

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»

Кафедра інженерії програмного забезпечення

КУРСОВИЙ ПРОЄКТ
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

з дисципліни: Backend-розробка

на тему:

«CMS-система для інтернет-магазину комп'ютерної техніки»

студента II курсу групи ІПЗ-21-5
спеціальності 121 «Інженерія
програмного забезпечення»

Волкова Олександра Максимовича

(прізвище, ім'я та по-батькові)

Керівник ктн, доц. Андрій МОРОЗОВ

Дата захисту: " 17 " січня 2023 р.

Національна шкала _____

Кількість балів: _____

Оцінка: ECTS _____

Члени комісії

(підпис)

(підпис)

(підпис)

Олена ЧИЖМОТРЯ
(прізвище та ініціали)

Ірина ДМИТРЕНКО
(прізвище та ініціали)

Денис ФУРІХАТА
(прізвище та ініціали)

Житомир – 2023

ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
Факультет інформаційно-комп'ютерних технологій
Кафедра інженерії програмного забезпечення
Освітній рівень: бакалавр
Спеціальність 121 «Інженерія програмного забезпечення»

«ЗАТВЕРДЖУЮ»
В.о.зав. кафедри
_____ А.В.Морозов
“ ____ ” _____ 20__ р.

ЗАВДАННЯ
НА КУРСОВИЙ ПРОЄКТ СТУДЕНТУ
Волкову Олександрю Максимовичу

1. Тема роботи: CMS – система для інтернет-магазину електроніки, керівник курсового проєкту: к.т.н., доцент Андрій МОРОЗОВ
2. Строк подання студентом: “ 17 ” січня 2023 р.
3. Вихідні дані до роботи: розробити CMS-систему для інтернет-магазину електроніки.
4. Зміст розрахунково-пояснювальної записки(перелік питань, які підлягають розробці)
 1. Постановка завдання
 2. Аналіз аналогічних розробок
 3. Алгоритми роботи програми
 4. Опис роботи програми
 5. Програмне дослідження
5. Перелік графічного матеріалу(з точним зазначенням обов'язкових креслень)
 - 1.Посилання на репозиторій: <https://gitlab.com/2021-2025/ipz-21-5/volkov-alexander/ip-kurswork/-/tree/master>
6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посади консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1,2,3	Денис ФУРІХАТА, асистент каф. КН		

7. Дата видачі завдання “ 1 ” грудня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів курсового проєкту	Строк виконання етапів проєкту	Примітки
1	Постановка задачі	01.12-05.12	Виконано
2	Пошук, огляд та аналіз аналогічних розробок	06.12 – 10.12	Виконано
3	Формулювання технічного завдання	11.12 – 15.12	Виконано
4	Опрацювання літературних джерел	16.12 – 18.12	Виконано
5	Проектування структури	19.12 – 25.12	Виконано
6	Написання програмного коду	26.12 – 12.01	Виконано
7	Налагодження	13.01 – 14.01	Виконано
8	Написання пояснювальної записки	15.01 – 16.01	Виконано
9	Захист	17.01.2023	

Студент

(підпис)

Олександр ВОЛКОВ.

(прізвище та ініціали)

Керівник проєкту

(підпис)

Андрій МОРОЗОВ

(прізвище та ініціали)

РЕФЕРАТ

Завданням на курсовий проєкт було створення інтернет-магазину електроніки.

Пояснювальна записка до курсового проєкту на тему розробка CMS-системи для інтернет-магазину електроніки складається з вступу, трьох розділів, висновків, списку використаної літератури та додатків.

Текстова частина викладена на 44 сторінках друкованого тексту.

Пояснювальна записка має 10 сторінок додатків. Список використаних джерел містить 10 найменувань і займає 1 сторінку. В роботі наведено 38 рисунків. Загальний обсяг роботи – 56 сторінок.

Ключові слова: Rozetka, інтернет-магазин, веб-додаток, БД, PHP, CMS.

					ДУ «Житомирська політехніка».23.121.07.000 - ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Олександр ВОЛКОВ			Розробка CMS-системи для інтернет-магазину електроніки	Літ.	Арк.	Аркушів
Перевір.		Денис ФУРІХАТА					4	70
Керівник						ФІКТ Гр. ІПЗ-21-5[2]		
Н. контр.								
Зав. каф.								

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	6
ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ	9
1.1 Аналіз задачі, засобів та методів її вирішення	9
1.2 Аналіз існуючого програмного забезпечення за тематикою	9
курсового проєкту (роботи).....	9
1.3 Технічне завдання на курсову роботу	13
1. Загальне положення	13
2. Підстава для розробки	14
3. Вимоги до програми.....	14
Висновки до першого розділу	16
РОЗДІЛ 2 ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ	17
2.1 Проєктування загального алгоритму роботи програми	17
2.2 Розробка функціональних алгоритмів роботи програми	20
2.3 Розробка програмного забезпечення.....	25
Висновки до другого розділу	28
РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ	29
3.1 Опис роботи з програмним додатком	29
3.2 Тестування роботи програмного забезпечення.....	39
Висновки до третього розділу	43
ВИСНОВКИ.....	44
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	45
ДОДАТКИ.....	46
Додаток А	34

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – База даних.

КП – Курсовий проєкт.

ПЗ – Програмне забезпечення.

CMS (Content Management System) – це система управління контентом сайту.

MVC – Модель–вигляд–контролер (або Модель–представлення–контролер, Model-view-controller, MVC) — архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми

В сьогодення, серед суспільства широко використовуються веб-додатки такі, як інтернет-магазини. Це в швидкий спосіб дозволяє здійснювати покупки віддалено, не йдучи до магазину. Інтернет-магазини можуть використовуватися не тільки за прямим призначенням – покупка товарів, а також можуть містити сторінку для здійснення благодійних внесків, для рекламування інших ресурсів, товарів та магазинів. Також замовлення товарів через інтернет зменшує ризик захворювання коронавірусною хворобою, оскільки людина в більшості випадків має менший контакт з людьми у порівнянні з покупкою товару у звичайному магазині, що є дуже актуально за останні декілька років. Замовлення товарів через інтернет є зручним, оскільки сайт може містити як завгодно багато товарів (оголошень), на противагу звичайному магазину, який в більшості випадків не може тримати всі товари на вітрині, оскільки товари займають фізичну площу. Актуальність вибраної теми збільшує те, що товарами, розробленими мною в магазині, є електричні прилади, адже в наш час це дуже важливо мати гаджети які забезпечують постійний зв'язок, прилади, які дають резервне джерело енергії у разі відключення основного джерела. Таким чином, використання інтернет-магазинів електроніки є дуже популярним, і їх популярність, в найближчому майбутньому, буде тільки зростати.

Об'єктом дослідження є методи та засоби розробки CMS системи для адміністрування інтернет-магазину.

Предметом дослідження є використання веб-технологій для забезпечення інформаційних потреб веб-додатку.

Метою створення курсового проєкту є дослідження особливостей проєктування та реалізації системи адміністрування інтернет-магазину електроніки.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		7

Завданням на курсову роботу є:

- опрацювання теоретичних відомостей з проєктування та реалізації CMS;
- пошук та аналіз веб-сайтів суміжних з обраною темою, визначення їх унікальності, переваг та недоліків;
- розробка адаптивного інтерфейсу веб-сайту за допомогою мови гіпертекстової розмітки – HTML5, мови програмування – JS, а також таблиці стилів – CSS3;
- написання серверної частини сайту мовою PHP;
- проєктування БД для функціонування інтернет-магазину за допомогою веб-додатку phpMyAdmin.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		8

РОЗДІЛ 1. АНАЛІЗ ПРОБЛЕМАТИКИ, МЕТОДІВ ТА ЗАСОБІВ ВИРІШЕННЯ ЗАДАЧІ

1.1 Аналіз задачі, засобів та методів її вирішення

Задачею курсового проєкту є написання інтернет-магазину, який повинен мати достатній функціонал, простоту і зручність користування, інтерфейс сайту повинен бути зрозумілим для користувача, веб-сайт повинен належним чином реагувати на введення коректних та некоректних даних. При написанні проєкту, слід використати набуті знання з мови PHP для створення серверної частини сайту, паттерн проєктування MVC, середовище СУБД, а також HTML5, CSS3, JS.

Завдання потребувало використання програми для створення веб-додатку і тому, для реалізації описаного вище програмного продукту, чудово підходить крос-платформове інтегроване середовище розробки PhpStorm. Цей додаток має великий функціонал для написання веб-додатку: автоматичне виправлення помилок, наявність передових технологій веб-розробки, повноцінна підтримка PHP разом з базами даних і SQL, можливість взаємодії з системою контролю версій за допомогою інтерфейсу.

У курсовому проєкті однією з перепон, яка виникла, було знаходження середовища зберігання даних, розроблюваного мною веб-додатку та середовище для адміністрування СУБД MySQL. Для вирішення цієї проблеми, я використав веб-додаток phpMyAdmin. Цей програмний продукт простий у користуванні, має зручний інтерфейс і зберігає дані в табличному вигляді, що є надзвичайно зручно при їх перегляді та редагуванні.

1.2 Аналіз існуючого програмного забезпечення за тематикою курсового проєкту (роботи).

В сьогодення існує багато різних інтернет-магазинів електроніки, усі вони мають певні переваги та недоліки. Для себе я виділяю чотири найбільш популярні інтрнет-магазини електроніки, які слід розглянути: Rozetka, Фокстрот, Comfy та Brain.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		9

Rozetka – інтернет магазин, який займається продажем електроніки. Даний веб-продукт націлений на українську аудиторію, оскільки домен верхнього рівня є буквосполучення “ua”.

Посилання на ПО: <https://rozetka.com.ua>

Вигляд вікна інтернет-магазину – рис. 1.1

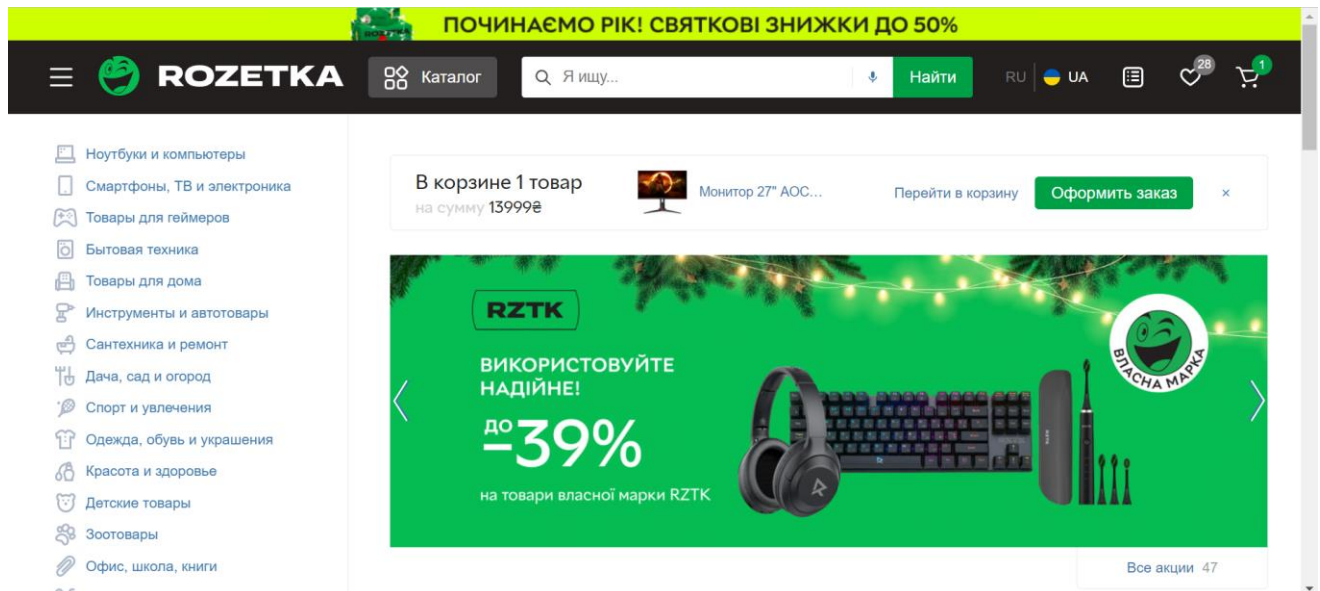


Рис. 1.1. Інтерфейс інтернет-магазину “Rozetka”

Переваги:

- 1) Веб-додаток має велику кількість електроніки, що дозволяє майже завжди знайти необхідний пристрій для покупки.
- 2) Застосунок має зручний інтерфейс в якому легко орієнтуватися та знаходити необхідні категорії та товари.
- 3) Застосунок має привабливий дизайн та гарно підібрану кольорову гаму сайту.
- 4) Веб-додаток має програмно реалізований список бажаних товарів, який дозволяє користувачу не запам'ятовувати ці товари та полегшує пошук товару.

Недоліки:

- 1) Відсутність темної теми сайту.
- 2) Для покупки товару потрібно пройти обов'язкову реєстрацію або авторизацію на сайті за допомогою інших сервісів.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				10
Змн.	Арк.	№ докум.	Підпис	Дата		

Фокстрот – інтернет-магазин електроніки в Україні.

