

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

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

«Інтернет магазин дитячих іграшок»

Виконав студент групи ІПЗ-20-4
КОНОНЧУК Богдан Володимирович

Керівник роботи:
КУБРАК Юрій Олександрович

Консультант:
ЄФРЕМОВ Юрій Миколайович

Рецензент:
САВІЦЬКИЙ Роман Святославович

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

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«26» лютого 2024 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОНОНЧУК Богдан Володимирович**

Керівник роботи: КУБРАК Юрій Олександрович

Тема роботи: **«Інтернет магазин дитячих іграшок»**,

затверджена Наказом закладу вищої освіти від «23» лютого 2024 р., №74/с

Вихідні дані для роботи: **інтернет-магазин дитячих іграшок**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Кубрак Ю.О., Єфремов Ю. М.	26.02.2024р.	26.02.2024р.
2	Кубрак Ю.О., Єфремов Ю. М.	12.04.2024р.	12.04.2024р.
3	Кубрак Ю.О., Єфремов Ю. М.	03.05.2024р.	03.05.2024р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з комплексної програмної реалізації веб-сайту інтернет-магазину дитячих іграшок, а також пояснювальної записки. Пояснювальна записка до дипломної роботи містить 63 сторінок, 36 ілюстрацій, 4 таблиці, 30 літературних джерел та додатки.

В роботі визначено ключові завдання, що пов'язані з розробкою веб-сайту, проведено аналіз існуючого інструментарію, подібного до розроблюваного, а також наведено сценарії роботи програмного продукту. Представлено огляд загальної структури додатку, описано алгоритми взаємодії між окремими компонентами, а також наведено опис об'єктної структури (діаграму класів) додатку.

Ключові слова: ВЕБ-САЙТ, ВЕБ-ДОДАТОК, ІНТЕРНЕТ-МАГАЗИН, CMS, MVC.

					ІПЗ.КР.Б. – 121 – 24 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Конончук Б.В.			Інтернет-магазин дитячих іграшок Пояснювальна записка	Літ.	Арк.	Аркушів
Керівник		Кубрак Ю.О.						
Консультант		Єфремов Ю.М.					3	80
Рецензент.		Савіцький Р.С.				Житомирська політехніка, група ІПЗ 20-4		
Зав. каф.								

ABSTRACT

The final bachelor's thesis consists of a comprehensive software implementation of an online toy store website and an explanatory note. The explanatory note to the thesis contains 63 pages, 36 illustrations, 4 tables, 30 references, and appendices.

The work identifies the key tasks related to the development of a website, analyzes existing tools similar to the one being developed, and provides scenarios for the operation of the software product. An overview of the general structure of the application is presented, the algorithms of interaction between individual components are described, and the object structure (class diagram) of the application is described.

Keywords: WEBSITE, WEB APPLICATION, INTERNET SHOP, CMS, MVC.

					ИПЗ.КР.Б. – 121 – 24 – ИПЗ	
						4

Зміст

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	7
Вступ	8
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ ІНТЕРНЕТ-МАГАЗИНУ ІГРАШОК.....	10
1.1 Постановка задачі	10
1.2 Аналіз аналогів програмного продукту.....	12
1.3 Вибір архітектури інтернет-магазину	15
1.4 Обґрунтування вибору інструментальних засобів	18
Висновки до розділу 1.....	22
РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-САЙТУ ІНТЕРНЕТ-МАГАЗИНУ ДИТЯЧИХ ІГРАШОК	23
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	23
2.2 Розробка бази даних системи	31
2.3 Проектування та реалізація алгоритмів роботи системи	34
2.4 Реалізація веб-сайту інтернет-магазину дитячих іграшок	39
Висновки до розділу 2.....	44
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-САЙТОМ ІНТЕРНЕТ- МАГАЗИНУ ДИТЯЧИХ ІГРАШОК	45
3.1 Порядок налаштування параметрів системи	45
3.2 Структура інтерфейсу веб-сайту інтернет-магазину дитячих іграшок	47

3.3 Тестування роботи програмного комплексу	53
Висновки до розділу 3.....	58
ВИСНОВКИ.....	59
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	61
ДОДАТКИ.....	64

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

CMS – Content management system;

MVC – Model-View-Controller;

ОС – операційна система;

ПЗ – програмне забезпечення;

СУБД – система управління базами даних;

Вступ

Актуальність теми. Розробка інтернет-магазину дитячих іграшок є актуальною через зростаючий попит на онлайн-торгівлю та необхідність створення зручних платформ для купівлі дитячих товарів. В умовах сучасного цифрового світу, де багато батьків шукають іграшки для своїх дітей в Інтернеті, важливо забезпечити зручний, швидкий та безпечний спосіб купівлі. Це дозволяє зекономити час, отримати доступ до широкого асортименту та порівняти ціни на різних платформах [1].

Створення інтернет-магазину дитячих іграшок включає в себе аналіз сучасних тенденцій в електронній комерції, інтеграцію з платіжними системами, забезпечення безпеки даних користувачів, а також розробку зручного інтерфейсу для покупців. Такий проект може мати велике практичне значення, сприяючи розвитку бізнесу і задоволенню потреб споживачів [2].

Мета роботи: Розробка алгоритмів та реалізація програмного комплексу інтернет-магазину дитячих іграшок, що забезпечує зручний та ефективний процес купівлі-продажу.

Завдання роботи: Проведення аналізу вимог до інтернет-магазину дитячих іграшок. Визначення архітектури та узагальненої структури системи. Обґрунтування та вибір засобів реалізації системи. Розробка інформаційного та математичного забезпечення системи. Проектування структурних складових та алгоритмів роботи системи. Реалізація програмного комплексу та проведення тестування.

Об'єкт дослідження: Автоматизація процесів купівлі-продажу в інтернет-магазині дитячих іграшок.

Предмет дослідження: Використання веб-орієнтованих технологій для розробки системи інтернет-магазину дитячих іграшок.

Для досягнення поставлених цілей використовувалися методи системного аналізу, об'єктно-орієнтованого проектування, веб-програмування та дизайну.

Інформаційною базою дослідження стали наукові публікації, технічна документація, а також сучасні стандарти та рекомендації у галузі розробки веб-орієнтованих систем.

Розроблений програмний комплекс інтернет-магазину дитячих іграшок має високу практичну цінність, оскільки дозволяє суттєво спростити та автоматизувати процеси купівлі-продажу, забезпечуючи високий рівень зручності для користувачів та ефективність для підприємств.

Випускна кваліфікаційна робота бакалавра складається з вступу, трьох розділів, висновків, списку використаних джерел із найменувань, додатків, 4 таблиці та 36 ілюстрацій по тексту. Повний обсяг роботи становить сторінок, з них сторінок основного тексту.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ ІНТЕРНЕТ-МАГАЗИНУ ІГРАШОК

1.1 Постановка задачі

На сьогоднішній день майже кожна комерційна організація має власний інтернет-магазин. В умовах використання сучасних інформаційних технологій – це необхідний інструмент, що дозволяє розширити поле рекламної діяльності та залучити тим самим нових клієнтів, які зможуть реєструватися на сайті, купувати обрані товари та залишати відгуки.

Автоматизація бізнес-процесів – це впровадження програмної системи, яка виконує типові процедури по сучасним алгоритмам, таким як складання і виписка документів, контроль виконання облікових, складських операцій. Завдяки подібним нововведенням рівень роботи підприємства якісно підвищується [3].

Розбираючись в тому, що таке автоматизація бізнесу і що вона дає, слід зазначити наступні позитивні аспекти її введення:

- Ефективність: Зменшення часу, необхідного для виконання завдань, та зниження ймовірності помилок;
- Економія ресурсів: Зменшення потреби в людських ресурсах для виконання рутинних операцій;
- Покращення якості: Забезпечення стабільності та консистентності у виконанні завдань;
- Аналітика та звітність: Збір та аналіз даних для прийняття кращих управлінських рішень;
- Адаптивність: Здатність швидко реагувати на зміни у бізнес-середовищі;
- Контроль: Підвищення рівня контролю над процесами та ресурсами.

Успішний сайт для Інтернет-продажів має велику кількість складових, які сприяють зручності для користувачів, довіри покупців та ефективній роботі магазину.

Ось деякі ключові аспекти, які важливі для успішних Інтернет-продажів:

1. Привабливий та інтуїтивно зрозумілий дизайн:

- Привабливий та професійний вигляд.
 - Легка навігація та інтуїтивно зрозумілі категорії.
2. Швидка та ефективна сторінка завантаження:
- Мінімізація часу завантаження сторінок.
 - Оптимізація для мобільних пристроїв.
3. Безпека та довіра:
- Використання SSL-шифрування для захисту особистої інформації покупців [4].
 - Наявність політики конфіденційності та умов користування.
4. Зручний процес оформлення замовлення:
- Мінімізація кількості кроків для оформлення замовлення.
 - Зручні опції оплати та доставки.
5. Якісні зображення та опис товарів:
- Чіткі та високоякісні фотографії товарів.
 - Інформативні та зрозумілі описи товарів.
6. Відгуки та рейтинги:
- Можливість залишати відгуки покупцями.
 - Відображення рейтингу та відгуків на сторінці товару.
7. Програма лояльності та знижки:
- Пропозиції для повторних покупців.
 - Знижки, програми лояльності та бонуси.

Успішний інтернет-магазин здатний надавати зручний та приємний досвід користувачам, а також ефективно взаємодіяти з покупцями для забезпечення їхньої лояльності та задоволення.

1.2 Аналіз аналогів програмного продукту

При аналізі вже існуючого ПЗ за тематикою дипломної роботи було виявлено декілька проектів. Всі вони схожі один на одного і мають схожу реалізацію.

Аналогічні програмні додатки за заданою темою:

Магазин – Будинок іграшок;

Посилання - <https://bi.ua/ukr/> [5].

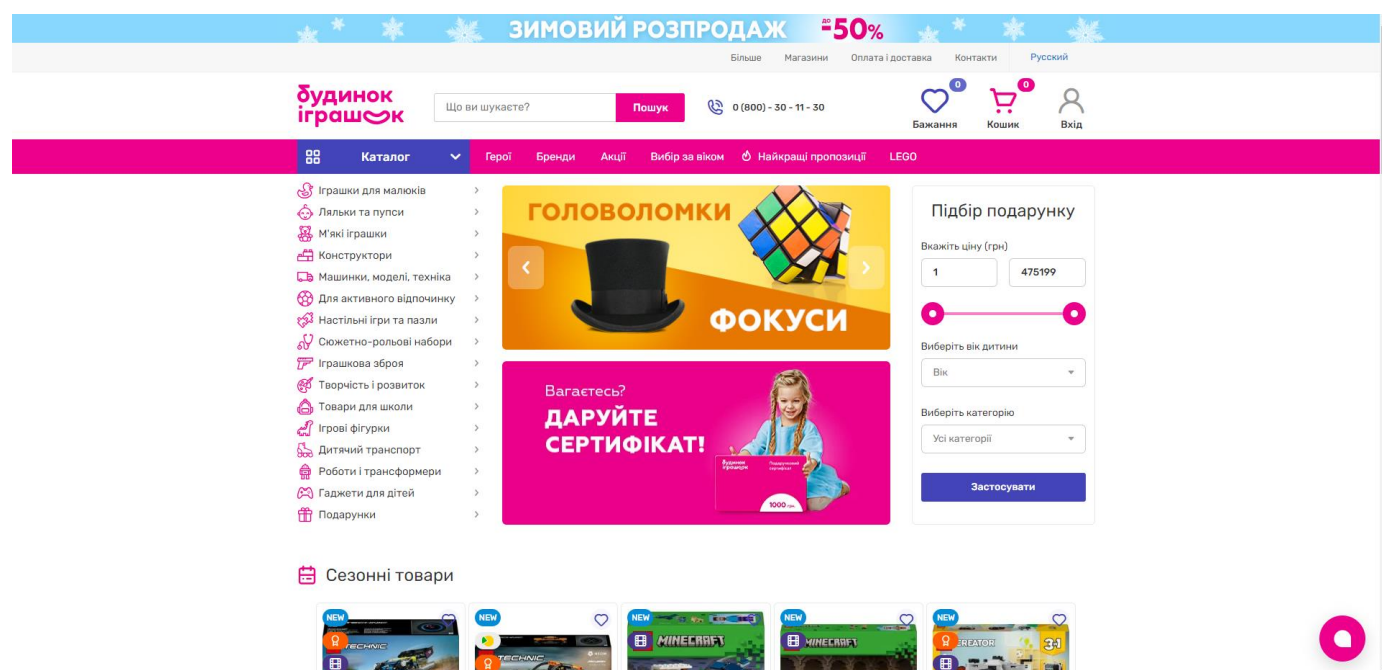


Рисунок 1.1 Будинок іграшок

Інтернет-магазин “Будинок іграшок” є одним із лідерів в сфері продажу дитячих іграшок, адже він має зручний користувацький інтерфейс, великий асортимент товарів, доступні ціни, онлайн підтримку. Також даний магазин надає можливість купувати подарункові сертифікати що допомагає залучати нових покупців. На сайті постійно присутні акційні пропозиції, що покращує показники продажів.

Основні особливості сайту:

- Зручний інтерфейс, зрозуміла навігація
- Зручний інструмент фільтрації та сортування товарів
- Онлайн підтримка користувачів
- Мобільний застосунок для ОС Android та iOS
- Статті та відео-огляди товарів

- Швидкий пошук товарів

Розглянемо наступний приклад:

Магазин – babyplus.ua;

Посилання - <https://babyplus.ua/ua> [6].

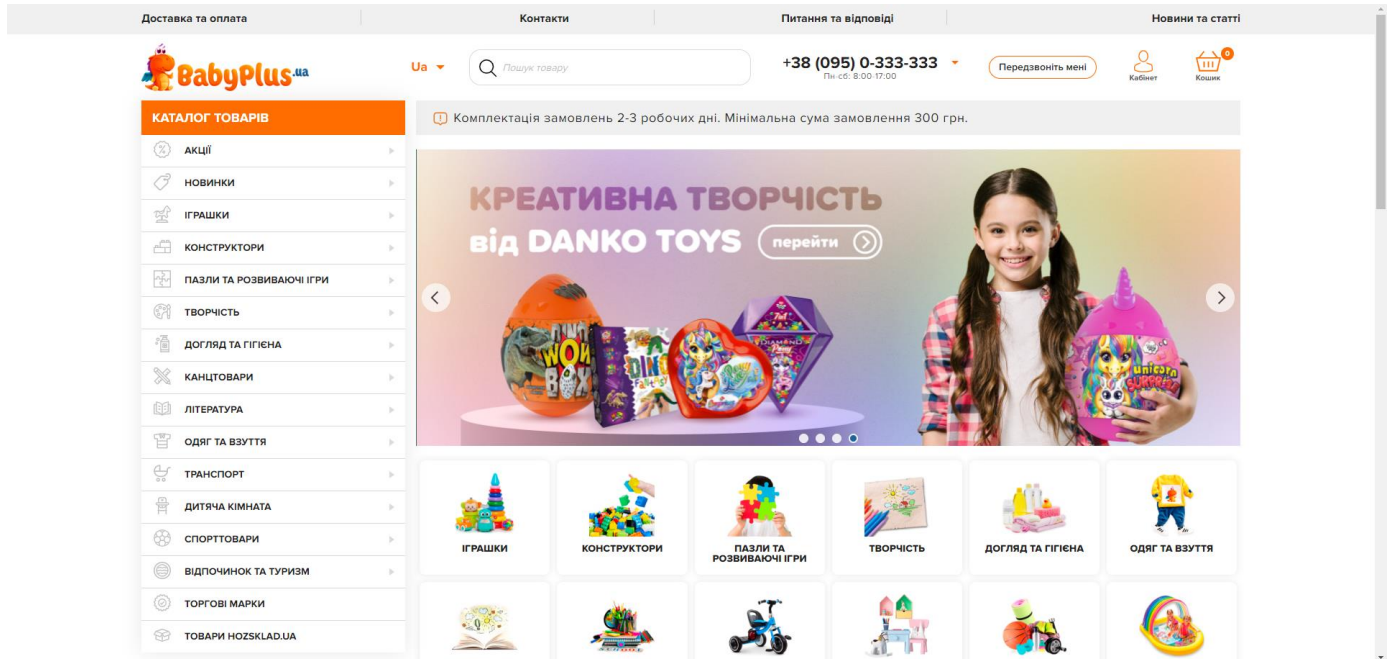


Рисунок 1.2 babyplus.ua

Інтернет-магазин “babyplus.ua” є досить популярним в сфері продажу дитячих іграшок, він має великий асортимент товарів та доступні ціни. Користувацький інтерфейс досить зручний, але його можна було б покращити. Навігація не така зручна, як на сайті “Будинок іграшок” .

