

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

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

на тему:

Веб-додаток «Вирішення логістичних проблем на прикладі купівлі та доставки автомобілей з Європейського Союзу»

Виконав студент групи ПЗ-19-3
РУДИК Денис Олександрович

Керівник роботи:
САВІЦЬКИЙ Роман Святославович

Рецензент:
ЧИЖМОТРЯ Олексій Володимирович

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

«ЗАТВЕРДЖУЮ»

В.о. зав. кафедри інженерії
програмного забезпечення
Андрій МОРОЗОВ
«14» лютого 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **РУДИК Денис Олександрович**

Керівник роботи: **САВІЦЬКИЙ Роман Святославович**

Тема роботи: **Веб-додаток «Вирішення логістичних проблем на прикладі купівлі та доставки автомобілей з Європейського Союзу»,**

затверджена Наказом закладу вищої освіти від «14» лютого 2023 р., №70с

Вихідні дані для роботи: **використання вебдодатку для аналізу асортименту, оцінка цін, огляд якості, оцінка процесу замовлення та доставки.**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Савіцький Р.С.	05.05.2023р.	06.05.2023р.
2	Савіцький Р.С.	30.05.2023р.	01.06.2023р.
3	Савіцький Р.С.	07.06.2023р.	08.06.2023р.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка задачі. Пошук, огляд та аналіз аналогічних розробок. Формулювання технічного завдання. Опрацювання літературних джерел.	5 квітня 2023 – 13 квітня 2023	Виконано
2	Проектування структури системи	14 квітня 2023 – 4 травня 2023	Виконано
3	Написання програмного коду	5 травня 2023 – 16 червня 2023	Виконано
4	Тестування	17 червня 2023 – 19 червня 2023	Виконано
5	Попередній захист	21 червня 2023	Виконано

РЕФЕРАТ

Випускна робота бакалавра складається з програмного комплексу додатку з вирішення логістичних проблем на прикладі купівлі та доставки автомобілей з Європейського Союзу та пояснювальної записки. Пояснювальна записка до випускної роботи містить 70 сторінок, 39 ілюстрацій та 9 таблиць.

Метою роботи є створення застосунку для вирішення логістичних проблем купівлі та доставки автомобілей з Європейського Союзу, з імплементацією модуля реалізації підбору машин, опублікування нових оголошень, створення гарячої лінії з питань у вигляді сторінки з контактами, відстеження автомобіля.

В роботі було визначено основні завдання на розробку додатку, проаналізовано аналоги розроблених додатків, обґрунтовано вибір архітектурного патерна для реалізації додатку. Наведено сценарії роботи програмного комплексу, загальна структура додатку, опис бази даних (Supabase), опис алгоритмів взаємодії окремих модулів додатку, алгоритми роботи основних модулів, реалізованих з використанням мови програмування JavaScript.

КЛЮЧОВІ СЛОВА: ОГОЛОШЕННЯ, ПОШУК, РОБІТНИКИ, ГАРЯЧА ЛІНІЯ, БАЗА ДАНИХ, ВИРІШЕННЯ ЛОГІСТИЧНИХ ПРОБЛЕМ.

					ІПЗ.КР.Б-121-23-ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Денис РУДИК			Веб-додаток «Вирішення логістичних проблем на прикладі купівлі та доставки автомобілей з Європейського Союзу» Пояснювальна записка	Літ.	Арк.	Аркуші
Керівник		Роман САВІЦЬКИЙ					2	
						Житомирська політехніка, група ІПЗ-19-3		
Н. контр.								
Зав. каф.		Андрій МОРОЗОВ						

ABSTRACT

Bachelor's thesis consists of a software complex of an application for solving logistical problems using the example of purchasing and delivering cars from the European Union, and an explanatory note. The explanatory note for the thesis consists of 70 pages, 39 illustrations, and 9 tables.

The aim of the thesis is to create an application for solving logistical problems in purchasing and delivering cars from the European Union, with the implementation of a module for car selection, publishing new advertisements, creating a hotline in the form of a contact page, and vehicle tracking.

The thesis defines the main tasks for developing the application, analyzes analogues of the developed applications, justifies the choice of an architectural pattern for implementing the application. It provides scenarios for the operation of the software complex, the overall structure of the application, a description of the database (Supabase), descriptions of the algorithms for the interaction of individual application modules, and algorithms for the operation of the main modules implemented using the JavaScript programming language.

