КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ТОРГОВЕЛЬНО-ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук та інформаційних технологій

ВИПУСКНИЙ КВАЛІФІКАЦІЙНИЙ ПРОЕКТ

на тему:

«Розробка Web-додатку для електронної торгівлі побутовою технікою з інтерактивним модулем доставки товарів»

Студента 4 курсу, 10 групи,
спеціальності
122 «Комп'ютерні науки»

_____
підпис студента

Лихошапка
Богдана
Олександровича

Науковий керівник
кандидат фізико-математичних
наук, доцент

_____
підпис керівника

Самойленко
Анна
Тимофіївна

Гарант освітньої програми
кандидат технічних наук, доцент

_____
підпис керівника

Демідов
Павло
Георгійович

Київ 2021

Київський національний торговельно-економічний університет

Факультет інформаційних технологій
Кафедра комп'ютерних наук та інформаційних систем
Спеціальність 122 «Комп'ютерні науки»

Затверджую
Зав. кафедри_____Пурський О.І.
«15» грудня 2020р

Завдання
на випускну кваліфікаційну роботу (проект) студенту

Лихошапка Богдана Олександровича
(прізвище, ім'я, по батькові)

1. Тема випускної кваліфікаційної роботи(проекту)

« Розробка Web-додатку для електронної торгівлі побутовою технікою з інтерактивним модулем доставки товарів*».*

2. Строк здачі студентом закінченої роботи *31 травня 2021 року.*

3. Цільова установка та вихідні дані до роботи

Мета роботи: розробка прототипу та працюючої реалізації інтернет магазину для електронної торгівлі побутовою технікою з інтерактивним модулем доставки товарів.

Об'єкт дослідження: способи реалізації сучасних Web-додатків та їх функціонування в мережі Інтернет.

Предмет дослідження: інформаційні технології в системі функціонування електронної комерції в Інтернеті.

4. Перелік графічного матеріалу _____
_____
_____
_____

5. Консультанти по роботі із зазначенням розділів, за якими здійснюється консультування:

Розділ	Консультант (прізвище, ініціали)	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Самойленко А.Т.	15.12.2020 р.	15.12.2020 р.
2	Самойленко А.Т.	15.12.2020 р.	15.12.2020 р.
3	Самойленко А.Т.	15.12.2020 р.	15.12.2020 р.

6. Зміст випускної кваліфікаційної роботи (проекту) (перелік питань за кожним розділом)

ВСТУП

РОЗДІЛ 1. ЗАГАЛЬНІ ВІДОМОСТІ ПРО ВЕБ-РОЗРОБКУ, ПЕРЕВАГИ ТА ПРИЧИНИ КОМЕРЦІАЛІЗАЦІЇ ІНТЕРНЕТУ

1.1. Поняття веб-розробки та її основні етапи.

1.2. Причини комерціалізації інтернету та основні цілі створення інтернет магазину.

РОЗДІЛ 2. ТЕХНІЧНІ ОСНОВИ РЕАЛІЗАЦІЇ ПРОЕКТУ

2.1 Функціональне розділення компонентів веб-додатку.

2.2 Опис основних програмних засобів для реалізації додатку .

2.3 Змішана модель взаємодії веб-компонентів системи та її переваги.

РОЗДІЛ 3. ОПИС ТЕХНІЧНОЇ РЕАЛІЗАЦІЇ ПРОЕКТУ

3.1 Модель функціонування додатку.

3.2 UML діаграма бази даних.

3.3 MVC реалізація додатку.

3.4 Архітектура роботи геомодуля.

ВИСНОВКИ

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТОК

7. Календарний план виконання роботи

№ Пор.	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	фактично
1	2	3	4
1	*Вибір теми випускної кваліфікаційної роботи*	*01.10.2020*	*01.10.2020*
2	*Розробка та затвердження завдання на випускну кваліфікаційну роботу*	*15.12.2020*	*15.12.2020*
3	*Вступ*	*03.02.2021*	
4	*РОЗДІЛ 1. Теоретичні аспекти механізмів оцінювання рівня соціально-економічного розвитку*	*26.02.2021*	
5	*РОЗДІЛ 2. Математична модель визначення інтегрального показника рівня соціально-економічного розвитку*	*06.04.2021*	
6	*РОЗДІЛ 3. Автоматизована система оцінювання показників соціально-економічного розвитку регіонів України*	*12.05.2021*	
7	*Висновки*	*15.05.2021*	
8	*Здача випускної кваліфікаційної роботи на кафедру науковому керівнику*	*20.05.2021*	
9	*Попередній захист випускної кваліфікаційної роботи*	*26.05.2021*	
11	*Виправлення зауважень, зовнішнє рецензування випускної кваліфікаційної роботи*	*27.05.2021*	
12	*Представлення готової зшитої випускної кваліфікаційної роботи на кафедру*	*31.05.2021*	
13	*Публічний захист випускної кваліфікаційної роботи*	*За розкладом роботи ЕК*	

8. Дата видачі завдання «15» грудня 2020 р.

Керівник випускної кваліфікаційної роботи (проекту) Самойленко А.Т.
(прізвище, ініціали, підпис)

Гарант освітньої програми Демідов П.Г.
(прізвище, ініціали, підпис)

Завдання прийняв студент-дипломник Лихошапко Б.О.
(прізвище, ініціали, підпис)

9. Відгук керівника випускної кваліфікаційної роботи (проекту)

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Керівник випускної кваліфікаційної роботи (проекту)
_____ 31.05.2021 р.
(підпис, дата)

10.Висновок про випускну кваліфікаційну роботу (проект)

Випускна кваліфікаційна робота (проект) студента _____
(прізвище, ініціали)
може бути допущена до захисту в екзаменаційній комісії.

Гарант освітньої програми_____ Демідов П.Г.
(підпис, прізвище, ініціали)

Завідувач кафедри_____ Пурський О.І.
(підпис, прізвище, ініціали)

«_____»_____2021 р.

Анотація

У випускній кваліфікаційній роботі здійснено комплексну розробку Web-додатку для електронної торгівлі побутової технікою з використанням модуля геолокації, що відслідковує місцеположення кур'єра товарів у режимі реального часу. У роботі розглядаються основи функціонування глобальної мережі інтернет та основні технічні можливості та засоби реалізації програмної частини веб-застосунків. Також було запропоновано та реалізовано архітектурний паттерн MVC на back-end частині додатку та використання асинхронного коду у мові програмування JavaScript на стороні клієнта, з подальшим використанням зібраних геодоних клієнтом на веб-платформі.

Ключові слова: веб-додаток, глобальна мережа інтернет, модуль геолокації, паттерн проектування MVC, back-end, front-end, асинхронний код.

Annotation

In the final qualification work was realized the complex development of the Web-application for e-commerce trading of household appliances with the use of the geolocation module that tracking the location of the courier of goods in real time. In project we browse the basics of the global network named Internet and the main technical capabilities and tools for implementing the software part of web applications. The MVC architectural pattern on the back-end part of the application and the usage of asynchronous code in the client-side JavaScript programming language were also proposed and implemented, with the subsequent use of the collected geodata by the client on a web platform.

Keywords: web application, Internet, geolocation module, MVC design pattern, back-end, front-end, asynchronous code.

ЗМІСТ

ВСТУП

Інтернет-бізнес починає привертати все більше значення в сучасному світі, і в нашій країні відповідно, у зв'язку з тенденцією до загальної глобалізації економіки. Інтернет і електронна торгівля грають в цьому процесі одну з найважливіших ролей. Поява абсолютного нового виду зв'язку зумовило перегляд підприємців процесу організації бізнесу.

Якщо 15 років тому компанія виходила в мережу виключно для підтримки свого престижу, то сьогодні – це як мінімум ще один значний канал продажів. В кризові 2008-2009 роки, підприємства традиційного сектора торгівлі були змушені скоротити штат працівників і обсяги продажів, при цьому інтернет-сектор навпаки збільшив торгівельний оборот за даний період на 46%. Якщо дивитися на теперішній час, то в умовах пандемії, бізнес починає збільшувати свою присутність у електронній комерції та переводити традиційні бізнес процеси в режим онлайн, виграючи у конкурентів, які не наважилися на подібні бізнес рішення раніше.

Інтернет відкриває широкі можливості не тільки для великих підприємств. Багато підприємців при старті розвитку власного бізнесу відразу відкривають свою справу в Інтернеті, так як для цього потрібна менша сума первинних інвестицій, ніж у сфері традиційної торгівлі. А компанії, які працюють за принципом прямих поставок, вкладають кошти виключно в веб-сайт і рекламу.

Західні покупці, а в слід за ними і українські, все частіше купують товари, не виходячи з дому, посилаючись на нестачу часу, невеликий асортимент товарів у традиційних магазинах і високі ціни. В Україні в початку 2000-х тільки ентузіасти зробили електронні покупки, сьогодні, за даними статистики, хоча б раз це зробили 4 з 5 чоловік. Розвиток бізнесу в інтернеті на сьогоднішній день дуже перспективний напрямок для підприємців, що і обумовлює актуальність даної роботи.

Розробка веб-сайтів електронної комерції – це нелегке завдання. У нього є свої проблеми. Оскільки велика кількість користувачів відвідують сайт електронної комерції щодня, тому потрібно його підтримувати на постійній основі, а саме проводити постійну інтеграційну розробку, удосконалювати певні модулі, які були реалізовані раніше і також оптимізувати навантаження користувачів на орендовану веб-інфраструктуру. Для цього потрібно залучати експертів веб-розробників від певних компаній, що займаються впровадженням та інтеграцією сучасних веб-підходів та рішень для електронної комерції.

Ціль випускної роботи полягає в моделюванні та створенні інтернет-магазину, визначення цілей та причин створення власного веб-ресурсу та розгляду програмних рішень та приймів, які були застосовані для реалізації у даній роботі.

Мета даної роботи досягається вирішенням наступних завдань:

- виявляти генезис електронної комерції, її основні передумови та тенденції розвитку;
- охарактеризувати загальні принципи побудови інтернет-магазинів;
- розробка концепції інтернет-магазину;
- моделювання архітектури веб-додатку;
- реалізація передавання даних геолокації;
- вибір інструментів для розробки інтернет магазину;
- інтерактивна взаємодія з користувачем у режимі реального часу.

РОЗДІЛ 1.
ЗАГАЛЬНІ ВІДОМОСТІ ПРО ВЕБ-РОЗРОБКУ, ПЕРЕВАГИ ТА ПРИЧИНИ КОМЕРЦІАЛІЗАЦІЇ ІНТЕРНЕТУ

1.1 Поняття веб-розробки та її основні етапи

WEB-розробка – це процедура створення WEB-додатку або WEB-сайту. Основними етапами цього процесу є такі заходи, як WEB-дизайн, верстка сторінок сайту, WEB-програмування на стороні сервера і клієнта, а також роботи по конфігурації WEB-сервера[1].

Основні етапи розробки WEB-сайту[6]:

- проектування WEB-додатку або самого сайту, а саме збір і подальший аналіз всіх вимог, вироблення технічного завдання, складання проекту інтерфейсів;
- вироблення концепції сайту з урахуванням креативу;
- розробка дизайнерської концепції інтернет ресурсу;
- розробка макетів сторінок сайту;
- створення і виконання FLASH-елементів та мультимедіа;
- верстання шаблонів і сторінок;
- роботи з програмним забезпеченням, як-то створення функціональних інструментів, або ж інтеграція в уже існуючу систему управління контентом - CMS;
- розміщення на сайті і оптимізація його текстових матеріалів;
- тестування сайту і внесення при необхідності коректувань;

- запуск створеного проекту на громадському майданчику в мережі інтернет;
- роботи з обслуговування вже діючого порталу або його програмної частини.

Однак, в залежності від необхідного завдання, якісь з вищевказаних етапів в процесі WEB-розробки, можуть і використовуватися, або ж бути тісно взаємопов'язані між собою.