Основні особливості сайту:

- Зручний інтерфейс, зрозуміла навігація
- Зручний інструмент фільтрації та сортування товарів
- Статті та відео-огляди товарів
- Наявність та можливість завантаження та перегляду сертифікатів якості продукції
- Швидкий пошук товарів

Переглянувши дані інтернет-магазини можна виділити для себе основні елементи сайту, які варто додати до свого продукту, такі як:

- Перегляд товарів по категоріям
- Можливість зберігати товари які сподобались
- Чіткі та якісні фото товарів
- Детальний опис товарів
- Зрозуміла проста навігація

1.3 Вибір архітектури інтернет-магазину

Для того щоб розроблювана система могла добре функціонувати довгий час, вона має відповідати поставленим вимогам, мати високу якість та бути доступною і легкою у використанні для звичайних користувачів. На всі ці параметри впливає архітектура програмного забезпечення.

Архітектура програмного забезпечення - це високорівневий дизайн програмного продукту, який визначає його структуру, компоненти, взаємодію між ними, а також правила і принципи, які визначають його організацію [7].

Основною метою архітектури програмного забезпечення є створення раціональної та ефективної організації коду, яка легко змінюється, розширюється та підтримується протягом тривалого періоду.

Основні аспекти архітектури програмного забезпечення включають:

1. Компоненти та модулі: Розділення системи на окремі компоненти або модулі для полегшення розробки, тестування та підтримки.
2. Структура: Організація та взаємозв'язок між компонентами для забезпечення логічної та фізичної структури системи.
3. Патерни проектування: Використання визнаних шаблонів проектування для вирішення типових проблем та забезпечення кращих практик [8].
4. Архітектурні стилі: Визначення загальної архітектурної концепції, такої як клієнт-сервер, шарова архітектура, архітектура «мікросервісів» тощо [9].
5. Правила взаємодії: Визначення протоколів та правил взаємодії між компонентами для забезпечення ефективності та надійності системи.
6. Надійність та безпека: Забезпечення стійкості та захищеності системи від непередбачуваних ситуацій та зловмисних атак.
7. Масштабованість: Здатність системи працювати ефективно при збільшенні обсягу даних чи навантаження.

Архітектура програмного забезпечення є ключовим етапом в розробці програмного продукту, оскільки вона визначає його загальну структуру та впливає на його подальшу розробку, тестування, впровадження та підтримку. Давайте розглянемо декілька архітектурних патернів, а потім оберемо найкращий для нашої задачі.

Ось стислий порівняльний опис деяких архітектурних патернів:

1. Монолітна архітектура [10]:

- Опис: Усі компоненти розташовані в одній програмі.
- Переваги: Простота, швидке розгортання.
- Недоліки: Складно розширювати та масштабувати.

2. Шарова архітектура:

- Опис: Розподілення функціоналу на різні шари (представлення, бізнес-логіка, доступ до даних).
- Переваги: Чітке розділення відповідальності, легше тестування.
- Недоліки: Складно розширювати при великій складності.

3. Сервіс-орієнтована архітектура (SOA):

- Опис: Розподілення функціоналу на сервіси, що взаємодіють через мережу.
- Переваги: Легше масштабувати та розширювати.
- Недоліки: Витрати на управління сервісами.

4. Мікросервісна архітектура:

- Опис: Розподілення системи на невеликі автономні сервіси.
- Переваги: Легше масштабувати, розширювати, можливість використання різних технологій для кожного сервісу.
- Недоліки: Управління багатьма сервісами.

5. JAMstack (JavaScript, APIs, Markup):

- Опис: Статичні файли для контенту, динаміка через JavaScript та API.

- Переваги: Висока продуктивність, безпека, простота розгортання та масштабування.
- Недоліки: Менше придатний для складних електронних комерційних рішень.

Для обраного проекту, на мою думку, найкращим варіантом буде монолітна архітектура. Розглянемо детальніше обрану архітектуру:

Монолітна архітектура є традиційним та простим способом організації програмного забезпечення, де весь функціонал розташований в одній програмі, в звичайному випадку — в одному великому веб-застосунку. Основною ідеєю монолітної архітектури є те, що всі частини програми (модулі, компоненти, функціонал) об'єднані в одному цілому.

Основні характеристики монолітної архітектури:

1. Єдина програма: Усі компоненти та функціонал знаходяться в одній кодовій базі та одному процесі.
2. Простота розробки: Розробка, тестування та розгортання стають простішими, оскільки усі частини програми взаємодіють напряму між собою.
3. Легкість тестування: Одна кодова база спрощує тестування, оскільки весь функціонал логічно пов'язаний між собою.
4. Однорідність: Однорідний стек технологій та мов програмування в усьому проекті.

1.4 Обґрунтування вибору інструментальних засобів

Кожен веб-сайт має клієнтську (Frontend) та серверну (Backend) частини. Frontend — це та частина веб-сайту або веб-додатка, яку користувачі бачать та з якою вони взаємодіють. Все, що відображається на екрані користувача та взаємодіє з його діями, відноситься до Frontend. Це включає в себе:

- HTML (HyperText Markup Language): Визначає структуру сторінки та розміщення її елементів [11].
- CSS (Cascading Style Sheets): Відповідає за вигляд та стиль сторінки, включаючи кольори, шрифти, розташування елементів та інше [12].
- JavaScript: Додає динамічність та інтерактивність до сторінок. Він використовується для створення анімацій, обробки подій користувача та взаємодії з сервером [13].

Frontend розробники працюють над тим, щоб забезпечити привабливий вигляд та легкість у використанні веб-сторінки для кінцевого користувача.

Backend — це та частина веб-сайту або веб-додатка, яка відповідає за обробку даних та бізнес-логіку. Це та "невидима" частина, яка працює "позаду" та не взаємодіє безпосередньо з користувачем. Серверна частина включає в себе:

- Сервер: Обробляє запити від клієнтів, виконує логіку додатку та відправляє відповіді назад.
- База даних: Зберігає дані, які використовуються в додатку.
- Бізнес-логіка: Визначає, які дії повинні виконуватися з даними та які відповіді надсилати.

Backend розробники працюють над тим, щоб забезпечити ефективну та безпечну обробку даних, взаємодію з базою даних, обслуговування запитів від фронтенду та багато іншого. Frontend і Backend взаємодіють між собою, створюючи повноцінний веб-сайт або веб-додаток.

Для Frontend веб-сайту інтернет-магазину будуть використовуватись HTML, CSS та JavaScript. Обрані технології розробки мають багато переваг. Основні переваги:

1. **Широке поширення та підтримка:** HTML, CSS і JavaScript є основними технологіями для розробки веб-інтерфейсів та мають величезну спільноту розробників. Їхнє використання забезпечує стандартизацію та сумісність з більшістю сучасних браузерів.
2. **Простота вивчення та використання:** HTML, CSS і JavaScript є відносно простими мовами, що полегшує їх вивчення та використання. Нові розробники можуть швидко отримати базові навички та розпочати роботу над проектами.
3. **Ефективність для веб-інтерфейсів:** Кожна мова має своє призначення: HTML визначає структуру сторінок, CSS стилізує їх, а JavaScript надає динамічність та інтерактивність. Ця комбінація дозволяє легко створювати ефективні та дружелюбні інтерфейси для користувачів.
4. **Широкі можливості анімацій і інтерактивності:** JavaScript дозволяє легко створювати анімації, переходи та інтерактивні ефекти, що можуть покращити користувацький досвід та зробити інтернет-магазин більш привабливим.
5. **SEO-оптимізація:** HTML забезпечує структуру сторінок, що полегшує процес індексації веб-пошуковими системами. Додавання метатегів, описів та іншого SEO-відповідного контенту може покращити видимість в пошукових системах.
6. **Легкість інтеграції:** HTML, CSS і JavaScript легко інтегруються з іншими технологіями, бібліотеками і фреймворками. Це дозволяє розробникам використовувати різноманітні інструменти для покращення функціоналу та ефективності веб-сайтів.

Обираючи HTML, CSS і JavaScript для Frontend розробки, розробники отримують доступ до потужних інструментів, які дозволяють швидко створювати ефективні та привабливі веб-сайти, зокрема інтернет-магазини.

PHP є популярною мовою програмування для розробки Backend веб-сайтів інтернет-магазинів. PHP є високорівневою мовою, яка легко інтегрується з HTML та іншими Frontend технологіями, такими як CSS та JavaScript [14]. Основні переваги використання PHP для Backend розробки:

1. **Широка підтримка:** PHP має велику і активну спільноту розробників, що означає, що ви зможете знайти багато ресурсів, документації та сторонніх бібліотек для використання.
2. **Сумісність з багатьма серверами:** PHP може запускатися на багатьох серверах, таких як Apache [15], Nginx [16], або вбудований сервер PHP. Це дозволяє вибирати сервер, який вам більше підходить.
3. **Легкість вивчення:** PHP вважається однією з найлегших мов програмування для вивчення.
4. **Розширені можливості обробки форм та роботи з базами даних:** PHP часто використовується для обробки форм на веб-сайтах та взаємодії з базами даних, що робить його популярним для розробки динамічних веб-сайтів, таких як інтернет-магазини.
5. **Широкі можливості фреймворків:** Існує багато PHP-фреймворків, таких як Laravel, Symfony, CodeIgniter та інші, які допомагають спростити та прискорити процес розробки, надаючи структуру та готові рішення для роботи з різними аспектами веб-розробки.

Враховуючи ці фактори, PHP може бути гарним вибором для розробки Backend частини інтернет-магазину [17].

Система управління базами даних (СУБД) є важливою складовою для веб-сайтів з численними функціональними можливостями, особливо якщо сайт включає в себе збереження та обробку великої кількості структурованої інформації [18]. Для розробки веб-сайту інтернет-магазину іграшок було обрано СУБД MySQL [19].

MySQL є однією з популярних систем управління базами даних (СУБД), і вона широко використовується в сполученні з технологіями HTML, CSS, JavaScript та PHP

для розробки інтернет-магазинів. Ось деякі переваги використання MySQL у контексті інтернет-магазину:

1. **Відмінна підтримка PHP:** MySQL взаємодіє з PHP дуже ефективно, що робить їх популярною комбінацією для розробки веб-додатків.
2. **Широка спільнота і документація:** MySQL має велику та активну спільноту розробників. Це означає, що ви завжди можете знайти відповіді на свої питання або розв'язати проблеми через широкий обмін знаннями.
3. **Повна підтримка транзакцій:** MySQL підтримує транзакції, що робить його ідеальним вибором для ситуацій, де важлива консистентність даних, наприклад, при обробці платежів в інтернет-магазині.
4. **Безпека та управління доступом:** MySQL має вбудовані механізми безпеки та управління доступом, що дозволяє налаштовувати рівні доступу користувачів до бази даних. Це важливо для забезпечення безпеки інформації користувачів та транзакцій в інтернет-магазині.
5. **Гнучкість та розширюваність:** MySQL дозволяє ефективно обробляти великі обсяги даних та легко масштабується. Це особливо важливо для інтернет-магазинів, які можуть зростати та змінюватися з часом.
6. **Оптимізація продуктивності:** MySQL пропонує інструменти для оптимізації продуктивності, такі як індексація, кешування та оптимізація запитів, що дозволяє прискорити роботу інтернет-магазину та зменшити завантаження бази даних.

Узагальнюючи, MySQL є надійним та ефективним рішенням для Backend розробки інтернет-магазину.

Висновки до розділу 1

В першому підрозділі кваліфікаційної роботи бакалавра було розглянуто особливості веб-сайтів, визначено мету та завдання на дипломну роботу.

В другому підрозділі кваліфікаційної роботи бакалавра було розглянуто та проаналізовано існуючі веб-сайти інтернет-магазинів за тематикою дипломної роботи. А саме веб-сайти інтернет-магазинів “Будинок іграшок” та “babyplus.ua”. Після чого було визначено особливості та функції власного продукту.

В третьому підрозділі кваліфікаційної роботи бакалавра було розглянуто популярні архітектури програмного забезпечення та обрано архітектуру власного майбутнього веб-сайту.

В четвертому підрозділі було розглянуто що таке Frontend та Backend, та обрано засоби розробки програмного продукту.

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-САЙТУ ІНТЕРНЕТ-МАГАЗИНУ ДИТЯЧИХ ІГРАШОК

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

У цьому розділі кваліфікаційної роботи бакалавра представлено аналіз вимог до програмного продукту з акцентом на виявленні функціональних та нефункціональних вимог. Крім того, визначено вимоги до часових характеристик реакції додатку на дії користувача, а також окреслено різні варіанти використання. Для забезпечення чіткого розуміння структури додатково включено узагальнену діаграму елементів (сутностей системи), їх структури та методів з описом.

Аналіз вимог передбачає докладне вивчення потреб та очікувань користувачів щодо функціональності та характеристик веб-сайту інтернет-магазину. Основною метою аналізу вимог є зрозуміти, що потрібно втілити в програмному забезпеченні для задоволення потреб користувачів. За допомогою опитувань та зворотного зв'язку були визначені функціональні вимоги, які окреслюють специфічні особливості та можливості, якими повинен володіти програмний продукт. На додаток були зібрані нефункціональні вимоги, які стосуються таких аспектів, як продуктивність, стабільність роботи і зручність використання.

Функціональні вимоги:

- **Реєстрація та автентифікація користувачів:** Система повинна надавати можливість реєстрації нових користувачів, зберігання їх даних у безпечному вигляді та автентифікацію під час входу в систему.
- **Перегляд товарів:** Користувачі повинні мати доступ до каталогу товарів, а також до детального перегляду кожного товару, що включає ціну, наявність, категорію, вікову категорію, розмір, опис та фотографію.
- **Фільтри та пошук:** Система повинна надавати можливість фільтрації товарів за віковою категорією, ціною та наявністю. Пошук товарів повинен бути швидким та ефективним.

- **Список бажаного:** Користувачі повинні мати можливість додавати товари до списку бажаного, переглядати вміст списку і видаляти товари зі списку.
- **Кошик:** Користувачі повинні мати можливість додавати товари до кошику, переглядати вміст кошику, змінювати кількість кожного товару в кошику, переглядати підсумкову ціну усіх товарів, видаляти товари з кошику.
- **Оформлення замовлення:** Система повинна забезпечувати можливість оформлення замовлення, вибору способу доставки та оплати.
- **Новини:** Користувачі повинні мати можливість переглядати список усіх новин на сайті, а також переглядати повний текст новини.
- **Особистий кабінет:** Користувачі повинні мати можливість переглядати та оновлювати особисті дані в особистому кабінеті, а також переглядати список замовлень.
- **Управління замовленнями:** Користувачі повинні мати можливість переглядати історію своїх замовлень та їх статуси.
- **Додавання, редагування, приховування товарів адміністратором:** Адміністратор повинен мати можливість додавати, редагувати та приховувати товари на веб-сайті інтернет-магазину.
- **Додавання, редагування, видалення новин адміністратором:** Адміністратор повинен мати можливість додавати, редагувати та видаляти новини на веб-сайті інтернет-магазину.
- **Додавання, редагування, видалення категорій товарів:** Адміністратор повинен мати можливість додавати, редагувати та видаляти категорії товарів на веб-сайті інтернет-магазину.