KEYWORDS: ADVERTISEMENTS, SEARCH, WORKERS, HOTLINE, DATABASE, LOGISTICAL PROBLEM SOLVING.

ЗМІСТ

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ДОДАТКУ АВТО З ЄВРОПИ	8
1.1 Постановка задачі.....	8
1.2 АНАЛІЗ АНАЛОГІВ ПРОГРАМНОГО ПРОДУКТУ	10
1.3 ВИБІР АРХІТЕКТУРИ ОБМІНУ ІНФОРМАЦІЄЮ	13
1.4 ОБҐРУНТУВАННЯ ВИБОРУ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ ТА ВИМОГИ ДО АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ	20
Висновки до першого розділу.	22
РОЗДІЛ 2: ПРОЕКТУВАННЯ ДОДАТКУ З ВИРІШЕННЯ ЛОГІСТИЧНИХ ПРОБЛЕМ НА ПРИКЛАДІ КУПІВЛІ ТА ДОСТАВКИ АВТОМОБІЛЕЙ З ЄВРОПЕЙСЬКОГО СОЮЗУ.....	24
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи.....	24
2.2 Розробка бази даних системи	29
2.3 ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ АЛГОРИТМІВ РОБОТИ.....	30
2.4 РЕАЛІЗАЦІЯ ФУНКЦІОНАЛУ В ДОДАТОК.....	31
Висновки до другого розділу	33
РОЗДІЛ 3: ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ	35
3.1 Порядок встановлення та налаштування параметрів системи.	35
3.2 СТРУКТУРА ІНТЕРФЕЙСУ	36
3.3 ТЕСТУВАННЯ РОБОТИ ПРОГРАМНОГО КОМПЛЕКСУ	47
Висновки до третього розділу	49
ВИСНОВКИ	50
ДОДАТКИ	54
<i>ДОДАТОК А</i>	<i>55</i>

API – прикладний програмний інтерфейс.

БД – база даних.

ПЗ – програмне забезпечення.

CRUD – create, read, update, delete (з англ. створити, прочитати, оновити, видалити) .

VIN – унікальний номер автомобіля, що складається з 17 цифр. У них зашифровано марку авто, його характеристики, рік випуску.

GET, POST, PUT і DELETE – методи, які використовуються в протоколі HTTP (Hypertext Transfer Protocol) для взаємодії між клієнтом і сервером у веб-розробці.

GET (отримати) – це метод запиту, який використовується для отримання ресурсу (наприклад, веб-сторінки або даних) з сервера. При виконанні GET-запиту клієнт запитує сервер надати вказаний ресурс.

POST (створити) – метод запиту, який використовується для надсилання даних на сервер для створення нового ресурсу. Часто POST-запити використовуються для відправки форм на сервер або для додавання нових записів у базу даних.

PUT (оновити) – метод запиту, який використовується для зміни або оновлення існуючого ресурсу на сервері. При виконанні PUT-запиту клієнт надсилає дані на сервер з метою замінити існуючий ресурс новими даними.

DELETE (видалити) – це метод запиту, який використовується для видалення ресурсу на сервері. При виконанні DELETE-запиту клієнт запитує сервер видалити вказаний ресурс.

ВСТУП

Актуальність теми. Покупка автомобіля в Інтернеті є актуальною темою у наш час. Інтернет надає широкий вибір автомобілів з різних регіонів та дозволяє здійснити покупку безпосередньо зручно з дому. Це створює потребу в ефективному пошуці, фільтрації, розрахунку вартості для доставки автомобілів з країн Європейського Союзу до клієнтів.

Купівля автомобілів з-за кордону, зокрема з Європейського Союзу, стає все більш поширеною практикою, оскільки споживачі шукають більш широкий вибір моделей, кращу цінову пропозицію або конкретні технічні характеристики. Це створює потребу в ефективних логістичних рішеннях для перевезення автомобілів через міжнародні кордони.

Специфічні вимоги та виклики при доставці автомобілів: Автомобілі - це великі, цінні та вразливі товари, які потребують особливої уваги при транспортуванні. Вони вимагають безпечного фіксування, захисту від пошкоджень та відповідного страхування. Логістика доставки автомобілів з Європейського Союзу може включати додаткові виклики, пов'язані з відповідними митними процедурами, регулюванням транспорту та документацією.

