openTab(...)">Опис</div>';
    echo '</div>';

    // Вивід форматованого опису
    echo '<div id="descriptionTab" class="tab-content">';
    echo nl2br(htmlspecialchars($single_product['description']));
    echo '</div>';
    echo '</div>';

} else {
    // РЕЖИМ КАТАЛОГУ (Сітка товарів)
    echo '<div class="products-grid">';

    if (!empty($products)) {
        foreach($products as $p) {
            // Перевірка на входження в ТОП-5 для відображення бейджа
```

```
            if (!empty($p['is_top'])) {
                echo '<div class="top-badge">🔥 ТОП Продажів</div>';
            }
            // ... (Вивід картки товару) ...
        }
    }
    echo '</div>';
}
```

Реалізація поліморфного інтерфейсу: залежно від наявності параметра id в URL-адресі, система генерує або таблицю товарів з фільтрами, або сторінку конкретного продукту з розширеним описом та характеристиками.

Також, для підвищення якості обслуговування, у систему інтегровано модуль інтелектуального помічника (AI-чат). Клієнтська частина реалізації чат-бота наведена нижче.

```
async function sendMessage() {
    const text = aiInput.value.trim();
    if (!text) return;

    // Додавання повідомлення користувача в інтерфейс
    addMessage(text, 'user');
    aiInput.disabled = true;

    // Індикатор завантаження
    const loadingId = addMessage('Друкую...', 'bot', true);

    try {
        // Відправка запиту до API штучного інтелекту
        const response = await fetch('chat_api.php', {
            method: 'POST',
            headers: { 'Content-Type': 'application/json' },
            body: JSON.stringify({ message: text })
        });

        const data = await response.json();

        // Видалення індикатора та показ відповіді
        document.getElementById(loadingId).remove();
        addMessage(data.reply, 'bot');
    } catch (error) {
        document.getElementById(loadingId).remove();
        addMessage('Сервіс тимчасово недоступний.', 'bot');
    }
}
```

Наведено функцію sendMessage, яка забезпечує асинхронний обмін даними з мікросервісом chat_api.php. Це дозволяє користувачам отримувати миттєві консультації щодо товарів без участі оператора.

3.4.6 Розробка скрипту для реалізації адміністративної частини системи

Цей скрипт реалізує панель адміністратора для керування товарами – додавання, редагування та видалення.

Система перевіряє роль користувача в сесії, і якщо вона відрізняється від admin, виконується примусове перенаправлення на сторінку авторизації.

Програмний код для ініціалізації сесії адміністратор, наведено нижче.

```
session_save_path(__DIR__ . '/tmp_sessions');
if (!file_exists(__DIR__ . '/tmp_sessions')) mkdir(__DIR__ . '/tmp_sessions', 0777,
true);
session_start();

// Перевірка прав доступу (Role-Based Access Control)
if (!isset($_SESSION['user_id']) || $_SESSION['role'] !== 'admin') {
    header("Location: login.php");
    exit;
}

// Підключення до бази даних
$conn = new mysqli("localhost", "root", "mysql", "production");
if ($conn->connect_error) die("Помилка з'єднання: " . $conn->connect_error);
$conn->set_charset("utf8mb4");
```

Реалізовано механізм безпеки. Якщо у глобальному масиві $_SESSION відсутній ключ role або його значення не дорівнює admin, скрипт негайно припиняє виконання та перенаправляє користувача на сторінку авторизації. Це унеможливлює несанкціоноване управління каталогом.

Для відображення товарів у таблиці реалізовано функцію, яка виконує вибірку даних з можливістю пошуку, скрипт поданий нижче.

```
function getAllProducts($conn, $query = "") {
    $products = [];
    $search_condition = "";

    // Якщо є пошуковий запит, формуємо умову WHERE
    if (!empty($query)) {
```

```
        $safe_query = $conn->real_escape_string($query);
        $search_condition = " WHERE name LIKE '%$safe_query%'
                             OR category LIKE '%$safe_query%'
                             OR description LIKE '%$safe_query%'";
    }

    // Виконання запиту до БД
    $result = $conn->query("SELECT * FROM products" . $search_condition . " ORDER BY
id ASC");

    if ($result) {
        while ($row = $result->fetch_assoc()) {
            $products[] = $row;
        }
    }
    return $products;
}
```

Функція getAllProducts є універсальною: вона може повертати як повний список товарів, так і відфільтрований за ключовим словом. Використання методу real_escape_string захищає від SQL-ін'єкцій при пошуку.

Додавання нових товарів включає роботу з файловою системою для збереження зображень. Код обробки POST-запиту на додавання наведено нижче.

```
if ($action === 'add') {
    $imagePath = '';
    // Перевірка наявності завантаженого файлу
    if (isset($_FILES['image']) && !empty($_FILES['image']['name'])) {
        $targetDir = "uploads/";
        if (!file_exists($targetDir)) mkdir($targetDir, 0777, true);
        $imagePath = $targetDir . basename($_FILES["image"]["name"]);
        // Переміщення файлу з тимчасової папки
        move_uploaded_file($_FILES["image"]["tmp_name"], $imagePath);
    }

    // Вставка даних у таблицю products
    $stmt = $conn->prepare("INSERT INTO products (name, category, description,
full_description, price, quantity, image) VALUES (?, ?, ?, ?, ?, ?, ?)");
    $stmt->bind_param("ssssdis", $name, $category, $desc, $full_desc, $price,
$quantity, $imagePath);

    if (!$stmt->execute()) echo json_encode(["status"=>"error", "error"=>$stmt-
>error]);
    else echo json_encode(["status" => "success"]);
    exit;
}
```

Продемонстровано обробку мультимедійних даних. Зображення зберігається в директорії uploads/, а шлях до нього записується в базу даних. Використання підготовлених запитів (prepare/bind_param) гарантує безпеку вставки даних.

Редагування товару вимагає складнішої логіки, оскільки користувач може оновити лише текстові поля, не змінюючи фотографію, скрипт поданий нижче.

```
if ($action === 'edit' && $id > 0) {
    // Якщо завантажено нове фото — оновлюємо шлях
    if ($newImageUploaded) {
        $stmt = $conn->prepare("UPDATE products SET name=?, category=?,
description=?, full_description=?, price=?, quantity=?, image=? WHERE id=?");
        $stmt->bind_param("ssssdisi", $name, $category, $desc, $full_desc, $price,
$quantity, $imagePath, $id);
    } else {
        // Якщо фото не змінювалось — оновлюємо тільки текст
        $stmt = $conn->prepare("UPDATE products SET name=?, category=?,
description=?, full_description=?, price=?, quantity=? WHERE id=?");
        $stmt->bind_param("ssssdii", $name, $category, $desc, $full_desc, $price,
$quantity, $id);
    }

    if (!$stmt->execute()) echo json_encode(["status"=>"error", "error"=>$stmt-
>error]);
    else echo json_encode(["status" => "updated"]);
    exit;
}
```

Реалізовано розгалуження логіки оновлення (UPDATE). Це дозволяє уникнути перезапису поля image порожнім значенням, якщо адміністратор вирішив змінити лише ціну або опис.

Інтерфейс побудовано за принципом SPA (Single Page Application). Оновлення таблиці товарів відбувається динамічно без перезавантаження сторінки за допомогою технології AJAX, скрипт поданий нижче.