Нефункціональні вимоги:

- **Зручний інтерфейс:** Інтерфейс системи повинен бути інтуїтивно зрозумілим та зручним для використання.
- **Продуктивність:** Система повинна забезпечувати швидку реакцію на запити користувачів.

- **Безпека:** Всі дані користувачів повинні бути захищені.
- **Надійність:** Система повинна бути стабільною і забезпечувати безперебійне функціонування навіть під час високого навантаження.

Вимоги до часових характеристик реакції системи на дії користувача:

- **Час завантаження сторінки:** Менше 3 секунд.
- **Час обробки транзакцій** (наприклад, додавання товару до кошика): Менше 2 секунд.

Варіанти використання системи:

1. Реєстрація та вхід в систему:

- Користувач переходить на сторінку реєстрації, вводить особисті дані та створює обліковий запис.
- Користувач входить у систему, використовуючи свої облікові дані.

2. Перегляд каталогу товарів:

- Користувач переглядає каталог товарів, вибирає вікову категорію, застосовує фільтри.
- Користувач переглядає детальну інформацію про конкретний продукт.

3. Перегляд категорій товарів:

- Користувач переглядає категорії товарів та переходить до конкретної категорії.
- Користувач переглядає детальну інформацію про конкретний продукт.

4. Додавання товарів до списку бажаного:

- Користувач додає обрані товари до списку бажаного або видаляє.

5. Додавання товарів до кошику:

- Користувач додає обрані товари до кошику, змінює їх кількість або видаляє.

6. Перегляд інформації про доставку та оплату:

- Користувач переходить на сторінку Доставка та оплата та ознайомлюється з інформацією.

7. Оформлення замовлення:

- Користувач переходить до оформлення замовлення, вибирає спосіб доставки та оплати, підтверджує замовлення.

8. Перегляд новин:

- Користувач переглядає список останніх новин.
- Користувач переглядає повний текст конкретної новини.

9. Перегляд особистого кабінету:

- Користувач переглядає особисті дані в особистому кабінеті та оновлює їх при потребі.
- Користувач переглядає список своїх замовлень.

10. Керування товарами адміністратором:

- Адміністратор додає новий товар, заповнює всі необхідні поля та обирає фото.
- Адміністратор приховує товар.

11. Керування категоріями адміністратором:

- Адміністратор додає нову категорію.
- Адміністратор видаляє категорію.

12. Керування новинами адміністратором:

- Адміністратор додає нову новину.
- Адміністратор видаляє новину.



Рисунок 2.1 – Діаграма варіантів використання

Діаграма варіантів використання, що зображена на рисунку 2.1 допомагає зрозуміти загальну структуру та функціональні можливості системи, а також полегшує процес розробки та тестування, забезпечуючи чітке уявлення про те, як система повинна функціонувати з точки зору кінцевих користувачів і адміністраторів [20].

Об'єктно-орієнтована структура. Об'єктно-орієнтована структура системи інтернет-магазину дитячих іграшок включає основні класи, їх властивості та методи, а також взаємозв'язки між цими класами. Дана структура допомагає організувати програмний код таким чином, щоб він був зрозумілим, підтримуваним та легко

розширюваним. Діаграма класів – ключовий інструмент для моделювання структури об’єктно-орієнтованих систем. Вона дає змогу представити класи, їхні атрибути, методи та взаємозв’язки між ними [21].

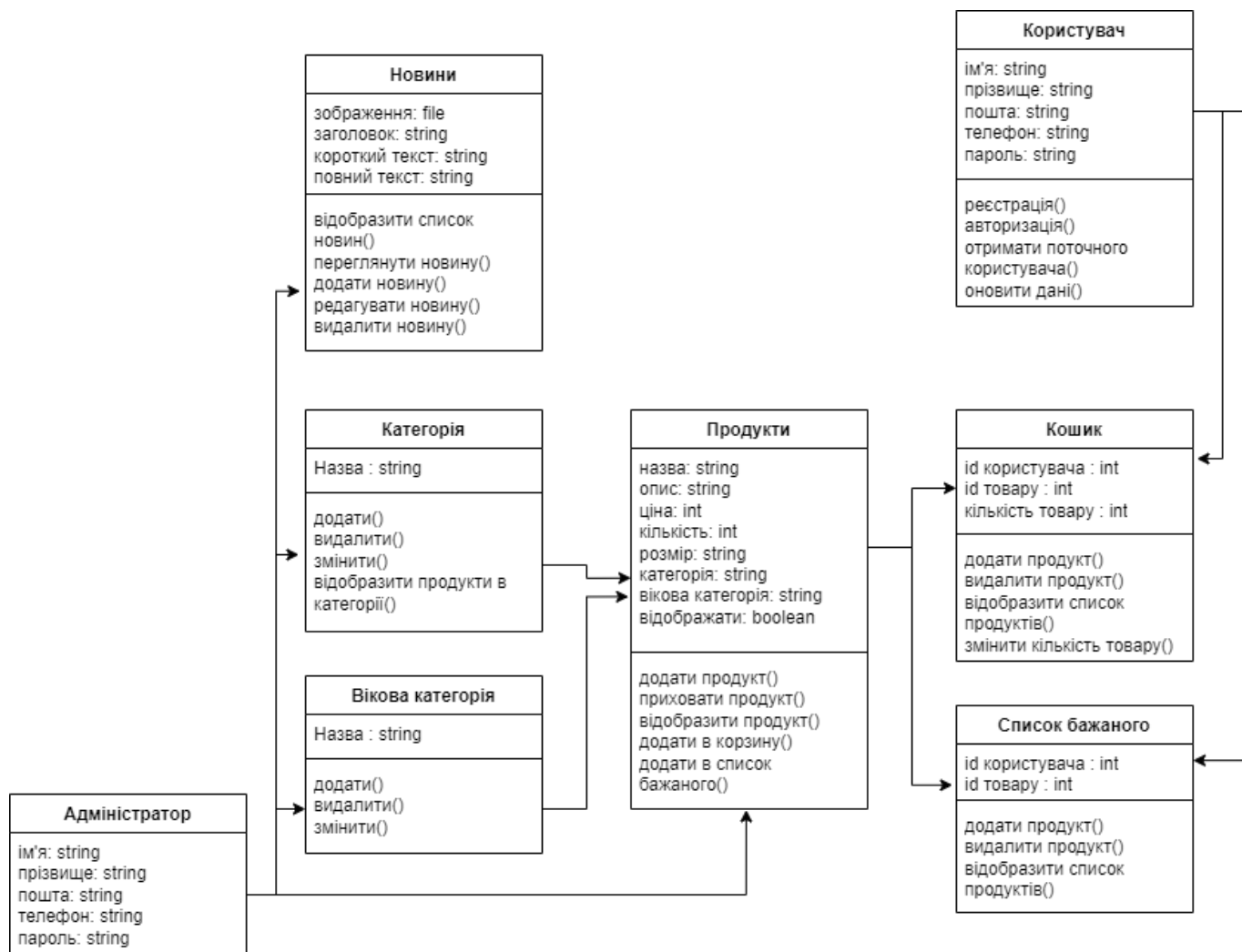


Рисунок 2.2 – Діаграма класів

Інтернет-магазин дитячих іграшок спроектовано за шаблоном MVC (Model-View-Controller) [22]:

- Model (Модель): Відповідає за управління даними. Взаємодіє з базою даних і забезпечує доступ до даних.
- View (Представлення): Відповідає за відображення даних користувачеві. Формує користувацький інтерфейс і представляє дані, отримані від моделі.

- **Controller (Контролер):** Відповідає за обробку запитів від користувача. Взаємодіє з моделлю для отримання даних і передає їх у представлення для відображення.

Архітектурний патерн MVC використовується в основі багатьох фреймворків для створення веб-застосунків. Цей шаблон допомагає розділити обов'язки між компонентами, що робить додаток більш структурованим, підтримуваним і масштабованим [23].

Розглянемо кожен клас детальніше:

Agecategory (Вікова категорія) – призначений для керування віковими категоріями. Методи: **AddAgecategory** – метод для додавання нової вікової категорії. **UpdateAgecategory** – метод для оновлення існуючої вікової категорії. **DeleteAgecategory** – метод для видалення існуючої вікової категорії. **GetAgecategories** – повертає всі існуючі в базі даних вікові категорії. **GetAgecategoryById** – повертає вікову категорію за заданим ідентифікатором.

Basket (Кошик) – призначений для керування кошиком. Методи: **AddToBasket** – метод для додавання товару в кошик. **RemoveFromBasket** – метод для видалення товару з кошику. **GetProductsInBasket** – метод повертає всі продукти які користувач додав у кошик. **UpdateQuantityInBasket** – метод, що оновлює кількість кожного товару в кошику.

Category (Категорія) – призначений для керування категоріями товарів. Методи: **AddCategory** – метод для додавання нової категорії на сторінку каталогу товарів. **UpdateCategory** – метод для зміни існуючих категорій. **DeleteCategory** – метод для видалення категорій. **GetCategories** – метод, що повертає всі існуючі в базі даних категорії. **GetCategoryById** – метод, що повертає категорію за заданим ідентифікатором.

Favorite (Список бажаного) – призначений для керування списком бажаного. Методи: **AddToFavorite** – метод для додавання товару в список бажаного. **RemoveFromFavorite** – метод для видалення товару зі списку бажаного.

GetProductsInFavorite – метод для отримання товарів у списку бажаного.
IsInFavorite – метод для перевірки чи продукт додано в список бажаного.

News (Новини) – призначений для керування новинами. Методи: AddNews – метод для додавання новин. GetLastNews – метод, що повертає задану кількість останніх новин. GetNewsById – метод, що повертає новину за заданим ідентифікатором. ChangePhoto – метод для заміни фото новини. UpdateNews – метод для редагування існуючої новини. DeleteNews – метод для видалення існуючої новини.

Products (Продукти) – призначений для керування продуктами. Методи: AddProduct – метод для додавання нового товару. GetAllProducts – повертає усі товари з бази даних. GetProductsByCategory – повертає усі товари заданої категорії. GetProductById – повертає товар за заданим ідентифікатором. UpdateProduct – метод для редагування інформації про існуючий товар. HideProduct – метод для позначення товару відсутнім в наявності. DisplayProduct – метод для позначення товару присутнім в наявності.

Users (Користувач) – призначений для керування користувачами. Методи: AddUser – метод для реєстрації нового користувача. AuthUser – метод для авторизації зареєстрованого користувача. GetCurrentUser – метод для отримання поточного користувача. UpdateUser – метод для оновлення даних користувача.

Таким чином, дана система реалізує функціонал веб-сайту інтернет-магазину [24].

2.2 Розробка бази даних системи

На даному етапі розробимо структуру реалізації Web-орієнтованої системи для інтернет-магазину [25]. Функціонуючу структуру інтернет-магазину представимо наступним чином:

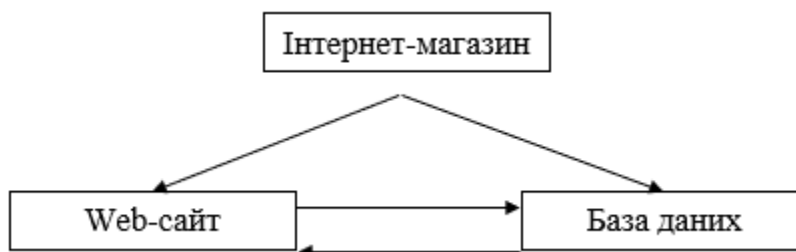


Рисунок 2.3 – Функціонуюча структура інтернет-магазину

База даних інтернет-магазину складається з семи таблиць, розглянемо їх структуру [26]:

Таблиця products зберігає інформацію про товари.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐ 1	id	int(11)			Ні	Немає		AUTO_INCREMENT	
☐ 2	name	varchar(40)	utf8mb4_unicode_ci		Ні	Немає			
☐ 3	description	varchar(500)	utf8mb4_unicode_ci		Ні	Немає			
☐ 4	size	varchar(10)	utf8mb4_unicode_ci		Ні	Немає			
☐ 5	category_id	int(11)			Так	1			
☐ 6	agecategory_id	int(11)			Так	1			
☐ 7	price	int(11)			Ні	Немає			
☐ 8	amount	int(11)			Ні	0			
☐ 9	visible	int(11)			Ні	0			
☐ 10	photo	varchar(255)	utf8mb4_unicode_ci		Так	NULL			

Рисунок 2.4 – Структура таблиці products

Таблиця category зберігає інформацію про категорії товарів.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐ 1	id	int(11)			Ні	Немає		AUTO_INCREMENT	
☐ 2	name	varchar(255)	utf8mb4_unicode_ci		Ні	Немає			

Рисунок 2.5 – Структура таблиці category

Таблиця agecategory зберігає інформацію про вікові категорії товарів.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1 id	int(11)			Hi	Немає		AUTO_INCREMENT	
☐	2 age	varchar(40)	utf8mb4_unicode_ci		Hi	Немає			

Рисунок 2.6 – Структура таблиці agecategory

Таблиця users зберігає інформацію про зареєстрованих користувачів.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1 id	int(11)			Hi	Немає		AUTO_INCREMENT	
☐	2 login	varchar(50)	utf8mb4_unicode_ci		Hi	Немає			
☐	3 password	varchar(255)	utf8mb4_unicode_ci		Hi	Немає			
☐	4 access	tinyint(1)			Hi	0			
☐	5 lastname	varchar(255)	utf8mb4_unicode_ci		Hi	Немає			
☐	6 firstname	varchar(255)	utf8mb4_unicode_ci		Hi	Немає			
☐	7 telnumber	varchar(10)	utf8mb4_unicode_ci		Так	NULL			

Рисунок 2.7 – Структура таблиці users

Таблиця user_basket зберігає товари в кошику зареєстрованих користувачів.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1 id	int(11)			Hi	Немає		AUTO_INCREMENT	
☐	2 user_id	int(11)			Hi	Немає			
☐	3 product_id	int(11)			Hi	Немає			
☐	4 count	int(11)			Hi	1			

Рисунок 2.8 – Структура таблиці user_basket

Таблиця user_favorite зберігає товари в списку бажаного зареєстрованих користувачів.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1 id	int(11)			Hi	Немає		AUTO_INCREMENT	
☐	2 user_id	int(11)			Hi	Немає			
☐	3 product_id	int(11)			Hi	Немає			

Рисунок 2.9 – Структура таблиці user_favorite

Таблиця news зберігає усі додані новини.