Метою випускної роботи є розробка вебдодатку для логістичних проблем на прикладі купівлі та доставки транспорту з Європейського Союзу.

Дослідити особливості купівлі та продажу автомобілів з Європейського Союзу:

- дослідити процедури і правила, пов'язані з імпортом та експортом автомобілів з країн Європейського Союзу;
- вивчити вимоги до документації та процедури розмитнення при ввезенні автомобілів з ЄС;
- розібратися з податками, митами та іншими фінансовими аспектами, пов'язаними з купівлею та продажем автомобілів з ЄС.

Дослідити ринок автомобільного бізнесу в Україні, пов'язаний з купівлею та продажем автомобілів з ЄС:

					ІПЗ.КР.Б-121-23-ПЗ	
						6

- вивчити ринкові тенденції, включаючи популярні моделі та бренди, попит та пропозицію, конкуренцію тощо;

- проаналізувати вплив ввезення автомобілів з ЄС на внутрішній ринок України, включаючи вплив на ціни, продажі нових автомобілів тощо.

Об'єктом дослідження є пришвидшення процесу купівлі, продажу та вирішення логістичних проблем для доставки замовлення до покупця.

Предметом дослідження є розробка вебдодатку для пришвидшення процесу купівлі, продажу, транспортування автомобілів.

В роботі було використано методи об'єктно-орієнтованого програмування і проектування, а також принципи розробки дизайну інтерфейсів.

Розроблена система може бути використана компаніях, які займаються продажем автомобілей з ЄС.

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ДОДАТКУ АВТО З ЄВРОПИ

1.1 Постановка задачі

Розробка вебдодатку для вирішення логістичних проблем представляє собою створення компонентів: простір опублікування пропозицій, калькулятор розмитнення, пошукова система по VIN номеру, фільтрація оголошень та додаткова контактна інформація для оформлення замовлення замовлення.

Вебдодаток повинен реалізовувати наступні можливості:

- опубліковувати оголошення;
- мати змогу фільтрувати оголошення;
- відстежувати авто по VIN номеру;
- обраховувати повну суму розмитнення;
- ознайомити користувача з оформленням замовлення.

Вебсервіс має представляти собою вебдодаток, за допомогою якого можна буде отримувати інформацію про наявність автомобілів. У цьому додатку повинні бути два типи користувачів: користувач та користувач з правами адміністратора.

Користувач повинен мати:

- ознайомлюватись з інформацією сайту;
- переглядати оголошення;
- користуватися відстеженням по VIN номеру;
- користуватися фільтром;
- отримувати додаткову інформацію за телефонами гарячих ліній.

Для користувача з правами адміністратора повинен бути реалізований наступний функціонал:

- ознайомлюватись з інформацією сайту;
- переглядати оголошення;
- користуватися відстеженням по VIN номеру;

					ІПЗ.КР.Б-121-23-ПЗ	
						8

- користуватися фільтром;
- отримувати додаткову інформацію за телефонами гарячих ліній;
- опубліковувати власні оголошення;
- редагувати їх та видаляти.

Основними етапами розробки описаного додатку:

1. Розробка вебдизайну.

1.1. Розробка дизайну в Figma.

1.2. Реалізація дизайну в коді HTML/CSS .

1.3. Перенесення коду в бібліотеку React.

1.4. Реалізація функціоналу додатку.

1.4.1. Створення структури проекту.

1.4.2. Створення компонентів.

1.4.3. Реалізація функціоналу навігації.

1.4.4. Створення авторизації через Google.

1.5. Реалізація генерації посилання на зображення в БД.

2. Розробка клієнтської частини сервісу.

2.1. Проектування загальної концепції дизайну та користувацького інтерфейсу для навігаційних та інтерактивних елементів.

2.2. Розробка клієнтської частини та реалізація взаємодії з БД.

3. Розробка вебсервісу.

3.1. Реалізація API [1].

3.1.1. Написання програмних модулів для:

3.1.1.1. Авторизації користувачів.

3.1.1.2. Пошуку авто.

3.1.1.2.1. Можливості перегляду інформації про наявність автомобілів.

3.1.1.2.2. Редагування інформації характеристик авто.

3.1.1.2.3. Редагування особистої інформації.

4. Реалізація генерації id користувача.

5. Створення компоненту системи розпізнавання користувача по id користувача.
6. Створення демонстраційних прикладів та тестового контенту.
7. Розробка плану тестування системи:
 - 7.1. Мануальне тестування.
 - 7.2. Тестування графічного інтерфейсу.