Технічне завдання (ТЗ)

Його розробку для WEB-фахівців виконує, зазвичай, менеджер всього веб-проекту. Ну, а робота з самим замовником починається з заповнення брифу, де він викладає свої бажання щодо структури сайту і його візуалізації, уточнює помилки і недоробки, в разі наявності, в минулій версії WEB-сайту, приводячи свої приклади, як у його конкурентів. На підставі брифа, менеджер створює ТЗ, враховуючи при цьому, наявні можливості дизайнерських та програмних інструментів. Сам такий етап закінчується лише після затвердження ТЗ клієнтом. Однак, слід зауважити, що всі етапи проекту WEB-сайту досить сильно залежні від безлічі різних факторів, як, наприклад, величина обсягу інтернет-порталу, його функціональність, а також завдання для яких призначений створюваний інтернет-ресурс і багато чого іншого. Але, тим не менш, є і деяка кількість етапів, які неодмінно присутні при плануванні абсолютно будь-якого майбутнього проекту.[6]

Дизайн сторінок WEB-сайту: основних і типових

Будь-яка робота по інтернет-сайту починається зі створення його дизайну, зазвичай використовуючи для цього графічний редактор. WEB-дизайнер створює, кілька таких варіантів, але в суворій відповідності з ТЗ. При цьому, окремо розробляється дизайн «Головної» сторінки сайту, і далі - дизайн інших типових сторінок, наприклад: новини, статті, про нас, каталог. Власне, сам дизайн являє собою графічний файл, як малюнок, що включає в себе більш дрібні картинки.

При цьому фахівець обов'язково враховує всі обмеження для стандарту HTML, тобто не створює дизайн, який неможливо буде згодом реалізувати стандартними HTML-засобами. Винятком є лише Flash-дизайн.

Кількість самих ескізів і порядок їх пред'явлення замовнику заздалегідь обумовлюються з менеджером усього проекту, який виконує контроль запланованих термінів. Ще, також у великих WEB-фірмах в процесі бере участь і Арт-директор, який контролює якість виконання графіки. Цей етап також, як і попередній, закінчується його затвердженням у замовника.[9]

Верстка сторінок і шаблонів в HTML

Затверджений клієнтом дизайн далі передається фахівцеві-верстальщику, який «нарізає» графічне зображення на окремі семантичні блоки, з яких пізніше буде складена HTML-сторінка. В ході такої роботи створюється програмний код, який можливо переглянути за допомогою будь-якого браузера (інтернет-оглядача). Згодом дані сторінки, будуть застосовуватися, як HTML-шаблони.[9]

Програмування

Після проведених, вище згаданих заходів, готові файли в форматі HTML передаються до WEB-програміста. Розробка програмного забезпечення інтернет-сайту цілком може виконуватися, як «з самого нуля», так і на підставі системи CMS, найчастіше так званого «CMS-движка».

У разі застосування системи управління сайтом слід зазначити, що вона сама, в якомусь сенсі слова, вже готовий сайт, що включає в себе замінні блоки. Ну, а самого програміста, в такому випадку, буде більш правильно називати «CMS-фахівцем», який повинен замінити існуючий стандартний шаблон, на новий оригінальний, розроблений на базі початкового WEB-дизайну, з урахуванням індивідуальних побажань замовника.

При розробці програмного забезпечення інтернет-сайту фахівця з CMS також встановлюються контрольні строки проведення робіт.[3]

Тестування, як заключний етап WEB-розробки інтернет-сайту

Сам такий процес цілком може містити в собі самі різні види перевірок, як-то, наприклад: зовнішній вигляд сторінки сайту зі збільшеними шрифтами, при різних розмірах вікна браузера, або через відсутність дозволу на використання Flash-плеєра у клієнтському браузері, і багато іншого. Також використовується і

призначене для користувача тестування, так зване – юзабіліті, що являє собою тестування, яке проводиться задля покращення взаємодії користувачів з програмним забезпеченням.

Виявлені помилки в роботі сайту відправляються для їх виправлення до тих самих пір, поки виконавець їх не усуне. У цьому випадку терміни роботи контролює все той же проектний менеджер. Хоча, на етапі тестування ще залучають до роботи і самого дизайнера, щоб він здійснював авторський нагляд за використовуваними графічними ресурсами.[6]

Розміщення нового порталу в Інтернет-мережі

Файли розробленого WEB-сайту поміщають на сервері провайдера, де здійснюють необхідні налаштування. Слід зазначити, що на такому етапі інтернет-сайт ще поки закритий для широкого кола користувачів.

Наповнення сайту контентом і його публікація

Новий інтернет-сайт наповнюють контентом, тобто різними текстами, зображеннями, файлами для завантажень і тому подібним. Іноді самі тексти складаються копірайт-фахівцем фірми-розробника сайту, іноді ними займається сам клієнт, наймаючи автора з бірж фрілансу. Таке питання вирішується ще на стадії складання самого ТЗ, так як, якщо контент розробляється копірайтером розробника, то це обов'язково узгоджується із замовником одночасно з іншими етапами проекту, що реалізовується.

1.2 Причини комерціалізації інтернету та основні цілі створення інтернет магазину

Електронна комерція, також відома як e-commerce або інтернет-комерція, відноситься до купівлі та продажу товарів або послуг через Інтернет, а також до передачі даних для виконання цих транзакцій.

Електронна комерція часто використовується для продажу фізичних продуктів в Інтернеті, але вона також може описувати будь-які види комерційних транзакцій, які здійснюються через глобальну мережу Інтернет.

Історія електронної комерції починається з першого в історії онлай-продажу, а саме 11 серпня 1994 року відбулася перша онлайн транзакція, в наслідок якої був проданий компакт-диск на сайті американської платформи для роздрібної торгівлі NetMarket. Це перший приклад того, як споживач купує продукт у певного бренду через всесвітню павутину або «електронну комерцію», як ми її називаємо сьогодні.

З тих пір електронна торгівля розвивалася, щоб спростити пошук і проведення транзакцій продуктів та послуг через інтернет-магазини і торгові майданчики. Незалежні фрілансери, малі підприємства і великі корпорації отримали вигоду з електронної комерції. Вона дозволяє їм продавати свої товари і послуги в таких масштабах, які були неможливі при традиційній формі роздрібної торгівлі.

Швидко розвиваються високі технології, які міцно входять в усі сфери життя і діяльності людини. Залишається все менше людей, які не знають про існування мережі Інтернет. Адже «всесвітня павутина» відкриває нам багато нових можливостей, в тому числі і в сфері ведення бізнесу. Тому в сучасному світі створення інтернет-магазинів користується великим попитом. Онлайн магазин – це площадка з незначними витратами і дуже зручний інструмент ведення бізнесу, що дозволяє вирішити багато проблем і отримати швидких результатів.

Чому інтернет-магазини настільки масові на сьогоднішній день? Адже люди так звикли до звичайних магазинів. В першу чергу, створення інтернет-магазину вигідно його власнику. З його допомогою компанія може поліпшити і закріпити свої позиції на ринку певних товарів або послуг, підвищити популярність бренду, збільшити прибуток. Покупець теж зможе оцінити всі переваги придбання товарів в онлайн режимі. Сучасний темп життя змушує нас максимально економити час на багатьох речах, в тому числі і на похід в магазини. Опиняючись перед вибором - піти в магазин або зробити покупку не виходячи з дому - потенційний покупець, звичайно, віддасть перевагу другому варіанту.[4]

Основними перевагами створення інтернет-магазину є наступними:

- Він може працювати цілодобово,без вихідних і святкових днів;

- Працюючи безперервно, він буде постійно приносити реальний прибуток;

- Власнику магазину не потрібно платити за оренду приміщення і оплачувати працю продавців (досить декількох адміністраторів). Ви будете позбавлені від постійних перевірок і візитів співробітників пожежних інспекцій і санепідемстанцій;

- Віртуальна площа інтернет-магазину не обмежена;

- З його допомогою можна збільшити клієнтську базу: ознайомитися з пропонованим асортиментом товарів і послуг зможе будь-який покупець, в якій би точці світу він не знаходився. Головне, щоб у потенційних клієнтів був доступ до інтернету;

- Вирішивши створити свій інтернет-магазин, власник заощаджує не тільки час, а й гроші. Адже будівництво звичайного магазину вимагає великих фінансових витрат, і навряд чи його можна буде побудувати за кілька днів;

- Інтернет-магазин дає можливість спілкуватися з потенційними клієнтами і відповідати на всі їхні запитання в онлайн режимі цілодобово.

На даний момент на українському ринку інтернет-торгівлі помітні п'ять ключових тенденцій, внаслідок яких розвиток та обороти на ринку електронної комерції набувають нового розмаху[16]:

- **Динамічний розвиток маркетплейсів**:

За даними дослідницької компанії TNS, в липні 2018 року в нас в країні лідерами за охопленням аудиторії в сфері e-commerce є три сайти: OLX (50,2%), Prom.ua (42,6%), Rozetka (39,4 %). Всі вони знаходяться в ТОП-10 найпопулярніших сайтів в Україні і належать до категорії маркетплейсів. Тобто, це онлайн-майданчики, які збирають, систематизують інформацію про товари і послуги компаній і надають її покупцям в зручному вигляді.[2]

- **Трансграничная торгівля:**

В Україні транскордонна торгівля один з найбільш перспективних векторів

розвитку e-commerce. Проведене GfK дослідження показало, що українці активно здійснюють покупки на Amazon, AliExpress і eBay.

- **Інтеграція офлайн і онлайн продажів:**

Більше 36% українських рітейлерів займаються впровадженням мультиканальної стратегії з онлайн і офлайн каналами продажів.

Наочним прикладом служить стратегія інтернет-магазину Rozetka по розширенню офлайн присутності. У 2008 році у Rozetka був офлайн-магазин площею 120 кв м.

- **Швидко зростають витрати на рекламу для мобільних пристроїв[5]:**

Західний ринок давно активно інвестує в мобільну рекламу. Якщо оцінювати глобальні інтернет-бюджети, то мобайл перевищив десктоп в 2016 році, і обсяг цього ринку зростає з кожним роком.

Український ринок теж слідує за мобайл-тенденцією, але з великим запізненням.

За прогнозом Всеукраїнської рекламної коаліції, в 2018 році обсяг ринку мобільної реклами складе один мільярд гривень, без урахування контекстної реклами. Для порівняння, в 2017 році на мобільну рекламу в Україні витратили 0,4 млрд грн.

При цьому на рекламу для десктопів в 2018 році витратять 2 млрд грн.

- **Впровадження нових технологій в інтернет продажу:**

Інтернет-магазини активно тестують такі перспективні технології, як VR / AR (віртуальна і доповнена реальність), чат-боти, віртуальні помічники.

РОЗДІЛ 2.
ТЕХНІЧНІ ОСНОВИ РЕАЛІЗАЦІЇ ПРОЕКТУ

2.1 Функціональне розділення компонентів веб-додатку

Web-додаток являє собою особливий тип програмного забезпечення, що побудовані на основі клієнт-серверної архітектури. Особливість даних додатків полягає в тому, що сам web-додаток знаходиться та виконується розподілено у мережі, як на стороні клієнта так і на стороні сервера. Основна робота додатку базується на отримані запитів від користувача(клієнта), їх обробці та видачі. Передача усіх запитів і отримання результату роботи web-додатка на сервері відбувається через глобальну мережу інтернет за допомогою сучасних прикладних протоколів, які формують собою основу стеку TCP/IP[17].

Відображенням даних, отриманих з серверної частини програми та відправленням даних запиту клієнта на сервер зазвичай займається спеціальне програмне забезпечення – браузер(Internet Explorer, Mozilla, Opera, Google Chrome і т.д.). Як відомо, однією із основних функцій браузера є рендеринг даних, отриманих з інтернету, у вигляді сторінки, яка описується за допомогою мови розмітки HTML та стилізується за допомогою мови стилів CSS. Додатково до базової розмітки та стилів web-додатку, браузер може завантажувати та підключати спеціальні скрипти, які можуть виконуватися виключно на стороні клієнта, якщо вони були згенеровані та видані серверною частиною додатку[8].

