

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЗАПОРІЗЬКА ПОЛІТЕХНІКА»

Факультет комп'ютерних наук та технологій
Кафедра комп'ютерних систем та мереж

Пояснювальна записка

до дипломного проєкту (роботи)

магістра

(ступінь вищої освіти)

на тему СИСТЕМА АДМІНІСТРАТИВНОГО КЕРУВАННЯ
ПОШТОВИМИ ПОСИЛКАМИ

Виконав: студент 2 курсу, групи КНТ-622м
спеціальності _____

123 Комп'ютерна інженерія

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Спеціалізовані комп'ютерні системи

ТУР'ЯНСЬКИЙ Є. Д.

(ПРИЗВИЩЕ та ініціали)

Керівник ТЯГУНОВА М.Ю.

(ПРИЗВИЩЕ та ініціали)

Рецензент КАПЛІЄНКО Т.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

ЗАТВЕРДЖУЮ

Зав. кафедри Кудерметов Р.К.

“ ” 2023 року

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

ТУР'ЯНСЬКОГО Євгенія Дмитровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Система адміністративного керування поштовими посилками

керівник проєкту (роботи) к. т. н., доцент, ТЯГУНОВА Марія Юріївна

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

затверджені наказом вищого навчального закладу від “24” жовтня 2023 року №400

2. Строк подання студентом проєкту (роботи) 10 грудня 2023 року

3. Вихідні дані до проєкту (роботи) створення поштового сервісу, супутнього функціоналу та адміністративної панелі

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1) Аналіз предметної області;

2) Аналіз та вибір засобів розробки;

3) Проєктування сайту;

4) Розробка сайту

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

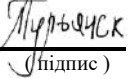
6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4	ТЯГУНОВА М. Ю., доцент		
нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання 01.10.2023 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Аналіз предметної області, аналогів у сфері та необхідного до реалізації функціоналу	10.10.2023 р.	
2	Визначення вимог до створення системи	12.10.2023 р.	
3	Визначення функціоналу до розробки	13.10.2023 р.	
4	Аналіз та вибір засобів розробки	15.10.2023 р.	
5	Розробка сайту	17.10.2023 р.	
6	Аналіз отриманих результатів	01.11.2023 р.	
7	Оформлення результатів у ПЗ	10.11.2023 р.	

Студент  **Євгеній ТУР'ЯНСЬКИЙ**
(підпис) (ім'я, ПРИЗВИЩЕ)

Керівник проєкту (роботи)  **Марія ТЯГУНОВА**
(підпис) (ім'я, ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 60 с., 21 рис., 14 джерел.

APACHE, JQUERY, MYSQL, PHP, ВЕБРОЗРОБКА

Об'єкт дослідження – системи доставки посилкок.

Предмет дослідження – розподілений вебдодаток на базі вебсерверу Apache та база даних.

Мета роботи – розробити систему керування посилками через вебсайт, який пропонує зручний доступ до широкого спектру поштових операцій.

У дипломній роботі було розроблено сайт-запрошення на захід та методи для роботи з СУБД. Використання PHPStorm дозволяє підвищити як швидкість процесу розробки вебдодатків, так і проведення процесу дебагу.

У першій частині проаналізовано предметну область, найпоширеніші існуючі вебсайти слуб доставки товарів в Україні і зроблено висновок щодо їх переваг та недоліків.

У другій частині проведено ґрунтовний аналіз і вибір засобів розробки системи. Проведено практичне дослідження та надано його результати щодо вибору мови створення сайту PHP 8.0.

У третій частині описано проєктування сайта, створення дизайну сайта та проєктування його бази даних.

У четвертому розділі наведено реалізацію сайта та проаналізовано отримані результати.

ABSTRACT

Pages: 60, 21 pictures, 14 sources.

APACHE, JQUERY, MYSQL, PHP, БЕБПОЗРОБКА

The object of research is parcel delivery system.

The subject of the study is a distributed web application based on the Apache web server and a database.

The goal of the work is to develop a parcel management system through a website that offers convenient access to a wide range of postal operations.

In the diploma work, a site-invitation to the event and methods for working with DBMS were developed. Using PHPStorm allows you to increase both the speed of the web application development process and the debugging process.

In the first part, the subject area, the most common existing websites of goods delivery services in Ukraine were analyzed and a conclusion was drawn regarding their advantages and disadvantages.

In the second part, a thorough analysis and selection of system development tools was carried out. A practical study was conducted and its results were provided regarding the choice of PHP 8.0 website creation language.

The third part describes the design of the site, the creation of the site design and the design of its database.

The fourth chapter describes the implementation of the site and analyzes the obtained results.

ЗМІСТ

Вступ.....	7
1 Аналіз предметної області.....	9
1.1 Сайт Нової пошти.....	10
1.2 Компанія Meest Express	13
1.3 Сайт Укрпошти.....	15
1.4 Вибір типу сайту	18
1.5 Основні вимоги.....	21
1.6 Структура сайту.....	22
2 Аналіз і вибір засобів розробки	23
2.1 Клієнтська частина сайту	23
2.2 Серверна частина сайту	24
2.3 База даних	26
2.4 Вибір локального серверу	27
2.5 Редактор коду та локальний сервер.....	28
3 Проєктування сайту	39
3.1 Створення дизайну сайту	39
3.2 Проєктування бази даних	40
3.3 Створення проєкту	41
4 Розробка сайту	43
4.1 Front-end та JS розробка	43
4.2 PHP розробка	46
4.3 Результати створеної системи.....	56
Висновки	59
Перелік джерел посилання	60

ВСТУП

У зв'язку з прогресом інтернет-технологій, програмісти часто зустрічаються з необхідністю адаптації продуктів для різних платформ та розробки додаткових інтернет-клієнтів. Згідно з прогнозами аналітиків, у 2021 році близько 80% усього інтернет-трафіку припадатиме на мобільні пристрої, що зумовило зростання попиту на мобільні версії вебдодатків і сервісів.

Сьогодні вебсайт може слугувати діловою карткою будь-якої організації, компанії чи особистого портфоліо. Існування файлообмінників, інтернет-магазинів, онлайн-платформ і соціальних мереж підтверджує це. Володіючи власним сайтом, можна вільно управляти інформацією, актуалізувати її за потребою. Таким чином, відвідувачі вашого сайту завжди будуть мати доступ до оновленої інформації.

Вебсайт дозволяє передавати необмежену кількість даних, що є неможливим для традиційних ЗМІ через обмеження у кількості символів та доступному просторі для друку, а радіорекламу легко можна пропустити або неправильно зрозуміти. Вебсторінка є доступною цілодобово, сім днів на тиждень, дозволяючи користувачам отримати доступ у будь-який зручний час. Основна мета вебсайту – передача інформації як потенційним клієнтам, так і загальній аудиторії.

Із зростанням кількості людей, які використовують інтернет для роботи, спілкування та отримання новин, традиційні засоби комунікації як телефон, газети та телебачення втрачають свою актуальність. Більшість інформації люди шукають онлайн, тому наявність власного сайту робить компанію більш доступною для споживачів.

Служба доставки - це підприємство або організація, яка спеціалізується на транспортуванні та постачанні товарів або послуг від одного місця до іншого. Такі служби займаються доставкою різних видів товарів, включаючи продукти харчування, товари для дому, медичні препарати, електроніку та багато іншого.

Служба доставки повинна мати ефективну систему транспорту для переміщення товарів від точки відправлення до пункту призначення. Логістичне планування грає важливу роль у визначенні оптимального маршруту та способу транспортування. Ефективне управління маршрутами дозволяє зменшити час та витрати на доставку. Використання спеціальних алгоритмів та технологій дозволяє оптимізувати маршрути доставки.

Сучасні служби доставки активно використовують інформаційні технології для відстеження та управління доставкою. Споживачі можуть отримати інформацію про місцезнаходження своїх товарів в режимі реального часу. Вони мають службу підтримки або онлайн-системи, які надають інформацію та вирішують проблеми клієнтів щодо доставки.

Служби доставки грають ключову роль у сучасному бізнесі та споживацькому середовищі, дозволяючи ефективно та швидко переміщати товари з одного місця до іншого. А реалізація роботи таких служб за допомогою вебсайта є максимально ефективною. Таким чином, обрана тема магістерської роботи є актуальною.

Мета роботи – розробити систему керування посилками через вебсайт, який пропонує зручний доступ до широкого спектру поштових операцій.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- проаналізувати предметну область та існуючі системи доставки;
- визначити вимоги до створюваної системи;
- спроектувати систему;
- провести дослідження мови програмування для створення системи;
- реалізувати систему з використанням обраних засобів розробки;
- проаналізувати отримані результати.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Вебсайт або просто сайт – це комплекс взаємопов'язаних вебсторінок або іноді одна окрема сторінка. Ці сторінки доступні онлайн через HTTP чи HTTPS протоколи. Кожен вебсайт займає унікальне місце в інтернеті, яке ідентифікується унікальною адресою, званою URL. Власником вебсайту може бути фізична або юридична особа.

Сайт складається з сторінок, об'єднаних спільною базовою URL-адресою, логічною структурою, оформленням (дизайном) та/або технічною структурою. У свою чергу, Всесвітня мережа – це велика кількість вебсайтів. Вебсторінки створюються за допомогою HTML або XHTML.

Коли користувач переглядає вебсторінку в браузері, HTML файли завантажуються на його комп'ютер та відображаються на екрані. Для доступу до інтернету та перегляду вебсторінок можна використовувати різноманітні пристрої, такі як комп'ютери, кишенькові ПК, деякі ігрові консолі та інші пристрої, що підтримують HTML/XHTML.

На вебсторінках може використовуватись JavaScript та інші скриптові мови, що підтримуються браузером, для надання інтерактивності. Для навігації між сторінками сайту та зв'язку з іншими сайтами у всесвітній мережі використовуються гіпертекстові посилання. Посилання, що ведуть на сторінки в межах одного сайту, називаються внутрішніми, а ті, що з'єднують різні сайти – зовнішніми.

Перед тим як розроблювати сайт було проаналізовано існуючі аналоги: сайт Нової Пошти, Укрпошти і Міст Експрес.

1.1 Сайт Нової пошти

"Нова Пошта" - це найбільша та одна з найпопулярніших логістичних компаній в Україні, яка спеціалізується на доставці посилок та інших товарів. Основна діяльність компанії пов'язана із забезпеченням послуг кур'єрської доставки, перевезення вантажів, а також надання ряду додаткових послуг, пов'язаних із логістикою та експрес-доставкою.

"Нова Пошта" має широку мережу відділень та терміналів по всій Україні. Клієнти можуть використовувати ці точки для відправлення та отримання посилок.

Компанія активно використовує інформаційні технології для полегшення взаємодії з клієнтами. Є онлайн-сервіси для відстеження посилок, розрахунку вартості доставки, та інші. Крім звичайних посилок, "Нова Пошта" надає послуги з доставки важких та негабаритних вантажів, а також спеціалізовані послуги для бізнес-клієнтів.

Компанія постійно впроваджує нові технології та методи логістики для покращення ефективності своєї діяльності.

Нова Пошта" часто співпрацює з інтернет-магазинами для забезпечення доставки товарів їхнім клієнтам. Ця компанія відіграє важливу роль у логістичній інфраструктурі України, забезпечуючи швидку та надійну доставку по всій країні.

Сайт компанії "Нова Пошта" надає широкий функціонал для зручності клієнтів. Основні можливості включають:

- відстеження посилок;
- розрахунок вартості доставки;
- онлайн-замовлення кур'єрської доставки;
- список відділень та терміналів;
- калькулятор вартості вантажу;
- особистий кабінет;

– інформаційний розділ.

Клієнти можуть введенням номера накладної відстежувати рух своїх посилок через мережу "Нової Пошти". Сайт надає можливість розрахувати вартість доставки для конкретної посилки або вантажу. Клієнти можуть замовити кур'єрську доставку заборони чи відправлення посилок через онлайн-форму.

Інтерактивна карта або список містить інформацію про розташування відділень та терміналів компанії. Для вантажів з великим обсягом або вагою, клієнти можуть скористатися калькулятором для швидкого розрахунку вартості.

Зареєстровані користувачі можуть створювати особисті кабінети, де вони можуть зберігати історію посилок, відстежувати статуси доставки, та здійснювати інші операції. Сайт містить розділ з корисною інформацією про нові послуги, акції та інші аспекти діяльності компанії.

Основні сторінки сайту (novaposhta.ua) представлені на рис. 1.1 - 1.3.

