ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та інформаційних технологій
Кафедра інформаційних систем та технологій

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття ступеня вищої освіти бакалавр

на тему: **«Проєктування інтернет-магазину з продажу автомобілів із використанням фреймворків та бібліотек JavaScript»**

Виконав: здобувач вищої освіти
за освітньою програмою
Інформаційні управляючі системи
спеціальності 126 Інформаційні
системи та технології
ступеня вищої освіти бакалавр
групи 126ІСТ_бз_2022
Голуб Олег Романович
Керівник: Копішинська Олена
Петрівна
Рецензент: Муравльов Володимир
В’ячеславович

Полтава – 2026 року

ПОЛТАВСЬКИЙ ДЕРЖАВНИЙ АГРАРНИЙ УНІВЕРСИТЕТ
Навчально-науковий інститут економіки, управління, права та інформаційних технологій
Кафедра інформаційних систем та технологій

Освітня програма Інформаційні управляючі системи
Спеціальність 126 Інформаційні системи та технології
Рівень вищої освіти перший (бакалаврський)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Юрій УТКІН
«10» квітня 2025 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Голуба Олега Романовича

1. Тема кваліфікаційної роботи:
«Проєктування інтернет-магазину з продажу автомобілів із використанням фреймворків та бібліотек JavaScript»,
Керівник роботи: к. ф.-м. н., доцент, професор кафедри інформаційних систем та технологій Копішинська Олена Петрівна.
Затверджено наказом закладу вищої освіти від «19» березня 2026 року № 282-ст

2. Строк подання здобувачем вищої освіти роботи «26» травня 2026 року.

3. Вихідні дані до роботи: інформаційні ресурси бібліотек JavaScript React https://react.dev/, Redux https://redux.js.org/, редактор вихідного коду https://code.visualstudio.com/, https://www.figma.com/design/, статистичні дані https://mate.academy/

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):

Розділ 1. Теоретичний аналіз технологій розробки вебдодатку із заданим функціоналом
Розділ 2. Формування стеку технологій розроблення інтернет-магазину з продажу автомобілів
Розділ 3. Розробка і тестування інтернет-магазину з продажу автомобілів

5. Перелік графічного матеріалу: схеми, рисунки, діаграми за темою та об’єктом дослідження

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання отримав
Оцінювання економічної ефективності результатів дослідження	Загребельна І. Л., к. е. н., доцент, доцент кафедри економіки та публічного управління	24.03.2026 р.	09.04.2026 р.

7. Дата видачі завдання «10» квітня 2025 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк виконання етапів кваліфікаційної роботи	Примітка
1	Вибір і затвердження теми роботи	01.04.2025 р.	
2	Складання і затвердження розгорнутого плану та завдання на кваліфікаційну роботу	02.04.2025 р. – 10.04.2025 р.	
3	Опрацювання джерел інформації	03.06.2025 р. – 15.06.2025 р.	
4	Збір, вивчення і обробка інформації, необхідної для виконання роботи	16.06.2025 р. – 04.07.2025 р.	
5	Виконання теоретико-методологічного розділу роботи	08.09.2025 р. – 15.11.2025 р.	
6	Виконання дослідницько-аналітичного розділу роботи	16.11.2025 р. – 30.01.2026 р.	
7	Виконання проєктно-рекомендаційного розділу роботи	01.02.2026 р. – 21.03.2026 р.	
8	Оцінювання економічної ефективності результатів дослідження	24.03.2026 р. – 09.04.2026 р.	
9	Оформлення тексту роботи	10.04.2026 р. – 25.05.2026 р.	
10	Попередній захист роботи на кафедрі	26.05.2026 р.	
11	Доопрацювання роботи з урахуванням зауважень і пропозицій	27.05.2026 р. - 28.05.2026 р.	
12	Нормоконтроль	29.05.2026 р.	
13	Захист кваліфікаційної роботи	03.06.2026 р.	

Здобувач вищої освіти **Олег ГОЛУБ**

Керівник роботи **Олена КОППШИНСЬКА**

ЗМІСТ

ВСТУП 6

РОЗДІЛ 1 ТЕОРЕТИЧНИЙ АНАЛІЗ ТЕХНОЛОГІЙ РОЗРОБКИ ВЕБДОДАТКУ ІЗ ЗАДАНИМ ФУНКЦІОНАЛОМ 8

1.1 Огляд існуючих підходів до розроблення сучасних вебсайтів 8

1.2 Базові технології розроблення клієнтської частини вебдодатку 10

1.3 Базові технології розроблення серверної частини вебдодатку 19

РОЗДІЛ 2 ФОРМУВАННЯ СТЕКУ ТЕХНОЛОГІЙ РОЗРОБЛЕННЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ АВТОМОБІЛІВ 23

2.1 Постановка задачі. Оцінка складності розробки 23

2.2 Підбір технологій для розробки вебсайту 25

РОЗДІЛ 3 РОЗРОБКА і ТЕСТУВАННЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ АВТОМОБІЛІВ 31

3.1 Програмна реалізація основних функцій інтернет-магазину 31

3.2 Інтеграції зовнішніх сервісів за допомогою API 37

3.3 Тестування та реліз вебсайту 39

3.4 Оцінювання економічної та технічної ефективності вебсайту 45

ВИСНОВКИ 49

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 51

ДОДАТКИ 55

ВСТУП

Актуальність розробки інтернет-магазину з продажу автомобілів в Україні пояснюється стрімкою цифровізацією економіки та зміною поведінки споживачів, які дедалі частіше надають переваги онлайн-платформам для пошуку, порівняння та купівлі товарів, у тому числі таких дороговартісних, як автомобілі. Розвиток електронної комерції, забезпечення стабільного інтернет-покриття та доступність мобільних пристроїв створюють сприятливі умови для переходу авторинку в онлайн-середовище. Це вигідно і продавцям, і покупцям, адже традиційні способи купівлі автомобілів часто супроводжуються значними витратами часу, обмеженим вибором та недостатньою прозорістю інформації, що підсилює запит на зручні цифрові рішення. Мова йде про середній сегмент ринку автомобілів, а також уживаних автомобілів, де цінується зручність, гнучка цінова політика, можливість вибору і порівняння рішень. Елітні салони швидше використовують інформаційно-комерційні вебсайти рекламного характеру.

Додатковим аргументом щодо актуальності розробки є потреба у створенні ефективних, безпечних та інформативних вебплатформ, які забезпечують повний цикл взаємодії з клієнтом – від підбору автомобіля до оформлення покупки. В умовах економічної нестабільності економіки в Україні інтернет-магазини дозволяють бізнесу оптимізувати витрати, розширити географію продажів і підвищити рівень довіри клієнтів за рахунок відкритості даних, відгуків та аналітики. Таким чином, розробка сучасного вебзастосунку для продажу автомобілів є не лише актуальним, але й стратегічним напрямом розвитку електронної комерції в Україні.

Метою кваліфікаційної роботи є розробка функціонального інтернет-магазину з продажу автомобілів із використанням сучасних вебтехнологій.

Завдання кваліфікаційної роботи включає:

- дослідження потреб ринку з продажів автомобілів;
- огляд існуючих комерційних вебсайтів з продажів автомобілів;

– вибір технологій та розроблення функціонального вебсайту зі зручним інтерфейсом для покупців і продавців автомобілів;

– забезпечення безпеки та захисту персональних даних клієнтів, а також тестування готового інтернет-магазину.

Об'єкт дослідження – розроблення інтернет-магазину з продажу автомобілів у формі SPA-додатку.

Предмет дослідження – особливості використання класичних базових вебтехнологій HTML&CSS, програмних бібліотек JavaScript при створенні інтернет-магазину, зокрема React, NodeJS та Styled Component.

Методи дослідження: інформаційно-пошуковий, аналітичний, емпіричний, порівняльний, прикладного програмування, графічний.

Інформаційна база кваліфікаційної роботи: навчально-наукові ресурси мережі інтернет, редактори програмного коду, бібліотеки JavaScript, порівняльні характеристики існуючих інтернет-магазинів з продажу автомобілів, статистичні дані про ринок послуг https://jobs.dou.ua/.

Результати роботи були перевірені шляхом тестування сайту та здійснення пробних замовлень, а також пройшли апробацію у вигляді публікації тез доповідей на студентській науковій конференції.

Практична значущість роботи: сайт, що був розроблений в рамках проєкту, готовий до хостингу та масштабування, включає в себе каталог автомобілів з різними параметрами та можливість фільтрування за ключовими характеристиками (ціною, маркою, роком випуску та іншими). Також вебсайт надає можливість оформлення замовлення, спілкування з менеджерами та отримання детальної інформації про автомобілі.

Робота складається зі вступу, трьох розділів, висновків та списку літератури (39 найменувань). Обсяг роботи становить 50 сторінок, включає 18 рисунків, 3 таблиці.

РОЗДІЛ 1
ТЕОРЕТИЧНИЙ АНАЛІЗ ТЕХНОЛОГІЙ РОЗРОБКИ ВЕБДОДАТКУ ІЗ ЗАДАНИМ ФУНКЦІОНАЛОМ

1.1 Огляд існуючих підходів до розроблення сучасних вебсайтів

Вебсайт – це колекція вебсторінок та пов'язаних з ними файлів, які розташовані на вебсервері та доступні через інтернет. Вебсайт може містити різноманітний контент, такий як текст, зображення, відео, аудіо, графіку та інші елементи, які відображаються в браузері користувача [1]. Вебсайти використовуються для різних цілей, включаючи розповсюдження інформації, комунікацію, електронну торгівлю, рекламу, навчання та розваги. Вони можуть мати різну структуру та функціональність, включаючи статичні сторінки, динамічний контент, форми взаємодії з користувачем, бази даних, інтерактивні елементи та інше. Розвиток технологій створення вебсайтів є динамічним процесом, що відображає загальні тенденції еволюції інформаційних технологій [2]. Веброзробка пройшла шлях від простих статичних сторінок до складних інтерактивних вебзастосунків, які за функціональністю не поступаються традиційному програмному забезпеченню. Основні етапи розвитку можна розглядати через призму змін у підходах до реалізації Front-End і Back-End частин вебдодатків, а також через появу нових інструментів і архітектурних рішень [3].

Поява розділених технологій для інтерфейсу та функціоналу виникла від потреби в розподілі ролей та вдосконаленні процесу розробки вебдодатків. Перші вебсайти були простими HTML-сторінками, де весь код для відображення і функціональності знаходився в одному файлі.

З плином часу вебсайти стали складнішими та більш вимогливими щодо функціональності та інтерактивності. Розділений підхід виник як відповідь на цю складність, щоб розділити завдання та обов'язки між розробниками і дизайнерами.

Розділена технологія передбачає розбиття Front-End (інтерфейсу) та Back-End (функціоналу) на окремі складові частини. Front-End відповідає за відображення та інтерактивність користувача, тоді як Back-End займається обробкою даних, виконанням бізнес-логіки та взаємодіє з базою даних.

Front-End частина складається з HTML, CSS та JavaScript, і ці технології використовуються для відображення та взаємодії з користувачем. HTML відповідає за структуру сторінки, CSS – за її вигляд, а JavaScript – з а динамічні ефекти та взаємодію з користувачем [4]. Крім того, для покращення роботи з CSS можуть використовуватися різні препроцесори, такі як Sass, Less або Stylus .

Back-End частина складається з мов програмування, баз даних та серверного програмного забезпечення. Найпоширеніші мови програмування для Back-End - це Java, Python, Ruby, PHP та JavaScript (з використанням Node.js). Бази даних використовуються для збереження даних та їхньої організації. Серверне програмне забезпечення, таке як Apache, Nginx або IIS, відповідає за обробку запитів від клієнтів та відправку відповідей.

Загалом, структура сучасних вебсайтів може бути досить складною і вимагати від розробників знань та досвіду у різних технологіях. Проте, використання правильного стеку технологій та добре спроектованої архітектури може допомогти створити потужну та ефективну вебплатформу.

Історія створення вебсайтів розпочалася у 1990-х роках з простих HTML-сторінок без будь-яких динамічних елементів. Згодом почали використовуватися скрипти на JavaScript для валідації форм та реалізації простих ефектів, таких як розкриття і приховування блоків ті ін.