#	Ім'я	Тип	Зіставлення	Атрибути	Нуль	За замовчуванням	Коментарі	Додатково	Дія
☐	1 id 🔑	int(11)			Hi	Немає		AUTO_INCREMENT	
☐	2 title	varchar(255)	utf8mb4_unicode_ci		Hi	Немає			
☐	3 short_text	text	utf8mb4_unicode_ci		Hi	Немає			
☐	4 text	text	utf8mb4_unicode_ci		Hi	Немає			
☐	5 datetime	datetime			Hi	Немає			
☐	6 datetime_lastedit	datetime			Так	NULL			
☐	7 photo	varchar(255)	utf8mb4_unicode_ci		Так	NULL			
☐	8 user_id 🔑	int(11)			Hi	Немає			

Рисунок 2.10 – Структура таблиці news

Розглянемо структуру створеної бази даних:

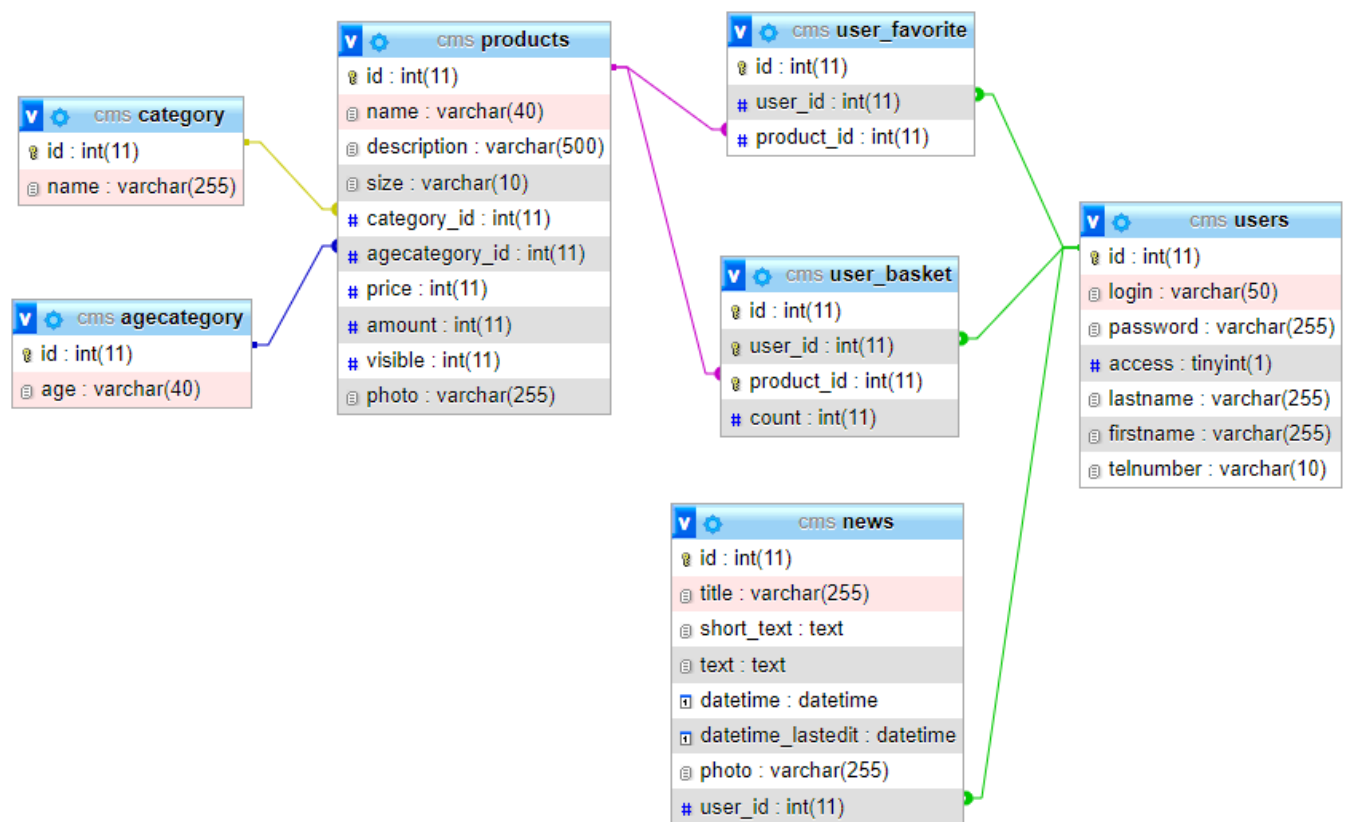


Рисунок 2.11 – Структура бази даних

Зображена структура забезпечує ефективну організацію даних та дозволяє здійснювати необхідні операції з даними в рамках інтернет-магазину.

2.3 Проектування та реалізація алгоритмів роботи системи

Розробка та реалізація системних алгоритмів відповідають за виконання різних обчислювальних завдань, обробку даних і прийняття рішень в системі. Було створено декілька діаграм активності для візуалізації та представлення потоку діяльності в системі. Вони пропонують чіткий спосіб зображення послідовних і паралельних дій, які відбуваються під час виконання системних процесів.

Діаграма активності для реєстрації / авторизації: ця діаграма показує послідовність дій користувача, таких як реєстрація чи авторизація на сайті.

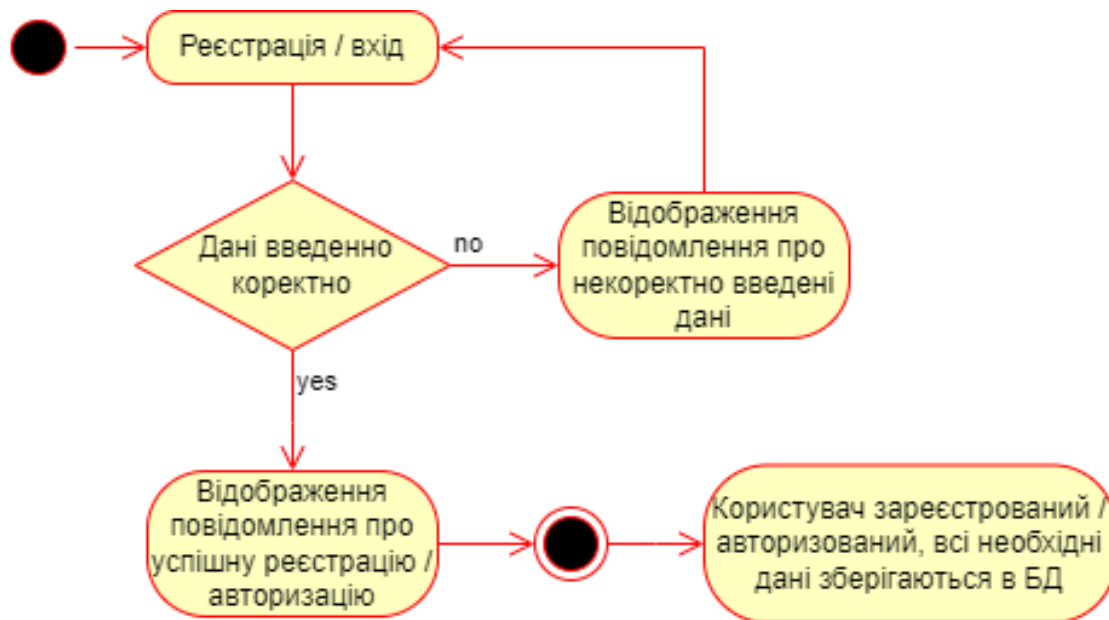


Рисунок 2.12 – Діаграма активності для реєстрації / авторизації

На основі діаграми активності для реєстрації / авторизації було створено наступну таблицю специфікації:

Таблиця 2.1

Назва варіанту використання: реєстрація / авторизація		Рівень: високий
Основний актор: користувач		
Короткий опис:	Реєстрація або авторизація на сайті	
Передумови:	Відкрити сторінку реєстрації / авторизації на сайті	
Включення:	Ініціалізація	

Основний потік:	Користувач може зареєструватись / авторизуватись на сайті, заповнивши всі необхідні поля форми реєстрації / авторизації
Альтернативні / виключні потоки	Повідомлення про помилку

Діаграма активності для перегляду товарів: ця діаграма показує послідовність дій користувача при перегляді товарів на сайті.



Рисунок 2.13 – Діаграма активності для перегляду товарів

На основі діаграми активності перегляду товарів було створено наступну таблицю специфікації:

Назва варіанту використання: перегляд товарів		Рівень: високий
Основний актор: користувач		
Короткий опис:	Перегляд товарів на сайті	
Передумови:	Відкрити сторінку з товарами	
Включення:	Ініціалізація	
Основний потік:	Користувач може переглянути список товарів, переглянути детальну інформацію про товар, додати товар в список бажаного та в кошик	
Альтернативні / виключні потоки	При додаванні товару в кошик, якщо товар вже було додано то кількість товару збільшується. При додаванні товару в список бажаного, якщо товар вже було додано то товар видаляється зі списку.	

Діаграма активності для перегляду списку бажаного: ця діаграма показує послідовність дій користувача при перегляді списку бажаного на сайті.

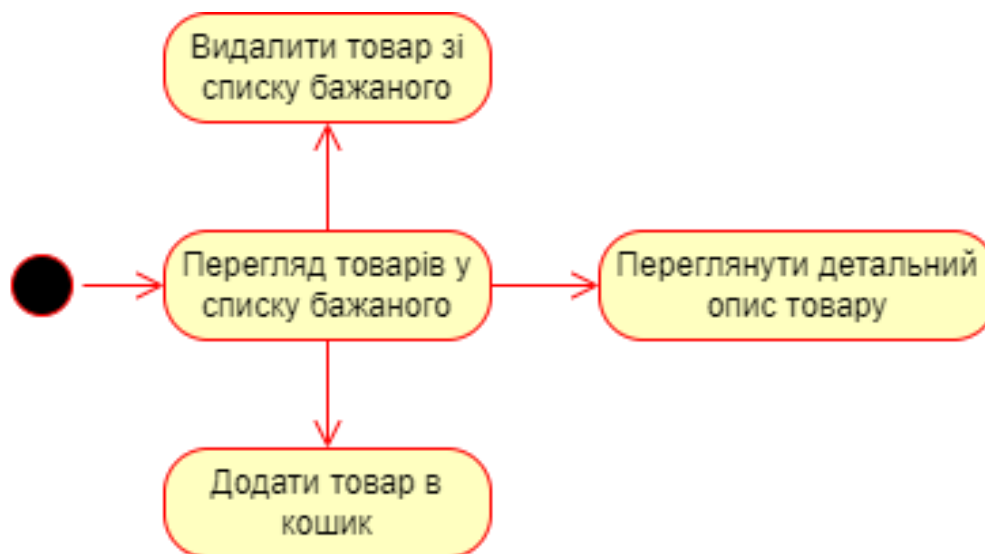


Рисунок 2.14 – Діаграма активності для перегляду списку бажаного

На основі діаграми активності для перегляду списку бажаного було створено наступну таблицю специфікації:

					ІПЗ.КР.Б. – 121 – 24 – ПЗ	36

Назва варіанту використання: перегляд списку бажаного		Рівень: високий
Основний актор: користувач		
Короткий опис:	Перегляд списку бажаних товарів	
Передумови:	Відкрити список бажаного	
Включення:	Ініціалізація	
Основний потік:	Користувач може переглянути продукти в списку бажаного, видалити продукти зі списку бажаного, додати продукти зі списку бажаного в кошик, переглянути детальну інформацію про товар	
Альтернативні / виключні потоки	Повідомлення про помилку	

Діаграма активності для перегляду кошику: ця діаграма показує послідовність дій користувача при перегляді списку бажаного на сайті.



Рисунок 2.15 – Діаграма активності для перегляду кошику

На основі діаграми активності для перегляду кошику було створено наступну таблицю специфікації:

Таблиця 2.4

Назва варіанту використання: перегляд кошику		Рівень: високий
Основний актор: користувач		
Короткий опис:	Перегляд товарів у кошику	
Передумови:	Відкрити кошик	
Включення:	Ініціалізація	
Основний потік:	Користувач може переглянути продукти в кошику, видалити продукти з кошику, переглянути детальну інформацію про товар, змінити кількість кожного товару, оформити замовлення	
Альтернативні / виключні потоки	Повідомлення про помилку	

Ці діаграми активності та таблиці специфікації допоможуть уявити та візуалізувати послідовність дій та взаємодію користувача з веб-сайтом інтернет-магазину дитячих іграшок.

2.4 Реалізація веб-сайту інтернет-магазину дитячих іграшок

Для розробки веб-сайту було використано середовище розробки JetBrains PhpStorm. Спочатку було створено дизайн макету сайту за допомогою HTML5, CSS3 та вставок PHP. Потім було створено усі необхідні компоненти для керування контентом та обробки подій [27].

Розглянемо основні методи, що використовуються в проєкті:

Метод run() головного класу ядра системи Core виконує основний процес роботи системи, він відповідає за обробку HTTP-запиту і виклик відповідного контролера та його методу на основі URL-адреси:

```
public function run()
{
    $path = $_GET['path'];
    $pathParts = explode('/', $path);
    $className = ucfirst($pathParts[0]);
    if (empty($className))
        $fullClassName = 'controllers\\Products';
    else
        $fullClassName = 'controllers\\'.$className;
    $methodName = ucfirst($pathParts[1]);
    if (empty($methodName))
        $fullMethodName = 'actionIndex';
    else
        $fullMethodName = 'action'.$methodName;
    if (class_exists($fullClassName)) {
        $controller = new $fullClassName();
        if (method_exists($controller, $fullMethodName)){
            $method = new \ReflectionMethod($fullClassName, $fullMethodName);
            $paramsArray = [];
            foreach ($method->getParameters() as $parameter)
            {
                array_push($paramsArray, isset($_GET[$parameter->name]) ? $_GET[$parameter->name] :
null);
            }
            $result = $method->invokeArgs($controller, $paramsArray);
            if (is_array($result)){
                self::$mainTemplate->setParams($result);
            }
        }
        else
            throw new \Exception('404 Not Found');
    }
    else
        throw new \Exception('404 Not Found');
```

```
}
```

Метод `actionAdd()` контролера `Products`, який обробляє додавання товару на веб-сайт. Він перевіряє доступ користувача, обробляє дані форми, завантажує зображення товару та зберігає їх, а також відображає відповідні повідомлення про успіх або помилки. Якщо користувач не має необхідного доступу, відображається сторінка з повідомленням про заборону доступу.

```
public function actionAdd()
{
    $titleForbidden = 'Доступ заборонено!';
    $title = 'Додавання товару';
    if (empty($this->user) || $this->userModel->GetCurrentUser()['access'] != 1)
        return $this->render('forbidden', null, [
            'mainTitle' => $titleForbidden,
            'pageTitle' => $titleForbidden,
            'pageClass' => 'other'
        ]);
    if ($this->isPost())
    {
        $result = $this->productModel->AddProduct($_POST);
        if ($result['error'] === false){
            $allowed_types = ['image/png', 'image/jpeg'];
            if(is_file($_FILES['file']['tmp_name']) && in_array($_FILES['file']['type'], $allowed_types)) {
                switch ($_FILES['file']['type']){
                    case 'image/png' :
                        $extension = 'png';
                        break;
                    default:
                        $extension = 'jpg';
                }
                $name = $result['id'].'_'.uniqid().'.'.$extension;
                move_uploaded_file($_FILES['file']['tmp_name'], 'files/products/'.$name);
                $this->productModel->ChangePhoto($result['id'], $name);
            }
            return $this->renderMessage('ok', 'Товар успішно додано!', null, [
                'mainTitle' => $title,
                'pageTitle' => $title,
                'pageClass' => 'other'
            ]);
        }
        else{
            $message = implode('<br/>', $result['messages']);
            return $this->render('form', ['model' => $_POST], [
                'mainTitle' => $title,
                'pageTitle' => $title,
                'MessageText' => $message ,
            ]
        )
    }
}
```

```

        'MessageClass' => 'danger',
        'pageClass' => 'other'
    });

    }
    }else
    return $this->render('form', ['model' => $_POST], [
        'mainTitle' => $title,
        'pageTitle' => $title,
        'pageClass' => 'other'
    ]);
    }
}