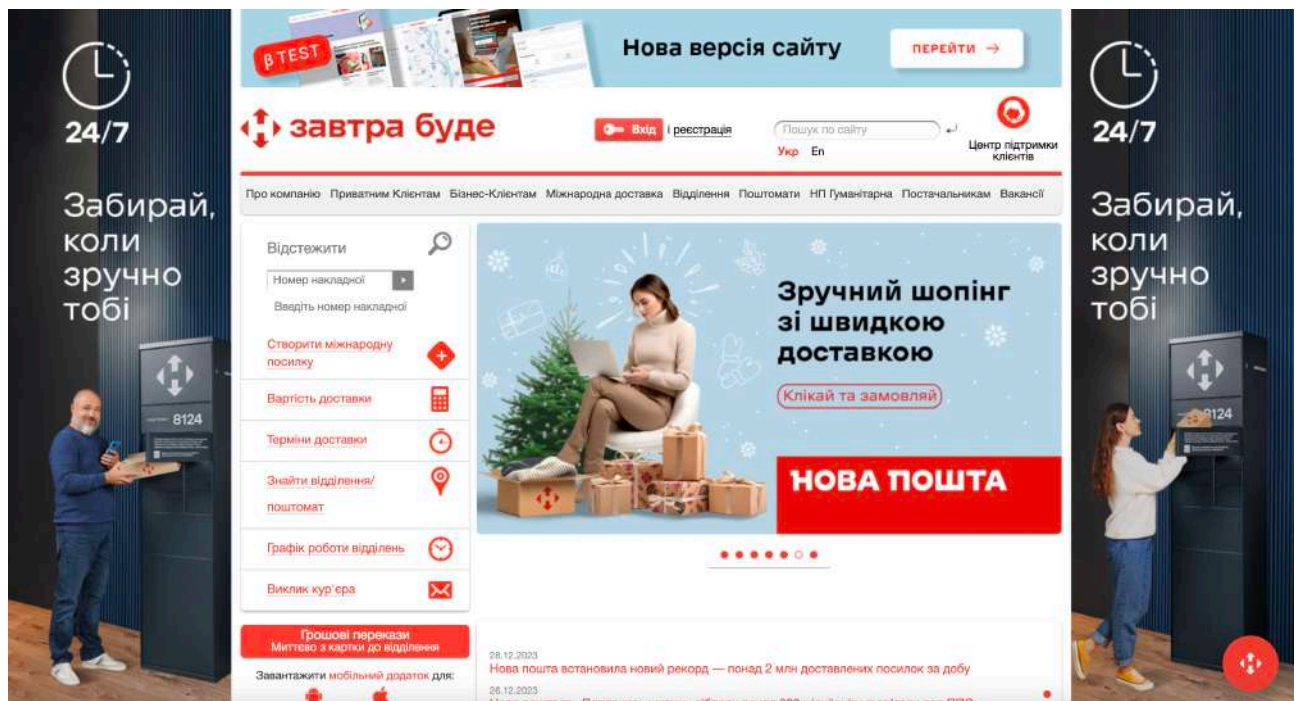


Рисунок 1.1 – Головна сторінка

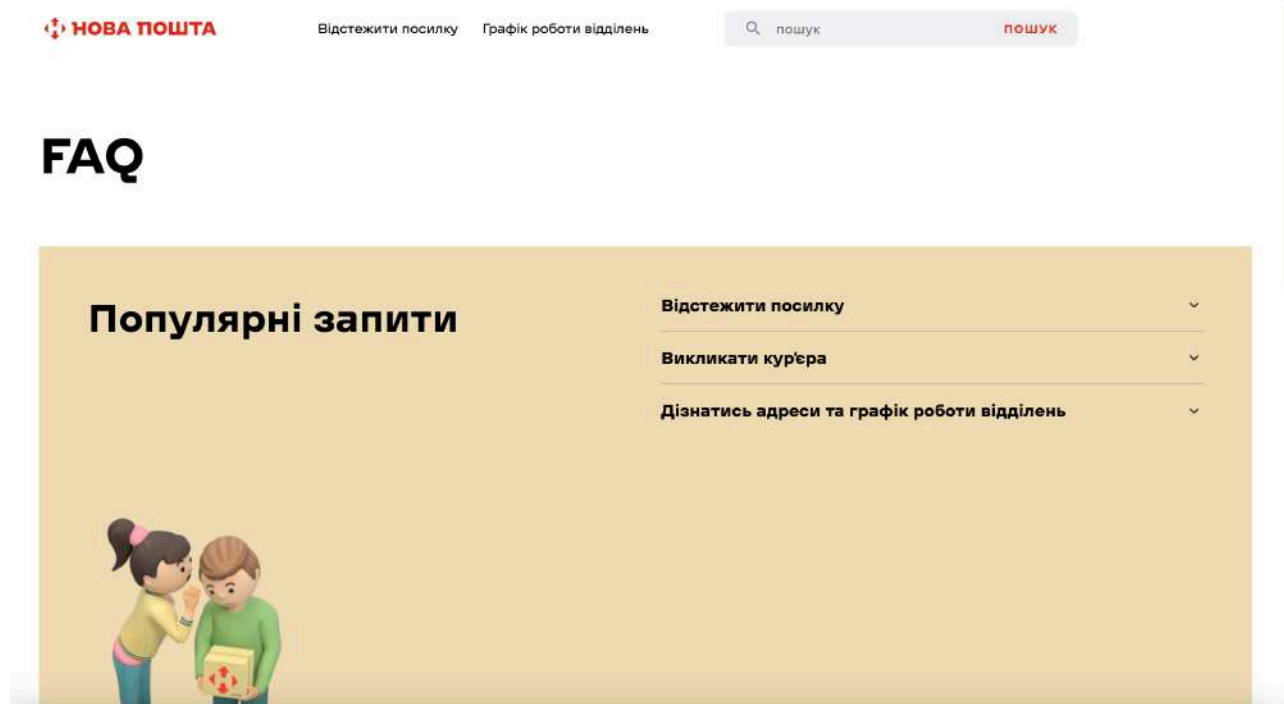


Рисунок 1.2 – сторінка FAQ

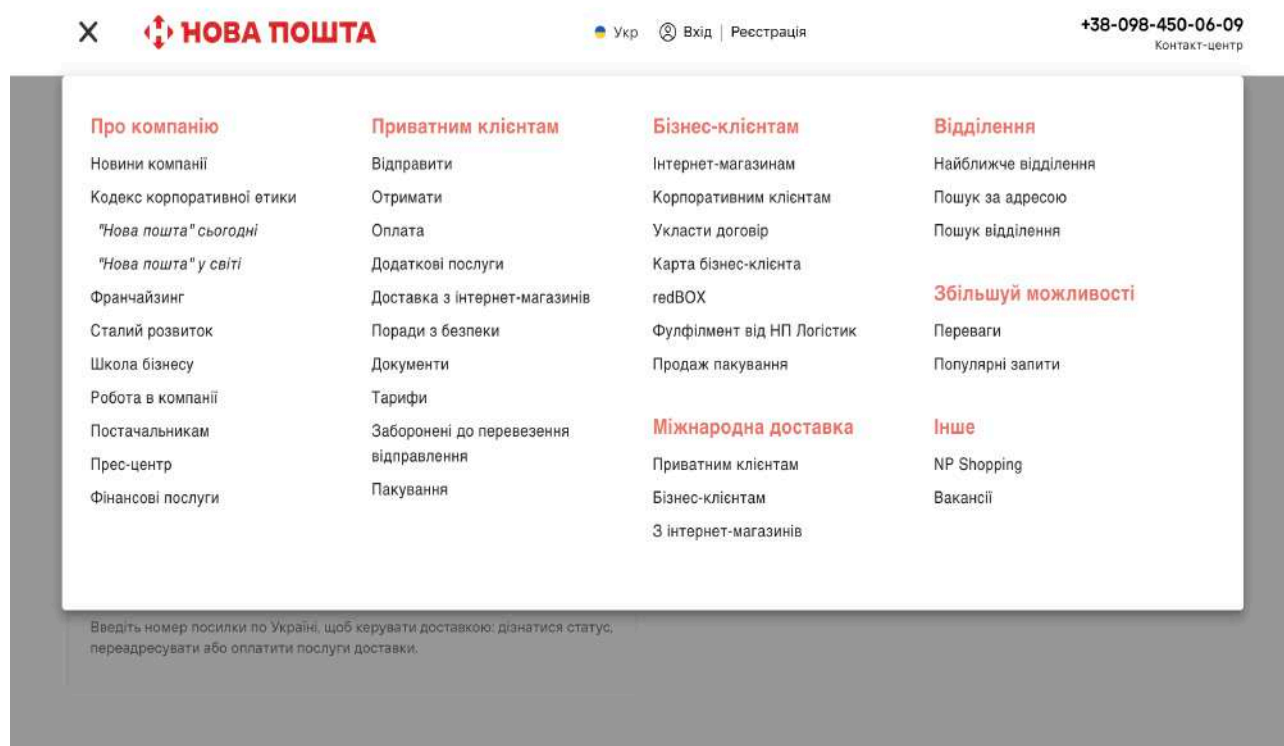


Рисунок 1.3 – Меню сайту

1.2 Компанія Meest Express

Meest Express — це компанія, яка спеціалізується на міжнародних перевезеннях та кур'єрських послугах в Україні. Meest Express пропонує послуги міжнародної доставки для клієнтів в Україні, зокрема для надсилань до і з різних країн світу.

Крім міжнародних пересилок, компанія також може надавати внутрішні кур'єрські послуги для внутрішніх відправлень в межах України. Як і багато інших кур'єрських служб, Meest Express надає можливість онлайн-відстеження для відправлень. Це дозволяє клієнтам відстежувати статус своїх посилок в режимі реального часу.

Компанія може мати власні відділення та термінали для зручності клієнтів, де можна здійснити відправлення чи отримати посылку.

В деяких випадках Meest Express може пропонувати додаткові послуги, такі як страхування посилок чи інші опції.

Сайт компанії надає можливість відстеження місця розташування вантажу або посилок в режимі реального часу.

Клієнти можуть здійснювати та оплачувати замовлення на послуги доставки через вебсайт. Також на сайті міститься інформація про розклади доставки, тарифи та інші умови послуг і розміщується контактна інформація, така як телефони, електронна пошта або онлайн-чат для зв'язку з представниками компанії.

Зручний особистий кабінет для клієнтів, де вони можуть управляти своїми замовленнями, слідкувати за статусами доставок та виконувати інші операції.

Основні сторінки сайту (<https://ua.meest.com/>) представлені на рис. 1.4 - 1.6.

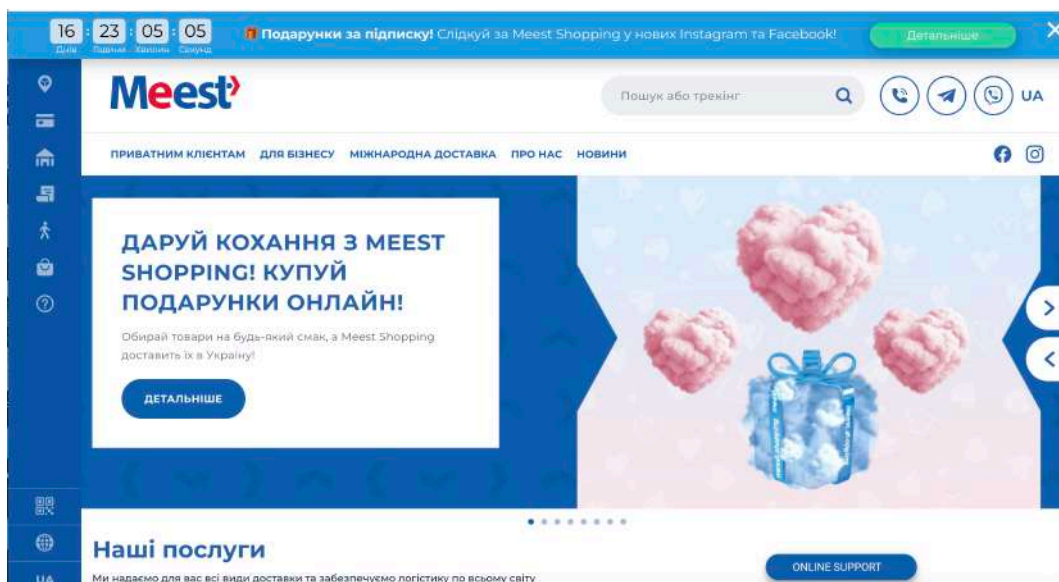


Рисунок 1.4 – Головна сторінка



Рисунок 1.5 – Меню сайту

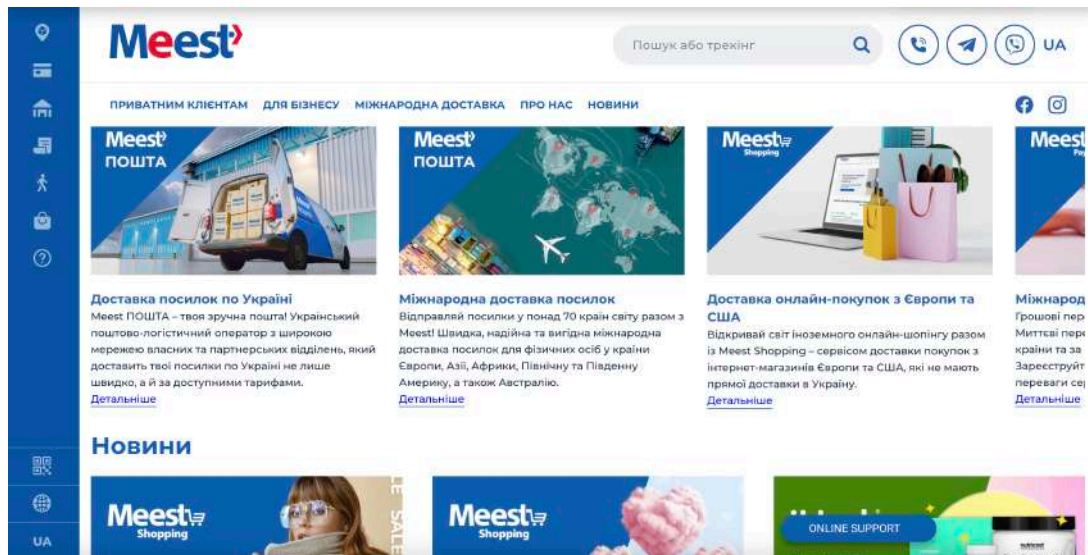


Рисунок 1.6 – Новини

1.3 Сайт Укрпошти

Укрпошта — це національний оператор послуг поштового зв'язку в Україні. Ось деякі ключові аспекти та функції Укрпошти:

- поштові послуги;
- відстеження посилок;
- поштові відділення;
- електронні послуги;
- платіжні послуги;
- онлайн-сервіси;
- інновації.

Укрпошта надає широкий спектр послуг, включаючи внутрішню і міжнародну доставку листів, посилок та інших видів вантажів.

Як і багато інших поштових операторів, Укрпошта забезпечує можливість відстеження місця розташування посилок в режимі реального часу. Укрпошта має розгалужену мережу поштових відділень, розташованих у різних районах та населених пунктах України.

Окрім традиційних поштових послуг, Укрпошта розвиває електронні сервіси, такі як електронна реєстрація та відправка посилок через Інтернет.

Укрпошта надає платіжні послуги, включаючи прийом та видачу грошових переказів. Клієнти можуть скористатися різноманітними онлайн-сервісами на вебсайті Укрпошти, такими як відстеження посилок, розрахунок вартості послуг та інші.

Укрпошта активно впроваджує інноваційні рішення, щоб поліпшити ефективність та зручність послуг для своїх клієнтів.