На стороні сервера потоки виконання серверних скриптів, запускаються за домогою передачі та делегування запиту клієнта від спеціального програмного забезпечення(Web-сервера) до серверного коду, який реалізовується додатком. Одними із найвідоміших web-серверів є Nginx та Apache, які підтримують виконання та роботу серверних мов програмування.

В результаті передачі запитів клієнта на серверну частину веб-додатку, ми отримуємо необхідність у реалізації серверних компонентів web-додатку ,функціонал яких заточений формування HTML-сторінки, яка і відправляється клієнту. Виходить, що результат роботи web-додатку ідентичний результату запиту до звичайного web-сайту, однак web-додаток генерує HTML код сторінки в залежності від запиту, який зробив користувач, а не просто передає HTML сторінку у тому вигляді, у якому вона знаходиться на стороні сервера. Тобто, можна зробити наступний висновок, що web-додаток динамічно формує відповідь за допомогою серверних програмних компонентів, а не просто повертає статичний результат[3].

В цілому узагальнити роботу клієнт-серверної архітектури можна за допомогою розбиття додатку на два головних компоненти архітектури, а саме frontend(скрипти, які передані серверним кодом та виконуються на стороні клієнта) та backend(серверний код) та детально розглянути призначення, функції та логіку додатку у тому або іншому компоненті[19].

Front-end – це створення користувальницького інтерфейсу, функціональності та інтерактивності, що виконується на стороні клієнта веб-ресурсу або програми . Щоб зрозуміти, що таке *front-end* розробка, необхідно відкрити сторінку будь-якого ресурсу та переглянути код даної сторінки в браузері. Цей код є прикладом роботи *front-end* розробки, що завантажується в браузер користувача, де відбувається його парсинг, і даний код можна побачити власними очима. Код сторінки описує все, що користувач бачить перед собою: кольори, верстку, шрифти, розташування графічних елементів тощо[20].

Сучасна *front-end* розробка тримається на трьох основних мовах: *HTML*,

CSS,

JavaScript. Дану структуру називають делегуванням відповідальності, де *HTML* відповідає за структуру, *CSS* за дизайн, а *JavaScript* за інтерактивність. Щоб веб-ресурс був більш досконалим, *front-end* розробник співпрацює з дизайнерами, програмістами та *UX*-аналітиками, для створення зручного та конкурентноспроможного продукту[28].

HTML (HyperText Markup Language) – мова розмітки документів, за допомогою якої формується структура сторінки: заголовки, абзаци, списки тощо. Структура визначає елементи, необхідні для розміщення статичного або динамічного вмісту, а також є основною платформою для веб-ресурсів. Через розмаїття пристроїв з доступом в Інтернет та інтерфейсів для взаємодії з мережею такий базовий аспект, як структура, є надзвичайно важливою частиною документа, оскільки вона повинна забезпечувати форму, порядок та гнучкість. До структури документів *HTML* ставляться високі вимоги. Всі частини документа мають бути відокремленими одна від одної, кожна з них оголошена та укладена в певні теги. Тегами в *HTML* є ключові слова (оточені кутовими дужками), що можуть доповнюватися властивостями, відомими як атрибути чи *inline*-стилями. Більшість 52 тегів *HTML* парні, всередині яких знаходиться вміст, хоча й існує невелика кількість непарних тегів.

Дуже важливою складовою в *HTML* верстці є семантика, яка відповідає за відповідність тегів до вмісту інформації, що знаходиться всередині них. Дотримуючись підходу із використанням семантики, можна гарантувати швидше завантаження сторінок та набагато якісніше ранжування пошуковими системами.[22] *CSS* (*Cascading Style Sheets*) – мова для опису і стилізації зовнішнього вигляду документа. Завдяки *CSS*-коду браузер розуміє, як саме відображати елементи. *CSS* задає кольори і параметри шрифтів, визначає, як будуть розташовуватися різні блоки веб-ресурсу тощо. *CSS* має доволі простий синтаксис, що складається із двох частин: селектора та блоку стилів, оточеного фігурними дужками. Селекторів існує надзвичайно багато, але застосовують, як

правило, лише два основних –по класу і за ідентифікатором. Один *CSS*-файл містить інформацію про стиль всього веб-ресурсу, а це означає, що корегуючи лише одне правило можна кардинально змінити зовнішній вигляд. Сторінки завантажуються набагато швидше через те, що сучасні браузери використовують кешування для стилів та здатні їх застосовувати для всіх сторінок, а не до кожної окремо[23].

JavaScript – це мова, що створювалася для того, щоб оживити веб-сторінки. Його завдання реагувати на дії користувача – обробляти кліки мишкою, переміщення курсора, натискання клавіш .Існує можливість посилання запитів на сервер та завантаження даних без перезавантаження сторінки, що дозволяє вводити повідомлення і багато іншого. На *JavaScript* програмуються сценарії, які є набором інструкцій, яким повинен слідувати комп'ютер для досягнення тієї чи іншої мети. Сценарій, по своїй суті, це така сама програма, єдиною відміністю якого є те, під керуванням якого середовища він виконуються. Сценарії на мові *JavaScript* інтерпретуються, але інтерпретація виконується не в явному вигляді, а засобами браузера[21].

JavaScript допомагає поліпшити інтерактивність веб-сторінок за допомогою наступних можливостей[25]:

– доступ до контенту (дозволяє виділити чи знайти на сторінці необхідний елемент з певним атрибутом або *id*, довідатися яку інформацію було введено в текстове поле тощо);

– зміна контенту (використовується для додавання на сторінку або для видалення з неї елементів, атрибутів і тексту);

– програмування правил (дозволяє вказати послідовність операцій, які повинен виконати браузер крок за кроком);

– реагування на події (надає можливість створити сценарій, підготовлений до запуску після конкретної події).

Back-end розробка – це набір апаратно-програмних засобів, за допомогою яких реалізується логіка роботи ресурсу. Тобто, це те, що приховано від очей

користувача і відбувається поза його браузером та комп'ютером. Яскравий приклад *back-end* розробки: коли користувач вводить запит на сторінці пошуковика та натискає клавішу *Enter*, робота *front-end* закінчується і починається *back-end*. Запит відправляється на сервер, де розташовані алгоритми пошуку. Сервер – це більш потужний комп'ютер, що виконує певні функції. Він зберігає дані та відповідає на запити користувачів. Щойно на моніторі з'являється інформація, яку шукав користувач, знову відбувається повернення в зону *front-end*. По суті, *back-end* – це процес об'єднання сервера з користувачем за допомогою баз даних, *API* та операційних систем. *Back-end* розробник може застосовувати різноманітні інструменти, що доступні на сервері. Як правило, в його розпорядженні наявні розмаїті мови програмування, зокрема *PHP*, *Java* та *Python*. Також для *back-end* розробки використовуються різні системи управління базами даних: *MySQL*, *MongoDB, Cassandra* тощо. Самою поширеною комбінацією в *back-end* розробці є поєднання *PHP* з *MySQL*. Залежно від цілей, *back-end* розробник може виконувати наступні функції: забезпечувати кібербезпеку ресурсу та даних користувачів, створювати та інтегрувати бази даних, налаштовувати технології резервного копіювання та відновлення[26].

2.2 Опис основних програмних засобів для реалізації додатку

PHP - це популярна мова програмування, особливо в середовищі веб-розробників. Автором початкової версії є Расмус Лердорф, ідея якого полягала в розробці набору інструментів для спрощення процесу створення динамічних веб-сторінок. Незважаючи на те, що сучасний PHP є мовою загального призначення, його найчастіше використовують як серверний інструмент для генерації HTML-коду, який потім інтерпретується веб-браузером. Він має простий синтаксис, частково схожий на Java і C ++. Даний проект постійно розвивається програмістами-ентузіастами зі всього світу, на даний момент актуальною є 7-я версія мови. За статистикою, кожен шостий програмний продукт створений на

PHP[11].

Для чого взагалі може знадобитися мова програмування при створенні сайту? Кому він стане в нагоді, а кому - не дуже? Давайте це вияснимо.

Починаючи писати програми для інтернету, багато початківців програмістів стикаються з такою помилкою. Вони розглядають систему браузер-сервер, як звичайну та інтерактивну програму. Натиснув кнопку - система зреагувала. Провів мишкою - зреагувала. Вся інформація, яка доступна клієнту - доступна і програмі, програма весь час знаходиться в оперативній пам'яті.

Але веб програмування це дещо інше. У момент, коли користувач бачить перед собою сторінку і починає здійснювати якісь дії з нею, PHP вже завершив роботу. І користувач взаємодіє не з PHP скриптом, а зі своєю сторінкою HTML, яку він отримав в браузері. Результатом роботи скрипта на PHP в більшості випадків є звичайний текст HTML сторінки, який згодом передається браузеру у відповідь. Сервер і браузер спілкуються, посилаючи один одному запити по особливому протоколу - HTTP. З'єднання може ініціювати тільки браузер. Він посилає серверу запит на отримання певного ресурсу. А сервер у відповідь насилає клієнту файли. Тобто коротко кажучи ,клієнт запросив ресурси – сервер їх надіслав, і відразу ж сервер забуває про запит клієнта. Звідси може виникнути цілком зрозуміле питання , чи можна точно дізнатися, скільки користувачів на даний момент перебувають на сайті, скільки часу вони взаємодіяли з сайтом тощо. Не можна. тому, що "на сайті" немає жодного користувача. Вони з'єднуються, запитують сторінку, і від'єднуються, не маючи постійного з'єднання з сервером. Дізнатися можна тільки приблизно, записуючи час кожного з'єднання і вибираючи записи за певний проміжок часу. Отже, звідси стає ясно, що сервер може дізнатися про клієнта дуже мало інформації, необхідної для повноцінного функціонування додатку, сервер володіє тільки інформацією про дані користувача в момент надсилання HTTP запиту клієнтом при цьому не запам’ятовуючи інформацію та стани запитів , а скрипт, який виконується на сервері віддає користувачеві сторінку і миттєво закінчує роботу, всі дані при цьому зникають,

але при правильній побудові веб-додатку ми можемо взаємодіяти з базою даних, яка і буде зберігати надіслані користувацькі дані через HTTP запити та фіксувати стани запитів[18].

З вище зазначеної інформації можна прийти до висновку, що основними функціями серверної мови сценаріїв є наступними[11]:

- «оживлення» HTML сторінок за допомогою динамічності. Звичайні HTML-сторінки статичні. Статичність (або незмінність) означає, що після того, як сторінку створили і завантажили на сайт, при кожному зверненні до цієї самої сторінки браузер покаже її будь-якому користувачеві в незмінному вигляді;
- Занесення даних користувачів які були надіслані за допомогою HTTP запитів до бази даних, щоб певним чином зберігати стан додатку та дані користувачів у ньому;

Сфери застосування PHP[11]:

- Написання скриптів і повноцінних веб-додатків, що виконуються на стороні сервера. Це найпопулярніша сфера застосування, оскільки мова спочатку створювалася саме для веб-розробоки. Для повноцінної роботи веб-програми, написаної на мові програмування PHP, необхідні сервер, парсер (CGI-додаток) і клієнтське ПЗ (веб-браузер), яке відображає результат виконання коду;
- Створення сценаріїв, що виконуються в консольному режимі. Такі міні-додатки можуть працювати на будь-якому ПК. Для їх виконання потрібнен лише парсер. Оскільки PHP містить потужні інструменти для роботи у консольному режимі, то такі сценарії найчастіше створюють для обробки текстових даних;
- Для написання графічних інтерфейсів. PHP має безліч відгалужень, створених для реалізації різних завдань. Одним з таких відгалужень є PHP-GTK. Його зазвичай використовують ті програмісти, які звикли до синтаксису PHP.

