

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій
(повне найменування факультету)

Кафедра комп'ютерних систем та мереж
(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавра

(ступінь вищої освіти (освітній ступінь))

на тему **РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ АВІАТЕХНІЧНИХ ЗАСОБІВ**

Виконав: студент 4 курсу, групи КНТ-510
спеціальності _____

123 «Комп'ютерна інженерія»

(код і найменування спеціальності)

Освітня програма (спеціалізація) _____

«Комп'ютерна інженерія»

РЯБЧЕНКО Н. А.

(ПРИЗВИЩЕ та ініціали)

Керівник КУДЕРМЕТОВ Р.К.

(ПРИЗВИЩЕ та ініціали)

Рецензент ЄФИМЕНКО М.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування вищого навчального закладу)

Факультет Комп'ютерних наук і технологій
Кафедра «Комп'ютерні системи та мережі»
Ступінь вищої освіти бакалаврський
Спеціальність 123 Комп'ютерна інженерія
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерна інженерія
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮЗавідувач кафедри КУДЕРМЕТОВ Р.К.

“ ” 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА

РЯБЧЕНКА Назара Андрійовича

(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка інтернет-магазину авіатехнічних засобівкерівник проєкту (роботи) к. т. н., доцент, КУДЕРМЕТОВ Равіль Камилович

(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом вищого навчального закладу від “01” квітня 2024 року № 105

2. Строк подання студентом проєкту (роботи) 31 травня 2024 року3. Вихідні дані до проєкту (роботи) опис предметної області, наявні засоби та технології розробки версервісів4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз предметної області;2) Проєктування системи;3) Обґрунтування вибори програмних засобів;4) Розробка системи.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Креслення 1 – Моделювання системи;Плакат 1 – Архітектура інтернет-магазину;Плакат 2 – Прототипи сторінок інтернет-магазину;Плакат 3 – Сторінки інтернет-магазину.

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	КУДЕРМЕТОВ Р. К., к.т.н., доцент		
нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання 01.03.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області, виділення основних користувачів	01.03.2024 р.	
2	Визначення вимоги до системи	10.03.2024 р.	
3	Моделювання прецедентів інтернет-магазину, створення діаграм послідовності прецедентів	15.03.2024 р.	
4	Вибір архітектури інтернет-магазину	25.03.2024 р.	
5	Проектування бази даних, створення схеми бази даних,	05.04.2024 р.	
6	Проектування інтерфейсу користувача	10.04.2024 р.	
7	Вибір програмного забезпечення	15.04.2024 р.	
8	Розробка бази даних	20.04.2024 р.	
9	Розробка інтерфейса користувача	28.04.2024 р.	
10	Тестування системи	10.05.2024 р.	
11	Оформлення отриманих результатів у ПЗ	15.05.2024 р.	
12	Оформлення графічного матеріалу	25.05.2024 р.	
13	Проходження нормоконтролю, перевірки на плагіат рецензування	31.05.2024 р.	

Студент (ка)


 (підпис)

Назар РЯБЧЕНКО

(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)


 (підпис)

Равіль КУДЕРМЕТОВ

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 83с., 25 рис., 5 табл., 1 дод., 17 джерел.

UML-ДІАГРАМА, БАЗА ДАНИХ, БІЗНЕС-ПРОЦЕСИ, ВЕБСЕРВЕР,
ВЕБСИСТЕМА, ВИМОГИ, ІНТЕРНЕТ-МАГАЗИН, КЛІЄНТ-СЕРВЕРНА
АРХІТЕКТУРА

Об'єкт розробки – електронний магазин товарів і послуг авіатехнічних засобів.

Метою роботи є розробка електронного магазину товарів і послуг авіатехнічних засобів Virtual Airbase-Shop.

Основні етапи виконання дипломного проєкту - проведення аналізу предметної області та технічного завдання, проєктування та моделювання Інтернет-магазину, проведення аналізу та вибору програмних засобів, розробка системи.

Проєкт складається з чотирьох розділів.

Перший розділ присвячено опису підприємства Virtual Airbase-Shop. Проводиться аналіз предметної області та розробка вимог до програмного забезпечення.

У другому розділі проводиться проєктування розроблювальної системи, а саме, моделювання програмного забезпечення, розробка архітектури програмного забезпечення, проєктування бази даних та інтерфейсу користувача.

Третій розділ присвячено обґрунтуванню вибору програмних засобів.

Четвертий розділ присвячено розробці системи. У ньому наводиться процес розробки БД та вебсистеми, тестування інтерфейсу користувача.

В результаті виконання дипломного проєкту розроблено електронний магазин товарів і послуг авіатехнічних засобів, яких відповідає всім вимогам технічного завдання.

ЗМІСТ

Вступ	6
1 Аналіз предметної області	7
2 Проєктування системи	8
2.1 Розробка вимог та специфікацій вимог до програмного забезпечення	8
2.2 Моделювання програмного забезпечення	16
2.3 Архітектура програмного забезпечення	25
2.4 Проєктування бази даних	28
2.5 Проєктування інтерфейсу користувача	31
3 Обґрунтування вибору програмних засобів	33
3.1 Клієнтська частина сайту	33
3.2 Серверна частина сайту	38
3.3 Система управління базами даних	44
3.4 Вебсервер	47
4 Розробка системи	52
4.1 Розробка бази даних	52
4.2 Web-розробка	54
4.3 Тестування користувацького інтерфейсу	55
Висновки	61
Перелік джерел посилання	62
Додаток А. Програмна реалізація	64

Графічний матеріал:

Плакат 1 – Моделювання системи;

Плакат 2 – Архітектура інтернет-магазину;

Плакат 3 – Прототипи сторінок інтернет-магазину.

Плакат 4 – Сторінки інтернет-магазину.

ВСТУП

У сучасному світі з розвитком інформаційних і мережевих технологій рівень інформатизації діяльності підприємств став дуже високий. У зв'язку з цим виникла необхідність у веденні комерційної діяльності та надання послуг в мережі Інтернет для розширення клієнтської бази підприємства по всьому світу.

Найбільш популярним і зручним засобом здійснення фінансових операцій між клієнтом і продавцем є інтернет-магазин. Інтернет-магазин – це сайт, який торгує товарами за допомогою мережі Інтернет. Він дозволяє користувачам онлайн, в своєму браузері, сформулювати замовлення на покупку, вибрати спосіб оплати і доставки замовлення, оплатити замовлення.

Обсяг інформації, що передається по різних мережах неухильно зростає. Одночасно з цим зростає і швидкість мереж. Кількість даних, які переносяться з одного комп'ютера на інший, обчислюються вже не мегабайтами, як це було ще десять років тому, а гігабайтами і навіть терабайтами.

Здавалося б, при повсюдному поширенні високошвидкісних мереж проблема повільній швидкості роботи в Інтернеті повинна була б повністю зникнути. Однак не всі провайдери надають можливість безлімітного використання трафіку і не всі користувачі можуть його собі дозволити (і не всім він потрібен). У свою чергу розмір сайтів і їх сценарна складність неухильно ростуть, що збільшує трафік і уповільнює роботу в Інтернеті як для користувачів оплачують трафік, так і для безлімітних користувачів. І сьогодні нерідко можна потрапити в ситуацію, коли сайти завантажуються досить тривалий час і не завжди повністю, а відправлення баз даних великого розміру вже стає проблемою для повільних або сильно завантажених каналів зв'язку.

Тому актуальним завданням є розробка сучасного, безпечного і легкого Інтернет-магазину для продажу авіатехнічних засобів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Virtual Airbase-Shop – представництво авіаконцерну, призначене для продажу авіатехнічних засобів. Virtual Airbase-Shop здійснює продаж авіатехнічних засобів відповідно до міжнародного та національного законодавства. Під авіатехнічними засобами маються на увазі безпілотні літальні апарати (БПЛА) різних видів.

Інтернет-магазин - це Web-сайт, який приймає замовлення на матеріальні або електронні товари і послуги від відвідувачів в режимі реального часу [1]. Ринок інтернет-магазинів на сьогоднішній день є перенасиченим, проте, навіть не дивлячись на це, інтернет-магазини приваблюють простотою організації. Багато в чому на це вплинув досвід інших країн, в яких, не дивлячись ні на що, інтернет бізнес не тільки успішно функціонує, а й успішно розвивається.

Інтернет-магазин дозволяє користувачам онлайн, в своєму браузері, сформувані замовлення на покупку, вибрати спосіб оплати і доставки замовлення, оплатити замовлення, а також здійснює доставку товару. Управління інтернет-магазином - це процес планування, координації та контролю бізнес-процесів в інтернет-магазині. Суть даного процесу полягає в оптимальному використанні як інтернет ресурсів, так і трудових ресурсів.

Типи бізнес-процесів інтернет-магазину:

- основні – спрямовані на створення цінності, що продається;
- забезпечуючі – не впливають на виробництво цінності, проте забезпечують існування підприємства в цілому;
- керівничі – допомагають контролювати функціонування підприємства, безперебійність основних процесів.

Використання електронних систем оплати дозволяє позбутися від необхідності наявності в штаті співробітників бухгалтера, що спрощує ведення комерційної діяльності. Функції доставки товару виконуються службою доставки, що є поштовою організацією.

Розглянемо співробітників інтернет-магазину.

Адміністратор інтернет-магазину є головною ланкою в бізнес-процесі. Адміністратор поєднує функції контент-менеджера і своєчасно оновлює і наповнює базу даних інтернет-магазину, в залежності від наявності/відсутності товару.

Працівник складу перевіряє наявність товару на складі і передачу товару службі доставки.

Функціональні обов'язки адміністратора:

- своєчасне оновлення бази даних інтернет-магазину.

Функціональні обов'язки працівника складу:

- перевірка наявності товару на складі;
- передачу товару службі доставки.

Даний інтернет-магазин повинен дозволяти автоматизувати наступні бізнес-процеси:

- продаж товарів;
- оновлення бази даних товарів;
- обробку заявок клієнтів;
- взаємодію з системою оплати;
- інформування працівників складу про замовлення;
- передачу інформації про замовлення службі доставки.

2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Розробка вимог і специфікацій вимог до програмного забезпечення

У цьому розділі наведені вимоги до програмного забезпечення. У ньому викладені вимоги до розробляється програмного продукту [2]. Він охоплює загальний опис програмного продукту, включаючи функціональні вимоги, вимоги

до характеристик користувача, вимоги до апаратного та програмного забезпечення, перелік обмежень, а також перелік функціональних вимог.

2.1.1 Призначення

Цей розділ призначений для визначення можливостей інтернет-магазину, щоб служити керівництвом для розробників, з одного боку, і перевірки автентичності програмного забезпечення для потенційного клієнта - з іншого.

2.1.2 Область дії

Інтернет-магазин призначений для надання каналу збуту для постачальників товарів, а для клієнтів - є майданчиком для здійснення покупок з використанням мережі Інтернет, не відвідуючи магазин фізично. Модуль адміністрування дозволить оновлювати і наповнювати базу даних інтернет-магазину.

2.1.3 Визначення

HTML (HyperText Markup Language) - стандартизована мова розмітки документів у Всесвітній павутині.

HTTP (HyperText Transfer Protocol) - протокол прикладного рівня передачі даних (початково - у вигляді гіпертекстових документів в форматі HTML, зараз використовується для передачі довільних даних).

HTTPS (HyperText Transfer Protocol Secure) - розширення протоколу HTTP для підтримки шифрування з метою підвищення безпеки.

IIS (Internet Information Services, до версії 5.1 - Internet Information Server) - пропріетарний набір серверів для декількох служб Інтернету від компанії Microsoft.

MySQL - вільна реляційна система управління базами даних.

PHP (Hypertext Preprocessor) - скриптова мова загального призначення, інтенсивно застосовується для розробки вебдодатків.

Serial Attached SCSI (SAS) - комп'ютерний інтерфейс, розроблений для обміну даними з такими пристроями, як жорсткі диски, накопичувачі на оптичному диску і т. д. SAS використовує послідовний інтерфейс для роботи з безпосередньо підключаються накопичувачами (Direct Attached Storage (DAS) devices) .

SEO (Search Engine Optimization) - комплекс заходів щодо внутрішньої і зовнішньої оптимізації для підняття позицій сайту в результатах видачі пошукових систем по певних запитах користувачів, з метою збільшення мережевого трафіку (для інформаційних ресурсів) і потенційних клієнтів (для комерційних ресурсів) і подальшої монетизації (отримання доходу) цього трафіку.

SQL (structured query language) - декларативна мова програмування, застосовувана для створення, модифікації та управління даними в реляційній базі даних, керованої відповідною системою управління базами даних.

SRS (Software Requirement Specification) - специфікація для певного програмного виробу, програми або набору програм, які виконують певні функції в специфічному середовищі.

SSH (Secure Shell) - мережевий протокол прикладного рівня, що дозволяє здійснювати віддалене управління операційною системою і тунелювання TCP-з'єднань (наприклад, для передачі файлів).

SSL (Secure Sockets Layer) - криптографічний протокол, який передбачає більш безпечний зв'язок.

Адміністратор, відвідувач, покупець, працівник складу - особи беруть участь в роботі системи.

Система управління базами даних (СУБД) - сукупність програмних і лінгвістичних засобів загального або спеціального призначення, що забезпечують управління створенням і використанням баз даних.

2.1.4 Загальний опис

У цьому розділі наведено загальний опис розроблюваного програмного продукту, класи та характеристики користувачів, операційне середовище, перелік обмежень дизайну і реалізації, припущення і залежності [3].

2.1.4.1 Загальний погляд на продукт

Розроблюваний інтернет-магазин - це продукт, який поєднує універсальний і зручний для кінцевого користувача дизайн, що дозволяє забезпечити працездатність Інтернет-магазину на максимально можливій кількості пристроїв з використанням кроссплатформених сумісних інтернет-технологій і простоту

внутрішнього адміністрування сайту. У ньому забезпечено дотримання сучасних стандартів безпеки вебсайтів для захисту персональних даних покупців та інформації інтернет-магазину, а також дотримані правила забезпечення SEO-оптимізації для підвищення рейтингу сайту в результатах пошукових запитів користувачів мережі Інтернет.

2.1.4.2 Класи та характеристики користувачів

Опис класів і характеристик користувачів подані у таблиці 2.1.

Таблиця 2.1 - Опис класів і характеристик користувачів

Класи користувачів	Характеристика користувачів
Відвідувач	Відвідувач - користувач мережі Інтернет, який відвідав сторінку Інтернет-магазину. Він може переглянути каталог авіатехнічних засобів, опис авіатехнічних засобів, контактну інформацію магазину, сформувати кошик і здійснити замовлення.
Покупець	Покупець - відвідувач, який сформував корзину і здійснив замовлення.
Адміністратор Інтернет-магазину	Адміністратор - співробітник власника інтернет-магазину, який має можливість керувати всіма сутностями інтернет-магазину. Він може оновлювати базу даних товару. Розмежування рівня доступу до редагування сутностей реалізується за допомогою механізму прав
Працівник складу	Працівник складу - співробітник власника інтернет-магазину, який інформує адміністратора про стан товару на складі і передає товар службі доставки.

2.1.4.3 Операційне середовище

Інтернет-магазин буде працювати з клієнтським ПЗ що підтримує технологію HTML 4.01 або вище.

Інтернет-магазин буде працювати на обладнанні під керуванням Windows Server 2003 або вище, і використовувати IIS версії 6.0 або вище.

Інтернет-магазин буде використовувати СУБД MySQL версії 4.0.12 або вище.

2.1.4.4 Обмеження дизайну і реалізації

Документація системи по конструкції, коду, і супроводу повинна відповідати стандартам IEEE, згаданим в стандарті IEEE-830-1998.

Весь код повинен відповідати стандарту HTML не нижче версії 4.01.

Всі програмні коди, що відповідають за клієнт-серверну взаємодію повинні бути написані на мові PHP.

Інтернет-магазин повинен використовувати стандарт ANSI SQL-99.

2.1.4.5 Припущення і залежності

Інтернет-магазин справно функціонує тільки за умови, що все необхідне для нього обладнання знаходиться в робочому стані.

Інтернет-магазин справно функціонує тільки за умови, що між серверним вузлом і користувальницької стороною існує стабільний (безперервний протягом сесії використання системи) канал зв'язку.

2.1.5 Функції продукту

У цьому розділі наведено перелік функціональних вимог до програмного продукту.

2.1.5.1 Зміна кошика

Покупець може зробити, скасувати або змінити замовлення продукту або продуктів з кошика покупок.

2.1.5.2 Сплата

Покупець може вибрати електронну систему оплати, за допомогою якої хоче здійснити замовлення і, надавши свої реквізити і пройшовши перевірку, здійснити оплату. Безпека надаватиметься третьою стороною, тобто системою оплати.

2.1.5.3 Сповіщення про замовлення

Після всіх транзакцій генерується текстовий файл, одна копія якого надсилається на адресу електронної пошти клієнта, друга - надсилається в службу доставки і працівникові складу, а третя - зберігається на сервері.

2.1.6 Вимоги до зовнішніх інтерфейсів

У цьому розділі наведено перелік вимог до апаратних і програмних інтерфейсів.

2.1.6.1 Інтерфейси обладнання

Жорсткі диски з інтерфейсом SAS дозволяють досягти пропускну здатності 6 Гбіт/сек, що дозволить зменшити апаратні обмеження на швидкість читання/запису при зверненнях до файлової системи. На серверах повинен бути встановлений апаратний RAID5.

Gigabit Ethernet Network Adapter (гігабітний мережевий інтерфейс) - необхідний для швидкого обміну інформацією між серверами системи, що зменшить вплив технічних характеристик обладнання на швидкість передачі даних. Наявність гігабітних мережевих інтерфейсів на серверному обладнанні системи має на увазі наявність апаратного керованого гігабітного комутатора, який дозволить об'єднати сервери в локальну обчислювальну мережу.

2.1.6.2 Програмні інтерфейси

Програмний продукт знаходиться на сервері, тому для нього потрібно будь-скриптова мова, такий як PHP.

Для програмного продукту потрібно СУБД для зберігання інформації про товари і клієнтів, така як MySQL.

Для програмного продукту потрібно DNS (domain name space) для розміщення в мережі Інтернет.

Для доступу користувача до програмного продукту потрібен веббраузер, вимоги до якого описані в розділі 2.1.7.4.

2.1.6.3 Інтерфейси передачі даних

Програмний продукт повинен обробляти запити на сортування і фільтрування даних каталогу.

Програмний продукт повинен обробляти запити на формування замовлення.

Адміністратор інтернет-магазину повинен мати можливість своєчасно оновлювати базу даних інтернет-магазину.

2.1.6.4 Графічні інтерфейси

Графічні інтерфейси повинні бути захищені від несанкціонованих і нетипових дій користувача.

Графічні інтерфейси повинні виводити повідомлення про помилку в разі виникнення некоректних або помилкових дій системи.

Графічні інтерфейси повинні виводити повідомлення про помилку в разі появи некоректних дій користувача.

Вважається неприйнятним, якщо будь-якими діями в вебінтерфейсі користувач міг викликати недієздатність системи частково або повністю для інших користувачів системи.

2.1.7 Інші нефункціональні вимоги

У цьому розділі наведено перелік нефункціональних вимог.

2.1.7.1 Вимоги до продуктивності

Інтернет-магазин повинен обробляти одночасні запити з часом відгуку, що не перевищує 3 сек на стороні СУБД. Максимальний час очікування клієнтом обробки свого запиту (без врахування затримки до серверів) становить 5 секунд, в залежності від якості з'єднання клієнта.

Інтернет-магазин повинен бути функціонуючим 99,9% часу, тобто простоювати не більше 9 годин на рік.

Інтернет-магазин повинен обробляти базу даних з 10000000 записів із середнім об'ємом однієї записи 1 Кб.

Система повинна обробляти пікове навантаження при виході з ладу до 30% вузлів системи.

Час від надходження запиту на сервер до віддачі контенту користувачеві (тобто загальний час обробки запиту на серверної частини) має становити не більше 5 сек при нормальному навантаженні на обладнання.

2.1.7.2 Вимоги до збереження даних

Всі дані Інтернет-магазину повинні зберігатися в базі даних, яка відповідає ANSI SQL-99 і використовує СУБД MySQL версії 4.0.12 і вище.

Всі дані системи, в тому числі бази даних, а також програмний і виконуваний код компонентів системи повинні мати дві резервні копії. Повинна здійснюватися «гаряча» система резервного копіювання даних в базах даних і «холодна» для всіх інших файлів. Критерієм старіння резервної версії вважається вихід трьох новіших версій резервної копії даних.

Резервне копіювання даних повинне здійснюватися під час найменшого навантаження на систему (з 03:00 до 06:00 за місцевим часом знаходження сервера) кожен день.

Обмін даними між системою і клієнтом повинен відбуватися у вигляді передачі зашифрованих даних з ключем не менше 256 біт за технологією Secure Socket Layer (SSL) через захищене HTTP-з'єднання (HTTPS).

2.1.7.3 Вимоги до безпеки

2.1.7.3.1 Вимоги до розмежування доступу до інформації

Користувач системи має доступ до інформації, доступної для його класу. Клієнт може переглядати каталог і інформацію товару і формувати замовлення. Відвідувач може робити, то ж що і клієнт (у разі здійснення замовлення він стає клієнтом). Адміністратор має доступ до всіх даних системи.

2.1.7.3.2 Вимоги до аутентифікації і авторизації

Операційна система на серверах повинна забезпечувати ідентифікацію та аутентифікацію адміністратора ОС при його локальних і віддалених запитах на доступ. ОС повинна надавати можливість для ідентифікації і аутентифікації за ідентифікатором (коду) і пароллю умовно-постійної дії.

2.1.7.3.3 Вимоги до криптографічного захисту інформації

Віддалений доступ до обладнання адміністратором повинен бути здійснений по протоколу SSH v.2 з шифруванням SSL-ключем довжиною 256 біт. Всі паролі (як адміністраторські, так і призначені для користувача) повинні зберігатися в системі в зашифрованому вигляді. Відновлення забутого пароля таким чином можливо тільки шляхом завдання нового пароля.

2.1.7.4 Вимоги до портативності

Продукт заснований на HTML, MySQL, і PHP. Таким чином, клієнтська частина повністю переносима, і будь-яка система, яка використовує будь-який веббраузер з підтримкою HTML 4.01, повинна мати можливість використовувати функції системи, включаючи будь-яку апаратну платформу, яка доступна або буде доступна в майбутньому. Кінцевий користувач використовує цю систему на будь-якій ОС. Система повинна працювати на ПК, ноутбуках, КПК і т. д.