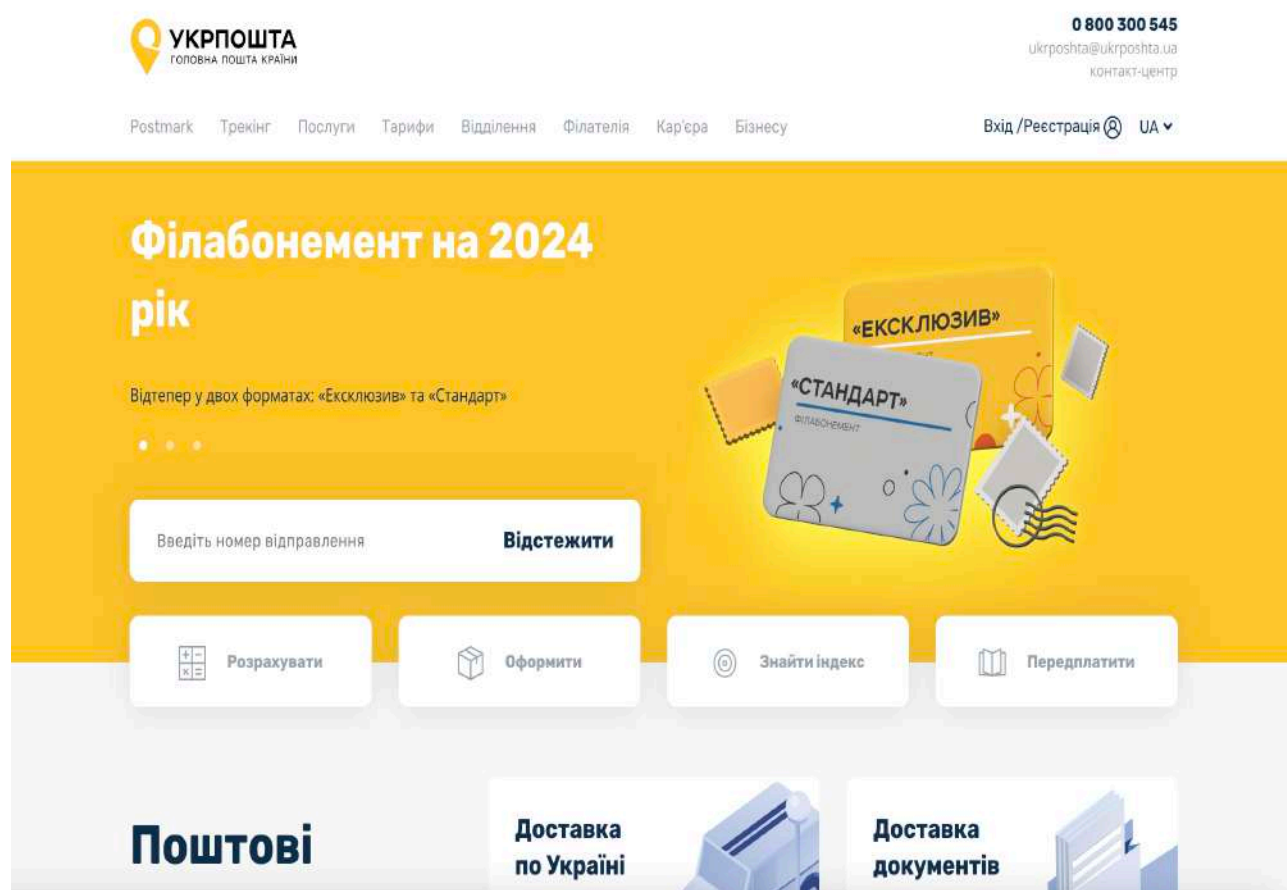


Рисунок 1.7 – Головна сторінка

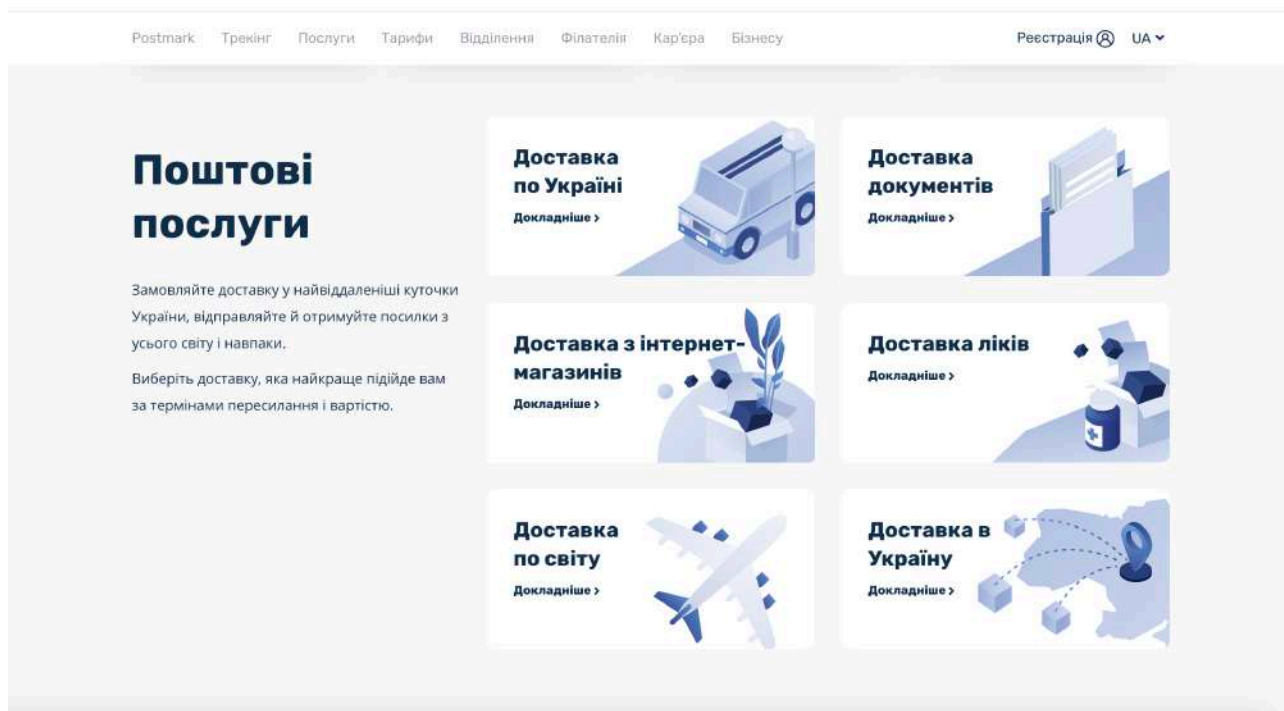


Рисунок 1.8 – Перелік послуг

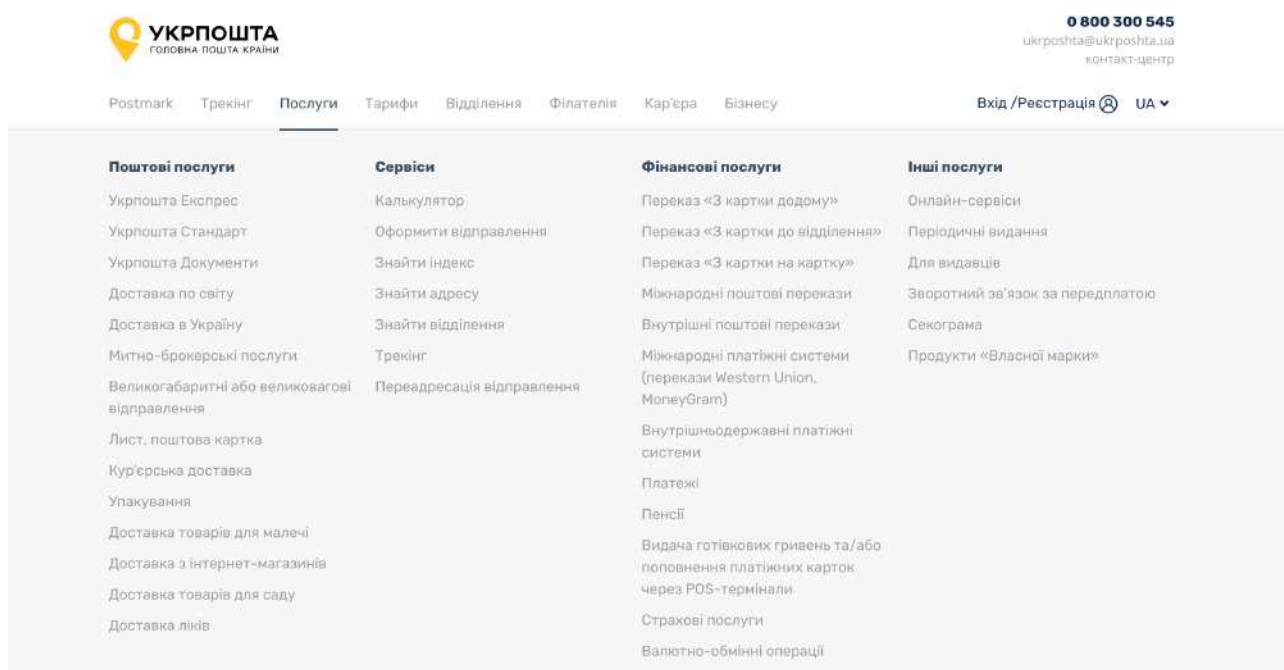


Рисунок 1.9 – Карта сайту

1.4 Вибір типу сайту

Було обрано сайт відправлення, отримання та повернення посилок як до відділень так й кур'єром. Сайт буде реалізовувати поштовий функціонал у вебпросторі.

Нова Пошта, Міст Експрес та Укрпошта - це три з найбільших і найпопулярніших поштових онлайн сервісів в Україні, і кожен з них має свої особливості та функціонал. Вони грають важливу роль у забезпеченні доставки товарів та листів на території країни, і вибір сервісу може вплинути на якість та швидкість доставки.

Нова Пошта відома своєю широкою мережею відділень та терміновою доставкою. Сервіс надає можливість стежити за відправленнями онлайн, розраховувати вартість доставки та вибирати способи оплати. Важливою функцією є послуга "Самовивіз", яка дозволяє отримати посилку в одному з відділень самостійно. Інтерфейс сайту є зручним та інтуїтивно зрозумілим.

Міст Експрес відомий своєю спеціалізацією на експрес-доставці великих вантажів та товарів. Сервіс також пропонує доставку листів і малих посилок. Важливою функцією є можливість відстеження вантажу онлайн і отримання інформації про час доставки. Міст Експрес також надає послуги зберігання і переадресації вантажів.

Укрпошта є державним оператором і надає поштові послуги на всій території України. Сервіс відомий своєю доступністю та розповсюдженістю. Укрпошта надає послуги стандартної та експрес-доставки. Важливою функцією є доступність послуги "Грошовий переказ", яка дозволяє відправити гроші разом з листом.

Кожен із цих сервісів має свої переваги та особливості. Нова Пошта та Міст Експрес часто використовуються для доставки товарів та великих вантажів, в той час як Укрпошта може бути зручним варіантом для листів та невеликих посилок.

Важливо також бути уважним до вартості послуг та строків доставки при виборі конкретного сервісу.

Сайт поштових послуг — це вебресурс, який пропонує зручний доступ до широкого спектру поштових операцій. Такий сайт повинен забезпечувати відвідувачам інструменти для відправлення, відстеження та управління посилками, а також давати детальну інформацію про послуги, тарифи та правила доставки.

Потенційний вебсайт поштової доставки може вирізнятися технічними перевагами порівняно з існуючими послугами як, наприклад, автоматизована маршрутизація та оптимізація. Система використовує передові алгоритми автоматизованої маршрутизації для оптимального вибору маршруту доставки. Це не лише забезпечує мінімальні терміни доставки, але і дозволяє оптимізувати витрати на транспорт. Система відстеження в реальному часі - інтегрована система відстеження в реальному часі надає можливість користувачам точно визначати місцезнаходження своїх поштових відправлень. Це допомагає уникнути затримок і збільшує рівень довіри до сервісу. Гнучкі API для інтеграцій - система надає гнучкі API для інтеграції з корпоративними інформаційними системами. Це спрощує процес взаємодії зі службою доставки для корпоративних клієнтів. Криптографічний захист даних – застосування сучасних криптографічних методів для захисту особистої і конфіденційної інформації клієнтів. Це гарантує високий рівень безпеки та конфіденційності усіх поштових відправлень. Оптимізація швидкості та надійності - використання технологій кешування, оптимізація мережі та постійний моніторинг роблять систему надзвичайно швидкою та надійною. Це сприяє ефективності та безперебійності процесу міжнародної поштової доставки.

Такі технічні аспекти дозволять нашому вебсайту виділятися серед існуючих міжнародних поштових послуг та забезпечити його технічну досконалість.

Автоматизована маршрутизація та оптимізація в сфері міжнародної поштової доставки може бути ключовим аспектом системи, спрямованої на

забезпечення ефективності та економії ресурсів. Даний функціонал базується на використанні передових алгоритмів, розроблених з урахуванням географічних особливостей, обсягу поштових відправлень та вартості транспортних ресурсів.

Система визначає оптимальний маршрут для кожної поштової відправки, враховуючи різноманітні параметри, такі як відстань, час, вартість та можливості транспортних засобів. Автоматизована маршрутизація дозволяє нам не лише гарантувати мінімальні терміни доставки, але і ефективно оптимізувати витрати на транспортні послуги. Система може врахувати густоту поштового трафіку на різних маршрутах, розподіл популяції в країнах та регіонах, а також інші фактори, які можуть впливати на шлях доставки. Використання цих алгоритмів у віртуальній системі дозволяє досягти оптимальності в реальному часі.

Важливим аспектом автоматизованої маршрутизації є також адаптивність системи до змінних умов, таких як транспортні пригоди, політичні обставини чи природні катастрофи. Система надає можливість автоматичного перерозподілу маршрутів та адаптації до нових обставин для забезпечення безперебійності та швидкості доставки.

Враховуючи вищезазначені аспекти, система автоматизованої маршрутизації створює ефективну та гнучку інфраструктуру для надання мінімальних термінів доставки та оптимізації витрат у сфері міжнародної поштової доставки.

Оптимізація швидкості та надійності є ключовим елементом в ефективному функціонуванні системи поштової доставки. Використання передових технологій, таких як технології кешування, оптимізація мережі та систематичний моніторинг, спрямовані на забезпечення максимальної продуктивності та надійності у всьому процесі доставки.

Механізми кешування використовуються для збереження часто використовуваних даних у швидкодіючих областях, що дозволяє системі оперативно надавати доступ до цих даних. Це зменшує час відгуку системи та прискорює процес обробки та відправки поштових відправлень. Оптимізація мережі гарантує ефективний обмін інформацією між різними етапами доставки,

що сприяє покращенню загального часу доставки та виключенню зайвих затримок.

Постійний моніторинг стану системи є важливим елементом в забезпеченні її надійності. Систематичний аналіз та відстеження параметрів, таких як швидкість передачі даних, стан мережі та робота серверів, дозволяють оперативно виявляти та вирішувати будь-які можливі проблеми. Це важливо для забезпечення безперебійності процесу доставки та уникнення можливих технічних збоїв чи відмов.

1.5 Основні вимоги

Основні вимоги до такого сайту включають:

- легкість використання - інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам легко навігувати та виконувати необхідні дії;
- безпека даних - надійний захист особистої і фінансової інформації користувачів;
- інтеграція - синхронізація з логістичними системами для актуального відстеження та платіжними системами для зручності оплати;
- доступність інформації - повна і актуальна інформація про послуги, доступна 24/7;
- підтримка клієнтів - ефективні канали зворотного зв'язку та підтримки, такі як онлайн-чати та гарячі лінії;
- доступність - сайт повинен коректно працювати на всіх пристроях і мати швидке завантаження сторінок;
- багатомовність - підтримка декількох мов для задоволення потреб міжнародної аудиторії;
- seo-оптимізація - структурування сайту та вмісту для кращого ранжування в пошукових системах. Такий сайт не тільки спрощує процеси

взаємодії з клієнтами, а й служить важливим інструментом для розвитку та розширення бізнесу.