Ключові переваги PHP:

- Простий і інтуїтивно зрозумілий синтаксис. PHP швидко освоюють навіть програмісти-новачки. Він увібрав усі кращі особливості таких популярних мов, як C, Java і Perl. Код PHP легко читається незалежно від способу використання (для

написання невеликих скриптів або створення потужних додатків з використанням об'єктно-орієнтованого підходу до реалізації програми);

- Кросплатформеність і гнучкість. PHP сумісний з усіма популярними платформами (Linux, Windows, MacOS). Написані на ньому додатки успішно працюють на різному серверному ПЗ (IIS, Nginx, Apache і багатьох інших);
- Відмінна масштабованість. PHP дозволяє домогтися максимальної продуктивності додатків, написаних на ньому, зі зростанням кількості апаратних ресурсів, що взаємодіють з ним. Веб-додатки, розподілені на кілька серверів, здатні справлятися з істотними навантаженнями (великим трафіком);
- Вбудовуваність в HTML-документи. Звичайну HTML-сторінку можна легко модифікувати, змінювати на ній контент шляхом вставки блоків коду PHP. Вони додаються подібно HTML-тегам, не порушуючи структуру документа;
- Активний розвиток і вдосконалення. Спільнота розробників постійно працює над впровадженням додаткового функціоналу, що розширює можливості мови, спрощенням синтаксису і поліпшенням захисту від можливих атак;
- Простий пошук рішень виникаючих проблем. В інтернеті існує величезна кількість форумів, присвячених програмуванню на PHP;
- Широкі перспективи подальшого розвитку. Більшість CMS були створені на чистому PHP також створено багато фреймворків для роботи з даною мовою програмування.

MySQL – система управління базою даних.

MySQL - це система керування реляційними базами даних (СУРБД). Де «реляційні» означає, що дані зберігаються у вигляді таблиць. Можливо, одна з найстаріших СУРБД, так як була написана ще в 80-х роках на мовах C і C ++[27].

Взагалі, системи типу MySQL можуть використовуватися будь-якими сайтами або додатками для зберігання різних даних. Наприклад, соціальні мережі зберігають всю інформацію про користувачів, повідомлення, картинки, відео і тд. Звичайно, можна реалізувати і свою систему зберігання. Але який сенс винаходити велосипед, якщо вже є варіанти, які гарантують надійність, безпеку і

роботу без втрат.

Варто знати, як взаємодіяти з даними системами, як мінімум з однієї причини - ви зможете аналізувати / знаходити будь-яку інформацію, яка є в базі. Наприклад, потрібно отримати інформацію про те, яка категорія споживачів зробила найбільше онлайн покупок за останній квартал тощо.

Але повернемося до MySQL. У цієї системи відкритий код, тобто можна змінювати його або доповнювати під свої потреби. Так як можливість завантажити програмне забезпечення доступна будь-кому. Також система сумісна з безліччю платформ - MacOS, Windows, Linux, Ubuntu.

Цікавий момент: багато, в силу популярності цієї системи, використовують MySQL як ім'я загальне для поняття СУРБД. Її використовують гіганти ринку, такі як Facebook, Google, Twitter і тд. Хоча існує безліч інших систем.

Поняття MySQL і SQL не одне і теж саме. MySQL є однією з найпопулярніших торгових марок програмного забезпечення СУРБД, яка реалізує модель клієнт-сервер при запитах на зберігання даних. Відповідно виникає питання, як клієнт і сервер взаємодіють в середовищі СУРБД? Вони використовують специфічну мову структурованих запитів (SQL). Якщо ви коли-небудь стикалися з іншими іменами, в яких є SQL, такими як PostgreSQL і сервер Microsoft SQL, вони, швидше за все, є брендами, які також використовують синтаксис SQL. Програмне забезпечення СУРБД часто пишеться на інших мовах програмування, але завжди використовує SQL в якості основного мови для взаємодії з базою даних[10]

Оператори SQL можуть вказати серверу виконати певні операції[10]:

- Запит даних: запит конкретної інформації з існуючої бази даних;
- Обробка даних: додавання, видалення, зміна, сортування та інші операції для зміни даних, значень або візуальних елементів;
- Ідентифікація даних: визначення типів даних, наприклад, зміна числових даних в цілі числа. Це також включає визначення схеми або взаємозв'язку кожної таблиці в базі даних;

- Контроль доступу до даних: забезпечення методів безпеки для захисту даних, у тому числі прийняття рішення про те, хто може переглядати або використовувати будь-яку інформацію, що зберігається в базі даних.
- можливість криптографії даних;
- Робота з довгим текстом. Функції COMPRESS () і UNCOMPRESS () дозволяють зберігати в базі довгий текст без впливу на продуктивність.
- Також довгий текст впливає на вимоги до обсягу дискового простору, тому краще стискати його через COMPRESS;
- Повнотекстове індексування полів VARCHAR і TEXT. Наприклад, якщо ви зберігаєте в базі статті або анонси і хочете надати користувачеві можливість пошуку.

Переваги MySQL:

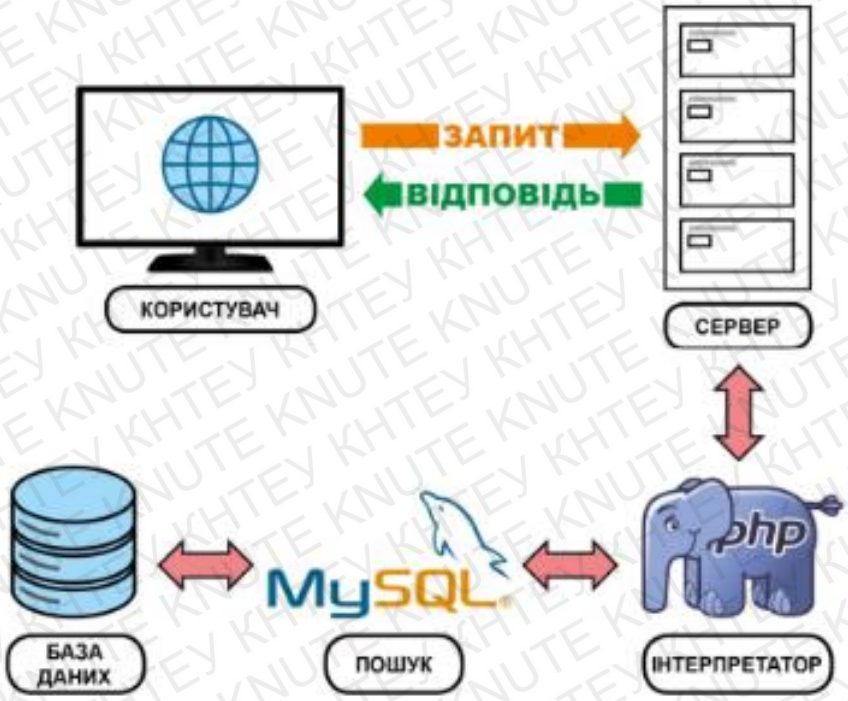
- Простота. MySQL легко встановлюється, має зрозумілий інтерфейс, а різноманітність плагінів і додаткових додатків спрощує роботу з БД;
- Функціонал. Включає в себе практично весь необхідний набір інструментів, який може стати в нагоді при розробці будь-якого проекту;
- Безпека. Багато системи безпеки вже вбудовані і працюють за замовчуванням;
- Масштабованість. Може використовуватися в роботі як з малим, так і з великим об'ємом даних;

- Швидкість. Є однією з найшвидших серед наявних на сучасному ринку.

Рис. 2.2.1 Взаємодія веб-сервера, PHP та бази даних у межах клієнт-серверної архітектури.

2.3 Змішана модель взаємодії веб-компонентів та її переваги

На сьогодні існує кілька основних архітектур, які визначають спосіб взаємодії між frontend та backend компонентами:



- **Традиційна модель (Multi Page Application)**: В цьому випадку HTTP-запити відправляються безпосередньо на серверну частину додатку, а сервер відповідає HTML-сторінкою. Між отриманням запиту і відповіддю, сервер зазвичай шукає за запитом інформацію в базі даних і вбудовує її в шаблон (ERB, Blade, EJS, Handlebars). Коли сторінка завантажена в браузері, HTML визначає, що буде показано, CSS - як це буде виглядати, а JS - всякі особливості

взаємодії[7].

- **Змішана модель**: використовує для зв'язку AJAX (Asynchronous JavaScript and XML). Це означає, що JavaScript, завантажений в браузері, відправляє HTTP-запит (XHR, XML HTTP Request) зсередини сторінки і (так склалося історично) отримує XML-відповідь. Зараз для відповідей також можна використовувати формат JSON. Це означає, що у вашого сервера повинна бути кінцева точка, яка відповідає на запити JSON або XML-кодом. Два приклади протоколів, які використовуються зазвичай у даній моделі – це REST і SOAP протоколи[13].
- **Односторінкова модель (Single Page Application):** AJAX дозволяє завантажувати дані без оновлення сторінки. Найбільше це використовується в таких фреймворків, як Angular і Ember. Після складання такі додатки відправляються в браузер, і будь-який інший рендеринг виконується на стороні клієнта (в браузері). Також веб-додатки на основі даної моделі, вимагають все меншого підключення до мережі також прогресивні веб-додатки завантажуються лише один раз і працюють (майже) завжди. Ви можете зберігати базу даних в браузері. У деяких випадках вашим додаткам потрібен бекенд тільки при першому завантаженні, а потім лише для синхронізації / захисту даних. Такий рівень сталості означає, що велика частина логіки додатка знаходиться безпосередньо в клієнті[28].
- **Ізоморфна модель:** Деякі бібліотеки і фреймворки, наприклад, React і Ember, дозволяють вам виконувати додатки як на сервері, так і в клієнті. В цьому випадку для зв'язку фронтенда з бекендом додаток також використовує і AJAX, і обробляється на сервері HTML, який вертається у відповідь клієнту. Бекенд, в свою чергу, стає легшим і легшим. Такі технології, як сховища документів і графові бази даних, призводять до скорочення кількості звернень до бекенд сторони для повторного агрегування даних. Завдання клієнта - уточнити, які дані йому потрібні (бази даних графів), або витягти всі різні фрагменти даних, які йому потрібні (REST API)[29].

Внаслідок порівняльного аналізу моделей між собою, я визначив, що використання змішаної моделі є оптимальним варіантом до проектування веб-додатків електронної комерції, по наступним причинам:

- Зменшення навантаження на сервер та збільшення швидкість відповіді від сервера: Якщо значну частину логіки додатку перенести з сервера на клієнт, то це зменшить кількість відправлених запитів користувачів на сервер. Також серверна частина додатку буде взаємодіями лише з базою даних та відправляти відповідь клієнту у JSON форматі, що само по собі являється менш затратною операцією ніж повний рендеринг HTML сторінки, як це відбувається у традиційній моделі взаємодії компонентів;
- Основна логіка безпеки та прав доступу, знаходиться на backend: даний факт зменшує кількість атак на веб-ресурс та кількість підроблених даних, які відправляються на сервер, тому що якщо основну логіку по безпеці додатку розмістити на frontend частині, доступ до якого має кожен клієнт який взаємодії з ним веб-ресурсом, то кожен зможе підробляти код, дані, логіку та системи шифрування у додатку[12];
- Оптимізації рендерингу на стороні клієнта: Якщо користувач на клієнті використовує мобільний прилад для взаємодії з веб-додатком, який має меншу кількість системних ресурсів ніж наприклад персональні комп’ютери, то і відповідно повний процес рендерингу краще розділити між сервером та клієнтом, щоб зменшити кількість навантаження додатку на роботу системи клієнта.