Посилання на ПО: <https://www.foxtrot.com.ua/>

Вигляд вікна інтернет-магазину – рис. 1.2

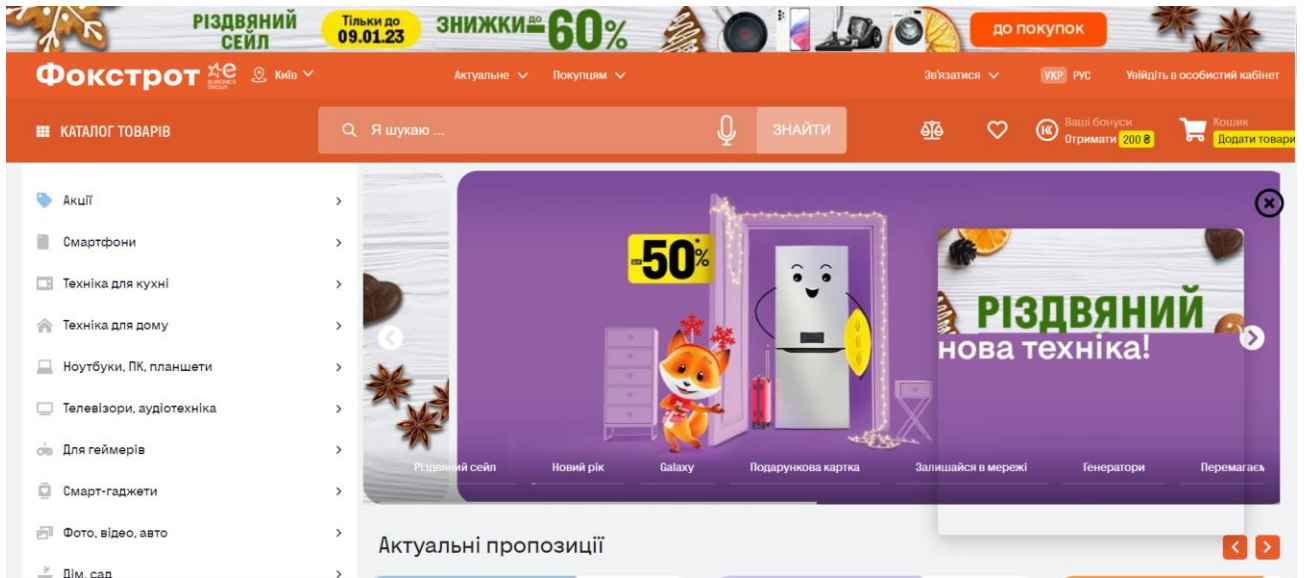


Рис. 1.2. Інтерфейс інтернет-магазину “Фокстрот”

Переваги:

- 1) Зрозумілий та зручний поділ товарів на категорії, що допомагає потенційному покупцю орієнтуватися на сайті.
- 2) Можливість зміни мови веб-сайту, що допомагає розширити аудиторію потенційних покупців.
- 3) Немає необхідності реєстрації або авторизації на сайті для здійснення покупки.

Недоліки:

- 1) Відсутня можливість вибору темної теми сайту.
- 2) Проблема з адаптивністю на деяких проміжках по горизонталі у тематичному блоці “Топ-пропозиції”.

Comfy – сучасний, клієнтоорієнтований веб-сайт для продажу електроніки з досить хорошим функціоналом.

Посилання на ПО: <https://comfy.ua>

Вигляд вікна інтернет-магазину – рис. 1.3

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				11
Змн.	Арк.	№ докум.	Підпис	Дата		

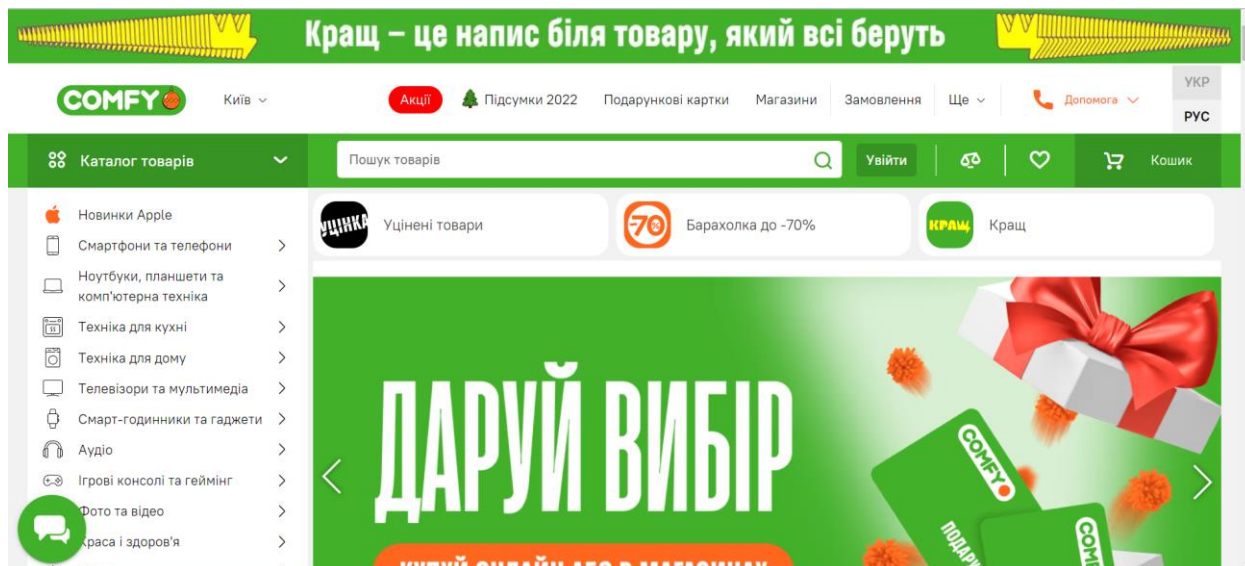


Рис. 1.3. Інтерфейс інтернет-магазину “Comfy”

Переваги:

- 1) Можливість зміни мови інтерфейсу.
- 2) Привабливий дизайн сайту.
- 3) Немає необхідності реєстрації на сайті для оформленн замовлення;
- 4) Зрозумілий поділ товарі на категорії.
- 5) При пошуку товарів за допомогою спеціального поля, окрім запропонованих назв товарів, відображаються фото товару.

Недоліки:

- 1) На сайті використано багато анімації контенту, що може не всім відвідувачам сайту сподобатися.
- 2) Наявність лише однієї теми сайту.

Brain – інтернет магазин для продажу електроніки.

Вигляд вікна інтернет-магазину – рис. 1.4

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				12
Змн.	Арк.	№ докум.	Підпис	Дата		



Рис. 1.4. Інтерфейс інтернет-магазину “Brain”

Переваги:

- 1) Зрозумілий інтерфейс сайту, який допомагає з легкістю орієнтуватися.
- 2) Можливість зміни мови інтерфейсу.
- 3) Можливість придбання товару, як з реєстрацією так і без неї.

Недоліки:

- 1) Не надто привабливий дизайн сайту.
- 2) При тестуванні адаптивного режиму сайту, при розмірі вікна для мобільних телефонів, вдалося розкрити “меню-бургер” не з першого разу.

1.3 Технічне завдання на курсову роботу

1. Загальне положення

1.1. Найменування програмного засобу

Повне найменування програмної системи: "Розробка інтернет-магазину електроніки" (надалі "веб-додаток"). Коротка назва програмної системи - "Інтернет-магазин"

1.2. Призначення розробки та область застосування

Веб-сайт "Інтернет-магазин електроніки" призначений для розміщення каталогу електронних пристроїв та здійснення їх пошуку, продажу та реклами, також існує система перегляду товарів за категорією.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				13
Змн.	Арк.	№ докум.	Підпис	Дата		

Веб-сайт "Інтернет-магазин електроніки" дозволить швидко переглядати інформацію про товари, та забезпечить адміністратора сайту своєчасною та повною інформацією про актуальний стан замовлення покупок інтернет-магазину, а також надасть користувачам можливість вибору електро-приладів різних типів, в залежності від приналежності їх до певної категорії. В доповнення до вище сказаного, адміністратор сайту має можливість змінювати статус зареєстрованої особи, надавати користувачу права адміністратора.

1.3. Найменування розробника та замовника

Розробник даного продукту - студент групи ІПЗ-21-5 Волков Олександр Максимович (надалі "розробник").

Замовник програмного продукту – кафедра інженерії програмного забезпечення Державного університету «Житомирська політехніка» в межах виконання курсової з дисципліни «Backend-розробка» Фуріхата Денис Вікторович, Морозов Андрій Васильович

2. Підстава для розробки

2.1. Документ на підставі якого ведеться розробка

Розробка ведеться на підставі навчального плану за напрямом 121 «Інженерія програмного забезпечення».

3. Вимоги до програми

3.1. Вимоги до функціональних характеристик

3.1.1. Загальні вимоги

Веб-додаток має забезпечувати:

- можливість дистанційної роботи з робочих станцій локальної та глобальної мережі підприємства;
- постійний доступ користувачів веб-додатку;
- оформлення замовлення;
- організацію управління сайтом;
- можливість доступ до бази даних;

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				14
Змн.	Арк.	№ докум.	Підпис	Дата		

3.1.2. Загальні вимоги

Розробити інтернет-магазин електроніки, що підтримує виконання наступних операцій:

1. Пошук товарів за категоріями.
2. Додавання, оновлення та видалення категорій та товарів.
3. Додавання інформації про товар.
4. Можливість коментування покупки з вказанням його рейтингу.
5. Перегляд інформації про електро-товари.
6. Можливість перегляду історії покупок.
7. Можливість додавання адміністратором реклами для слайдеру з вказанням URL-адреси на конкретний товар.
8. Можливість зміни прав доступу користувача.
9. Можливість зміни статусу заповнення.
10. Можливість виконати налаштування профілю (зміна паролю, електронної пошти, власних даних, а також змога видалити профіль при необхідності).
11. Можливість “живого пошуку” товарів на сайті.

3.1.3. Організація вхідних і вихідних даних

Вхідними даними є інформація про товар (назва, ціна, кількість, повний опис товару, короткий опис товару, категорія товару і відображення товару користувачу).

Організація вхідних і вихідних даних повинна відповідати інформаційній структурі виконуваних операцій, вхідним та вихідним паперовим документами.

Введення оперативних даних повинно виконуватися з використанням діалогових екранних форм, побудованих на основі візуальних компонентів. Введення даних виконується на основі затверджених форм документів: анкета, заява, інформаційна довідка та в режимі online оператором зі слів користувача.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				15
Змн.	Арк.	№ докум.	Підпис	Дата		

3.1.4. Часові характеристики і розмір пам'яті, необхідної для роботи програми.

Час реакції програми на дії користувача (маніпуляції з пристроями введення даних) не повинен перевищувати 0,25 с.

Час виконання команд меню не більше 1 с.

Відображення масивів даних за запитами не більше 3 хвилин.

Доступність БД – 90% цілодобово.

Операції з'єднання з БД не більше 1 хвилини.

Обсяг оперативної пам'яті, необхідний для роботи програми не менше 1 Гб. Дисковий простір, необхідний для збереження програми і файлів даних не більше 300 Мбайт для робочої станції та 20 ГБайт..

Інсталяційний пакет програми, що містить у складі БД не повинні перевищувати 100 Мбайт.

3.2. Вимоги до методів рішення і мов програмування

Вибір методів рішення здійснюється розробникам без узгодження з замовником.

3.2.1. Вимоги до системи програмних засобів

OpenServer, PHP 5, MySQL, HTML 5, CSS 3, JavaScript:

Вимоги до програмного забезпечення робочої станції: PHP Storm

Висновки до першого розділу

В даному розділі було поставлено та проаналізовано задачу курсового проєкту, було обговорено проблеми, що виникли під час розробки програмного продукту, вибрано програму та мову розробки, за допомогою яких, буде реалізовуватися цей проєкт. Розглянуті аналогічні існуючі додатки з вказанням їх переваг та недоліків, розглянуто вимоги, які повинні бути дотриманні в проєкті.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				16
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2 ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Проєктування загального алгоритму роботи програми

Одним з завдань проєкту є створення загального алгоритму роботи програми.

Після того як користувач зайшов на сайт інтернет-магазину, програма очікує дій користувача.

Користувач може виконати наступні дії у вкладці входу:

1. Вести логін та пароль, та за умови правильного вводу даних, натиснувши кнопку “Увійти”, в залежності від отриманих прав, перейти в головне меню магазину або на сторінку адміністратора.
2. Якщо користувач ще не зареєстрований, він має змогу пройти реєстрацію натиснувши кнопку “Реєстрація”.
3. Якщо користувач не бажає здійснювати реєстрацію, він має змогу перейти в головне меню магазину.

Користувач може виконати наступні дії у головному меню інтернет-магазину (коли обліковий запис не має прав адміністратора):

1. Погортати рекламний слайдер.
2. Обрати необхідну категорію товарів, після чого відкриється сторінка з товарами, які належать даній категорії.
3. Здійснити перегляд кошику та за наявності товарів оформити замовлення.
4. Здійснити перегляд історії покупок (за умови, що користувач є зареєстрований та авторизований в інтернет-магазині).
5. Здійснити перегляд товарів, які віднесені до списку охочих купити (за умови, що користувач є зареєстрований та авторизований в інтернет-магазині).
6. Написати коментар для товару з вказанням рейтингу (за умови що товар придбаний користувачем)

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				17
Змн.	Арк.	№ докум.	Підпис	Дата		

7. Здійснити налаштування профілю (змінити особисті дані, логін, пароль або видалити поточний акаунт).

Користувач може виконати наступні дії у головному меню інтернет-магазину (коли обліковий запис є з правами адміністратора):

1. Переглянути точки видачі розетки на міні-карті (для логістики).
2. Додати, змінити або видалити категорію.
3. Додати товар до нової категорії або змінити та видалити товар в певній категорії.
4. Змінити фото слайдеру та посилання на ці фото.
5. Змінити статус замовлення клієнта.
6. Змінити права облікового запису.
7. Здійснити налаштування профілю (змінити особисті дані, логін, пароль або видалити поточний акаунт).