1.6 Структура сайту

Структура сайту поштових відправлень має бути орієнтованою на зручність користувачів та ефективність виконання основних завдань, таких як відправлення, відстеження відправлень, доступ до персональних облікових записів та отримання інформації. Ось приклад типової структури такого сайту:

- головна сторінка (Home), що включає у себе огляд послуг, пошукову строку або форму для відстеження відправлень, посилання на головні функції, як-от відправлення посилки, ціни, відділення;
- відправлення (Send) включає форму для створення нової відправки, інструкції та вимоги до пакування та відправлення, інформацію про способи доставки та тарифи;
- відстеження (Tracking) включає ввід номера відстеження для перевірки статусу посилки та історію відстежень для зареєстрованих користувачів;
- особистий кабінет (My Account) включає управління персональними даними та адресою, перегляд історії відправлень та збережених відомостей та управління налаштуваннями сповіщень;
- інформація (Information) містить детальну інформацію про тарифи та умови послуг, часті запитання (FAQ), правила та регуляції;
- підтримка (Support) містить форму зворотного зв'язку, контактну інформацію, інтерактивна підтримку (чат-бот, онлайн-чат);
- філіали та локації (Locations) - пошуковик відділень, інформація про розташування та години роботи;
- новини та оновлення (News & Updates) - останні новини та оголошення, інформація про зміни в роботі сервісу;

- контакти (Contact) - форма для надсилання запитань, дані для зв'язку з керівництвом та службою підтримки;
- правова інформація (Legal) - політика конфіденційності, умови використання сервісу, авторські права та торгові марки.

Така структура має бути логічною та інтуїтивно зрозумілою, щоб користувачі могли легко знаходити необхідну інформацію та виконувати потрібні дії без зайвих кроків. Крім того, сайт повинен бути адаптованим для використання на різних пристроях, включаючи мобільні телефони та планшети.

2 АНАЛІЗ І ВИБІР ЗАСОБІВ РОЗРОБКИ

Вебсайти побудовані на моделі "клієнт-сервер", де задачі та обробка даних розподіляються між серверами, що надають послуги, та клієнтами, які ці послуги використовують. Для взаємодії між клієнтом і сервером використовується інтернет як основна платформа.

2.1 Клієнтська частина сайту

Клієнтська сторона – це та частина програмного забезпечення, що функціонує на боці користувача, а саме браузер, через який "клієнт" (користувач) вступає до сайту. До відомих браузерів належать Google Chrome, Mozilla FireFox, Internet Explorer, Opera та Safari.

HTML стандартизована як універсальна мова розмітки документів, що використовується на всіх вебсайтах. Браузер тлумачить HTML-код, щоб візуалізувати його на пристроях, таких як комп'ютери, планшети та телефони.

HTML, який зараз визнаний стандартом вебу, визначається та підтримується World Wide Web Consortium (W3C). Хоча HTML сам по собі не може виконувати динамічні функції, він відповідає за структурування та форматування документів, подібно до функцій, які надає Microsoft Word.

CSS є мовою каскадних таблиць стилів, яка призначена для стилізації елементів, написаних у мові розмітки, і відділяє вміст від його візуального представлення на сайті.

JavaScript, одна з трьох основних технологій веброзробки, є мовою програмування, яка підтримує об'єктно-орієнтоване, імперативне та частково функціональне програмування, і володіє властивостями, такими як динамічна типізація, автоматичне управління пам'яттю та використання функцій як об'єктів першого класу.

jQuery – це відома JavaScript-бібліотека з відкритим кодом, що була запущена Джоном Ресігом у січні 2006 року і широко використовується на більшості високореєтингових сайтів за даними W3Techs[1]. jQuery спрощує навігацію по DOM, анімацію, обробку подій та AJAX-операції, також надає розробникам інструменти для створення плагінів. Завдяки їй, розробники можуть абстрагуватися від низькорівневої взаємодії та створювати вражаючі візуальні ефекти, зосереджуючись на реалізації конкретних завдань, не турбуючись про розбіжності в роботі різних браузерів[2].

2.2 Серверна частина сайту

Серверна сторона веброзробки займається створенням "мізків" додатка, керуючи всіма основними функціями вебдодатка. Ці функції залежать від логіки і даних, які вебдодаток має обробляти. Фронтенд часто асоціюють з частиною користувача, тоді як бекенд - з серверною частиною вебдодатка.

У розробці було використано мову програмування PHP, яка перетворюється вебсервером на HTML, що потім відправляється клієнту. На відміну від клієнтських скриптових мов як JavaScript, PHP-код залишається невидимим для користувача, що забезпечує певний рівень безпеки, хоча це може вплинути на інтерактивність сторінки.

Ефективність є ключовою у програмуванні вебдодатків, призначених для численних користувачів. Однією з переваг PHP є її інтерпретований характер, який дозволяє швидко обробляти скрипти. Незважаючи на те, що компільовані програми працюють швидше, продуктивність PHP вважається достатньою для створення серйозних вебдодатків.

PHP відноситься до сім'ї скриптових мов разом з JavaScript і Python. PHP переважно використовується для серверної взаємодії, тоді як JavaScript застосовний і для фронтенд, і для бекенд-розробок. Python зазвичай використовується для бекенду. У розробці також було використано jQuery як інструмент для взаємодії з HTML-тегами та виконання асинхронних PHP-операцій. На рисунку 2.1 зображено загальну схему роботи сайту.

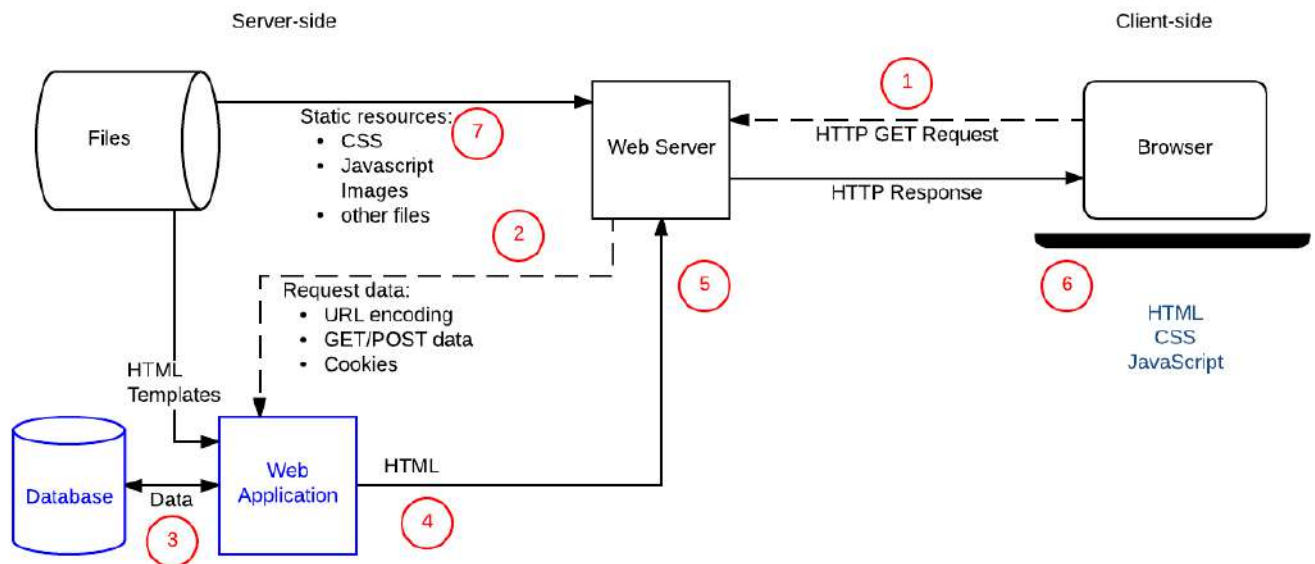


Рисунок 2.1 – Схема роботи сайту

2.3 База даних

MySQL є безкоштовною реляційною системою управління базами даних (СУБД), створеною компанією "ТсХ" для ефективної обробки великих обсягів даних. Відома своєю відкритістю коду, ця СУБД була розроблена як альтернатива платним системам і відрізняється від mSQL, розширюючи свої можливості з часом. MySQL використовується переважно для розробки динамічних вебсторінок завдяки своїй сумісності з багатьма програмними мовами[3].

В MySQL для збереження даних можуть використовуватися різні двигуни зберігання, які різняться способами запису та читання даних, хоча самі двигуни є прозорими для користувача. Популярні двигуни включають MyISAM, InnoDB, MEMORY та Berkeley DB.

У контексті порівняння, було розглянуто Postgresql, який є об'єктно-реляційною СУБД, що працює на одному двигуні зберігання. Він розглядає таблиці як об'єкти, які можуть успадковуватися, і управляється через об'єктно-орієнтовані функції. Відрізняється складною структурою файлів і записів.

MySQL не підтримує всі останні можливості стандарту SQL, але прагне до спрощення заради зручності користувачів, натомість Postgresql намагається відповідати сучасним стандартам SQL, що може призводити до підвищеної складності[4].

MySQL завантажує відповіді запитів у пам'ять клієнта, що може бути неефективним при великих обсягах даних. Postgresql же використовує курсори для роботи з даними, що зберігаються на сервері, і підтримує різноманітні типи індексів для оптимізації роботи з даними[5].

MySQL переважає Postgresql у швидкості обробки запитів, використовуючи первинний ключ, завдяки ефективній роботі з індексами.

Однак, PostgreSQL може пропонувати більшу функціональність і гнучкість для певних застосунків. У результаті, для подальшої розробки було обрано MySQL.

2.4 Вибір локального серверу

Локальний сервер — це програма, що дозволяє розробникам виконувати і тестувати вебсайти в режимі реального часу на їхніх власних комп'ютерах без потреби використання вебхостингу.

У сфері локальних серверів були проаналізовані Apache та Nginx — два найбільш популярні вебсервери з відкритим кодом, що забезпечують понад половину всього вебтрафіку. Вони обидва підтримують роботу з багатьма програмами та можуть взаємодіяти з іншими додатками, але вони не є взаємозамінними через унікальні функції кожного з них.

Apache має модулі мультипроцесорної обробки, що дають адміністраторам можливість налаштовувати обробку з'єднань, в той час як Nginx, який був розроблений пізніше, використовує асинхронні, неблокуючі алгоритми. Nginx створює процеси "воркери", які можуть одночасно обслуговувати тисячі з'єднань, завдяки швидкому циклу перевірки та обробки подій.

Коли мова йде про статичний та динамічний контент, Apache може обслуговувати статичний контент за допомогою стандартних методів доступу до файлів, з продуктивністю, яка залежить від вибраного модулю MPM. У свою чергу, Nginx сам по собі не може обробляти динамічний контент і повинен делегувати обробку PHP або іншого динамічного контенту зовнішньому процесору.

На підставі порівняння та аналізу потреб проєкту для подальшої розробки було вибрано Apache як локальний вебсервер.

2.5 Редактор коду та локальний сервер

Існує безліч редакторів коду, серед яких Sublime Text 3 виділяється своїми базовими можливостями, як-от автозавершення коду, підсвічування синтаксису, та функцією складання блоків коду. Завдяки Package Control у Sublime Text можна додати розширений функціонал, включаючи інструменти для налагодження, нові теми оформлення, та підтримку intellisense.

Atom — це інший безкоштовний, багатоплатформний редактор з відкритим кодом, створений GitHub. Він пропонує базові можливості, подібні до Sublime Text, а також має вбудовану підтримку різноманітних мов програмування та інтеграцію з GitHub. Atom також оснащений менеджером пакетів для пошуку, встановлення та створення власних пакетів, що розширюють його можливості.

Однак для розробки сайту було обрано PhpStorm, комерційне Інтегроване Середовище Розробки (IDE) від JetBrains, побудоване на платформі IntelliJ IDEA. PhpStorm — це розумний редактор для PHP, HTML і JavaScript, який пропонує функції аналізу коду, запобігання появі помилок та рефакторинг. IDE включає вбудований магазин додатків для розширення функціоналу, таких як кастомізація кольору тексту та підтримка класів та методів, створених за допомогою Artisan[6].

Зі сторони засобів безпеки рівня програми PHP включає надійні механізми шифрування. Також PHP сумісний з багатьма додатками інших розробників, що дозволяє легко інтегрувати його з захищеними технологіями електронної комерції. Вихідний код PHP не можна переглянути у браузері, оскільки він виконується на сервері[7].

У плані гнучкості, PHP використовується не лише у поєднанні з HTML, але й із JavaScript, WML, XML та іншими мовами програмування. PHP-код може передаватися будь-яким браузерам і пристроям, в тому числі стільниковим

телефонам, портативним комп'ютерам. PHP-код можна виконувати в режимі командного рядка.

Використання класів у PHP подібне до C, клас об'явлюється ключовим словом `class`, може мати властивості та методи певної приватності, а саме публічними, що доступні звідусіль, захищеними, до яких має доступ сам клас та його наслідники, та приватними, до яких не має доступ ніхто, окрім самого класу. PHP підтримує усі механізми ООП, такі як інкапсуляція, поліморфізм та наслідування. Також використання ключових слів `final`, `abstract`, `extend`, `implement`.

PHP 5 [8] був випущений ще в 2004 році та працює під управлінням нового Zend Engine II. Рушій Zend - це набір компонентів, які роблять PHP - PHP. Найважливішим компонентом Zend є віртуальна машина Zend, яка включає компоненти Zend Compiler та Zend Executor. Ми також можемо додати розширення OPcache zend до тої ж категорії. Ці три компоненти є ядром PHP, вони є критичними і найскладнішими частинами вихідного коду. Рушій Zend, що лежить в основі інтерпретатора, був повністю перероблений у Zend Engine 2, відкривши шлях для подальших удосконалень. PHP 5 включав нові функції, такі як покращена підтримка ООП, об'єкти даних PHP, також відомі як розширення PDO (це легкий і послідовний інтерфейс доступу до баз даних) та багато іншого.

Серед інших корисних доповнень в PHP 5 присутня нова схема обробки виключень. Конструкція `try/catch/throw` дозволяє весь код обробки помилок локалізувати в одному місці сценарію[9].

Всі основні бібліотеки для роботи з XML, що запозичені з PHP 4, були піддані серйозній переробці. Такі популярні розширення, як SAX, DOM і XSLT, тепер використовують інструмент libxml2, що робить їх ще ефективнішими.

У PHP 5 також включені два нові модулі для роботи з протоколами - SimpleXML і SOAP. SimpleXML дозволяє значно спростити роботу з XML-даними, представляючи вміст XML-документа у вигляді PHP-об'єкту.

Розширення SOAP дозволяє будувати на PHP сценарії, що обмінюються інформацією з іншими додатками за допомогою XML-повідомлень поверх

існуючих вебпротоколів, наприклад HTTP. Модуль для роботи з SOAP для PHP 5 надає засіб швидкого створення ефективних SOAP-клієнтів та серверів. Новий модуль PHP 5 MySQLi (MySQL Improved) призначений для роботи з MySQL-сервером версій 4.1.2 і вище, реалізує не тільки процедурний, але й об'єктно-орієнтований інтерфейс до MySQL. Додаткові можливості цього модуля включають SSL, контроль транзакцій, підтримка реплікації і ін. Офіційна підтримка безпеки для PHP 5.6 закінчилася 31 грудня 2018 року.