У 2000-х роках з’явилися технології, такі як CSS та AJAX, які дозволили створювати більш складні інтерфейси користувача та динамічно змінювати вміст сторінки без перезавантаження [5]. Також з’явилася можливість створювати вебдодатки з використанням серверної логіки та баз даних, що дозволяє зберігати та обробляти інформацію з більшою ефективністю та безпекою.

Сучасні вебсайти складаються з Front-End та Back-End, які розробляються окремо та з’єднуються за допомогою API. Це дозволяє забезпечити більшу

модульність та масштабованість вебдодатків, а також розподіляти завдання між розробниками та забезпечувати більш високу швидкість розробки.

1.2 Базові технології розроблення клієнтської частини вебдодатку

Клієнтська частина, або Front-end – це частина веброзробки, яка займається розробкою клієнтської (браузерної) частини вебдодатку або сайту. Front-end розробники відповідають за створення інтерфейсу користувача (UI) з використанням HTML, CSS та JavaScript.

HTML (HyperText Markup Language) це мова розмітки гіпертексту – найбазовіший будівельний блок інтернету. Він визначає значення та структуру веб-контенту. Використовуючи HTML, можна визначати різні елементи вебсторінки, такі як заголовки, абзаци, списки, таблиці, зображення та інші, а також встановлювати зв’язки між вебсторінками за допомогою гіперпосилань.

HTML-елемент відокремлюється від іншого тексту в документі за допомогою «тегів», які складаються з назви елемента, оточеної символами <teg>. Назва елемента всередині тегу не враховує регістр. Тобто її можна писати у верхньому регістрі, нижньому регістрі або їх комбінацією. Наприклад, тег <title> можна писати як <Title> або <TITLE>. Однак, загальноприйнятою та рекомендованою практикою є написання тегів у нижньому регістрі [6]. Тег <html> представляє корінь (елемент верхнього рівня) HTML-документа, тому його також називають кореневим елементом. Усі інші елементи повинні бути нащадками цього елемента, як показано на рис. 1.1.

HTML є основним компонентом вебсторінок та є стандартом в інтернеті. Більшість браузерів підтримують HTML, що дозволяє відображати вебсторінки з мінімальними відмінностями на різних пристроях та операційних системах [7]. HTML є однією з базових мов веброзробки та часто використовується в поєднанні з CSS та JavaScript для створення більш складних та інтерактивних вебсайтів.

```
<!doctype html>
<html lang="en-US">
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>My test page</title>
  </head>
  <body>
    <img src="" alt="My test image" />
  </body>
</html>
```

Рисунок 1.1 – Приклад запису базової структури HTML-документу

CSS (Cascading Style Sheets) – це мова опису стилів, яка використовується для задання зовнішнього вигляду вебсторінок. CSS дозволяє відділити структуру та зміст вебсторінок від їхнього вигляду та оформлення. Завдяки CSS можна відокремити дизайн вебсторінок від коду, що дозволяє зберігати код сторінок більш чистим і легким для редагування. CSS дозволяє задавати такі стилі, як кольори, фон, шрифти, відступи, рамки, анімації та багато іншого. Використовуючи CSS, можна значно поліпшити зовнішній вигляд вебсторінок та зробити їх більш привабливими та зрозумілими для користувачів [8].

JavaScript – це мова програмування, яка використовується для створення динамічного та інтерактивного контенту на вебсторінках. JavaScript можна використовувати для зміни структури та вмісту сторінки, а також для створення анімації, валідації форм, перевірки даних, інтерактивних ефектів та багатьох інших функцій. JavaScript є однією з основних технологій, що використовуються на сучасних вебсайтах. Вона може бути виконана на бічному (клієнтському) бо на серверному боці за допомогою різних фреймворків та бібліотек [9].

З часом все більше і більше з’являлось технологій, однак вони всі базувалися на 3 технологіях: HTML, CSS, JavaScript. Зараз можна назвати дуже багато технологій для прискорення розробки, наприклад для покращення розробки з використанням CSS можна використовувати CSS-препроцесори, такі

як Sass, Less, Stylus, які дозволяють писати CSS код більш організовано та ефективно. З їх допомогою можна використовувати змінні, міксіни, функції та інші корисні функції для розробки CSS.

Один із сучасниїх етапів розвитку вебтехнологій пов’язаний із переходом до повноцінних клієнтських застосунків, відомих як Single Page Application (SPA). Завдяки розвитку потужних JavaScript-фреймворків, таких як Angular, React та Vue.js, значна частина логіки почала переноситися на клієнтську сторону [10]. Front-End став самостійним рівнем архітектури, який відповідає не лише за відображення даних, а й за управління станом застосунку, маршрутизацію та взаємодію з API. У цей період набуває популярності архітектура REST, яка забезпечує стандартизовану взаємодію між клієнтом і сервером. Back-End, у свою чергу, трансформується у набір сервісів, що надають дані через API.

React – це бібліотека JavaScript для створення інтерфейсів користувача. Вона дозволяє розробникам створювати вебсайти та вебдодатки, які можуть швидко та ефективно оновлюватися без перезавантаження сторінки. React був розроблений компанією Facebook і випущений в 2013 р.

React базується на концепції компонентів, що дозволяє розробникам створювати окремі блоки коду, які можуть бути використані повторно в різних частинах вебдодатку. Це робить код більш організованим та легким для зберігання.

React також використовує віртуальну DOM (Document Object Model), що дозволяє зменшити кількість змін, які необхідно робити безпосередньо в реальному DOM, що забезпечує швидку та ефективну роботу вебдодатку. Розглянемо більш детально переваги React над іншими бібілотеками, які узагальнені спільното веброзробників.

1. Єдина «парасолька». React є лише бібліотекою для відображення інтерфейсу, а не повноцінним фреймворком. Це означає, що можливо використовувати React разом з будь-яким іншим інструментом або бібліотекою, що надає більшу гнучкість та контроль над вашим проектом.

2. Висока продуктивність. React використовує віртуальну DOM (Document Object Model), що дозволяє реалізувати швидкі та ефективні переробки елементів візуального інтерфейсу. Це дозволяє досягнути високої продуктивності вебдодатків, навіть при великій кількості даних.

3. Модульність. React дозволяє розбити ваш інтерфейс на незалежні компоненти, що підвищує його модульність. Це означає, що можна повторно використовувати код компонентів, що дозволяє зменшити час розробки.

4. React має просту та логічну структуру, що дозволяє легко навчитися його використанню. Більшість розробників, які знайомі з JavaScript, можуть швидко вивчити React.

5. React має велику та активну спільноту розробників, яка забезпечує підтримку, допомогу та постійне вдосконалення бібліотеки.

6. Гнучкість. React дозволяє змінювати та розширювати його функціонал за допомогою сторонніх бібліотек та плагінів. Це дозволяє збільшити функціональність вашого проекту та забезпечити більшу гнучкість в розробці.

7. Розробка мобільних додатків. React Native – це фреймворк для розробки мобільних додатків з використанням React. Він дозволяє розробляти нативні додатки для iOS та Android, використовуючи звичний для розробників React підхід до розробки інтерфейсу. Це забезпечує зменшення часу розробки та спільний код для різних платформ.

Загалом, варто відзначити, що React є дуже популярною технологією, найчастіше використовується для проєктів e-commerce, розробки функціональних маркетплесів, які потребують обробки великої кількості даних, забезпечують інтерактивіні інтерфейси [11]. В літературі та на відкритих майданчиках часом ведуться дискусії щодо точного віднесення React до бібліотек чи фреймворків.

Vue.js – це прогресивний фреймворк для створення користувацьких інтерфейсів вебдодатків. Він дозволяє розробляти високоякісні вебдодатки з високою продуктивністю та ефективністю. Основні переваги Vue.js порівняно з іншими фреймворками [12]:

1. Легкість вивчення та використання: Vue.js має просту та зрозумілу структуру, що дозволяє легко вивчити його використання. Також він має документацію, яка дозволяє швидко знайти рішення для більшості проблем.

2. Простота інтеграції: Vue.js можна легко інтегрувати з будь-яким іншим інструментом або бібліотекою, що забезпечує більшу гнучкість та контроль над вашим проектом.

3. Модульність: Vue.js дозволяє розбити ваш інтерфейс на незалежні компоненти, що підвищує його модульність. Це означає, що в процесі роботи можна повторно використовувати код подібних компонентів, що дозволяє зменшити час розробки та покращити читабельність, чистоту, оптимальність коду, уникнути надмірнсті за рахунок універсалізації.

4. Висока продуктивність: Vue.js використовує віртуальну DOM, що дозволяє забезпечити високу продуктивність та ефективність роботи вебдодатків.

5. Активна спільнота розробників: Vue.js має велику та активну спільноту розробників, яка забезпечує підтримку, допомогу та постійне вдосконалення фреймворку. Це означає, що можливо швидко знайти відповіді на свої запитання та знайти відповідні рішення для вашого проекту.

Angular – це повноцінний фреймворк для розробки вебдодатків, розроблений і підтримуваний компанією Google. Він базується на мові програмування TypeScript і має вбудовану підтримку компонентної архітектури [13]. Переваги Angular порівняно з іншими фреймворками:

1. Широкі можливості: Angular має вбудовану підтримку багатьох функцій, таких як маршрутизація, форми, аутентифікація, перевірка стану і багато іншого. Це дозволяє створювати повноцінні вебдодатки без додаткових зусиль.

2. Строга типізація: TypeScript забезпечує строгу типізацію, що допомагає зменшити кількість помилок в ході розробки та полегшує роботу з кодом.

3. Компонентна архітектура: Angular побудований на компонентній архітектурі, що дозволяє розбити додаток на незалежні компоненти, що

підвищує його модульність та забезпечує можливість повторного використання коду.

4. Оптимізація продуктивності: Angular використовує віртуальну DOM, що дозволяє досягнути високої продуктивності вебдодатків, навіть при великій кількості даних.

5. Підтримка компанією Google: Angular є фреймворком, який розробляється і підтримується компанією Google, що забезпечує його стабільність, підтримку та постійний розвиток.

6. Можливості тестування: Angular має вбудовану підтримку для тестування, що дозволяє легко виявляти та виправляти помилки в ході розробки вебдодатку.

7. Широке співтовариство: Angular має велике та активне співтовариство, що забезпечує наявність безлічі різноманітних додатків, бібліотек, пакетів та рішень для різних задач.

Загалом, Angular є потужним фреймворком з багатьма перевагами порівняно з іншими фреймворками. Його вбудовані можливості, строга типізація, компонентна архітектура, оптимізація продуктивності, підтримка Google, можливості тестування та активне співтовариство роблять Angular популярним вибором для розробників вебдодатків. Однак, використання Angular потребує певного рівня володіння TypeScript та інших технологій, що може бути викликом для початківців.

Окрім наведених вище засобів, для покращення розробки JavaScript можна використовувати інші фреймворки і бібліотеки, такі як jQuery, Lodash, Moment.js та інші, які дозволяють працювати з DOM, маніпулювати даними та працювати зі строками та датами [14]. Окремі фреймворки стали золоти мстандартом при написанні інтерактивних, динамічних елементів вебінтерфейсу.

Для організації та керування структурою вебдодатку можна використовувати різні інструменти, такі як Webpack, Gulp, Grunt, Parcel, які дозволяють автоматизувати процес збірки та оптимізації вебдодатків. Окрім названих, для розробки вебдодатків можна використовувати різноманітні

інструменти тестування, які дозволяють перевіряти працездатність та функціональність додатку, такі як Jest, Mocha, Chai та інші.

UX/UI (User Experience/User Interface) – це скорочення, яке поєднує два поняття, що відносяться до процесу розробки програмного забезпечення та вебдизайну. UX зосереджений на взаємодії користувача з продуктом і його відчуттями під час цієї взаємодії, тоді як UI стосується зовнішнього вигляду продукту та його елементів [15].

UX/UI дизайнер – це креативний фахівець, який проектує призначені для користувача інтерфейси. UX і UI - це дві різних сторони дизайну, але частіше за все завдання по обох напрямках тісно пов’язані між собою, а тому їх робить один універсальний фахівець [16].