```
// Функція отримання даних з сервера
async function loadProducts(query=""){
    try {
        const res = await fetch('admin_dashboard.php?q=' +
encodeURIComponent(query));
        const productsData = await res.json();
        renderTable(productsData);
    } catch (e) { console.error(e); }
}

// Функція побудови HTML-коду таблиці
function renderTable(products) {
```

```
    const tbody = document.querySelector("#productTable tbody");
    tbody.innerHTML = "";

    products.forEach(p => {
        // Захист від XSS-атак шляхом екранування символів
        let safeName = p.name.replace(/</g, "&lt;");

        tbody.innerHTML += `
        <tr>
            <td>#${p.id}</td>
            <td>${p.image ? `<img src="${p.image}" class="prod-img">` : ''}</td>
            <td>${safeName}</td>
            <td>${p.price} ₴</td>
            <td>
                <button class="btn btn-sm" onclick="openEdit(${p.id})">Ред.</button>
                <button class="btn btn-sm btn-danger"
onclick="deleteProduct(${p.id})">Вид.</button>
            </td>
        </tr>`;
    });
}
```

Використання Fetch API для отримання JSON-даних. Функція renderTable динамічно формує рядки таблиці, підставляючи дані про товари.

Для відправки форм (додавання та редагування) використовується об'єкт FormData, що дозволяє передавати файли асинхронно, скрипт поданий нижче.

```
document.getElementById('addForm').onsubmit = async e => {
    e.preventDefault(); // Заборона стандартної відправки форми
    const formData = new FormData(e.target);

    const res = await fetch('', {method:'POST', body: formData});
    const data = await res.json();

    if(data.status === 'success'){
        alert("✅ Товар додано!");
        e.target.reset();
        // Оновлення таблиці без перезавантаження сторінки
        loadProducts(document.getElementById('searchInput').value);
    } else {
        alert("❌ Помилка: " + data.error);
    }
};
```

Показано перехоплення події onsubmit. Це дозволяє відправити дані на сервер у фоновому режимі, отримати відповідь і миттєво оновити список товарів, що значно покращує досвід користувача (User Experience). Для редагування

використовується модальне вікно, яке заповнюється даними обраного товару з локальної пам'яті браузера, скрипт поданий нижче.

```
function openEdit(id) {
    // Пошук товару в масиві за ID
    const product = productsData.find(p => p.id == id);
    if (!product) return;

    // Відкриття модального вікна
    document.getElementById('editModal').style.display = 'flex';

    // Заповнення полів форми
    document.getElementById('editId').value = product.id;
    document.getElementById('editName').value = product.name;
    document.getElementById('editCategory').value = product.category;
    document.getElementById('editPrice').value = product.price;
}
```

Наведено реалізацію клієнтської функції openEdit, яка відповідає за підготовку графічного інтерфейсу до процесу редагування даних. Ця функція викликається при натисканні кнопки «Редагувати» у рядку таблиці товарів. Видалення записів є критичною операцією, тому перед виконанням запиту реалізовано підтвердження дії, скрипт поданий нижче.

```
async function deleteProduct(id) {
    if (!confirm("Видалити цей товар остаточно?")) return;

    const formData = new FormData();
    formData.append('action', 'delete');
    formData.append('id', id);

    const res = await fetch('', {method: 'POST', body: formData});
    const data = await res.json();

    if (data.status === 'deleted') {
        loadProducts(document.getElementById('searchInput').value);
    }
}
```

Показано асинхронне видалення запису. Після успішного виконання операції на сервері список товарів автоматично оновлюється, відображаючи актуальний стан бази даних.

3.4.7 Розробка скрипту для реалізації модуля аналітики та звітності

Для забезпечення ефективного управління продажами розроблено модуль аналітики (sales_dashboard.php). Він дозволяє адміністратору моніторити ключові показники ефективності (KPI), аналізувати динаміку замовлень та переглядати детальну історію транзакцій.

Система перевіряє роль користувача в сесії, і якщо вона відрізняється від admin, виконується примусове перенаправлення на сторінку авторизації. Програмний код для ініціалізації сесії адміністратора. Програмний код наведено нижче.

```
session_save_path(__DIR__ . '/tmp_sessions');
if (!file_exists(__DIR__ . '/tmp_sessions')) mkdir(__DIR__ . '/tmp_sessions', 0777,
true);
session_start();

// Перевірка авторизації (Role-Based Access Control)
if (!isset($_SESSION['user_id']) || $_SESSION['role'] !== 'admin') {
    header("Location: login.php");
    exit;
}

// Підключення до бази даних
$conn = new mysqli("localhost", "root", "mysql", "production");
if ($conn->connect_error) die("Помилка з'єднання: " . $conn->connect_error);
$conn->set_charset("utf8mb4");
```

Наведено стандартний блок ініціалізації: старт сесії, перевірка ролі та підключення до БД з кодуванням utf8mb4 для коректної роботи з кирилицею.

Основним завданням серверної частини є збір статистики та вибірка замовлень. Реалізовано гнучкий пошук, який дозволяє знаходити замовлення за різними критеріями (ID, ім'я клієнта, телефон), скрипт наведений нижче.

```
// Ініціалізація змінних статистики
$stats = [
    'total_sales' => 0,
    'orders_count' => 0,
    'products_sold' => [],
];

// Обробка пошукового запиту
$search = isset($_GET['search']) ? trim($_GET['search']) : '';
$where_clause = "";
```

```
if ($search) {
    $s = $conn->real_escape_string($search);
    // Пошук за ID, даними зареєстрованого користувача або гостя
    $where_clause = "
        WHERE o.id LIKE '%$s%'
        OR u.first_name LIKE '%$s%'
        OR u.last_name LIKE '%$s%'
        OR u.email LIKE '%$s%'
        OR o.phone LIKE '%$s%'
    ";
}

// Вибірка замовлень з приєднанням даних користувачів (LEFT JOIN)
$sql = "
    SELECT o.*, u.first_name, u.last_name, u.email, u.phone
    FROM orders o
    LEFT JOIN users u ON o.user_id = u.id
    $where_clause
    ORDER BY o.created_at DESC
";
$result = $conn->query($sql);
```

Використано оператор LEFT JOIN для об'єднання таблиці замовлень (orders) і користувачів (users). Це дозволяє отримати повну інформацію про клієнта, навіть якщо замовлення було оформлено без реєстрації (як гість).

Інформація про товари в замовленні зберігається у форматі JSON. Для побудови аналітики (які товари продаються найкраще) необхідно десеріалізувати ці дані та провести їх агрегацію, скрипт наведений нижче.

```
while ($row = $result->fetch_assoc()) {
    // Підрахунок загальних KPI
    $stats['orders_count']++;
    $stats['total_sales'] += $row['total'];

    // Декодування вмісту кошика з JSON
    $cart_items = json_decode($row['cart'], true);

    if (is_array($cart_items)) {
        foreach ($cart_items as $item) {
            $name = $item['name'] ?? 'Невідомий товар';
            $qty = intval($item['qty'] ?? 0);
            $price = floatval($item['price'] ?? 0);
            $line_total = $qty * $price;

            // Накопичення статистики по кожному товару
            if (!isset($stats['products_sold'][$name])) {
                $stats['products_sold'][$name] = ['qty' => 0, 'revenue' => 0];
            }
```

```
            $stats['products_sold'][$name]['qty'] += $qty;
            $stats['products_sold'][$name]['revenue'] += $line_total;
        }
    }
    // ... (збереження даних для відображення в таблиці) ...
}

// Сортування товарів за кількістю продажів (ТОП-5)
uasort($stats['products_sold'], function($a, $b) {
    return $b['qty'] <=> $a['qty'];
});
$top_products = array_slice($stats['products_sold'], 0, 5);
```

Алгоритм, який проходить по кожному замовленню, розбирає JSON-рядок кошика та сумує продажі для кожного товару. Це дозволяє динамічно визначити "бестселери" без створення окремих складних звітів у БД.

Для наочного представлення результатів використовуються графіки. Дані, підготовлені на сервері (PHP), передаються у клієнтський JavaScript для побудови діаграм, скрипт поданий нижче.