Структура CMS-системи:

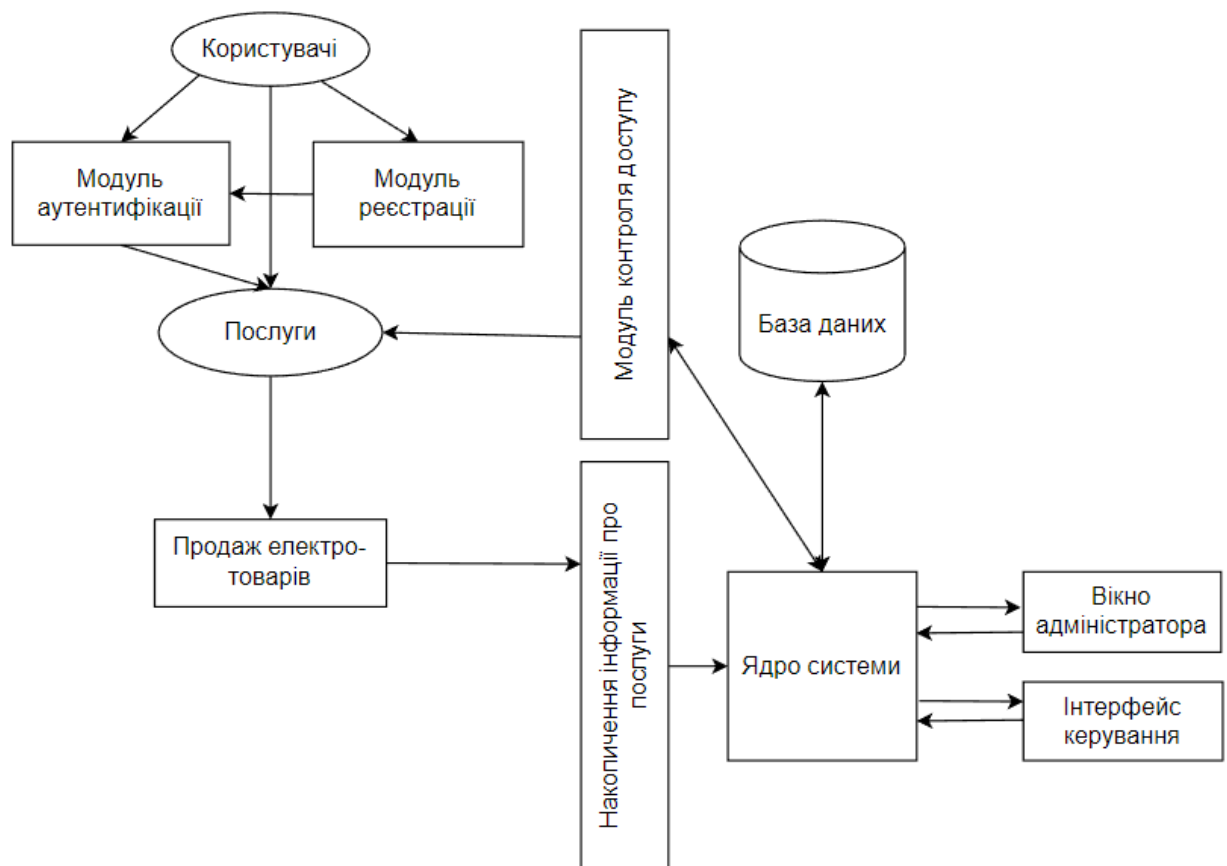


Рис. 2.1. Структура системи інтернет-магазину електротоварів

Діаграма БД “SHOPCMS”:

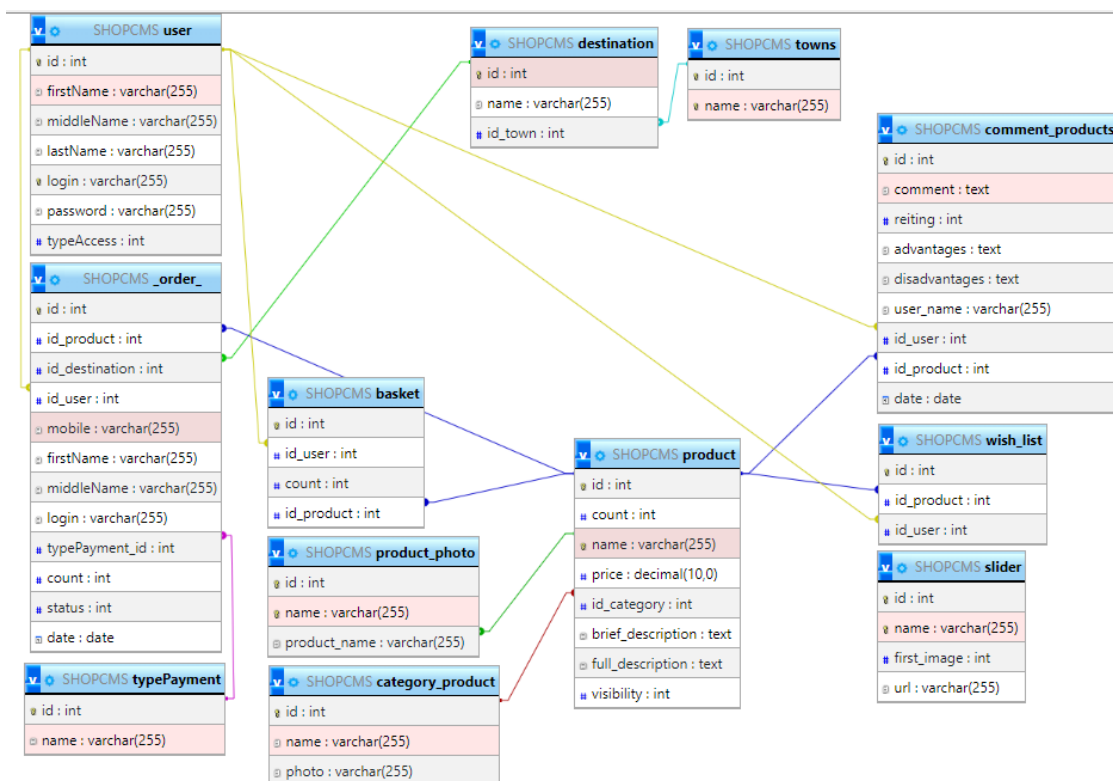


Рис. 2.2. Діаграма „сутність-зв’язок” (фізичний рівень) системи

Діаграма БД “SHOPCMS”:

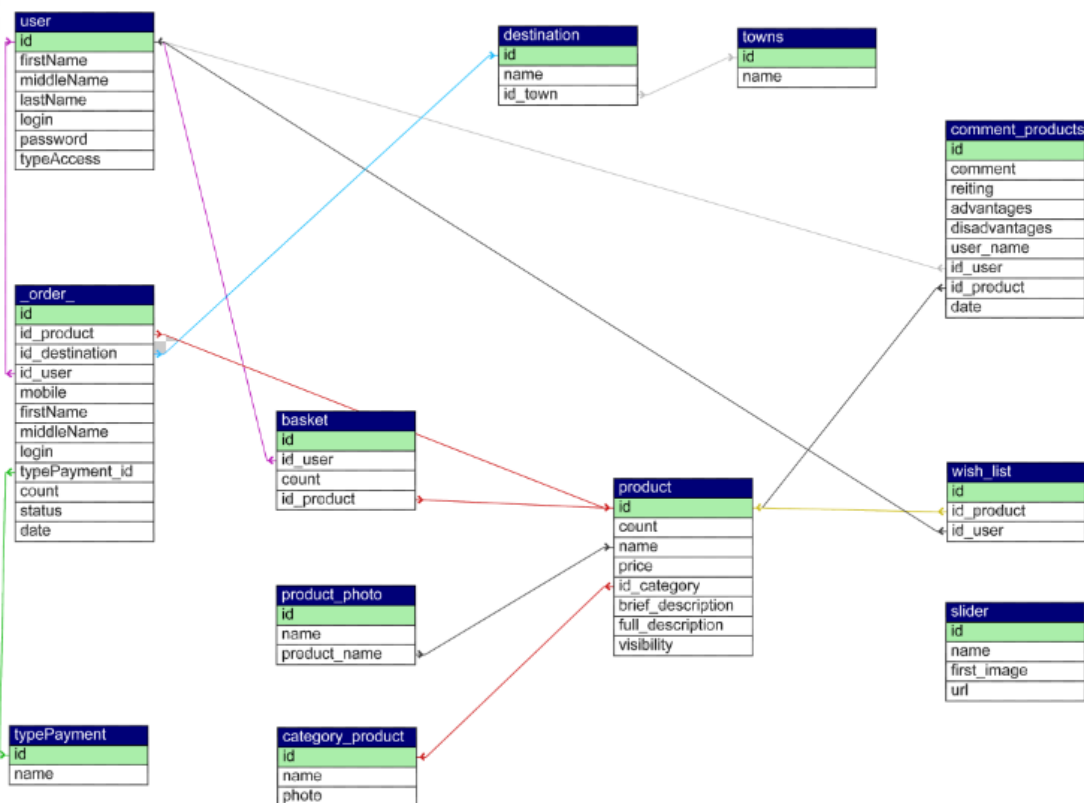


Рис. 2.3. Діаграма „сутність-зв’язок” (логічний рівень) системи

2.2 Розробка функціональних алгоритмів роботи програми

Програма складається з декількох основних функціональних алгоритмів, тож опишу їх словесним способом:

1. Алгоритм реєстрації

Суть алгоритму реєстрації полягає в наступному: на початковому етапі користувач заповнює всі поля реєстрації. Під час заповнення полів форми спрацьовує подія “change” і якщо заповнювальний текст не відповідає заданим регулярним виразам, то відбувається підсвітка відповідного поля вводу червоним кольором, що сигналізує про помилку. При відправці форми на сервер, для реєстрації користувача передбачена функція `public function registerAction()` класу `class UserController`. Функція перевіряє чи дійсно була відправлена форма методом “POST” і якщо умова виконується, то перевіряється кожне поле форми на валідацію. Для перевірки електронної пошти використовую спеціальну функцію `filter_var`, також перевіряю чи є користувач вже зареєстрований з такою електронною поштою. Якщо умови валідації не виконуються, то заносу відповідні повідомлення в спеціально створений масив.

Перевірка пароля відбувається регулярним виразом та перевірка пароля з його повторенням, а перевірка особистих даних користувача (імені, прізвища та по-батькові) відбувається також за допомогою регулярного виразу. Якщо кількість помилок виявилось більше 0, то повертається вікно реєстрації з відповідним повідомленням про помилку.

```
if (count($errors) > 0)
{
    $model = $_POST;
    return $this->render(null, [
        'errors' => $errors,
        'model' => $model
    ]);
}
```

2. Алгоритм авторизації

Суть алгоритму авторизації полягає в наступному: якщо користувач вже був авторизований, то при повторному вході в інтернет-магазин, йому

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				20
Змн.	Арк.	№ докум.	Підпис	Дата		

вже не потрібно проходити авторизацію знову, він буде відразу перекинутий в головне меню магазину. За умови заповнення даних форми авторизації і натисканні кнопки “Увійти” відбувається прийом відправлених даних методом `public function loginAction()` класу `UserController`. Відбувається перевірка чи була відправлена форма авторизації і якщо перевірка є істинною то виконується метод `getUserByLoginAndPassword` класу `User`, де відбувається звернення через клас `Core` ядра системи до бази даних і спроба знайти користувача з відповідним логіном та паролем.

```
$user = User::getUserByLoginAndPassword($_POST['login'], $_POST['password']);
```

Якщо користувач не знайдений, то створюється змінна, в яку вноситься інформація про помилку. Якщо ж користувач все таки знайдений, то відбувається занесення даного користувача в сесію (клас `class User` метод

```
public static function authenticationUser()
public static function authenticationUser($user)
{
    $_SESSION['user'] = $user;
}
```

, заповнення його корзини викликавши метод ініціалізації корзини:

```
public static function initializeBasket()
{
    if (User::isAuthenticatedUser() && !User::isUserAdmin())
    {
        $products =
self::getProductsById(User::getCarrentAuthenticatedUser()['id']);
        foreach ($products as $product) {
            $_SESSION['basket'][$product['id_product']] = $product['count'];
        }
    }
}
```

, та списку товарів які користувач бажав би купити:

```
public static function initializeWishList(){
    if (User::isAuthenticatedUser() && !User::isUserAdmin()){
        $products =
self::getProductsById(User::getCarrentAuthenticatedUser()['id']);
        foreach ($products as $product) {
            $_SESSION['wish'][$product['id_product']] = 1;
        }
    }
}
```

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				21
Змн.	Арк.	№ докум.	Підпис	Дата		

, за умови, що вже були якісь товари в цих списках, і на кінцевому етапі відбувається переадресація користувача на головну сторінку сайту. За умови неправильно введеного логіна і пароля, відбувається виведення відповідного повідомлення про помилку на сторінці авторизації.

3. Алгоритм покупки товарів:

Суть алгоритму покупки товарів полягає в наступному: при натисканні іконки у вигляді кошика на товарі, який бажаємо придбати, відбувається подія “click”, з наступним пошуком data-атрибуту відповідного товару, який містить його ідентифікаційний номер в базі даних і виклику метода `function sendRequestAddProductToBasket(productId)` в JS-файлі, який передається даний атрибут у вигляді параметру. Метод призначений для здійснення аякс-запиту до метода `addAction` класу `BasketController`, який в свою чергу звертається до метода `public function addAction()` класу `class BasketController` попередньо перевірючи чи переданий необхідний параметр ідентифікаційного номеру товару.