```

Цей скрипт оновлює загальну ціну товарів у кошику на основі кількості та ціни кожного товару. Оновлює загальну ціну товарів при зміні кількості. Перевіряє, чи кількість товару не перевищує доступний максимум. Надсилає запит на сервер для оновлення кількості товару в кошику. Обробляє відповідь від сервера і повідомляє про успіх чи помилку:

```

<script>
$(document).ready(function() {
    function updateTotalPrice() {
        let totalPrice = 0;
        $('.count-input').each(function() {
            let count = parseInt($(this).val());
            let price = parseFloat($(this).data('price'));
            totalPrice += count * price;
        });
        $('#total-price').text(totalPrice.toFixed(2));
    }
    $('.count-input').off('input');
    $('.count-input').on('input', function() {
        let max = $(this).data('max');
        let value = parseInt($(this).val(), 10);
        let productId = $(this).data('id');
        if (value > max) {
            $(this).val(max);
            value = max;
        }
        updateTotalPrice();
        let requestId = Math.random().toString(36).substr(2, 9);
        console.log(`Sending request ${requestId} to update quantity`, { id: productId, quantity: value });
        $.post('/basket/updateQuantity', { id: productId, quantity: value }, function(response) {
            console.log(`Response from server for request ${requestId}:`, response);
            let data;
            try {
                data = JSON.parse(response);
            }
        });
    });
});

```

```

    } catch (e) {
        console.error(`Error parsing JSON response for request ${requestId}:`, e);
        alert('Помилка оновлення кількості продукту. JSON Error');
        return;
    }
    if (data.status === 'ok') {
        console.log(`Request ${requestId} succeeded`);
    } else {
        console.error(`Request ${requestId} failed with status:`, data.status);
        alert('Помилка оновлення кількості продукту. Server Error');
    }
    }).fail(function(jqXHR, textStatus, errorThrown) {
        console.error(`Request ${requestId} failed:`, textStatus, errorThrown);
        alert('Помилка при надсиланні запиту на сервер.');
    });
});
updateTotalPrice();
});
</script>

```

Метод AddToBasket моделі Basket додає товар до кошика: Перевіряє, чи активна сесія, і запускає її, якщо ні. Якщо користувач авторизований, оновлює кількість товару в базі даних для цього користувача або додає новий запис, якщо товар ще не в кошику. Якщо користувач не авторизований, оновлює кількість товару в сесії або додає новий товар до сесійного кошика:

```

public function AddToBasket($product_id, $count = 1){
    if (session_status() == PHP_SESSION_NONE) {
        session_start();
    }
    $user = (new \models\Users()->GetCurrentUser());
    if ($user) {
        $user_id = $user['id'];
        $existingItem = \core\Core::getInstance()->getDB()->select('user_basket', 'count', ['user_id' =>
        $user_id, 'product_id' => $product_id]);
        if ($existingItem) {
            $newCount = $existingItem[0]['count'] + $count;
            \core\Core::getInstance()->getDB()->update('user_basket', ['count' => $newCount], ['user_id' =>
        $user_id, 'product_id' => $product_id]);
        } else {
            \core\Core::getInstance()->getDB()->insert('user_basket', ['user_id' => $user_id, 'product_id' =>
        $product_id, 'count' => $count]);
        }
    } else {
        if (!isset($_SESSION['basket']) || !is_array($_SESSION['basket'])) {
            $_SESSION['basket'] = [];
        }
        if (isset($_SESSION['basket'][$product_id])) {

```

```

        $_SESSION['basket'][$product_id] += $count;
    } else {
        $_SESSION['basket'][$product_id] = $count;
    }
}

```

Висновки до розділу 2

В другому розділі кваліфікаційної роботи бакалавра було наведено результати детального аналізу вимог до розроблюваного програмного продукту, зокрема Інтернет-магазину дитячих іграшок. Визначено функціональні та нефункціональні вимоги від різних типів користувачів. Розглянуто часові характеристики реакції системи на дії користувачів, що забезпечують комфортне та ефективне користування сайтом.

Окрім цього, специфіковано варіанти використання системи, що охоплюють основні сценарії взаємодії користувачів з інтернет-магазином. Створено узагальнену схему елементів (сутностей системи), їх структури та методів із описом, що допомагає зрозуміти архітектуру та логіку роботи системи.

Було представлено фізичну схему даних та описано структуру збереження даних. Наведено схему бази даних, яка показує таблиці, їхні поля та взаємозв'язки між ними. Це допомагає зрозуміти, як дані організовані та керуються в системі, забезпечуючи ефективність та надійність роботи Інтернет-магазину.

Також були створені діаграми активності та відповідні таблиці специфікації, що ілюструють взаємодію користувачів та системи в різних сценаріях використання. Діаграми активності демонструють послідовність дій, які виконуються у конкретних випадках взаємодії з системою. Таблиці специфікації надають докладну інформацію про кожен крок в цих сценаріях, включаючи вхідні та вихідні дані, взаємодію з іншими компонентами системи та очікувані результати.

Були розглянуті ключові аспекти реалізації програми, включаючи архітектуру, використані технології та методи. Програмний код представлений у вигляді фрагментів, які ілюструють реалізацію певних функцій або модулів системи. Кожен фрагмент коду супроводжується поясненням, що розкриває його суть, особливості реалізації та способи взаємодії з іншими частинами системи.

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-САЙТОМ ІНТЕРНЕТ-МАГАЗИНУ ДИТЯЧИХ ІГРАШОК

3.1 Порядок налаштування параметрів системи

Розглянемо файли що відповідають за налаштування параметрів створеної системи:

```
.htaccess:
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^(.*)$ index.php?path=$1 [NC,L,QSA]
```

Цей файл є частиною налаштувань сервера Apache і використовується для перенаправлення запитів на різні URL-адреси. У цьому конкретному випадку, режим перезапису URL (RewriteEngine) увімкнений. Перші дві умови (RewriteCond) перевіряють, чи не існує вказаного файлу чи каталогу, щоб не виконувати перезапис, якщо файл чи каталог існують. Якщо обидві перевірки успішні, застосовується правило перезапису (RewriteRule), яке перенаправляє всі запити до файлу index.php з параметром "path", що містить оригінальний URL. Параметр [NC] означає "не чутливий до регістру" (не враховувати регістр символів), [L] - "останнє правило" (не виконувати інші правила), [QSA] - "приймати параметри запиту" (додати параметри запиту до нового URL).

```
config.php:
<?php
$Config = [
    'Database' => [
        'Server' => 'localhost',
        'Username' => 'cmsAdmin',
        'Password' => '1234',
        'Database' => 'cms'
    ],
    'NewsCount' => 5
];
```

Цей файл містить конфігураційні параметри для системи. У розділі 'Database' вказані параметри для підключення до бази даних, які використовуються для

забезпечення доступу до даних. Параметр 'NewsCount' визначає кількість новин, які будуть відображатися на сторінці. Ці значення можна змінити відповідно до потреб.

Розглянемо метод __autoload головного класу ядра системи Core:

```
public static function __autoload($className)
{
    $fileName = $className.'.php';
    if (is_file($fileName))
        include ($fileName);
}
```

Цей метод є автозавантажувачем класів і викликається автоматично при спробі створення екземпляра невідомого класу. Він завантажує файл класу, якщо такий існує.

Метод init головного класу ядра системи Core:

```
public function init()
{
    session_start();
    spl_autoload_register('\core\Core::__autoload');
    self::$mainTemplate = new Template();
}
```

Цей метод ініціалізує систему, запускаючи сесію та реєструючи автозавантажувач класів. Він також ініціалізує головний шаблон системи.

3.2 Структура інтерфейсу веб-сайту інтернет-магазину дитячих іграшок

Розглянемо інтерфейс розробленого інтернет-магазину. При відкритті веб-сайту користувач спочатку бачить головну сторінку, на якій відображаються всі товари. Товари яких немає в наявності відображаються в кінці сторінки. У верхній частині сторінки розміщується навігаційне меню, список бажаного та корзина [28].

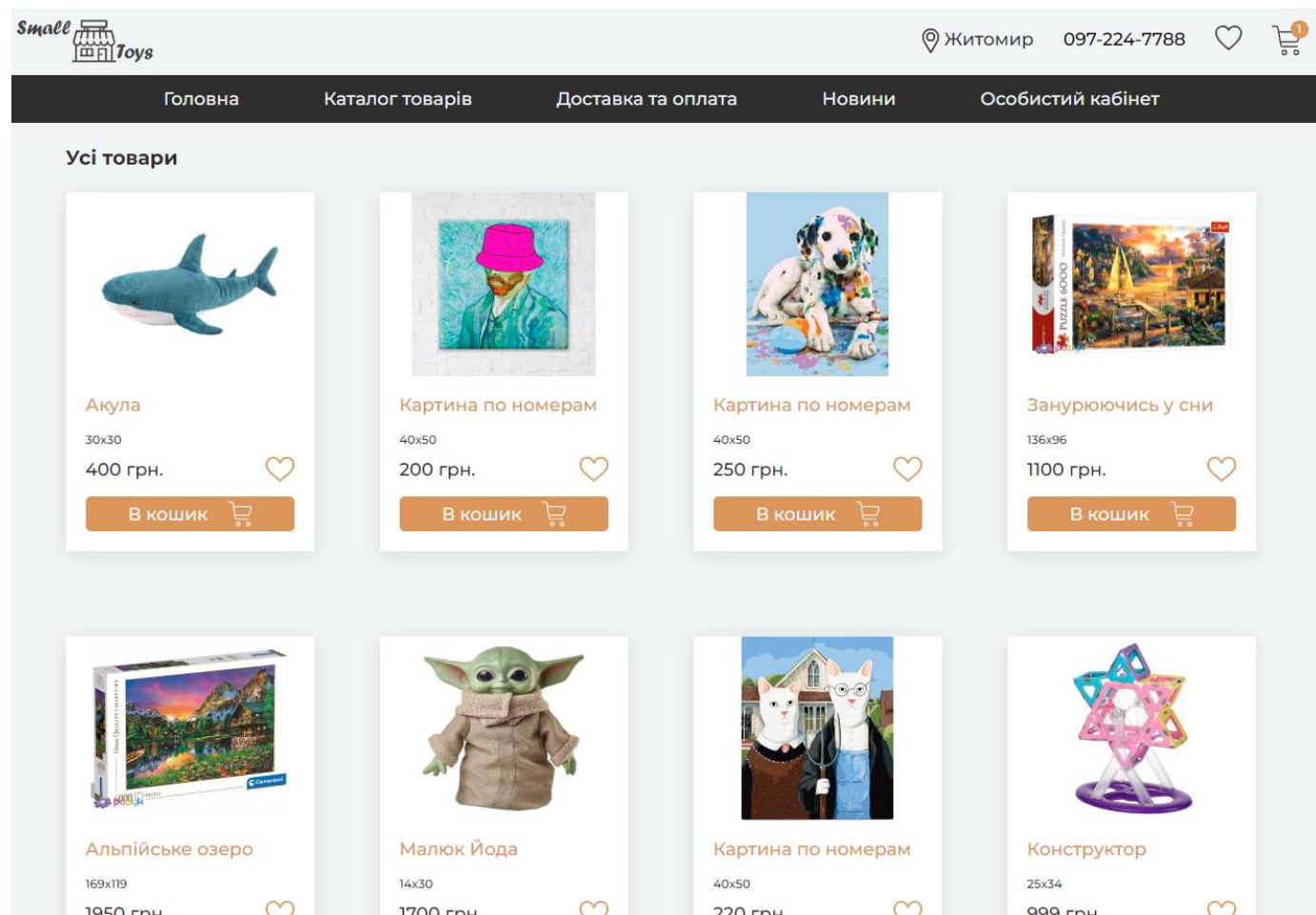


Рисунок 3.1 – Головна сторінка, верхня частина сайту

У нижній частині сторінки знаходяться контактні дані інтернет-магазину, соціальні мережі та графік роботи.

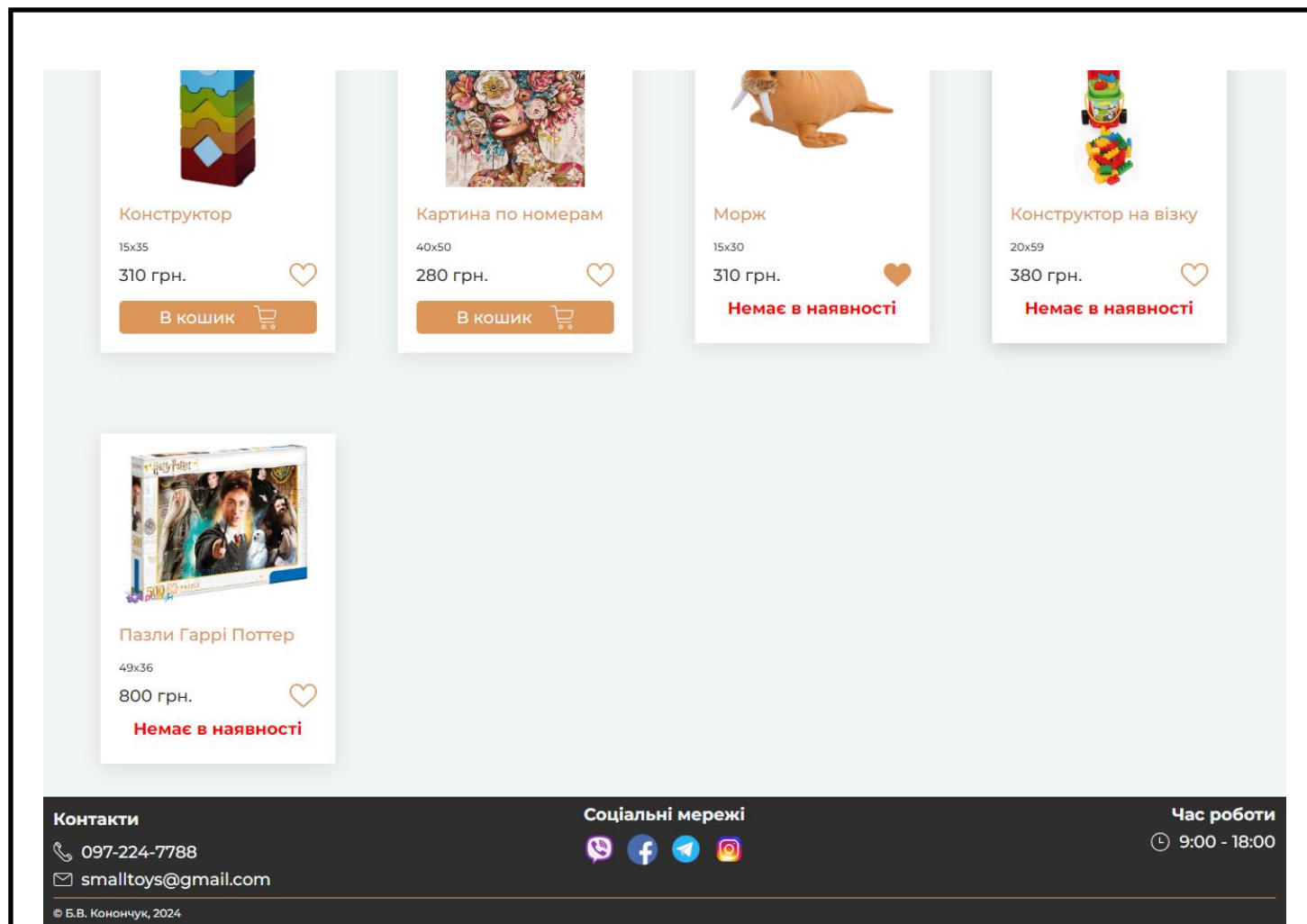


Рисунок 3.2 – Головна сторінка, нижня частина сайту

Користувач може переглянути детальну інформацію про товар, таку як ціна, категорія товару, вікова категорія, розмір та опис товару.