2.2 Моделювання програмного забезпечення

У цьому розділі проведено моделювання програмного продукту.

2.2.1 Діаграми UML

UML (Unified Modeling Language) – уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування [4]. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення. UML є мовою широкого профілю, це відкритий стандарт, що використовує графічні позначення для створення абстрактної моделі системи, яка називається UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем.

UML може бути застосовано на всіх етапах життєвого циклу аналізу бізнес-систем і розробки прикладних програм. Різні види діаграм які підтримуються UML, і найбагатший набір можливостей представлення певних аспектів системи робить UML універсальним засобом опису як програмних, так і ділових систем. Такі діаграми підвищують супроводжуваність проекту і полегшують розробку документації. Моделі розроблені в UML, дозволяють значно спростити процес кодування і направити зусилля програмістів безпосередньо на реалізацію системи.

2.2.1.1 Діаграма варіантів використання

Діаграма варіантів використання описує функціональне призначення системи або, іншими словами, те, що система буде робити в процесі свого функціонування. Вона є вихідним концептуальним поданням або концептуальною моделлю системи в процесі її проєктування і розробки.

Суть даної діаграми полягає в наступному: проєктована система представляється у вигляді безлічі так званих варіантів використання, що надаються системою безлічі акторів або сутностей, що взаємодіють з системою. При цьому актором (actor) або дійовою особою називається будь-яка сутність, що взаємодіє з системою ззовні. Це може бути людина, технічний пристрій, програма або будь-яка інша система, яка може служити джерелом впливу на моделируемую систему так, як визначить сам розробник. У свою чергу, варіант використання (use case) служить для опису сервісів, які система надає актору. Іншими словами, кожен варіант використання визначає деякий набір дій, який чинять системою при діалозі з актором.

Варіанти використання визначають функціональні можливості. Кожен з них представляє певний спосіб використання. Таким чином, кожен варіант використання відповідає послідовності дій для того, щоб клієнт міг отримати певний результат.

При цьому загальні властивості варіантів використання можуть бути представлені трьома різними способами, а саме – за допомогою відношень включення, розширення і узагальнення.

Відношення асоціації – одне з фундаментальних понять у мові UML і в тій чи іншій мірі використовується при побудові всіх графічних моделей систем у формі канонічних діаграм.

Включення (англ. include) у мові UML – це різновид відношення залежності між базовим варіантом використання і його спеціальним випадком. При цьому відношенням залежності (англ. dependency) є таке відношення між двома елементами моделі, при якому зміна одного елемента (незалежного) приводить до зміни іншого елемента (залежного).

Відношення розширення (англ. extend) визначає взаємозв'язок базового варіанта використання з іншим варіантом використання, функціональна поведінка якого задіюється базовим не завжди, а тільки при виконанні додаткових умов.

На рисунку 2.1, зображена діаграма варіантів використання для Інтернет-магазину. Опис акторів приведено в розділі 2.1.4.2.

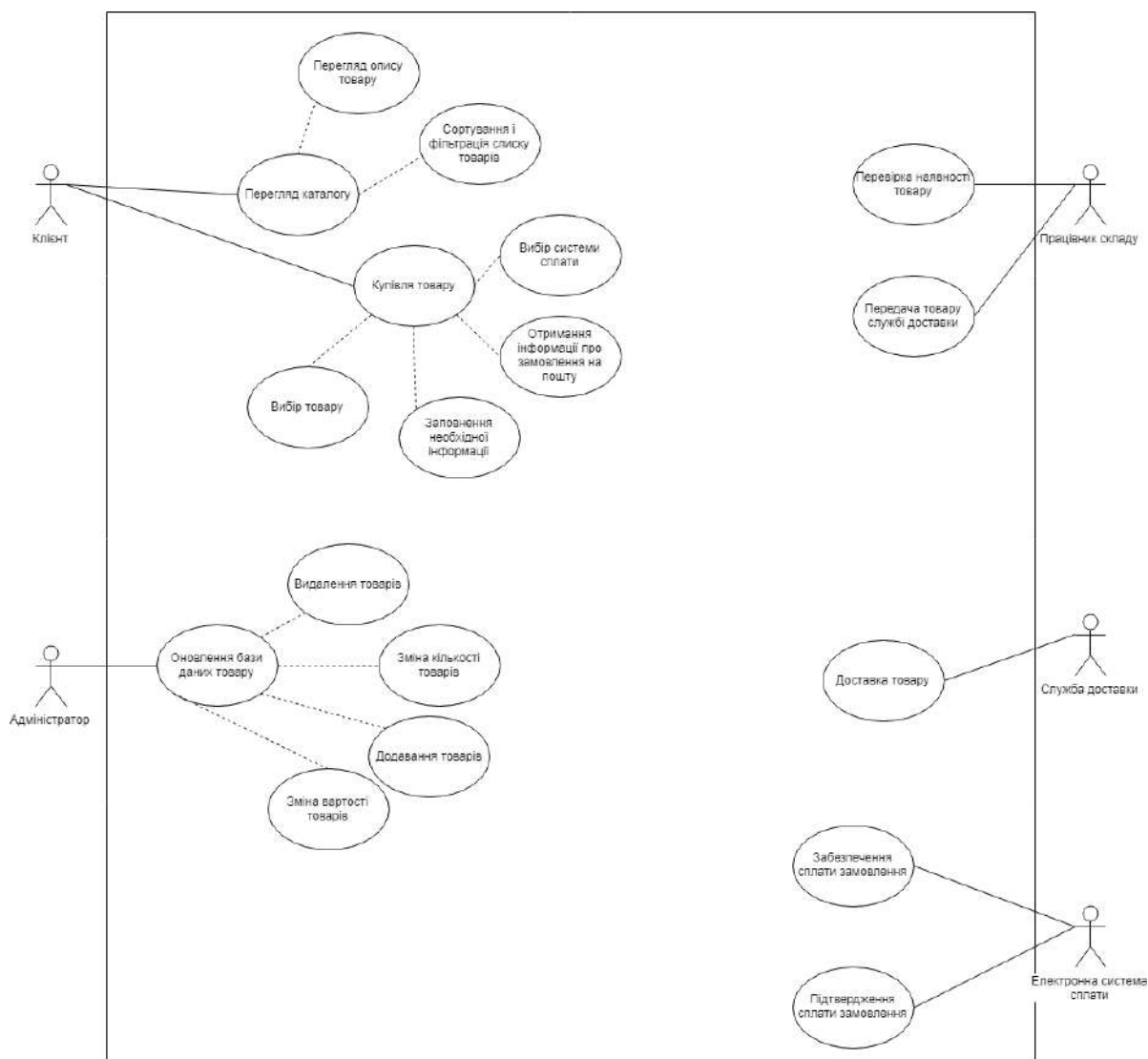


Рисунок 2.1 – UML-діаграма варіантів використання системи

2.2.1.2 Діаграми послідовностей

Для моделювання взаємодії об'єктів в мові UML використовуються відповідні діаграми взаємодії. Одним з аспектів взаємодії є час. Для подання

тимчасових особливостей передачі і прийому повідомлень між об'єктами використовується діаграма послідовності.

Хоча розглянута діаграма варіантів використання і використовується для специфікації динаміки поведінки систем, час в явному вигляді в ній не присутня. Однак часовий аспект поведінки може мати суттєве значення при моделюванні синхронних процесів, що описують взаємодії об'єктів. Саме для цієї мети в мові UML використовуються діаграми послідовності.

2.2.1.2.1 Прецедент «Купівля товару»

На рисунку 2.2 представлена діаграма послідовності для прецеденту «Купівля товару».

Основний виконавець: Клієнт.

Зацікавлені сторони і їх потреби:

- клієнт (хоче купити товар);
- підприємство (хоче отримати грошові кошти від продажу товару).

Результати (постумови): клієнт отримав товар, підприємство отримало кошти.

Розглянемо основний потік (основний успішний сценарій).

- клієнт відкриває сторінку сайту з каталогом;
- вебсистема відправляє запит в БД на каталог;
- БД повертає каталог;
- на сторінці сайту відображається каталог;
- клієнт формує замовлення;
- клієнт натискає кнопку «Submit»;
- вебсистема виконує контроль введення клієнта;
- вебсистема відкриває форму для введення інформації про клієнта;
- клієнт вводить свої реквізити;
- клієнт натискає кнопку «Submit»;
- вебсистема виконує контроль введення клієнта;
- реквізити передаються системі оплати;

- система оплати перевіряє правильність реквізитів;
- система оплати підтверджує проходження операції оплати;
- вебсистема передає зміни в каталозі товарів БД;
- БД здійснює відповідні зміни інформації;
- вебсистема відправляє інформацію про замовлення на пошту клієнта;
- вебсистема відправляє інформацію про замовлення службі доставки;
- служба доставки доставляє замовлення.

Розглянемо альтернативні потоки (розширення).

В основному потоку «вебсистема виконує контроль введення клієнта» вказані некоректні дані:

- вебсистема інформує клієнта, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- клієнт виправляє дані;
- система здійснює перевірку даних.

В основному потоку «вебсистема виконує контроль введення клієнта» вказані некоректні дані:

- вебсистема інформує клієнта, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- клієнт виправляє дані;
- система здійснює перевірку даних.

В основному потоку «система оплати перевіряє правильність реквізитів» вказані некоректні дані:

- вебсистема інформує клієнта, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- клієнт виправляє дані;
- система здійснює перевірку даних.

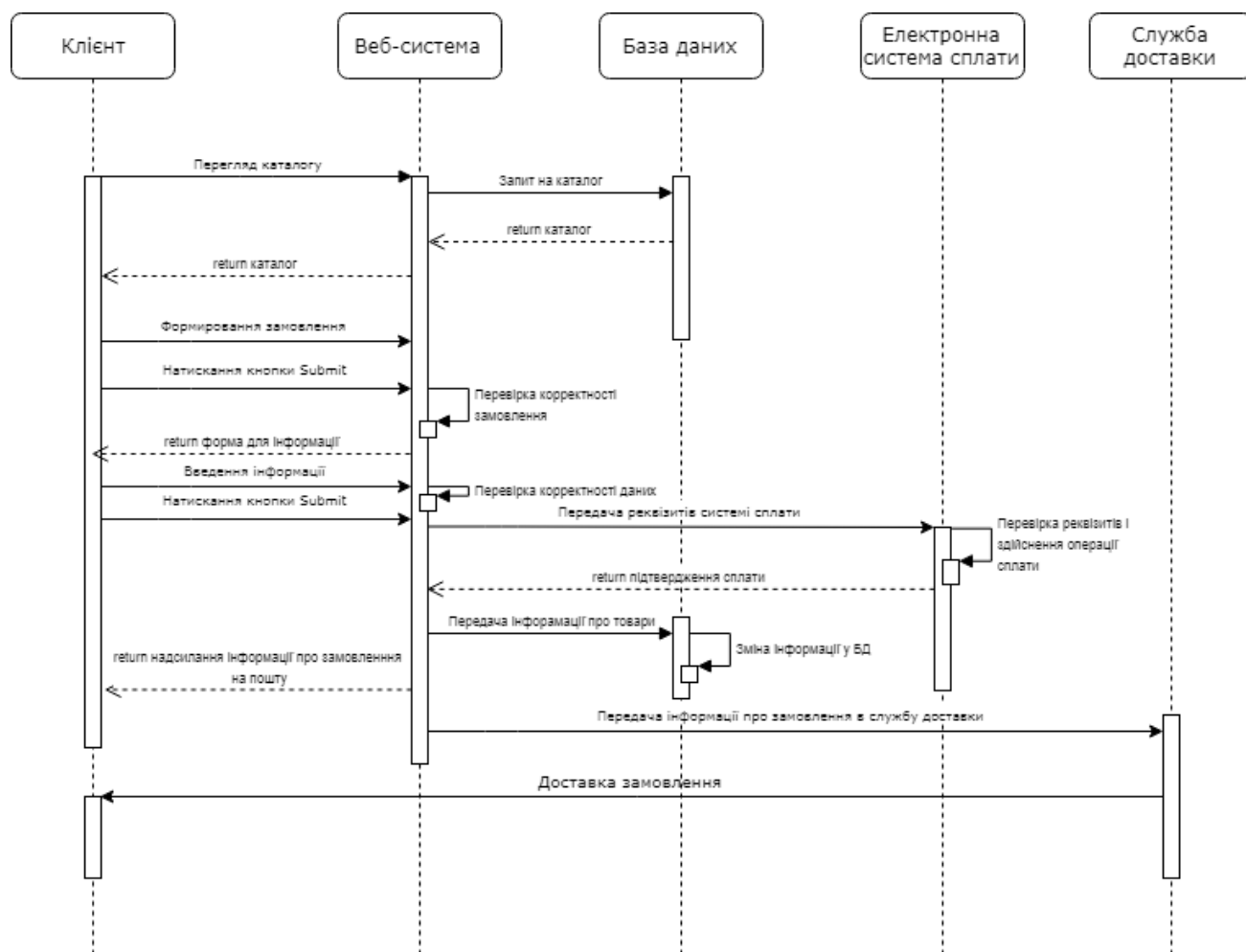


Рисунок 2.2 – UML-діаграма послідовності «Купівля товару»

2.2.1.2.2 Прецедент «Додавання товару»

На рисунку 2.3 представлена діаграма послідовності для прецеденту «Додавання товару».

Основний виконавець: Адміністратор.

Зацікавлені сторони і їхні потреби:

- адміністратор (хоче додати товар);
- підприємство (хоче мати актуальний каталог товарів Інтернет-магазину).

Результати (постумови): Адміністратор додав товар. Підприємство має актуальний каталог товарів Інтернет-магазину.

Розглянемо основний потік (основний успішний сценарій):

- адміністратор відкриває сторінку авторизації;
- адміністратор вводить логін і пароль;

- вебсистема виконує контроль введення адміністратора;
- вебсистема відправляє запит в БД на каталог;
- БД повертає каталог;
- на сторінці сайту відображається каталог;
- адміністратор натискає кнопку «Add»;
- вебсистема відкриває форму для додавання товару;
- адміністратор вводить дані про товар;
- вебсистема виконує контроль введення даних про товар адміністратором;

- вебсистема передає зміни в каталозі товарів БД;
- БД здійснює відповідні зміни інформації;
- БД повертає каталог;
- на сторінці сайту відображається каталог;

Розглянемо альтернативні потоки (розширення).

В основному потоку «вебсистема виконує контроль введення адміністратора» вказані некоректні дані:

- вебсистема інформує адміністратора, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- адміністратор виправляє дані;
- система здійснює перевірку даних.

В основному потоку «вебсистема виконує контроль введення даних про товар адміністратором» вказані некоректні дані:

- вебсистема інформує адміністратора, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- адміністратор виправляє дані;
- система здійснює перевірку даних.

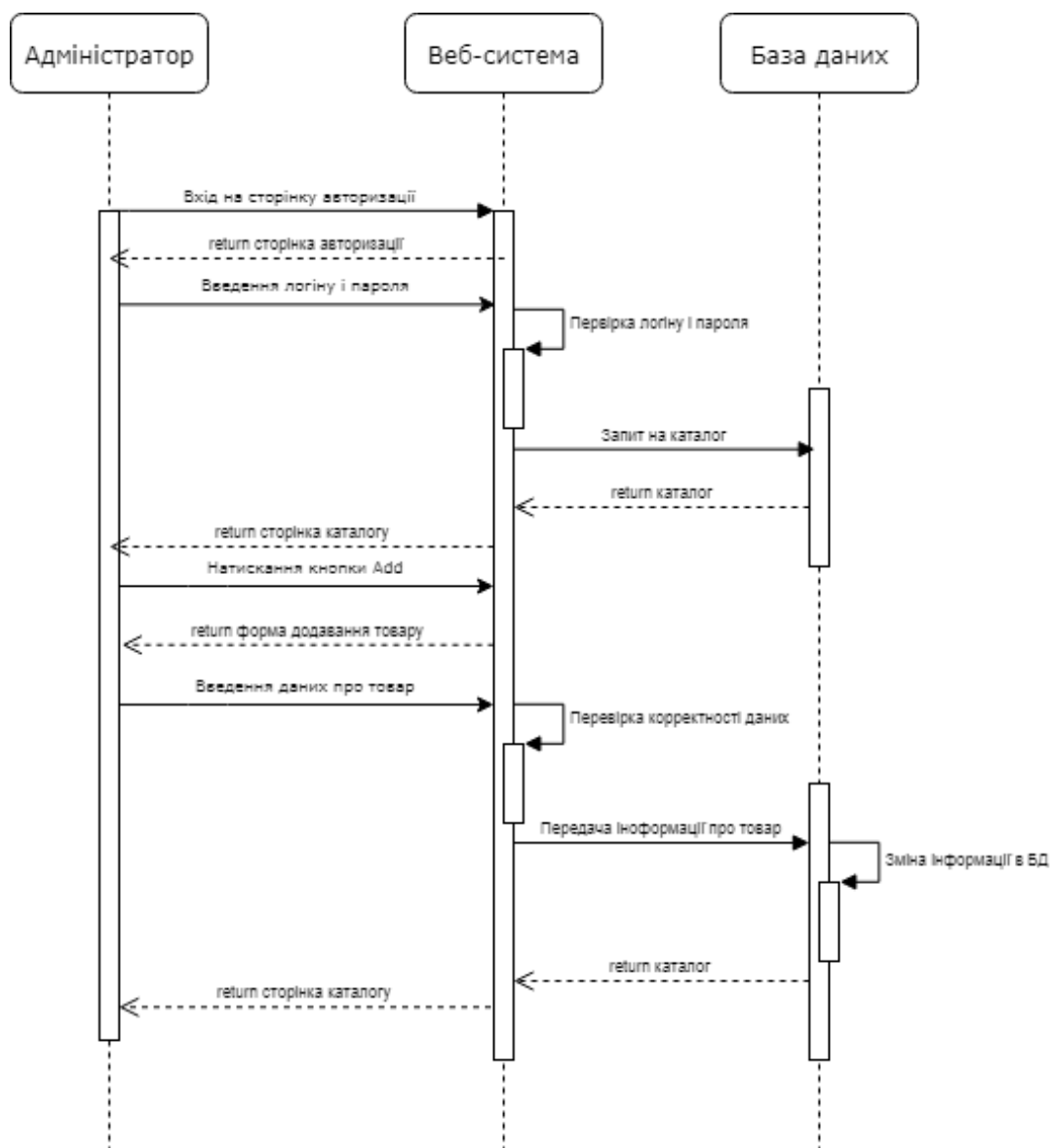


Рисунок 2.3 – UML-діаграма послідовності «Додавання товару»

2.2.1.2.3 Прецедент «Видалення товару»

На рисунку 2.4 представлена діаграма послідовності для прецеденту «Видалення товару».

Основний виконавець: Адміністратор.

Зацікавлені сторони і їх потреби:

- адміністратор (хоче видалити товар);
- підприємство (хоче мати актуальний каталог товарів інтернет-магазину).

Результати (постумови): адміністратор видалив товар, підприємство має актуальний каталог товарів інтернет-магазину.

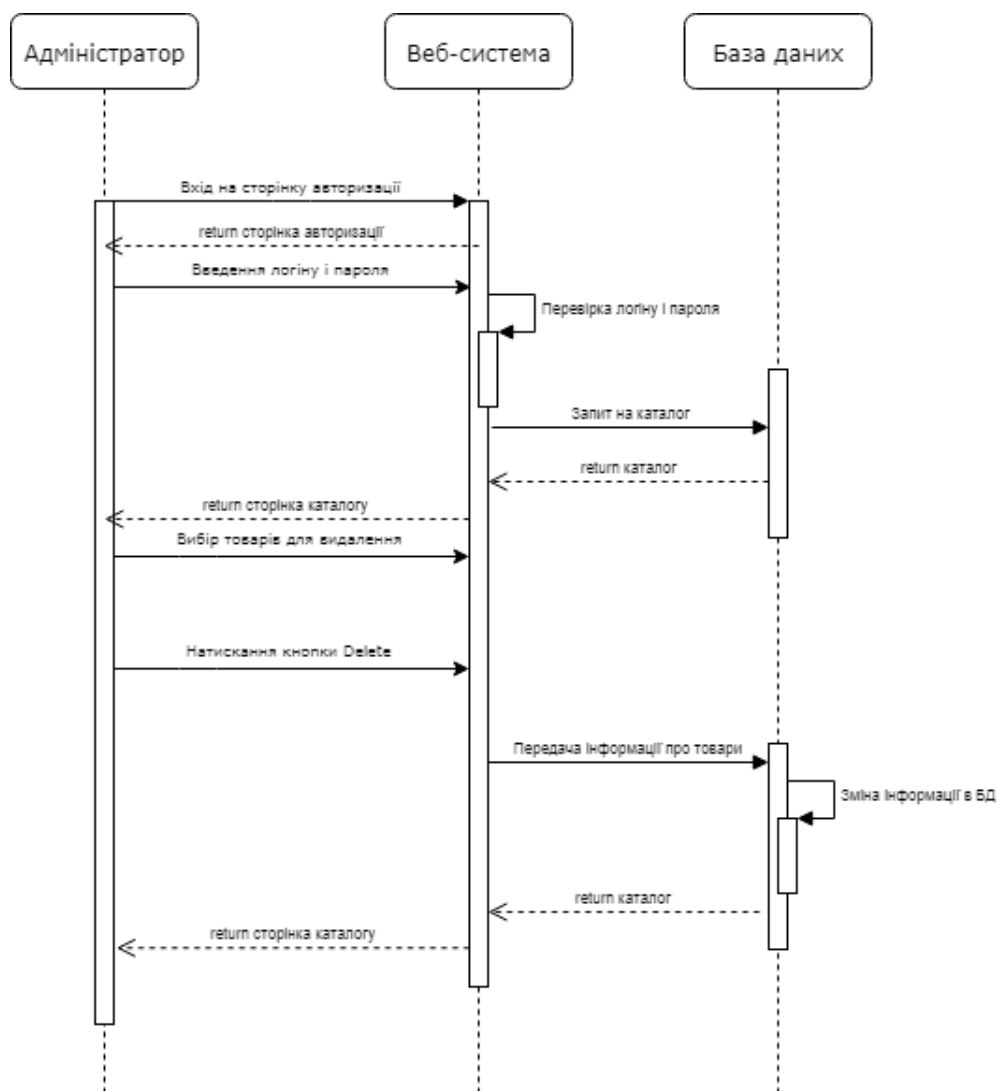


Рисунок 2.4 – UML-діаграма послідовності «Видалення товару»

Розглянемо основний потік (основний успішний сценарій):

- адміністратор відкриває сторінку авторизації;
- адміністратор вводить логін і пароль;
- вебсистема виконує контроль введення адміністратора;
- вебсистема відправляє запит в БД на каталог;
- БД повертає каталог;
- на сторінці сайту відображається каталог;
- адміністратор обирає товари для видалення;
- адміністратор натискає кнопку «Delete»;
- вебсистема передає зміни в каталозі товарів БД;

- БД здійснює відповідні зміни інформації;
- БД повертає каталог;
- на сторінці сайту відображається каталог.