Схематичне зображення змішаної моделі взаємодії компонентів на рисунку

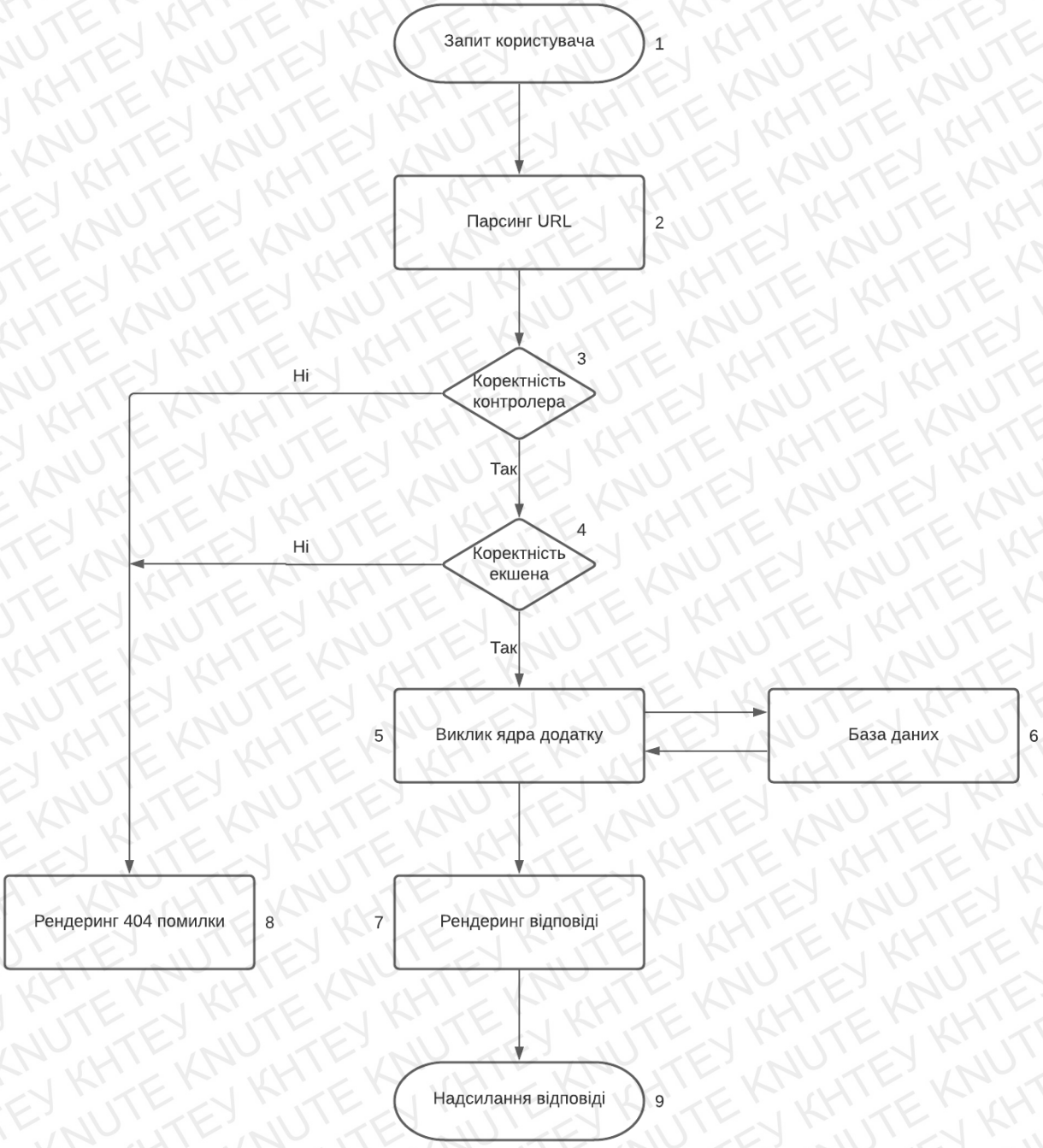
нижче.

Рис. 2.3.1 Змішана модель взаємодії веб-компонентів

РОЗДІЛ 3.
ОПИС ТЕХНІЧНОЇ РЕАЛІЗАЦІЇ ПРОЕКТУ

3.1 Модель функціонування додатку

Опис технічної реалізації проекту хотілося би розпочати з загальної композиції усього додатку, адже це дозволяє у спрощеному та схематичному



вигляді отримати початковий погляд на спосіб функціонування системи, визначення її основних складових, вразливих місць та архітектуру.

Рис. 3.1.1 Модель реалізації додатку

Етапи процесу виконання та обробки запиту клієнта на основі діаграми:

1) Запит користувача – це процес отримання даних та стану запиту клієнта з HTTP заголовків даного протоколу для подальшої їх передачі та обробки у серверних скриптах додатку.

2) Парсинг URL – це процес розбору строки вхідного запиту (URL строки) від користувача задля отримання назви контролерів та екшенів, які в свою чергу визиваються та згодом виконують та обробляють той або інший запит. Тому, задля того, аби отримати точні назви контролерів та екшенів, ми формуємо URL строку таким чином, щоб на її основі отримати всі потрібні назви програмних компонентів. Заданий підхід є гарною практикою у написанні сучасних додатків та називається процесом роутингу.

3) Коректність контролера – це процес перевірки існування файлу контролера у додатку на основі розібраної строки вхідного запиту , задля подальшого виконання та обробки відповіді клієнта.

4) Коректність екшена – це процес перевірки існування методу або функції у контролері додатку задля подальшої обробки вхідного запиту та взаємодії з базою даних проекту.

5) Виклик ядра додатку – це процес, який виконується лише при умові правильності переданих контролерів та екшенів додатку. На даному етапі відбуваються виклики екшенів контролерів та робота з вхідними параметрами, які можуть бути отримані з різних типів запитів, які надсилає клієнт, а саме: GET, POST, UPDATE, DELETE, HEAD, PUT , TRACE, PATCH. На основі переданих вхідних параметрів, ми можемо певним чином взаємодіяти з базою даних, а саме обновляти та отримувати дані з зовнішнього сховища даних.

6) База даних – це процес обробки та зберігання даних у додатку. Сховище даних допомагає всій системі – зберігати свій поточний стан роботи.

7) Рендеринг відповіді – це процес формування HTML сторінки, на основі динамічних даних, які ми отримуємо з сховища даних та певним чином оброблюємо їх серверними мовами програмування. До сформованої сторінки також динамічно добавляються стороні ресурси: каскадні таблиці стилів, шрифти, зображення, іконки та JavaScript команди.

8) Рендеринг 404 помилки – це процес формування HTML шаблону помилки запиту на веб-ресурс. Даний тип помилки дає користувачеві інформацію про те, що надісланий запит не є коректним і веб-ресурс до якого відбулося звернення – не існує.

9) Надсилання відповіді – це процес виводу зформованої HTML сторінки з буферу у вихідний потік, який згодом буде добавлений у відповідь клієнту через HTTP протокол з подальним його рендерингом у браузері клієнта.

3.2 UML діаграма бази даних

Переваги використання реляційних баз даних у проекті в порівнянні із двовимірними файлами:

- СУРБД забезпечують більш швидкий доступ до даних, чим двовимірні файли;
- СУРБД може просто відправити запит на пошук наборів даних, відібраних за певним критерієм;
- СУРБД мають убудований механізм для роботи з паралельним доступом, так що вам, як програмістові, турбуватися про нього не потрібно;
- СУРБД забезпечують довільний доступ до даних;
- СУРБД мають убудовані системи підтримки привілеїв.

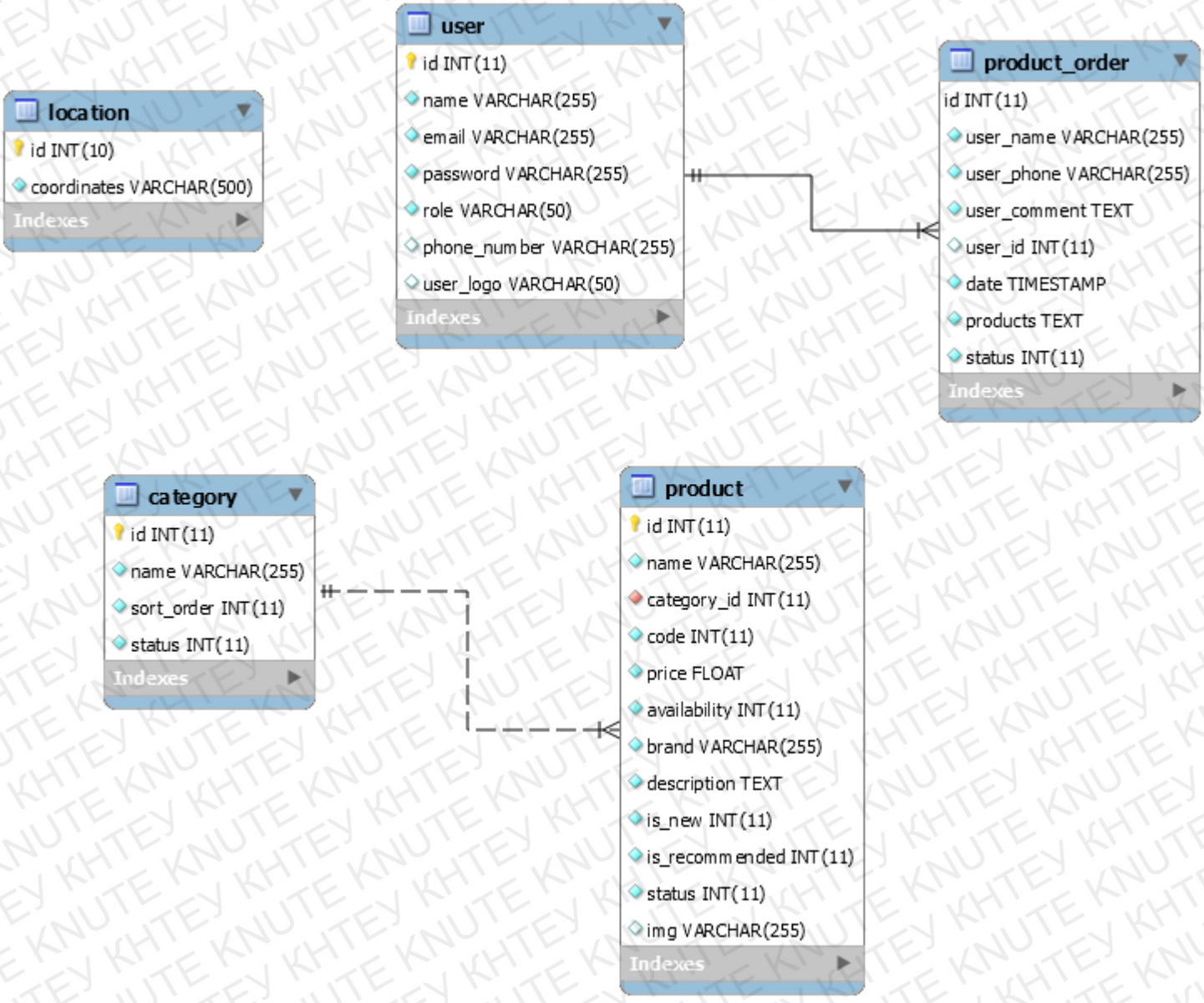
Якщо говорити більш предметно, то використання реляційної бази даних дає можливість швидко й без особливих зусиль відповісти на такі питання, як: "звідки ваші покупці?", "який тип продукції продається найбільше ефективно?" або "які покупці витрачають більше грошей?"

База даних складається з таблиць. Кожна таблиця являє собою набір упорядкованих стовпців, називаних полями (атрибутами). Кожний стовпець має унікальне ім'я усередині таблиці й має певний тип даних. Таблицю можна також представляти як набір рядків (або записів, кортежів), що складаються зі стовпців. Кожний стовпець відповідає одному запису в таблиці. У кожного рядка ті самі стовпці (атрибути). Деякі стовпці мають певну функцію - вони є ключами. За значенням у таких стовпцях можна однозначно визначити унікальність запису в таблицях. Ключ також часто має сполучну функцію, тобто забезпечує зв'язок даних із двох різних таблиць, що дозволяє співвідносити запису у двох таблицях за спеціальними правилами. Сам зв'язок, утворений ключами, називається відношенням між таблицями. Відношення може бути трьох типів: "один до

одного", "один до багатьох" і "багато до багатьох". Все разом: таблиці, правила їхніх відносин і зв'язки, - називається схемою бази даних і являє собою структуроване креслення, що візуально відображає структуру бази даних[25].