PHP 6 [11] планувався як версія, що забезпечить нативну підтримку Unicode на базовому мовному рівні. Однак гілку розробки закрили, і більшість нововведень було перенесено на PHP 5.3 та 5.4.

PHP 7 [12] був випущений у грудні 2015 року і приніс багато нових функцій [13], таких як:

Zend Engine 3 (покращення продуктивності та підтримка 64-розрядного integer для Windows);

- уніфікований синтаксис змінних;
- компіляційний процес на базі AST;
- Closure::call();
- ?? як null coalesce оператор;
- Unicode синтаксис;
- об'ява типу повернення;
- об'ява скалярних типів;
- <=> "spaceship" тристоронній оператор порівняння;
- анонімні класи;
- заміна багатьох внутрішніх “помилки” PHP на сучасні виключення;
- короткий синтаксис для імпорту багатьох об'єктів з одного namespace.

Найціннішою частиною релізу став Zend v3. Основою PHP 7 є гілка PHP, яка спочатку отримала назву PHP наступного покоління (phpng). Його мета полягала в оптимізації продуктивності, що можна було б зробити шляхом рефакторингу Zend Engine і зберегти майже повну сумісність мови. 14 липня 2014 року тести на основі WordPress, які служили основним орієнтиром для

phpng, показали майже 100% збільшення продуктивності. Зміни в phpng полегшують покращення продуктивності в майбутніх версіях, оскільки більш компактні структури даних підходять для успішної майбутньої міграції на компілятор JIT. Перероблений рушій Zend отримав назву Zend Engine 3.

Порівняно PHP 7 з PHP 5.6 має наступні новації (більшість із цих покращень стосуються рушія Zend):[14]

- рефакторинг базових структур даних;
- покращена умова виклику віртуальної машини;
- новий API розбору параметрів;
- новий менеджер пам'яті;
- численні вдосконалення виконавця віртуальної машини;
- значно зменшено використання пам'яті;
- покращено функції `__call()` та `__callStatic()`;
- покращена конкатенація рядків;
- покращений пошук символів у рядках.

Покращення продуктивності PHP 7.1 [15]:

- нова система оптимізації SSA (вбудована в opcache);
- глобальна оптимізація байт-коду PHP на основі виведення типів;
- вузькоспеціалізовані обробники операційних кодів для віртуальних машин.

Багато фатальних помилок були перероблені у винятки в PHP 7. Ці винятки успадковують клас `Error`, який, у свою чергу, реалізує інтерфейс `Throwable`. Це означає, що користувацькі обробники помилок можуть бути не викликані, тому що замість виклику помилки, буде виняток.

Раніше деякі внутрішні класи повертали `null` або марний об'єкт, якщо конструктор зазнав невдачі. Всі вбудовані класи тепер у такому випадку будуть викидати виняток `Exception`, так само, як це вже роблять користувацькі класи.

PHP 7 використовує абстрактне синтаксичне дерево для аналізу файлів з вихідним кодом. Це дозволило внести безліч покращень у мову, які раніше були

неможливі через обмеження парсера, що використовувався в попередніх версіях PHP, але призвело до видалення деяких особливих можливостей з міркувань узгодженості та порушило зворотну сумісність.

Оголошення скалярних типів було введено у двох варіантах: примусовий (за замовчуванням) та суворий. Наступні типи можуть використовуватися для оголошення параметрів (в обох варіантах): рядки (string), цілі (int), числа з плаваючою точкою (float) та логічні значення (bool). Вони доповнюють аргументи інших типів, введених у PHP 5: імена класів, інтерфейсів, array та callable.

У PHP 7 додана підтримка оголошення типу, що повертається. Аналогічно як і оголошення типів аргументів, оголошення типу значення, що повертається вказує, значення якого типу повинна повернути функція. Для оголошення типу значення, що повертається доступні всі ті ж типи, що і для оголошення типів аргументів.

PHP 8 [10] був випущений 26 листопада 2020 р. PHP 8 є повноцінним релізом та має великі зміни, порівняно з попередніми версіями. Найбільш затребуваною інновацією є компілятор Just-in-time (JIT). PHP JIT реалізовано як майже незалежну частину OPcache. Його можна вмикати/вимикати під час компіляції PHP та під час виконання. Коли ввімкнено, код файлів PHP зберігається в додатковій області спільної пам'яті OPcache, а `op_array→opcodes[].handler(s)` зберігають покажчики на точки входу JIT-коду. Нові функції та зміни, які включені в компілятор JIT PHP 8, можуть забезпечити істотне підвищення продуктивності в деяких випадках. Було заявлено, що поліпшення для більшості вебсайтів буде менш істотним, ніж оновлення з PHP 5 до PHP 7. Покращення продуктивності за рахунок додавання компілятора JIT, як очікується, буде більш суттєвим для математичних операцій, ніж для звичайних випадків веброзробників. Також JIT надає майбутній потенціал для переміщення деякого коду з C на PHP через покращення продуктивності для деяких випадків використання.

ЛІТ постійно перевіряє виконуваний код і шукає частини, які можна прискорити. У цей момент код компілюється або перекомпілюється на льоту, що покращує швидкість.

Отож, ЛІТ переводить гарячі частини проміжного коду в машинний. Оминаючи компіляцію, вона зможе покращити продуктивність та використання пам'яті.

Переваги ЛІТ компілятора приблизно (як зазначено в RFC):

- значно краща продуктивність для математичних операцій;
- трохи краща продуктивність для типових вебзавдань;
- потенціал переміщення додаткового коду з С на РНР.

Також згадати null-safe виклик методів, що є переробкою для оператора IF, який не має оператора ELSE і був представлений наступним чином:

```
$human_readable_date = $user->getBirthday()?-
>diffForHumans();
```

Також додано масиви, що починаються з негативного індексу, більш зрозумілі порівняння та вирази відповідності.

Тож як OPcache робить РНР швидшим і які зміни в процесі виконання за допомогою ЛІТ?

OPcache покращує продуктивність РНР, зберігаючи попередньо скомпільовані байти сценаріїв у спільній пам'яті, усуваючи необхідність завантажувати та аналізувати сценарії РНР для кожного запиту.

Коли OPcache увімкнено, інтерпретатор РНР виконує зазначені вище чотири кроки лише під час першого запуску сценарію. Оскільки байтові коди РНР зберігаються у спільній пам'яті, вони одразу доступні як низькорівневе проміжне представлення та можуть бути негайно виконані у віртуальній машині Zend.

У РНР 8.0.0 як продовження позиційних параметрів з'явилися іменовані аргументи. З їхньою допомогою аргументи функції можна передавати за ім'ям параметра, а не за його позиції. Таким чином аргумент стає

самодокументованим, незалежним від порядку та зазначеного значення за умовчанням. Іменовані аргументи передаються шляхом додавання через двокрапки імені параметра перед його значенням. Імена параметрів можна використовувати як зарезервовані ключові слова. Ім'я параметра має бути ідентифікатором, тобто він не може бути створений динамічно.

Було додано атрибути, що пропонують можливість додавати структуровані, машиночитані метадані для наступних декларацій у коді: класи, методи, функції, параметри, властивості та константи класу. Прив'язані метадані можна отримати під час виконання за допомогою Reflection API. Таким чином, атрибути можна розглядати як мову конфігурації, вбудовану безпосередньо в код.

За допомогою атрибутів можна розділити абстрактну реалізацію будь-якого функціоналу та особливості його використання у коді. У певному сенсі це можна порівняти з поділом інтерфейсу та його реалізацій. Але інтерфейси та реалізації – це про код, а атрибути – про додавання додаткової інформації та конфігурацію. Інтерфейси можуть бути реалізовані тільки класами, тоді як атрибути також застосовні для методів, функцій, параметрів, властивостей і констант класів. Таким чином, вони є набагато більш гнучким механізмом, ніж інтерфейси.

Синтаксис атрибутів складається із кількох частин. По-перше, декларація атрибута завжди починається з символу `#`[і закінчується]. У середині перерахування з одного або більше розділених комою атрибутів. Атрибути можна задавати за допомогою неповних, повних та абсолютних імен.

Аргументи атрибутів опціональні, але якщо вони є, то заключаються в дужки `()`. Аргументи атрибутів можуть бути або конкретними значеннями, або константними виразами. Для аргументів можна використовувати як позиційний синтаксис, так і синтаксис іменованих аргументів. Коли атрибут затребується за допомогою Reflection API, його ім'я трактується як ім'я класу, а аргументи передаються його конструктор. Таким чином, для кожного атрибута має існувати відповідний клас.

Починаючи з PHP 8.0.0, параметри конструктора можна використовувати для завдання відповідних властивостей об'єкта. Це досить поширена практика — надавати властивостям об'єкта параметри, передані в конструктор, не роблячи жодних додаткових перетворень. Визначення властивостей класу в конструкторі дозволяє значно скоротити кількість шаблонного коду такого випадку. Починаючи з PHP 8.0.0, до властивостей та методів можна також звертатися за допомогою оператора "nullsafe": `?->`. Оператор `nullsafe` працює так само, як доступ до властивості або методу, як зазначено вище, за винятком того, що якщо розіменування об'єкта видає `null`, то буде повернутий `null`, а не викинутий виняток. Якщо розіменування є частиною ланцюжка, решта ланцюжка пропускається. Аналогічно висновку кожного звернення в `is_null()`, але компактніший.

У роботі було проведено дослідження, яке має такі результати (рис. 2.2).

```
-- Generic PHP information

PHP version: 5.6.40-54+ubuntu20.04.1+deb.sury.org+1
Server: PHP 5.6.40-54+ubuntu20.04.1+deb.sury.org+1 Development Server
Maximum execution time: 30 seconds

Maximum memory size: 1.00 B (1 bytes)
Maximum upload size: 2.00 MB (2097152 bytes)

Modules loaded:

-- Disk performance information
Results sorted by file size (in bytes) in milliseconds (less is better), for a total of 100 cycles:
+-----+-----+-----+-----+-----+-----+-----+-----+
|      512      |      1024     |      2048     |      4096     |      8192     |      16384    |      32678    |      65536    |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 0.0077745914459229 | 0.0070915222167969 | 0.0076439380645752 | 0.0086843967437744 | 0.0097451210021973 | 0.011658906936646 | 0.013409052981567 | 0.017565727233887 |
+-----+-----+-----+-----+-----+-----+-----+-----+

-- CPU performance information
Math operation results by function in milliseconds (less is better), for a total of 99999 cycles:
+-----+-----+-----+-----+-----+-----+-----+-----+
|      abs      |      acos     |      asin     |      atan     |      bindec   |      floor    |      exp      |      sin      |      tan      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 0.028146982192993 | 0.027643918991089 | 0.028227090835571 | 0.027456998825073 | 0.033526182174683 | 0.026874780654907 | 0.029170989990234 | 0.030976057052612 | 0.032726049423218 |
+-----+-----+-----+-----+-----+-----+-----+-----+

String operation results by function in milliseconds (less is better), for a total of 99999 cycles:
+-----+-----+-----+-----+-----+-----+-----+-----+
|  addslashes  |  chunk_split  |  metaphone   |  strip_tags   |  md5         |  sha1        |  strtoupper   |  strtolower   |  strrev      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 0.041321039199829 | 0.036525964736938 | 0.044986009597778 | 0.04217791557312 | 0.046206951141357 | 0.049161911010742 | 0.03412389755249 | 0.034275054931641 | 0.032958030700684 |
+-----+-----+-----+-----+-----+-----+-----+-----+

Loop operation results in milliseconds (less is better), for a total of 99999 cycles: 0.0030131340026855

If/Else operation results in milliseconds (less is better), for a total of 99999 cycles: 0.0036921501159668

== MySQL performance information
MySQL version:
Query (Encode + random) operation results in milliseconds (less is better), for a total of 100000 cycles: 4.5061111450195E-5
```

Рисунок 2.2 - Вигляд отриманих результатів

Для відображення інформації існує окремий метод, який використовує метод іншого класу, а саме візуального представлення на рядки, для покращення розуміння під час читання. Його реалізація наведена нижче.

Лістинг 2.1 – Метод виведення результуючої інформації

```
public static function print(string $input, string
$delimiter = "\n\n"): void
{
    \print_r($input . $delimiter);
}
```

Контрольні показники використовуються для досягнення конкурентного результату з будь-яким іншим подібним предметом. У нашому випадку - відмінність версії PHP для найшвидшого виконання. Усі результати були внесені до таблиці Excel, де було побудовано порівняльні графіки.

Тести були проведені на продуктивність дискових і процесорних операцій та перетворення рядків в залежності від версії мови.

Результати аналізу наведені нижче і представляють операції для кожної версії PHP, для виявлення найшвидшої. Для всіх тестів нижче значення - краще.

Тести було виконано на локальній машині, яка має: Ryzen 5600H, 2x8 GB 3200 MHz CL22 PC4-25600 RAM, SSD Samsung M.2 512 GB, Ubuntu 21.04.

Використані версії PHP: 5.6.40, 7.3.31, 8.0.1[17]

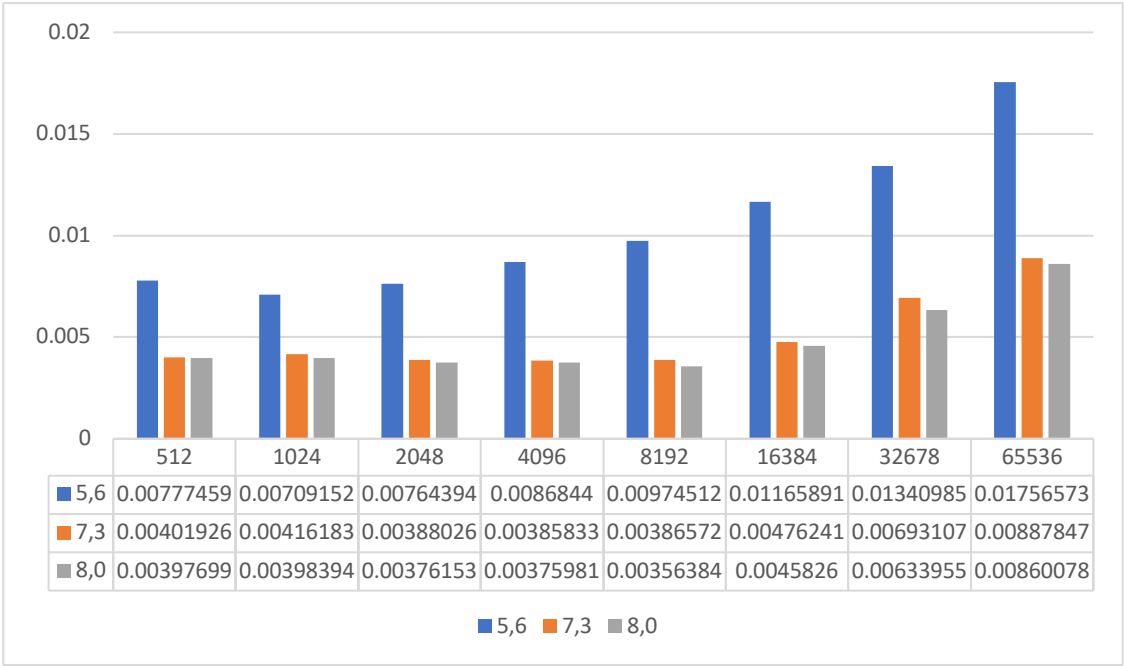


Рисунок 2.3 - Тести запису на диск

Як можна побачити з графіку 2.3, версія 8.0 перемагає для запису файлу будь-якого з вказаних раніше розмірів.