Розглянемо альтернативні потоки (розширення).

В основному потоку «вебсистема виконує контроль введення адміністратора» вказані некоректні дані:

- вебсистема інформує адміністратора, про поля з некоректними даними та повідомляє його про необхідність виправити їх;
- адміністратор виправляє дані;
- система здійснює перевірку даних.

2.3 Архітектура програмного забезпечення

2.3.1 Загальна архітектура програмного продукту

Інтернет-магазин працює на клієнт-серверній архітектурі. Клієнт-серверна архітектура - обчислювальна або мережева архітектура, в якій завдання або мережева навантаження розподілені між постачальниками послуг, що називаються серверами, і замовниками послуг, що називаються клієнтами. Фактично клієнт і сервер - це програмне забезпечення [5]. Зазвичай ці програми розташовані на різних обчислювальних машинах і взаємодіють між собою через обчислювальну мережу за допомогою мережевих протоколів, але вони можуть бути розташовані також і на одній машині. Програми-сервери очікують від клієнтських програм запити і надають їм свої ресурси у вигляді даних або у вигляді сервісних функцій. Оскільки одна програма-сервер може виконувати запити від безлічі програм-клієнтів, її розміщують на спеціально виділеній обчислювальній машині, налаштованої особливим чином, як правило, спільно з іншими програмами-серверами, тому продуктивність цієї машини повинна бути високою. Через особливу роль такої машини в мережі, специфіки її обладнання та

програмного забезпечення, її також називають сервером, а машини, які виконують клієнтські програми, відповідно, клієнтами.

Перевагами даної архітектури є: відсутність дублювання коду програми-сервера програмами-клієнтами; знижені вимоги до клієнтських комп'ютерів, тому що всі обчислення виконуються на сервері; всі дані зберігаються на сервері, який, як правило, захищений набагато краще за більшість клієнтів і дозволяє організовувати контроль повноважень, щоб вирішувати доступ до даних тільки клієнтам з відповідними правами доступу.

Недоліками даної архітектури є: непрацездатність сервера може зробити непрацездатною всю обчислювальну мережу; висока вартість обладнання.

2.3.2 Архітектура реалізації програмного продукту

Програмна реалізація системи складається з шести підсистем.



Рисунок 2.5 – Архітектура реалізації Інтернет-магазину

2.3.2.1 Підсистема користуvalьницького інтерфейсу

Користувальницький інтерфейс отримує команди і дані від користувача. Підсистема необхідна для більш зручного і доступного управління Інтернет-магазином. Вона є інтерфейсом між користувачами і самою системою.

Інтерфейс взаємодіє з більшістю підсистем інтернет-магазину. Підсистема користуvalьницького інтерфейсу пов'язана з підсистемою авторизації по керуванню і за даними. Вона передає підсистемі авторизації логін і пароль, введені адміністратором, і передає команду про початок авторизації.

Підсистема користуvalьницького інтерфейсу пов'язана з підсистемою розподілу доступу по керуванню і за даними. Залежно від рівня доступу користувача (адміністратор або звичайний користувач) підсистема розподілу доступу віддає команди відкриття доступу до деяких дій, можливим в системі для адміністратора.

Підсистема користуvalьницького інтерфейсу пов'язана з підсистемою управління даними в БД по керуванню і за даними. Підсистема управління даними в БД передає користуvalьницькому інтерфейсу записи з бази даних, які той вимагає, також додає записи, отримані від користуvalьницького інтерфейса.

2.3.2.2 Підсистема авторизації

Підсистема авторизації дозволяє зайти в модуль адміністрування інтернет-магазину. Підсистема зв'язується з базою даних для отримання та перевірки введених логіна і пароля.

Підсистема авторизації пов'язана з підсистемою розподілу доступу за даними. Вона передає рівень доступу користувача підсистемі розподілу доступу.

Підсистема авторизації пов'язана з підсистемою захисту введення по керуванню. Підсистема захисту введення контролює правильність введених логіна і пароля.

2.3.2.3 Підсистема розподілу доступу

Підсистема розподілу доступу формує набір можливостей користувача в залежності від його рівня доступу.

Підсистема розподілу доступу пов'язана з підсистемою управління даними по керуванню. Залежно від рівня доступу користувача підсистема розподілу доступу визначає, з якими даними працюватиме підсистема управління даними в базі даних.

2.3.2.4 Підсистема управління даними в БД

Підсистема управління даними в БД дозволяє переглядати, змінювати і видаляти дані в БД.

Підсистема управління даними в БД пов'язана з підсистемою управління записами за даними і керуванням. Підсистема управління даними в БД визначає, які дані і в яку таблицю бази даних підсистема управління записами повинна додати або змінити.

2.3.2.5 Підсистема управління записами

Підсистема управління записами виконує додавання, зміну та видалення записів в базі даних.

2.3.2.6 Підсистема захисту введення

Підсистема захисту введення призначена для контролю користувальницького введення (клієнтського та адміністративного).

2.4 Проєктування бази даних

В результаті проєктування повинна бути визначена структура бази, тобто склад таблиць, їх структура і логічні зв'язки.

2.4.1 Концептуальне моделювання БД

На концептуальному рівні здійснюється інтегрований опис предметної області, для якої розробляється БД, незалежно від її сприйняття окремими користувачами і способів реалізації в комп'ютерній системі [6]. Предметна область - частина дійсного світу, для якої здійснюється концептуальне моделювання. Концептуальна модель предметної області - це формальне

зображення сукупності думок, які характеризують можливі стани предметної області, а також переходи з одного стану в інший (включаючи класифікацію наявних в предметній області сутностей, чинних правил, законів, обмежень).

У даній БД використовується 3 сутності. А саме:

- customers (SID, Name, CCN, Country, Address, Email, Sum, Visittime);
- orders (SID, PID, Product, Order, Price, Visittime);
- pricelist (PID, Product, Warehouse, Bcost).

На підставі складених сутностей визначаємо зв'язки між ними. Між сутностями «customers» і «orders» зв'язок М: М, т. як в таблиці «customers» може повторюватися один і той же SID в межах однієї і тієї ж сесії роботи в браузері, і в цю сесію може бути здійснена покупка багатьох товарів.

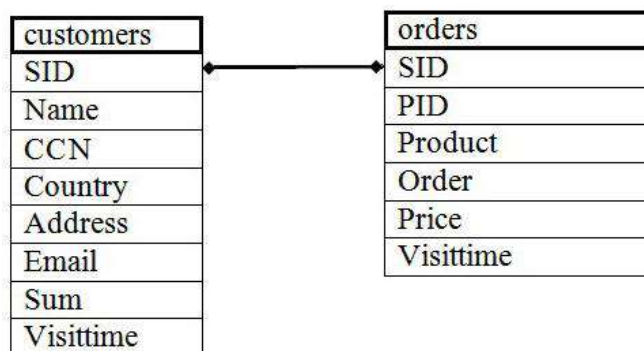


Рисунок 2.6 – Зв'язок між сутностями «customers» і «orders»

Між сутностями «pricelist» і «orders» зв'язок 1: М, так як в таблиці «pricelist» не можуть повторятися товари з одним і тим же PID, але ці товари можна замовити безліч разів.

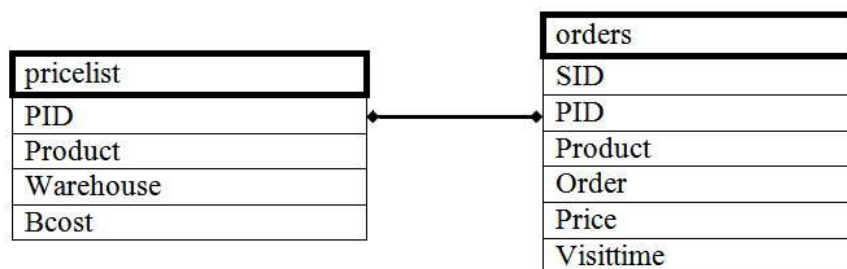


Рисунок 2.7 – Зв'язок між сутностями «pricelist» і «orders»

В ході концептуального моделювання отримали наступну ER-модель (рис. 2.8)

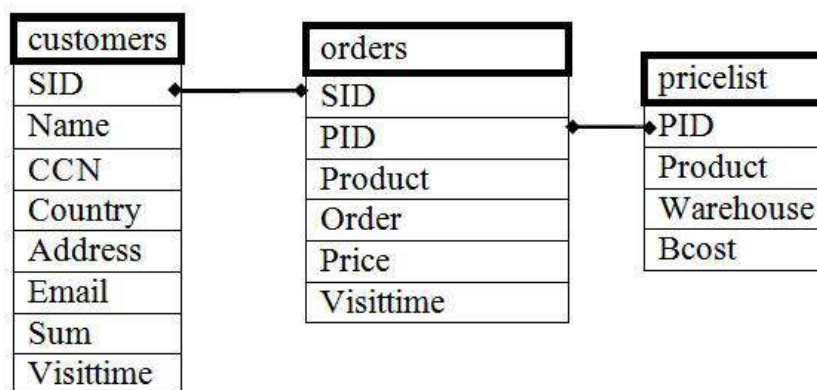


Рисунок 2.8 – ER-модель

2.4.2 Логічне проєктування бази даних

Логічне проєктування - створення схеми бази даних на основі конкретної моделі даних, наприклад, реляційної моделі даних. Для реляційної моделі даних логічна модель це набір схем відносин, зазвичай із зазначенням первинних ключів, а також «зв'язків» між відносинами, які представляють собою зовнішні ключі.

Перетворення концептуальної моделі в логічну модель, як правило, здійснюється за формальними правилами. Цей етап може бути в значній мірі автоматизований.

На етапі логічного проєктування враховується специфіка конкретної моделі даних, але може не враховуватися специфіка конкретної СУБД.

При проєктуванні логічної структури реляційної бази даних визначається оптимальний склад таблиць для зберігання вихідної інформації. Для кожної таблиці вказується її назва, перелік полів і первинний ключ. Зв'язки ідентифікуються між таблицями.

В рамках логічного проєктування БД може формулюватися обмеження цілісності, приймаються рішення про створення індексів і т.д.

Таблиця 2.2 – Структура сутності «pricelist»

Поле	Тип даних	Атрибути	Обов'язковість	Примітка
PID	smallint(5)	unsigned	Так	key
Product	text		Так	
Warehouse	smallint(5)	unsigned	Так	
Bcost	float	unsigned	Так	

Таблиця 2.3 – Структура сутності «customers»

Поле	Тип данных	Атрибути	Обов'язковість	Примітка
SID	varchar(32)		Так	key
Name	Text		Так	
CCN	Text		Так	
Country	set('Ukraine', 'Russia', 'United Kingdom', 'USA')		Так	
Address	Longtext		Так	
Email	Text		Так	
Sum	Float	unsigned	Так	
Visittime	Datetime		Так	

Таблиця 2.4 – Структура сутності «orders»

Поле	Тип даних	Атрибути	Обов'язковість	Примітка
SID	varchar(32)		Так	key
PID	smallint(5)		Так	
Product	text		Так	
Order	smallint(5)	unsigned	Так	
Price	float	unsigned	Так	
Visittime	datetime		Так	

2.5 Проектування інтерфейсу користувача

Головна сторінка містить елемент інтерфейсу navbar що містить список сторінок Інтернет-магазину, елемент інтерфейсу carousel що містить зображення товарів, області для інформації про товари магазину й елемент інтерфейсу footer що містить інформацію про Інтернет-магазин і його розробника.

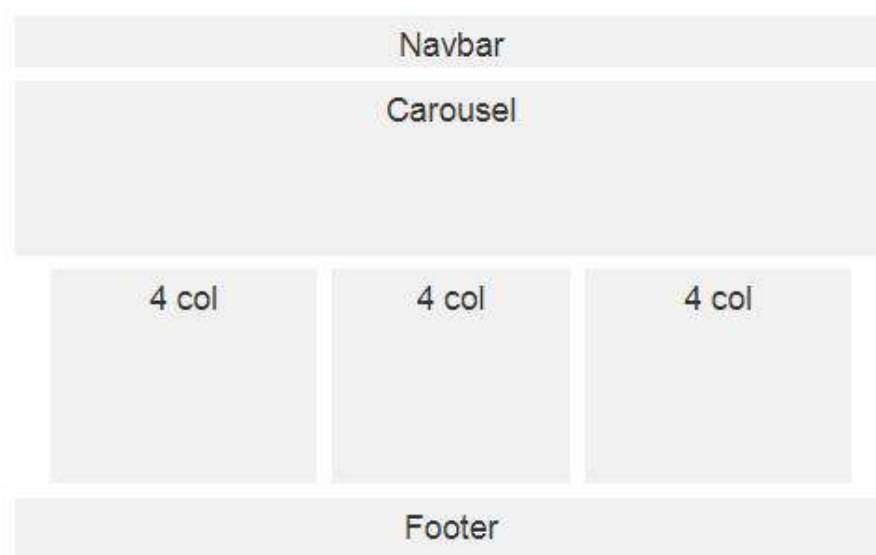


Рисунок 2.9 – Макет головної сторінки

Сторінка каталогу містить логотип магазину і підпис сторінки каталогу. У центрі сторінки розташований сам каталог у вигляді таблиці що містить дані по товару. Над каталогом розташовані елементи керування для сортування та фільтрації вмісту каталогу. Під каталогом розташовані елементи управління для підтвердження і скидання замовлення.



Рисунок 2.10 – Макет сторінки каталогу

Сторінка опису товарів містить елемент інтерфейсу navbar що містить список сторінок інтернет-магазину. Інформація про товари занесена в табличні блоки - один містить зображення товару, інший коротку характеристику та посилання на сайт виробника.

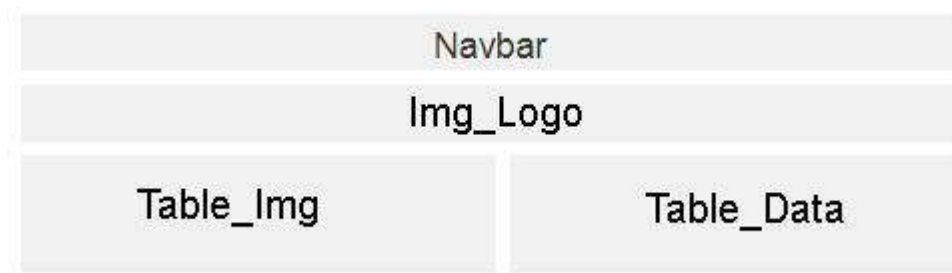


Рисунок 2.11 – Макет сторінки опису товарів

3 ОБГРУНТУВАННЯ ВИБОРУ ПРОГРАМНИХ ЗАСОБІВ

На сьогоднішній день існує величезна кількість засобів для проектування і розробки вебсайтів, тому в даному розділі буде обгрунтований вибір програмних засобів, які необхідні для розробки Інтернет-магазину.

3.1 Клієнтська частина сайту

Клієнтська частина сайту реалізує користувацький інтерфейс, формує запити до сервера і обробляє відповіді від нього. Для її реалізації використовують наступні технології:

- HTML – стандартизована мова розмітки документів у мережі Інтернет [7];
- CSS – формальна мова опису зовнішнього вигляду документа, написаного з використанням мови розмітки [8];
- JavaScript – мова сценаріїв для надання інтерактивності вебсторінкам [9].

HTML разом із CSS та JavaScript – це три основні технології побудови вебсторінок, що є стандартизованими технологіями представлення вебсайтів в Інтернеті і не мають аналогів. Однак, існують бібліотеки і фреймворки, такі як jQuery і Bootstrap, які використовують ці технології для полегшення розробки та

об'єднання різних компонентів великого програмного проєкту, такого як Інтернет-магазин. Фреймворк Bootstrap і бібліотека jQuery мають велику кількість аналогів, що не відрізняються функціоналом, а тільки лише синтаксисом написання операцій, тому що в їх основі лежать стандартизовані технології HTML, CSS, і JavaScript, що обумовлює недоцільність їх порівняння.

3.1.1 Мова HTML

HTML (англ. HyperText Markup Language) – стандартна мова розмітки вебсторінок в Інтернеті. Більшість вебсторінок створюються за допомогою мови HTML (або XHTML). Документ HTML оброблюється браузером та відтворюється на екрані у звичному для людини вигляді.

HTML є похідною мовою від SGML, успадкувавши від неї визначення типу документа та ідеологію структурної розмітки тексту.

Попри те, що HTML – штучна комп'ютерна мова, вона не є мовою програмування.

HTML впроваджує засоби для:

- створення структурованого документа шляхом позначення структурного складу тексту: заголовки, абзаци, списки, таблиці, цитати та інше;
- отримання інформації із Всесвітньої мережі через гіперпосилання;
- створення інтерактивних форм;
- включення зображень, звуку, відео, та інших об'єктів до тексту.

3.1.2 Мова CSS

CSS (англ. Cascading Style Sheets) – спеціальна мова, що використовується для опису зовнішнього вигляду сторінок, написаних мовами розмітки даних.

CSS використовується авторами та відвідувачами вебсторінок, щоб визначити кольори, шрифти, верстку та інші аспекти вигляду сторінки. Одна з головних переваг – можливість розділити зміст сторінки від вигляду документу (що описується в CSS).

Таке розділення може покращити сприйняття та доступність контенту, забезпечити більшу гнучкість та контроль за відображенням контенту в різних умовах, зробити контент більш структурованим та простим, прибрати повтори і

прискорити завантаження сторінок і зменшити обсяг інформації, що передається, та навантаження на сервер та канал передачі, що досягається за рахунок того, що сучасні браузері здатні кешувати інформацію про стилі і використовувати її для всіх сторінок, а не завантажувати для кожної. CSS також дозволяє адаптувати контент до різних умов відображення (на екрані монітора, мобільного пристрою (КПК), у роздрукованому вигляді, на екрані телевізора, пристроях з підтримкою шрифту Брайля або голосових браузерів та ін.).

CSS має різні рівні та профілі. Один і той самий HTML або XML документ може бути відображений по-різному залежно від використаного CSS. Наступний рівень CSS створюється на основі попередніх, додаючи нову функціональність або розширюючи вже наявні функції. Рівні позначаються як CSS1, CSS2 та CSS3.

Профілі CSS – сукупність правил CSS одного або більше рівнів, створені для окремих типів пристроїв або інтерфейсів. Наприклад, існують профілі CSS для принтерів, мобільних пристроїв тощо.

Стандарт CSS визначає порядок та діапазон застосування стилів, тобто, в якій послідовності і для яких елементів застосовуються стилі. Таким чином, використовується принцип каскадності, коли для елементів вказується лише та інформація про стилі, що змінилася або не визначена загальнішими стилями.

Існують такі стилі для відображення сторінки:

- стилі автора (зовнішні або внутрішні таблиці стилів чи стилі окремого елемента);
- стилі користувача (локальний css-файл, вказаний користувачем для використання на сторінках і вказаний в налаштуваннях браузера);
- стилі браузера (стандартний стиль переглядача, що використовується коли немає інформації про стиль елемента або вона неповна).

Специфікації CSS були створені та розвиваються Консорціумом Всесвітньої мережі.

3.1.3 Мова JavaScript

JavaScript – мультіпарадигменна мова програмування. Підтримує об'єктно-орієнтований, імперативний і функціональний стилі. Є реалізацією мови ECMAScript.

JavaScript зазвичай використовується як вбудована мова для програмного доступу до об'єктів додатків. Найбільш широке застосування знаходить в браузерях як мова сценаріїв для надання інтерактивності вебсторінкам.

Основними архітектурними рисами JavaScript є динамічна типізація, автоматичне керування пам'яттю, прототипне програмування і представлення функцій як об'єктів першого класу.

Структурно JavaScript можна представити у вигляді об'єднання трьох чітко окремих одна від одної частин

- ядро (ECMAScript);
- об'єктна модель браузера (Browser Object Model);
- об'єктна модель документа (Document Object Model).

ECMAScript не є браузерною мовою і в ньому не визначаються методи введення і виведення інформації. Це, скоріше, основа для побудови скриптових мов. Специфікація ECMAScript описує типи даних, інструкції, ключові і зарезервовані слова, оператори, об'єкти, регулярні вирази, не обмежуючи авторів похідних мов в розширенні їх новими складовими.

Об'єктна модель браузера – браузер-специфічна частина мови, що є прошарком між ядром і об'єктною моделлю документа. Основне призначення об'єктної моделі браузера – управління вікнами браузера і забезпечення їх взаємодії. Крім управління вікнами, в рамках об'єктної моделі браузера, браузерами зазвичай забезпечується підтримка наступних сутностей:

- керування фреймами;
- підтримка затримки у виконанні коду і зациклення із затримкою;
- системні діалоги;
- керування адресою відкритої сторінки;
- керування інформацією про браузер;

- керування інформацією про параметри монітора;
- обмежене керування історією перегляду сторінок;
- підтримка роботи з HTTP cookie.

Об'єктна модель документа – інтерфейс програмування додатків для HTML і XML-документів. Відповідно до неї, документ (наприклад, вебсторінка) може бути представлений у вигляді дерева об'єктів, що володіють рядом властивостей, які дозволяють здійснювати з ним різні маніпуляції:

- генерація і додавання вузлів;
- отримання вузлів;
- зміна вузлів;
- зміна зв'язків між вузлами;
- видалення вузлів.

3.1.4 Фреймворк Bootstrap

Bootstrap – це один з найпопулярніших безкоштовних наборів інструментів з відкритим кодом, який містить шаблони CSS та HTML для тексту, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript [10]. Цей фреймворк використовується сайтами NASA та MSNBC.