Метод `public static function addToBasket($productId, $count = 1)` робить перевірку, чи існує масив сесії для товарів кошика, якщо не існує – то створює його. Далі відбувається або занесення елемента в масив сесії “basket” з ключем номеру товару та значенням 1, або збільшення заданої кількості товарів на 1 (якщо змінюємо кількість елементів в самому кошику), далі можемо перейти в кошик та редагувати необхідну кількість товарів принцип роботи методів аналогічно описаному вище (пошук data атрибуту з ідентифікаційним номером товару, здійснення аякс-запиту за допомогою js до відповідного методу збільшення чи зменшення товарів в сесії, а отже і товарів для замовлення). При натисненні кнопки “Оформити замовлення” відбувається переадресація на відповідну сторінку, де користувачу необхідно заповнити відповідну форму, вказавши необхідну інформацію для здійснення успішного запиту на покупку товару. Якщо користувач підтверджує замовлення, то всі данні, які він мав заповнити відправляються на сервер, де вони починають обробку в методі `public function orderAction()` класу `class`

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				22
Змн.	Арк.	№ докум.	Підпис	Дата		

BasketController. Спочатку відбувається перевірка чи відправлення форма методом “POST” `if (Core::getInstance()->requestMethod == "POST")` і якщо дійсно дані були відправлені на сервер необхідним методом, то відбувається створення відповідної змінної та присвоєння їй одного значення з суперглобально асоціативного масиву “\$_POST”. Наступним кроком відбувається перевірка, чи є користувач, який здійснив замовлення авторизований в магазині, якщо так, то відбувається пошук даного користувача. Надалі відбувається додаткова перевірка створених змінних на порожнечу. Якщо змінна порожня, то відбувається занесення повідомлення про помилку в спеціально створений масив. Якщо виявляється, що число отриманих помилок більше 0, то відбувається повернення сторінки оформлення замовлення з вказанням біля необхідного поля повідомлення про помилку, якщо ж помилки відсутні, то відбувається цикл по масиву `$_SESSION['basket']` з подальшим формуванням інформації про товар та звернення до методу `public static function createOrder($filedList)` класу `Order` в який передається раніше створений масив з інформацією про товар і який призначений для вставки в БД відповідного замовлення. Далі відбувається звернення до методу `public static function deleteProductFromBasketStorage($productId, $userId)` класу `Basket`, передавши у якості параметрів ідентифікаційний номер товару та користувача, після чого метод здійснює видалення відповідного запису з бд в таблиці кошику, далі спрацьовує метод `public static function updateProduct($id, $row)` класу `Product`, який призначений для оновлення кількості товарів в бд (зменшує їх кількість).

```
foreach ($_SESSION['basket'] as $key => $value){
    $orderList = [
        'id_product'=>$key, 'id_destination'=>$selectDestination,
        'id_user'=>$userId, 'mobile'=>$mobile,
        'typePayment_id'=>$typePayment, 'count'=>$value,
        'date'=>date("Y-m-d")];
    Order::createOrder($orderList);
    Basket::deleteProductFromBasketStorage($key, $userId);
    Product::updateProduct($key, [
        'count'=>Product::getProductById($key)['count'] -$value
    ]);
}
```

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				23
Змн.	Арк.	№ докум.	Підпис	Дата		

Якщо ж користувач не є зареєстрованим, то відбуваються всі ті самі дії описані вище, але деякі відправляємо дані відрізняються (на противагу ідентифікаційного номера користувача відправляється логін та особисті дані користувача). Останнім етапом відбувається очищення сесії для корзини `$_SESSION['basket'] = [];` та переадресація на сторінку з успішним повідомленням про здійснення замовлення.

```
$this->redirect('/basket/order_success');
```

4. Алгоритм додавання адміністратором товарів в категорію

Суть даного алгоритму полягає в наступному: для здійснення додавання товару в категорію адміністратору потрібно заповнити відповідні поля, які описують товар, та відправити дану форму з полями на сервер. На сервері згадані поля опрацьовує метод `public function addAction()` класу `class ProductController`. На початковому етапі виконується перевірка чи користувач є адміністратором, якщо ні то виводиться відповідні сторінка з повідомленням про помилку.

```
if (!User::isUserAdmin())
{
    return $this->error(403);
}
```

Наступним кроком відбувається перевірка чи дійсно форма була відправлена на сервер методом “POST”, якщо ні, то повертається вікно з полями для додавання, а якщо форма відправлена необхідним методом, то відбувається перевірка даних, які прийшли з цією формою. Якщо якесь значення форми не відповідає заданій умові, то заноситься відповідне повідомлення про помилку в спеціально створений масив. Якщо помилок не виявлено, то додаємо товар до БД, викликавши на виконання метод `Product::addProduct($_POST);` класу `class Product` та передавши в метод в якості параметра суперглобальний асоціативний масив `$_POST`, далі відбувається виклик метода `PhotoProduct::addPhoto($_POST['name'], $_FILES['file']['tmp_name']);` класу `class PhotoProduct`, який призначений зберігати назви фотографій в БД відповідного товару. В якості параметру даний метод приймає назву товару та суперглобальний асоціативний масив

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				24
Змн.	Арк.	№ докум.	Підпис	Дата		

\$_FILES. Останнім кроком відбувається переадресація на головну сторінку адміністратора `$this->redirect('/')`; . Якщо все таки помилки виявлені, то відбувається повернення форми для додавання товару з вказанням повідомлення про помилку під відповідним полем вводу.

2.3 Розробка програмного забезпечення

Розробку CMS – системи розпочав зі створення та налаштування конфігураційного файлу `.htaccess` веб-сервера Apache для створення єдиної точки входу в систему. В даному файлі написав умови переадресації та файл (`index.php`) в який буду передана url адреса. Також, в даному файлі реалізував автозавантажувач класів. Наступним кроком було створення ядра системи, початковим етапом якого є створення класу `Core`. Клас `Core` – головний клас ядра системи. Включає в себе такі важливі методи як:

getInstance – повертає екземпляр ядра системи. Даний метод являється статичним `public static function getInstance()`. Функція його полягає в перевірці, чи є змінна `$instance` порожня `if (empty(self::$instance))`, якщо так, то створюється екземпляр класу, який присвоюється цій змінній `self::$instance = new Core()` після чого відбувається її повернення `return self::$instance`, але якщо змінна не є порожньою, то відбувається миттєве її повернення оминаючи перевірку.

initialize – виконує ініціалізацію системи. В цьому методі відбувається запуск сесії, створення об'єкта класу `DB` (клас що призначений для роботи з базою даних), визначається яким методом здійснено запит і присвоєння його в поле класу;

run – виконує основний процес роботи системи. На початковому етапі відбувається парсинг змінної, яка передається методом “GET”, яка визначена в файлі `.htaccess` і приймає рядок url адреси. Система побудована таким чином, що відкриття тієї чи іншої сторінки або виконання тієї чи іншої дії залежить від назви модуля та від назви метода, який знаходиться в данному модулі. Тому правильна url – адреса буде складатися з назви модуля та дії, які повинні існувати в проєкті. Щоб відокремити модуль від дії

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				25
Змн.	Арк.	№ докум.	Підпис	Дата		

використовуємо спеціальний метод `$pathParts = explode('/', $path);`
 Отримавши масив здійснюємо перевірку чи знаходиться в ньому назва модуля : `if (empty($moduleName)) $moduleName = "site";`, якщо модуль пустий то відбувається присвоєння назви головного модуля. Далі відбувається перевірка на існування дії даного модуля:

```
if (empty($actionName))
    if (User::isUserAdmin())
    {
        $actionName = "admin";
    }
    else
    {
        $actionName = "index";
    }
```

Якщо ім'я дії порожня, визначаю їх сам: для адміністратора назва дії є “admin”, а для звичайного користувача “index”. Далі утворюємо з назви модуля назву контролера з відносною адресою його розміщення:

```
$controllerName = '\controllers\\'.ucfirst($moduleName). 'Controller';
```

Визначаю ім'я метода даного контролера: `$actionName = $actionName. 'Action';`. Виконую перевірку на існування контролера в проєкті. Якщо контролер існує, то створюю його екземпляр і перевіряю існування метода в даному контролері: `if (method_exists($controller, $actionName))`, за умови істинності перевірки, звертаюсь до даного контролера і викликаю відповідний метод в ньому, а результат записую в змінну: `$actionResult = $controller->$actionName($pathParts)` з наступною перевіркою належності результату виконання до класу Error: `if ($actionResult instanceof Error)`. Якщо умова виконується, то записую в змінну статус код який повертає метод даного контролера: `$statusCode = $actionResult->code;` Далі відбувається присвоєння властивості класу результат виконання метода певного контролера: `$this->pageParams['content'] = $actionResult;`

Якщо не відбувається виконання умови існування метода в контролері, то в змінну записується статус код, який буде сигналізувати про помилку: `$statusCode = 404;`.

Таким чином, наприкінці даного метода класу Core, перевіряється отриманий статус-код:

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				26
Змн.	Арк.	№ докум.	Підпис	Дата		

```

$statusCodeType = intval($statusCode / 100);
if ($statusCodeType == 4 || $statusCodeType == 5) {
    $siteController = new SiteController();
    $this->pageParams['content'] = $siteController-
>errorAction($statusCode);
}

```

Якщо ми отримаємо цілочисельне значення, яке дорівнює 4 або 5 при діленні статус-кода на 100, то це означає, що трапилась певна помилка і в властивість присвоємо відповідне значення конструкції “switch”, що відповідає коду помилки.

done – виконує завершення роботи системи та виводу результату. Даний метод призначення для виводу html-коду сторінки.

construct – конструктор використовується, для створення об’єкта, який є екземпляром класу ядра системи.

Для взаємодії з БД в системі реалізовано спеціальний клас з необхідними методами, які здійснюють вставку, оновлення, видалення та читання даних з БД. В даному класі реалізовано конструктор, в якому відбувається під’єднання до БД. Робота кожного метода зумовлена прийняттям необхідних параметрів, формуванням в методі відповідної SQL команди з захистом від SQL-ін’єкції та здійснення запиту на її виконання.

В проєкті необхідно було реалізувати клас Template, який є класом-шаблонізатором. Даний клас передбачає такі основні методи:

- *setParams* – даний метод приймає параметри, які будуть передані на html-сторінку. Суть даного методу полягає в занесенні в спеціальну властивість класу за допомогою допоміжного метода *setParam* ім’я параметра та його значення, які потім можуть бути прочитані на html-сторінці та використані їх значення.
- *getHtml* – метод, який призначений для накопичення в буфері інформації (html-коду та параметрів), яка потім використовується в класі *Core* в методі *done*.

В проєкті було необхідно реалізувати клас Controller, який буде батьківським класом для реалізованих мною контролерів. В класі присутні такі методи як:

- render – призначений для повернення html коду разом з параметрами реалізованих нами представлень;
- redirect – метод який призначений для переадресацію користувача за вказаною адресою;
- error - метод, який призначений для створення об'єкту класу Error в який вказується тип помилки та повідомлення.

Наступна логіка реалізації CMS – системи полягає в створенні класа модуля в якому буде реалізовані необхідні методи для проведення потрібних дій з конкретної таблиці БД, файлу представлення в якому буде написаний html – код сторінки та оброблені вхідні параметри. В кінцевому етапі відбувається реалізація контролера, який необхідний для виконання різноманітних перевірок з даними, та виклику необхідних методів моделі, а в кінцевому етапі поверненні необхідного файлу з html – кодом сторінки та параметрів, які використовуються на цій сторінці.

Висновки до другого розділу

Описав, як загальний алгоритм роботи програми так, і основні, реалізовані мною функціональні алгоритми. Навів схему структури системи інтернет-магазину електроніки та схеми структури БД реалізованої мною CMS-системи (фізичний та логічний рівень). Пояснив принцип дії алгоритму основних реалізованих методів та класів даного курсового проєкту.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				28
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3 ОПИС РОБОТИ З ПРОГРАМНИМ ДОДАТКОМ ТА ЙОГО ТЕСТУВАННЯ

3.1 Опис роботи з програмним додатком

Після переходу неавтеризованого користувача на сайт інтернет-магазину, перед ним з'являється вікно, де він може пройти авторизацію (рис. 3.1)

Рис. 3.1. Вікно авторизації користувача

Якщо користувач бажає здійснити реєстрацію, він повинен натиснути кнопку “Реєстрація”, після чого перед ним з’явиться відповідне вікно (рис. 3.2)

Рис. 3.2. Вікно реєстрації користувача

Для здійснення успішної реєстрації користувачу слід вказати прізвище, ім'я та по-батькові, які повинні обов'язково починатися з великої літери та

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				29
Змн.	Арк.	№ докум.	Підпис	Дата		

бути записаними літерами кирилиці. Електронна пошта повинна відповідати стандартам синтаксису та не повинна бути вже використана в реєстрації раніше, а пароль слід придумати такий, який повинен складатися з великих та малих літер англійського алфавіту, з чисел, та не повинен бути довжиною менше 8 символів і повинен співпадати зі своїм повторенням.