Окрім зазначених вище технологій, варто окреслити й інші засоби, які потрібно знати UX/UI дизайнеру і якими професійно володіти:

1. Figma (дизайн інтерфейсу з можливістю колективної роботи)
2. Adobe Photoshop і Illustrator (додатковий функціонал)
3. Adobe XD (створення прототипів, дизайн інтерфейсу)
4. Invision App (створення прототипів з можливістю колективної роботи)
5. Balsamiq (створення макетів)
6. RedPen (колективний задачник)
7. Notion (менеджер завдань)
8. Framer (дизайн інтерактивних мобільних інтерфейсів)

Основні правила UX/UI дизайну є керівними принципами, які сприяють створенню високоякісного дизайну та покращенню взаємодії користувача з продуктом [17]. Розглянемо кожне з правил детальніше:

– Врахування потреб та очікувань користувачів: дизайн повинен бути спрямований на задоволення потреб і вимог цільової аудиторії. Для цього необхідно провести дослідження користувачів, зрозуміти їхні цілі, бажання і очікування від продукту.

– Створення зручного та інтуїтивно зрозумілого інтерфейсу: інтерфейс повинен бути логічним, простим у використанні і відповідати ментальним

моделям користувачів. Елементи управління, функції та контент мають бути розташовані таким чином, щоб користувачі могли легко розуміти, як вони працюють і як з ними взаємодіяти.

– Забезпечення ефективного взаємодії користувача з продуктом: дизайн має сприяти безпроблемній та ефективній взаємодії між користувачем і продуктом. Це може бути досягнуто шляхом розумного розташування елементів, використання зрозумілих символів та мікроінтеракцій, які надають зворотний зв’язок користувачу під час його взаємодії з інтерфейсом.

– Зменшення кількості кроків, необхідних для виконання завдання: мінімізація кількості кроків та обмеження зусиль, які необхідно зробити користувачеві для досягнення своєї мети, є важливим аспектом UX/UI дизайну. Це може включати скорочення кількості полів для заповнення у формах, використання прогресивної дисклозії для поступового відкриття інформації та інші методи спрощення взаємодії.

– Використання чітких та зрозумілих елементів управління: елементи управління, такі як кнопки, посилання, меню, повинні бути легкими для сприйняття та натискання користувачем. Їхні функції мають бути очевидними та консистентними на всіх сторінках продукту.

– Забезпечення однорідного дизайну та навігації на всіх сторінках продукту: Чітка структура і навігація допомагають користувачам легко орієнтуватися в продукті та знаходити потрібну інформацію. Використання однакових стилів, кольорів, шрифтів та інших елементів дизайну на всіх сторінках сприяє єдності і забезпечує зручну навігацію.

– Тестування та збір фідбеку від користувачів для постійного вдосконалення: тестування продукту з реальними користувачами та збір їхнього фідбеку дозволяє виявити слабкі місця і поліпшити дизайн. Важливо залучати користувачів у процес розробки, щоб забезпечити, що їхні потреби та вимоги враховані.

Загалом, UX/UI дизайн має на меті створити продукт, який є зручним, привабливим та легким у використанні для користувачів. Дотримання цих

правил допомагає покращити якість взаємодії та задоволення користувачів, що в свою чергу сприяє успіху продукту на ринку [17].

Співвідношення кольорів в UX/UI дизайні має велике значення для створення настрою та передачі інформації. Основні принципи вибору кольорів включають:

Слідування колірній гармонії, яка включає в себе використання кольорових комбінацій, які взаємодіють між собою гармонійно. Наприклад, використання аналогічних кольорів створює спокійну і збалансовану атмосферу. Комплементарні кольори (кольори, які знаходяться протилежні один одному на колірному колесі) можуть створювати контраст і виразності. Такі відповідність та контрастність між кольорами допомагають покращити сприйняття та розуміння інформації [18].

Контекст також важливий при виборі кольорової схеми. Наприклад, якщо продукт пов’язаний з певним брендом, варто враховувати його кольори та стиль для створення єдності і розпізнаваності. Також важливо враховувати цільову аудиторію та її вподобання щодо кольорів. Деякі культурні аспекти також можуть впливати на сприйняття та значення кольорів, тому важливо бути свідомим цих нюансів при виборі кольорової палітри.

Узгоджений вибір кольорів у UX/UI дизайні сприяє створенню гармонійного, привабливого та зрозумілого інтерфейсу, який відповідає потребам та очікуванням користувачів.

Крім того, UX/UI дизайн включає такі елементи, як інформаційна архітектура, принципи навігації, типографіка, анімація, розташування елементів на сторінці та відступи. Всі ці елементи повинні працювати разом для створення приємного, функціонального та заохочуючого досвіду для користувачів [19].

Навчання користувачів, збір та аналіз даних, тестування продукту та вдосконалення на основі фідбеку користувачів також важливі складові процесу UX/UI дизайну. Дизайнери UX/UI постійно працюють над удосконаленням продуктів, забезпечуючи максимальну задоволеність користувачів та досягаючи бізнес-цілей компанії.

1.3 Базові технології розроблення серверної частини вебдодатку

Серверна частина сайту, або вебдодатку – це програмне забезпечення, яке виконується на сервері та забезпечує обробку запитів користувачів, доступ до баз даних, зберігання та надсилання даних до клієнтської частини сайту.

Технології, що використовуються для розробки серверної частини, можуть бути різними залежно від мов програмування та фреймворків. Деякі з найпопулярніших технологій для розробки серверної частини сайту [20]:

1. Node.js: платформа, яка дозволяє використовувати JavaScript для розробки серверної частини сайту [21]. Вона базується на двигуні JavaScript V8 від Google та надає розробникам можливість розробляти серверні додатки, API та інші рішення.

2. PHP: мова програмування, яка використовується для розробки вебдодатків та серверних додатків. Вона має багато фреймворків та інструментів для розробки серверної частини сайту.

3. Java: мова програмування, яка часто використовується для розробки вебдодатків. Java має різноманітні фреймворки, такі як Spring, Hibernate та Struts, що дозволяють швидко та ефективно розробляти серверну частину.

4. Ruby on Rails: фреймворк, який дозволяє швидко створювати вебдодатки. Він використовує мову програмування Ruby та базується на парадигмі MVC (Model-View-Controller).

5. Django: фреймворк, який використовує мову програмування Python та дозволяє швидко розробляти серверну частину сайту. Django має вбудовану адміністративну панель та багато інших функцій, що спрощують розробку вебдодатків.

Це лише кілька з технологій, які використовуються для розробки серверної частини сайту. Вибір технології залежить від конкретного проєкту та вимог до нього. Наприклад, Node.js зазвичай використовують для розробки додатків, які мають велику кількість взаємодій з користувачами та потребують високої швидкодії обробки запитів, PHP – для розробки більш традиційних вебсайтів з

більш простими функціями, а Java та Ruby on Rails використовують для розробки великих та складних вебдодатків. Розглянемо ближче Node.js, тому що вона є дійсно однією з найпопулярніших технологій для розробки серверної частини.

Ось декілька переваг Node.js:

1. Використання JavaScript для розробки серверної частини дозволяє використовувати одну мову програмування для клієнтської та серверної частини.

2. Висока швидкість: Node.js використовує двигун JavaScript V8 від Google, що забезпечує високу швидкість виконання коду.

3. Асинхронний ввід/вивід: Node.js забезпечує асинхронну обробку ввідно-вивідних операцій, що дозволяє зменшити час очікування та покращити продуктивність додатків.

4. Масштабованість: Node.js забезпечує легку масштабованість додатків за рахунок можливості розподіленої обробки запитів.

5. Велика кількість модулів: Node.js має велику кількість модулів та бібліотек, які дозволяють розробникам швидко та ефективно створювати різноманітні додатки.

6. Open-source: Node.js є відкритим програмним забезпеченням, що дозволяє розробникам використовувати його безкоштовно та долучатися до розвитку платформи.

7. Компанії, які використовують Node.js: Node.js використовують багато відомих компаній, таких як Netflix, LinkedIn, PayPal та багато інших, що свідчить про популярність та ефективність цієї технології.

8. Екосистема NPM: Node.js має потужну систему пакетів та модулів, яка називається NPM (Node Package Manager). Це дозволяє розробникам швидко знаходити, встановлювати та використовувати сторонні модулі.

Також однією з популярних Front-End технологій називають TypeScript, який по суті є розширенням мови JavaScript, що дозволяє додавати типи до звичайного JavaScript коду, допомагає виявляти помилки в процесі компіляції, забезпечує більшу безпеку та надійність в коді та полегшує розробку більших та складних проєктів.

TypeScript за призначенням – це мова програмування, яка розширює JavaScript, дозволяючи розробникам використовувати строгу типізацію, класи, інтерфейси та інші функції [22]. Приклад типізації в TypeScript показано на рисунку 1.2.

```
1   type Status = 'CDD' | 'CDI' | 'Stagiaire';
2
3   interface Employee {
4       firstName: string;
5       lastName: string;
6       birthdate: Date;
7       status: Status;
8   }
9
10  interface Ubidreams {
11      employees: Employee[];
12      turnover: number;
13  }
```

Рисунок 1.2 – Приклад типізації TypeScript

Переваги TypeScript [23]:

1. Строга типізація: TypeScript дозволяє використовувати строгу типізацію, що забезпечує більшу безпеку в ході розробки та дозволяє виявляти помилки в коді на ранніх етапах.

2. TypeScript підтримує об’єктно-орієнтоване програмування (ООП), що дозволяє розробникам використовувати класи, інтерфейси, наслідування та інші ООП-концепції для покращення організації коду.

3. TypeScript дозволяє використовувати нові функції, які не підтримуються в чистому JavaScript, такі як перевантаження функцій, декоратори та інші.

4. Підтримка від IDE: TypeScript дозволяє підтримку від різних редакторів коду, таких як Visual Studio, Visual Studio Code, Sublime Text, Atom, WebStorm та інші.

5. Краща читабельність коду: TypeScript дозволяє додавати анотації типів, що підвищує читабельність коду та дозволяє іншим розробникам краще зрозуміти код, який вони пишуть.

6. Підтримка різних фреймворків: TypeScript може бути використаний з різними фреймворками, такими як Angular, React, Vue та інші, що забезпечує більшу гнучкість при розробці вебдодатків.

7. Багатий набір інструментів та бібліотек: TypeScript має широкий вибір інструментів та бібліотек, які допомагають розробникам полегшити роботу та покращити продуктивність, такі як TypeScript Compiler (tsc), ts-node, TypeORM, NestJS, і багато інших.

TypeScript покращує читабельність коду і може бути використаний з різними фреймворками, роблячи його гнучким варіантом для розробки вебдодатків. Використання TypeScript може покращити безпеку та ефективність розробки проектів, забезпечуючи більшу надійність та зручність у процесі програмування.

РОЗДІЛ 2
ФОРМУВАННЯ СТЕКУ ТЕХНОЛОГІЙ РОЗРОБЛЕННЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ АВТОМОБІЛІВ

2.1 Постановка задачі. Оцінка складності розробки

Основним завданням, яке потрібно вирішити в процесі виконання кваліфікаційної роботи, було розроблення вебсайту для пошуку та розміщення оголошень про продаж автомобілів. SPA (Single Page Application) є оптимальним вибором для розробки вебсайту з пошуку і розміщення оголошень про продаж автомобілів з урахуванням того, що така модель забезпечує більш ефективну навігацію, покращує продуктивність, кешування даних, може забезпечити більш плавну взаємодію користувача [24].

Згідно поставленого завдання та ідеї проєкту інтернет-магазин з продажу автомобілів повинен надавати користувачам зручний інтерфейс для швидкого та ефективного пошуку автомобілів з урахуванням різних параметрів, а також можливість розміщення оголошень для продажу авто. Сформуємо та узагальнимо основні функціональні вимоги до вебсайту, враховуючи рекомендації [25].

1. Під час реєстрації та авторизації користувачі повинні мати можливість створити обліковий запис та увійти в систему для доступу до додаткових функцій, зокрема, розміщення оголошень та налаштування профілю.

2. Впровадження внутрішньої системи пошуку автомобілів має забезпечити користувачам можливість здійснювати пошук за різними ключовими параметрами: марка, модель, рік випуску, тип кузова, об’єм двигуна тощо. Пошук може бути здійснений за допомогою фільтрів.

3. Результати пошуку повинні бути відображені у зручному форматі, показуючи коротку інформацію про автомобілі та зображення. Користувачі повинні мати можливість переглянути докладнішу інформацію про певний автомобіль.

4. Розміщення оголошень – одна з ключових вимог: користувачі повинні мати можливість розміщувати оголошення про продаж своїх автомобілів. Для цього необхідно надати форму для заповнення деталей автомобіля, включаючи марку, модель, рік випуску, фотографії та контактну інформацію.