Bootstrap – це вебфреймворк, який зосереджений на спрощенні розробки інформативних динамічних вебсторінок і вебдодатків. Основною метою додавання Bootstrap до вебпроєкту є застосування його кольорів, розмірів, шрифтів та макета до цього проєкту. Після додавання до проєкту Bootstrap надає основні визначення стилів для всіх елементів HTML. Кінцевим результатом є єдиний вигляд тексту, таблиць і елементів форми в усіх веббраузерах. Крім того, розробники можуть скористатися перевагами класів CSS, визначених у Bootstrap, для подальшого налаштування зовнішнього вигляду їхнього вмісту. Bootstrap також поставляється з декількома компонентами JavaScript у вигляді jQuery плагінів. Вони забезпечують додаткові елементи інтерфейсу користувача, такі як діалогові вікна, підказки та каруселі. Кожен компонент Bootstrap складається з HTML-структури, CSS-декларацій, а в деяких випадках і з кодом JavaScript. Вони

також розширюють функціональність деяких існуючих елементів інтерфейсу, включаючи, наприклад, функцію автоматичного заповнення для полів введення.

3.1.5 Бібліотека jQuery

jQuery – бібліотека JavaScript, що фокусується на взаємодії JavaScript і HTML [11]. Бібліотека jQuery допомагає легко отримувати доступ до будь-якого елементу DOM, звертатися до атрибутів і вмісту елементів DOM, маніпулювати ними.

jQuery також надає розробникам можливість створювати плагіни до бібліотеки JavaScript. Це дає змогу розробникам створювати абстракції для низькорівневої взаємодії та анімації, просунутих ефектів і високорівневих тематичних віджетів. Модульний підхід до бібліотеки jQuery дозволяє створювати потужні динамічні вебсторінки та вебпрограми.

Також jQuery ліквідує міжбраузерну несумісність коду JavaScript і забезпечує постійний інтерфейс, який працює в різних браузерах.

3.2 Серверна частина сайту

Серверна частина сайту отримує запит від клієнта, виконує обчислення, після цього формує вебсторінку і відправляє її клієнту по мережі з використанням протоколу HTTP або HTTPS. Крім цього, вона забезпечує збереження призначених для користувача даних та взаємодію зі сторонніми web-сервісами. Існує велика кількість мов для написання серверної частини сайту. У цьому розділі будуть розглянуті одні з найпоширеніших мов, якими є PHP і ASP.NET.

Оскільки розроблюваний програмний продукт є Інтернет-магазином, то необхідно забезпечити можливість оплати товару. Для реалізації можливості оплати товару буде використаний онлайн-сервіс з прийому платежів. Подібні сервіси надають практично однаковий функціонал і відрізняються лише кількістю підтримуємих платіжних систем (в залежності від країни) і тому проводити

порівняння подібних сервісів недоцільно. У якості подібної системи був обраний онлайн-сервіс з прийому платежів на сайті PayMaster, який буде розглянуто в цьому розділі.

3.2.1 Мова PHP

PHP – скриптова мова загального призначення, інтенсивно застосовується для розробки вебдодатків [12]. В даний час підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних вебсайтів.

Мова і його інтерпретатор (Zend Engine) розробляються групою ентузіастів в рамках проєкту з відкритим кодом. Проєкт поширюється під власною ліцензією, несумісною з GNU GPL. PHP простий для освоєння, і в той же час здатний задовольнити запити як професійних програмістів, так і початківців розробників.

PHP-скрипти зазвичай обробляються інтерпретатором в порядку, що забезпечує кросплатформеність розробленого додатка:

- лексичний аналіз вихідного коду і генерація лексем;
- синтаксичний аналіз отриманих лексем;
- генерація байт-коду;
- виконання байт-коду інтерпретатором (без створення виконуваного файлу).

Для збільшення швидкодії додатків можливе використання спеціального програмного забезпечення, так званих акселераторів. Принцип їх роботи полягає в кешуванні одного разу згенерованого байт-коду в пам'яті і / або на диску, таким чином, з процесу роботи програми виключаються етапи 1-3, що в загальному випадку веде до значного прискорення роботи.

Важливою особливістю є те, що розробнику немає необхідності піклуватися про розподіл і звільнення пам'яті. Ядро PHP реалізує засоби для автоматичного керування пам'яттю; вся виділена пам'ять повертається системі після завершення роботи скрипта.

Інтерпретатор складається з ядра і підключаємих модулів, «розширень», що представляють собою динамічні бібліотеки. Розширення дозволяють доповнити

базові можливості мови, надаючи можливості для роботи з базами даних, сокетамі, динамічною графікою, криптографічними бібліотеками, документами формату PDF і тому подібним. Будь-який бажаючий може розробити своє власне розширення і підключити його. Існує величезна кількість розширень, як стандартних, так і створених сторонніми компаніями і ентузіастами, однак в стандартну поставку входить лише кілька десятків добре зарекомендувавших себе. Безліч розширень є в репозиторії PECL.

Інтерпретатор PHP має спеціальний конфігураційний файл – `php.ini`, що містить безліч налаштувань, зміна яких впливає на поведінку інтерпретатора. Є можливість відключити використання ряду функцій, змінити обмеження на використовувану скриптом оперативну пам'ять, час виконання, обсяг завантажуваних файлів, налаштувати журналювання помилок, роботу з сесіями і поштовими сервісами, підключити додаткові розширення, а також багато іншого. Можливе дроблення великого конфігураційного файлу на частини. Наприклад, широко поширена практика винесення налаштувань розширень в окремі файли. Параметри інтерпретатора можуть бути перевизначені в файлах конфігурації HTTP-сервера (наприклад, `.htaccess` в Apache) або в самому скрипті під час виконання за допомогою команди `ini_set`.

В даний час PHP використовується сотнями тисяч розробників. Популярність в області побудови вебсайтів визначається наявністю великого набору вбудованих засобів для розробки вебдодатків. Основні з них:

- автоматичне вилучення POST і GET-параметрів, а також змінних оточення вебсервера в зумовлені масиви;
- взаємодія з великою кількістю різних систем управління базами даних (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape и Apache Derby, Informix, Ovrinos SQL, Lotus Notes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird / InterBase, Paradox File Access, MaxDB, Інтерфейс PDO, Redis);
- автоматизована відправка HTTP-заголовків;
- робота з HTTP-авторизацією;

- робота с cookies і сесіями;
- робота з локальними і віддаленими файлами, сокетами;
- підтримка різноманітних вебсерверів, включаючи Apache і IIS;
- обробка файлів, що завантажуються на сервер.

До найбільших сайтів, які використовують PHP, відносяться Facebook, Wikipedia та ін. Також PHP є основою таких популярних серверних фреймворків, як Yii, WordPress, OpenCart та ін.

3.2.2 Мова ASP.NET

ASP.NET – платформа розробки вебдодатків, до складу якої входить: вебсервіси, програмна інфраструктура, модель програмування, від компанії Microsoft, яка функціонує під управлінням вебсервера Microsoft IIS і поширюється по пропрієтарній ліцензії [13]. ASP.NET входить до складу платформи .NET Framework і є розвитком більш старої технології Microsoft ASP.

ASP.NET зовні багато в чому зберігає схожість з більш старою технологією ASP, що дозволяє розробникам відносно легко перейти на ASP.NET. У той же час внутрішній устрій ASP.NET істотно відрізняється від ASP, оскільки вона заснована на платформі .NET і, отже, використовує всі нові можливості, що надаються цією платформою.

ASP.NET підтримує кілька моделей програмування для створення вебдодатків:

- ASP.NET Web Forms – платформа для побудови модульних сторінок з компонентів з обробкою подій користувацького інтерфейсу на стороні сервера;
- ASP.NET MVC – дозволяє створювати вебсторінки, використовуючи шаблон проєктування модель-представлення-контролер;
- ASP.NET Web Pages – надає спрощений синтаксис для додавання динамічного коду і доступу до даних безпосередньо всередині розмітки HTML;
- ASP.NET Web API – платформа для створення веб-API зверху .NET Framework;

- ASP.NET WebHooks – реалізує шаблон Webhook для підписки і публікації подій через HTTP;
- SignalR – комунікаційне середовище реального часу для двонаправленого зв'язку між клієнтом і сервером.

Оскільки ASP.NET ґрунтується на Common Language Runtime, яка є основою всіх додатків Microsoft .NET, розробники можуть писати код для ASP.NET, використовуючи мови програмування, що входять в комплект .NET Framework (C #, Visual Basic.NET, J # і JScript. NET). ASP.NET має перевагу в швидкості в порівнянні зі скриптовими технологіями, так як при першому зверненні код компілюється і поміщається в спеціальний кеш, і згодом тільки виконується, не вимагаючи витрат часу на парсинг, оптимізацію і т. д.

Програмна модель ASP.NET ґрунтується на протоколі HTTP і використовує його правила взаємодії між сервером і браузером. При формуванні сторінки закладена абстрактна програмна модель Web Forms і на ній заснована основна частина реалізації програмного коду.

.NET використовує технологію рендерингу «відвіданих композитів» («visited composited»). Під час компіляції файл шаблону (.aspx) складається в код ініціалізації, який створює дерево керування (комполит), що представляє вихідний шаблон. Літерний текст переходить в екземпляри класу Literal control, а елементи керування сервером в екземпляри специфічного класу керування. Код ініціалізації поєднується з кодом, написаним користувачем (зазвичай шляхом збірки декількох часткових класів), і переходить до специфічного класу сторінки.

Фактичні запити на сторінку обробляються у кілька кроків. По-перше, під час ініціалізації створюється екземпляр класу сторінки та виконується код ініціалізації. Це створює початкове дерево керування, яке маніпулюється методами сторінки в наступних кроках. Оскільки кожен вузол у дереві являє собою елемент керування, представлений як екземпляр класу, код може змінювати структуру дерева, а також маніпулювати властивостями / методами окремих вузлів. Нарешті, під час етапу візуалізації користувач (відвідувач) має

відвідати кожен вузол у дереві, вимагаючи, щоб кожен вузол сам використовував методи відвідувача. Вихідний HTML надсилається клієнту.

Після того, як запит було оброблено, екземпляр класу сторінки відкидається, а разом з ним і все дерево керування. Це викликає труднощі у програмістів-початківців ASP.NET, які звертаються до членів екземпляру класу, які відкидаються з кожним циклом запиту / відповіді сторінки.

Для розробки даного проєкту була вибрана мова PHP, т. як він підтримується різними вебсерверами, такими як Apache і IIS, є відкритим вільно поширюваним програмним забезпеченням з відкритим вихідним кодом, легший у вивченні, і повністю підходить для реалізації функціоналу Інтернет-магазину.

3.2.3 Онлайн-сервіс для прийняття платежів PayMaster

Онлайн-сервіс для прийняття платежів – вебсервіс, що дозволяє сплачувати за послуги та товари через мережу Інтернет, використовуючи різні способи сплати, такі як: електронні гроші, банківські картки, рахунок мобільного телефона. Робота подібних сервісів регулюється згідно з місцевим законодавством.

Компанія (продавець) в загальному випадку взаємодіє з PayMaster наступним чином:

- продавець генерує (або просить) форму заказа оплати для PayMaster;
- покупець відправляє цю форму на платіжну сторінку PayMaster;
- на сайті PayMaster покупець обирає спосіб платежу;
- продавцю відправляється предзапит на підтвердження рахунку;
- покупець завершує платіж, продавцю приходить сповіщення про платіж.
- покупець повертається на сайт продавця з результатом транзакції.

Взаємодія даного конкретного Інтернет-магазину з системою PayMaster докладно описана у розділі 3 Моделювання програмного забезпечення.

Реальна інтеграція даного вебсервіса з Інтернет-магазином не проводилася, бо для використання цього сервісу необхідно бути реальною юридичною особою, з ліцензією на провдення торговельних операцій, що не є можливим в рамках

виконання дипломного проєкту, проте проведена оптимізація для найбільш простої інтеграції цього сервіса в реальний Інтернет-магазин.

3.3 Система управління базами даних

База даних – представлена в об'єктивній формі сукупність самостійних матеріалів, систематизованих таким чином, щоб ці матеріали могли бути знайдені і оброблені за допомогою комп'ютерів. Вони призначені для задоволення інформаційних потреб користувачів, а в технічному сенсі – управління системами управління. Бази даних для сайтів використовуються з метою зберігання різної інформації і, спрощено, представляють собою деякий набір взаємозв'язаних таблиць. Саме в базах даних зберігається на сервері необхідна для роботи сайту інформація, наприклад, інформація про клієнтів, каталог товарів, статистичні дані і т. д.

Система управління базами даних (СУБД) – це спеціальний пакет програм, який забезпечує створення, супроводження та використання баз даних багатьма користувачами. СУБД забезпечує програмні засоби для створення, завантаження, отримання та поновлення даних, контролює дії, пов'язані з введенням-виведенням даних, вирішує питання спільного їх використання та захисту. СУБД служить для підтримки бази даних в актуальному стані і забезпечує ефективний доступ користувачів до тих даних, які містяться в ній в рамках, наявних у користувачів повноважень і рівнів доступу.

Для порівняння були відібрані SQLite і MySQL, т. як вони володіють найбільш широким функціоналом для виконання поставленого завдання.

3.3.1 СУБД SQLite

SQLite – вбудовувана реляційна система керування базами даних [14]. Втілена у вигляді бібліотеки, де реалізовано багато зі стандарту SQL-92.

SQLite не використовує парадигму клієнт-сервер, тобто движок SQLite не є окремо працюючим процесом, з яким взаємодіє програма, а являє собою бібліотеку, з якої програма компонується, і движок стає складовою частиною програми. Таким чином, в якості протоколу обміну використовуються виклики функцій (API) бібліотеки SQLite. Такий підхід зменшує накладні витрати, час відгуку і спрощує програму. SQLite зберігає всю базу даних (включаючи визначення, таблиці, індекси і дані) в єдиному стандартному файлі на тому комп'ютері, на якому виконується програма. Простота реалізації досягається за рахунок того, що перед початком виконання транзакції записи весь файл, який зберігає базу даних, блокується.

Кілька процесів або потоків можуть одночасно без будь-яких проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, якщо ніяких інших запитів в даний момент не обслуговується; в іншому випадку спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

У комплекті поставки йде також функціональна клієнтська частина у вигляді виконуваного файлу `sqlite3`, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина є кросплатформенною утилітою командного рядка.

Завдяки архітектурі движка можливо використовувати SQLite як на вбудовуваних системах, так і на виділених машинах з гігабайтними масивами даних.

SQLite підтримує динамічну типізацію даних. Можливі типи значень: `INTEGER`, `REAL`, `TEXT` і `BLOB`. Так само підтримується спеціальне значення `NULL`.

Сама бібліотека SQLite написана на C; існує велика кількість прив'язок до інших мов програмування, в тому числі Delphi, C++, Java, C#, VB.NET, Python, Perl, Node.js, PHP, PureBasic, Tcl (засоби для роботи з Tcl включені в комплект поставки SQLite), Ruby, Haskell, Scheme, Smalltalk, Lua і Parser, а також до

багатьох інших. Простота і зручність вбудовування SQLite привели до того, що бібліотека використовується в браузерах, музичних плеєрах і багатьох інших програмах.

3.3.2 СУБД MySQL

MySQL – вільна реляційна система управління базами даних [15]. Розробка та підтримка сайту MySQL здійснює корпорація Oracle. MySQL є рішенням для малих і середніх додатків. Продукт поширюється як під GNU General Public License (безкоштовно для некомерційного використання), так і під власною комерційною ліцензією.

В якості мови запитів, в залежності від версії, використовуються різні версії стандарту SQL, зокрема MySQL 4.0 практично повністю реалізує ANSI SQL-99 з розширеннями, а MySQL 5.0 – SQL-2003. Зазвичай MySQL використовується як сервер, до якого звертаються локальні або видалені клієнти, проте в дистрибутив входить бібліотека внутрішнього сервера, що дозволяє включати MySQL в автономні програми.

Гнучкість СУБД MySQL забезпечується підтримкою великої кількості типів таблиць. Максимальний розмір таблиць в MySQL 3.22 до 4 ГБ, в наступних версіях обмежень немає. Розмір таблиці обмежений її типом. У загальному випадку тип MyISAM (система зберігання даних в СУБД MySQL, яка ґрунтується на принципах ISAM (Indexed Sequential Access Method – індексного-послідовний метод доступу) і володіє в порівнянні з ним низкою корисних доповнень) обмежений граничним розміром файлу в файлової системі операційної системи. У разі InnoDB (підсистема низького рівня для зберігання даних в СУБД MySQL) одна таблиця може зберігатися в декількох файлах, які мають єдине табличний простір. Розмір останнього може досягати 64 терабайт. На відміну від MyISAM, в InnoDB є значне обмеження на кількість стовпців, яке можна додати в одну таблицю. Розмір сторінки пам'яті за замовчуванням становить 16 КБ, з яких під дані відведено 8123 байта.

MySQL має API для мов Delphi, C, C ++, Ейфель, Java, Лисп, Perl, PHP, Python, Ruby, Smalltalk, Компонентний Паскаль і Tcl, бібліотеки для мов

платформи .NET, а також забезпечує підтримку для ODBC за допомогою ODBC Драйвера MyODBC. MyODBC працює на всіх системах Microsoft Windows і на більшості платформ Unix.

Для Oracle MySQL існує додаток MySQL Workbench для візуального проєктування баз даних, в складі якого входять інструменти проєктування, моделювання, створення та експлуатації БД.

MySQL підтримує macOS, -nix-подібні системи і версії Windows починаючи з Windows 95 до Windows 10.

Для розробки була обрана СУБД MySQL т. як вона є клієнт-серверною, має високу швидкість обробки даних, високий ступінь безпеки і зручна для роботи з великими базами даних, а також вже є вбудований функціонал для розробки БД при використанні PHP – PHPMyAdmin, в якій можна створювати й адмініструвати БД MySQL.

3.4 Вебсервер

Вебсервер – сервер, що приймає HTTP-запити від клієнтів, зазвичай веббраузерів, і видає їм HTTP-відповіді, як правило, разом з HTML-сторінкою, зображенням, файлом, медіа-потокком або іншими даними.

Для порівняння були відібрані Apache HTTP Server і Internet Information Server (IIS), т. як. Вони є найбільш широко використовуваними вебсерверами хостингових компаній.

3.4.1 Вебсервер Apache HTTP Server

Apache HTTP Server – вільно поширюваний вебсервер [16]. Apache є кросплатформовим ПО, підтримує операційні системи Linux, BSD, Mac OS, Microsoft Windows, Novell NetWare, BeOS.

Основними перевагами Apache вважаються надійність і гнучкість конфігурації. Він дозволяє підключати зовнішні модулі для надання даних,

використовувати СУБД для аутентифікації користувачів, модифікувати повідомлення про помилки і т. д., а також підтримує IPv6. Ядро Apache включає в себе основні функціональні можливості, такі як обробка конфігураційних файлів, протокол HTTP і система завантаження модулів. У модулях реалізуються такі речі, як: підтримка мов програмування, додавання функцій, виправлення помилок або модифікація основних функцій, посилення безпеки.

У версіях 1.3 і старше був реалізований механізм Server Side Includes, що дозволяє динамічно формувати HTML-документи на стороні сервера. Управління SSI займається модуль `mod_include`, що включений в базову поставку Apache. Адміністратор може встановити власні сторінки і обробники для всіх HTTP помилок і подій, таких як 404 (Not Found) або 403 (Forbidden). У тому числі існує можливість запуску скриптів і відображення повідомлень на різних мовах.

Система конфігурації Apache заснована на текстових конфігураційних файлах. Має три умовних рівня конфігурації:

- Конфігурація сервера (`httpd.conf`).
- Конфігурація віртуального хоста (`httpd.conf` с версії 2.2, `extra/httpd-vhosts.conf`).
- Конфігурація рівня каталогу (`.htaccess`).

Має власну мову конфігураційних файлів, що заснована на блоках директив. Практично всі параметри ядра можуть бути змінені через конфігураційні файли, аж до керування багатопроцесорними моделями.

Apache має різні механізми забезпечення безпеки і розмежування доступу до даних. Основними є:

- обмеження доступу до певних каталогів або файлів;
- механізм авторизації користувачів для доступу до каталогу на основі HTTP-аутентифікації (`mod_auth_basic`) та `digest`-аутентифікації;
- обмеження доступу до певних каталогів або всього серверу, що базується на IP-адресах користувачів;

- заборона доступу до певних типів файлів для всіх або частини користувачів, наприклад, заборона доступу до конфігураційних файлів і файлів баз даних;

- модулі, що реалізують авторизацію через СУБД або PAM (Pluggable Authentication Modules);

Для реалізації шифрування даних, що передаються між клієнтом і сервером, використовується механізм SSL, реалізований через бібліотеку OpenSSL.

Apache має вбудований механізм віртуальних хостів. Він дозволяє повноцінно обслуговувати на одному IP-адресу безліч сайтів (доменних імен), відображаючи для кожного з них власний вміст. Для кожного віртуального хоста можна вказати власні настройки ядра і модулів, обмежити доступ до всього сайту або окремих файлів.

Існує безліч модулів, що додають до Apache підтримку різних мов програмування і систем розробки. До них відносяться: PHP (mod_php), Python (mod_python, mod_wsgi), Ruby (apache-ruby), Perl (mod_perl), ASP (apache-asp), Tcl (rivet).

Крім того, Apache підтримує механізми CGI і FastCGI, що дозволяє виконувати програми на практично всіх мовах програмування, в тому числі C, C++, Lua, sh, Java.

3.4.2 Вебсервер IIS

IIS (Internet Information Server) – пропріетарний набір серверів для декількох служб Інтернету від компанії Microsoft. IIS поширюється з операційними системами сімейства Windows NT [17].

Основним компонентом IIS є вебсервер – служба WWW (яка називається також W3SVC), яка надає клієнтам доступ до сайтів за протоколами HTTP і, якщо проведена настройка, HTTPS.

Один сервер IIS може обслуговувати кілька сайтів (IIS 6.0 і вище). Кожен сайт має наступні атрибути:

- IP-адресу сайту;
- TCP-порт, на якому служба WWW очікує підключень до даного сайту;

– Заголовок вузла – значення заголовку Host-запиту HTTP, що вказує зазвичай DNS-ім'я сайту.

Таким чином, наприклад, один сервер з однією IP-адресою може обслуговувати на одному TCP-порту кілька сайтів. Для цього необхідно створити кілька DNS-записів, що вказують на IP-адресу сервера, і розрізняти сайти за заголовками вузла.