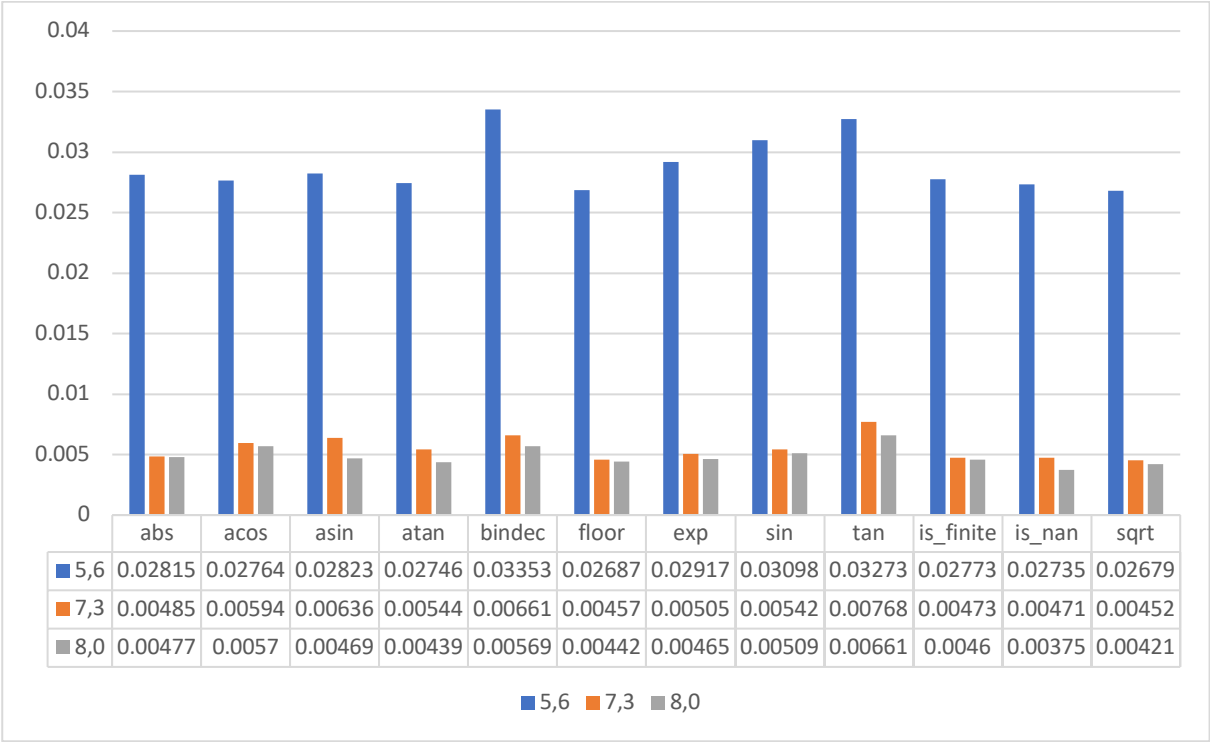


Рисунок 2.4 - Тести процесорних операцій

Як можна бачити на графіку 2.4, версія 8.0 також випереджує попередні версії під час виконання арифметичних операцій, при цьому версія 5.6 відстає у більш ніж 5 разів.

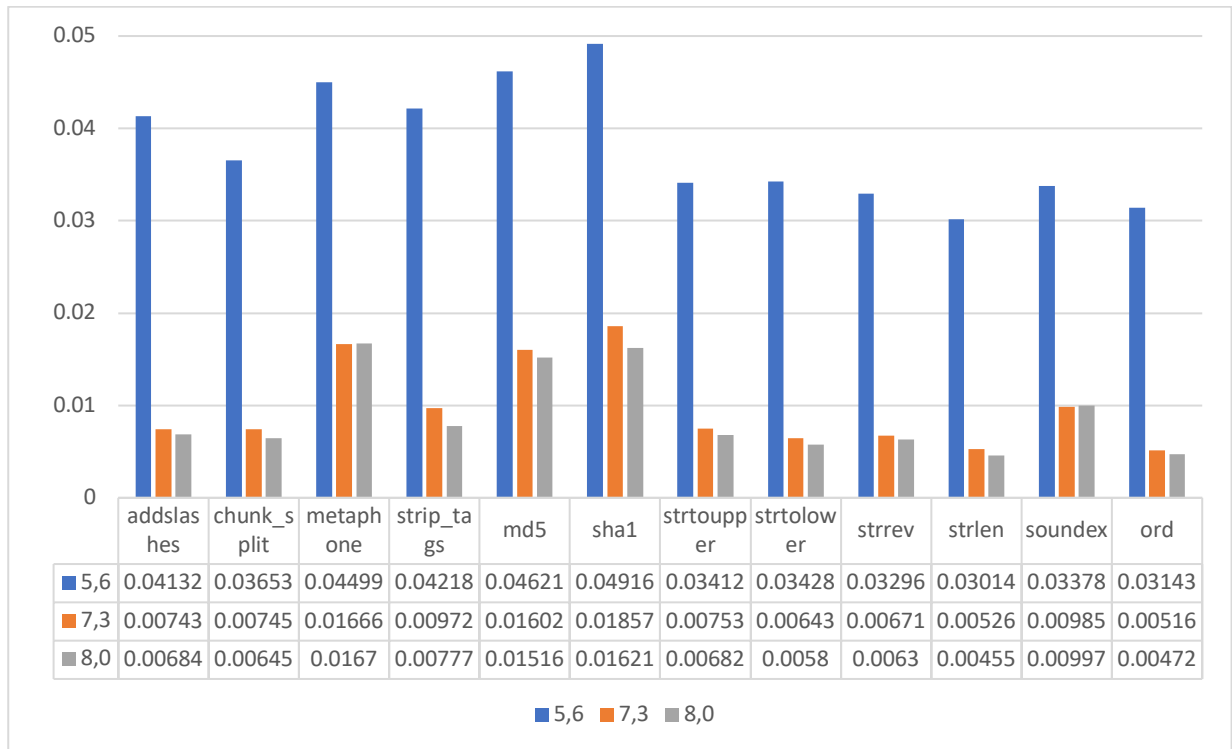


Рисунок 2.5 - Тести перетворення рядків

У тестах перетворення рядків версія 8.0 теж попереду конкурентних версій, версія 5.6 відстає у більш ніж 4 рази. Результати тестування наведено на графіку 2.5

Таблиця 2.1 - Перевірка швидкості операцій циклу

Loop operation results in milliseconds (less is better), for a total of 99999 cycles:	
5,6	0.0030131340026855
7,3	0.0012149810791016
8,0	0.00072503089904785

У таблиці 2.1 наведено час тестування операторів циклу. Версія 8.0 перемагає серед інших версій, версія 5.6 відстає майже у 4.5 рази, версія 7.3 відстає майже удвічі.

Таблиця 2.2 - Перевірка швидкості операторів If/Else

If/Else operation results in milliseconds (less is better), for a total of 99999 cycles:	
5,6	0.0036921501159668
7,3	0.0016400814056396
8,0	0.0012068748474121

У таблиці 2.2 наведено час тестування операторів умови. Версія 8.0 не так випереджає 7.3, однак швидше її на 25%.

3 ПРОЄКТУВАННЯ САЙТУ

3.1 Створення дизайну сайту

Головна мета дизайну полягає у забезпеченні знайомства користувача з вебсторінкою. Дизайн полегшує інтеракцію користувачів із сайтом, що сприятливо впливає на конверсію та поведінкові фактори. Він також формує необхідне сприйняття компанії.

Під час розробки зовнішнього вигляду сайту важливо створити дизайн, який чітко представлятиме його структуру. Основною ціллю дизайну є інтеграція всіх інформаційних блоків та створення приємного враження для відвідувачів.

Сайт служить маркетинговим інструментом та представництвом компанії в мережі. Клієнти, які відвідують сайт, повинні без зусиль знаходити всю потрібну інформацію.

Дизайн сторінки відстеження посилки представлений на ілюстрації 3.1.

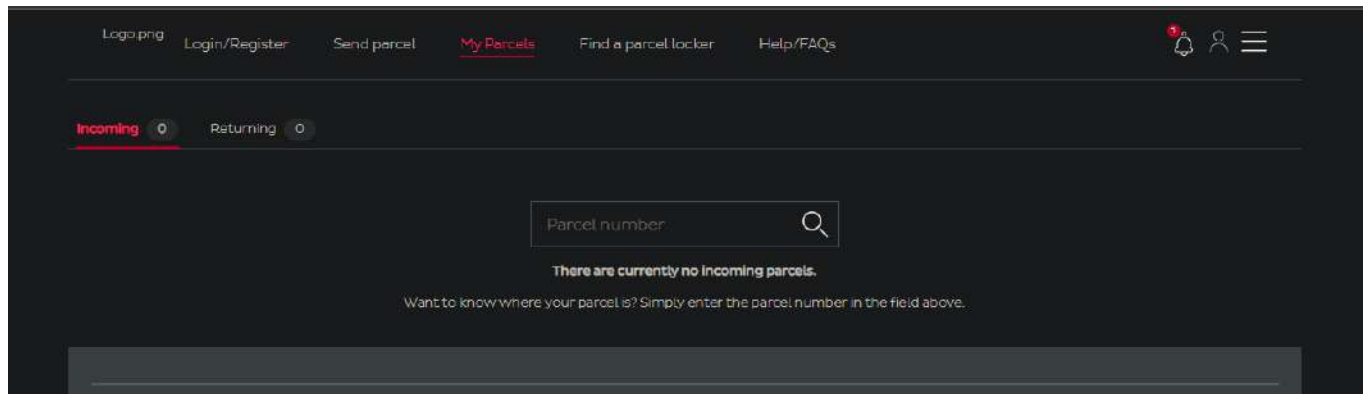


Рисунок 3.1 – Дизайн сторінки відстеження посилки

Загальний дизайн адмін-панелі зображений на рисунку 3.2.

Parcel No	From	To	Delivery	Pickup	Sent Date	Estimate Date
123456	New York, NY	Los Angeles, CA	Express	Office	2023-03-15	2023-03-17
789101	Chicago, IL	Houston, TX	Standard	Home	2023-03-16	2023-03-22

Рисунок 3.2 – Загальний дизайн адмін-панелі

Власне, дизайн задає загальний стиль сайту, допомагає відвідувачеві зрозуміти з першого погляду, що на нього тут чекає.

Грамотно розроблений дизайн – один з головних факторів, що визначають відвідування вебсторінки.

3.2 Проєктування бази даних

Процес проєктування бази даних розбивається на дві основні стадії: концептуальне та логічне моделювання. Під час цих етапів визначаються сутності та типи даних для їх атрибутів.

Логічне, або даталогічне, проєктування полягає у створенні схеми бази даних на базі вибраної моделі даних, такої як реляційна модель. В контексті реляційної моделі даних, даталогічна модель являє собою сукупність схем відносин, зазвичай із зазначенням первинних ключів та визначенням зв'язків, які виражені через зовнішні ключі.

Трансформація концептуальної моделі у логічну зазвичай проводиться за допомогою формалізованих правил, що дозволяє частково автоматизувати цей процес. На етапі логічного проєктування враховуються особливості моделі даних, але не обов'язково специфіки певної СУБД.

Під час розробки проєкту ми працювали з кількома сутностями в базі даних. Характеристика сутності «users» представлена на рисунку 3.3.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐ 1	id 	int(11)			No	None		AUTO_INCREMENT	 Change  Drop  More
☐ 2	name	varchar(50)	utf8mb4_general_ci		No	None			 Change  Drop  More
☐ 3	surname	varchar(50)	utf8mb4_general_ci		No	None			 Change  Drop  More
☐ 4	email	varchar(80)	utf8mb4_general_ci		No	None			 Change  Drop  More
☐ 5	sex	tinyint(4)			No	None			 Change  Drop  More
☐ 6	birthday	date			No	None			 Change  Drop  More
☐ 7	address	int(11)			No	None		FOREIGN_KEY	 Change  Drop  More
☐ 8	violations	smallint(6)			No	None			 Change  Drop  More

Рисунок 3.3 – Характеристика сутності «users»

3.3 Створення проєкту

Для реалізації проєкту був використано Laravel. Створення проєкту поштової доставки на фреймворку Laravel можна розділити на кілька основних етапів:

а) планування та аналіз вимог:

1) визначення цілей проєкту, його можливостей та функціоналу;

- 2) аналіз цільової аудиторії та вимог до системи;
- 3) складання технічного завдання.

б) проєктування системи:

- 1) створення концептуальної моделі бази даних;
- 2) розробка архітектури проєкту, включно з визначенням основних компонентів системи та їх взаємодії.

в) налаштування робочого середовища:

- 1) встановлення та конфігурація середовища розробника;
- 2) налаштування Laravel і його залежностей через Composer.

г) розробка:

- 1) розробка маршрутів, контролерів, моделей і переглядів (routes, controllers, models, views);
- 2) використання міграцій для створення та модифікації структури бази даних;
- 3) реалізація бізнес-логіки, включно з системами авторизації та валідації даних.

г) тестування:

- 1) написання та виконання юніт-тестів;
- 2) проведення функціонального тестування;
- 3) виявлення та виправлення помилок.

д) інтеграція та встановлення:

- 1) інтеграція з зовнішніми сервісами (наприклад, системи відстеження відправлень);
- 2) розгортання проєкту на сервері.

е) запуск та підтримка:

- 1) запуск проєкту в експлуатацію;
- 2) моніторинг роботи системи та її оптимізація;
- 3) внесення змін за необхідності на основі відгуків користувачів.

є) оновлення та масштабування:

- 1) оновлення системи для підвищення її продуктивності та безпеки;

2) масштабування системи з урахуванням зростаючих вимог до обробки даних та збільшення навантаження.

У процесі розробки важливо дотримуватися принципів чистого коду та використовувати кращі практики Laravel, щоб забезпечити високу якість та легкість підтримки проєкту.

4 РОЗРОБКА САЙТУ

4.1 Front-end та JS розробка

Під час створення нашого сайту використовується код розмітки CSS, за допомогою якого блоки сторінки розміщуються згідно потреб.