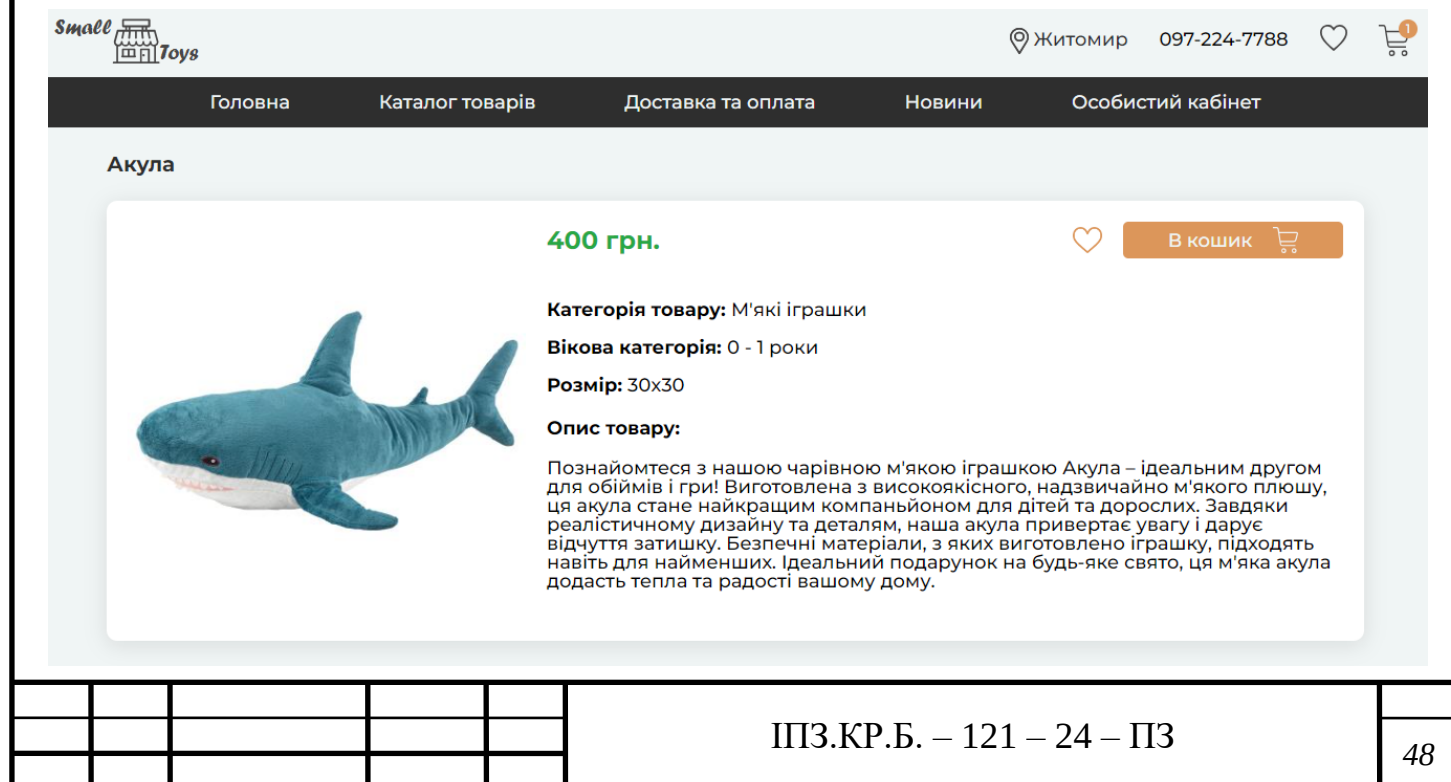


Рисунок 3.3 – Картка детальної інформації про продукт

На сторінці «Каталог товарів» користувач буде бачити список усіх доступних категорій продукції.

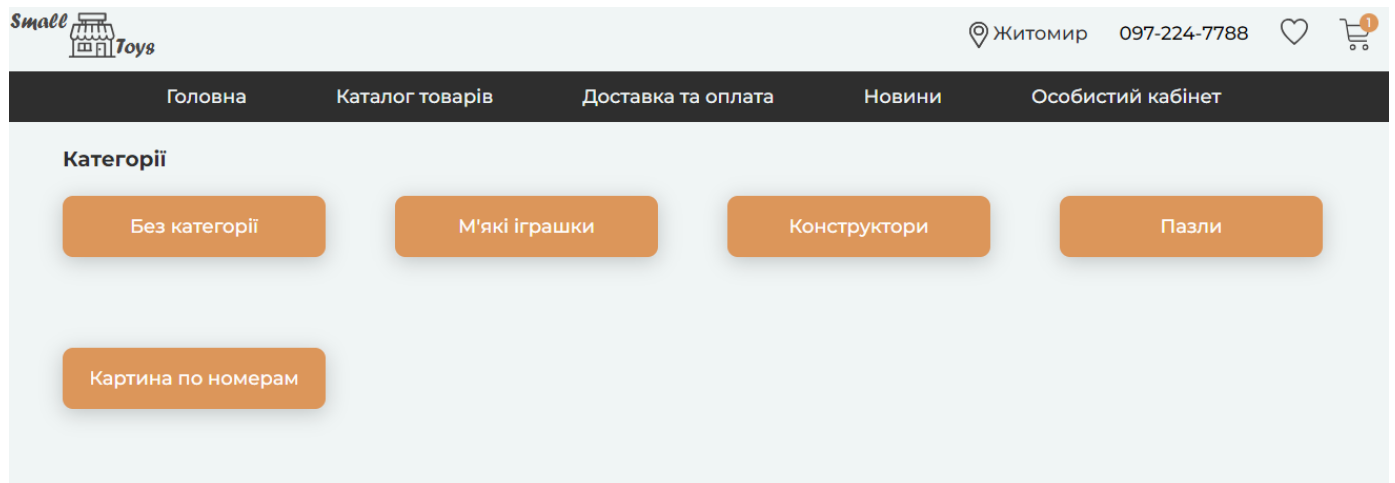


Рисунок 3.4 – Сторінка «Каталог товарів»

Коли користувач відкріє певну категорію, то отримає сторінку з усіма продуктами відповідної категорії.

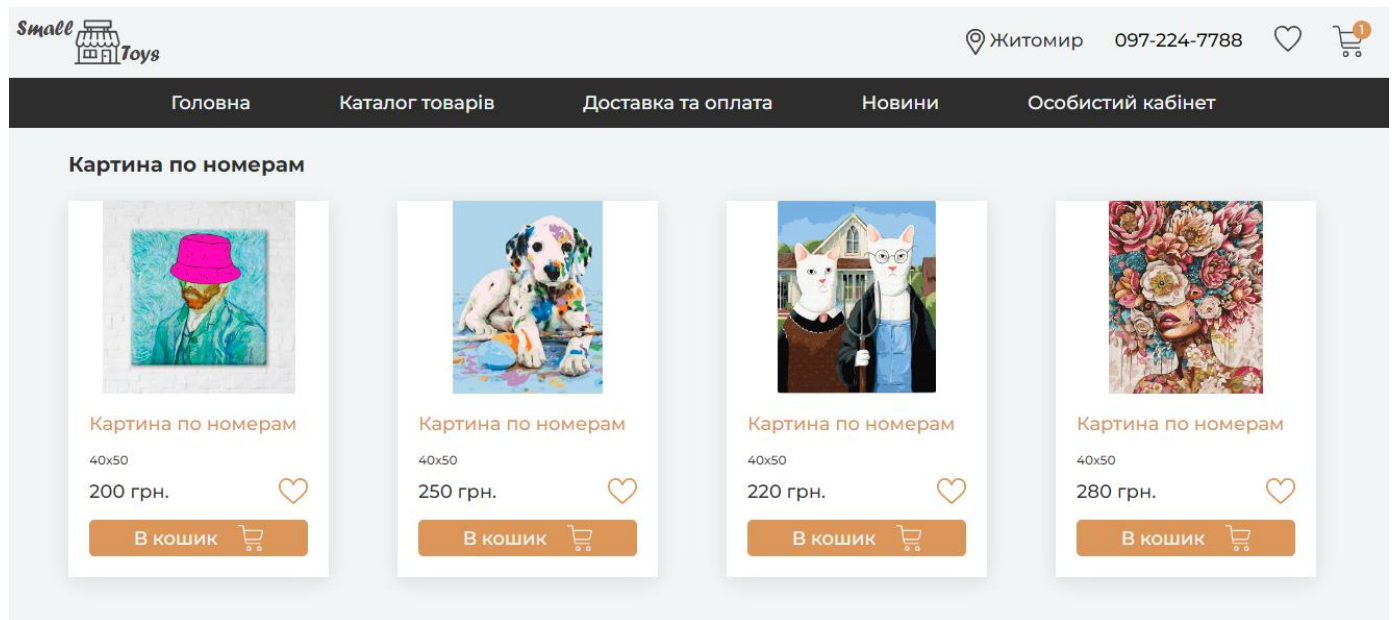


Рисунок 3.5 – Сторінка категорії товарів «Картини по номерам»

На сторінці «Доставка та оплата» користувач знайде всю необхідну інформацію про умови доставки та оплати інтернет-магазину.

Доставка та оплата

Способи доставки:

1. Адресна доставка з використанням кур'єрської служби "Нова Пошта" доступна по всій території України.
(за тарифами перевізника)
2. Доставка у відділення поштової служби "Нова Пошта" по всій території України.
(за тарифами перевізника)
3. Доставка у поштомат поштової служби "Нова Пошта" по всій території України.
(за тарифами перевізника)

Способи оплати:

1. Готівковий розрахунок
2. Через POS-термінал
3. Через смартфон кур'єра (при способі доставки №1)

Рисунок 3.6 – Сторінка «Доставка та оплата»

Коли користувач відкриє сторінку «Новини» то побачить список останніх новин.

Новини

Пазли та їх користь для дитячого розвитку



Збирання пазлів – одне з улюблених занять для дітей та дорослих при проведенні вільного часу. Це весела розвиваюча гра, суть якої полягає у тому, щоб з декількох десятків, сотень та навіть тисяч маленьких фрагментів, на які розділена картинка, відтворити її первісний вигляд.

[Читати далі](#)

Улюблені іграшки з безкоштовною доставкою!



Щоб покупки в нашому інтернет-магазині принесли ще більше задоволення, маємо для вас приємний сюрприз!

[Читати далі](#)

Рисунок 3.7 – Сторінка «Новини»

Натиснувши на кнопку «Читати далі» користувач може побачити повний текст новини.

Пазли та їх користь для дитячого розвитку



Збирання пазлів – одне з улюблених занять для дітей та дорослих при проведенні вільного часу. Це весела розвиваюча гра, суть якої полягає у тому, щоб з декількох десятків, сотень та навіть тисяч маленьких фрагментів, на які розділена картинка, відтворити її первісний вигляд. За допомогою цієї гри дитина розвиває у собі декілька корисних функцій, значимість яких важко переоцінити для її всебічного гармонійного розвитку.

Як тільки малюк навчиться складати пазли самостійно, можна одразу ж починати купувати головоломки з більшою кількістю елементів. Але пам'ятайте, що рівень їхньої складності необхідно підвищувати поступово, адже, наприклад, скласти картинку із 100 елементів дитині 3-х років буде досить непросто, хоча тут все залежить від індивідуального розвитку дитини.

Турботливим батькам слід пам'ятати, що спільне проведення часу під час укладання пазлів допомагає розвитку взаєморозуміння та взаємоповаги між батьками та дітьми. Дитина іноді потребуватиме допомоги від мами або тата, особливо тоді, коли пазли будуть ставати чимраз складнішими. Проте, ця допомога повинна бути помірною, оскільки самостійність та свобода дій дозволить дитині в майбутньому стати рішучою та успішною.

Рисунок 3.8 – Повний текст новини

В особистому кабінеті користувач знайде свої особисті дані та зможе їх оновити за потреби.

Особистий кабінет

Мої дані

Прізвище

Kononchuk

Ім'я

Bohdan

Email

ipz204_kbv@student.ztu.edu.ua

Номер телефону

0987654567

Оновити дані

Рисунок 3.9 – Сторінка «Особистий кабінет»

Також є можливість додати товар з акаунту адміністратора. Для додавання товарів було створено спеціальну форму.

Додавання товару

Назва товару

Опис товару

Параграф

B

I



Розмір

Категорія

Без категорії



Вікова категорія

Без вікової категорії



Ціна

Кількість

Відображати товар?

Так



Фото

Вибрати файл

Файл не вибрано

Зберегти

Рисунок 3.10 – Форма для додавання товару

3.3 Тестування роботи програмного комплексу

Проведемо функціональне тестування розробленого інтернет-магазину [29]. Функціональне тестування спрямоване на те, щоб кожна функція веб-сайту працювала відповідно до специфікації вимог. Тестування функціоналу веб-сайту показує "Що робить система" [30].

Для початку зареєструємось на сайті, введемо некоректні дані:

Small Toys

Житомир 097-224-7788

Головна Каталог товарів Доставка та оплата Новини Особистий кабінет

Реєстрація на сторінці

Реєстрація

testuser

Електронна адреса має містити знак "@". В електронній адресі "testuser" знака "@" немає.

Password

Lastname

Firstname

Зареєструватись

Вхід

Рисунок 3.11 – Форма для реєстрації, перевірка email

Реєстрація на сторінці

Поле "Пароль" не може бути порожнім!
Поле "Ім'я" не може бути порожнім!
Поле "Прізвище" не може бути порожнім!

Реєстрація

Email

Password

Password

Lastname

Firstname

Зареєструватись

Вхід

Рисунок 3.12 – Валідація форми реєстрації

Додамо декілька продуктів у список бажаного:

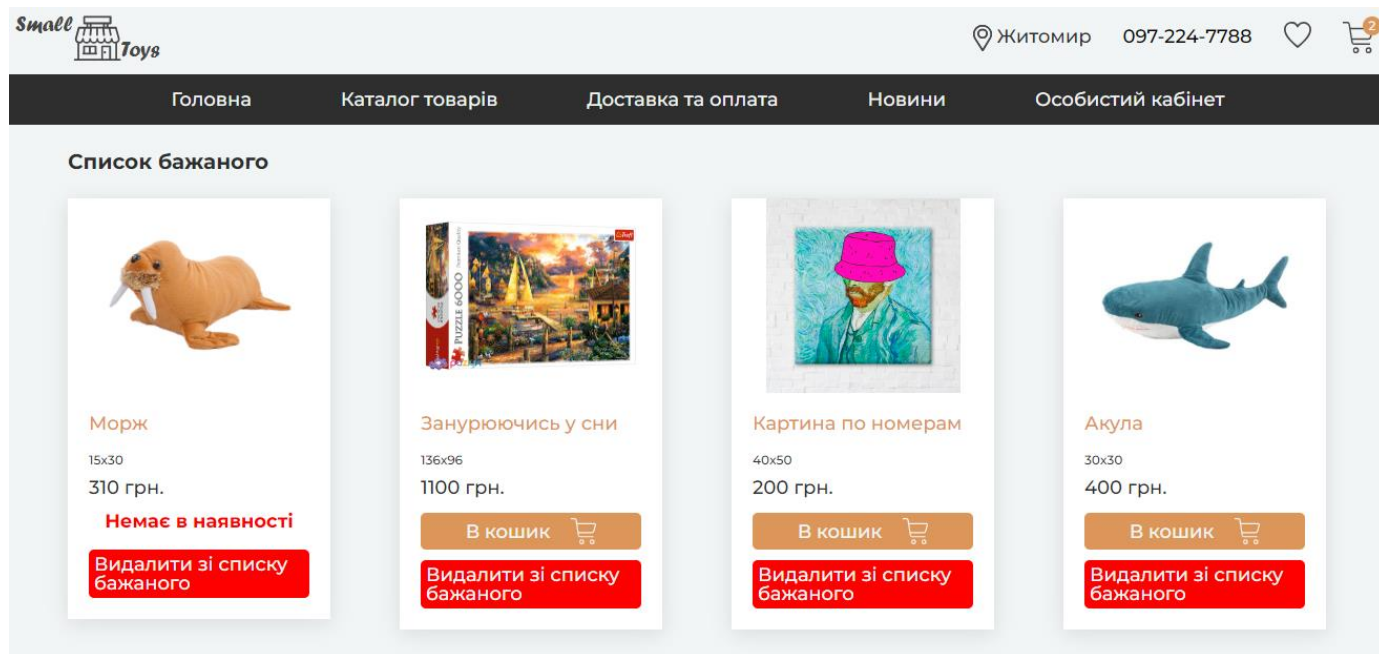


Рисунок 3.13 – Результат додавання товарів у список бажаного

Всі товари додаються в список бажаного та відображаються коректно. Тепер спробуємо видалити товар зі списку бажаного.