В IIS 6.0, доступному в складі систем Windows Server 2003, служба WWW зазнала серйозні зміни. Був доданий новий режим обробки запитів, званий режимом ізоляції робочих процесів (англ. Worker process isolation mode). У цьому режимі всі вебдодатки, які обслуговуються сервером, працюють в різних процесах, що підвищує стабільність і безпеку системи. Крім того, для прийому запитів HTTP був створений новий драйвер http.sys, який працює в режимі ядра, що прискорює обробку кожного запиту.

Всі запити до статичного вмісту, які не потребують виконання скриптів, виконуються самим драйвером http.sys в ядрі, що зближує вебсервер IIS з серверами режиму ядра.

При цьому запити до динамічного вмісту виконуються робочим процесом і завантаженими в його адресний простір модулями. З точки зору шляху виконання запитів не існує центрального процесу, що підвищує надійність в разі відмови, викликаного помилкою в скрипті чи іншому модулі виконання. Робочі процеси автоматично перезапускає при виникненні помилок.

Протокол SSL підтримується окремим процесом HTTP SSL, який служить мостом між протоколом TCP і драйвером http.sys. Починаючи з версії 6.0 служба WWW підтримує такі методи аутентифікації, тобто визначення особистості користувача по імені і паролі:

- анонімна аутентифікація – визначення особи користувача не виконується;
- базова аутентифікація – ім'я і пароль передаються по мережі відкритим текстом;

- digest-аутентифікація – пароль оброблюється хеш-функцією перед відправленням по мережі, що унеможлиблює його розшифровку у випадку перехоплення злочинцем;

- вбудована аутентифікація Windows – виконується спроба входу на сервер з тими самими обліковими даними, під якими працює браузер користувача;

- аутентифікація з використанням клієнтського сертифікату – для аутентифікації користувач повинен надати SSL-сертифікат.

IIS підтримує протоколи FTP, POP3, SMTP, NNTP.

Вебсервер IIS підтримує кілька різних технологій створення вебдодатків: ASP.NET, ASP, CGI, FastCGI, ISAPI, SSI. Сам сервер підтримує тільки CGI, FastCGI, ISAPI і SSI. Всі інші технології є надбудовами, що працюють через CGI, FastCGI або ISAPI.

За допомогою CGI програми для IIS можуть розроблятися на основі практично будь-яких, в тому числі сторонніх, інструментів, що допускають запис в стандартний потік виведення і читання змінних середовища – Perl, C / C ++ і навіть засобами інтерпретатора командного рядка Cmd.exe.

ISAPI – низькорівнева технологія, аналогічна інтерфейсу модулів Apache, що надає повний доступ до всіх можливостей IIS, можливість розробки вебдодатків в машинному коді і можливість перевизначення частини функцій IIS і додавання до нього функцій, як пов'язаних з генерацією контенту, так і не пов'язаних з цим. Технологія ISAPI дозволяє, з одного боку, створювати спеціальні програми для IIS, що вимагають особливо тісної взаємодії з механізмом сервера, а з іншого боку, є зручною платформою для організації ефективної взаємодії IIS з іншими технологіями розробки вебдодатків – наприклад, PHP і Perl.

В якості локального вебсервера був обраний IIS, тому що він входить в комплект постачання операційної системи Windows, простий в налаштуванні, має всі необхідні функціональні можливості без підключення додаткових модулів і має всі переваги Apache HTTP Server.

4 РОЗРОБКА СИСТЕМИ

4.1 Розробка бази даних

Для створення бази даних необхідно відкрити браузер, перейти в phpMyAdmin і створити нову базу даних «shop». У ній необхідно створити три сутності: «pricelist», «customers», «orders», а також задати зв'язку між ними в автоматично створеної службової сутності pma_relation.

Сутності, створені в phpMyAdmin, атрибути яких встановлені відповідно до визначених в результаті проектування БД параметрами, наведені на рисунках 4.1-4.5.

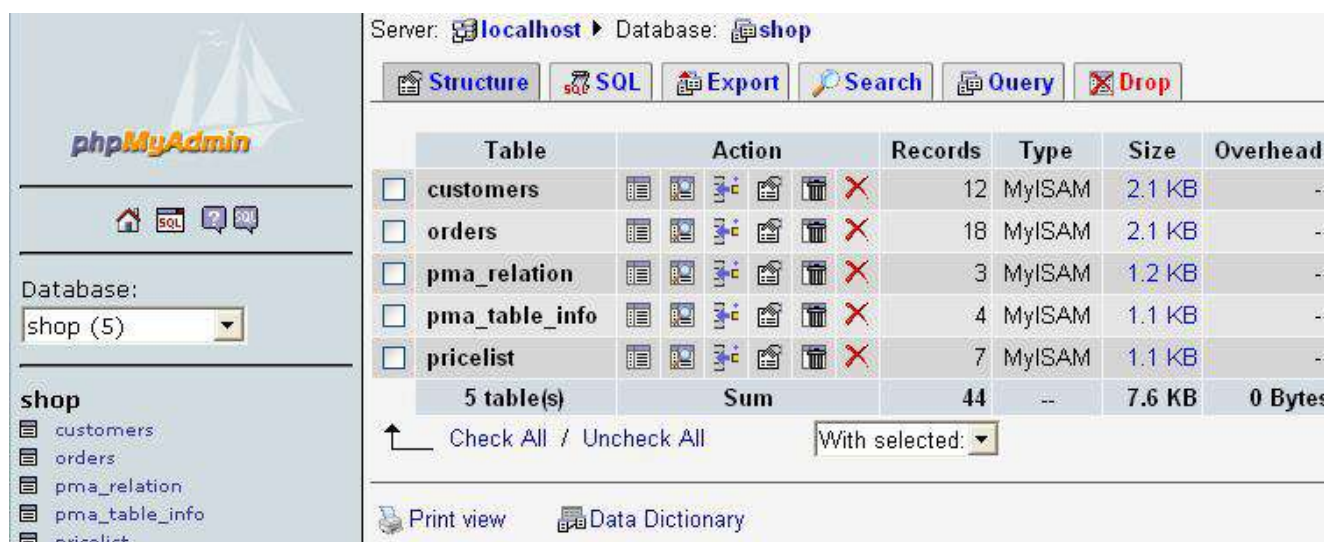


Рисунок 4.1 – Сутності, створені в phpMyAdmin

←T→	master_db	master_table	master_field	foreign_db	foreign_table	foreign_field
☐ [Icons]	shop	customers	SID	shop	orders	SID
☐ [Icons]	shop	orders	PID	shop	pricelist	PID
☐ [Icons]	shop	orders	SID	shop	customers	SID

Рисунок 4.2 – Сутність «pma_relation»

Server: localhost Database: shop Table: pricelist

Structure Browse SQL Search Insert Export Operations

	Field	Type	Attributes	Null	Default	Extra	Action
☐	PID	smallint(5)	UNSIGNED	No	0		      
☐	Product	text		No			      
☐	Warehouse	smallint(5)	UNSIGNED	No	0		      
☐	Bcost	float	UNSIGNED	No	0		      

Рисунок 4.3 – Сутність «pricelist»

Server: localhost Database: shop Table: customers

Structure Browse SQL Search Insert Export Operations Empty Drop

	Field	Type	Attributes	Null	Default	Extra	Action
☐	SID	varchar(32)		No			      
☐	Name	text		No			      
☐	CCN	text		No			      
☐	Country	set('Ukraine', 'Russia', 'United Kingdom', 'USA')		No			      
☐	Address	longtext		No			      
☐	Email	text		No			      
☐	Sum	float	UNSIGNED	No	0		      
☐	Visittime	datetime		No	0000-00-00 00:00:00		      

Рисунок 4.4 – Сутність «customers»

Server: localhost Database: shop Table: orders

Structure Browse SQL Search Insert Export Operations

	Field	Type	Attributes	Null	Default	Extra	Action
☐	SID	varchar(32)		No			      
☐	PID	smallint(5)		No	0		      
☐	Product	text		No			      
☐	Order	smallint(5)	UNSIGNED	No	0		      
☐	Price	float	UNSIGNED	No	0		      
☐	Visittime	datetime		No	0000-00-00 00:00:00		      

Рисунок 4.5 – Сутність «orders»

4.2 Web-розробка

Інтернет-магазин Virtual Airbase-Shop має модульну структуру для ефективного розвитку і масштабованості.

Всі інформаційні матеріали (дані / текст і зображення) і файли сценаріїв винесені в окремі папки. Адміністративна частина сайту відокремлена від користувача частини як програмно, так і фізично (окрема папка).

Інтернет-магазин побудований з урахуванням мінімізації числа переходів користувача між сторінками і SEO-оптимізації.

Налаштування Інтернет-магазину для зручності винесені в окремий конфігураційний файл config.php, вміст якого наведено в лістингу 4.1.

Лістинг 4.1 - Файл config.php

```
<?
    $mysql_servername='MYHOUSE-ED42F23'; // hostname:port -
localhost:3306 or computer name
    $mysql_username='Rostislav'; // MySQL username/login
    $mysql_password=''; // MySQL user password
    $dbname = 'shop';
    $base_path = "c:/Inetpub/wwwroot/shopdb/";
    $inet_path = "http://localhost/Shopdb/";
    //$ordersfile =
$base_path.'data/order_'.$_SESSION['uid'].'.txt';
    $ucartfile = $base_path.'data/cart_'.$_SESSION['uid'].'.txt';
    $pwpage=$inet_path.'admin/pasword.php';
    $adminpage=$inet_path.'admin/admin.php';
    $mainpage=$inet_path.'index.php';
    $cartpage=$inet_path.'cartdb.php';
?>
```

У лістингу A.1 додатку A наведено частину функції функціонального файлу `functions.php`, а саме функції для завантаження бази даних та функції форм додавання і оновлення товарів. Повний вміст функцій динамічної збірки сайту і службових функцій файлу `functions.php`, наведено в лістингу A.2.

Корзина споживача винесена в окремий файл `cartdb.php`, вміст якого наведено в лістингу A.3.

Усе користувальницьке (як клієнтське, так і адміністративне) введення контролюється за допомогою регулярних виразів з бібліотеки PCRE (Perl Compatible Regular Expressions) мови PHP, так як їх синтаксис значно потужніший і гнучкіший, ніж у стандартних регулярних виразів POSIX (Portable Operating System Interface). На важливих етапах користувацького вводу передбачена можливість його уточнення.

У разі виникнення помилок при експлуатації сайту передбачена система обробки помилок (по можливості зі збереженням попереднього користувацького вводу в залежності від правил забезпечення безпеки ресурсу) і інформування про них користувача.

Для доступу в адміністративну частину сайту передбачена авторизація.

В процесі розробки клієнтської і серверної частини сайту були використані засоби зазначені в розділі 3. Результат розробки в формі лістингів наведено в лістингах 4.2, 4.3 та в додатку A.

Лістинг A.2 – Файл `admin/index.php`

```
<?
include("../config.php");
header ('Location : '.$adminpage);
?>
```

Лістинг A.3 – Файл `password.php`

```
<?
session_start();
if (!isset($_SESSION["adminlog"], $_SESSION["idadmin"])) {

    switch(isset($_POST["Button"]) ? $_POST["Button"] :
$_POST["Button"]=="") {
```

```

case "Exit":
    session_unset();
    session_destroy();
    header('Location : '.$mainpage);
break;

default:
?>
<html>
<title>Authorization</title>
<body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
text="#000000" link="#0000FF">
<div align="center">
<form method="post" action="authorize.php">
<blockquote>
    <p>&nbsp;</p>
    <p>
        <INPUT NAME="logad"></p>
    <p> 
        <INPUT TYPE="password" NAME="psword"></p>
    <P> Press SUBMIT to confirm input data.
    <P> Press RESET to clear input data.
    <p><input type="submit" name="Submit" value="Submit" >
&nbsp;<input type="reset" name="Reset" value="Reset"> &nbsp;<input
type="submit" name="Button" value="Exit"></p>
    </blockquote>
    </form>
</div>
</body>
</html>
<? }
}
?>

```

4.3 Тестування користувацького інтерфейсу

Макет головної сторінки сайту, що зображений на рисунку 4.6 був переверстаний в HTML з використанням CSS і jQuery для додання інтерактивності сайту.

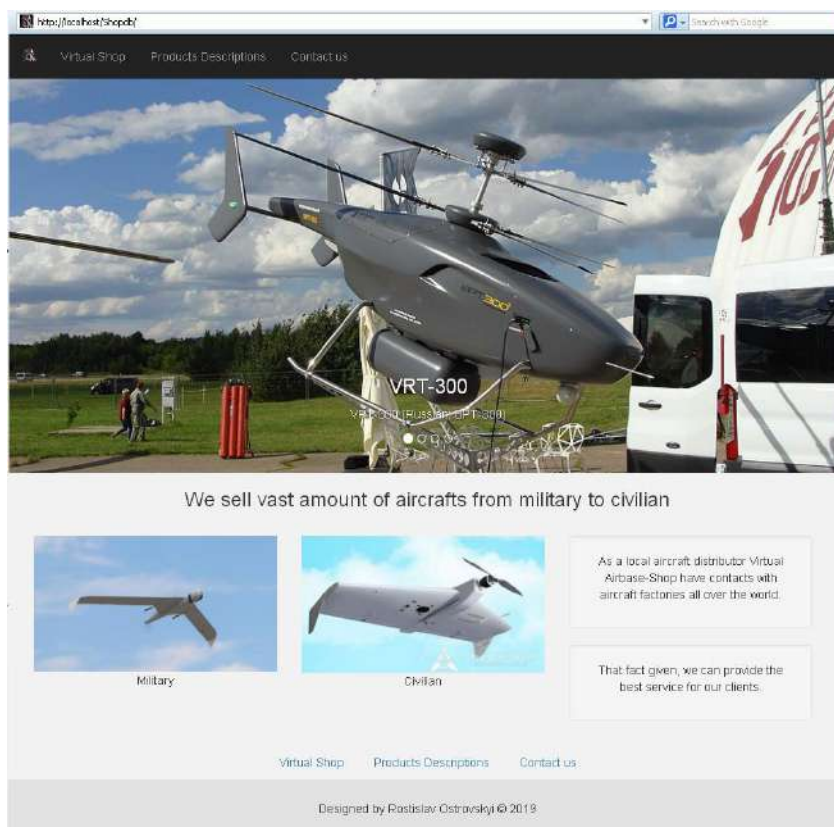


Рисунок 4.6 – Відображення сторінки index.html в браузері

Інші HTML-сторінки і їх розмітка динамічно генеруються в браузері користувача засобами PHP, вони зображені на рисунках 4.7-4.14.



Рисунок 4.7 – Відображення сторінки index.php в браузері

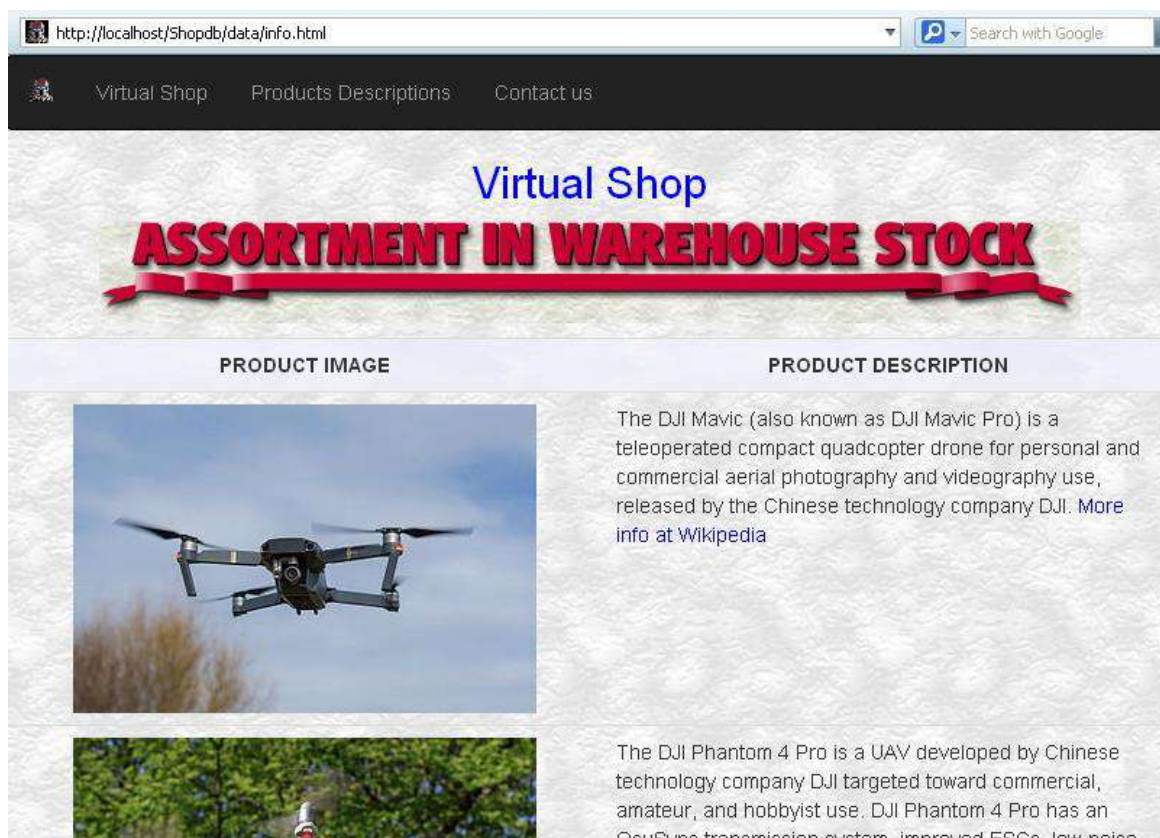


Рисунок 4.8 – Відображення сторінки info.html в браузері

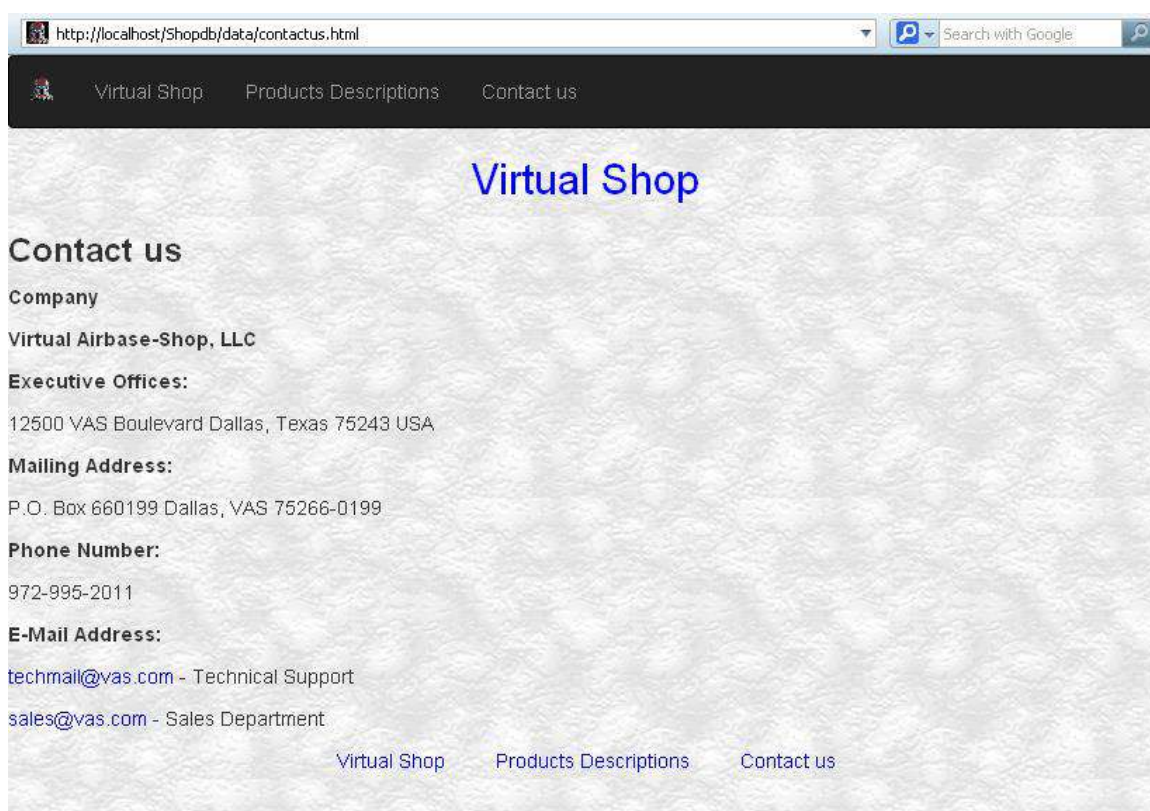


Рисунок 4.9 – Відображення сторінки contactus.html в браузері

Your Order

Name	Warehouse	Quantity	Ordered	Price (EUR)	Remove
VRT-300	49	48	1	3.6	☐

With Checked/Selected:

User List

Your name:
 Choose your country:

Your 16-digit Credit Card Number:
 Payment System provided by

Address:
 E-Mail:

Рисунок 4.10 – Відображення сторінки cartdb.php в браузері

Enter your login

Enter your password

Press SUBMIT to confirm input data.

Press RESET to clear input data.