Першим елементом сторінки є шапка сайту, або header. Вона відображається на всіх сторінках сайту. У лістингу 4.1 наведено частину коду файлу `head.blade.php` для сторінки, яку бачить користувач. Також на сторінці використано гугл-мапу з геовідміткою заходу. На лістингу 4.1 наведено код файлу `head.blade.php` для адмін-сторінки.

Лістинг 4.1 – head.blade.php

```

<head>
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootst
trap.min.css" integrity="sha384-
gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap-
select@1.13.14/dist/css/bootstrap-select.min.css">
    <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
    <script src="<?php echo VIEW?>/js/wizard.js"></script>
    <script src="<?php echo VIEW?>/js/countrypicker.js"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.
bundle.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/bootstrap-
select@1.13.14/dist/js/bootstrap-select.min.js"></script>
    <script
src="https://requirejs.org/docs/release/2.3.5/minified/require.js"
></script>
    <script src="<?php echo VIEW?>/js/map.js"></script>
    <link rel="stylesheet" href="<?php echo VIEW?>/css/style.css">
<script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyCGtPVaCt43Q
tPw3zoJOogbePaBfz28Z0w&callback=initMap&libraries=&v=weekly"
    async
></script>

```

Лістинг 4.2 – header.php

```

<head>
    <script src="https://code.jquery.com/jquery-
3.6.0.min.js"></script>
    <script src="<?php echo VIEW; ?>/js/grid.js"></script>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-
beta2/dist/css/bootstrap.min.css" rel="stylesheet"
integrity="sha384-
BmbxuPwQa2lc/FVzBcNJ7UAyJxM6wuqIj61tLrc4wSX0szH/Ev+nYRRuWlolflfl"
crossorigin="anonymous">
</head>

```

Для реалізації асинхронного збереження або відтворення інформації був використаний JQuery, який дозволяє робити запити до PHP. На лістингу 4.3 наведено приклад Ajax-запиту на збереження нового користувача з адмін-панелі.

Лістинг 4.3 – Ajax-запит

```
$.ajax({
    url: 'store',
    type: 'post',
    data: { "new_user_data": [first_name, last_name,
email]},
    success: function(response) {
        $('#user_table').html(response);
    }
});
```

Для зміни відображення поточних записів у різному порядку був написаний js-метод для запиту даних з бази відповідно до бажаного типу сортування. На лістингу 4.4 наведено код методу sortBy

Лістинг 4.4 – function sortBy()

```
let way = false;
function sortBy(column) {
    way = !way;
    var data = [column, way];

    $.ajax({
        url: 'sort',
        type: 'post',
        data: { "sort_user": data},
        success: function(response) {
            $('#user_table').html(response);
        }
    });
}
```

Форма реєстрації на захід має як обов'язкові поля, так і необов'язкові. Після введення обов'язкової інформації викликається метод next(). За допомогою інших методів, він перевіряє чи усі поля заповнені, чи дані цих полів є

коректними і, якщо це так, асинхронно виконує збереження інформації, ховає частину форми з обов'язковими полями, показує необов'язкові поля та створює cookie про те, що користувач зареєструвався та заповнив першу частину форми.

Якщо існує cookie про те, що користувач зареєструвався, то з'явиться кнопка для відображення усіх зареєстрованих на захід.

Якщо існує cookie про те, що користувач заповнив першу частину форми, то якщо він перезавантажить сторінку, він повернеться до другої частини форми. Якщо ж він натисне кнопку Finish, cookie реєстрації видалиться.

4.2 PHP розробка

PHP Laravel — це високорівневий фреймворк, який надає потужні засоби для розробки поштових сервісів. Його гнучкість та зручність у використанні дозволяють створювати складні системи електронної пошти з різноманітними функціональними можливостями.

Основні аспекти розробки поштового сервісу у Laravel охоплюють наступні етапи:

Реєстрація користувачів та автентифікація: Фреймворк Laravel має вбудовані засоби для реєстрації користувачів, управління ролями та автентифікації, що дозволяють створювати облікові записи користувачів та забезпечувати безпеку доступу до поштового сервісу.

Відправка електронних листів: Laravel надає зручні функції для відправлення електронних листів через різні провайдери, такі як SMTP або Sendmail. Для цього використовується Mailable клас, який дозволяє створювати та надсилати електронні повідомлення.

Управління повідомленнями: Laravel має можливості для управління повідомленнями, включаючи можливість надсилання сповіщень, роботу з

чергою повідомлень для відправки асинхронних електронних листів, а також реалізацію різних типів сповіщень для користувачів.

На лістингу 4.5 наведено код файлу `parcels-index.blade.php`. Цей файл буде викликаний для відображення головної сторінки управління посилок адміністратором.

Лістинг 4.5 – `parcels-index.blade.php`

```
@extends ('layouts.admin')
@section('content')
    <div class="container">
        <h2>Admin Page
            Post Parcel Details</h2>
        @if (count ($parcels) > 0)
            <table class="table">
                <thead>
                    <tr>
                        <th>ID</th>
                        <th>From</th>
                        <th>To</th>
                        <th>Delivery Option</th>
                        <th>Delivery Status</th>
                        <th>Sender Address</th>
                        <th>Recipient Address</th>
                        <th>Created At</th>
                        <th>Updated At</th>
                        <th>Updated By</th>
                    </tr>
                </thead>
                <tbody>

                    @foreach ($parcels as $parcel)
                        <tr>
                            <td>{{ $parcel->id }}</td>
                            <td>{{ $parcel->sender_name }}</td>
                            <td>{{ $parcel->recipient_name }}</td>
                            <td>{{ $parcel->delivery_option }}</td>
                            <td>{{ $parcel->delivery_status }}</td>
                            <td>{{ $parcel->sender_address }}</td>
                            <td>{{ $parcel->recipient_address }}</td>
                            <td>{{ $parcel->created_at }}</td>
                            <td>{{ $parcel->updated_at }}</td>
                            <td>{{ $parcel->updated_by }}</td>
                        </tr>
                    @endforeach
                </tbody>
            </table>
        @else
```

```

        <p>No post parcels available.</p>
    @endif
</div>
@endsection

```

На лістингу 4.6 наведено код файлу edit-parcel.blade.php. Цей файл використовується для редагування інформації про посилку для адміна.

Лістинг 4.6 – edit-parcel.blade.php

```

@extends ('layouts.app')
@section('content')
    <div class="container">
        <h2>Edit Parcel Details</h2>
        <form method="post"
action="{{route('parcels.update', ['parcel ' => $parcel->id]) }}">
@csrf
            @method('PUT')
            <div class="form-group">
                <label for="to">To: </label>
                <input type="text" class="form-control" id="to"
name="to" value="{{old('to', $parcel->to)}}">
            </div>

            <div class="form-group">
                <label for="delivery option">Delivery
Option:</label>
                <input type="text" class="form-control"
id="delivery option" name="delivery option"
value="{{ old('delivery option', $parcel-
>delivery_option) }}">
            </div>

            <button type="submit" class="btn btn-primary">Update
Parcel</button>
        </form>
    </div>
@endsection

```

Існує контролер, який відповідає за сортування та пошук записів. Його імплементацію наведено на лістингу 4.7.

Лістинг 4.7 – Методи сортування та пошуку

```

public function search(): void
{
    if (array_key_exists(Constants::SEARCH_USER, $_POST))
    {
        $usersData = $this->modelDatabase->searchUser($_POST[Constants::SEARCH_USER]);
        $this->render->view('UserTable', compact('usersData'));
    }
    unset($_POST[Constants::SEARCH_USER]);
}
public function sort(): void
{
    if (array_key_exists(Constants::SORT_USER, $_POST)) {
        $usersData = $this->modelDatabase->sortBy($_POST[Constants::SORT_USER]);
        $this->render->view('UserTable', compact('usersData'));
    }
    unset($_POST[Constants::SORT_USER]);
}

```

Основним компонентом є роутер, який представляє собою клас, відповідальний за реалізацію маршрутизації за допомогою URL. Роутер необхідний не лише для навігації між сторінками, але й для покращення якості коду. Оскільки JQuery не здатний виконати конкретний метод контролера, існують лише два можливих варіанти вирішення цієї проблеми. Це може бути використання роутера (разом з допоміжними класами) або створення об'єкту "ControllerNameHandler", в якому здійснюється ініціалізація потрібного контролера та виклик потрібного методу.

Лістинг 4.8 – клас Router

```

<?php

namespace App\Core;

use Exception;

/**
 * Class Router
 * @package App\Core
 */
class Router
{
    /**
     * Request types
     * @var array[]
     */
    public $routes = [
        'GET' => [],
        'POST' => []
    ];

    /**
     * Return static Router($this)
     * @param $file
     * @return static
     */
    public static function load($file)
    {
        $router = new static();
        require $file;

        return $router;
    }

    /**
     * Set controller as GET requester
     * @param $uri
     * @param $controller
     */
    public function get($uri, $controller): void
    {
        $this->routes['GET'][$uri] = $controller;
    }

    /**
     * Set controller as POST requester
     * @param $uri
     * @param $controller
     */
    public function post($uri, $controller): void
    {
        $this->routes['POST'][$uri] = $controller;
    }
}

```

Для ефективного використання роутера необхідно використовувати клас Request. Один із методів цього класу розділяє отриманий URL та повертає його складові елементи, а інший метод визначає тип запиту (GET або POST). Відповідний код подано на лістингу 4.9.

Лістинг 4.9 – клас Request

```
<?php

namespace App\Core;

/**
 * Class Request
 * @package App\Core
 */
class Request
{
    /**
     * Refactor URI to get after domain's text without
slashes
     * @return string
     */
    public static function uri(): string
    {
        return trim(
            parse_url($_SERVER['REQUEST_URI'],
PHP_URL_PATH), '/'
        );
    }

    /**
     * Return request type
     * @return string
     */
    public static function method(): string
    {
        return $_SERVER['REQUEST_METHOD'];
    }
}
```

Для обробки дій користувача було розроблено контролери, які містять методи, що відповідають за опрацювання кожної окремої опції.

Контролер у програмуванні — це клас, який використовується для обробки різних запитів та взаємодії з іншими компонентами додатку. В Laravel

контролери використовуються для управління логікою додатку, обробки HTTP-запитів, взаємодії з базою даних та визначення того, яке представлення відправити користувачеві.

Контролер важливий для організації логіки програми, роботи з моделями, обробки вхідних запитів та відділення логіки від представлення. У Laravel контролери використовуються для визначення маршрутів, роботи з middleware, взаємодії з моделями та передачі даних у представлення. Контролери полегшують розробку та утримання коду, забезпечуючи чистоту архітектури додатку.

У Laravel контролер - це клас, який відповідає за обробку HTTP-запитів. Він розташовується в директорії `app/Http/Controllers` та містить методи, які обробляють різні дії. Контролери допомагають організувати код, забезпечуючи чітку розділеність між різними аспектами логіки додатку.

В Laravel маршрути визначаються у файлі `routes/web.php` або `routes/api.php`. Контролери також можуть використовувати middleware для виконання певних дій перед або після обробки запиту. Наприклад, `middleware auth` може вимагати, щоб користувач був аутентифікований перед тим, як він отримає доступ до методу контролера. Контролери також можуть використовувати middleware для виконання певних дій перед або після обробки запиту. Наприклад, `middleware auth` може вимагати, щоб користувач був аутентифікований перед тим, як він отримає доступ до методу контролера.

На лістингу 4.10 наведено код методів відображення головної сторінки адмін панелі та редагування посилки.

Лістинг 4.10 – методи контролеру ParcelsController

```

    public function index()
    {
        $parcels = $this->parcelsService->getViewableParcels();
        return view('parcels.index', compact('parcels'));
    }

    public function edit(EditRequest $request)
    {
        $parcel = $this->parcelsService->findParcelById($request->get('id'));
        return view('parcel.edit', compact('parcel'));
    }

    public function store(StoreRequest $request)
    {
        $parcel = $this->parcelsService->findParcelById($request->get('id'));
        $parcel->fill($request->all())->save();
        return view('parcel.edit', compact('parcel'));
    }

```

Кастомний Request в Laravel є спеціальним класом, який ви самостійно визначаєте для обробки та валідації вхідних даних з HTTP-запитів. Головна його мета - це надати вам повний контроль над процесом обробки даних перед тим, як вони потрапляють у ваш додаток.

Один з ключових аспектів кастомних Request - це можливість визначення правил валідації. Ви можете вказати, які поля мають бути обов'язковими, які типи даних приймають, і навіть можливі власні правила для специфічних валідаційних сценаріїв.

Крім того, кастомний Request дозволяє визначати методи для форматування та трансформації даних перед їх обробкою. Це може включати в себе, наприклад, конвертацію рядків у верхній чи нижній регістр, або будь-які інші маніпуляції даними за вашим вибором.

Також, використання кастомних Request може сприяти підвищенню рівня безпеки додатка. Можна визначити фільтри та правила очистки для даних, щоб уникнути атак типу "Injection" та інших потенційних безпекових загроз.

Застосування кастомних Request також спрощує логіку контролерів, особливо коли ви хочете використовувати один і той же набір правил для кількох методів контролера чи навіть різних контролерів.

Щоб зменшити обсяг рядків контролеру та підвищити перевикористання коду, було обрано підхід репозиторіїв та сервісів. Сервіси - це класи, методи яких можуть бути використані різними контролерами, але зберігати бізнес-логіку для кожного виклику. Таким чином нам не треба буде прописувати кожен раз правила та умови доступу до певної інформації.

У програмуванні термін "репозиторій" вказує на шаблон проектування, який використовується для відокремлення логіки доступу до даних від іншої бізнес-логіки. Репозиторії дозволяють створити абстракцію між логікою додатку та джерелами даних, такими як бази даних чи інші зовнішні сервіси.

У Laravel репозиторії часто використовуються для взаємодії з базами даних. Основна ідея полягає в тому, щоб мати окремий клас або набір класів, які відповідають за всі операції з базою даних, і використовувати ці класи замість прямого виклику операцій бази даних в інших частинах програми.

Наприклад, у Laravel можна створити репозиторій для моделі Post, який буде включати методи для отримання всіх постів, отримання конкретного поста за його ідентифікатором, створення нового поста, оновлення та видалення поста. Такий репозиторій буде взаємодіяти з базою даних, але решта коду буде вільно використовувати цей репозиторій, не знаючи деталей його внутрішньої реалізації.

У контексті Laravel репозиторії дозволяють полегшити управління доступом до даних, підвищити читабельність та тестуваність коду.

Деякі ключові факти та аспекти використання репозиторіїв у Laravel:

- відокремлення логіки;
- покращення тестуваності;
- зручніша зміна джерела даних;
- можливості кешування;
- покращення читабельності коду;

– спрощення міграцій.

Репозиторії відокремлюють логіку доступу до даних від решти додатку. Це сприяє підтримці чистоти коду та розділенню відповідальностей між різними компонентами системи.