5. Користувачі повинні мати доступ до особистого профілю, налаштовувати свої персональні дані, переглядати статистику та виконувати інші дії, пов’язані з їх обліковим записом.

Використовуючи інструменти та технології, такі як Figma для дизайну та прототипування, необхідно розробити інтерфейс вебсайту, який відповідає вимогам і забезпечує зручне та ефективне користування для користувачів.

При виборі технологій для розробки вебсайту слід звернути увагу на наступні аспекти [26]:

1. Вимоги проекту: слід ретельно проаналізувати всі вимоги проекту, включаючи функціональні та нефункціональні вимоги, обсяг даних, кількість користувачів, очікувану навантаженість, вимоги до безпеки тощо.

2. Рівень складності: слід оцінити рівень складності проекту, включаючи функціональність, логіку, розмір проекту, кількість розробників, які будуть працювати над проектом.

3. Екосистема: слід вивчити систему технологій та фреймворків, щоб оцінити наявність готових рішень та плагінів, які можуть допомогти розробникам під час роботи.

4. У процесі розробки слід враховувати швидкість робіт та вплив використання конкретних технологій і фреймворків.

5. Важливо враховувати можливість масштабування проєкту, тобто, за необхідності, додавати нові функції або розширювати функціональність в майбутньому.

6. Враховувати сумісність технологій та фреймворків з різними браузерами та операційними системами, які можуть використовувати на сайті.

7. Необхідність враховувати безпекові аспекти, такі як вразливості, захист від атак, можливість шифрування даних, захист від витоків інформації тощо.

8. Важливість підтримки: слід враховувати рівень підтримки технологій та фреймворків, так як підтримка може бути важливою для забезпечення безпеки та розширення функціональності в майбутньому.

9. Враховувати вартість використання конкретних технологій та фреймворків, так як це може бути важливим фактором для бізнесу.

Таким чином, вибір технологій для розробки вебсайту повинен бути зроблений на основі ретельного аналізу всіх вищезгаданих аспектів, а також інших факторів, які можуть бути важливими для конкретного проекту.

2.2 Підбір технологій для розробки вебсайту

Figma – це відоме хмарне програмне забезпечення для спільного дизайну інтерфейсів, яке дозволяє командам створювати, прототипувати та обговорювати цифрові продукти в режимі реального часу [27]. Було створене в 2016 р. і стало стандартом у сфері вебдизайну.

Використання Figma має низку суттєвих переваг, що визначають його популярність серед дизайнерів та розробників. Створення логотипу вебсайту в середовищі Figma представлено для прикладу на рисунку 2.1.

Рисунок 2.1 – Створення логотипу вебсайту в сервісі для розробки інтерфейсів Figma

Перш за все, платформа надає хмарну модель роботи, яка дозволяє отримувати доступ до проєктів з будь-якого пристрою без необхідності встановлення додаткового ПЗ. Це допомагає підвищити гнучкість процесу розробки та значно спрощує організацію командної роботи, оскільки кілька користувачів можуть одночасно редагувати макет у режимі реального часу. Такий підхід забезпечує ефективну комунікацію між учасниками проекту, включаючи дизайнерів, розробників та клієнтів, а також скорочує час на узгодження змін.

Важливою перевагою Figma є наявність інструментів для створення інтерактивних прототипів, щоб моделювати поведінку майбутнього вебсайту на етапі проєктування. Це дає розуміння користувацького досвіду (UX) і дозволяє виявити потенційні проблеми інтерфейсу до початку впровадження ПЗ. Figma підтримує компонентний підхід до проєктування, який передбачає створення елементів інтерфейсу, що використовуються повторно. Це забезпечує узгодженість стилю та спрощує масштабування проекту. Інтеграція з іншими інструментами розробки, можливість експорту графічних ресурсів та підтримка сучасних стандартів адаптивного дизайну роблять цю платформу ефективною для створення високоякісних та конкурентоспроможних продуктів.

Проаналізувавши сучасні технології для розробки вебсайтів, було обрано для розробки вебсайту відповідні інструменти. Для клієнтської частини вебсайту:

ReactJS - це один з найпопулярніших фреймворків для розробки інтерфейсів користувача. Він забезпечує швидку та ефективну розробку вебдодатків за рахунок використання компонентної архітектури.

ReactJS дозволяє легко створювати перевикористовувані компоненти та швидко оновлювати лише ті частини сторінки, які змінилися [28]. Це дозволяє знизити час розробки та поліпшити продуктивність вебдодатку. Приклад синтаксису представлено на рисунку 2.2. Styled Components – це бібліотека для створення компонентів з вбудованими стилями. Вона забезпечує зручний спосіб

написання CSS-стилів для ваших компонентів та дозволяє зберігати їх разом з компонентами.

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import './index.scss';
import App from './App';
import reportWebVitals from './reportWebVitals';
import { BrowserRouter } from 'react-router-dom';
import { Provider } from 'react-redux';
import { store } from './redux/store'

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <BrowserRouter>
    <Provider store={store}>
      <App />
    </Provider>
  </BrowserRouter>
);
```

Рисунок 2.2 – Приклад синтаксису ReactJS

Це дозволяє працювати зі стилями прямо в JavaScript – це компонент, який в стилі підсовує функцію від якихось аргументів. У Styled Component треба звертатися до функції і вона, по суті, може робити будь-що, повертаючи будь-яке рядкове значення для стилю цього компонента [29]. Приклад синтаксису представлено на рисунку 2.3.

Axios – це бібліотека для виконання HTTP-запитів з JavaScript, яка дозволяє легко взаємодіяти зі сторонніми API та забезпечує зручний спосіб обробки відповідей від сервера.

```
const Image = styled.img`
    background: black;
    border: none;
    max-height: 140px;
    height: 100%;
    max-width: 100%;
    border-top-left-radius: 5px;
    border-bottom-left-radius: 5px;
    object-fit: cover;
    object-position: center center;
```

Рисунок 2.3 – Приклад синтаксису Styled components

SCSS – це препроцесор CSS, який дозволяє використовувати змінні, функції та інші покращення, що полегшують написання CSS-коду. Використання SCSS дозволяє зменшити кількість повторюваного коду та полегшити розуміння стилів вебдодатку [30]. Він працює на основі синтаксису CSS, але додає деякі нові функціональні можливості. SCSS дозволяє використовувати змінні для зберігання

значень, таких як кольори, розміри шрифтів, відступи та інші.

Для серверної частини вебсайту застосуємо NodeJS – це платформа для розробки серверних додатків з використанням JavaScript. Вона забезпечує швидку та ефективну розробку серверних додатків за рахунок використання неблокуючого вводу-виводу та асинхронної обробки запитів. При розробці Node.js за основу було взято двигун виконання JavaScript під назвою V8, який був створений компанією Google і використовувався в браузері Google Chrome. Оскільки після створення Node.js Javascript код можна запустити фактично в будь-якому середовищі, за допомогою цієї бібліотеки можна написати серверну частину вебпрограми [20].

Express – це фреймворк для розробки вебдодатків з використанням Node.js. Він дозволяє швидко створювати API для взаємодії Front-End та Back-End. В Express є багато корисних функцій, таких як обробка маршрутів, робота з HTTP-запитами та відповідями, обробка запитів з різними HTTP-методами (GET, POST, PUT, DELETE) та ін. Приклад коду сервера ExpressJS - на рисунку 2.4.

```
async function start() {
  try {
    // const uri = process.env.MONGODB_URI;
    await mongoose.set("strictQuery", false);
    await mongoose.connect(`mongodb+srv://${process.env.DB_USER}:${process.env.
DB_PASS}@cluster0.xoc2ydp.mongodb.net/${process.env.DB_NAME}
?retryWrites=true&w=majority`, { useNewUrlParser: true })
    app.listen(process.env.PORT || 3001, () => console.log("Server started"))
    console.log(app.all)
  } catch (error) {
    console.log(error)
  }
}

start()
```

Рисунок 2.4 – Приклад коду створення сервера ExpressJS

Bcrypt – це криптографічна бібліотека для Node.js, яка дозволяє захистити паролі користувачів. Вона забезпечує хешування паролів з використанням солі, що ускладнює атаки зламу пароля. Bcrypt є стандартом для захисту паролів у більшості вебдодатків [31]. Шифрування паролю користувача та запис в базу даних продемонстровано на рисунку 2.5.

```
const salt = bcrypt.genSaltSync(10)
const hash = bcrypt.hashSync(password, salt)

const newUser = new User({
    email,
    password: hash,
    fullName
})
```

Рисунок 2.5 – Шифрування паролю користувача та запис в базу даних

Jsonwebtoken (JWT) – це бібліотека для створення та перевірки JSON Web Token. JWT – це стандарт, який використовується для автентифікації та авторизації користувачів у вебдодатках. JWT дозволяє створювати токени, які містять інформацію про користувача та його права доступу, які можуть передаватись між сервером та клієнтом для забезпечення безпеки додатка.

Nodemon – це інструмент, який автоматично перезапускає сервер при зміні файлів в проєкті. Це дозволяє значно зменшити час розробки, оскільки розробник може зосередитись на написанні коду, не витрачаючи часу на ручне перезапуску сервера при кожній зміні файлу. Nodemon створений для розробників, що працюють з серверною частиною проектів, такими як вебдодатки чи API.

MongoDB – це документо-орієнтована база даних, яка дозволяє зберігати дані у вигляді JSON-подібних документів. Вона дуже популярна в середовищі Node.js та часто використовується для зберігання даних у вебдодатках. MongoDB дозволяє швидко зберігати та отримувати дані з бази даних, а також має велику кількість інструментів для адміністрування бази даних. Вона також має

вбудовану підтримку MapReduce-style Aggregation та Geospatial індексів [32]. Модель схеми MongoDB новин на вебсайті відображено на рисунку 2.6.

```
import mongoose from "mongoose";

const NewsSchema = new mongoose.Schema(
  {
    title: {
      type: String
    },
    text: {
      type: String
    },
    image: {
      type: String
    }
  },
  { timestamps: true }
)

export default mongoose.model('News', NewsSchema)
```

Рисунок 2.6 – Модель схеми MongoDB новин на вебсайті

Також MongoDB дозволяє розробникам працювати з гнучкими схемами даних. Можна зберігати документи різної структури в одній колекції, що дає можливість адаптувати схему даних під змінні потреби додатка. Це особливо корисно при розробці вебдодатків, де структура даних може змінюватися з часом.

РОЗДІЛ 3
РОЗРОБКА І ТЕСТУВАННЯ ІНТЕРНЕТ-МАГАЗИНУ З ПРОДАЖУ АВТОМОБІЛІВ

3.1 Програмна реалізація основних функцій інтернет-магазину

Перед початком розробки треба визначити структуру сторінок вебсайту з продажу автомобілів, кількість та переходи між ними.

Структура сторінок вебсайту спроєктована наступним чином:

1. «Головна». Головна сторінка має важливе завдання зацікавити відвідувачів і надати їм загальне уявлення про сайт. Її мета полягає в тому, щоб швидко і ефективно передати основну інформацію про вас, вашу компанію, послуги або продукти, що пропонуються.

2. «Всі публікації». Головне джерело інформації, оскільки на цій сторінці користувачі будуть знаходитися та взаємодіяти з вебсайтом більшість часу.

3. «Публікація». Друга по важливості сторінка після всіх публікацій, є сторінка однієї публікації, ці дві сторінки працюють разом і є невід'ємними.

4. «Реєстрація/Авторизація». Сторінки авторизації виконують важливу функцію, тому що від працездатність їх залежить вся робота сайту. Треба уважно протестувати їх, тому що вони можуть бути під впливом різноманітних уразливостей, які можуть бути використані зловмисниками для незаконного доступу або злому безпеки користувачів.

Ось декілька типових уразливостей, на які варто звернути увагу:

– атаки на перехоплення сесії;

– недостатній перевірка введених даних;

– недостатній захист від атак на перелом паролю (brute force);

– недостатній захист від капчі. Капча використовується для відокремлення людей від автоматизованих ботів.

Для захисту клієнтської частини від спаму та ботів використовується Google ReCAPTCHA.

React компонент ReCAPTCHA продемонстровано на рисунку 3.1.