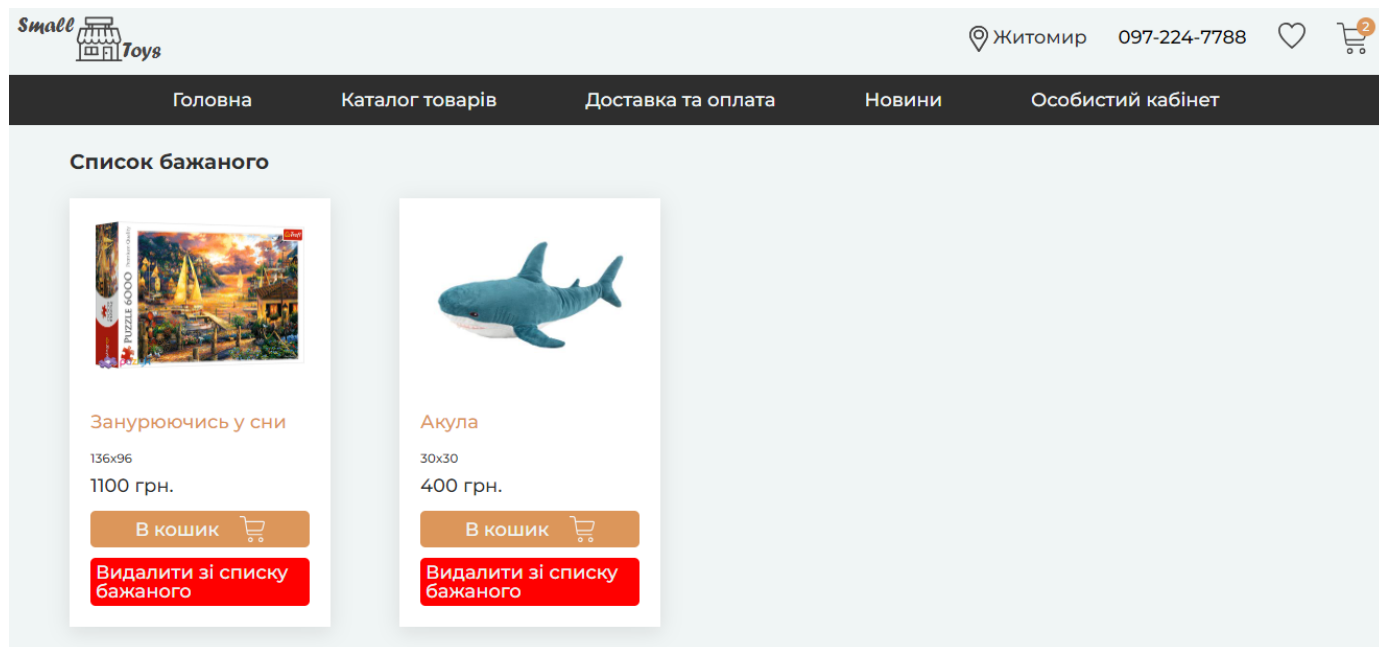


Рисунок 3.14 – Результат видалення товарів зі списку бажаного

Товари видаляються успішно. Далі перевіримо чи додаються товари в кошик по натисканню відповідної кнопки в картці продукту.

Кошик

Сума: 1500.00 грн.

Оформити замовлення



Занурюючись у сни

136x96

1100 грн.

Кількість:

1

Видалити з кошика



Акула

30x30

400 грн.

Кількість:

1

Видалити з кошика

Рисунок 3.15 – Результат додавання товарів у кошик

Усі товари успішно додалися у кошик користувача. Спробуємо змінити кількість кожного товару та перевіримо чи змінюється загальна сума.

Кошик

Сума: 4100.00 грн.

Оформити замовлення



Занурюючись у сни

136x96

1100 грн.

Кількість:

3

Видалити з кошика



Акула

30x30

400 грн.

Кількість:

2

Видалити з кошика

Рисунок 3.16 – Перевірка обчислення суми в кошику

Кількість товарів в кошику та сума змінюються динамічно на сторінці, без її оновлення.

Авторизуємось під акаунтом адміністратора то спробуємо додати новий товар.

Спочатку перевіримо чи працює валідація форми.

Додавання товару

Поле "Назва товару" не може бути порожнім!
Поле "Опис товару" не може бути порожнім!
Поле "Розмір" не може бути порожнім!
Поле "Ціна" не може бути порожнім!

Назва товару
Product name

Опис товару
Параграф **B** *I*

Розмір
Size

Категорія
Без категорії

Вікова категорія
Без вікової категорії

Ціна
Price

Рисунок 3.17 – Результат перевірки валідації форми

Валідація працює коректно, адміністратор не може додати продукт не заповнивши обов'язкові поля. Далі заповнимо форму коректними даними та перевіримо чи додається продукт на сайт.

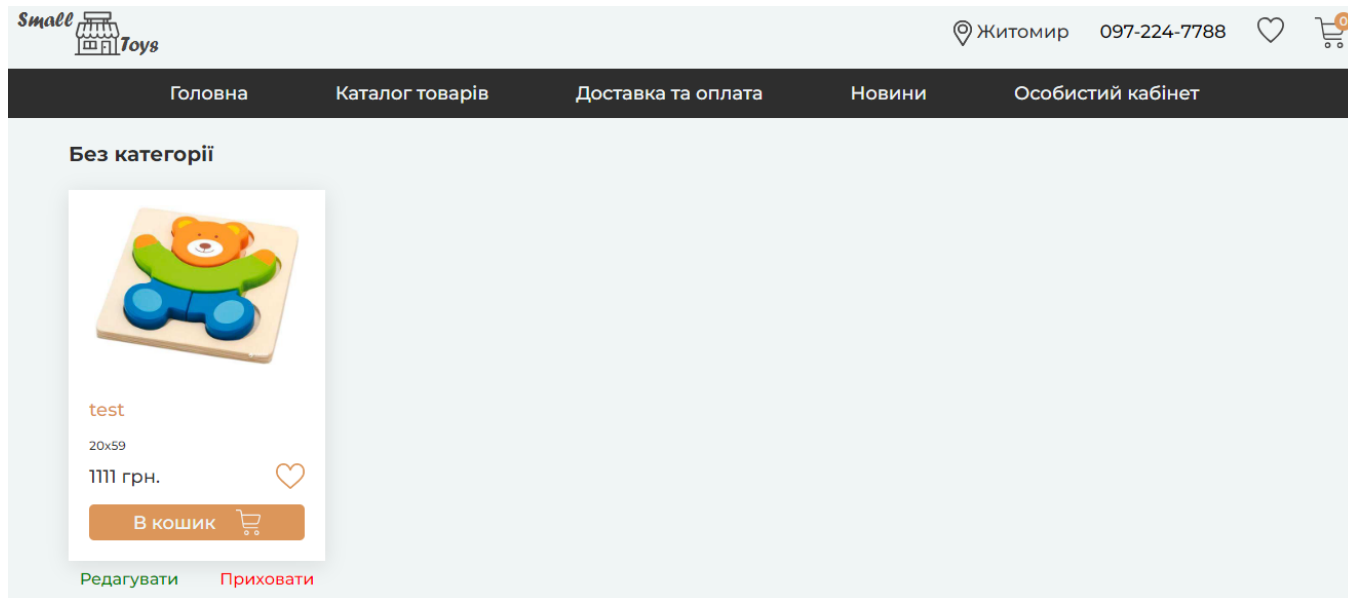


Рисунок 3.18 – Результат додавання нового товару

Новий товар було додано успішно. Далі спробуємо зняти товар з продажу, натиснувши відповідно кнопку під карткою продукту.

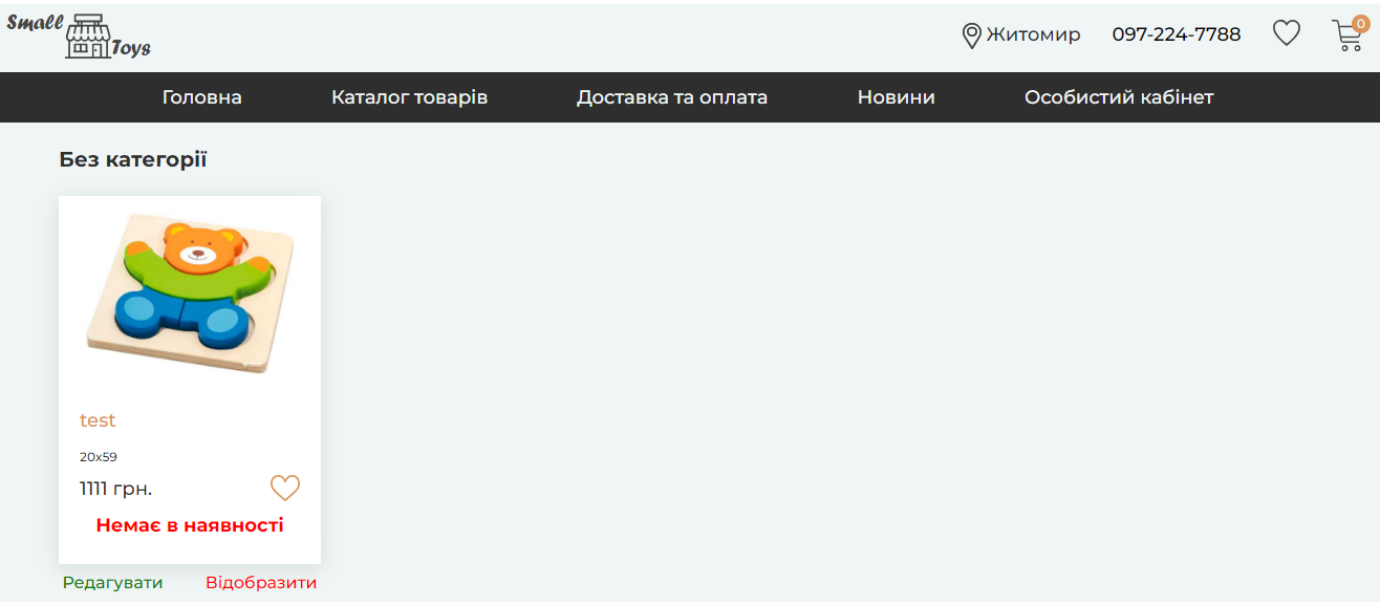


Рисунок 3.18 – Товар було знято з продажу

Товар було успішно знято з продажу, користувачі можуть додавати його в список бажаного, але не зможуть придбати даний товар, поки адміністратор не поверне товар у продаж.

Висновки до розділу 3

В третьому розділі кваліфікаційної роботи бакалавра детально розглянуто всі аспекти інтерфейсу реалізованої системи, порядок її роботи, налаштування, а також проведено тестування алгоритмів. Це дозволяє підтвердити точність і коректність роботи програмного комплексу.

Було наведено опис структури та основних елементів інтерфейсу додатку, що забезпечує зручність та інтуїтивність взаємодії користувача із системою. Розглянуто порядок роботи системи з наведенням тестового прикладу, який ілюструє функціонування основних компонентів і процесів. Представлено порядок налаштування та встановлення системи.

Цей розділ кваліфікаційної роботи бакалавра забезпечує повне розуміння того, як система виглядає, як працює, як її налаштовувати і як переконатися в правильності її роботи.

ВИСНОВКИ

У процесі виконання випускної кваліфікаційної роботи бакалавра було розроблено інтернет-магазин дитячих іграшок, що відповідає сучасним вимогам до таких систем. Основною метою роботи було створення функціонального та зручного у використанні веб-додатку, який забезпечує простий та швидкий доступ до широкого асортименту товарів для дітей.

У ході дослідження та розробки вдалося досягти таких результатів:

Основний результат відповідно до мети: Інтернет-магазин дитячих іграшок було успішно розроблено та реалізовано. Система дозволяє користувачам реєструватися, переглядати каталог товарів, додавати товари до кошика та списку бажаного, переглядати товари по категоріям, а також переглядати новини та інформацію про доставку та оплату.

Перелік реалізованих функцій із оцінкою ступеня відповідності завданню:

- Реєстрація та вхід в систему: Забезпечено можливість створення та входу в обліковий запис, що відповідає завданню щодо управління користувачами.
- Перегляд каталогу товарів та детальної інформації про продукти: Реалізовано функціонал для зручного перегляду та фільтрації товарів за категоріями.
- Додавання товарів до списку бажаного та кошику: Користувачі можуть легко додавати товари до кошику та списку бажаного.
- Перегляд новин: Реалізовано можливість перегляду останніх новин, що відповідає завданню інформування користувачів.
- Перегляд особистого кабінету: Користувачі можуть переглядати та оновлювати свої особисті дані.
- Керування товарами, категоріями та новинами адміністратором: Адміністратор має можливість додавати, змінювати та видаляти товари, категорії та новини, що відповідає завданню адміністрування системи.

Практична цінність результатів:

Розроблений інтернет-магазин дитячих іграшок має високу практичну цінність, оскільки забезпечує зручний доступ до широкого асортименту товарів для дітей. Система є гнучкою та масштабованою, що дозволяє легко додавати нові функції та розширювати асортимент товарів.

Напрямки подальшого наукового дослідження та вдосконалення програмного комплексу:

Для подальшого вдосконалення системи можна рекомендувати такі напрямки:

- Впровадження функціоналу оформлення замовлень, онлайн оплати та перегляду історії замовлень.
- Оптимізація швидкості завантаження сторінок та обробки запитів для підвищення продуктивності системи.
- Розробка мобільної версії інтернет-магазину для забезпечення зручного доступу з мобільних пристроїв.
- Впровадження системи лояльності для постійних клієнтів, що включатиме бонусні програми та знижки.