```
<script>
    // Передача даних з PHP у JS (безпечна конвертація в JSON)
    const labels = <?= json_encode(array_keys($top_products)) ?>;
    const dataQty = <?= json_encode(array_column($top_products, 'qty')) ?>;
    const dataRev = <?= json_encode(array_column($top_products, 'revenue')) ?>;

    // Ініціалізація стовпчастої діаграми (Кількість продажів)
    if(labels.length > 0) {
        new Chart(document.getElementById('qtyChart').getContext('2d'), {
            type: 'bar',
            data: {
                labels: labels,
                datasets: [{
                    label: 'Продано (шт)',
                    data: dataQty,
                    backgroundColor: 'rgba(59, 130, 246, 0.8)',
                    borderRadius: 6
                }]
            },
            options: {
                responsive: true,
                scales: { y: { beginAtZero: true } }
            }
        });

        // Ініціалізація кільцевої діаграми (Структура доходу)
        new Chart(document.getElementById('revenueChart').getContext('2d'), {
```

```
            type: 'doughnut',
            data: {
                labels: labels,
                datasets: [{
                    label: 'Дохід (грн)',
                    data: dataRev,
                    backgroundColor: ['#3b82f6', '#10b981', '#f59e0b', '#ef4444',
'#8b5cf6'],
                    borderWidth: 0
                }]
            },
            options: {
                responsive: true,
                cutout: '65%' // Стиль "пончик"
            }
        });
    }
</script>
```

Показано інтеграцію з бібліотекою Chart.js. Функція json_encode використовується для передачі масивів даних із PHP у JavaScript. Це дозволяє побудувати два типи діаграм: стовпчасту (для кількості проданих одиниць) та кільцеву (для відображення частки кожного товару у загальному доході).

3.4.8 Розробка скрипту для реалізації модуля складського обліку

Для оперативного контролю залишків продукції розроблено окремий модуль моніторингу складу (stock_dashboard.php). Його функціонал спрямований на візуалізацію поточного стану запасів та автоматичне визначення критичних рівнів товарних позицій.

Нижче наведено код підключення до бази даних та виконання SQL-запиту для отримання повного переліку товарів, відсортованого за зростанням ID.

```
session_save_path(__DIR__ . '/tmp_sessions');
if (!file_exists(__DIR__ . '/tmp_sessions')) mkdir(__DIR__ . '/tmp_sessions', 0777,
true);
session_start();

// Перевірка авторизації (доступ лише для admin)
if (!isset($_SESSION['user_id']) || $_SESSION['role'] !== 'admin') {
    header("Location: login.php");
    exit;
}

// Підключення до БД
```

```
$conn = new mysqli("localhost", "root", "mysql", "production");
if ($conn->connect_error) die("Помилка з'єднання: " . $conn->connect_error);
$conn->set_charset("utf8mb4");

// Отримання списку товарів (ID, назва, ціна, залишок)
$sql = "SELECT id, name, category, price, quantity, image FROM products ORDER BY id
ASC";
$result = $conn->query($sql);

$conn->close();
```

Цей фрагмент коду виконує стандартну процедуру ініціалізації сесії та перевірки прав доступу. SQL-запит оптимізовано: вибираються лише необхідні поля (id, name, quantity, price), що зменшує навантаження на мережу та пришвидшує обробку даних порівняно з вибіркою SELECT *.

Нижче наведено програмний код, у ньому показано алгоритм, який під час генерації HTML-таблиці перевіряє поле quantity та динамічно призначає CSS-клас для візуального оформлення статусу товару.

```
while($row = $result->fetch_assoc()):
    $qty = intval($row['quantity']);

    // Логіка визначення статусу
    if ($qty == 0) {
        $badgeClass = 'badge-critical'; // Червоний (Критично)
        $statusText = 'Немає';
    } elseif ($qty < 100) {
        $badgeClass = 'badge-warning';  // Жовтий (Увага)
        $statusText = 'Закінчується';
    } else {
        $badgeClass = 'badge-ok';       // Зелений (Норма)
        $statusText = 'В наявності';
    }
?>
<tr class="product-row">
    <td>#<?= $row['id'] ?></td>
    <td>
        <span style="font-weight:600; color:var(--primary);"><?=
htmlspecialchars($row['name']) ?></span>
    </td>
    <td><span class="qty-val"><?= $qty ?></span> шт.</td>
    <td><span class="badge <?= $badgeClass ?>"><?= $statusText ?></span></td>
</tr>
```

У циклі while відбувається ітерація по всіх отриманих записах. Для кожного товару виконується умовна перевірка:

– якщо кількість дорівнює 0, присвоюється клас badge-critical (сигналізує про необхідність термінової закупівлі);

– якщо кількість менше 100, присвоюється badge-warning (попередження про низький запас);

– в інших випадках товар маркується як badge-ok. Такий підхід дозволяє адміністратору візуально фільтрувати проблемні позиції без необхідності аналізувати кожну цифру окремо.

Код нижче демонструє JavaScript-код, який відстежує введення тексту в поле пошуку та приховує рядки таблиці, що не відповідають запиту.

```
<script>
    // Додавання прослуховувача події 'keyup' на поле вводу
    document.getElementById('searchInput').addEventListener('keyup', function() {
        let filter = this.value.toLowerCase();
        let rows = document.querySelectorAll('#stockTable tbody tr');

        rows.forEach(row => {
            // Отримання текстового вмісту рядка
            let text = row.innerText.toLowerCase();

            // Перевірка на входження пошукового запиту
            if (text.includes(filter)) {
                row.style.display = ''; // Показати рядок
            } else {
                row.style.display = 'none'; // Приховати рядок
            }
        });
    });
</script>
```

Скрипт використовує метод querySelectorAll для отримання всіх рядків таблиці. При кожному натисканні клавіші (keyup) відбувається перебір рядків (forEach). Якщо текст у рядку містить введену підстроку (includes), властивість style.display очищується (рядок стає видимим), інакше – встановлюється в none (рядок зникає). Це забезпечує миттєву реакцію інтерфейсу і не створює навантаження на сервер, оскільки всі дані вже завантажені в браузер.

3.4.9 Розробка скрипту для реалізації підсистеми інтелектуальної підтримки (AI-чат)

Для підвищення конверсії продажів та покращення взаємодії з клієнтами розроблено модуль інтелектуального асистента (chat_api.php). Цей скрипт виступає посередником (Back-end Proxy) між інтерфейсом користувача та хмарним API Google Gemini.

У коді нижче наведено алгоритм вибірки товарів з бази даних MySQL та формування текстового контексту, який згодом буде передано нейромережі.

```
// Ініціалізація контексту
$contextData = "Список товарів поки недоступний.";

// Підключення до БД
$conn = new mysqli('localhost', 'root', 'mysql', 'production');

if (!$conn->connect_error) {
    // Вибірка перших 50 товарів для контексту
    $result = $conn->query("SELECT name, price FROM products LIMIT 50");

    if ($result) {
        $list = [];
        while ($row = $result->fetch_assoc()) {
            // Форматування рядка: Назва (Ціна)
            $list[] = "- {$row['name']} ({$row['price']} грн)";
        }
        if (!empty($list)) {
            // Об'єднання масиву в один рядок
            $contextData = implode("\n", $list);
        }
    }
    $conn->close();
}
```

Цей фрагмент реалізує підхід RAG (Retrieval-Augmented Generation) на базовому рівні. Скрипт зчитує назви та ціни товарів з таблиці products і перетворює їх у текстовий список ($contextData). Ця змінна стає "короткочасною пам'яттю" для моделі, дозволяючи їй оперувати реальними даними магазину під час генерації відповіді.

Код нижче демонструє універсальну функцію sendCurl, яка використовує бібліотеку cURL для відправки запитів та отримання відповідей від зовнішнього API.