```
<ReCAPTCHA
  sitekey="*API KEY*"
  onChange={handleCaptcha}
  hl="uk"
  size={(width < 400) ? "compact" : "normal"}
/>
```

Рисунок 3.1 – React компонент Google ReCAPTCHA

5. «Профіль користувача». Сторінка профілю користувача відповідає за надання інформації про користувача, його публікації та статистику.

6. «Створення публікації». Сторінка з формою, яка передає інформацію від користувача до бази даних.

7. «Умови використання». Сторінка з умовами використання (Terms of Service) є важною складовою будь-якого вебсайту або онлайн-платформи. Основною метою цієї сторінки є встановлення правових рамок та умов, за якими користувачі можуть використовувати вебсайт.

Далі продемонстровані функції та можливості, які були впроваджені на головну сторінку. Ось основні з них:

На головній сторінці є слайдер з актуальною інформацією вебсайта або рекламою. Слайдери традиційно використовують для привернення уваги за рахунок динаміки та наповнення зображеннями. Також на головній сторінці є «Швидкий пошук за фільтрами» для спрощення пошуку та для пришвидшення роботи з вебсайтом. Розроблено змішаний список популярних параметрів фільтрів, де можна одразу вибрати певні опції та перейти на сторінку з результатами пошуку. Головна сторінка показана на рисунку 3.2.

Слайдер працює за принципом каруселі, при використанні цього компонента передається властивість data, яка містить масив зображень. Компонент використовує хук useState для збереження даних про розміри вікна браузера (windowSize) та поточного зміщення (offset) каруселі.

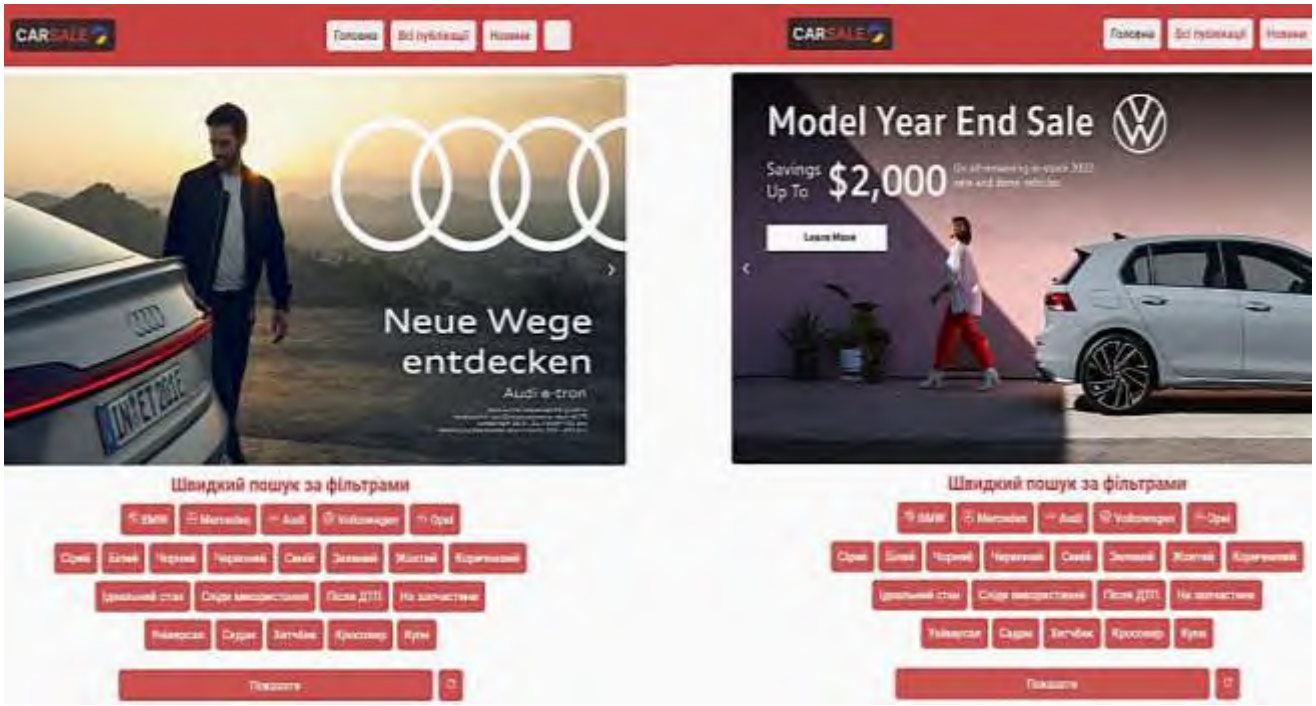

Рисунок 3.2 – Головна сторінка вебсайту з роботою слайдера

Тут також використовується хук useEffect для відслідковування змін розмірів вікна та додавання/видалення подій зміни розміру вікна. Використання React хуків в коді можна побачити на рисунку 3.3.

Компонент також використовує функції hendleLeftArrowClick та hendleRightArrowClick для обробки кліків на стрілки вліво та вправо відповідно. Ці функції перевіряють, чи пройшло більше 500 мілісекунд з останнього виконання функції, і змінюють offset залежно від напрямку руху (додаток Б, рисунок Б.1).

При перегляді користувач бачить, що використовуються два ефекти (useEffect). Перший ефект відслідковує зміни розміру вікна та оновлює windowSize, а також створює масив елементів для каруселі (items). Другий ефект встановлює події touchstart та touchmove для обробки руху по дотику на мобільних пристроях.

Нарешті, компонент повертає JSX-розмітку, яка включає стрілки вліво та вправо для переміщення по каруселі, вікно для відображення зображень та підкреслення активного зображення. Зображення залежать від поточного значення offset та windowSize.

```
const [windowSize, setWindowSize] = useState([window.innerWidth, window.
innerHeight])

useEffect(() => {
    const handleWindowResize = () => {
        setWindowSize([window.innerWidth, window.innerHeight]);
    };

    window.addEventListener('resize', handleWindowResize);