Рис. 3.2.1 UML модель бази даних проекту

UML (Unified Modeling Language) – уніфікована мова моделювання, використовується у парадигмі об'єктно-орієнтованого програмування. Є невід'ємною частиною уніфікованого процесу розробки програмного забезпечення.



UML є мовою широкого профілю, це відкритий стандарт, що використовує

графічні позначення для створення абстрактної моделі системи, яка називається UML-моделлю. UML був створений для визначення, візуалізації, проектування й документування в основному програмних систем. UML не є мовою програмування, але в засобах виконання UML-моделей як інтерпретованого коду можлива кодогенерація[25].

На основі UML діаграми проекту можна описати наступні таблиці та їх функціональне призначення:

1) Category – таблиця, яка відповідає за зберігання та обробку даних, які стосуються категорій товарів в інтернет-магазині.

Маніпуляції над таблицею у додатку: додавання, редагування та видалення категорій товарів з адміністраторської частини додатку.

Поля таблиці:

1) Id – первинний ключ таблиці, у якому міститься унікальний ідентифікатор для кожного запису категорії.

2) Name – поле, у якому зберігається текстова назва категорії.

3) Sort_order – додаткове поле, по якому можна сортувати категорії по пріоритетам сортування.

4) Status – поле, яке вказує програмному коду, чи потрібно відображати дану категорію на сайті (активний стан) чи ні(пасивний стан).

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
sort_order	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'0'
status	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'1'
		☐	☐	☐	☐	☐	☐	☐	☐	

Характеристики полей в СУРБД:

2) Product - таблиця, яка відповідає за зберігання та обробку даних наявних товарів у інтернет-магазині. Дана таблиця має зовнішній зв’язок з таблицею Category.

Маніпуляції над таблицею у додатку: додавання, редагування та видалення

товарів з адміністраторської частини додатку. Визначення статусів товарів та порядку їх відображення на веб-сторінках.

Поля таблиці:

1) *Id* - первинний ключ таблиці, у якому міститься унікальний ідентифікатор для кожного запису доданого товару.

2) *Name* – поле, у якому зберігається текстова назва товару.

3) *Category_id* – поле, у якому зберігається зовнішній ключ до категорій таблиці Category.

4) *Code* – поле, у якому міститься код артикула запису товара.

5) *Price* – поле, у якому зазначається ціна товару при добавленні його у внутршню мережу асортименту товарів інтернет магазину.

6) *Availability* – поле, у якому зберігається мітка, яка визначає наявність товару на складі.

7) Brand – поле, у якому зберігається текстове значення назви бренду запису товару.

8) Description – поле, у якому зберігається короткий опис доданого товару.

9) Is_new – поле, яке вказує користувачеві інформацію про те, чи є певний товар новинкою в асортименті інтернет-магазину чи ні.

10) Is_recommended – поле, яке вказує користувачеві про те, чи являється певний товар актуальною новинкою на даний момент в асортименті чи ні.

11) Status – поле, в якому зберігається мітка, за допомогою якої програмний код визначає чи потрібно показувати певні записи товарів на сторінках чи ні.

12) Img – поле, в якому зберігається назва зображення товару, яка згодом буде зображена на сторінках інтернет-магазину.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
category_id	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	
code	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	
price	FLOAT	☐	☑	☐	☐	☐	☐	☐	☐	
availability	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	
brand	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
description	TEXT	☐	☑	☐	☐	☐	☐	☐	☐	
is_new	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'0'
is_recommended	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'0'
status	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'1'
img	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Характеристики полей в СУРБД:

3) User – таблиця у якій зберігається та оброблюється вся інформацію про користувачів (адміністраторів,кур'єрів) та клієнтів системи. Дана таблиця має зовнішній зв'язок з таблицею product_order.

Маніпуляції над таблицею у додатку: перегляд даних зареєстрованих клієнтів на сайті, додавання кур'єрів у внутрішню систему розподілення замовлень, адміністрування всіх користувачів додатку.

Поля таблиці:

1) Id - первинний ключ таблиці, у якому міститься унікальний ідентифікатор для кожного запису доданого користувача у систему.

2) Name – поле, у якому зберігається текстова назва користувача у системі.

3) Email – поле, у якому зберігається електронна пошта зареєстрованих користувачів та кур'єрів, які були додані адміністратором через відповідну панель управління.

4) Password – поле, у якому зберігаються паролі усіх користувачів системи.

5) Role – поле, у якому до кожного запису користувача зберігається статус користувача(адміністратор, кур'єр, клієнт) та його привілегії у системі інтернет-магазину.

6) Phone_number – поле, у якому зберігається номер телефону, по якому можна зв'язатися з клієнтом або відповідним кур'єром.

7) User_logo – поле, у якому зберігаються зображення кур'єрів, які були додані у кур'єрську систему розподілення замовлень.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	☑	☑	☐	☐	☐	☐	☑	☐	
name	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
email	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
password	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
role	VARCHAR(50)	☐	☑	☐	☐	☐	☐	☐	☐	
phone_number	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
user_logo	VARCHAR(50)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Характеристики полей в СУРБД:

4) Product_order - таблиця у якій зберігається та оброблюється вся інформація про замовлення клієнтів.

Маніпуляції над таблицею у додатку: перегляд даних замовлення користувачів, зміна статусів записів замовлень, формування та виведення кошику замовлень від певного користувача.

Поля таблиці:

1) Id - первинний ключ таблиці, у якому міститься унікальний ідентифікатор для кожного запису замовлення у системі.

2) User_name – поле, у якому зберігається ім'я клієнта, який оформив покупку.

3) User_phone – поле, у якому клієнт надає свій персональний номер телефону, по якому адміністратор інтернет-магазину може зв'язатися з ним.

4) User_comment – поле, у якому зберігається побажання клієнта та його додаткові питання при оформленні замовлення.

5) User_id – це поле, у якому зберігаються зовнішні ключі користувачів, які вже зареєстровані на платформі.

6) Date – це поле у якому зберігається дата оформлення замовлення клієнтом.

7) Products – це поле, у якому зберігаються кошики замовлень клієнтів з певним асортиментом товарів.

8) Status – це поле, у якому зберігається мітка статусу замовлення(нове

замовлення, активне замовлення, виконане замовлення).

Характеристики полей в СУРБД:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(11)	☑	☑	☐	☐	☐	☐	☑	☐	
user_name	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
user_phone	VARCHAR(255)	☐	☑	☐	☐	☐	☐	☐	☐	
user_comment	TEXT	☐	☑	☐	☐	☐	☐	☐	☐	
user_id	INT(11)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
date	TIMESTAMP	☐	☑	☐	☐	☐	☐	☐	☐	CURRENT_TIMESTAMP
products	TEXT	☐	☑	☐	☐	☐	☐	☐	☐	
status	INT(11)	☐	☑	☐	☐	☐	☐	☐	☐	'1'
		☐	☐	☐	☐	☐	☐	☐	☐	

5) Location – таблиця, у якій зберігають дані кординат кур'єрів у русі задля виведення цих кординат користувачеві та формування карти місцеположення доставки замволень.

Маніпуляції над таблицею у додатку: занесення та виведення координат кур'єрів.

Поля таблиці:

1) Id – поле, унікальної пари координат кур'єра у русі.

2) Coordinates – частина запису, яка зберігаю строку координат у вигляді : "довгота, широта".

Характеристики полей в СУРБД:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT(10)	☑	☑	☐	☐	☑	☐	☑	☐	
coordinates	VARCHAR(500)	☐	☑	☐	☐	☐	☐	☐	☐	
		☐	☐	☐	☐	☐	☐	☐	☐	

3.3 MVC реалізація додатку

Статична сторінка на HTML не вміє реагувати на дії користувача. Для двосторонньої взаємодії потрібні динамічні веб-сторінки. MVC - ключ до розуміння розробки динамічних веб-додатків, тому розробнику потрібно знати дану модель та використовувати її у своїх проектах.

MVC розшифровується як модель-представлення-контролер (від англ. Model-view-controller). Це спосіб організації коду, який передбачає виділення блоків, що відповідають за вирішення різних завдань. Один блок відповідає за дані додатку, інший відповідає за зовнішній вигляд, а третій контролює роботу додатка[14]. *Компоненти MVC:*

- Модель - це компонент, який відповідає за дані, а також визначає структуру програми. Наприклад, якщо ви створюєте To-Do додаток, код компонента model визначатиме список завдань і окремі завдання;
- Представлення - це компонент, який відповідає за взаємодію з користувачем. Тобто код компонента view визначає зовнішній вигляд програми і способи його використання;
- Контролер - цей компонент відповідає за зв'язок між model і view. Код компонента controller визначає, як сайт реагує на дії користувача. По суті, це мозок MVC-додатків.

Мета шаблону – гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість
повторного використання окремих компонентів програми. Крім того використання
цього шаблону у великих системах призводить до певної впорядкованості їх структури і робить їх зрозумілішими завдяки зменшенню складності[14].

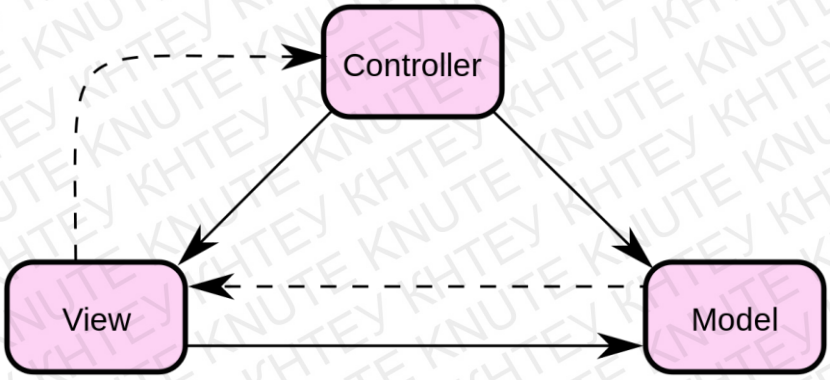


Рис 3.3.1 Схематичне зображення компонентів MVC

Опис компонентів Models в проекті:

1) Blog – це модель, яка відповідає за роботу з постами блога (взаємодією з таблицями у базі даних, які містять усі потрібні дані щодо збереження та обробки записів даних постів блогу), введенням якого займається компанія внаслідок своєї комерційності діяльності.

Методи даної моделі:

- *getBlogItemByID*: даний метод відшуковує у таблицях бази даних, які пов'язані з постами, конкретний пост по заданому ідентифікатору, який буде передаватися у додаток по строках запиту від користувача. Результат методу – це отримання та передавання даних конкретного посту у вигляди додатку;
- *getBlogList*: даний метод робить вибірку усіх постів із відповідних таблиць у базі даних та передає отриманий масив даних постів у види додатку, які рендеряться на основі цього масиву.

2) Category – це модель, яка відповідає за роботу з категоріями товарів у нашому інтернет-магазині, а саме дозволяє робити вибірку категорій, їх видалення, добавлення та редагування напряму із додатку.

Методи даної моделі:

- *getCategoriesList*: метод, який робить вибірку усіх категорій товарів у

додатку з таблиці Category бази даних та вертає результат;

- *getCategoriesListAdmin:* метод, який повертає усю доступну інформацію про властивості категорій(назву, порядок сортування, статус тощо), якою оперує адміністратор системи. Даний метод, виконується лише на стороні адміністратора;
- *getStatusText:* метод, який на основі статусу категорії товару, який повертає метод *getCategoriesListAdmin,* відображає текстову назву статусу категорії(Відображається, скрито);
- *createCategory*: метод, який дослівно з переводу означає створення та занесення даних нової категорії до бази даних. Даний метод доступний лише з адміністраторської частини додатку;
- *deleteCategoryById:* метод, який на основі переданого йому ідентифікатора категорії зі строки вхідного параметру, видаляє відповідну категорію з таблиці Category бази даних додатку;
- *getCategoryById:* метод, який повертає значення конкретної категорії товарів по переданому в роботу додатку ідентифікатору;
- *updateCategoryId* – це метод, який доступний лише з панелі адміністратора. Дозволяє оновлювати дані у таблиці Category через форму редагування категорії. Редагує наступні поля: id, name, sort_order, status.