Рисунок 4.11 – Відображення сторінки pasword.php в браузері

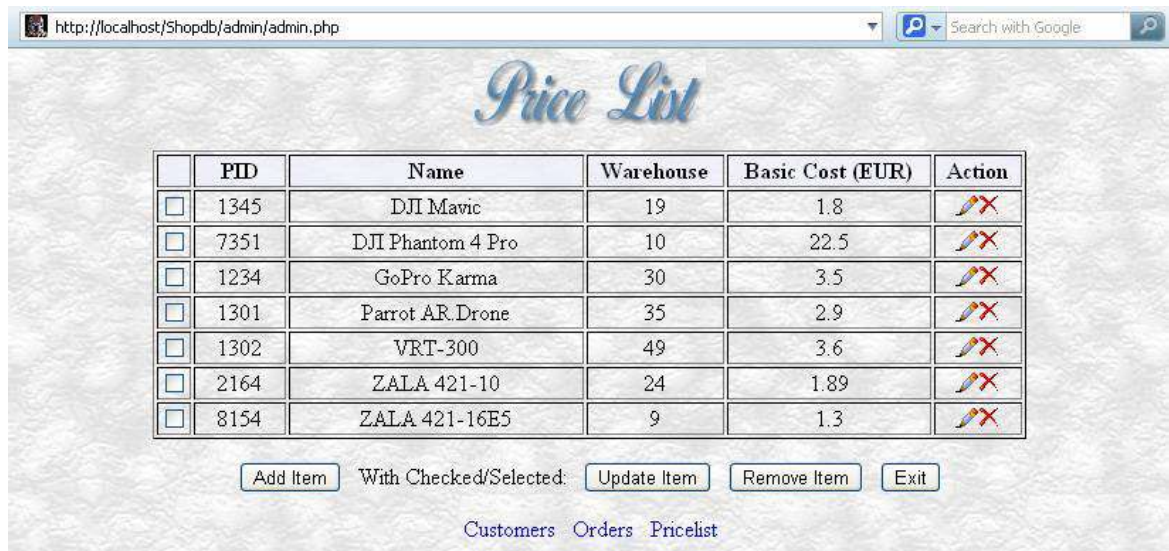
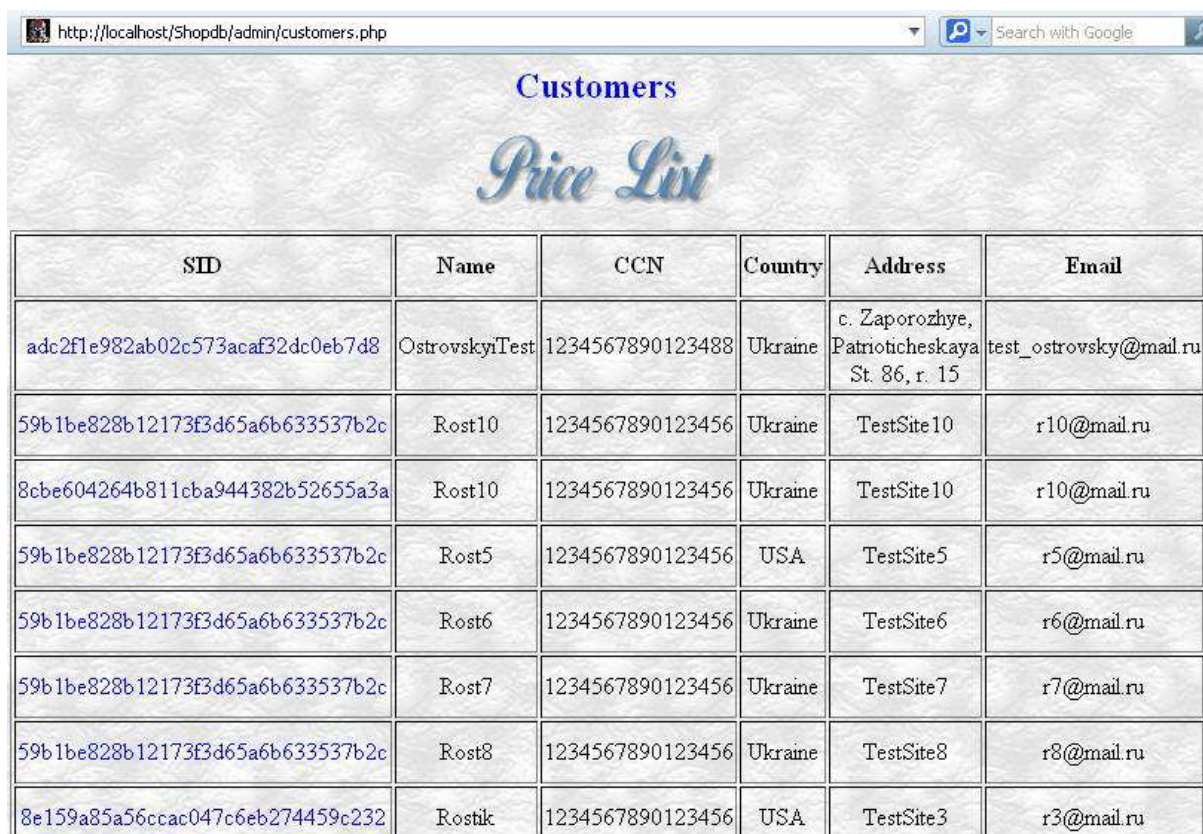


Рисунок 4.12 – Відображення сторінки admin.php в браузері



Рисунок 4.13 – Відображення сторінки orders.php в браузері



http://localhost/Shopdb/admin/customers.php

Search with Google

Customers

Price List

SID	Name	CCN	Country	Address	Email
adc2f1e982ab02c573acaf32dc0eb7d8	OstrovskiyTest	1234567890123488	Ukraine	c. Zaporozhye, Patrioticheskaya St. 86, r. 15	test_ostrovsky@mail.ru
59b1be828b12173f3d65a6b633537b2c	Rost10	1234567890123456	Ukraine	TestSite10	r10@mail.ru
8cbe604264b811cba944382b52655a3a	Rost10	1234567890123456	Ukraine	TestSite10	r10@mail.ru
59b1be828b12173f3d65a6b633537b2c	Rost5	1234567890123456	USA	TestSite5	r5@mail.ru
59b1be828b12173f3d65a6b633537b2c	Rost6	1234567890123456	Ukraine	TestSite6	r6@mail.ru
59b1be828b12173f3d65a6b633537b2c	Rost7	1234567890123456	Ukraine	TestSite7	r7@mail.ru
59b1be828b12173f3d65a6b633537b2c	Rost8	1234567890123456	Ukraine	TestSite8	r8@mail.ru
8e159a85a56ccac047c6eb274459c232	Rostik	1234567890123456	USA	TestSite3	r3@mail.ru

Рисунок 4.14 – Відображення сторінки customers.php в браузері

ВИСНОВКИ

В ході проходження переддипломної практики спроектовано електронний магазин товарів і послуг авіатехнічних засобів, яких відповідає всім вимогам технічного завдання. Були отримані практичні вміння в проєктуванні вебсистеми.

Було проведено аналіз предметної області та виделені основні задачі системи електронного магазину:

- продаж товарів;
- оновлення бази даних товарів;
- обробку заявок клієнтів;
- взаємодію з системою оплати;
- інформування працівників складу про замовлення;
- передачу інформації про замовлення службі доставки.

Розроблено вимоги до програмного продукту та здійснено моделювання програмного забезпечення, в результаті якого складено наступні UML-діаграми.

Проведено проєктування архітектури програмного продукту, бази даних та його інтерфейсу.

Проведено обґрунтування вибору засобів реалізації проєкту Інтернет-магазину. Web-система була створена за допомогою таких технологій: HTML, CSS, JavaScript, jQuery, Bootstrap Framework, MySQL, IIS, PHP. Розроблений інтернет-магазин був протестован та відповідає всім визначеним вимогам.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

- 1) Internet Information Services URL:
https://uk.wikipedia.org/wiki/Internet_Information_Services (дата звернення 10.03.2024).
- 2) Специфікація вимог до програмного забезпечення URL:
https://uk.wikipedia.org/wiki/Специфікація_вимог_до_програмного_забезпечення (дата звернення 10.03.2024).
- 3) Специфікація вимог до програмного забезпечення (SRS): шаблон та підказки URL: <https://visuresolutions.com/uk/специфікація-вимог-до-програмного-забезпечення-srs-шаблон-підказки/> (дата звернення 10.03.2024).
- 4) Unified Modeling Language URL:
https://uk.wikipedia.org/wiki/Unified_Modeling_Language (дата звернення 10.03.2024).
- 5) Клієнт-серверна архітектура URL: https://uk.wikipedia.org/wiki/Клієнт-серверна_архітектура (дата звернення: 10.03.2024).
- 6) Етап концептуального проектування URL:
<https://studfile.net/preview/7543084/page:3/> (дата звернення: 10.03.2024).
- 7) HTML URL: <https://uk.wikipedia.org/wiki/HTML> (дата звернення: 10.03.2024).
- 8) CSS URL: <https://uk.wikipedia.org/wiki/CSS> (дата звернення: 10.03.2024).
- 9) JavaScript URL: <https://uk.wikipedia.org/wiki/JavaScript> (дата звернення: 10.03.2024).
- 10) Bootstrap URL: <https://uk.wikipedia.org/wiki/Bootstrap> (дата звернення 10.03.2024).
- 11) JQuery URL: <https://uk.wikipedia.org/wiki/JQuery> (дата звернення 10.03.2024).
- 12) PHP URL: <https://uk.wikipedia.org/wiki/PHP> (дата звернення: 10.03.2024).

13) ASP.NET URL: <https://uk.wikipedia.org/wiki/ASP.NET> (дата звернення: 10.03.2024).

14) SQLite URL: <https://uk.wikipedia.org/wiki/SQLite> (дата звернення 10.03.2024).

15) СКБД MySQL і доступ до БД в PHP URL: <http://www.znannya.org/?view=mysql-intro> (дата звернення: 10.03.2024).

16) Apache HTTP Server URL: https://uk.wikipedia.org/wiki/Apache_HTTP_Server (дата звернення 10.03.2024).

17) What Is IIS Server? URL: <https://www.solarwinds.com/resources/it-glossary/iis-server> (дата звернення 10.03.2024).

Програмна реалізація

Лістинг А.1 – Функції завантаження БД і функції форм додавання та оновлення товарів

```

<?
function dbload($dbn)
{
    global $load_tabledata, $db_link, $mysql_servername,
$mysql_username, $mysql_password;

    $db_link=mysql_connect($mysql_servername,$mysql_username,$mysql
_password) or die ("Couldn't connect");
    mysql_select_db($dbn,$db_link) or die ("Couldn't set
$dbname : " .mysql_error());
    $table_query="Select Product, Warehouse, Bcost, PID From
pricelist ORDER BY Product ASC";
    $load_tabledata=mysql_query($table_query,$db_link) or die
("Query Failed");
}
function show_forms($format)
{
    global $load_tabledata, $db_link;
    switch ($format) {

        case "Add":
            ?>
                <body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
text="#000000" link="#0000FF" vlink="#0000FF" alink="#0000FF">
                <FORM METHOD="POST" ACTION="add_item_test.php">
                    <div align="center">
                        <p> PID <input name="nPID" size="8"
maxlength="4" value="<? if (isset($_POST["nPID"])) echo
$_POST["nPID"];?>">
                        Product <input name="nProd" size="30"
maxlength="30" value="<? if (isset($_POST["nProd"])) echo
$_POST["nProd"];?>">
                        Quantity <input name="nKolvo" size="8" maxlength="4"
value="<? if (isset($_POST["nKolvo"])) echo $_POST["nKolvo"];?>">
                        Basic Cost <input name="nBcost" size="8" maxlength="6"
value="<? if (isset($_POST["nBcost"])) echo
$_POST["nBcost"];?>"></p>

                        <p><INPUT TYPE=SUBMIT name="Button" value="Submit">

&nbsp;
                        <INPUT TYPE=RESET></p>

                    </div>

```

```

        </FORM>
    </body>
    <?
    break;

    case "Upd":
    ?>
    <body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
text="#000000" link="#0000FF" vlink="#0000FF" alink="#0000FF">
    <FORM METHOD="POST" ACTION="add_item_test.php"></blockquote>
    <?
    global $i;
    if (isset($load_tabledata)) {
        while ($row = mysql_fetch_array($load_tabledata,
MYSQL_ASSOC)) { $rows[]=$row; }
    }
    if (isset($rows)) {
        session_register("rows","uprow","fmethod");
        $_SESSION["rows"]=$rows; // sohranyaetsya kol-vo vibrannih
ryadov
        if (isset($_GET["slrow"])) {
            $_SESSION["uprow"]=$_GET["slrow"]; // v
$_SESSION["uprow"] hranitsya ID ryada vibrannogo cherez $_GET
            $_SESSION["fmethod"]="get"; // zapominaetsya method
obrabotki formi
        } else { $_SESSION["uprow"]=$_POST["selrow"]; }
    }

    $crows=count($_SESSION["rows"]);
    for ($i=0; $i<=($crows-1); $i++) {
        if (!isset($_GET["slrow"])) {
            if ($_SESSION["fmethod"]!="get") {
                foreach ($_SESSION["uprow"] as $upr) {
                    if
($upr==$_SESSION["rows"][$i]["PID"]) {
                        uform();
                    }
                }
            } else if
($_SESSION["uprow"]==$_SESSION["rows"][$i]["PID"]) {
                uform();
            }
        } else if
($_SESSION["uprow"]==$_SESSION["rows"][$i]["PID"]) {
            uform();
        }
        if (($crows-1)==0) { break; }
    }
    ?>
    <div align="center">
    <p><font color="#0066CC">Press UPDATE to confirm
changes.</font></p>

```

```

        <p><font color="#0066CC">Press RESET to clear input
data.</font></p>
        <p><INPUT TYPE="submit" name="Button" value="Update">
&nbsp; <INPUT TYPE=RESET></p>
        </div>
    </blockquote>
</FORM>
<?
break;
}
}

function uform()
{
global $i;
?>
<div align="center">
    <p> PID <input name="outputPID[]" disabled Value="<? if
(isset($_POST["uprow"][$i])) { echo $_POST["uprow"][$i]; } else {
echo $_SESSION["rows"][$i]["PID"]; } ?>">
        Product <input name="nProd[]" size="30" maxlength="30"
value="<? if (isset($_POST["nProd"][$i])) { echo
$_POST["nProd"][$i]; } else { echo $_SESSION["rows"][$i]["Product"];
}?>">
        Quantity <input name="nKolvo[]" size="8" maxlength="4"
value="<? if (isset($_POST["nKolvo"][$i])) { echo
$_POST["nKolvo"][$i]; } else { echo
$_SESSION["rows"][$i]["Warehouse"]; }?>">
        Basic Cost <input name="nBcost[]" size="8" maxlength="6"
value="<? if (isset($_POST["nBcost"][$i])) { echo
$_POST["nBcost"][$i]; } else { echo $_SESSION["rows"][$i]["Bcost"];
}?>">
        <input type="hidden" name="uprow[]" Value="<? if
(isset($_POST["uprow"][$i])) { echo $_POST["uprow"][$i]; } else {
echo $_SESSION["rows"][$i]["PID"]; } ?>">
    </p>
</div>
<?
}

```

Лістинг А.2 – Файл functions.php

```

<?
function dbload($dbn)
{
    global $load_tabledata, $db_link, $mysql_servername,
$mysql_username, $mysql_password;

    $db_link=mysql_connect($mysql_servername,$mysql_username,$mysql
_password) or die ("Couldn't connect");
    mysql_select_db($dbn,$db_link) or die ("Couldn't set
$dbname : " .mysql_error());
}

```

```

        $table_query="Select Product, Warehouse, Bcost, PID From
pricelist ORDER BY Product ASC";
        $load_tabledata=mysql_query($table_query,$db_link) or die
("Query Failed");
    }
    function show_headers($title)
    {
    ?>
        <html>
    <?

        switch ($title) {

    case "1":
    ?>
        <head>
        <title>Virtual Shop</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <meta name="Keywords" content="Aircraft, Aviation,
Civilian, Military, Shop, Buy, Antonov, Kalashnikov, Mikoyan,
Ruslan, Sukhoi, Tupolev, An, MiG, Su, Tu, DJI, GoPro, Parrot, UAV,
VRT, ZALA, Ан, МиГ, Су, Ту, БЛА, БПЛА, ВРТ, Зала">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        <script language="JavaScript" src="scripts/crcdom.js"
type="text/javascript"></script>
        <STYLE type="text/css">
        /* domCollapse styles */
        .trigger{
            cursor:hand;
            color:#990000;
        }
        .triggerhover{
            cursor:hand;
            color:#990000;
            text-decoration: underline;
        }
        .triggerexpanded {
            cursor:hand;
            color:#990000;
            text-decoration: underline;
        }
        .elementnormal{
            position:static;
            left:0;
        }
        .elementcollapsed{
            position:absolute;
            left:-999em;
            top:-5000;
            height:1px;

```

```

    }
    #domcollapseall ul{
        margin:0;
        padding:0;
    }
    #domcollapseall li{
        display:inline;
        padding-right:1em;
    }
</STYLE>
</head>
<body bgcolor="#FFFFFF" background="skins/Relief.jpg"
link="#0000FF">
    <?
    break;

    case "2":
    ?>
        <head>
        <title>Cart</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

        <?
        break;

    case "3":
    ?>
        <head>
        <title>Admin Page</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

        <?
        break;

    case "4":
    ?>
        <head>
        <title>Add Item</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

```

```

    <?
    break;

    case "5":
    ?>
        <head>
        <title>Customers</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

    <?
    break;

    case "6":
    ?>
        <head>
        <title>Orders</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

    <?
    break;

    case "7":
    ?>
        <head>
        <title>Change Item Data</title>
        <meta http-equiv="Content-Type" content="text/html;
charset=windows-1251">
        <meta http-equiv="expires" content="0">
        <LINK REL="SHORTCUT ICON"
HREF="http://localhost/Shopdb/IPC.ico">
        </head>

    <?
    break;
    }
}
function show_forms($formt)
{
global $load_tabledata, $db_link;
    switch ($formt) {

        case "Add":
    ?>
        <body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
text="#000000" link="#0000FF" vlink="#0000FF" alink="#0000FF">
        <FORM METHOD="POST" ACTION="add_item_test.php">

```

```

        <div align="center">
            <p> PID <input name="nPID" size="8"
maxlength="4" value="<? if (isset($_POST["nPID"])) echo
$_POST["nPID"];?>">
                Product <input name="nProd" size="30"
maxlength="30" value="<? if (isset($_POST["nProd"])) echo
$_POST["nProd"];?>">
                    Quantity <input name="nKolvo" size="8" maxlength="4"
value="<? if (isset($_POST["nKolvo"])) echo $_POST["nKolvo"];?>">
                        Basic Cost <input name="nBcost" size="8" maxlength="6"
value="<? if (isset($_POST["nBcost"])) echo
$_POST["nBcost"];?>"></p>

            <p><INPUT TYPE=SUBMIT name="Button" value="Submit">

&nbsp;

            <INPUT TYPE=RESET></p>

        </div>

    </FORM>
</body>
<?
break;

    case "Upd":
        ?>
        <body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
text="#000000" link="#0000FF" vlink="#0000FF" alink="#0000FF">
        <FORM METHOD="POST" ACTION="add_item_test.php"></blockquote>
        <?
        global $i;
        if (isset($load_tabledata)) {
            while ($row = mysql_fetch_array($load_tabledata,
MYSQL_ASSOC)) { $rows[]=$row; }
        }
        if (isset($rows)) {
            session_register("rows","uprow","fmethod");
            $_SESSION["rows"]=$rows; // sohranyaetsya kol-vo vibrannih
ryadov
            if (isset($_GET["slrow"])) {
                $_SESSION["uprow"]=$_GET["slrow"]; // v
$_SESSION["uprow"] hranitsya ID ryada vibrannogo cherez $_GET
                $_SESSION["fmethod"]="get"; // zapominaetsya method
obrabotki formi
            } else { $_SESSION["uprow"]=$_POST["selrow"]; }
        }

        $crows=count($_SESSION["rows"]);
        for ($i=0; $i<=($crows-1); $i++) {
            if (!isset($_GET["slrow"])) {
                if ($_SESSION["fmethod"]!="get") {
                    foreach ($_SESSION["uprow"] as $upr) {

```

```

                                if
($upr==$_SESSION["rows"][$i]["PID"]) {
                                uform();
                                }
                                }
                                } else if
($ _SESSION["uprow"]==$_SESSION["rows"][$i]["PID"]) {
                                uform();
                                }
                                } else if
($ _SESSION["uprow"]==$_SESSION["rows"][$i]["PID"]) {
                                uform();
                                }
                                if (($crows-1)==0) { break; }
                                }
?>
    <div align="center">
    <p><font color="#0066CC">Press UPDATE to confirm
changes.</font></p>
    <p><font color="#0066CC">Press RESET to clear input
data.</font></p>
    <p><INPUT TYPE="submit" name="Button" value="Update">
&ampnbsp <INPUT TYPE=RESET></p>
    </div>
</blockquote>
</FORM>
<?
break;
}
}

function uform()
{
global $i;
?>
<div align="center">
    <p> PID <input name="outputPID[]" disabled Value="<? if
(isset($_POST["uprow"][$i])) { echo $_POST["uprow"][$i]; } else {
echo $_SESSION["rows"][$i]["PID"]; } ?>">
    Product <input name="nProd[]" size="30" maxlength="30"
value="<? if (isset($_POST["nProd"][$i])) { echo
$_POST["nProd"][$i]; } else { echo $_SESSION["rows"][$i]["Product"];
}?>">
    Quantity <input name="nKolvo[]" size="8" maxlength="4"
value="<? if (isset($_POST["nKolvo"][$i])) { echo
$_POST["nKolvo"][$i]; } else { echo
$_SESSION["rows"][$i]["Warehouse"]; }?>">
    Basic Cost <input name="nBcost[]" size="8" maxlength="6"
value="<? if (isset($_POST["nBcost"][$i])) { echo
$_POST["nBcost"][$i]; } else { echo $_SESSION["rows"][$i]["Bcost"];
}?>">

```

```

        <input type="hidden" name="uprow[]" Value="<? if
(isset($_POST["uprow"][$i])) { echo $_POST["uprow"][$i]; } else {
echo $_SESSION["rows"][$i]["PID"]; } ?>">
    </p>
</div>
<?
}

function show_table($letter,$sort_coldir)
{
    global $db_link,$load_tabledata;
    $sort_coldir_array=explode(" by ",$sort_coldir); //
    $sort_coldir_array[0] -> sort_column, $sort_coldir_array[1] ->
    sort_direction
    if ($letter!="Full Table"){
        if ($sort_coldir_array[0]=="Product"){
            $sort_filter_query="SELECT Product, Warehouse, Bcost, PID
FROM pricelist WHERE {$sort_coldir_array[0]} REGEXP '^[{$letter}]'
ORDER BY {$sort_coldir_array[0]} {$sort_coldir_array[1]}";
            $load_tabledata=mysql_query($sort_filter_query,$db_link) or
die ("Sort-Filter Query Failed");
        }
        if ($sort_coldir_array[0]=="Bcost"){
            $sort_filter_query="SELECT Product, Warehouse, Bcost, PID
FROM pricelist WHERE Product REGEXP '^[{$letter}]' ORDER BY
{$sort_coldir_array[0]} {$sort_coldir_array[1]}";
            $load_tabledata=mysql_query($sort_filter_query,$db_link) or
die ("Sort-Filter Query Failed");
        }
    } else {
        $sort_filter_query="SELECT Product, Warehouse, Bcost, PID
FROM pricelist ORDER BY {$sort_coldir_array[0]}
{$sort_coldir_array[1]}";
        $load_tabledata=mysql_query($sort_filter_query,$db_link) or
die ("Sort-Filter Query Failed");
    }
    if (mysql_num_rows($load_tabledata)!=0) {
        ?>
        <div align="center"><p></p></div>
        <div align="center">
            <p></p>
            <FORM METHOD="POST" ACTION="index.php">
            <p align="right"><font color="#0066CC">Filter Product </font>
            <SELECT name="abc">
                <OPTION <? if ((isset($_POST["abc"])) &&
($_POST["abc"]=="Full Table")) { ?>selected<? } ?>>Full
Table</option>
                <OPTION <? if ((isset($_POST["abc"])) && ($_POST["abc"]=="0-
9")) { ?>selected<? } ?>>0-9</option>
                <OPTION <? if ((isset($_POST["abc"])) && ($_POST["abc"]=="A-
L")) { ?>selected<? } ?>>A-L</option>