    return () => {
        window.removeEventListener('resize', handleWindowResize);
    };
}, []);
```

Рисунок 3.3 – Використання хука useState та хука useEffect в коді

На сторінці «Всі публікації» є розширена версія фільтрів, де можна підібрати по любому параметру авто. Тут також є бокси зі знайденими публікаціями авто та коротка інформація по цим публікаціям (додаток А, рисунок А.1).

Роботу фільтрів можна описати як роботу комбінації компонентів React, стану Redux та серверної частини на Node.js з використанням Express.js.

Клієнтська частина (React+Redux):

1. Створюються компоненти фільтрів, такі як випадаючі списки, чекбокси, радіокнопки і т.д., які будуть відображати доступні опції для фільтрації автомобілів.
2. Компоненти фільтрів взаємодіють зі станом Redux, диспетчеруючи дії для оновлення фільтрів.
3. При зміні вибраних опцій фільтрів, дії відправляються до Redux store, який зберігає стан фільтрів. Зміни фільтрів можуть тригерувати запити до серверної частини для отримання оновлених даних відповідно до вибраних фільтрів.

Серверна частина, основними технологіями якої є Node.js + Express.js, створена в такій послідовності:

1. Створюється серверна логіка для обробки запитів фільтрації товарів.
2. За допомогою Express.js встановлюються маршрути, які обробляють запити клієнта.

3. При отриманні запиту фільтрації, серверна частина виконує необхідні операції для відбору товарів згідно з вибраними фільтрами.

4. Результати фільтрації повертаються клієнту у відповідь на запит.

Під час реалізації фільтрів важливо забезпечити синхронізацію стану фільтрів між клієнтською та серверною частинами. Це означає, що вибрані фільтри повинні бути передані до сервера, щоб виконати фільтрацію на стороні сервера, а потім результати повернути назад до клієнта для відображення відфільтрованих товарів.

3. Бокси з публікаціями можна побачити на рисунку (додаток А, рисунок А.1). Додатково на сторінці «Всі публікації» є гарна підвантаження публікацій: такий прийом зараз є досить популярним, його назва Skeleton React Loader. Skeleton React Loader - це компонент, який використовується для створення ефекту скелету (від англ. skeleton loading) під час завантаження контенту на сторінці в React.

Коли сторінка завантажується, асинхронно запитується контент з сервера, і це може зайняти певний час. У цей час користувач може бачити порожні блоки або нерозроблені елементи, що може погіршити враження, тому використовують скелетні завантажувачі. Момент загрузки показано на рис. 3.4.

Рисунок 3.4 – Момент завантаження сторінки «Всі публікації»

На сторінці однієї публікації можна побачити всю інформацію про вибрану публікацію. Сторінка поділена на розділи: перший розділ з короткою інформацією, далі другий розділ з детальною інформацією, в третьому зазначений продавець авто, а в четвертому і п’ятому інформація про це авто з зовнішніх джерел. Приклад публікації передавлений в додатку (додаток А, рисунок А.2).

Сторінка профілю зареєстрованого користувача представляє собою невелике меню та основний блок, де можна вийти з аккаунту або додати оголошення. В основному блоці можна налаштувати профіль користувача та переглянути статистику. Додатково є іще один розділ для дилерських акаунтів. Дилерський акаунт представляє собою акаунт з більшою репутацією та має можливість відфільтруватись при пошуку. Для отримання дилерського акаунту необхідно зв’язуватися з адміністрацією сайту. Сторінку профілю показано на рисунку 3.5.

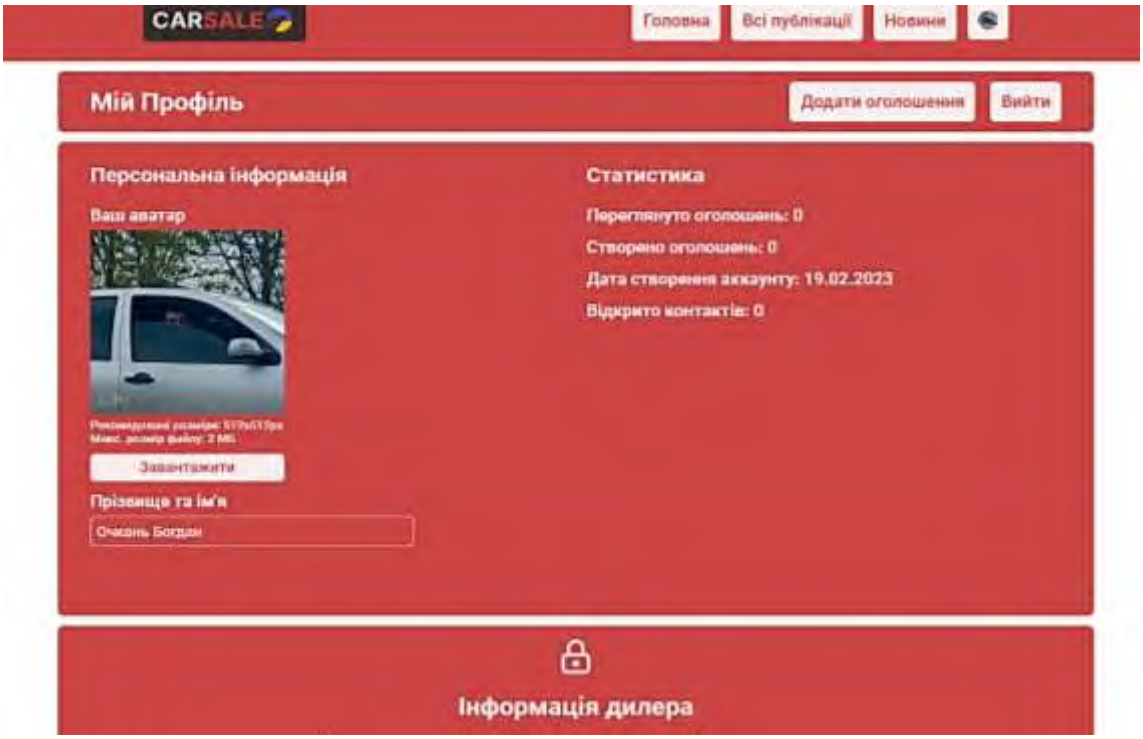

Рисунок 3.5 – Сторінка профілю

Сторінка створення публікації має два основних розділи, попередній перегляд, де можна подивитися як буде виглядати оголошення після заповнення

та публікації, другий розділ це сама форма деталей про авто та контактна інформація. (див. додаток А, рисунок А.3).

В цілому, вебсайт надає зручність, функціональність та різноманітні можливості для користувачів.

3.2 Інтеграції зовнішніх сервісів за допомогою API

Інтеграція зовнішніх сервісів до вебсайту за допомогою API (Application Programming Interface) є важливим аспектом розробки та функціонування сучасних сайтів. API дозволяє взаємодіяти з іншими програмними продуктами та сервісами, використовуючи стандартизовані методи та протоколи комунікації.

Прикладний програмний інтерфейс (інтерфейс програмування застосунків, інтерфейс прикладного програмування, API) – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення. Якщо спрощено, то це набір чітко визначених методів для взаємодії різних компонентів. API надає розробнику засоби для швидкої розробки програмного забезпечення. API може бути для веббазованих систем, операційних систем, баз даних, апаратного забезпечення, програмних бібліотек [33].

Інтеграція зовнішніх сервісів до сайту через API є важливою з таких причин:

1. Розширення функціональності: інтеграція зовнішніх сервісів дозволяє розширити функціональність вашого вебсайту. Можна використовувати функції та дані з інших платформ або сервісів, щоб покращити досвід користувача та надати нові можливості.

2. Обмін даними: за допомогою API можна обмінюватися даними між вашим сайтом та зовнішніми сервісами. Це дозволяє отримувати актуальну інформацію, таку як погода, новини, геолокація тощо, а також надсилати дані на інші платформи, наприклад, для обробки платежів або збереження даних.

3. Підвищення продуктивності: використання зовнішніх сервісів через API дозволяє ефективно використовувати наявні ресурси та функціонал, що зменшує навантаження на сервери та прискорює відповідь вебсайту.

4. Інтеграція з соціальними медіа: API соціальних медіа дозволяють відображати живі стрічки з соціальних мереж на вашому вебсайті, додавати кнопки поділу контенту, авторизацію через облікові записи соціальних мереж тощо. Це сприяє залученню аудиторії, підвищує взаємодію та розповсюдження контенту.

5. Партнерство та інтеграція зі сторонніми сервісами: інтеграція зовнішніх сервісів може бути корисною для партнерства з іншими компаніями або створення власних додатків, що використовують інші платформи або сервіси.

Приклади API-інтеграцій включають платіжні шлюзи, соціальні мережі, картографічні сервіси (наприклад, Google Maps), сервіси електронної пошти, системи аналітики, мобільні додатки тощо.

Для створення вебсайту було використано чотири API, а саме:

1. Google ReCAPTCHA – це система, що була початково розроблена в університеті Карнегі Мелон і базується на використанні CAPTCHA для оцифровування текстів книг заодно із захистом вебсайтів від доступу ботами до обмежених ресурсів. 16 вересня 2009 року Google придбав reCAPTCHA. У цей час reCAPTCHA оцифровує архіви газети New York Times. Вже опрацьовано випуски The New York Times за двадцять років і очікується, що буде оцифровано архіви ще за 110 років [31]. Її використання можна обґрунтувати тим, що вона добре захищає від бот систем при підборі паролю для злому акаунту, або для створення великої кількості акаунтів бот системою.

2. НБУ Курс валют – це JSON файл з актуальним курсом валют, потрібен для конвертування однієї валюти в іншу за актуальним курсом.

3. JSON (JavaScript Object Notatio) – це текстовий формат обміну даними між комп’ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дає змогу описувати об’єкти та інші структури даних [34]. Цей формат

використовується переважно для передавання структурованої інформації через мережу (завдяки процесу, що називають серіалізацією).

4. ChatGPT (Generative Pre-trained Transformer, або укр. Породжувальний попередньо тренований трансформер) – чат-бот зі штучним інтелектом, розроблений лабораторією OpenAI. Це велика статистична модель мови, оптимізована для ведення діалогів та відлагоджена завдяки технікам навчання з учителем та навчання з підкріпленням. В основі прототипу лежить модель OpenAI GPT-3.5 покращена версія GPT-3 [35]. Його використання забезпечує оцінку нейромережі: спочатку йому надається інформація про публікацію та з налаштуваннями йому відправляється прохання оцінити співвідношення ціни та якості і доцільності покупки цього авто.

5. Baza-gai – це сервіс який надає послуги отримання відкритої інформації по державному номеру авто, було надіслано повідомлення до керівництва сайта та був запит на API ключ, мені був надісланий API ключ та далі було зроблено запити при створені публікації до їх сервісу.

3.3 Тестування та реліз вебсайту

Тестування сайту є невід’ємною та критично важливою частиною процесу розробки та підтримки вебсайтів. Воно спрямоване на перевірку функціональності, якості, безпеки та коректності роботи сайту перед його впровадженням або після внесення змін. Ось деякі ключові аспекти, що підкреслюють важливість тестування сайту:

1. Виявлення помилок: тестування дозволяє виявити помилки, дефекти та несправності в роботі сайту перед тим, як він буде запущений для використання користувачами. Це дозволяє уникнути проблем, які можуть негативно вплинути на досвід користувачів та репутацію вашого бренду.