Якщо користувач введе неправильно дані при реєстрації, то система сповістить користувача про помилку, інакше він буде перекинутий на форму авторизації, для проходження якої, йому потрібно вказати правильно логін та пароль. Якщо користувач не є адміністратором, то його буде перекинуто на наступну сторінку (3.3)

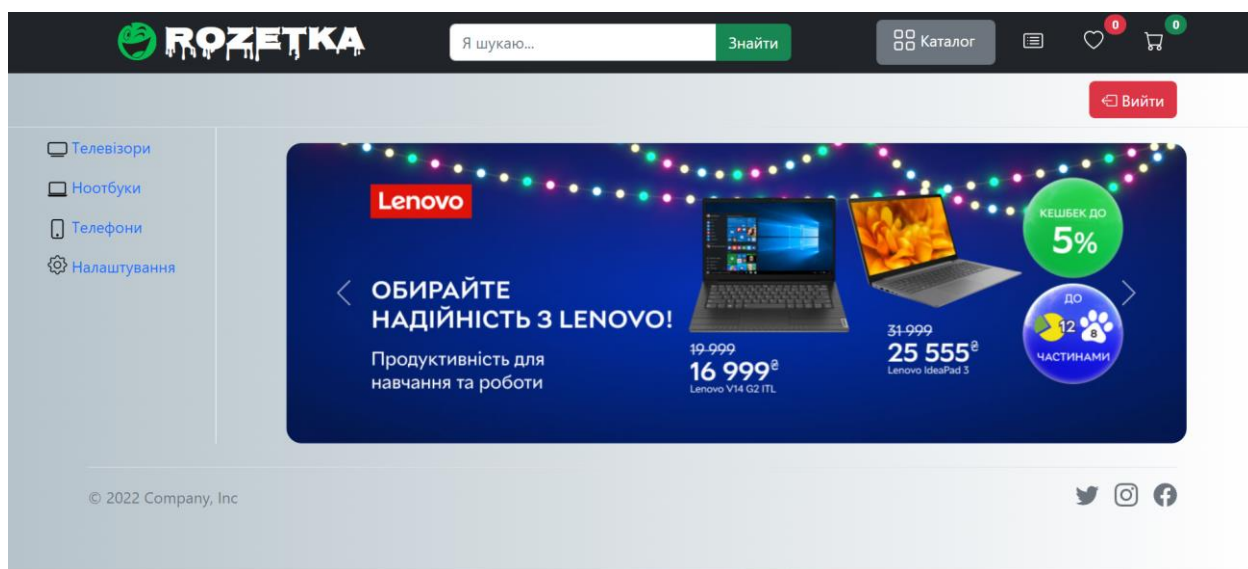


Рис. 3.3. Головне вікно інтернет-магазину (для користувача що не є адміністратором)

Перед користувачем з'являється можливість погортати рекламний слайдер та за бажанням натиснути на відповідну рекламу товару, після чого він буде перекинутий на відповідну сторінку товару.

Зліва, в головному меню інтернет-магазину, розміщені категорії товарів, при кліку на які, користувач буде перекинутий на сторінку з відображенням усіх товарів, що належать даній категорії. (рис. 3.4)

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		30

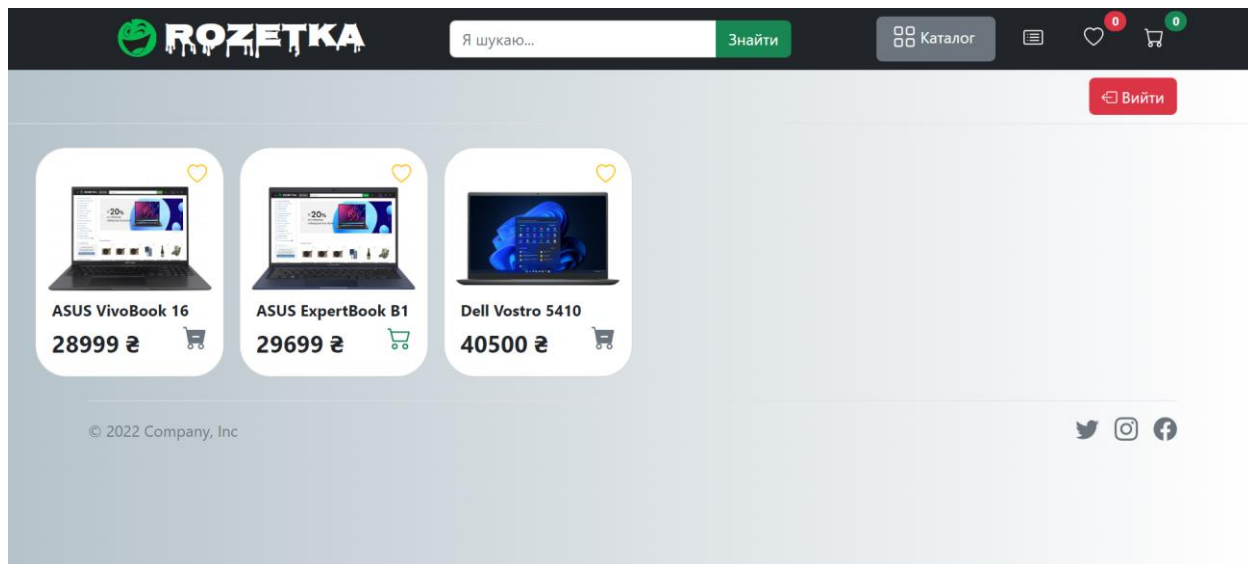


Рис. 3.4. Сторінка з відображенням товарів які належать до категорії
“ноутбуки”

На цій сторінці користувач може додати товар в кошик, натиснувши зелену кнопку, яка розміщена в правому нижньому кутку карточки товару, після чого дана кнопка стане повністю зелена і з’явиться галочка, яка символізує, що даний товар знаходиться в кошику, якщо кнопка на даному місці є сірого кольору, то це сигналізує, що товару немає в наявності і тому користувач не зможе його додати до кошику. Також користувач може додати товари в список бажань, натиснувши на кнопку в вигляді “сердечка”, після чого вона стане повністю замальована в жовтий колір, що є сигналом для користувача про знаходження даного товару в списку бажань. Натиснувши на зображення будь-якого з представлених товарів, користувача буде перекинуто на сторінку з інформацією про товар (рис. 3.5)

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				31
Змн.	Арк.	№ докум.	Підпис	Дата		



Рис. 3.5. Сторінка з відображенням інформації про товар та коментарів

Перед користувачем з'являється опис товару, а також список коментарів (за наявності). Якщо користувач здійснював покупку даного товару раніше та ще не залишав відгук, він має змогу це зробити натиснувши кнопку “Написати відгук”, після чого перед ним з'явиться спеціальне модальне вікно, де він може це зробити (рис 3.6)

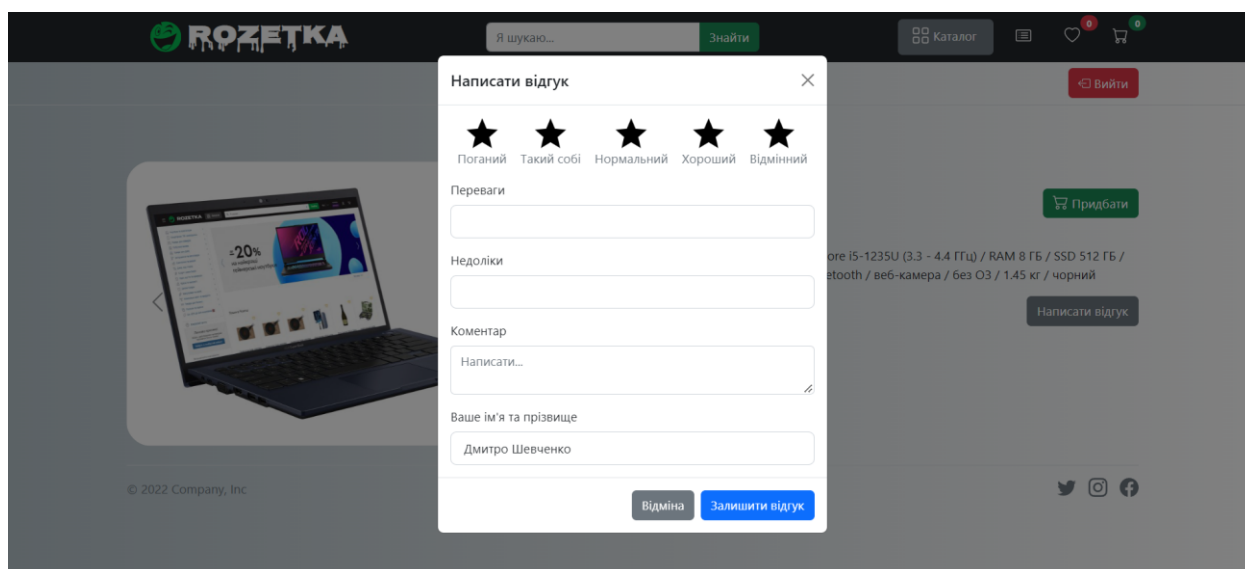


Рис. 3.6. Сторінка з модальним вікном для написання відгуку

В даному вікні потрібно вказати кількість зірок рейтингу, переваги та недоліки (за бажанням), загальний коментар, що є обов'язковим, а також прізвище та ім'я, після чого натиснути кнопку “Залишити відгук” для його надсилення або натиснути кнопку “Відміна”, якщо користувач не хоче надсилати відгук.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				32
Змн.	Арк.	№ докум.	Підпис	Дата		

У верхньому меню користувача (заголовку) знаходиться кнопка, у вигляді списку, після натискання якої, відкривається список замовлень та статус їх обробки, розгорнувши який, можна побачити опис замовлення (рис. 3.7)

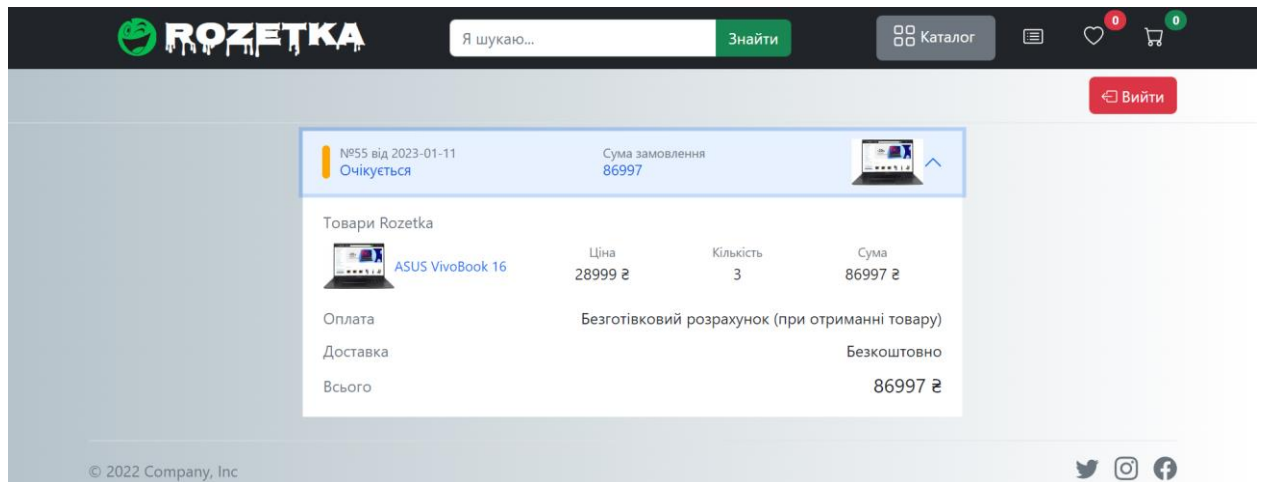


Рис. 3.7. Сторінка з відображенням списку замовлень (історії замовлень)

При натисканні кнопки у вигляді сердечка, у верхньому меню відкриється список товарів, які користувач відправив до списку бажань (рис. 3.8).

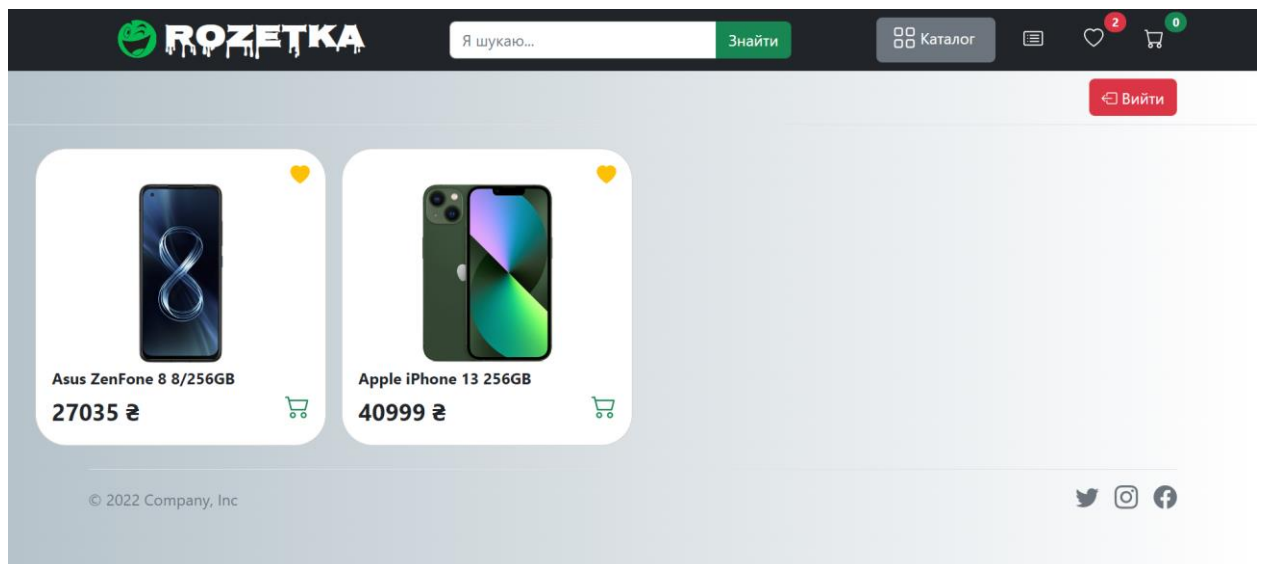


Рис. 3.8. Сторінка з відображенням товарів, які належать до списку бажань

При натисканні кнопки, у вигляді кошика, яка розміщена у верхньому навігаційному меню, відкриється кошик, де можна переглянути товари, які наявні в ньому, а при потребі редагувати потрібну кількість чи взагалі видалити (рис 3.9).