```
function sendCurl($url, $method = 'GET', $data = null) {
```

```
    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); // Вимкнення перевірки SSL (для
локальної розробки)

    if ($method === 'POST') {
        curl_setopt($ch, CURLOPT_POST, true);
        // Кодування даних у JSON формат
        curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($data));
        curl_setopt($ch, CURLOPT_HTTPHEADER, ['Content-Type: application/json']);
    }

    $response = curl_exec($ch);
    $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
    curl_close($ch);

    // Повернення результату та HTTP-коду
    return ['code' => $httpCode, 'response' => json_decode($response, true)];
}
```

Функція sendCurl інкапсулює логіку роботи з мережею. Вона налаштовує заголовки (Content-Type: application/json), метод передачі даних (POST) та обробляє відповідь. Параметр CURLOPT_SSL_VERIFYPEER встановлено у false, що спрощує налаштування на локальному сервері (Localhost), хоча у продакшн-середовищі рекомендується увімкнути перевірку сертифікатів.

У коді нижче наведено процес формування фінального запиту до моделі Gemini, який включає системні інструкції та повідомлення користувача

```
// Формування системної інструкції (Роль + Правила + Дані)
$systemInstruction = "Ти — корисний та розумний AI-асистент на сайті будівельного
магазину.
Твої правила:
1. Будь розумним: Ти можеш відповідати на БУДЬ-ЯКІ питання.
2. Будь продавцем: Якщо питання стосується будівництва — використовуй список товарів.
3. Наші товари:
$contextData";

// Об'єднання інструкції та запиту користувача
$finalPrompt = $systemInstruction . "\n\nКОРИСТУВАЧ ПИШЕ: " . $userMessage;

// URL для генерації контенту (Gemini API)
$generateUrl =
"https://generativelanguage.googleapis.com/v1beta/$validModelName:generateContent?key
=" . $apiKey;

$postData = [
    "contents" => [
        ["parts" => [["text" => $finalPrompt]]]
```

```
    ]
];

// Відправка запиту та отримання результату
$chatResult = sendCurl($generateUrl, 'POST', $postData);

// Вивід відповіді у форматі JSON
if (isset($chatResult['response']['candidates'][0]['content']['parts'][0]['text'])) {
    echo json_encode(['reply' =>
$chatResult['response']['candidates'][0]['content']['parts'][0]['text']]);
} else {
    echo json_encode(['reply' => "Помилка AI: Сервіс тимчасово недоступний."]);
}
```

Застосовано методику Prompt Engineering. Змінна $systemInstruction визначає "особистість" бота: він позиціонується як консультант будівельного магазину. Вставка змінної $contextData всередину інструкції дозволяє моделі "бачити" актуальні ціни та назви товарів. Отриманий від Google JSON-об'єкт розбирається, і текст відповіді (candidates -> content -> parts -> text) передається назад у JavaScript-клієнт для відображення у чаті.

3.4.10 Розробка скрипту для реалізації механізмів тестування та автоматизації обліку

Для перевірки коректності роботи системи на етапі розробки було створено модуль емуляції фінансових транзакцій (force_pay.php). Цей скрипт виконує дві критично важливі функції: зміну статусу замовлення та автоматичне списання реалізованої продукції зі складу.

У коді нижче наведено алгоритм, який генерує унікальний ідентифікатор транзакції та оновлює статус замовлення в базі даних, імітуючи відповідь від банку.

```
if (isset($_POST['order_id'])) {
    $id = intval($_POST['order_id']);

    // Генерація псевдо-ідентифікатора транзакції для тестів
    $fake_payment_id = "TEST_" . time();

    // Оновлення статусу замовлення на 'paid'
    $sql = "UPDATE orders
            SET payment_status = 'paid', payment_id = '$fake_payment_id'
            WHERE id = $id";

    if ($conn->query($sql)) {
```

```
        echo "<h2 style='color:green'>✅ Замовлення #$id оплачено!</h2>";
        // ... (далі списання товару) ...
    } else {
        echo "Помилка SQL: " . $conn->error;
    }
}
```

Цей фрагмент виконує UPDATE-запит до таблиці orders. Важливим моментом є фіксація payment_id. У реальній системі це номер квитанції від LiqPay, а в тестовому середовищі генерується часова мітка (time()). Це дозволяє системі розпізнати замовлення як завершене і розблокувати подальші дії (наприклад, відображення в аналітиці продажів).

Код нижче демонструє логіку, яка розбирає склад кошика оплаченого замовлення та виконує декремент (зменшення) кількості відповідних товарів у таблиці products.

```
// Отримання складу кошика конкретного замовлення
$res_cart = $conn->query("SELECT cart FROM orders WHERE id = $id");

if ($row_cart = $res_cart->fetch_assoc()) {
    $cart_items = json_decode($row_cart['cart'], true);

    if (is_array($cart_items)) {
        foreach ($cart_items as $prod_id => $item) {
            $qty_bought = intval($item['qty']);

            // Атомарне оновлення залишку товару
            // Використання GREATEST(0, ...) запобігає від'ємним значенням
            $update_stock = "UPDATE products
                             SET quantity = GREATEST(0, quantity - $qty_bought)
                             WHERE id = $prod_id";