2. Тестування сайту допомагає впевнитися, що вебсайт надає зручний та задовільний користувальницький досвід. Можна перевірити навігацію,

швидкість завантаження сторінок, функціональність взаємодії з елементами сайту, адаптивність до різних пристроїв та браузерів. Покращення користувальницького досвіду сприяє залученню та утриманню відвідувачів на вашому сайті.

3. Тестування функціональності допомагає перевірити, чи працюють всі функції та сервіси на сайті належним чином. Можна перевірити форми зворотного зв'язку, функцію пошуку, обробку платежів, реєстрацію користувачів та інші важливі елементи функціональності сайту і впевнитися, що сайт виконує свої завдання без помилок та перебоїв.

4. Тестування допомагає виявити потенційні слабкі місця та ризики безпеки на сайті і перевірити, чи належним чином захищений сайт від хакерських атак, перехоплення даних та інших загроз. Захист інформації та забезпечення конфіденційності даних є важливими аспектами перед запуском сайту.

5. Оптимізація продуктивності: тестування дозволяє оцінити продуктивність сайту та виявити можливі проблеми, що можуть сповільнювати завантаження сторінок або зменшувати швидкість роботи сайту в цілому. Можна перевірити час відгуку сервера, оптимізацію зображень, кешування та інші аспекти, що впливають на продуктивність.

На етапі тестування адаптації, коли перевіряється працездатність веб-додатку на різних пристроях і розмірах екранів, можуть виникати деякі типові помилки. Однак, існує кілька кращих практик, які можуть допомогти уникнути цих помилок:

6. Респонсивний дизайн: розробляючи вебдодаток, слід звертати увагу на респонсивний дизайн, що дозволяє адаптувати вигляд і макет сторінки до різних розмірів екранів. Використовування медіа-запитів та гнучких одиниць виміру, такі як відносні відсотки або вьюпорт, для забезпечення правильного відображення на різних пристроях.

В цілому, тестування сайту є важливим етапом процесу розробки, який допомагає забезпечити якість, функціональність, безпеку та задоволення користувачів, дозволяє виявити проблеми та недоліки до того, як сайт буде

опубліковано, що забезпечує успішне функціонування та репутацію проєкту [36].

Для прикладу візьмемо сторінку всіх публікацій, тому що користувач буде проводити більшість часу на ній. Перевірка адаптації на різних розмірах екрану:

1. Ширина екрану 1024 пікселів. Результат можна побачити на рисунку 3.6.

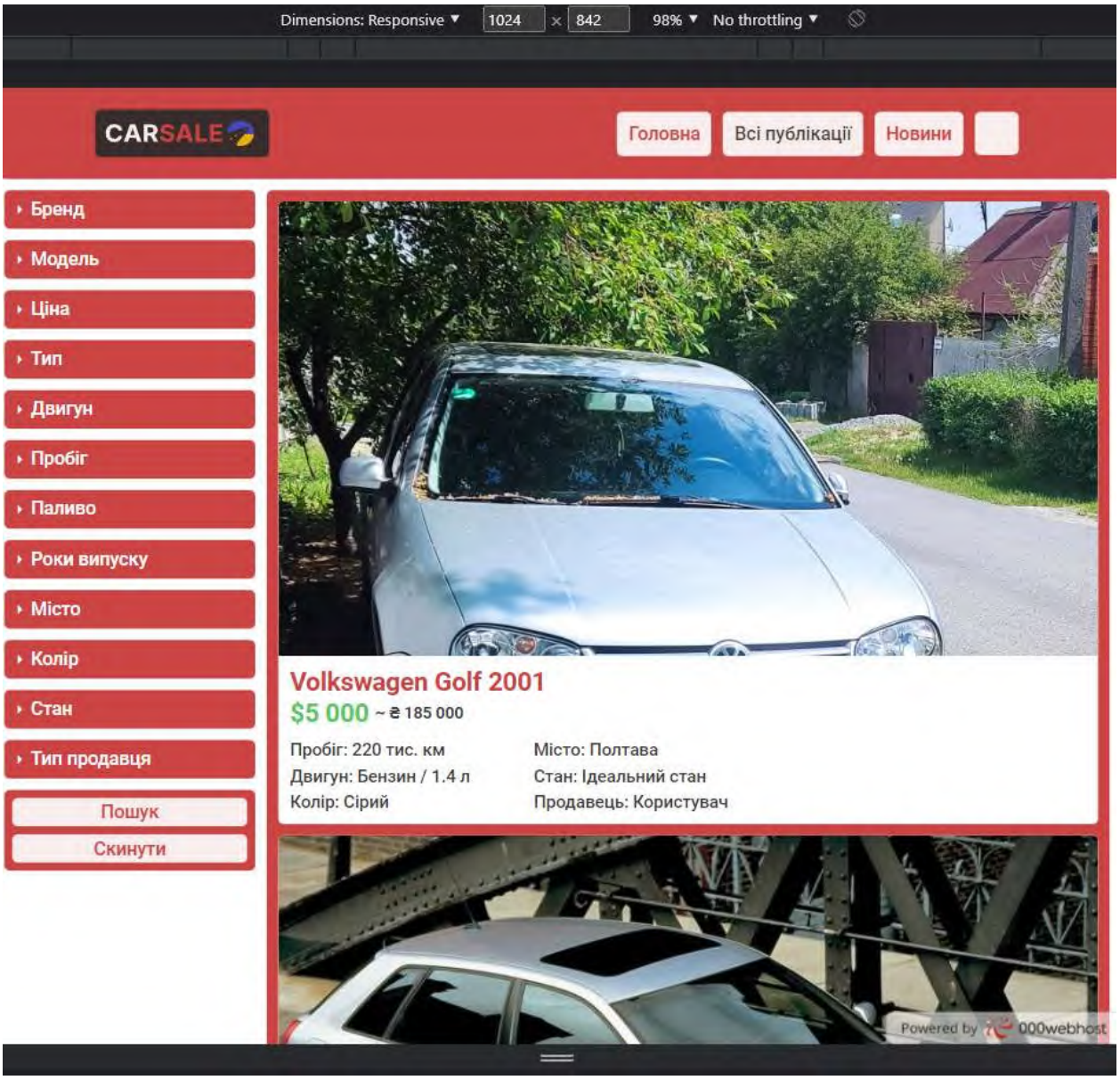

Рисунок 3.6 – Ширина екрану 1024 пікселів

Багато настільних комп'ютерів, які продаються на ринку, підтримують роздільність екрана 1024x768 або вище. Це включає в себе як персональні

комп'ютери, такі як ПК з операційною системою Windows або Mac, так і сервери з операційними системами Linux. Проведено тестування для варіанту ширини екрану 768 пікселів. Результат можна побачити на рисунку 3.7.

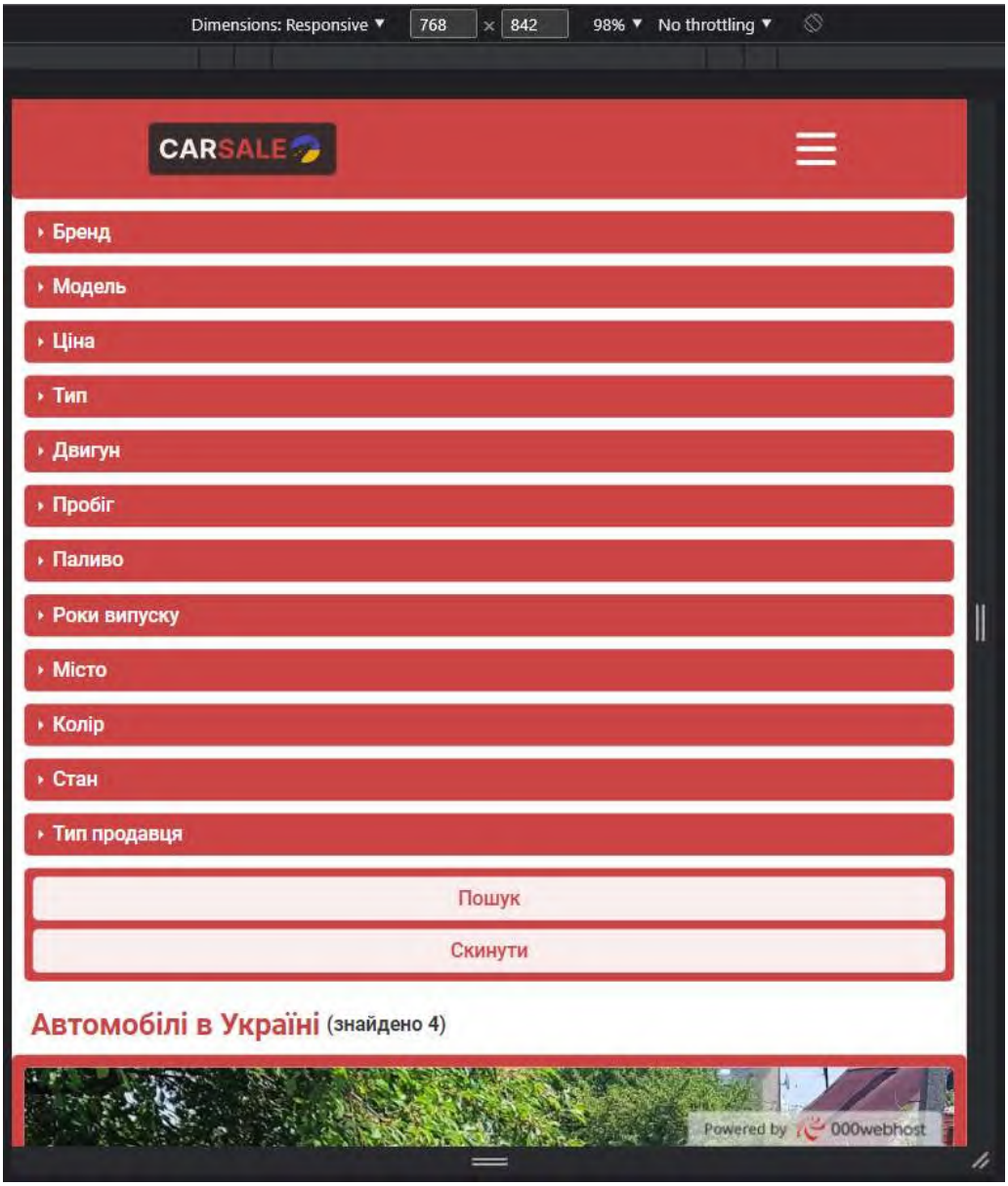

Рисунок 3.7 – Ширина екрану 768 пікселів

На визначеному моменті, екран переходить в інший режим, це працюють правила CSS, а саме @media запити. Медіа-запити дозволяють визначати, які стилі повинні бути застосовані до елементів на основі характеристик пристрою, таких як розмір екрану, орієнтація, роздільна здатність тощо.

2. Ширина екрану 375 пікселів. Результат можна побачити на рисунку 3.8

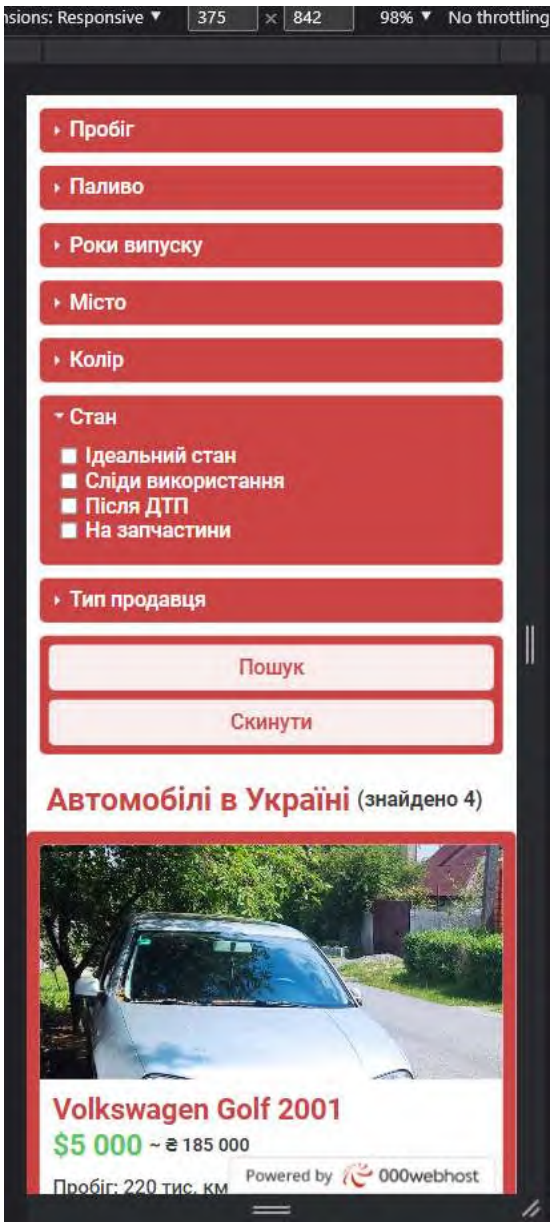

Рисунок 3.8 – Ширина екрану 375 пікселів

Нижче 375 пікселів зазвичай не роблять адаптацією, тому що актуальні розміри мобільних пристроїв зазвичай більше цієї ширини.

Після ретельного тесту адаптації можна зробити висновок, що створення респонсивного дизайну спрощує розробку та адаптацію.

Обрання відповідних сервісів для релізу вебсайту є важливим кроком, який впливає на його доступність, швидкість та зручність використання.

Для релізу сайту було обрано:

– Для серверної частини вебсайту, сервіс replit, він надає можливість розгортання серверної частини вебсайту без складнощів. Він дозволяє

розробникам легко створювати, тестувати та розгортати серверні додатки без складнощів. Інтерфейс Replit продемонстровано на рисунку 3.9

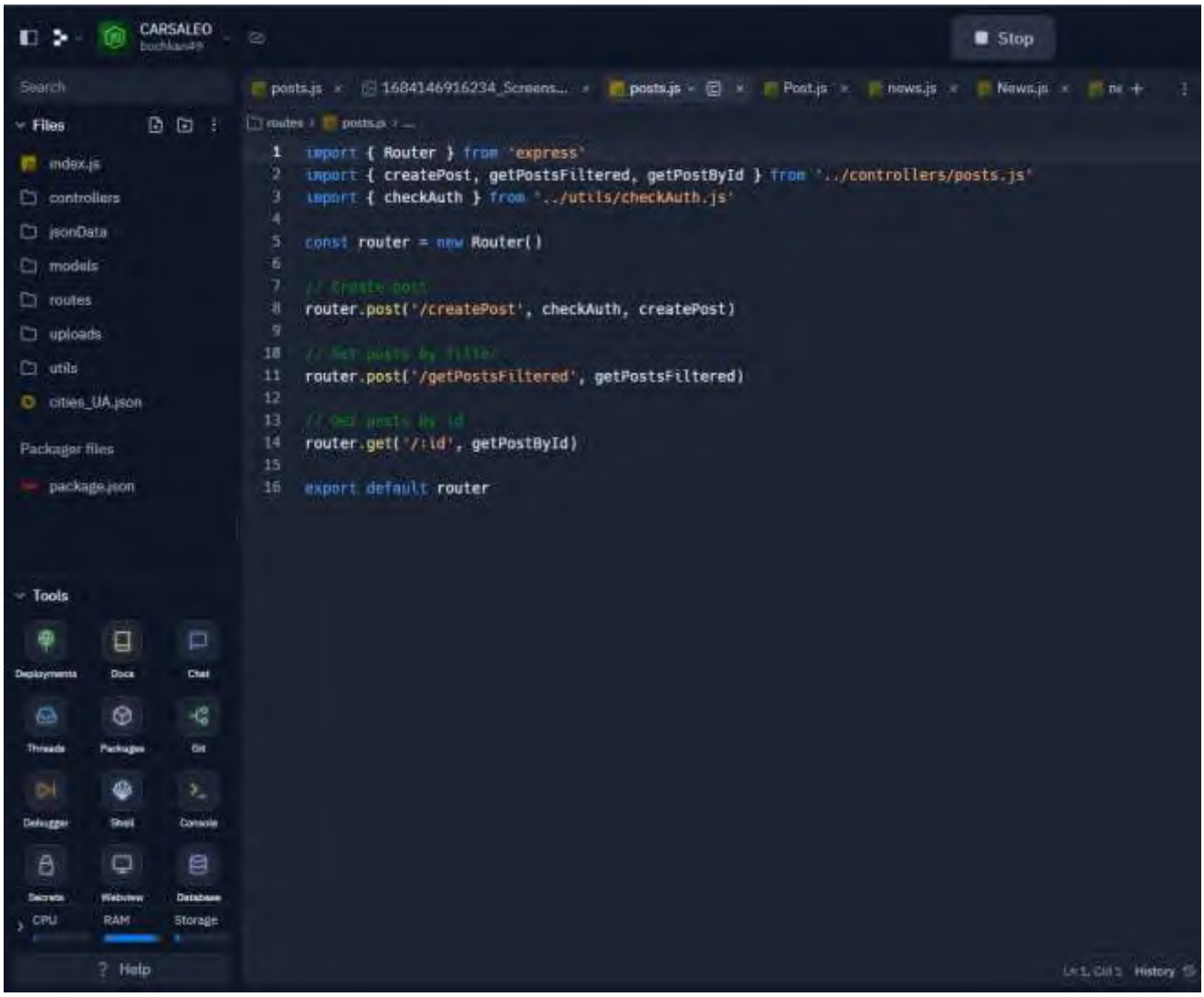

Рисунок 3.9 – Інтерфейс сервісу Replit

– Для клієнтської частини вебсайту було обрано 000webhost, це простий та безкоштовний хостинг вебсайтів зі зручним інтерфейсом та інтуїтивно-зрозумілими функціями. Він дозволяє безкоштовно розмістити клієнтську частину вебсайту та забезпечує доступ до необхідних ресурсів для оптимальної роботи сайту. Хостинг 000webhost має стабільну інфраструктуру, що дозволяє забезпечити надійність та доступність вебсайту. Перелік переваг 000webhost зображено на рисунку 3.10.

Рисунок 3.10 – Перелік послуг хостингу на 000webhost

Використання таких сервісів дозволяє зосередитися на розробці вебсайту, мінімізуючи складнощі та час, пов’язаний зі створенням серверної та клієнтської частини [37].

3.4 Оцінювання економічної та технічної ефективності вебсайту

Розглянемо наближений перелік витрат, пов’язаних із створенням та розміщенням вебсайту. Оскільки використовується безкоштовна версія програмного продукту Visual Studio Code як основа, до складу вартості створення вебсайту включаються такі елементи:

1. Вартість дизайну сайту (залежить від кількості сторінок).
2. Вартість розробки технічного завдання (ТЗ) так, щоб уявлення про проєкт у замовника і розробника максимально збігалися.
3. Вартість верстання вебсайту, яка залежить від багатьох факторів і включає заробітну плату для програміста (FullStack розробник, 650 грн/год.).
4. Витрати на розміщення в мережі інтернет (хостинг та домен).
5. Витрати на необхідні матеріали для комп’ютера.

Вартість розробки вебсайту може бути дуже гнучкою, при проєктуванні визначаються безліч деталей, функцій, можливостей тощо. З іншого боку

компанії та агентства які займаються розробкою сайтів мають різні рівні та вартості на послуги.

Для визначення середньої вартості розробки було взято статистку середньої вартості розробки інтернет-магазину [38]. В розділі ціни за типом, є тип інтернет-магазин, ціна для середнього бізнесу вказана від 100000 грн.

Якщо ж брати середню заробітну плату Fullstack JavaScript розробника за статистикою, вона становить 2600 доларів за місяць [39]. Тому можна зробити висновок що розробка вебсайту буде коштувати приблизно 2600 доларів. В цей перелік буде входити розробка клієнтської частини, серверної частини, просте налаштування SEO та просте тестування.

Припустимо, що компанія звернулася до веброзробника для створення сайту, і вони домовилися, що веброзробник зможе написати сайт протягом 4 тижнів за загальну вартість 2600 доларів. За потреби можна домовитись з розробником на підтримку вебсайту та виправлення помилок в подальшому, але це виконується за потреби, наприклад при покращені дизайну сайту, додаванні нових функцій або доробки теперішніх функцій.

В розрахунках використовується середній курс долара відносно гривні на квітень 2026 року, який становить 1 долар = 43,5 грн.

Розглянемо розрахунок витрат за рік на створення та утримання вебсайту (таблиця 3.1).

Таблиця 3.1 – Витрати за рік на утримання та створення вебсайту

Найменування	Сумма у долларах США	Сумма в грн
Послуга веброзробника (разова)	2600	113100
Хостинг та домен	37	1600
Використання інтернет (абонплата)	80	2880
Разом:	2717	117580

При роботі з програмним забезпеченням при створенні вебсайту необхідно задіяти як мінімум один персональний комп’ютер або ноутбук (таблиця 3.2).

Таблиця 3.2 – Вартість комп’ютерного обладнання

Найменування	Ціна 1 шт., $США	Ціна 1шт., грн	Кількість, шт.
Комп’ютер (Ноутбук) Lenovo	500	21750	1
Миша бездротоваDefender Ultra MM-315 Wireless Black	5	218	1

Метод лінійної амортизації є одним зі способів розрахунку амортизації основних засобів, включаючи комп'ютерну техніку. Для проведення розрахунку амортизації потрібно знати середньорічну вартість основних засобів (С) та норму амортизаційних відрахувань (Н). Формула для розрахунку амортизації за методом лінійної амортизації виглядає так:

$$A = C \times H \div 100\%, \quad (3.1)$$

де A – сума амортизаційних відрахувань за рік, грн;

H – норма амортизаційних відрахувань, %;

C – середньорічна вартість основних засобів.

Провівши розрахунок в доларах США при нормі амортизації 20 %, відповідно отримаємо за формулою (3.1): 505 × 20% ÷ 100% = 101.

Отже, сума амортизаційних відрахувань за рік становить 101 долар США.

Загальний кошторис витрат на виготовлення комерційного вебсайту наведено в таблиці 3.3.

Таблиця 3.3 – Кошторис витрат на розробку і утримання вебсайту

Найменування економічних елементів витрат	Сума, $	Сума, грн
Витрати на розробку	2600	113100
Хостинг та домен	37	1600
Використання інтернет (абонплата)	80	2880
Амортизація основних фондів	101	4394
Інші витрати (комунальні послуги)	820	35670
Разом:	3200	157644