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		33

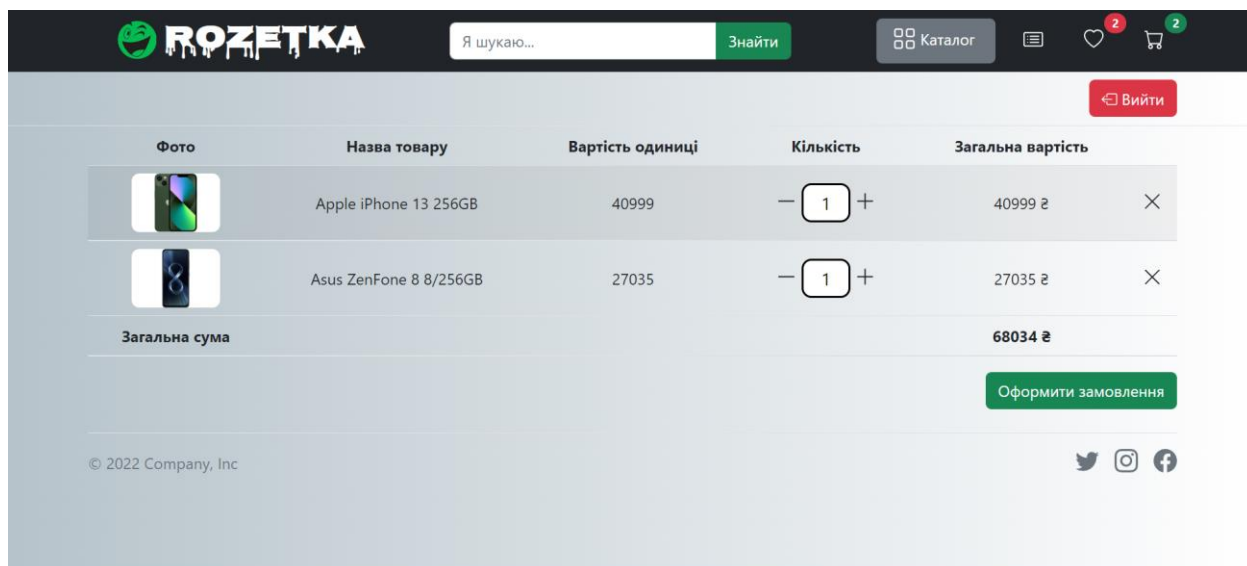


Рис. 3.9. Сторінка з відображенням товарів, які знаходяться в кошику
При натисканні кнопки “оформити замовлення” відбувається перехід на сторінку (рис. 3.10)

Оформлення замовлення

① Ваші контактні дані

Ім'я: Дмитро

Прізвище: Шевченко

Мобільний телефон:

Електронна пошта: volkov.aliksandr03@gmail.com

Замовлення на суму: 68034 ₴

① Товари продавця Rozetka

Товар	Ціна	Кількість	Сума
Apple iPhone 13 256GB	40999 ₴	1	40999 ₴
Asus ZenFone 8 8/256GB	27035 ₴	1	27035 ₴

② Доставка

Оберіть ваше місто:

Оберіть пункт видачі Rozetka:

③ Оплата

☒ Готівковий розрахунок (при отриманні товару)

☐ Безготівковий розрахунок (при отриманні товару)

Всього

2 товари на суму: 68034 ₴

Вартість доставки: 0 ₴

До сплати: 68034 ₴

Замовлення підтверджую

Отримання замовлення від 5 000 ₴ тільки за паспортом (Закон від 06.12.2019 № 361-IX)

Підтверджуючи замовлення, я приймаю умови:

- положення про збирання та захист персональних даних
- користувальницька угода

Рис. 3.10. Сторінка призначена для оформлення замовлення
Також в головному меню користувача розміщений пункт “Налаштування”, при натисканні на який, можна перейти на сторінку налаштувань (рис 3.11)

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				34
Змн.	Арк.	№ докум.	Підпис	Дата		

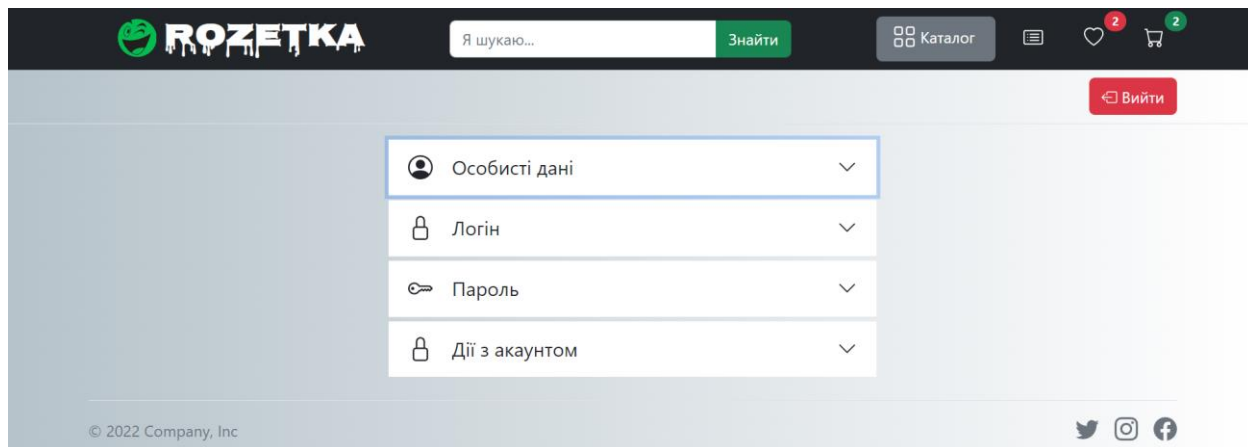


Рис. 3.11. Сторінка призначена для налаштування профілю

Розгорнувши пункт налаштувань “Особисті дані”, можна редагувати ім’я, прізвище та по-батькові. Пункт “Логін” призначений для редагування логіну (електронної пошти) облікового запису. Пункт “Пароль” призначений для зміни паролю і останній пункт “Дії з акаунтом” призначений для видалення поточного акаунта даного магазина.

Якщо авторизований користувач є адміністратором, то перед ним з’явиться наступне вікно головного меню (рис 3.12)

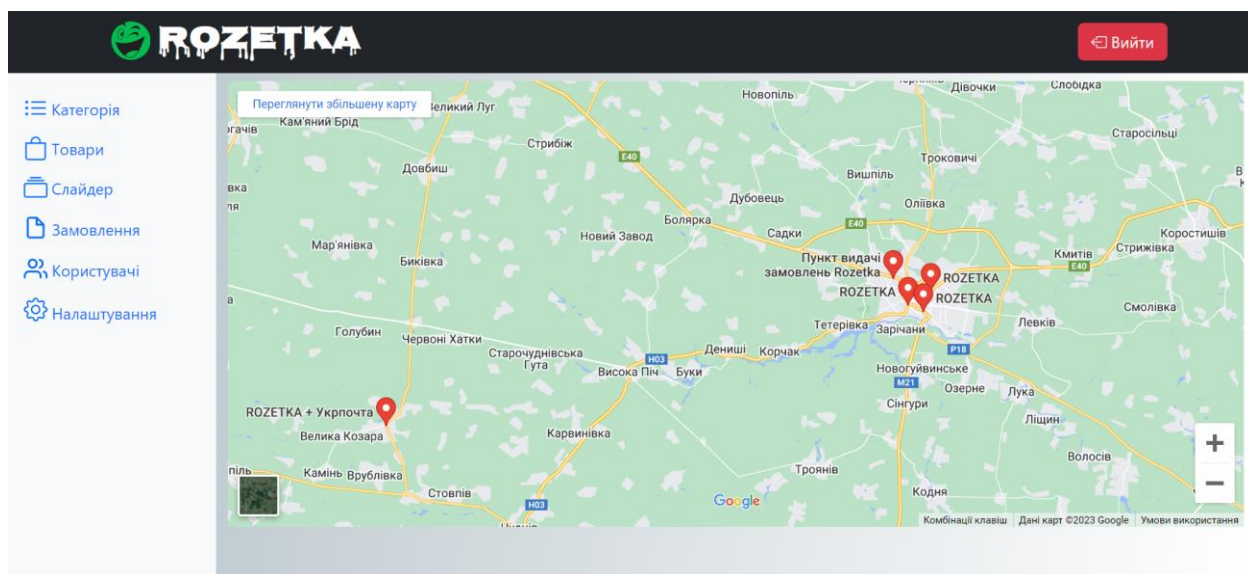


Рис. 3.12. Головне меню адміністратора

На даній сторінці розміщено карта пунктів видачі “Rozetka” з логістичних мотивів та меню дій. Натиснувши пункт меню “Категорія” користувач має можливість додати, змінити або видалити категорію (рис. 3.13), (рис. 3.14), (рис. 3.15) відповідно:

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				35
Змн.	Арк.	№ докум.	Підпис	Дата		

Рис. 3.13. Меню додавання категорії товару

Рис. 3.14. Приклад зміни категорію товару

Рис. 3.15. Приклад видалення категорії товару

В меню вибравши пункт “Товари” можна змінити, додати або видалити товар в категорії (рис. 3.16), (рис. 3.17), (рис. 3.18) відповідно:

ROZETKA Вийти

Категорія

Товари

Слайдер

Замовлення

Користувачі

Налаштування

Додавання товару

Назва товару

Категорія товару

Телевізори

Кількість товару

0

Ціна товару

0

Короткий опис товару

Повний опис товару

Рис. 3.16. Вікно для додавання товару в категорію

ROZETKA Вийти

Категорія

Товари

Слайдер

Замовлення

Користувачі

Налаштування

Зміна товару

Назва товару

ASUS VivoBook 16

Категорія товару

Ноутбуки

Кількість товару

0

Ціна товару

28999

Короткий опис товару

Екран 16" IPS (1920x1200) WUXGA+, матовий / Intel Core i5-1135G7 (2.4 - 4.2 ГГц) / RAM 16 Гб / SSD 512 Гб / Intel Iris Xe Graphics / без ОД / Wi-Fi / Bluetooth / веб-камера ОС / 1.88 кг / чорний

Повний опис товару

Рис. 3.17. Приклад зміни товару в категорії

ROZETKA Вийти

Категорія

Товари

Слайдер

Замовлення

Користувачі

Налаштування

Чи дійсно ви бажаєте видалити товар "ASUS VivoBook 16"?

Після видалення товару, відповідні фото товару будуть видалені!

Видалити Відмінити

Рис. 3.18. Приклад видалення товару в категорії

В пункті меню “Слайдер” можна додати або видалити фото і посилання при кліку на них (рис. 3.19), (рис. 3.20) відповідно:

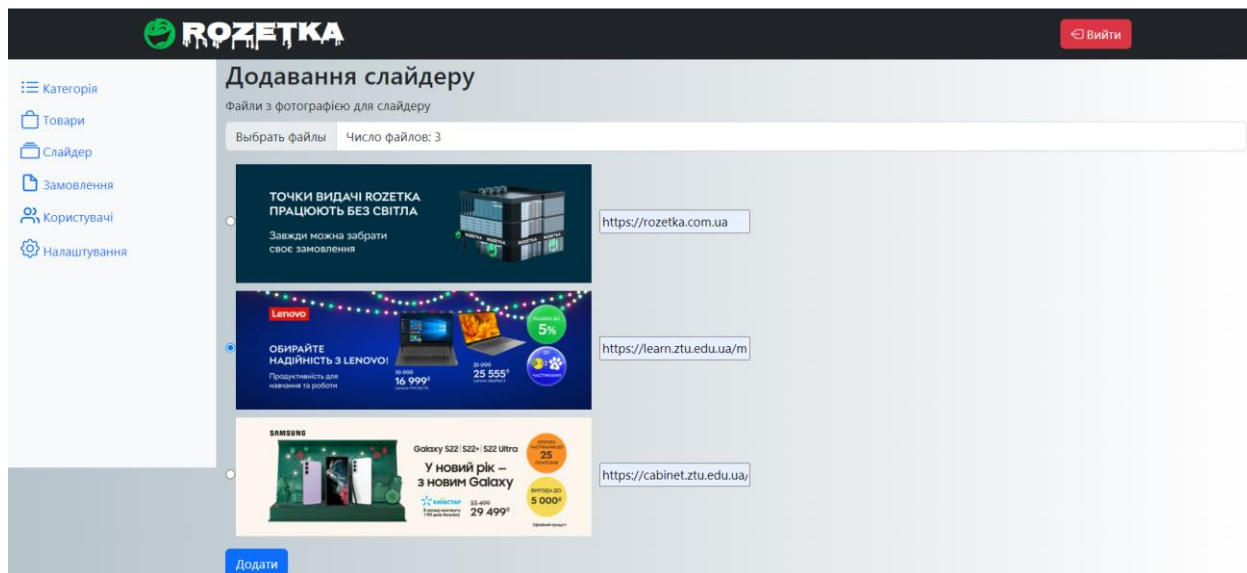


Рис. 3.19. Приклад додавання наповнення слайдеру

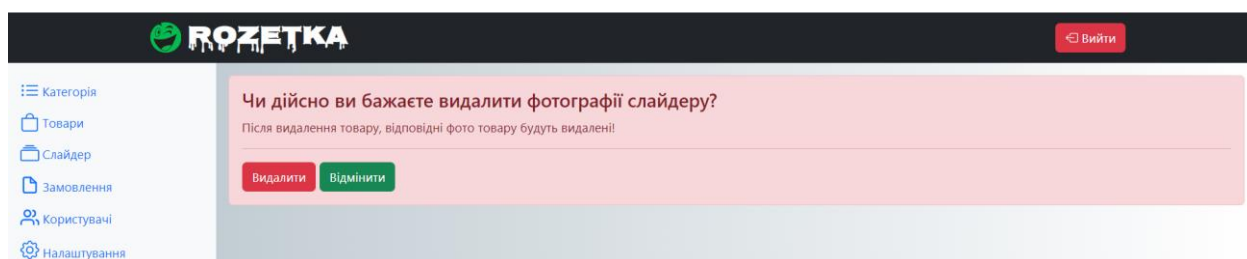


Рис. 3.20. Приклад видалення наповнення слайдеру

У пункті меню “Замовлення” можна переглянути замовлення, які надійшли до магазину та інформацію про замовника та місце доставки (рис. 3.21):

#	Ім'я замовника	Прізвище замовника	Електронна пошта замовника	Номер телефону замовника	Тип оплати	Адреса доставки	Назва товару	Кількість товару	Сума замовлення	
1	Дмитро	Андрійович	volkov.aliksandr03@gmail.com	+380987778821	Безготівковий розрахунок (при отриманні товару)	Велика Бердичівська, 63	ASUS VivoBook 16	3	86997 ₴	✓

Рис. 3.21. Демонстрація таблиці з усіма замовленнями

При натисненні зеленої кнопки в останній колонці таблиці можна змінити статус замовлення (з очікується на виконано).