```

```

        <OPTION <? if ((isset($_POST["abc"])) && ($_POST["abc"]=="M-
T")) { ?>selected<? } ?>>M-T</option>
        <OPTION <? if ((isset($_POST["abc"])) && ($_POST["abc"]=="U-
Z")) { ?>selected<? } ?>>U-Z</option>
    </SELECT>
    &nbsp; <font color="#0066CC"> Sort by </font>
    <SELECT name="sort">
        <OPTION <? if ((isset($_POST["sort"])) &&
($_POST["sort"]=="Product by ASC")) { ?>selected<? } ?>>Product by
ASC</option>
        <OPTION <? if ((isset($_POST["sort"])) &&
($_POST["sort"]=="Product by DESC")) { ?>selected<? } ?>>Product by
DESC</option>
        <OPTION <? if ((isset($_POST["sort"])) &&
($_POST["sort"]=="Bcost by ASC")) { ?>selected<? } ?>>Bcost by
ASC</option>
        <OPTION <? if ((isset($_POST["sort"])) &&
($_POST["sort"]=="Bcost by DESC")) { ?>selected<? } ?>>Bcost by
DESC</option>
    </SELECT>
    &nbsp; <input type="submit" name="Button" value="Select">
</p>

    <table width="620" border="1" align="center" id="Pricel" >
        <tr background="skins/Randomf.jpg">
            <th width="260"><div align="center">Name</div></th>
            <th width="76"><div
align="center">Warehouse</div></th>
            <th width="132"><div align="center">Basic Cost
(EUR)</div></th>
            <th width="132"><div align="center">To Buy</div></th>
        </tr>
        <?
        while ($row = mysql_fetch_array($load_tabledata, MYSQL_ASSOC))
        {
            ?><tr><?
                foreach ($row as $col_key => $col_val) {
                    if ($col_key != 'PID') {
                        if ($col_key == 'Product') {
                            echo "<td><div align='center'
class='trigger'>$col_val</div><div align='center'><a
href='data/info.html#$col_val' target='_top'><img
src='skins/b_docs.png' alt='Info' title='Info'></a></div></td>\n";
                        } else {
                            echo "<td><div
align='center'>$col_val</div></td>\n";
                        }
                    } else {
                        echo "<td><div align='center'><input type='hidden'
name='Id[]' value=$col_val><input name=$col_val maxlength='3'
size='10' value='0'></div></td>\n";
                    }
                }
            }
        }
    }

```

```

?></tr><?
}
mysql_free_result($load_tabledata);
mysql_close($db_link);
?>
</table>
<p>&nbsp;</p>
<p><INPUT TYPE=SUBMIT name="Button" value="Submit"> &nbsp;</p>
<INPUT TYPE=RESET></p>
</FORM>
</div>
<? } else { ?>
<div align="center">
<p></p>
<p>&nbsp;</p>
<p><a href="index.php" target="_top" title="Back to Full
MainPage"><img src='skins/Text_02_Back.jpg' alt='Back to Full
MainPage' title='Back to Full MainPage'></a></p>
</div>
<? } ?>
<script type="text/javascript">domCollapse();</script>
</body>
</html>
<? } ?>

```

Лістинг А.3 – Опис роботи кошику замовлень споживача cartdb.php

```

<?
    session_start();
    include("config.php");
    include("functions.php");
    $fcval=file($ucartfile);
    $nums=count($fcval);
    dbload($dbname);
    // don't use sorting-filtering functions for correct order
    processing or look for example in index.php (case "Submit")
    for ($i=0; $i<=($nums-1); $i++) {
    $fcstr[$i]=explode("|",$fcval[$i]); }

    while ($row = mysql_fetch_array($load_tabledata, MYSQL_NUM)) {
    $rows[]=$row; }
    $numr = count($rows);

    switch(isset($_POST["Button"]) ? $_POST["Button"] :
    $_POST["Button"]=="") {

    case "Update Order":

    $flag=1;
    for ($t=0; $t<=($nums-1); $t++) {
    if (!preg_match("/^[^\d)]/", $_POST["Kolvo"][$t])) {
    for ($i=0; $i<=($numr-1); $i++) {
    if ($fcstr[$t][0]==$rows[$i][3]) {

```

```

        // >= allows to preserve one last item
        if (($_POST["Kolvo"][$t]!=0) &&
($_POST["Kolvo"][$t]>$rows[$i][1])) {
            $flag=0;
            break;
        }
        } else continue;
    }

    } else $flag=0;
}

if ($flag==1) {

    if (file_exists($ucartfile)) {

        unlink($ucartfile);
        $fpnewcart=fopen($ucartfile,"w") or die ("Cannot open
file");

        for ($k=0; $k<=($nums-1); $k++) {
            if ($_POST["Kolvo"][$k]!=0) {
                for($i=0; $i<=($numr-1); $i++) {
                    if ($fcstr[$k][0]==$rows[$i][3]) {

                        $newPrice=bcmul($_POST["Kolvo"][$k],(float)$rows[$i][2],2);
                        $newQuantity=bcsb($rows[$i][1],$_POST["Kolvo"][$k]);
                        $newData =
"{$_POST['PID'][$k]}|{$rows[$i][0]}|{$rows[$i][1]}|{$newQuantity}|{$_
_POST['Kolvo'][$k]}|{$newPrice}|{$rows[$i][2]}\n";
                        fputs($fpnewcart,$newData);
                    } else continue;
                }
            }
        }
        fclose($fpnewcart);
    } else header('Location : '.$mainpage);
}
header('Location : '.$cartpage);

break;

case "Remove Item":

    if (file_exists($ucartfile)) {

        unlink($ucartfile);
        $fpnewcart=fopen($ucartfile,"w") or die ("Cannot open
file");

        $k=1;

```

```

for ($i=0; $i<=($nums-1); $i++) {
if ($_POST["rem"][$i+$k]==1) {
$K++;
} else {
$cart_item_info=$fcval[$i];
fputs($fpnewcart,$cart_item_info); }
}
fclose($fpnewcart);
}
header('Location :'.$cartpage);

break;

case "Confirm Order":

$flag=1;
for ($t=0; $t<=($nums-1); $t++) {
if (!preg_match("/^[^(\d)]/", $_POST["Kolvo"][$t])) {
for ($i=0; $i<=($numr-1); $i++) {
if ($fcstr[$t][0]==$rows[$i][3]) {
// >= allows to preserve one last item
if (( $_POST["Kolvo"][$t]!=0) &&
($_POST["Kolvo"][$t]>$rows[$i][1])) {
$flag=0;
break;
}
} else continue;
}
} else $flag=0;
}

$_POST["Name"]=substr($_POST["Name"],0,30);
$_POST["CCN"]=substr($_POST["CCN"],0,16);

if ((!empty($_POST["Name"])) && (!preg_match("/^[^(\w)|(\x7F-
\xFF)|(\s)]/", $_POST["Name"]))) {
if ((!empty($_POST["CCN"])) && (strlen($_POST["CCN"])==16)
&& (!preg_match("/^[^(\d){16}]/", $_POST["CCN"]))) {
if ((!empty($_POST["Address"])) &&
(!preg_match("/^[^(\w)|(\x7F-\xFF)|(\s)|(\.)(\,)|(\-
)]/", $_POST["Address"]))) {
if ((!empty($_POST["Email"])) &&
(ereg("([[:alnum:]]\.\-]+)(\@[[:alnum:]]\.\-]+\.\+)",
$_POST["Email"]))) {

if ($flag==1) {

$db_link=mysql_connect($mysql_servername,$mysql_username,$m
ysql_password) or die ("Couldn't connect");
mysql_select_db($dbname,$db_link) or die
("Couldn't set $dbname : " .mysql_error());
$newsum=0;
$visitdate=date("Y-m-d H:i:s");

```

```

        for ($k=0; $k<=($nums-1); $k++) {
            if ($_POST["Kolvo"][$k]!=0) {
                for ($i=0; $i<=($numr-1); $i++) {
                    if ($fcstr[$k][0]==$rows[$i][3]) {
                        $newPrice=bcmul($_POST["Kolvo"][$k],(float)$fcstr[$k][6],2);
                        $newsum=bcadd($newsum,$newPrice,2);
                        $newQuantity=bcsub($rows[$i][1],$_POST["Kolvo"][$k]);
                        // $orderinfostr .= "ID: {$fcstr[$k][0]}\nItem Name:
                        {$fcstr[$k][1]}\nQuantity: {$_POST['Kolvo'][$k]}\nPrice: {$newPrice}
                        UAH\n*****\n";
                        $change_query="UPDATE pricelist SET Warehouse =
                        '{$newQuantity}' WHERE PID='{$rows[$i][3]}'";
                        $load_change=mysql_query($change_query,$db_link) or die
                        ("Query change Failed");
                        //mysql_free_result($load_change);
                        $addord_query="INSERT INTO orders VALUES
                        ('{$_SESSION['uid']}', '{$fcstr[$k][0]}', '{$fcstr[$k][1]}', '{$_POST['
                        Kolvo'][$k]}', '{$newPrice}', '{$visitdate}')";
                        $load_order=mysql_query($addord_query,$db_link) or die
                        ("Query order Failed: ".mysql_error());
                        mysql_free_result($load_order);
                    } else continue;
                }
            }
            if ($newsum!=0) {
                $addcust_query="INSERT INTO customers VALUES
                ('{$_SESSION['uid']}', '{$_POST['Name']}', '{$_POST['CCN']}',
                '{$_POST['Country']}', '{$_POST['Address']}', '{$_POST['Email']}',
                '{$newsum}', '{$visitdate}')";
                $load_customer=mysql_query($addcust_query,$db_link) or die
                ("Query customer Failed");
                mysql_free_result($load_customer);
                mysql_close($db_link);
                unlink($ucartfile);
                mail($_POST["Email"],"Virtual Shop Order","You will receive
                your order in flow of an hour");
                session_unset();
                session_destroy();
                header('Location :'.$mainpage);
            } else {
                show_headers(2);
                echo "<h2 align=center><font color=#FF0000>You haven't ordered
                any Items</font></h2>";
            }
            } else {
                show_headers(2);
                echo "<h2 align=center><font color=#FF0000>Invalid Order of
                Item. Please make sure you have made correct Order</font></h2>";
            }
        } else {
            show_headers(2);

```

```

        echo "<h2 align=center><font color=#FF0000>Invalid E-
Mail</font></h2>";
    }
    } else {
        show_headers(2);
        echo "<h2 align=center><font color=#FF0000>Invalid
Address. It is possible you have entered inadmissible
symbol</font></h2>";
    }
    } else {
        show_headers(2);
        echo "<h2 align=center><font color=#FF0000>Invalid 16-
Digit Credit Card Number. It is possible you have entered
inadmissible symbol or incorrect Credit Card Number
length</font></h2>";
    }
    } else {
        show_headers(2);
        echo "<h2 align=center><font color=#FF0000>Invalid Name.
It is possible you have entered inadmissible symbol</font></h2>";
    }

default:
if (filesize($ucartfile)!=0) {
show_headers(2);
?>
<body bgcolor="#FFFFFF" background="skins/Relief.jpg"
text="#000000" link="#0000FF">
<div align="center">
<p></p>
<FORM METHOD="POST" ACTION="cartdb.php">
<table width="660" border="1" align="center" id="Order" >
<tr background="skins/Randomf.jpg">
<th width="216"><div align="center">Name</div></th>
<th width="86"><div
align="center">Warehouse</div></th>
<th width="86"><div align="center">Quantity</div></th>
<th width="96"><div align="center">Ordered</div></th>
<th width="100"><div align="center">Price
(EUR)</div></th>
<th width="56"><div align="center">Remove</div></th>
</tr>
<?
    for ($i=0; $i<=($nums-1); $i++) {

?>
    <tr>
<?
        for ($j=1; $j<=3; $j++) {
?>
            <td><div align="center"><? echo $fcstr[$i][$j];
?></div></td><? } ?>

```

```

        <td><div align="center"><input type="hidden" name="PID[]"
value="<? echo $fcstr[$i][0]; ?>"><input name="Kolvo[]"
maxlength="3" size="6" value="<? echo $fcstr[$i][4]; ?>"></div></td>
        <td><div align="center"><input name="Price[]"
maxlength="6" size="8" value="<? echo $fcstr[$i][5]; ?>"
disabled></div></td>
        <td><div align="center"><input type="hidden" name="rem[]"
value="0" ><input type="checkbox" name="rem[]" value="1"></div></td>
    </tr>
    <?
    }
    global $_POST;
    ?>
</table>
    <p><input type="submit" name="Button" value="Update
Order"> &nbsp; With Checked/Selected: &nbsp; <input type="submit"
name="Button" value="Remove Item"></p>
    <p align="center"></p>
    <p>Your name: <input name="Name" size="48" maxlenght="30"
value="<? if (isset($_POST["Name"])) echo $_POST["Name"]; ?>">&nbsp;
Choose your country
        <SELECT NAME="Country" SIZE=1 >
            <OPTION <? if (isset($_POST["Country"]) &&
($_POST["Country"]=='Ukraine')) { ?>selected<? } ?>> Ukraine
        </OPTION>
            <OPTION <? if (isset($_POST["Country"]) &&
($_POST["Country"]=='Russia')) { ?>selected<? } ?>> Russia </OPTION>
            <OPTION <? if (isset($_POST["Country"]) &&
($_POST["Country"]=='United Kingdom')) { ?>selected<? } ?>> United
Kingdom </OPTION>
            <OPTION <? if (isset($_POST["Country"]) &&
($_POST["Country"]=='USA')) { ?>selected<? } ?>> USA </OPTION>
        </SELECT></p>
    <p align="center">Your 16-digit Credit Card Number: <input
name="CCN" size="32" maxlenght="16" value="<? if
(isset($_POST["CCN"])) echo $_POST["CCN"]; ?>"> &nbsp; Payment
System provided by &nbsp; </p>
    <p> Address:
        <TEXTAREA NAME="Address" COLS=36 ROWS=3
wrap="VIRTUAL"><? if (isset($_POST["Address"])) echo
$_POST["Address"]; ?></TEXTAREA>
        &nbsp; E-Mail : <INPUT NAME="Email" SIZE="42"
value="<? if (isset($_POST["Email"])) echo $_POST["Email"]; ?>"></p>
    <p><input type="submit" name="Button" value="Confirm
Order"></p>
</FORM>
</div>
</body>
</html>
<?
} else {
    unlink($ucartfile);

```

```

        header('Location :'.$mainpage);
    }
    ?>

```

Лістинг А.3 – Файл customers.php

```

<? session_start();

    include ("../config.php");
    include ("../functions.php");

    $db_link=mysql_connect($mysql_servername,$mysql_username,$mysql_password) or die ("Couldn't connect");
    mysql_select_db($dbname,$db_link) or die ("Couldn't set $dbname : " .mysql_error());

    if (isset($_GET["SS"])) {
        $select_SID_query="Select * From customers Where SID='{$_GET['SS']}'";
        $load_tabledata=mysql_query($select_SID_query,$db_link) or die ("Query Failed");
    } else {
        $table_query="Select SID, Name, CCN, Country, Address, Email, Sum, Visittime From customers ORDER BY Name ASC";
        $load_tabledata=mysql_query($table_query,$db_link) or die ("Query Failed");
    }
    show_headers(5);
    ?>
    <body bgcolor="#FFFFFF" background="../skins/Relief.jpg" link="#0000FF" alink="#0000FF" vlink="#0000FF">
    <h2 align=center><font color=#0000FF>Customers</font></h2>
    <div align="center">
        <p></p>
        <table width="620" border="1" align="center" id="Customl">
        <tr background="skins/Randomf.jpg">
            <th width="260"><div align="center">SID</div></th>
            <th width="76"><div align="center">Name</div></th>
            <th width="132"><div align="center">CCN</div></th>
            <th width="132"><div align="center">Country</div></th>
            <th width="132"><div align="center">Address</div></th>
            <th width="132"><div align="center">Email</div></th>
            <th width="132"><div align="center">Sum
            (EUR)</div></th>
            <th width="132"><div align="center">Visit
            Time</div></th>
        </tr>
    <?
        while ($row = mysql_fetch_array($load_tabledata,
        MYSQL_ASSOC)) {

```

```

?><tr><?
    foreach ($row as $col_key => $col_val) {
        if ($col_key != 'SID') {
            echo "<td><div
align='center'>$col_val</div></td>\n";
        } else {
            echo "<td><div align='center'><a
href='orders.php?SS=$col_val'
target='_top'>$col_val</a></div></td>\n";
        }
    }
?></tr>
<? }
    mysql_free_result($load_tabledata);
    mysql_close($db_link);
?>
</table>
<p><a href="customers.php" target="_top">Customers</a> &nbsp;
<a href="orders.php" target="_top">Orders</a> &nbsp; <a
href="Admin.php" target="_top">Pricelist</a></p>
</div>
</body>
</html>

```

Лістинг А.4 – Файл orders.php

```

<? session_start();

include ("../config.php");
include ("../functions.php");

$db_link=mysql_connect($mysql_servername,$mysql_username,$mysql_password) or die ("Couldn't connect");
mysql_select_db($dbname,$db_link) or die ("Couldn't set $dbname : " .mysql_error());

show_headers(6);
?>
<body bgcolor="#FFFFFF" background="../skins/Relief.jpg"
link="#0000FF" alink="#0000FF" vlink="#0000FF">
<h2 align=center><font color=#0000FF>Orders</font></h2>
<div align="center">
<p></p>
<table width="620" border="1" align="center" id="Ordel" >
<tr background="skins/Randomf.jpg">
<th width="260"><div align="center">SID</div></th>
<th width="76"><div align="center">PID</div></th>
<th width="132"><div align="center">Product</div></th>
<th width="132"><div align="center">Order</div></th>
<th width="132"><div align="center">Price</div></th>
<th width="132"><div align="center">Visit
Time</div></th>