3) Order – модель, що відповідає за роботу з замовленнями клієнтів у системі додатку.

Методи даної моделі:

- *Save:* метод, який займається добавленням замовлень у систему ордерів додатку. Для валідного добавлення замовлення, методу потрібні наступні вхідні параметри: назва клієнта, телефонний номер клієнта, можливі запитання клієнта, ідентифікатор клієнта та JSON строка кошика товарів на основі якого відбулося замовлення;
- *getOrderList:* метод, який виводить вибірку усіх замовлень, які коли-небудь відбулися у системі. Зазвичай даний метод потрібно використувати лише при

роботі з правами адміністратора у додатку. Вихідна вибірка буде містити наступні параметри кожного замовлення: дата оформлення ордеру, статус замовлення, дані клієнта по яким можна легко сконтактувати з замовником;

- *getStatusText:* метод, який на основі статусу замовлення виводить текстове повідомлення статусу замовлення;
- *getOrderById:* метод, який за допомогою переданого ідентифікатора замовлення, виводить усі дані, які стосують даного замовлення;
- *updateOrderById:* метод, який приймає вхідні параметри, на основі якого ми оновлюємо значення певного запису замовлення у базі даних. Вхідні параметри є наступними: ідентифікатор замовлення, дані клієнта, дата оформлення замовлення та статус замовлення;
- *deleteOrderById:* метод, який на основі переданого ідентифікатора замовлення, видаляє замовлення з бази даних.

4) Product – це модель, яка працює з даними товарів у таблиці Product бази даних проекту. Дана модель, найчастіше використовується у системі, тому має найбільшу кількість методів.

Методи даної моделі:

- *getLatestProduct:* повертає дані останніх(найактуальніших) товарів, які були добавленні в систему;
- *getProductsListByCategory:* метод, який по вхідному параметру назви категорії, повертає усі релевантні товари. Зазвичай використовується при фільтрації товарів на головній сторінці додатку;
- *getProductById:* метод, який по переданому вхідному ідентифікатору продукту повертає усю інформацію про нього. Зазвичай використовується у панелі адміністратора для виведення даних, які згодом будуть редагуватися;
- *getTotalProductsInCategory:* повертає кількість товарів, які перебувають під певною категорією. Також використовується для підрахування кількості елементів у корзині;
- *getRecommendedProducts:* метод, який слугує для того аби вибирати з бази

даних товари, які помічені міткою рекомендовані;

- *getProductsByIds:* метод, який повертає товари, по масиву переданих ідентифікаторів. Використовується для вибірки даних товарів на основі збереженої JSON строки замовлення*;*
- *getProductsList:* метод, який вибирає усі продукти з бази даних, які мають статус як “відображатися” та виводить їх на головній сторінці додатку*;*
- *deleteProductById:* метод по якому відбувається видалення запису певного товару з бази даних;
- *createProduct:* метод добавлення товару через адмінпанель*;*
- *updateProductById:* метод, який на основі вхідних параметрів товару, обновляє відповідний запис у базі даних.

5) User – це модель, яка відповідає за дані користувачів(клієнтів, кур’єрів) у додатку.

Методи даної моделі:

- *Register:*метод, який реєструє та зберігає вхідні дані клієнта у системі додатку.
- *Edit:* метод, за допомогою якого користувач може редагувати облікові дані власного аккаунту.
- *checkName:* метод, який валідує правильність імені користувача при реєстрації.
- *checkPassword:* метод, який валідує коректність заданого паролю користувачем при реєстрації.
- *checkEmail*: метод, який валідує переданий адрес електронної почти.
- *checkEmailExists:* метод, який перевіряє, чи не був користувач зареєстрований раніше.
- *Auth:* метод, який стартує сесію клієнта після реєстрації.
- *checkLogged:* метод, який перевіряє активність сесії користувача у системі.
- *getUserById:* метод, який виводить дані клієнта або кур’єра у адмінпанелі.
- *isGuest:* метод, який перевіряє привілегії користувача додатку.

- *checkPhone:* метод, який валідує коректність номера телефонна користувача при процесі реєстрації.
- *getAllSuppliers:* метод, який повертає повний список усіх кур'єрів системи доставки.
- *createUser:* метод, за допомого якого ми створюємо користувача системи.
- *deleteUserById:* метод видалення користувачів системи.
- *updateUserById:* метод оновлення даних користувачів.

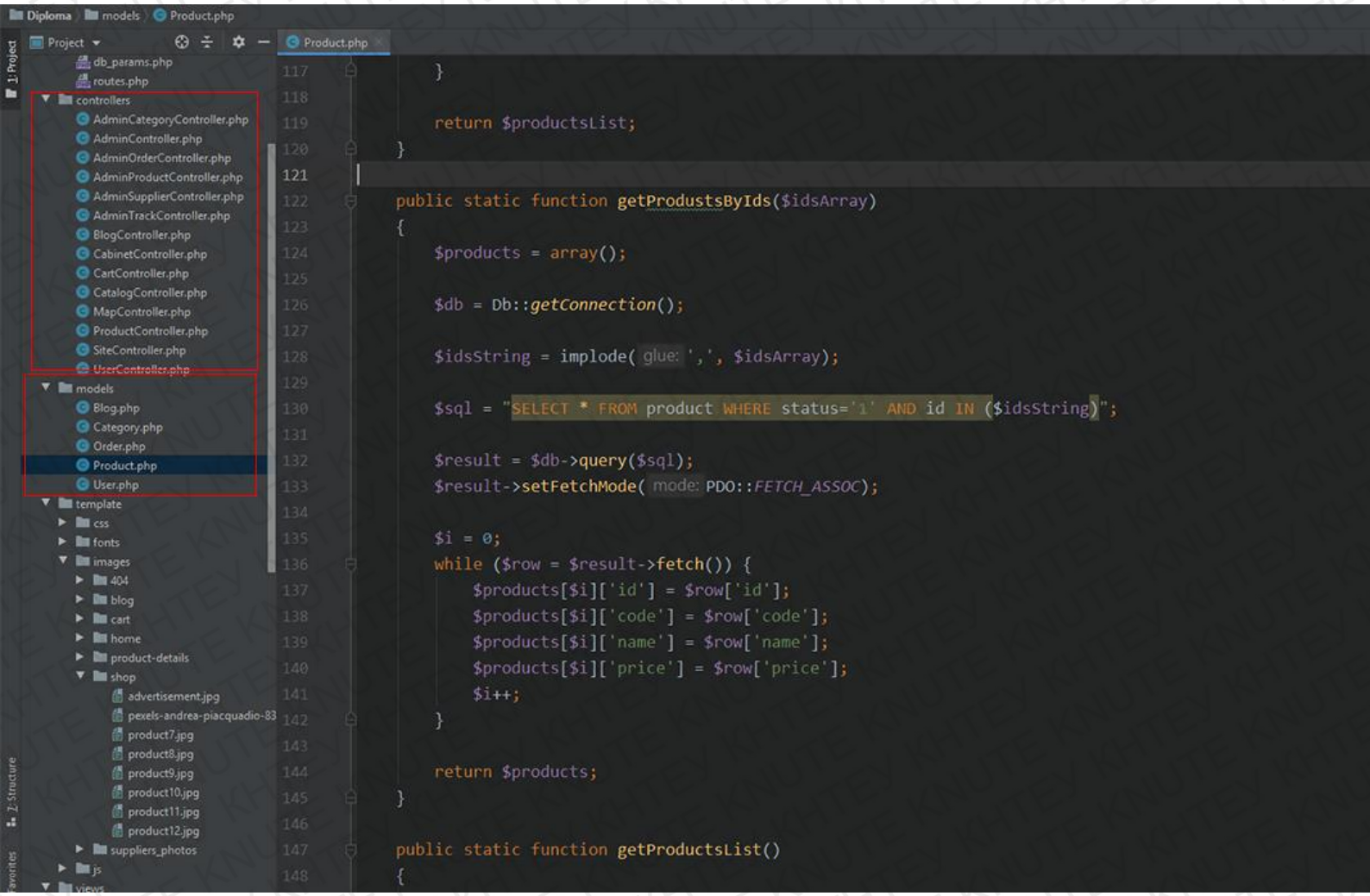


Рис. 3.3.2 Приклад методу моделі Product та розташування компонентів

3.4 Архітектура роботи геомодуля

Геолокація - це ідентифікація або оцінка реального географічного місця розташування об'єкта, такого як радарне джерело, мобільний телефон, базова станція або будь-який девайс підключений до Інтернету. У своїй простій формі геолокація включає генерацію набору географічних координат і тісно пов'язана з використанням систем позиціонування, але її корисність підвищується завдяки використанню цих координат для визначення значимого місця розташування, такого як адреса вулиці, поштового індексу[30].

IP Geolocation API - це невеликий фрагмент коду (API), який допомагає визначити приблизне фізичне місце розташування користувача за допомогою його віртуального адреси (IP). IP Geolocation - це метод визначення приблизного фізичного місця розташування підключеного до Інтернету пристрою через адреса, а IP Geolocation API - це доступність цього методу у вигляді API[30].

Причини використання геолокації у веб-додатках:

- збільшення інтерактивності з користувачем;
- вивід асортименту товарів під місцевість у якій проживає користувач;
- прискорення доставки замовлення та збільшення виконання замовлень на одиницю часу в цілому за рахунок вибору найближчого кур’єра до певного користувача;
- перевага над конкурентами, які не реалізовували модулі геолокації на своїх онлайн площадках;
- зручність адміністративного контролю над кур’єрами інтернет магазину.

Етапи роботи модуля геолокації:

1) При русі гаджета кур’єра у просторі відбувається запит до API геолокації. Даний запит ініціює ідентифікацію географічного місцеположення приладу.

2) Відповідь від стороннього API геолокації у якій додаток отримує точні дані місцеположення кур’єра системи задля подальшого їх збереження у сховища даних додатку. Дані геолокації складають з двох значень(довгота та широта).

3) Процес асинхронного збереження даних геолокації певної поточної кординати простору у базу даних.

4) Можливість зворотнього отримання останньої кординати геопозиції для адміністративної частини кур’єра задля можливості подальшого створення рендерингу карти не лише на стороні клієнта, а і на стороні кур’єрів системи.