            if ($conn->query($update_stock)) {
                // Логування дії для візуального контролю
                echo "<li>Товар ID $prod_id: списано $qty_bought шт.</li>";
            }
        }
    }
}
```

У цьому лістингу реалізовано ітеративний прохід по масиву товарів із JSON-поля cart. Для кожного товару виконується SQL-запит на оновлення. Особливу увагу слід звернути на використання SQL-функції GREATEST(0, quantity - $qty_bought). Це механізм захисту цілісності даних: навіть якщо станеться

системний збій і спроба списати більше товару, ніж є, залишок не стане від'ємним (не піде в "мінус"), а зупиниться на нулі.

3.5 Експериментальне дослідження та тестування системи

У рамках експериментальної частини роботи було протестовано реалізовану автоматизовану систему управління продажами. Дослідження проводилося шляхом моделювання реальних сценаріїв взаємодії користувача з вебзастосунком: від пошуку продукції в каталозі до фінального оформлення замовлення та зміни складських залишків. Нижче наведено результати перевірки ключових модулів системи.

На рисунку 3.4 зображена сторінка входу в особистий кабінет.

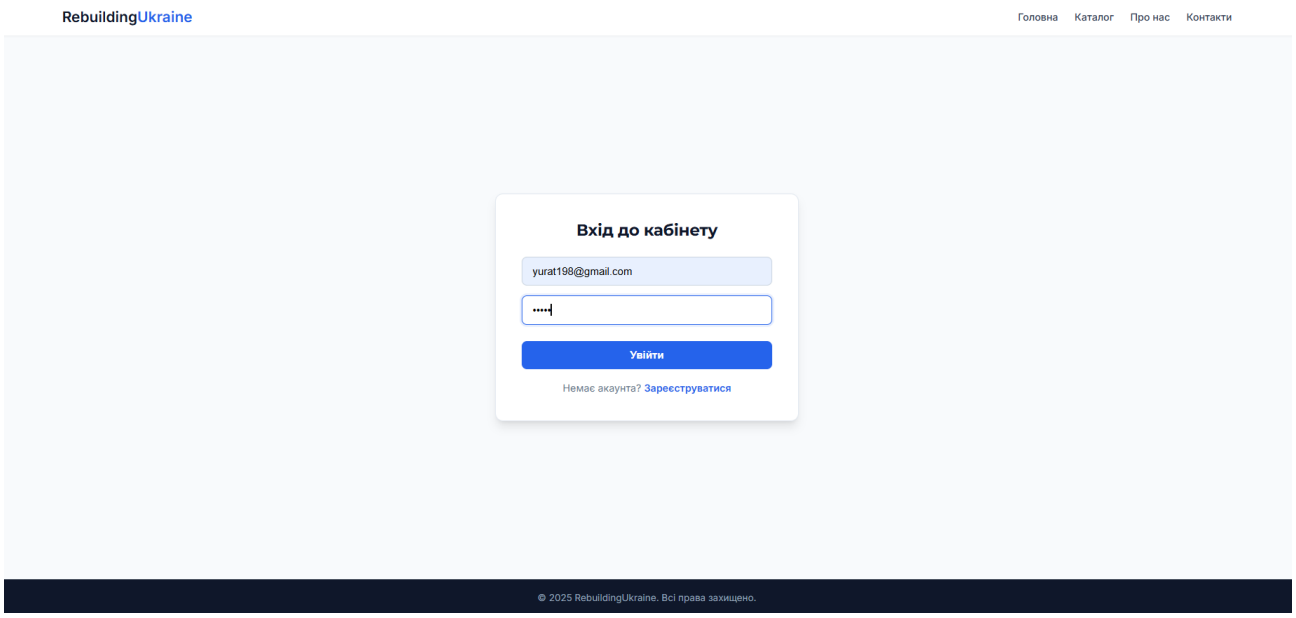


Рисунок 3.4 – Вхід до кабінету

Після входу в особистий кабінет користувач бачить особисті дані (рис. 3.5), які можна змінити, кошик з товарами та архів замовлень (рис. 3.6).

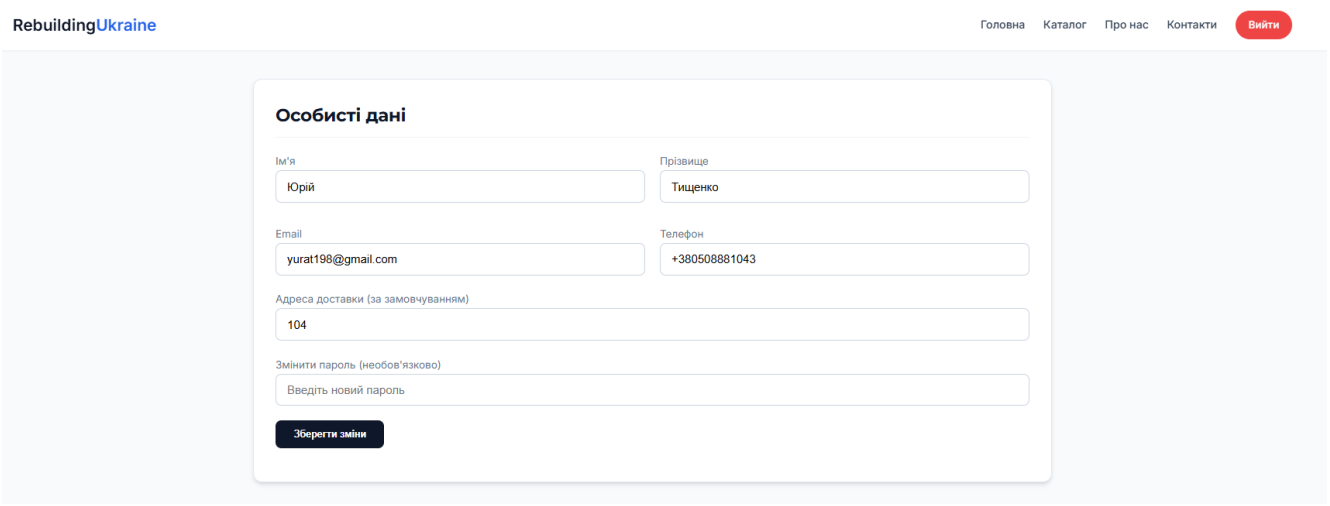


Рисунок 3.5 – Особисті дані

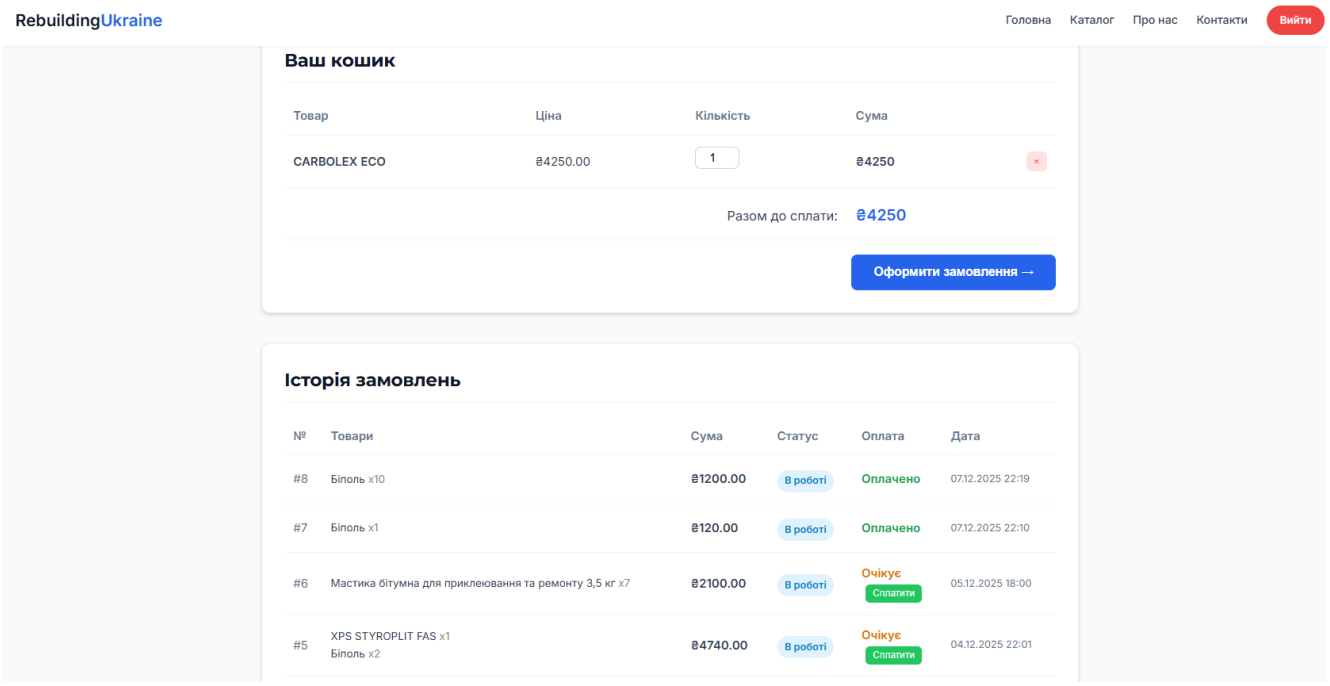


Рисунок 3.6 – Кошик, історія замовлень

Після оформлення замовлення його можна сплатити одразу або згодом, використовуючи платіжну систему LiqPay (рис. 3.7), яка була інтегрована в тестовому режимі. Для тесту сплатимо замовлення з id 6.

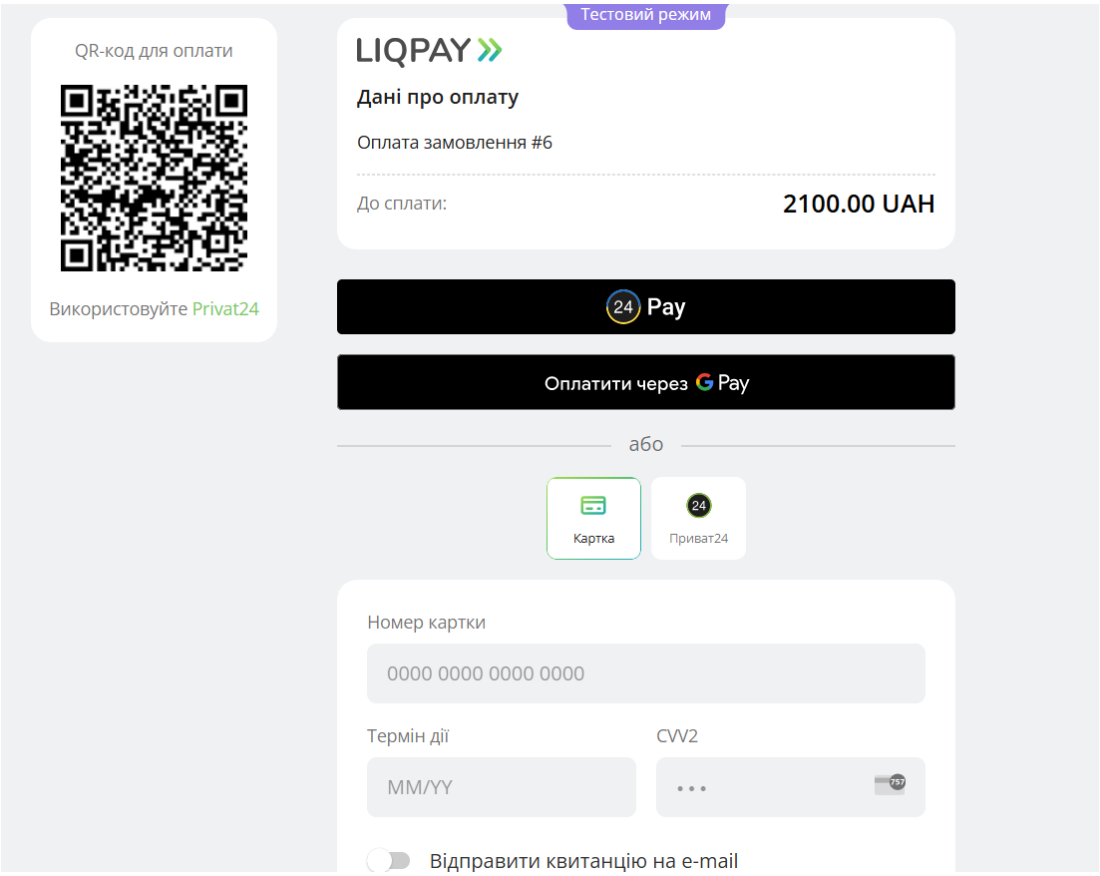


Рисунок 3.7 – Платіжна система LiqPay

Після оплати оновлюються залишки та змінюється статус замовлення у особистому кабінеті користувача (рис. 3.8).

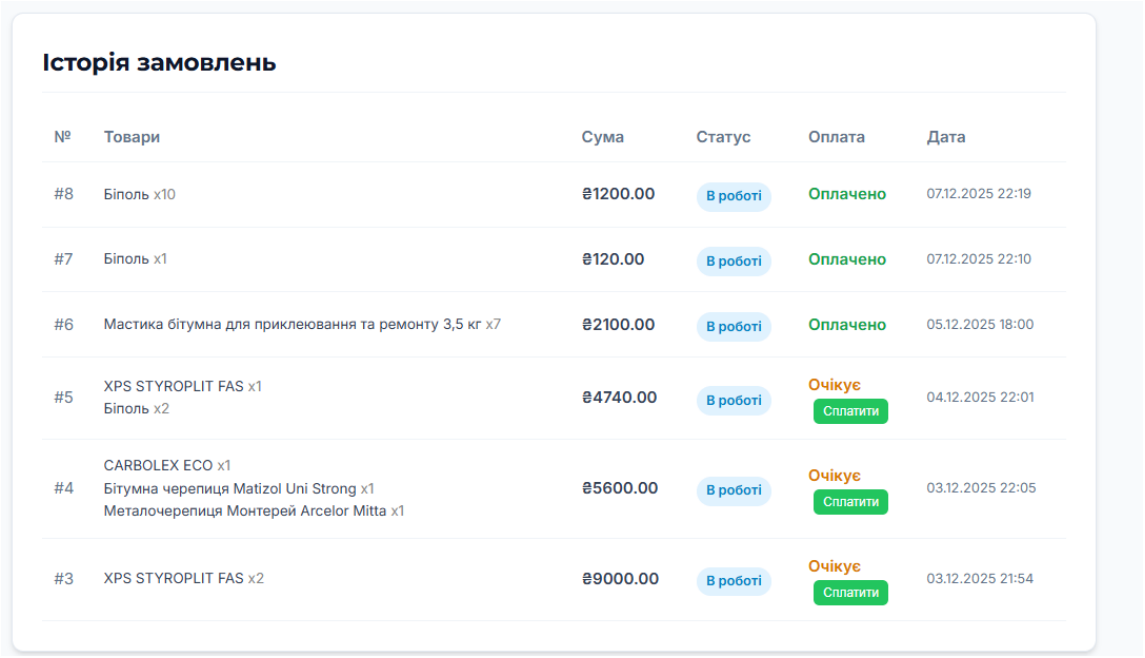


Рисунок 3.8 – Зміна статусу замовлення

Робота AI помічника зображена на рисунках 3.9-3.10.

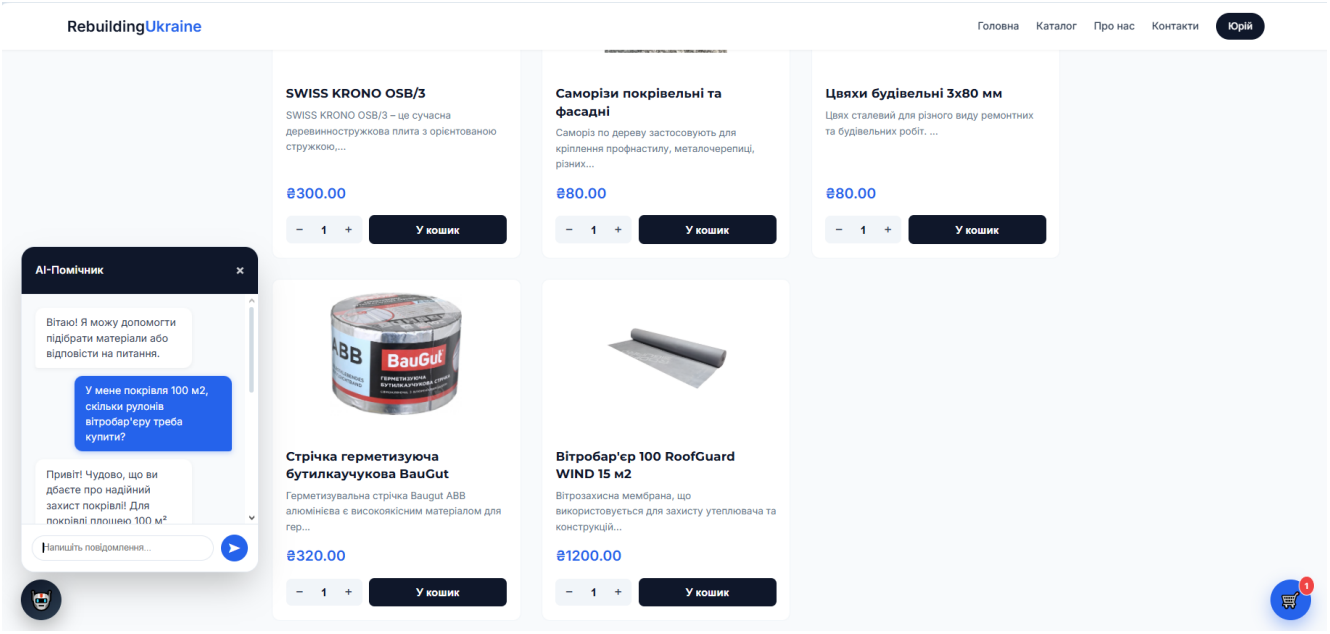


Рисунок 3.9 – Робота AI помічника

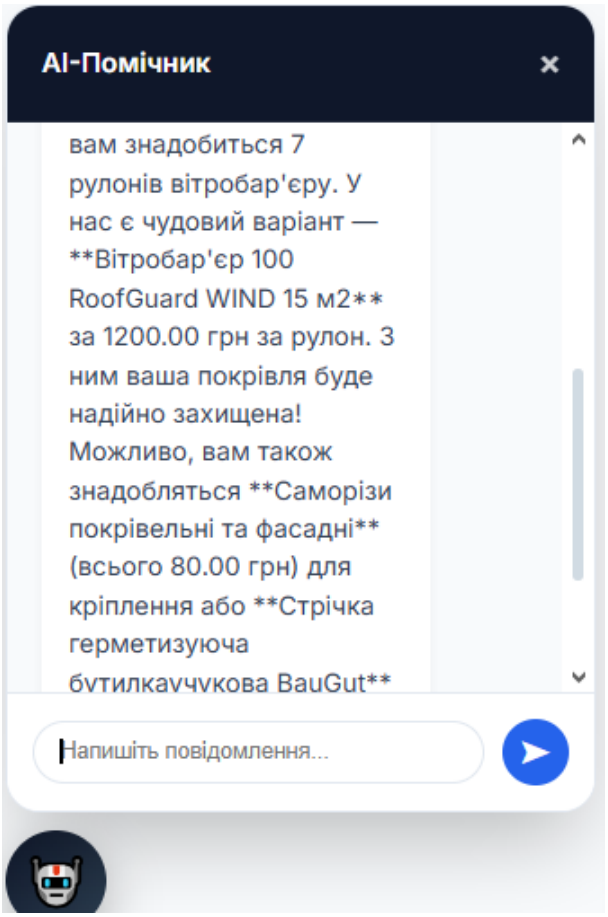


Рисунок 3.10 – Робота AI помічника

4 ОХОРОНА ПРАЦІ

Робота з програмним комплексом передбачає наявність адміністратора, який виконує функції оператора ЕОМ. Ця діяльність вимагає високої концентрації уваги при обробці великих масивів даних та прийнятті оперативних рішень. Оскільки робоче місце обладнане комп'ютерною технікою, на співробітника впливають типові для офісної роботи шкідливі фактори. Мета цього розділу – оцінити ці ризики та забезпечити безпечні умови праці, що відповідають чинному Закону України "Про охорону праці" [30].

4.1 Загальні положення

Площа на одне робоче місце з персональним комп'ютером для дорослих користувачів повинна складати не менше 6 $м^2$, а об'єм - не менше 20 $м^3$.

Розміщення елементів робочого місця не має заважати рухам та переміщенню для експлуатування ПК.

Монітор встановлюється так, щоб відстань від поверхні екрана до очей користувача була 600-700 мм залежно від розміру екрана.

Клавіатура розміщується на робочому або окремому столі на відстані 100-300 мм від краю з боку користувача. Положення клавіатури та кут її нахилу залежить від побажання користувача (як правило, в межах 5-15°). Не допускається хитання клавіатури.

ПК не встановлюється впритул до стіни, перегородки тощо. Не допускається загородження вентиляційних отворів ПК сторонніми предметами.