Завдяки використанню репозиторіїв, тестування стає зручнішим. Можна легко створювати тести, які перевіряють логіку доступу до даних, використовуючи імітації чи фейк-об'єкти.

Якщо ви використовуєте репозиторії, зміна джерела даних (наприклад, перехід з одного типу бази даних на інший) може відбутися без зміни коду в інших частинах програми. Репозиторій виконує роль посередника між додатком та джерелами даних.

Репозиторії можна використовувати для реалізації стратегій кешування даних. Це дозволяє зберігати кеш певних запитань та використовувати його, щоб зменшити час відповіді на повторні запити.

Використання репозиторіїв спрощує читання коду, оскільки операції з базою даних винесені в окремий шар абстракції. Це робить код більш зрозумілим та підтримуваним.

При необхідності змінити структуру бази даних, репозиторії дозволяють провести необхідні міграції з одного джерела даних на інше, не впливаючи на решту додатку.

Загалом, репозиторії в Laravel використовуються для створення чистого та добре структурованого коду, а також для полегшення управління доступом до даних у великих та складних проєктах.

Сервіс в програмуванні — це інструмент для структурування програмного коду, який дозволяє розділити функціонал додатка на окремі частини. У Laravel, сервіс може бути окремим класом або об'єктом, що відповідає за конкретну функціональність.

Цей підхід допомагає зробити код більш зрозумілим та модульним, сприяє легкості розробки та підтримки. Сервіси використовуються для групування

бізнес-логіки та можуть виконувати різноманітні завдання, такі як робота з базою даних, обробка бізнес-логіки, чи інтеграція з іншими сервісами.

Основні переваги використання сервісів у Laravel полягають в їхній здатності розділяти відповідальності між класами, полегшувати перевикористання коду, тестування та створення модульних компонентів для побудови ефективних додатків.

4.3 Результати створеної системи

Було створено систему, що дозволяє відправляти, отримувати та повертати міжнародні відправлення. Візуальне відображення декількох досягнутих сторінок наведено нижче.

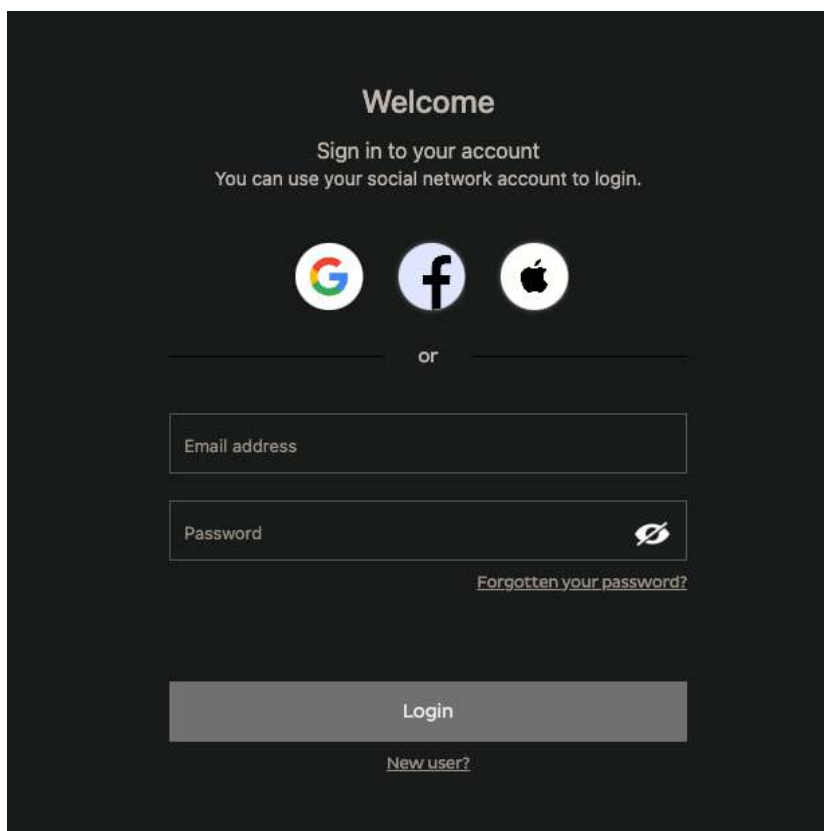


Рисунок 4.1 – Вигляд сторінки авторизації

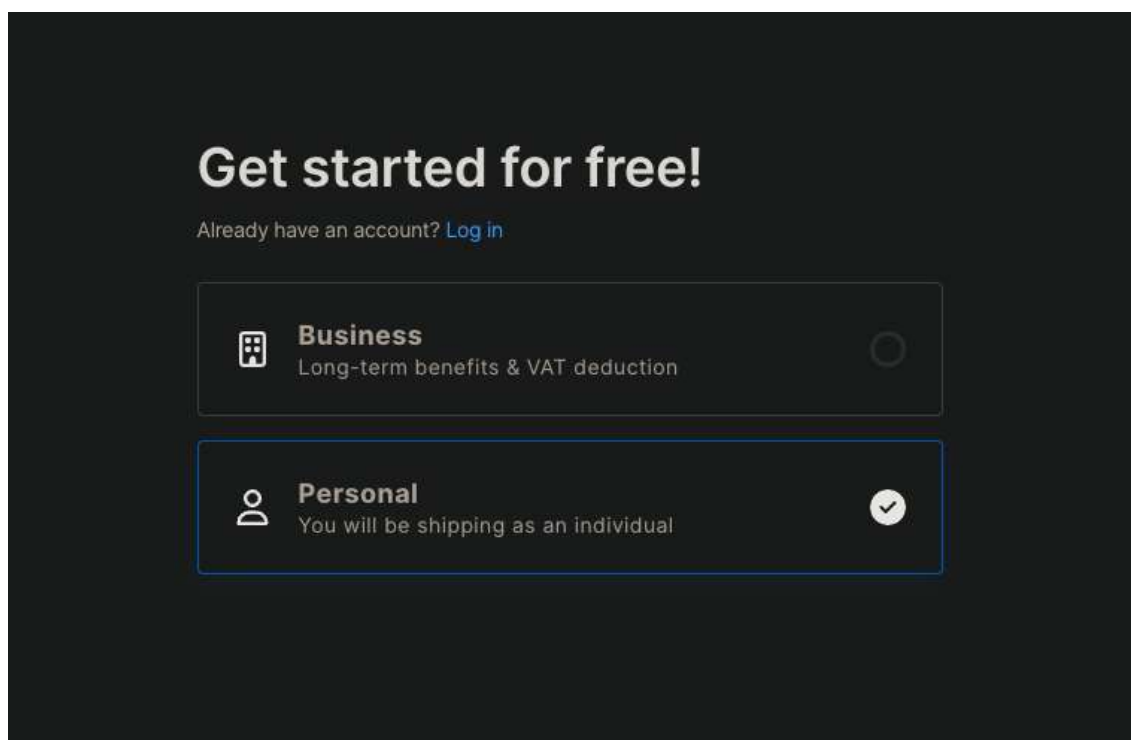


Рисунок 4.2 – Один з кроків реєстрації

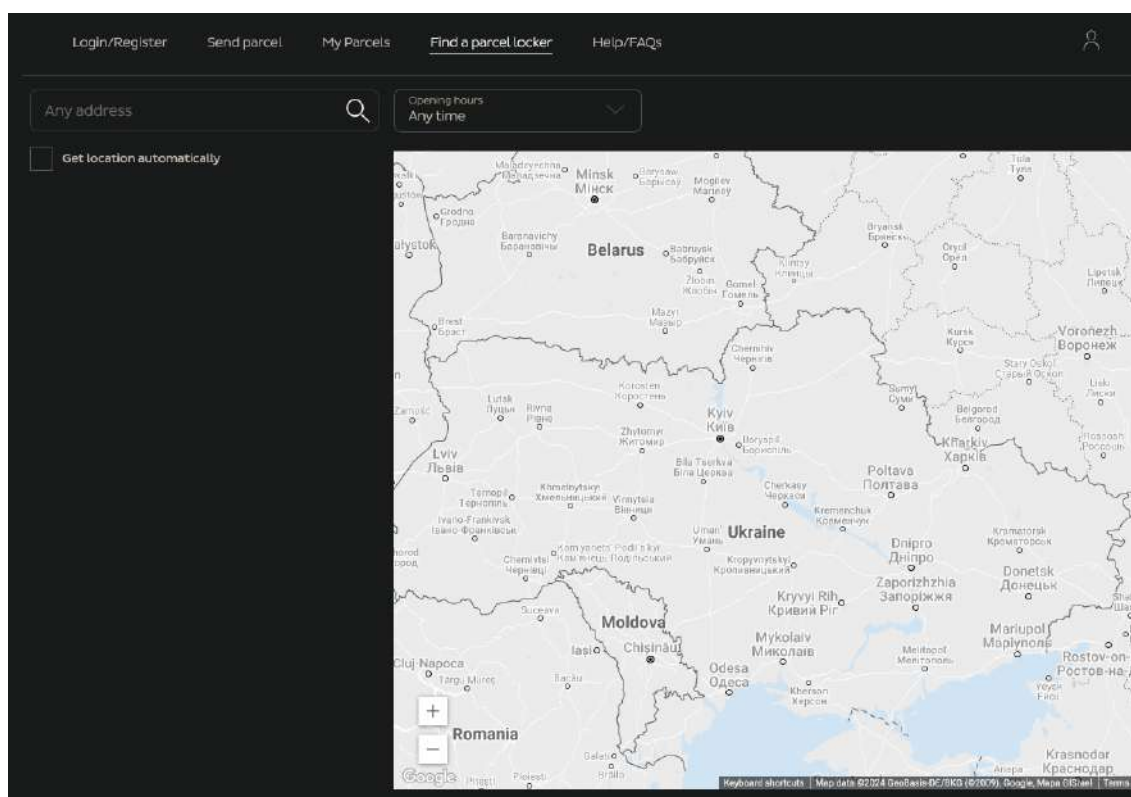


Рисунок 4.3 – Сторінка пошуку найближчої шафи

Orders

Orders list Documents

ORDER NUMBER ↑	NAME & SURNAME ↑	DATE ↑	AMOUNT ↑	PICK-UP DATE ↑	TOTAL PRICE ↑	STATUS ↑
No data						

← Previous page Page 1 of 1 10 rows ↓ Next page →

Рисунок 4.4 – Вигляд сторінки списку замовлень

ВИСНОВКИ

Таким чином, у дипломному проєкті розроблено систему керування посилками через вебсайт, який пропонує зручний доступ до широкого спектру поштових операцій.

Використання PHPStorm дозволяє підвищити як швидкість процесу розробки вебдодатків, так і проведення процесу дебагу.

Під час написання роботи було проаналізовано предметну область, найпоширеніші існуючі вебсайти слугб доставки товарів в Україні і зроблено висновки щодо їх переваг та недоліків.

Також проведено ґрунтовний аналіз і вибір засобів розробки системи. Проведено практичне дослідження та надано його результати щодо вибору мови створення сайту PHP 8.0, спроектовано і реалізовано сайт, створено дизайн та проєктування його бази даних.

За допомогою Laravel, вдалося створити поштовий сервіс, який використовує вбудовані інструменти, такі як Mailables та сервіси, для оптимальної взаємодії з базою даних та іншими компонентами додатка. Під час розробки акцент було зроблено на ефективному використанні можливостей Laravel, униканні зайвого коду та максимальному відокремленні логіки різних частин додатка.

Особлива увага приділялася забезпеченню безпеки поштового сервісу. Використання валідації даних та обробка помилок відправлення допомагали у підвищенні надійності та захищеності сервісу.

На підсумок, Laravel виявився не лише потужним інструментом для розробки, але й відмінним вибором для створення поштового сервісу з великим фокусом на ефективність, безпеку та зручність використання. Реалізація поштового сервісу у Laravel надає значний вплив на функціональність та взаємодію користувачів, підкреслюючи переваги використання даного фреймворку для подібних завдань.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Вступ до jQuery. URL: <http://kytok.org.ua/?p=545>
2. Матеріали офіційного сайту мови програмування PHP. URL: <http://www.php.net/Вступ до jQuery>. URL: <http://kytok.org.ua/?p=545>
3. Документація та розробка MySQL. URL: <https://dev.mysql.com/doc/>
4. PostgreSQL vs MySQL. URL: <https://habr.com/blog/248845/>
5. Порівняння MySQL та PostgreSQL. URL: <https://www.integrate.io/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/>
6. PhpStorm. URL: <https://itpro.ua/product/jetbrains-phpstorm/?tab=description>
7. Hendriyanto, Hendriyanto, and Pandu Adi Cakranegara. "Web-Based Online Sales Information System Using PHP and MYSQL Database in Nara Collection." JMKSP (Jurnal Manajemen, Kepemimpinan, dan Supervisi Pendidikan) 7.1 (2022): 35-52.
8. Jalolov, Tursunbek Sadridinovich. "Programming languages, their types and basics." Technical science research in Uzbekistan 1.5 (2023): P. 145-152.
9. Zammetti, Frank. Modern Full-Stack Development: Using TypeScript, React, Node.js, Webpack, and Docker. Apress, 2020.
10. CHEN, Xiaoming; ZHANG, Jian. The Applications PHP, HTML and MYSQL in Development of Website-Query Function. In: ICMLCA 2021; 2nd International Conference on Machine Learning and Computer Application. VDE, 2021. p. 1-4.
11. JavaScript. Повне керівництво, 7-е видання. Девід Фленаган. 2021. 722 с.
12. Kotenko, N., Zhyrova, T., Chybaievskyi, V., & Desiatko, A. (2019). Дослідження основних тенденцій сучасної розробки веб-сайтів. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 2019. – 6–15 с.

13. Herron, David. Node.js Web Development: Server-side web development made easy with Node 14 using practical examples. Packt Publishing Ltd, 2020.
14. Pantelelis, Michail; Kalloniatis, Christos. Mapping CRUD to Events-Towards an object to event-sourcing framework. In: Proceedings of the 26th Pan-Hellenic Conference on Informatics. 2022. p. 285-289.
15. Tiahunova M.Yu., The system for testing different versions of the PHP / M.Yu. Tiahunova, H.H. Kyrychek, Y.D. Turianskyi // Proceedings of the 3rd Edge Computing Workshop, doors-2023: April 7, 2023 : theses of reports. – Zhytomyr, Ukraine, 2023, – P. 112-129. (Vol-3374).
16. Muqorobin, Muqorobin; Rais, Nendy Akbar Rozaq. Comparison of PHP Programming Language with Codeigniter Framework in Project CRUD. International Journal of Computer and Information System (IJCIS), 2022, 3.3: 94-98.
17. Boras, Marko; Balen, Josip; Vdovjak, Kresimir. Performance evaluation of Linux operating systems. In: 2020 International Conference on Smart Systems and Technologies (SST). IEEE, 2020. p. 115-120.