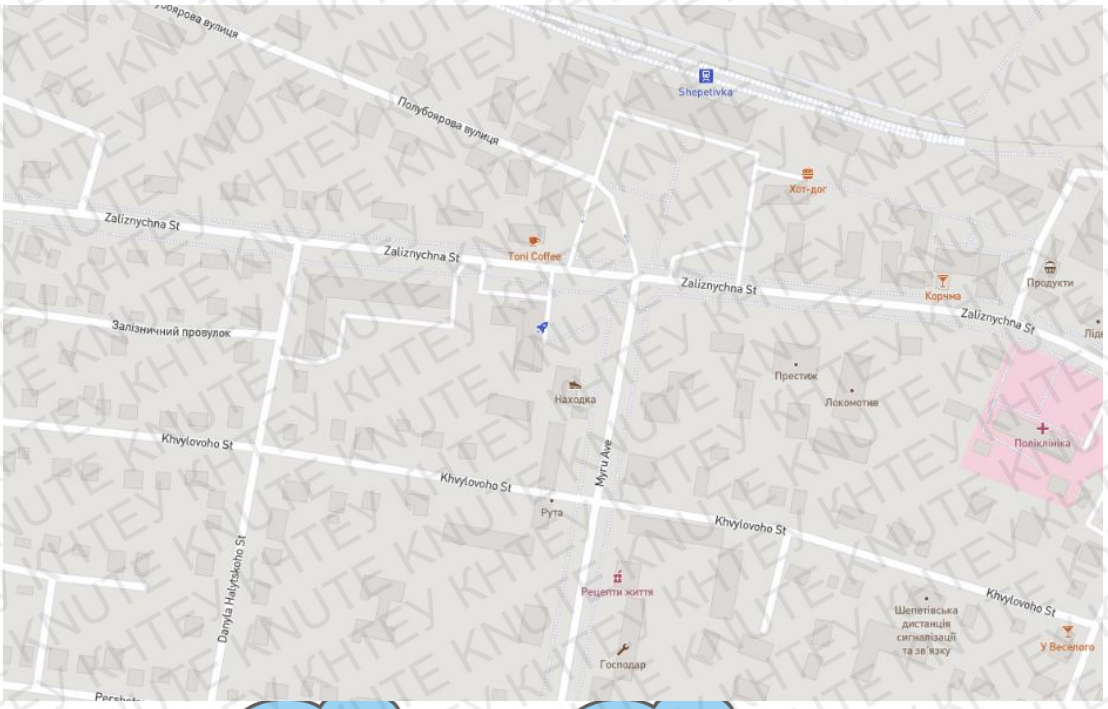
5) Асинхронний запит клієнта до бази даних для отримання даних місцеположення кур’єра у режимі реального часу.

6) Асихронна відповідь від додатку. У якій клієнту надсилається JSON строка представлення геоданих, які були отриманні з бази даних.

7) Обробка JSON строки представлення даних на стороні клієнта та подальша їх передача у стороннє API для рендерингу мапи поточного місцеположення кур’єра на основі отриманих даних.

8) Процес рендерингу мапи у браузері клієнта та постійне асинхронне оновлення стану координат у режимі реального часу.

Рис. 3.4.1 Схематичне зображення роботи модуля геолокації



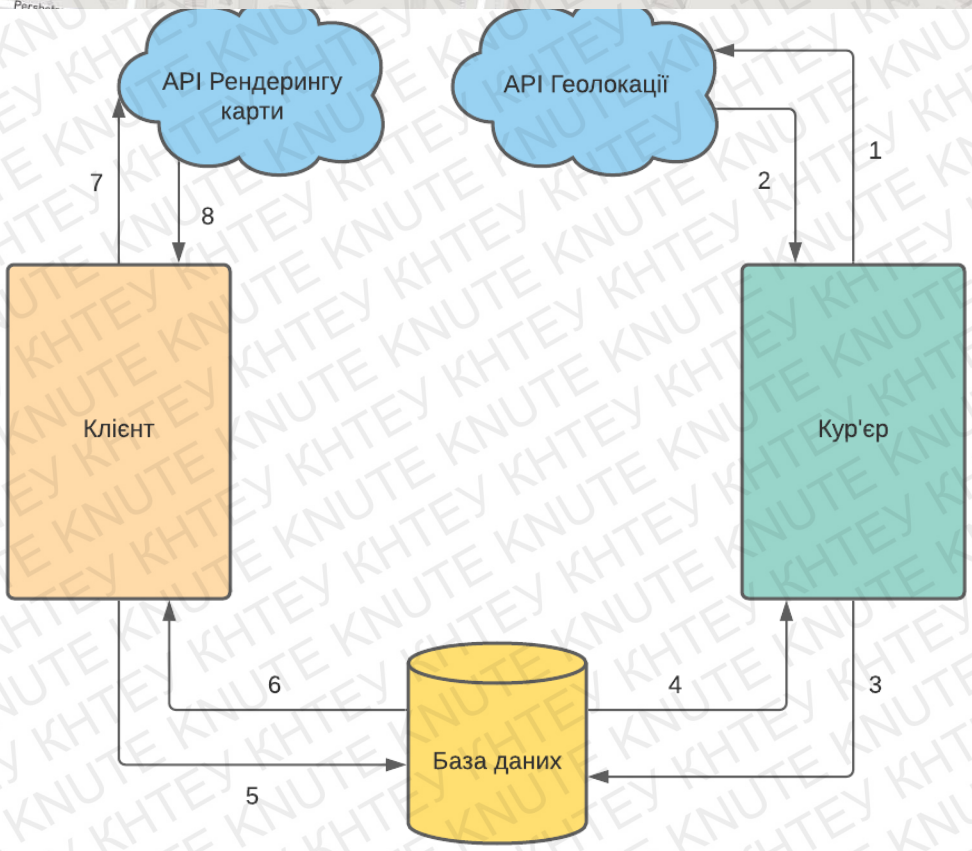


Рис. 3.4.2 Кінцевий результат роботи модуля геолокації

ВИСНОВКИ ТА РЕЗУЛЬТАТИ

Під час виконання випускної кваліфікаційної роботи було створено дизайн та розроблено комерційний інтернет-магазин по торгівлі побутової технікою з інтерактивним модулем доставки товарів.

Проаналізовано сучасні технології у створенні веб-сайтів, та їх детальний аналіз на основі переваг, функціоналу, способів застосування та можливості використання на різноманітних платформах. Використаноз клієнт-серверну архітектуру функціонування веб-додатків, яка має відмінності від функціонування інших видів додатків і відповідно вимагає зовсім інших підходів до архітектурних програмних рішень на основі розподілення веб-додатку на два головних та незалежних компоненти, а саме frontend та backend.

Детально порівняно наявні моделі взаємодії компонентів веб-систем та

вибрано оптимальний варіант для основи взаємодії компонентів даної проектної роботи. Також було протестовано веб-сайт у найпопулярніших браузерах. В ході тестування виявлено, що сайт працює коректно. Для готового програмного продукту було проведене повноцінне тестування: перевірка гіперпосилань, перевірка відповідності текстів заданій темі, перевірка відповідності сайту індивідуальному завданню, перевірка на кросбраузерність та юзабіліті.

Розглянуто технічні аспекти проекту у останньому розділі, який описує технічні реалізації проекту, а саме:

- Побудувано загальну модель функціонування проекту;
- Розглянуто UML діаграму бази даних проекту з детальним оглядом та описом таблиць бази даних та їх характеристик;
- Визначено сутність MVC архітектури додатку, розглянуто та проаналізувано наявні програмні компоненти даної архітектури на основі реалізованого проекту;
- Досліджено та описано усі наявні етапи у роботі модуля геолокації та визначено його архітектуру.

Отже, в ході виконання даної випускної кваліфікаційної роботи було продемонстровано практичні навички створення та розробки веб-додатку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ташков П.А. Веб-мастеринг. HTML, CSS, JavaScript, PHP, CMS, AJAX, раскрутка - Издательский дом "Питер", 15 дек. 2008 г. - 514 с.
2. Дарія Жиленко. Стаття рейтинг популярних сайтів за липень 2018. [Електронний ресурс] - https://tns-ua.com/news/reyting-populyarnih-saytiv-za-lipen-2018
3. Робин Никсон – Пошаговое руководство по созданию динамических веб-сайтов – Издательский дом "Питер", 2014 г. – 688 с.
4. Маркетингове агенство GFK. 17% українських онлайн-покупців здійснюють більше 20 покупок на рік. [Електронний ресурс] - https://www.gfk.com/ru/insights/online-shopping-2019
5. І.І. Новаківський, О.І. Дума. МОБІЛЬНА РЕКЛАМА ТА ПЕРСПЕКТИВИ ЇЇ РОЗВИТКУ. [Електронний ресурс] - http://ena.lp.edu.ua:8080/bitstream/ntb/7580/1/29.pdf

6. Поліна Приходько. Етапи розробки сайту. [Електронний ресурс] - https://wezom.com.ua/blog/etapy-razrabotki-sajta
7. Барри Поллард. HTTP-2 в действии, - Издательство: Manning Publications, 2019 г. – 410 с.
8. А.В. Кириченко. Javascript для FrontEnd-разработчиков. Написание. Тестировние. Развертывание - Издательство: Наука и Техника, 2020 – 322 с.
9. Алан Мурадов. Основы разработки веб-приложений и сайтов. Путеводитель по веб-технологиям - Издательство ЛитРес, 2020 г.
10. Б.А. Новиков, Е.А. Горшкова, Н. Графеева. Основы технологий баз данных второго изд – Издательство “ДМК Пресс”, 2020 г. – 583 с.
11. Максим Кузнецов, Игорь Симдянов. Самоучитель PHP 7, Максим Кузнецов, Игорь Симдянов 2018 – Издательство “БХВ-Петербург”, 2018 г. – 450 с.
12. М.В. Кузнецов, И.В. Симдянов. Головоломки на PHP для хакера – Издательство “БХВ-Петербург”, 2006 г. – 464 с.
13. Кристиан Дари, Богдан Бринзаре. Ajax и PHP – Издательство “Символ”, 2007 г. – 334 с.
14. Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Паттерны объектно-ориентированного проектирования – Издательство “Питер”, 2020 г. – 448 с.
15. Мельник О. Выбираем профессию frontend- и backend-разработчика: принципы и отличия / О. Мельник [Електронний ресурс] - https://skillbox.ru/media/code/frontend_i_backend_razrabotka/
16. Лариса Костянтинівна Гліненко. СТАН І ПЕРСПЕКТИВИ РОЗВИТКУ ЕЛЕКТРОННОЇ ТОРГІВЛІ УКРАЇНИ. [Електронний ресурс] - https://mmi.fem.sumdu.edu.ua/sites/default/files/mmi2018_1_83_102.pdf
17. Исаченко И.В. Программное обеспечение компьютерных сетей - Издательство “Инфра-М”, 2021 г. – 158 с.
18. Курс лекций "Тестирование програмного обеспечения". Архитектура

клиент-сервер. [Електронний ресурс] - https://sergeygavaga.gitbooks.io/kurs-lektsii-testirovanie-programnogo-obespecheni/content/lektsiya-6-ch1-arhitektura-klient-server.html

19. Андрей Павленко. Что такое backend и frontend? [Електронний ресурс] - https://otus.ru/nest/post/943/
20. Васильев А.Н. JavaScript в примерах и задачах / А.Н. Васильев. – Москва: Издательство „Э”, 2017 г. – 720 с.
21. Дакетт Д. JavaScripte и jQuery. Интерактивная веб-разработка / Д. Дакетт. – Москва: Издательство „Э”, 2017 г. – 640 с.
22. Мак-Дональд М. HTML5. Недостающее руководство / М. Мак-Дональд. – Санкт-Петербург: БХВ-Петербург, 2012 г. – 480 с.
23. Макфарланд Д. Новая большая книга CSS / Д. Макфарланд. – СанктПетербург: Питер, 2016 г. – 720 с.
24. Wodehouse C. A Beginner's Guide to Back-End Development / C. Wodehouse [Electronic resource]. – https://www.upwork.com/hiring/development/beginners-guide-to-back-end-development
25. Sitbon B.J. Step-By-Step Guide: How to Make a Professional Website in 2019 / B.J. Sitbon [Electronic resource]. – https://www.wix.com/blog/2019/03/how-to-make-website-guide
26. Lindley C. Front-end Developer Handbook 2019 / C. Lindley. [Electronic resource]. – https://frontendmasters.com/books/front-end-handbook/2019/
27. Дмитрий Сокол. Базы данных. [Електронний ресурс] - ru.hostings.info/schools/bazy-dannyh.html
28. Ян Мельник. SINGLE PAGE APPLICATION (SPA) И MULTI PAGE APPLICATION (MPA): ПРЕИМУЩЕСТВА И НЕДОСТАТКИ. [Електронний ресурс] - https://merehead.com/ru/blog/single-page-application-vs-multi-page-application/
29. Серверный или клиентских рендеринг на вебе: что лучше использовать у себя в проекте и почему. [Електронний ресурс] -

https://tproger.ru/translations/rendering-on-the-web/

30. Энтони Т. Хольденер. HTML5 Geolocation – O'REILLY, 2011 г. – 116с.