Під час переміщення ПК, периферійних пристроїв витягається вилка живлення з розетки.

Не допускається ушкодження чи модифікування шнура живлення. Заборонено ставити важкі речі на шнур живлення, тягнути чи надмірно перегинати його, скручувати та зав’язувати шнур живлення у вузол [31].

4.2 Вимоги безпеки перед початком роботи

Перевірити надійність встановлення обладнання на робочому столі. Монітор не має стояти на краю стола. Повернути монітор так, щоб було зручно дивитися на екран – під прямим кутом (а не збоку) і трохи зверху вниз; при цьому екран має бути трохи нахиленим - нижній край ближче до користувача.

Перевірити загальний стан обладнання, справність електропроводки, з'єднувальних шнурів, штепсельних вилок, розеток, заземлення захисного екрана.

У разі виникнення потреби вставити вилку в розетку треба впевнитися, що вона міцно тримається. Заборонено вставляти і виймати вилку мокрими руками.

Відрегулювати та зафіксувати висоту крісла та зручний для користувача нахил спинки.

За потреби приєднати до комп'ютера необхідну апаратуру (принтер, сканер тощо). Усі кабелі, що з'єднують системний блок із іншими пристроями, вмикати та вимикати лише при вимкненому комп'ютері.

Відрегулювати яскравість свічення, контрастність монітора.

Про всі виявлені несправності інформувати керівника відділу і не братися до роботи, доки їх не буде усунуто [31].

4.3 Вимоги безпеки під час виконання роботи

Під час роботи на ПК:

– стійко встановити клавіатуру на робочому столі, не допускаючи її хитання, водночас передбачити можливість її поворотів та переміщень;

– якщо в конструкції клавіатури не передбачено простору для упору долонь, клавіатуру розміщують на відстані не менше 100 мм від краю столу в оптимальній зоні моніторного поля;

– під час роботи на клавіатурі сидіти рівно, не напружуватися;

– щоб зменшити несприятливе навантаження на користувача при роботі з комп'ютерною мишею (вимушена поза, необхідність постійно контролювати

якість дій), забезпечити велику вільну поверхню столу для переміщення комп'ютерної миші та зручного упору ліктьового суглоба;

– періодично при вимкненому комп'ютері прибирати пил із поверхонь апаратури спеціальними серветками.

При роботі з ПК заборонено:

– самостійно розбирати та ремонтувати системний блок (корпус ноутбука), монітор, клавіатуру, комп'ютерну мишу тощо;

– встромляти сторонні предмети до вентиляційних отворів ПК, ноутбука або монітора;

– ставити на системний блок ПК та периферійні пристрої металеві предмети, ємкості з водою (вази, горщики для квітів, склянки), оскільки через потрапляння води у середину апарата може статися пожежа або ураження електрострумом.

Тривалість безперервної роботи за ПК не має перевищувати 2 години. Після цього необхідно зробити 15-хвилинну перерву.

Якщо виник зоровий дискомфорт або інші неприємні відчуття, необхідно зробити нетривалу перерву.

Для зниження нервово-емоційного напруження, стомлення зорового аналізатора, поліпшення мозкового кровообігу, подолання несприятливих наслідків гіподинамії, запобігання втомі доцільно під час декількох перерв виконувати комплекс вправ [31].

ВИСНОВКИ

Метою кваліфікаційної роботи було розроблення системи автоматизації для підвищення ефективності продажу готової продукції виробничого підприємства за рахунок розроблення автоматизованої системи.

Для досягнення поставленої мети було вирішено такі завдання:

– оглянуто та проаналізовано існуючі методи, засоби та автоматизованих систем для продажу готової продукції виробничого підприємства;