В пункті меню “Користувачі” можна переглянути всіх зареєстрованих користувачів та інформацію про них, а за потреби змінити їм тип доступу, вибравши його в випадяючому списку (передостання колонка таблиці) та натиснувши зелену кнопку в останній колонці таблиці застосувати зміни.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				38
Змн.	Арк.	№ докум.	Підпис	Дата		

В пункті “Налаштування” можна здійснити налаштування профілю аналогічно описаного вище (налаштування профіля користувача, який не є адміністратор).

3.2 Тестування роботи програмного забезпечення

При введенні некоректних, даних програма розпізнає ці помилки та виводить певне попередження. Наприклад, при введенні неправильних даних в вікно авторизації, програма видасть наступне попередження про помилку (рис. 3.22).

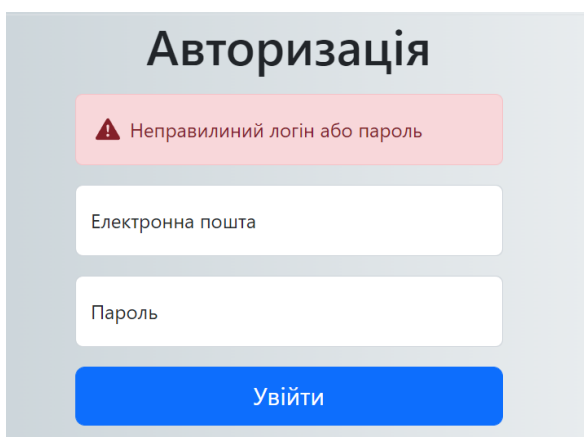


Рис. 3.22. Помилка при введенні неправильних даних для здійснення авторизації

У вікні реєстрації, при некоректних введених даних, також виникає реакція програми у вигляді виділення полів заповнення червоним кольором (рис. 3.23).

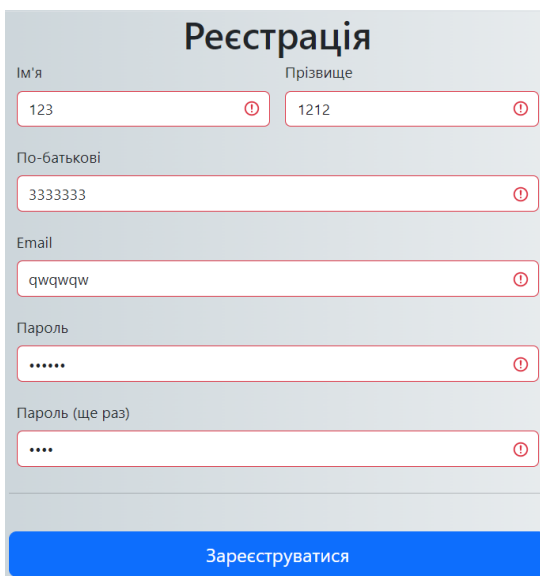


Рис. 3.23. Зміна кольору полів заповнення при неправильно введених даних

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				39
Змн.	Арк.	№ докум.	Підпис	Дата		

Якщо людина при реєстрації введе неправильно заповнені дані та відправить форму на сервер, то після обробки форми сервером під полями для введення даних з'являться наступні повідомлення (рис. 3.24).

Реєстрація

Ім'я: Помилка введені імені!

Прізвище: 1212 Помилка введені по-батькові!

По-батькові: Помилка введені прізвища!

Email: qwqwqw Помилка при введенні електронної пошти!

Пароль: Паролі не співпадають!

Пароль (ще раз): Паролі не співпадають!

[Зареєструватися](#)

Рис. 3.24. Реакція на помилку, яка виникає після відправки форми на сервер з неправильно заповненими полями вводу

При оформленні замовлення, за умови не правильно введених даних і відправки форми на сервер з'являться наступні повідомлення про помилки під полями які не пройшли валідацію (рис. 3.25).

Оформлення замовлення

① Ваші контактні дані

Ім'я: Дмитро Помилка введені імені!

Прізвище: Шевченко

Мобільний телефон: Помилка введення номера мобільного телефону!

Електронна пошта: volkov.aliksandr03@gmail.com

Замовлення на суму: 27035 ₴

① Товари продавця Rozetka

Товар	Ціна	Кількість	Сума
Asus ZenFone 8 8/256GB	27035 ₴	1	27035 ₴

② Доставка

Оберіть ваше місто:

Оберіть пункт видачі Rozetka:

Відсутнє місце доставки товару!

③ Оплата

☒ Готівковий розрахунок (при отриманні товару)

☐ Безготівковий розрахунок (при отриманні товару)

Всього

1 товари на суму 27035 ₴

Вартість доставки 0 ₴

До сплати 27035 ₴

[Замовлення підтверджую](#)

Отримання замовлення від 5 000 ₴ тільки за паспортом (Закон від 06.12.2019 № 361-IX)

Підтверджуючи замовлення, я приймаю умови:

- положення про збирання та захист персональних даних
- користувальницька угода

Рис. 3.25. Помилка, яка виникає при заповненні полів помилковими значеннями

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				40
Змн.	Арк.	№ докум.	Підпис	Дата		

При редагуванні даних профілю в пункті “Налаштування” за умови неправильно введених даних з’являться наступні повідомлення про помилки (рис. 3.26), (рис. 3.27) відповідно:

The screenshot shows a form titled "Особисті дані" (Personal Data). It contains three input fields: "Ім'я" (Name) with value "111", "Прізвище" (Surname) with value "222", and "По-батькові" (Patronymic) with value "333". Each field has a red border and a red exclamation mark icon indicating an error. Below the fields are two buttons: "Змінити" (Change) in blue and "Відміна" (Cancel) in red.

Рис. 3.26. Помилка, яка виникає при заповненні полів підпункту “Особисті дані” помилковими значеннями

The screenshot shows a form titled "Логін" (Login). It contains one input field labeled "Логін:" with value "1111m". The field has a red border and a red exclamation mark icon indicating an error. To the right of the field is a blue button labeled "Змінити" (Change).

Рис. 3.27. Помилка, яка виникає при заповненні поля підпункту “Логін” помилковими значеннями

При додавання категорії, за умови неправильно введених даних відображаються наступні повідомлення про помилку під полями, які не пройшли валідацію (рис. 3.28)

The screenshot shows a form titled "Додавання категорії" (Add Category). It contains two input fields: "Назва категорії" (Category Name) and "Файл з фотографією для категорії" (File with photo for category). The "Назва категорії" field has a red border and a red message "Відсутня назва категорії" (Category name is missing) below it. The "Файл з фотографією для категорії" field has a red border and a message "Виберіть файл" (Select file) and "Файл не вибран" (File not selected) below it. Below the fields is a blue button labeled "Створити категорію" (Create category).

Рис. 3.28. Помилка, яка виникає при заповненні полів для додавання категорії неправильними даними

При зміні категорії і вказанні неправильних даних буде відображено наступне повідомлення про помилку (рис. 3.29)

Зміна категорії

Назва категорії

Відсутня назва категорії

Файл з фотографією для категорії

Виберите файл

Рис. 3.29. Помилка, яка виникає при заповненні полів для редагування категорії неправильними даними

При додаванні товару в категорію і запис у поля неправильних даних з'являться наступні повідомлення про помилки (рис. 3.30)

Додавання товару

Назва товару

Відсутня назва товару

Категорія товару

Телевізори

Кількість товару

Ціна товару

Некоректно задана ціна товару

Короткий опис товару

Paragraph
B I @ :: = =

Повний опис товару

Paragraph
B I @ :: = =

Відображення товару покупцеві

Відображати

Файли з фотографією для товару

Вибрати файлы

Рис. 3.30. Реація програми на помилки, які пов'язані з некоректно введеними даними, при додаванні товару до категорії

При редагуванні опису товари і відправки форми на сервер з некоректно заповненими полями виникнуть наступні оповіщення системи про помилки (рис. 3.31)

Кількість товару
0
Відсутня кількість товару
Ціна товару
28999
Відсутня ціна товару

Рис. 3.31. Реакція програми на помилки, які пов'язані з некоректно введеними даними, при редагуванні товару в категорії

Висновки до третього розділу

Було проведено детальний опис роботи програмного продукту. Також було проведено опис місць виникнення можливих помилок та реакцію програми на них. Завдяки врахуванню всіх можливих варіантів роботи програми, вдалося уникнути непередбачуваних результатів роботи програмного продукту. Наведено скріншоти, що підтверджують описані дії даного розділу.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				43
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

Під час виконання курсового проєкту було отримано практичні навички побудови CMS-системи, досвід використання мов PHP, JS, HTML та таблиці стилів CSS.

Було детально проведено аналіз поставленої задачі та обрано засоби і методи її рішення, провів порівняльний аналіз схожих існуючих програмних додатків та навів їхні переваги та недоліки.

Було спроектовано загальний алгоритм роботи програми, що допоможе краще зрозуміти принцип дії програмного продукту. Розроблено, описано та пояснено принцип дії основних функціональних алгоритмів роботи програми та наведено скріншоти діаграми побудованої мною бази даних. Пояснено алгоритм роботи найважливіших реалізованих методів програмного додатку.

Було детально описано роботу програмного продукту та проведено його тестування.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				44
Змн.	Арк.	№ докум.	Підпис	Дата		

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. admin. Выборка данных. Команда SELECT [Электронный ресурс]/admin – 2017. – Режим доступа до ресурсу: <https://metanit.com/sql/sqlserver/4.2.php>
2. admin. Объекты и классы [Электронный ресурс]/admin – 2021. – Режим доступа до ресурсу: <https://metanit.com/php/tutorial/6.1.php>
3. admin. Объекты и классы [Электронный ресурс]/admin – 2021. – Режим доступа до ресурсу: <https://metanit.com/php/tutorial/6.1.php>
4. DOM-дерево [электронный ресурс] - 2021. Режим доступа: <https://learn.javascript.ru/dom-nodes>
5. Атрибуты и свойства [электронный ресурс] - 2022. Режим доступа: <https://learn.javascript.ru/dom-attributes-and-properties>
6. JSON.parse() [электронный ресурс] - 2022. Режим доступа: https://developer.mozilla.org/enUS/docs/Web/JavaScript/Reference/Global_Objects/JSON/parse
7. AJAX - объект XMLHttpRequest [электронный ресурс] - 2022. Режим доступа: https://html5css.ru/js/js_ajax_http.php
8. Методичні рекомендації для виконання курсової роботи [Електронний ресурс]/ Освітній портал ДУ “Житомирська політехніка” – 2022. – Режим доступу до ресурсу: https://learn.ztu.edu.ua/pluginfile.php/255836/mod_resource/content
9. Флэнаган Д. JavaScript. Подробное руководство, 6-е издание – Пер. с англ. – СПб: Символ-Плюс, 2012. – 1080 с.
10. Фрэйн, Б. HTML5 и CSS3. Разработка сайтов для любых браузеров и устройств / Б. Фрэйн ; [перевод с английского В. Черник]. - Санкт-Петербург [и др.] : Питер, 2014. - 298 с.