```

```

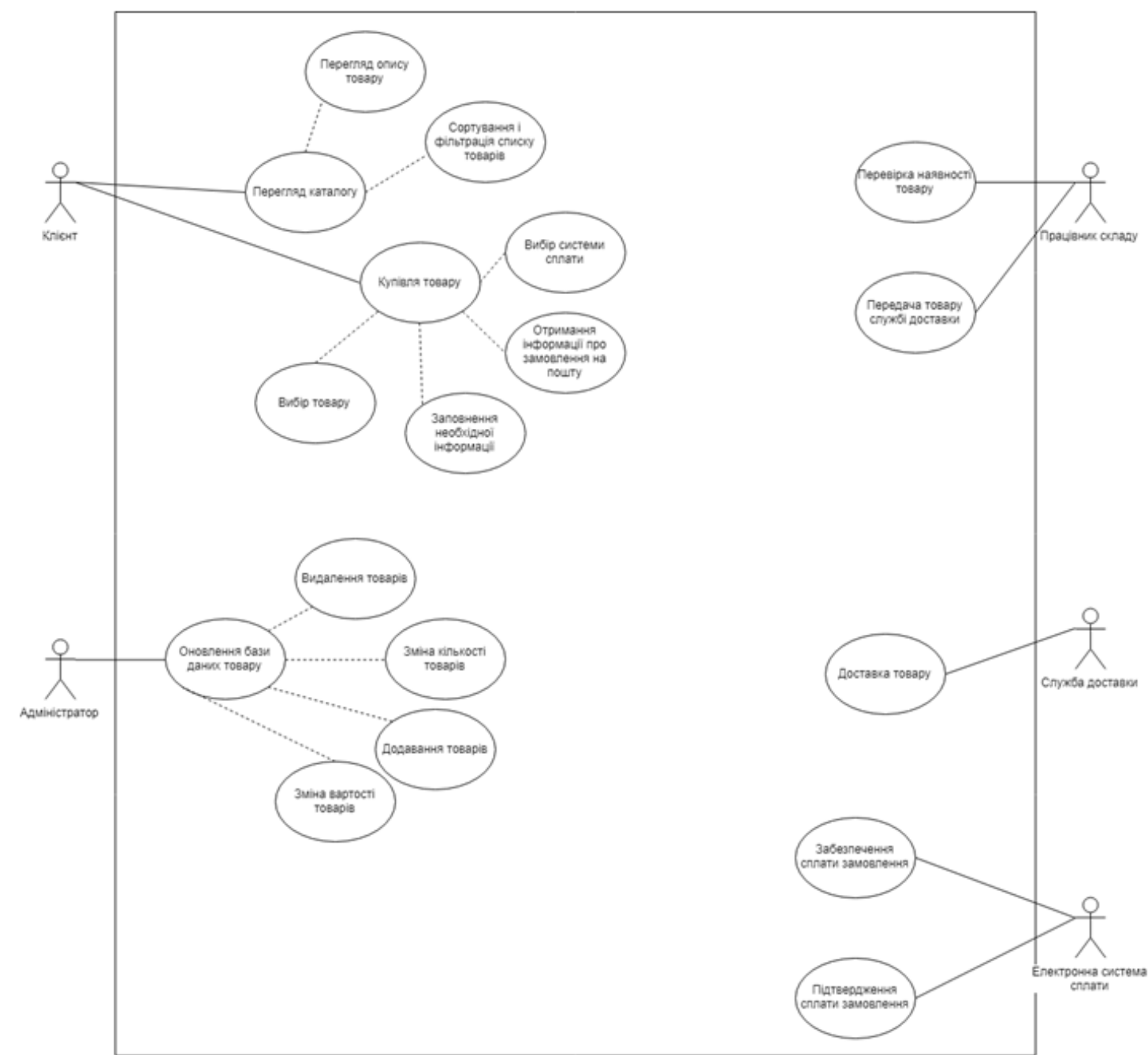
        </tr>
    <?
        if (isset($_GET["SS"])) {
            $select_order_query="SELECT * FROM orders WHERE
SID='{$_GET['SS']}'";
            $load_tabledata=mysql_query($select_order_query,$db_link)
or die ("Query select_order Failed ".mysql_error());
        } else {
            $table_query="Select * From orders ORDER BY Visittime
DESC";
            $load_tabledata=mysql_query($table_query,$db_link) or die
("Query Failed ".mysql_error());
        }

        while ($row = mysql_fetch_array($load_tabledata,
MYSQL_ASSOC)) {
            ?><tr><?
                foreach ($row as $col_key => $col_val) {
                    if ($col_key == 'SID') {
                        echo "<td><div align='center'><a
href='customers.php?SS=$col_val'
target='_top'>$col_val</a></div></td>\n";
                    } else if ($col_key == 'PID') {
                        echo "<td><div align='center'><a
href='Admin.php?PS=$col_val'
target='_top'>$col_val</a></div></td>\n";
                    } else {
                        echo "<td><div
align='center'>$col_val</div></td>\n";
                    }
                }
            ?></tr>
        <? }

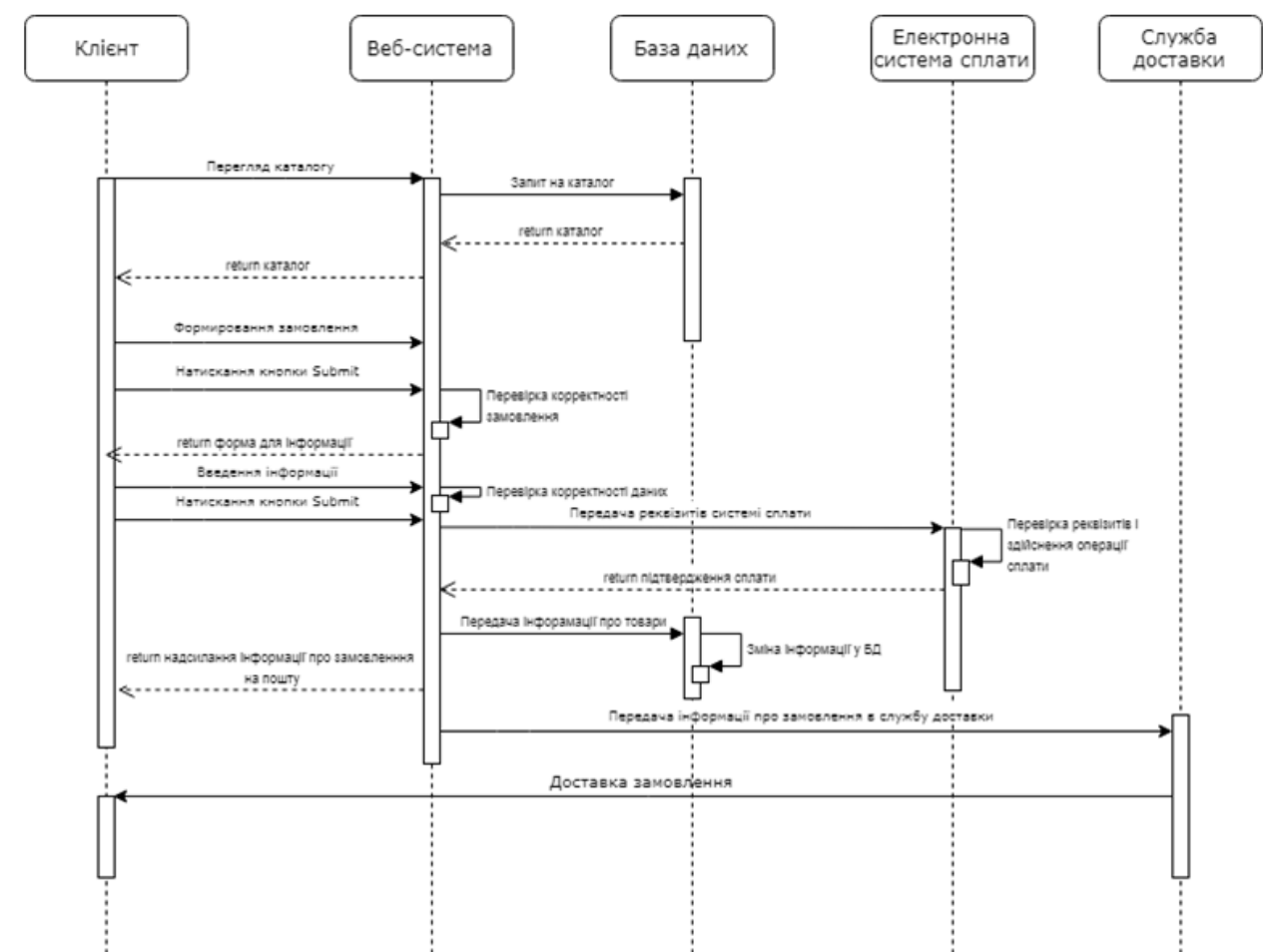
        mysql_free_result($load_tabledata);
        mysql_close($db_link);
    ?>
</table>
<p><a href="customers.php" target="_top">Customers</a> &nbsp;
<a href="orders.php" target="_top">Orders</a> &nbsp; <a
href="Admin.php" target="_top">Pricelist</a></p>
</div>
</body>
</html>

```

UML-діаграма прецедентів

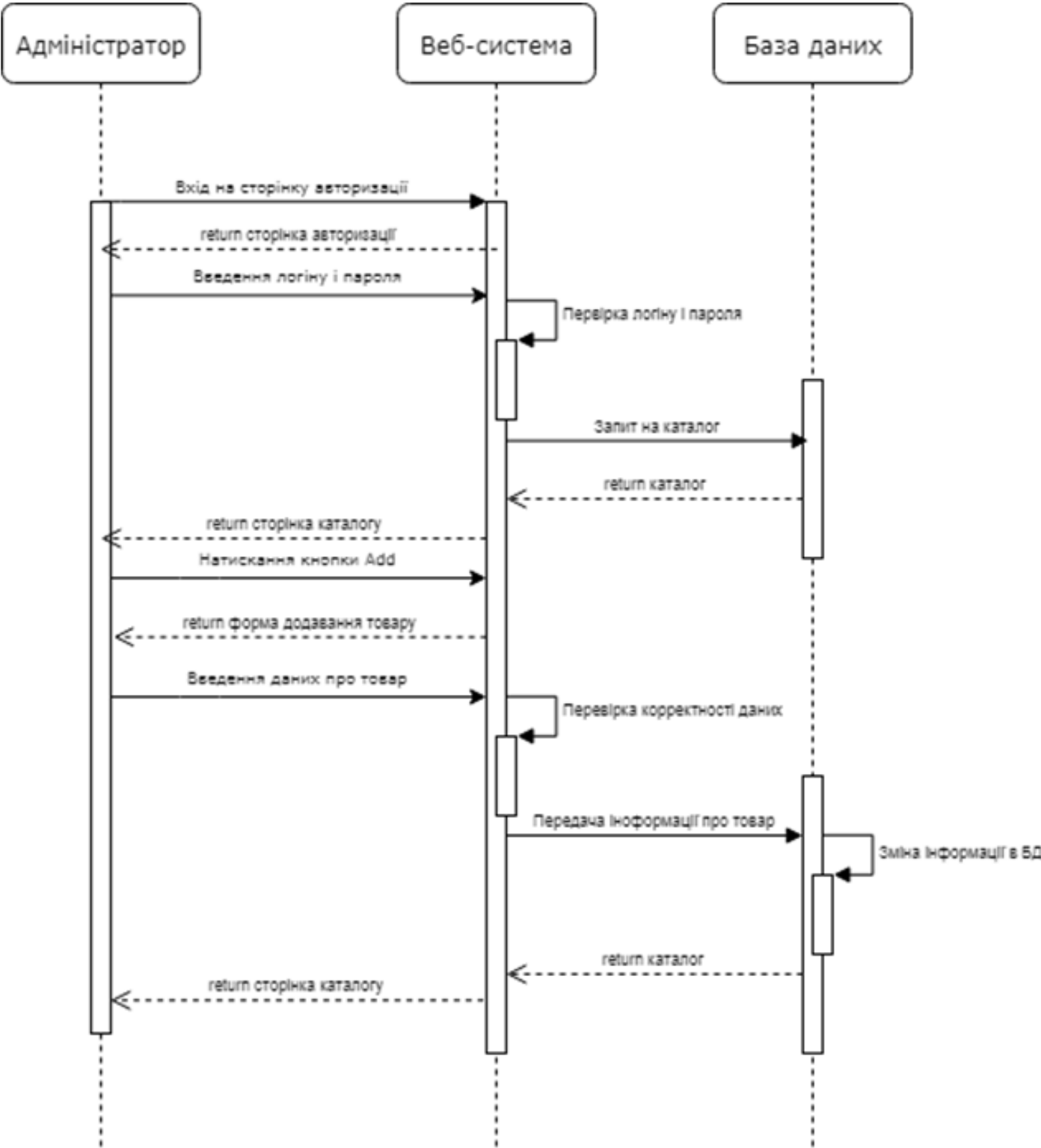


Діаграма послідовності «Купівля товару»

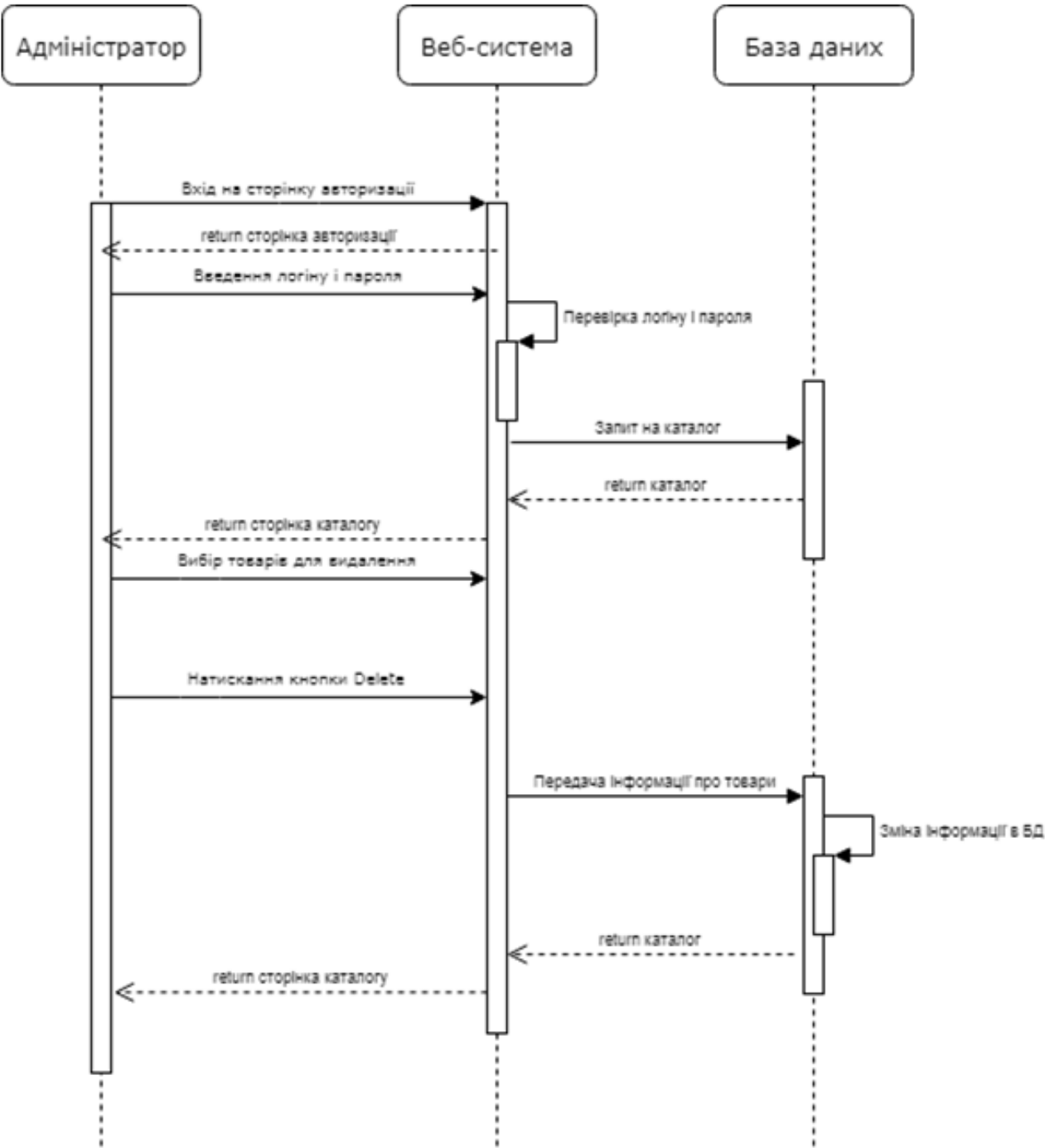


					13.02070849.00010 Пл1							
Зм.	Лист	№ документа	Підпис	Дата	Моделювання системи				Літ.		Лист	Листів
Розробив		Рябченко Н.А.									1	1
Перевірив		Кудерметов Р.К.										
Н. контр.		Щербак Н.В.										
Затвердив		Кудерметов Р.К.										
					НУ «Запорізька політехніка», група КНТ-510							

Діаграма послідовності «Додавання товару»



Діаграма послідовності «Видалення товару»



					13.02070849.00010 Пл1			
Зм.	Лист	№ документа	Підпис	Дата				
Розробив		Рябенко Н.А.						
Перевірів		Кудерметов Р.К.						
Н. контр.		Щербак Н.В.						
Затвердив		Кудерметов Р.К.						
						Літ.	Лист	Листів
							1	2
Моделювання системи						НУ «Запорізька політехніка», група КНТ-510		

Архітектура інтернет-магазину



					13.02070849.00010 Пл2								
Зм.	Лист	№ документа	Підпис	Дата									
Розробив		Рябченко Н.А.			Архітектура інтернет-магазину				Літ.	Лист	Листів		
Перевірів		Кудерметов Р.К.									1	1	
Н. контр.		Щербак Н.В.											
Затвердив		Кудерметов Р.К.							НУ «Запорізька політехніка», група КНТ-510				

Макет сторінки каталогу

Img_Logo

Filter

Sort

Table_Data

Submit

Reset

Макет головної сторінки

Navbar

Carousel

4 col

4 col

4 col

Footer

Макет сторінки опису товарів

Navbar

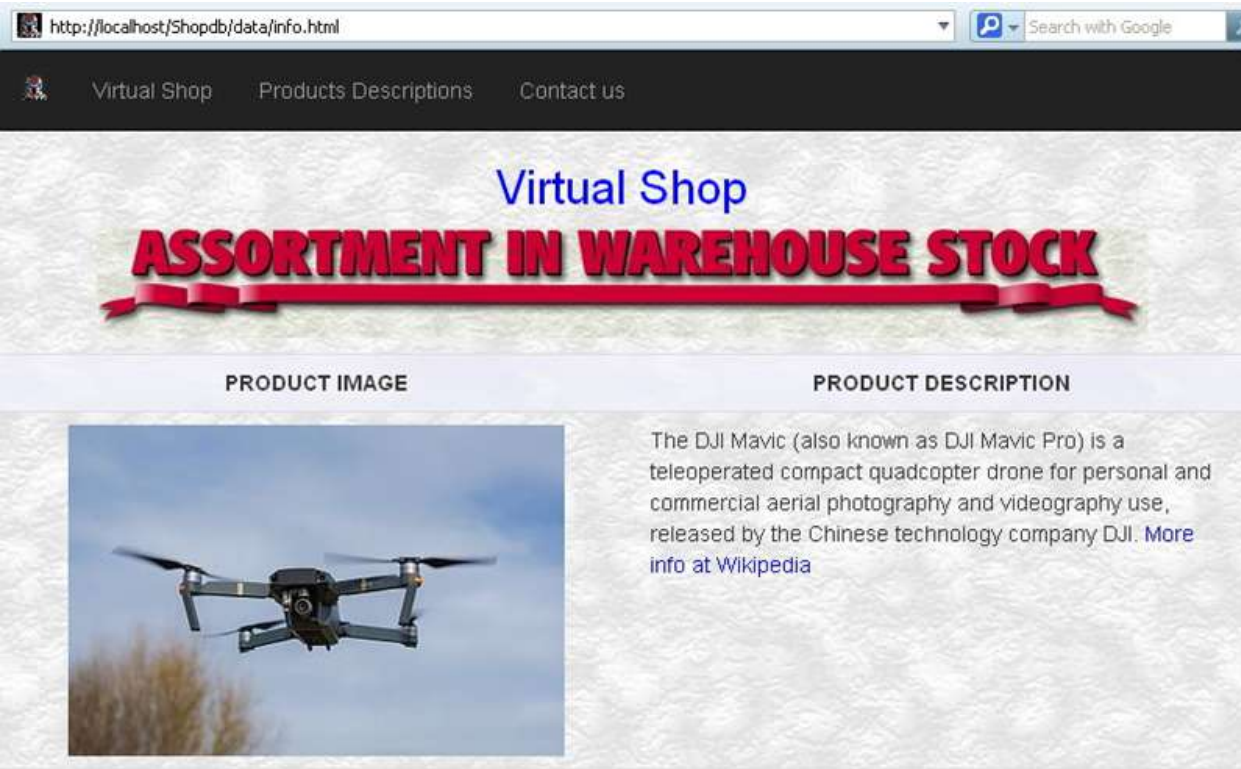
Img_Logo

Table_Img

Table_Data

					13.02070849.00010 ПлЗ			
Зм.	Лист	№ документа	Підпис	Дата	Прототипи сторінок інтернет-магазину	Літ.	Лист	Листів
Розробив	Рябченко Н.А.						1	1
Перевірив	Кудерметов Р.К.							
Н. контр.	Щербак Н.В.							
Затвердив	Кудерметов Р.К.				НУ «Запорізька політехніка», група КНТ-510			

Сторінка опису товару



Сторінка замовлення

http://localhost/Shopdb/cartdb.php

Your Order

Name	Warehouse	Quantity	Ordered	Price (EUR)	Remove
VRT-300	49	48	1	3.6	☐

With Checked/Selected:

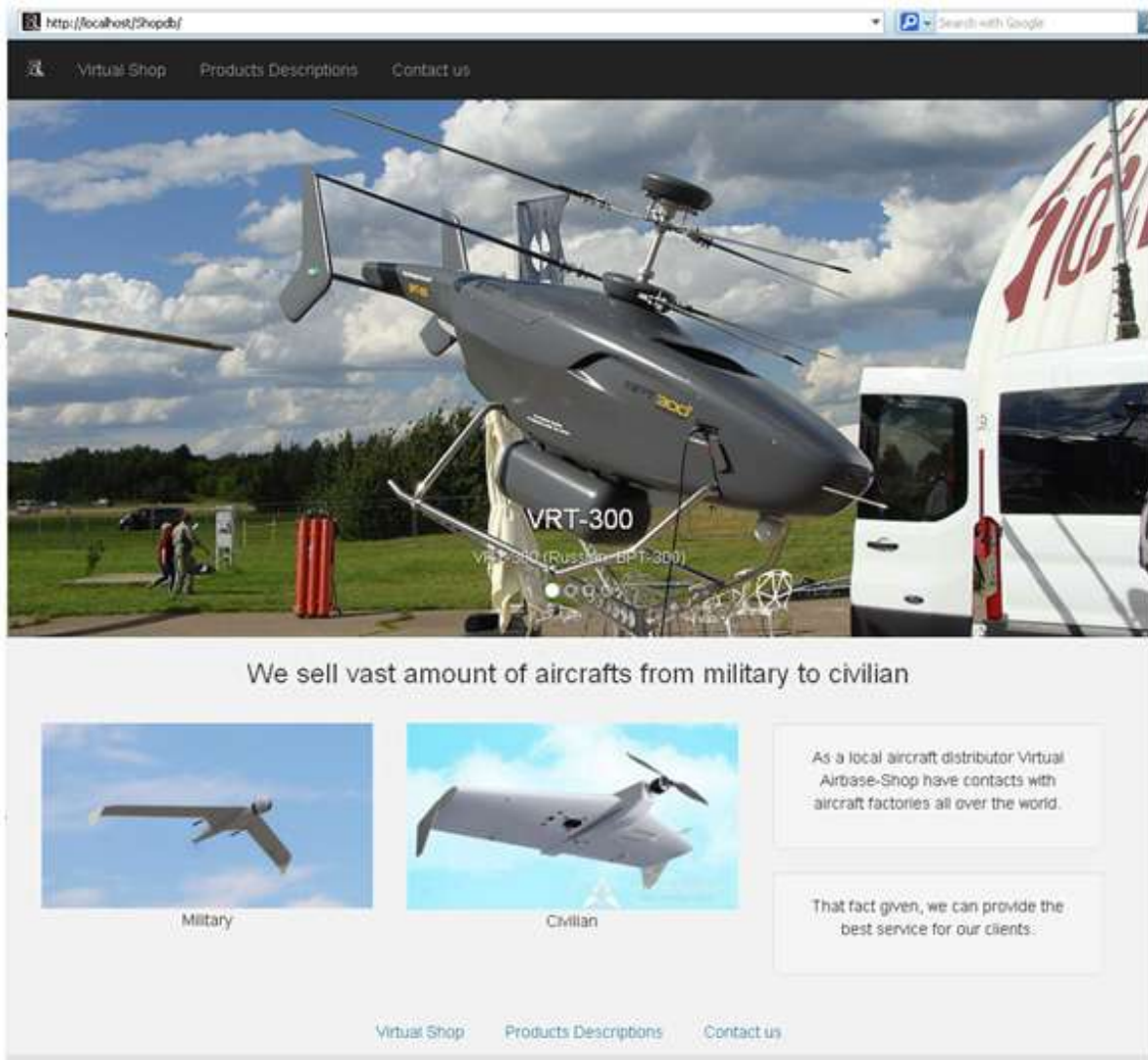
User List

Your name: Choose your country:

Your 16-digit Credit Card Number: Payment System provided by

Address: E-Mail:

Головна сторінка



					13.02070849.00010 Пл4			
Зм.	Лист	№ документа	Підпис	Дата	Сторінки інтернет-магазину			
Розробив	Рябенко Н.А.							
Перевірив	Кудерметов Р.К.							
Н. контр.	Щербак Н.В.							
Затвердив	Кудерметов Р.К.							
					Літ.			Лист
								1
								Листів
								1
					НУ «Запорізька політехніка», група КНТ-510			