– розроблено структурні схеми та алгоритм роботи автоматизованої системи для продажу готової продукції виробничого підприємства;

– реалізовано автоматизовану системи продажу готової продукції виробничого підприємства у вигляді програмного засобу;

– проведено тестування та випробувано автоматизовану систему для продажу готової продукції виробничого підприємства.

Створений веб-застосунок надає клієнтам цілодобовий доступ до повного каталогу будівельних матеріалів. Система дозволяє в реальному часі перевіряти наявність товару на складі та технічні характеристики, а замовлення можна оформити з будь-якого пристрою, підключеного до інтернету.

Особливу увагу приділено юзабіліті: інтерфейс спроєктовано так, щоб скоротити шлях користувача до покупки. Впровадження AI-консультанта додатково спрощує пошук та вибір потрібної продукції. Цей проєкт демонструє практичну цінність цифровізації в роздрібній торгівлі та закладає надійний фундамент для подальшого розширення функціонала й автоматизації бізнес-процесів. За рахунок розроблення даної системи було підвищено ефективність продажу готової продукції виробничого підприємства.

Отримані результати відповідають переліку Цілей сталого розвитку, зокрема Цілі 9 Промисловість, інновації та інфраструктура.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання / Нац. стандарт України. – Вид. офіц. – [Чинний від 2017 – 07 – 01]. – Київ: ДП «УкрНДНЦ», 2016. – 26 с.

2. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 174 Автоматизація, комп'ютерно-інтегровані технології та робототехніка, освітньо-професійних програм: «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / Упоряд. І. Ш. Невлюдов, Р. В. Артюх, В. В. Безкоровайний, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. Харків: ХНУРЕ, 2024. 57 с.

3. Тищенко Ю.О., Хрустальова С.В. Аналіз баз даних систем автоматизації // Computer-integrated technologies, automation and robotics 2025: Proceedings of II st All-Ukrainian Conference, Kharkiv, May 16–17, 2025: Thesises of Reports / [редкол.: І.Ш. Невлюдов (гол. ред.)]. – Харків : [електрон. версія], 2025. – С. 23-25. https://drive.google.com/file/d/1US-0D-v5V4MWtCYMcewmZ8At4dbV3z2K/view

4. Електронна комерція : конспект лекцій для студентів усіх форм навчання першого (бакалаврського) рівня вищої освіти спеціальності 073 – Менеджмент / М. Ю. Карпенко ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2021. – 146 с.

5. Колупаєва І. В., Безсонов С. Стан і напрями розвитку електронної комерції в Україні. Проблеми економіки. 2023. № 2. С. 74-80.

6. Березовська, Л., & Кириченко, А. (2022). РОЗВИТОК ЕЛЕКТРОННОЇ КОМЕРЦІЇ В УКРАЇНІ ТА ЄС. Економіка та суспільство, (42). [Електронний ресурс] URL: https://doi.org/10.32782/2524-0072/2022-42-15 (дата звернення 20.11.2025).

7. Тенденції будівельного ринку України під час воєнного стану. [Електронний ресурс] URL: https://rautagroup.com/uk/tendentsiyi-budivelnogo-rinku-ukrayini-pid-chas-voyennogo-stanu (дата звернення 20.11.2025).

8. Краус К.М., Краус Н.М., Манжура О.В. Електронна комерція та інтернет-торгівля. Київ: Видавництво ТОВ "Аграр Медіа Груп", 2021. 454 с.

9. Огляд основних видів електронної комерції: що треба знати про eCommerce у 2025 році. [Електронний ресурс] URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html (дата звернення 21.11.2025).