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				45
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

		Олександр ВОЛКОВ			ДУ «Житомирська політехніка».23.121.07.000 - ПЗ	Арк.
		Денис ФУРІХАТА				
Змн.	Арк.	№ докум.	Підпис	Дата		46

Клас Core:

```

<? php

namespace core;

use controllers\SiteController;
use models\User;

/**
 * Головний клас ядра системи
 * Singleton
 */
class Core
{
    private static ? Core $instance = null;
    public array $application;
    public DB $db;
    public $requestMethod;
    public array $pageParams;

    private function __construct()
    {
        global $pageParams;
        $this->application = [];
        $this->pageParams = $pageParams;
    }

    /**
     * Повертає екземпляр ядра системи
     * @return Core
     */
    public static function getInstance()
    {
        if (empty(self::$instance))
        {
            self::$instance = new Core();
        }
        return self::$instance;
    }

    /**
     * Ініціалізація системи
     */
    public function initialize()
    {
        session_start();
        $this->db = new DB(DATABASE_HOST, DATABASE_LOGIN, DATABASE_PASSWORD, DATA-
BASE_BASENAME);
        $this->requestMethod = $_SERVER['REQUEST_METHOD'];
    }

    /**
     * Виконує основний процес роботи CMS системи
     */
    public function run()
    {
        session_start();
        $path = $_GET['path'];
        $pathParts = explode('/', $path);
        $moduleName = strtolower(array_shift($pathParts));
        $actionName = strtolower(array_shift($pathParts));
        $statusCode = 200;
    }
}

```

```

if (empty($moduleName))
    $moduleName = "site";
if (empty($actionName))
    if (User::isUserAdmin())
    {
        $actionName = "admin";
    }
    else
    {
        $actionName = "index";
    }

$this->application['moduleName'] = $moduleName;
$this->application['actionName'] = $actionName;
$controllerName = '\controllers\' . ucfirst($moduleName) . 'Controller';
$actionName = $actionName . 'Action';
if (class_exists($controllerName))
{
    $controller = new $controllerName();
    if (method_exists($controller, $actionName))
    {
        $actionResult = $controller->$actionName($pathParts);
        if ($actionResult instanceof Error)
            $statusCode = $actionResult->code;
        $this->pageParams['content'] = $actionResult;
    }
    else
    {
        $statusCode = 404;
    }
}
else
{
    $statusCode = 404;
}
$statusCodeType = intval($statusCode / 100);
if ($statusCodeType == 4 || $statusCodeType == 5) {
    $siteController = new SiteController();
    $this->pageParams['content'] = $siteController->errorAction($statusCode);
}
}

/**
 * Завершення роботи системи та виведення результату
 */
public function done()
{
    $pathToMainPage = 'themes/light/mainPage.php';
    $tpl = new Template($pathToMainPage);
    $tpl->setParams($this->pageParams);
    $html = $tpl->getHTML();
    echo $html;
}
}

```

Клас DB:

```

<? php

namespace core;
/**
 * Клас до виконання запитів до БД
 */
class DB
{
    protected $pdo;
}

```



```

public function __construct($server, $login, $password, $database)
{
    $this->pdo = new \PDO("mysql:host={$server};dbname={$database};", $login, $password);
}

/**
 * @param $conditionsArray
 * @param string $sql
 * @return array
 */
public function extracted($conditionsArray, string $sql): array
{
    if (is_array($conditionsArray) && count($conditionsArray) > 0) {
        $whereParts = [];
        foreach ($conditionsArray as $key => $value) {
            $whereParts[] = "{$key} = ?";
        }
        $whereStr = implode(' AND ', $whereParts);
        $sql.= ' WHERE ' . $whereStr;
    }

    if (is_string($conditionsArray))
        $sql.= ' WHERE ' . $conditionsArray;
    return array($key, $value, $sql);
}

/**
 * Виконання запиту на отримання даних з вказаної таблиці БД
 * @param string $tableName Назва таблиці бази даних
 * @param string|array $fieldsList Список полів
 * @param array|null $conditionArray Асоціативний масив з умовою для пошуку
 * @return array|false
 */

public function select($tableName, $fieldsList = "*", $conditionsArray = null, array
$conditionLikeArray = null, $orderByArray = null, $limit = null, $offset = null)
{
    $fieldsStr = "*";
    if (is_string($fieldsList))
    {
        $fieldsStr = $fieldsList;
    }
    if (is_array($fieldsList))
    {
        $fieldsStr = implode(', ', $fieldsList);
    }
    $sql = "SELECT {$fieldsStr} FROM {$tableName}";
    list($key, $value, $sql) = $this->extracted($conditionsArray, $sql);

    if (empty($conditionsArray) && !empty($conditionLikeArray))
    {
        list($key, $value, $sql) = $this->extractedLike($conditionLikeArray, $sql);
    }

    if (is_array($orderByArray))
    {
        $orderByParts = [];
        foreach ($orderByArray as $key => $value) {
            $orderByParts[] = "{$key} {$value}";
        }

        $sql.= ' ORDER BY ' . implode(', ', $orderByParts);
    }
}

```

```

        if (!empty($limit))
    {
        if (!empty($offset))
        {
            $sql.= " LIMIT {$offset}, {$limit}";
        }
        else
        {
            $sql.= " LIMIT {$limit}";
        }
    }

    $res = $this->pdo->prepare($sql);
    if (is_array($conditionsArray) && count($conditionsArray) > 0)
        $res->execute(array_values($conditionsArray));
    else if (isset($conditionLikeArray) && count($conditionLikeArray) > 0)
        $res->execute(array_values($conditionLikeArray));
    else
        $res->execute();
    return $res->fetchAll(PDO::FETCH_ASSOC);
}

public function insert($tableName, $fieldsList)
{
    $fieldsArray = array_keys($fieldsList);

    $fieldsListString = implode(', ', $fieldsArray);

    $paramsArray = [];
    foreach ($fieldsList as $key => $value) {
        $paramsArray[] = ':' . $key;
    }

    $valuesListString = implode(', ', $paramsArray);
    $res = $this->pdo->prepare("INSERT INTO {$tableName} ($fieldsListString) VALUES($valuesListString)");
    $res->execute($fieldsList);
}

public function delete($table, $conditionArray = null)
{
    $sql = "DELETE FROM {$table}";
    list($key, $value, $sql) = $this->extracted($conditionArray, $sql);
    $res = $this->pdo->prepare($sql);
    if (is_array($conditionArray) && count($conditionArray) > 0)
        $res->execute(array_values($conditionArray));
    else
        $res->execute();
}

public function update($table, $fieldsList, $conditionArray)
{
    $sql = "UPDATE {$table} SET ";
    $setParts = [];
    $paramsArr = [];
    foreach ($fieldsList as $key => $value) {
        $setParts[] = "{$key} = ?";
        $paramsArr[] = $value;
    }

    $sql.= implode(', ', $setParts);
    if (is_array($conditionArray) && count($conditionArray) > 0)
    {
        $whereParts = [];
        foreach ($conditionArray as $key => $value) {
            $whereParts[] = "{$key} = ?";
        }
    }
}

```

```

        $paramsArr[] = $value;
    }

    $whereStr = implode(' AND ', $whereParts);
    $sql.= ' WHERE '. $whereStr;
}
if (is_string($conditionArray))
    $sql.= ' WHERE '. $conditionArray;
$res = $this->pdo->prepare($sql);
$res->execute($paramsArr);
}

public function extractedLike($conditionsArray, string $sql): array
{
    if (is_array($conditionsArray) && count($conditionsArray) > 0) {
        $whereParts = [];
        foreach ($conditionsArray as $key => $value) {
            $whereParts[] = "{$key} LIKE ?";
        }
        $whereStr = implode(' AND ', $whereParts);
        $sql .= ' WHERE ' . $whereStr;
    }

    if (is_string($conditionsArray))
        $sql .= ' WHERE ' . $conditionsArray;
    return array($key, $value, $sql);
}
}

```

Клас Template:

```

class Template
{
    protected array $parameters;
    protected $path;
    public function __construct($path)
    {
        $this->parameters = [];
        $this->path = $path;
    }

    public function setParam($name,$value)
    {
        $this->parameters[$name] = $value;
    }

    public function setParams($params)
    {
        foreach ($params as $name =>$value){
            $this->setParam($name,$value);
        }
    }
    public function getHTML()
    {
        ob_start();
        extract($this->parameters);
        include($this->path);
        $html = ob_get_contents();
        ob_end_clean();
        return $html;
    }
}

```

Клас Template:

```
<? php

namespace core;

/**
 * Базовий клас для всіх контролерів
 * @package core
 */
class Controller
{
    protected string $viewPath;
    protected $moduleName;
    protected $actionName;

    public function __construct()
    {
        $this->moduleName = Core::getInstance()->application['moduleName'];
        $this->actionName = Core::getInstance()->application['actionName'];
        $this->viewPath = "views/{$this->moduleName}/{$this->actionName}.php";
    }

    public function render($viewPath = null, $params = null)
    {
        if (empty($viewPath))
            $viewPath = $this->viewPath;
        $tpl = new Template($viewPath);
        if (!empty($params))
            $tpl->setParams($params);
        return $tpl->getHTML();
    }

    public function renderView($viewName)
    {
        $path = "views/{$this->moduleName}/{$viewName}.php";
        $tpl = new Template($path);
        if (!empty($params))
        {
            $tpl->setParams($params);
        }
        return $tpl->getHTML();
    }

    public static function redirect($url)
    {
        header("Location: {$url}");
        die;
    }

    public static function error($type, $message = null): Error
    {
        return new Error($type, $message);
    }
}
```

Клас SiteController:

```
<? php

namespace controllers;

use core\Controller;
```

```

use models\Basket;
use models\Category;
use models\Slider;

class SiteController extends Controller
{
    public function indexAction()
    {
        $rows = Category::getCategories();

        $slider = Slider::getPhoto();
        return $this->render(null, [
            'rows' => $rows,
            'slider' => $slider
        ]);
    }

    public function adminAction()
    {
        $rows = Category::getCategories();
        return $this->render(null, [
            'rows' => $rows
        ]);
    }

    public function errorAction($code)
    {
        switch ($code) {
            case 404:
                return $this->render('views/site/error-404.php');
                break;
            case 403:
                return $this->render('views/site/error-404.php');
                break;
        }
    }
}

```

Клас UserController:

```

<? php

namespace controllers;

use core\Controller;
use core\Core;
use models\Basket;
use models\Category;
use models\Order;
use models\User;
use models\Wish;

class UserController extends Controller
{
    public function indexAction()
    {
        $rows = Category::getCategories();
        return $this->render(null, [
            'rows' => $rows
        ]);
    }

    public function registerAction()
    {

```

```

        if (User::isAuthenticatedUser())
        {
            $this->redirect('/');
        }
        if (Core::getInstance()->requestMethod === 'POST')
        {
            $errors = [];
            if (!filter_var($_POST['login'], FILTER_VALIDATE_EMAIL))
                $errors['login'] = 'Помилка при введенні електронної пошти!';
            if (User::isLoginExist($_POST['login']))
                $errors['login'] = 'Даний логін є зайнятий!';

            if ($_POST['password1'] !== $_POST['password2'] && !preg_match('/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$/', $_POST['password1'])) {
                $errors['password'] = 'Паролі не співпадають!';
            }

            $patternName = '/[A-Яİİ][a-яіі]+/';
            if (!preg_match($patternName, trim($_POST['firstname'])))
            {
                $errors['firstname'] = "Помилка введені імені!";
            }
            if (!preg_match($patternName, trim($_POST['middlename'])))
            {
                $errors['middlename'] = "Помилка введені прізвища!";
            }
            if (!preg_match($patternName, trim($_POST['lastname'])))
            {
                $errors['lastname'] = "Помилка введені по-батькові!";
            }

            if (count($errors) > 0)
            {
                $model = $_POST;
                return $this->render(null, [
                    'errors' => $errors,
                    'model' => $model
                ]);
            }
            else
            {
                User::addUser($_POST['login'], $_POST['password1'], $_POST['firstname'],
                    $_POST['middlename'], $_POST['lastname']);
                return $this->renderView('register_success');
            }
        }
        else
        {
            return $this->render();
        }
    }

    public function loginAction()
    {
        if (User::isAuthenticatedUser())
        {
            $this->redirect('/');
        }
        $error = null;
        if (Core::getInstance()->requestMethod === 'POST')
        {
            $user = User::getUserByLoginAndPassword($_POST['login'], $_POST['password']);
            if (empty($user))

```

```

        {
            $error = 'Неправильний логін або пароль';
        }
        else
        {
            User::authenticationUser($user);
            Basket::initializeBasket();
            Wish::initializeWishList();
            $this->redirect('/');
        }
    }
    return $this->render(null, [
        'error' => $error
    ]);
}

public function logoutAction()
{
    if (User::isAuthenticatedUser() && !User::isUserAdmin())
    {
        $userId = User::getCarrentAuthenticatedUser()['id'];
        Basket::updateBasketInDB($userId);
        Wish::updateWishListInDB($userId);
    }
    User::logoutUser();
    $this->redirect('/user/login');
}

public function chartAction()
{
    if (!User::isUserAdmin())
    {
        return $this->error(403);
    }
    $users = User::selectUser();
    return $this->render(null, [
        'users'=>$users
    ]);
}

public function accessAction()
{
    if (!User::isUserAdmin())
    {
        return $this->error(403);
    }
    User::updateUser(intval($_GET['id']), [
        "typeAccess" => intval($_GET['type'])
    ]);
    return $this->render();
}

public function settingsAction()
{
    $userId = User::getCarrentAuthenticatedUser()['id'];
    if (Core::getInstance()->requestMethod == "POST")
    {
        if (User::isAuthenticatedUser())
        {
            $firstName = trim($_POST['firstName']);
            $middleName = trim($_POST['middleName']);
            $lastName = trim($_POST['lastName']);
            $login = trim($_POST['loginUserEdit']);
            $password = trim($_POST['password']);

```

```

        $passwordRepeat = trim($_POST['passwordRepeat']);
        $patternName = '/[A-Яİİ][a-яіі]+/';
        if (preg_match($patternName,$firstName) &&
preg_match($patternName,$middleName) && preg_match($patternName,$lastName))
        {
            User::updateUser($userId, [
                'firstName' => $firstName,
                'middleName'=>$middleName,
                'lastName'=>$lastName,

                ]);
        }
        else if (filter_var($login, FILTER_VALIDATE_EMAIL) && !User::isLoginExist($login))
        {
            User::updateUser($userId, [
                'login'=>$login
                ]);
        }
        else if ($password == $passwordRepeat && preg_match('/^(?=.*[a-z])(?=.*[A-Z])(?=.*\d)[a-zA-Z\d]{8,}$/',$password)){

            User::updateUser($userId, [
                'password'=>User::hashPassword($password)
                ]);
        }
    }
}

$user = User::selectUser('*', [
    'id'=>$userId
]);
return $this->render(null, [
    'user' => $user[0],

]);
}

public function deleteAction($params)
{
    $confirmDeleting = boolval($params[0] == 'confirm');
    if ($confirmDeleting) {
        User::deleteUser([
            'id' => User::getCarrentAuthenticatedUser()['id']
        ]);
        $this->redirect('/user/logout');

    }
    $userId = User::getCarrentAuthenticatedUser()['id'];
    return $this->render(null, [
        'id'=>$userId
    ]);
}

public function orderAction()
{
    $orders = Order::getAllOrders();
    if ($orders != null){
        return $this->render(null, [
            'orders'=>$orders
        ]);
    }
    return $this->render();
}
}

```