Таким чином, розроблений програмний комплекс має потенціал для подальшого розвитку та вдосконалення, що сприятиме підвищенню ефективності його функціонування та задоволенню потреб користувачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Розробка інтернет-магазину [Електронний ресурс] – Режим доступу до ресурсу: <https://esfirum.com/websites-development/ecommerce-development/>.
2. СУЧАСНІ ТЕНДЕНЦІЇ РОЗВИТКУ ЕЛЕКТРОННОЇ КОМЕРЦІЇ [Електронний ресурс]. – 2023. – Режим доступу до ресурсу: https://dspace.uzhnu.edu.ua/jspui/bitstream/lib/54992/1/%D0%97%D0%B0%D1%8F%D1%86%D1%8C_%D0%9A%D0%B0%D0%BF%D0%BA%D0%BE.pdf.
3. АВТОМАТИЗАЦІЯ БІЗНЕС-ПРОЦЕСІВ [Електронний ресурс] – Режим доступу до ресурсу: <https://kaf-av.tntu.edu.ua/index.php/mn-abiturient/mn-articles/740-art-automation-business-processes>.
4. Що таке SSL-сертифікат [Електронний ресурс] – Режим доступу до ресурсу: <https://hostiq.ua/ukr/info/what-is-ssl/>.
5. Інтернет-магазин \"Будинок Іграшок\" [Електронний ресурс] – Режим доступу до ресурсу: <https://bi.ua/ukr/>.
6. Інтернет-магазин \"BabyPlus\" [Електронний ресурс] – Режим доступу до ресурсу: <https://babyplus.ua/ua>.
7. Архітектура програмного забезпечення [Електронний ресурс] – Режим доступу до ресурсу: <https://wezom.com.ua/ua/blog/arhitektura-programmnogo-obespecheniya>.
8. ПАТЕРНИ ПРОЕКТУВАННЯ [Електронний ресурс] – Режим доступу до ресурсу: <https://refactoring.guru/uk/design-patterns>.
9. 9 Software Architecture Patterns for System Design Interviews [Електронний ресурс] – Режим доступу до ресурсу: <https://medium.com/javarevisited/9-software-architecture-patterns-for-system-design-interviews-d76793592903>.
10. Монолітна архітектура ПЗ. [Електронний ресурс] – Режим доступу до ресурсу: <https://qalight.ua/baza-znaniy/shho-take-monolitna-arhitektura/>.
11. HTML Tutorial [Електронний ресурс] – Режим доступу до ресурсу: <https://www.w3schools.com/html/>.

12. CSS Tutorial [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.w3schools.com/css/>.
13. Сучасний підручник з JavaScript [Електронний ресурс] – Режим доступу до ресурсу: <https://uk.javascript.info/>.
14. PHP Manual [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.php.net/manual/en/index.php>.
15. APACHE [Електронний ресурс] – Режим доступу до ресурсу:
<https://httpd.apache.org/>.
16. NGINX [Електронний ресурс] – Режим доступу до ресурсу: <https://nginx.org/en/>.
17. Creating Your Own CMS with PHP: A Step-by-Step Guide [Електронний ресурс] – Режим доступу до ресурсу: <https://achaltechnologies.hashnode.dev/creating-your-own-cms-with-php-a-step-by-step-guide>.
18. Що таке СУБД і для чого вони потрібні [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/systema-upravlinnia-bazamy-danykh/>.
19. MySQL [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.mysql.com/>.
20. ДЛЯ ЧОГО ПОТРІБНІ UML ДІАГРАМИ? [Електронний ресурс] – Режим доступу до ресурсу: <https://foxminded.ua/uml-diagramy/>.
21. UML для бізнес-моделювання: для чого потрібні діаграми процесів [Електронний ресурс] – Режим доступу до ресурсу:
<https://evergreens.com.ua/ua/articles/uml-diagrams.html>.
22. MVC Design Pattern [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.geeksforgeeks.org/mvc-design-pattern/>.
23. Побудова динамічних веб-застосунків за допомогою MVC [Електронний ресурс] – Режим доступу до ресурсу: <https://blog.ithillel.ua/articles/building-dynamic-web-applications-using-mvc>.
24. Глоба В. С. Розробка інформаційних ресурсів та систем. [Електронний ресурс] / В. С. Глоба. – 2020. – Режим доступу до ресурсу:

http://library.megu.edu.ua:8180/jspui/bitstream/123456789/1768/1/global_s_rozpodileni_sistemi_ta_merezhi_tom_1.pdf.

25. Розломий І. О. Фреймворки для розробки серверної частини web-додатків [Електронний ресурс] / І. О. Розломий, С. В. Науменко. – 2022. – Режим доступу до ресурсу: <https://kntu.kr.ua/doc/science/zahody/vikl/2022/2-tez.pdf#page=42>.
26. Introduction to tables [Електронний ресурс] – Режим доступу до ресурсу: <https://support.microsoft.com/en-us/office/introduction-to-tables-78ff21ea-2f76-4fb0-8af6-c318d1ee0ea7>.
27. Принципи веб-дизайну інтерфейсу користувача [Електронний ресурс] – Режим доступу до ресурсу: <https://stfalcon.com/uk/blog/post/user-interface-web-design-principles>.
28. Modern tools and current trends in web-development [Електронний ресурс]. – 2021. – Режим доступу до ресурсу: https://www.theseus.fi/bitstream/handle/10024/336520/Joel_Muittari.pdf.
29. What is UI Software testing? [Електронний ресурс] – Режим доступу до ресурсу: <https://www.zaptest.com/what-is-ui-software-testing-deep-dive-into-the-types-process-tools-implementation>.
30. Сорокун С. В. Особливості процесу тестування веб-сайту [Електронний ресурс] / С. В. Сорокун, Т. В. Коваленко. – 2021. – Режим доступу до ресурсу: <https://sci-conf.com.ua/wp-content/uploads/2021/10/INNOVATIONS-AND-PROSPECTS-OF-WORLD-SCIENCE-6-8.10.21.pdf#page=192>.

ДОДАТКИ

					ІПЗ.КР.Б. – 121 – 24 – ПЗ	
						64

Клас Core, головний клас ядра системи:

```
<?php
namespace core;
class Core
{
    private static $instance;
    private static $mainTemplate;
    private static $db;
    private function __construct()
    {
        global $Config;
        spl_autoload_register('\core\Core::__autoload');
        self::$db = new \core\DB($Config['Database']['Server'],
            $Config['Database']['Username'],
            $Config['Database']['Password'],
            $Config['Database']['Database']);
    }
    /**
     * Повертає екземпляр ядра системи
     * @return Core
     */
    public static function getInstance()
    {
        if (empty(self::$instance)) {
            self::$instance = new Core();
            return self::getInstance();
        }
        else
            return self::$instance;
    }
    /**
     * Отримати об'єкт з'єднання з БД
     */
    public function getDB(){
        return self::$db;
    }
    /**
     * Ініціалізація системи
     */
    public function init()
    {
        session_start();
        spl_autoload_register('\core\Core::__autoload');
        self::$mainTemplate = new Template();
    }
    /**
     * Виконує основний процес роботи CMS системи
     */
    public function run()
```

```

{
    $path = $_GET['path'];
    $pathParts = explode('/', $path);
    $className = ucfirst($pathParts[0]);
    if (empty($className))
        $fullClassName = 'controllers\\Products';
    else
        $fullClassName = 'controllers\\'.$className;
    $methodName = ucfirst($pathParts[1]);
    if (empty($methodName))
        $fullMethodName = 'actionIndex';
    else
        $fullMethodName = 'action'.$methodName;
    if (class_exists($fullClassName)) {
        $controller = new $fullClassName();
        if (method_exists($controller, $fullMethodName)){
            $method = new \ReflectionMethod($fullClassName, $fullMethodName);
            $paramsArray = [];
            foreach ($method->getParameters() as $parameter)
            {
                array_push($paramsArray, isset($_GET[$parameter->name]) ? $_GET[$parameter-
>name] : null);
            }
            $result = $method->invokeArgs($controller, $paramsArray);
            if (is_array($result)){
                self::$mainTemplate->setParams($result);
            }
        }
        else
            throw new \Exception('404 Not Found');
    }
    else
        throw new \Exception('404 Not Found');
}
/**
 * Автозавантажувач класів
 * @param $className string Назва класу
 */
public static function __autoload($className)
{
    $fileName = $className.'.php';
    if (is_file($fileName))
        include ($fileName);
}
/**
 * Завершення роботи системи та видалення результату
 */
public function done(){
    self::$mainTemplate->display('views/layout/index.php');
}

```

```
}
```

Клас Products для оперування продуктами на сайті:

```
<?php
namespace models;
use core\Model;
use core\Utils;
class Products extends Model
{
    /**
     * Метод для додавання товару
     */
    public function AddProduct($row){
        $userModel = new \models\Users();
        $user = $userModel->GetCurrentUser();
        if ($user == null || $user['access'] != 1) {
            $result = [
                'error' => true,
                'messages' => ['Доступ заборонено!']
            ];
            return $result;
        }
        $validateResult = $this->Validate($row);
        if (is_array($validateResult)){
            $result = [
                'error' => true,
                'messages' => $validateResult
            ];
            return $result;
        }
        $fields = ['name', 'description', 'size', 'category_id', 'agecategory_id', 'price', 'amount', 'visible',
'photo'];
        $rowFiltered = Utils::ArrayFilter($row, $fields);
        if ($rowFiltered['amount'] == 0)
            $rowFiltered['visible'] = 0;
        $id = \core\Core::getInstance()->getDB()->insert('products', $rowFiltered);
        return [
            'error' => false,
            'id' => $id
        ];
    }
    /**
     * Метод повертає всі товари
     */
    public function GetAllProducts(){
        return \core\Core::getInstance()->getDB()->select('products', '*', null, ['visible' => 'DESC']);
    }
    /**
     * Метод повертає всі товари певної категорії
     */
}
```

```

public function GetProductsByCategory($category_id)
{
    return \core\Core::getInstance()->getDB()->select('products', '*', ['category_id' => $category_id],
['visible' => 'DESC']);
}
/**
 * Метод повертає товар за id
 */
public function GetProductById($id){
    $product = \core\Core::getInstance()->getDB()->select('products', '*', ['id' => $id]);
    if (!empty($product))
        return $product[0];
    else
        return null;
}
/**
 * Метод для зміни товару
 */
public function UpdateProduct($row, $id){
    $userModel = new \models\Users();
    $user = $userModel->GetCurrentUser();
    if ($user == null || $user['access'] != 1)
        return false;
    $validateResult = $this->Validate($row);
    if (is_array($validateResult))
        return $validateResult;
    $fields = ['name', 'description', 'size', 'category_id', 'agecategory_id', 'price', 'amount', 'visible',
'photo'];
    $rowFiltered = Utils::ArrayFilter($row, $fields);
    if ($rowFiltered['amount'] == 0)
        $rowFiltered['visible'] = 0;
    \core\Core::getInstance()->getDB()->update('products', $rowFiltered, ['id' => $id]);
    return true;
}
/**
 * Метод для заміни фото у товарі
 */
public function ChangePhoto($id, $file){
    $folder = 'files/products/';
    $products = $this->GetProductById($id);
    if (is_file($folder.$products['photo']) && is_file($folder.$file))
        unlink($folder.$products['photo']);
    $products['photo'] = $file;
    $this->UpdateProduct($products, $id);
}
/**
 * Метод для позначення товару відсутнім в наявності
 */
public function HideProduct($id)
{

```

```

        \core\Core::getInstance()->getDB()->update('products', ['visible' => 0], ['id' => $id]);
        return true;
    }
    /**
     * Метод для позначення товару присутнім в наявності
     */
    public function DisplayProduct($id)
    {
        $errors = [];
        $product = $this->GetProductById($id);
        if($product['amount'] == 0) {
            $errors [] = 'Товар відсутній на складі! Неможливо повернути в продаж!';
        }
        if (count($errors) > 0) {
            return $errors;
        }
        else {
            \core\Core::getInstance()->getDB()->update('products', ['visible' => 1], ['id' => $id]);
            return true;
        }
    }
    /**
     * Метод для валідації
     */
    public function Validate($formRow){
        $errors = [];
        if (empty($formRow['name']))
            $errors [] = 'Поле "Назва товару" не може бути порожнім!';
        if (empty($formRow['description']))
            $errors [] = 'Поле "Опис товару" не може бути порожнім!';
        if (mb_strlen($formRow['description'], 'UTF-8')>500)
            $errors [] = 'Поле "Опис товару" не може бути більше 500 символів!';
        if (empty($formRow['size']))
            $errors [] = 'Поле "Розмір" не може бути порожнім!';
        if (empty($formRow['price']))
            $errors [] = 'Поле "Ціна" не може бути порожнім!';
        if (count($errors) > 0)
            return $errors;
        else
            return true;
    }
}

```

Клас Users для оперування користувачами :

```

<?php
namespace models;
use core\Model;
use core\Utils;
class Users extends Model
{

```

```

/**
 * Метод для реєстрації
 */
public function AddUser($userRow){
    $validateResult = $this->Validate($userRow);
    if (is_array($validateResult))
        return $validateResult;
    $fields = ['login', 'password', 'firstname', 'lastname'];
    $userRowFiltered = Utils::ArrayFilter($userRow, $fields);
    $userRowFiltered['password'] = md5($userRowFiltered['password']);
    \core\Core::getInstance()->getDB()->insert('users', $userRowFiltered);
    return true;
}
/**
 * Метод для входу на сайт
 */
public function AuthUser($login, $password){
    $password = md5($password);
    $users = \core\Core::getInstance()->getDB()->select('users', "*", [
        'login' => $login,
        'password' => $password
    ]);
    if (count($users) == 1){
        $user = $users[0];
        return $user;
    } else
        return false;
}
/**
 * Пошук користувача за логіном
 */
public function GetUserByLogin($login){
    $rows = \core\Core::getInstance()->getDB()->select('users', '*', ['login' => $login]);
    if (count($rows) > 0)
        return $rows[0];
    else
        return null;
}
/**
 * Метод перевіряє чи користувач уже залогінвся на сайт
 */
public function IsUserAuthenticated(){
    return isset($_SESSION['user']);
}
/**
 * Метод для отримання поточного користувача
 */
public function GetCurrentUser(){
    if($this->IsUserAuthenticated())
        return $_SESSION['user'];
}

```

```

        else
            return null;
    }
    /**
     * Метод для оновлення даних користувача
     */
    public function UpdateUser($userRow, $id){
        $validateResult = $this->ValidateUpdate($userRow);
        if (is_array($validateResult))
            return $validateResult;
        $fields = ['login', 'password', 'lastname', 'firstname', 'telnumber'];
        $userRowFiltered = Utils::ArrayFilter($userRow, $fields);
        \core\Core::getInstance()->getDB()->update('users', $userRowFiltered, ['id' => $id]);
        return true;
    }
    /**
     * Метод для валідації реєстрації
     */
    public function Validate($formRow){
        $errors = [];
        if (empty($formRow['login'])) {
            $errors[] = 'Поле "Email" не може бути порожнім!';
        } elseif (!filter_var($formRow['login'], FILTER_VALIDATE_EMAIL)) {
            $errors[] = 'Неправильний формат електронної пошти!';
        }
        $user = $this->GetUserByLogin($formRow['login']);
        if (!empty($user))
            $errors[] = 'Користувач вже існує!';
        if (empty($formRow['password']))
            $errors[] = 'Поле "Пароль" не може бути порожнім!';
        if ($formRow['password'] != $formRow['password2'])
            $errors[] = 'Паролі не співпадають!';
        if (empty($formRow['firstname']))
            $errors[] = 'Поле "Ім'я" не може бути порожнім!';
        if (empty($formRow['lastname']))
            $errors[] = 'Поле "Прізвище" не може бути порожнім!';
        if (count($errors) > 0)
            return $errors;
        else
            return true;
    }
    /**
     * Метод для валідації реєстрації
     */
    public function ValidateUpdate($formRow)
    {
        $errors = [];
        if (empty($formRow['lastname']))
            $errors[] = 'Поле "Прізвище" не може бути порожнім!';
        if (empty($formRow['firstname']))

```

```

        $errors [] = 'Поле "Ім'я" не може бути порожнім!';
    if (empty($formRow['telnumber']))
        $errors [] = 'Поле "Номер телефону" не може бути порожнім!';
    elseif (!preg_match('/^0[0-9]{9}$/', $formRow['telnumber']))
        $errors[] = 'Номер телефону має починатися з нуля, містити лише цифри і мати довжину 10
символів!';
    if (count($errors) > 0)
        return $errors;
    else
        return true;
    }
}

```