10. Електронна комерція: що це, які переваги, недоліки та як запустити бізнес. [Електронний ресурс] URL: https://smartlab-study.com/blog/elektronna-komertsiya-shcho-tse-yaki-perevahy-nedoliky-ta-yak-zapustyty-biznes-0 (дата звернення 21.11.2025).

11. Електронна комерція як сфера діяльності. [Електронний ресурс] URL: https://itforce.ua/blog/elektronna-komertsiia-yak-sfera-diialnosti/ (дата звернення 21.11.2025).

12. Основи UML. [Електронний ресурс] URL: https://docs.kde.org/trunk5/uk/umbrello/umbrello/uml-basics.html (дата звернення 22.11.2025).

13. Як будувати UML-діаграми. [Електронний ресурс] URL: https://dou.ua/forums/topic/40575 (дата звернення 23.11.2025).

14. Що таке діаграма класів UML і найкращий творець діаграм класів UML. [Електронний ресурс] URL: https://www.mindonmap.com/uk/blog/what-is-uml-class-diagram (дата звернення 22.11.2025).

15. Повне розуміння діаграми компонентів UML за допомогою легкого методу. [Електронний ресурс] URL: https://www.mindonmap.com/uk/blog/uml-component-diagram (дата звернення 22.11.2025).

16. Діаграми Прецедентів. [Електронний ресурс] URL: https://lvivqaclub.blogspot.com/2008/10/use-case-uml-diagram.html (дата звернення 22.11.2025).

17. Дізнайтеся все про діаграму активності UML. [Електронний ресурс] URL: https://www.mindonmap.com/uk/blog/uml-activity-diagram (дата звернення 23.11.2025).

18. Для чого потрібні UML діаграми? [Електронний ресурс] URL: https://surl.li/pxpsoz (дата звернення 23.11.2025).

19. Методи сучасної теорії управління: підручник / А.П. Ладанюк, Н.М. Луцька, В.Д. Кишенько, Л.О. Власенко, В.В. Іващук. – Київ : Видавництво Ліра-К, 2019. – 368 с

20. Основні можливості та вхід в phpMyAdmin. [Електронний ресурс] URL: https://thehost.ua/ua/wiki/ispmanager/database/phpmyadmin (дата звернення 24.11.2025).

21. Що таке PHPMyAdmin і що ми можемо з ним робити? [Електронний ресурс] URL: https://surl.li/koiboz (дата звернення 24.11.2025).

22. Visual Studio Code: Вікіпедія. [Електронний ресурс].- URL: https://uk.wikipedia.org/wiki/Visual_Studio_Code (дата звернення: 24.11.2025)

23. Що таке Visual Studio Code. [Електронний ресурс] URL: https://top-keis.com.ua/shho-take-visual-studio-code/ (дата звернення: 24.11.2025)

24. Для чого використовується AMPPS? [Електронний ресурс] URL: https://zoney.svoboda.cx.ua/ukraincyam/dlya-chogo-vikoristovuietsya-ampps.html (дата звернення: 25.11.2025)

25. Що таке мова програмування? [Електронний ресурс] URL: https://www.mathros.net.ua/shho-take-mova-programuvannja.html (дата звернення: 26.11.2025)

26. JavaScript Вікіпедія. [Електронний ресурс]. - URL: https://uk.wikipedia.org/wiki/JavaScript (дата звернення: 26.11.2025)

27. Що таке PHP? [Електронний ресурс] URL: https://site-konstruktor.com.ua/php (дата звернення 26.11.2025).

28. Мова запитів SQL. [Електронний ресурс] URL: https://romatishyn.blogspot.com (дата звернення 26.11.2025).

29. Поняття ER-моделі. Поняття сутності (entity). Атрибути. Види атрибутів. [Електронний ресурс] URL: https://www.bestprog.net/uk/2019/01/24/the-concept-of-er-model-the-concept-of-essence-and-communication-attributes-attribute-types-ua/ (дата звернення 27.11.2025).

30. Про охорону праці : Закон України від 14.10.1992 р. № 2694-XII : станом на 1 жовт. 2023 р. URL: https://zakon.rada.gov.ua/laws/show/2694-12#Text (дата звернення 28.11.2025).

31. Інструкція з охорони праці для працівників,зайнятих на роботах із персональними електронно-обчислювальними машинами [Електронний ресурс] URL: https://dsa.court.gov.ua/userfiles/media/new_folder_for_uploads/dsa/Dod_3_N_36.pdf (дата звернення 29.11.2025).