Комунальні послуги включають в себе інтернет та енергоживлення для комп’ютера, з витратою енергії приблизно 60 кВт/год на робочий день. Вартість енерговитрат розрахована за тарифами на 01.04.2026 р.

Таким чином, згідно попередньої оцінки, загальна вартість комерційного проєкту, ячкий може бути розроблений за 1 місяць, разом із річними витратами на утримання і супровід дорівнює 157644 грн.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було проведено аналіз та обрання оптимальних технологій розробки вебсайту з продажу автомобілів, який працює як інтернет-магазин. Було детально вивчено структуру сучасних вебсайтів та встановлено значущість як клієнтської, так і серверної частини, що включало аналіз мов програмування та інструментів для їх розробки.

Для розробки клієнтської частини вебсайту було вибрано мову програмування JavaScript і його бібліотеки, використано інструмент Figma для розробки інтерфейсу. Серверна частина успішно розгорнута на платформі Replit.

Під час створення інтернет-магазину було враховано вимоги до SPA-додатків (Single Page Application), що дозволяє знизити час завантаження сторінки та поліпшити користувацький досвід. Для забезпечення швидкої роботи сайту було використано NodeJS, який забезпечує масштабованість та високу продуктивність.

Дизайн сайту був розроблений з використанням бібліотеки Styled Component, що дозволяє легко керувати стилями та забезпечити їх перевикористовування. Крім того, вебсайт має Full-Stack структуру, що дозволяє працювати з базою даних та здійснювати операції з її збереженням та оновленням.

Було проведено огляд існуючих підходів до розроблення сучасних вебсайтів, зокрема вивчено різні методики та практики. Також розглянуті базові технології розроблення як клієнтської, так і серверної частини веб-сайту,

Обґрунтовано важливість вибору технологій та інтерфейсу вебсайту. Викладені були принципи UX/UI та виконано вибір та розробку кольорової палітри, що сприяє привабливості та гармонії. Окремий наголос зроблено на значенні доступності, що допомагає забезпечити рівний доступ до вебсайту для всіх користувачів. У той же час, продумано систему фільтрації спаму, верифікації користувачів та захисту аккаунтів за допомогою Google ReCAPTCHA.

Проведено розробку, тестування та випуск вебсайту. Були реалізовані основні функції та пройдено їх тестування для перевірки працездатності. Додатково були інтегровані зовнішні сервіси з метою покращення функціональності вебсайту. Після успішного тестування вебсайт був готовий до релізу. Крім того, було проведено економічне обґрунтування проекту, що дозволило оцінити вартість та ефективність розробки.

Таким чином, створення інтернет-магазину з продажу автомобілів на основі технологій React, NodeJS та Styled Component успішно завершено. Результатом проекту є зручний та безпечний інтерфейс для покупців та продавців автомобілів, який надає можливість здійснювати швидке та зручне замовлення, отримувати детальну інформацію про автомобілі та спілкуватися з менеджерами вебсайту.

Загальною метою цієї кваліфікаційної роботи було створення ефективного, привабливого та функціонального вебсайту з використанням сучасних технологій та методологій розробки. Завдяки аналізу, обранню оптимальних рішень та детальній розробці, ціль була досягнута.

Отриманий результат є значущим з практичної та наукової точок зору. З практичної точки зору, створений вебсайт може бути використаний в реальному бізнес-середовищі для продажу автомобілів, забезпечуючи зручність та ефективність процесу купівлі та продажу. Використання сучасних технологій та методологій розробки, таких як React, NodeJS та Styled Component, дозволяє забезпечити швидку та масштабовану роботу вебсайту, покращити користувацький досвід та забезпечити безпеку взаємодії.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Визначення веб-сайту: Типи, компоненти та приклади. URL: https://prehost.com/uk/viznachiennia-vieb-saitu/ (дата звернення 21.09.2025)

2. Котенко Н., Жирова Т., Чибаєвський В., Десятко А. Дослідження основних тенденцій сучасної розробки веб-сайтів. Електронне фахове наукове видання «*Кібербезпека: освіта, наука, техніка»*. 2019. № 1(5). С. 6–15. DOI: 10.28925/2663-4023.2019.5.615.

3. Freeman E., Robson E. *Head First HTML and CSS*. 2nd ed. Sebastopol: O'Reilly Media, 2012. 724 p..

4. Дорошевич О. Дослідження засобів розробки веб-сайтів. URL: https://ekmair.ukma.edu.ua/handle/123456789/18599 (дата звернення 21.10.2025).

5. Мосіюк О. О. Ключові аспекти вивчення Front-End технологій у процесі підготовки майбутніх учителів інформатики. *Наукові записки. Серія: Педагогічні науки*. 2019. № 177. С. 150–154.

6. HTML: HyperText Markup Language. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення 20.10.2025).

7. Creating your first HTML document. *Mozilla Foundation. Web technology for developers*: вебсайт. URL: https://developer.mozilla.org/ (дата звернення 28.10.2025).

8. Дакетт Д. HTML та CSS. Розробка та дизайн вебсайтів. К.: Ексмо, 2020. 480 с.

9. Кей С. Хорстман. Сучасний JavaScript для нетерплячих: ДМК-Пресс. 2021. 288 с.

10. Styled Components – стилізація React додатків: вебсайт. URL: https://highload.today/uk/styled-components-stilizatsiya-react-dodatkiv/ (дата звернення 18.03.2026).

11. Що таке React.js і як він працює? *Brander:* вебсайт. URL: https://brander.ua/technologies/reactjs (дата звернення 18.03.2026).

12. Прогресивний JavaScript фреймворк. URL:https://ua.vuejs.org/ (дата звернення 20.03.2026).

13. Getting started with Angular. URL: https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Frameworks_libraries/Angular_getting_started (дата звернення 20.03.2026).

14. Фаулер М. Рефакторинг коду на JavaScript: поліпшення проєкту існуючого коду: Діалектика. 2023. 464 с.

15. Семпл Д. Введення в UX/UI дизайн. Київ, Видавництво «БІНОМ», 2019. 272 с.

16. Леви Д. UX-стратегія. Чого хочуть користувачі та як їм це дати: К.: «Каравела», 2017. 304 с.

17. Що таке SEO оптимізація?: вебсайт. URL: https://www.taina.com.ua/shho-take-seo-optymizacija/ (дата звернення 21.03.2026)

18. Круг С. Не примушуйте мене думати. Вебюзабіліті та здоровий глузд. 3-тє видання: Ексмо, 2021. 256 с.

19. Мейер Э. CSS. Підручник професіонала. Київ: Видавництво «Діалектика», 2017. 720 с.

20. Що таке NodeJS простими словами: вебсайт. URL: https://dan-it.com.ua/uk/blog/chto-jeto-takoe-node-js-prostymi-slovami/ (дата звернення 12.04.2026).

21. Пауерс Ш. Вивчаємо NODE. Переходимо на бік сервера. 2022. 304 с.

22. JavaScript Підручник. Основи веб програмування. *W3SchoolsUA:* вебсайт. URL: https://w3schoolsua.github.io/js/index.html#gsc.tab=0 (дата звернення 28.04.2026).

23. Що таке Typescript і навіщо він потрібен URL: https://foxminded.ua/typescript/ (дата звернення 28.04.2026).

24. Як працює SPA (Single Page Application) — усе, що потрібно знати. URL: https://dou.ua/forums/topic/50894/ (дата звернення 28.04.2026).

25. Прикладний інтерфейс – Вікіпедія: вебсайт. URL: https://uk.wikipedia.org/wiki/Прикладний_програмний_інтерфейс (дата звернення 26.04.2026).

26. Все про професію UI/UX дизайнера: вебсайт. URL: https://dan-it.com.ua/uk/blog/vse-pro-profesiju-ui-ux-dizajnera/ (дата звернення 30.04.2026).

27. Figma Design. URL: https://www.figma.com/design/ (дата звернення 30.04.2026).

28. O. Kopishynska, I. Sliusar, V. Slyusar, Y. Utkin, O. Halych and O. Skryl. Features of Using Frameworks and Artificial Intelligence Language Models for JavaScript Code Optimization in Web Application Development. *Mater. 14th International Conference on Dependable Systems, Services and Technologies (DESSERT),* Athens, Greece, 2024, pp. 1-7. doi: 10.1109/DESSERT65323.2024.11122152.

29. Адам Д. Скотт. JavaScript з нуля: O'reilly. 2021. 320 с.

30. Хортон В. HTML5 і CSS3. Повністю практичний підхід: Київ, Видавництво "БІНОМ", 2019. 688 с.

31. reCAPTCHA – Вікіпедія. вебсайт. URL: https://uk.wikipedia.org/wiki/ReCAPTCHA: вебсайт. (дата звернення 12.05.2026)

32. Основи MongoDB. вебсайт. URL: https://codeguida.com/post/519 (дата звернення 03.05.2026)

33. Лаптон Е., Коул Д. Графічний дизайн. Нові основи: ArtHuss. 2020. 264 с.

34. JSON – Вікіпедія: вебсайт. URL: https://uk.wikipedia.org/wiki/JSON (дата звернення 03.05.2026).

35. ChatGPT– Вікіпедія: вебсайт. URL: https://uk.wikipedia.org/wiki/ChatGPT (дата звернення 09.05.2026).

36. Густафсон А. Адаптивний вебдизайн: Easy Readers, 2021. 144 с.

37. Бранзі К. Інтерфейси: Підручник для дизайнерів: Київ, Видавництво «Наш Формат», 2020. 352 с.

38. Ціна на створення сайту 2026. Скільки коштує сайт і інтернет магазин: вебсайт. URL: https://ifish.com.ua/ua/tsini-na-sajt/ (дата звернення 08.05.2026).

39. Статистика зарплат програмістів, тестувальників і PM в Україні. URL: https://jobs.dou.ua/salaries/?period=2022-12&position=Middle%20SE&technology=JavaScript (дата звернення 11.05.2026).