Результатом поставлених завдань є створення системи, яка буде самостійно ідентифікувати користувачі при реєстрації чи авторизації за допомогою попередньо створеного id користувача й передавати дані в БД. Вебсервіс буде являти собою застосунок, за допомогою якого користувач або користувач з правами адміністратора матиме змогу переглядати оголошення за допомогою браузера. Також, вебдодаток надає змогу зручно створювати й оновлювати дані про авто. Поєднання компонентів дає можливість автоматизувати процес купівлі та продажу авто й зберігати дану інформацію у базу даних.

1.2 Аналіз аналогів програмного продукту

Розробка сучасного вебдодатку з можливістю конкурентоспроможності й функціональності потребує порівняння його аналогів. Під час дослідження вебдодатків не було знайдено повністю аналогічних, проте було досліджено готові інструменти, які надають змогу реалізувати весь цей функціонал. В процесі проведення аналізу були досліджені наступні продукти: Boss Auto [2], Auto Terminal [3], Komis Auto[4].

Boss Auto (рис.1.1) – продаж автомобілей з Європи. Місія даної компанії - зробити це так, аби купівля була простою та надійною. Аби можна було придбати авто, не думаючи про скручений пробіг чи майстерно заховані пошкодження. Не думати, де та в чому був обман. Не торгуватися за машину, яка не варта тих грошей.

Основними перевагами Boss Auto є:

- велика кількість інформації;

					ІПЗ.КР.Б-121-23-ІЗ	
						10

- простота у використанні сайту;
- доступність;
- наявність соцмереж.

Недоліками є:

- малий функціонал;
- мала кількість послуг.



Рисунок 1.1 - Логотип інструменту Boss Auto

Auto Terminal (рис. 1.2) – організація, яка дозволяє ознайомитись їх роботою та скористатися послугою купівлі автомобіля з Європи. Вона підтримує багато різноманітних способів доставок, які містять в собі доставку з ЄС та США.

Основними перевагами Auto Terminal є:

- великий асортимент авто;
- можливість консультації з провідними спеціалістами;
- можливість створювати замовлення не тільки з ЄС, але й з США.

Недоліками є:

- сервіс не доступний.



Рисунок 1.2 - Логотип інструменту Auto Terminal

Komis Auto (рис. 1.3) – платформа, яка дозволяє придбати авто і впроваджувати своє рішення щодо доставки.

Основними перевагами LeewayHertz є:

					ІПЗ.КР.Б-121-23-ПЗ	
						11

- відносно невелика ціна;
- наявність реалізованої системи, яка непогано підходить для наших вимог;
- можливість замовляти авто не тільки з країн Європи, але і з Америки;
- Недоліками є:
- немає можливості авторизуватися під особистим акаунтом;
- дизайн не зовсім є зручним.



Рисунок 1.3 - Логотип платформи LeewayHertz

Відштовхуючись від отриманих даних, складемо порівняльну характеристику (табл. 1.1).

Таблиця 1.1

Характеристика	Boss Auto	Auto Terminal	Komis Auto	Власний вебдодаток
Зручність навігації	Чітка та зрозуміла	Потребує доопрацювання	Відносно зрозуміла, але потребує часу щоб розібратися	Все знаходиться в зручному місці. Містяться підказки.
Зручність інтерфейсу	+	-	+/-	+
Можливість авторизуватися	+	-	-	+
Наявність підтримки	+	-	+	+
Конфіденційність даних	+	-	-	+
Незалежність від сторонніх сервісів	+/-	-	-	+

Розробка додатку з даним функціоналом не є поширеним, не всі досліджені додатки відповідають тим критеріям, які бути поставлені при розробці.

Зручність і доступність: Вебдодаток дозволить користувачам з усього світу легко взаємодіяти з процесом купівлі та продажу автомобілів з Європейського Союзу. Вони зможуть переглядати доступні автомобілі, отримувати інформацію про їхню історію та стан, встановлювати зв'язок з продавцями та робити покупки зручним способом.

Оптимізація логістичних процесів: Додаток дозволить оптимізувати логістичні процеси, пов'язані з перевезенням автомобілів з Європейського Союзу до покупців. Можливість відстежування статусу доставки, управління логістикою та координація між продавцями, перевізниками та отримувачами автомобілів допоможе зменшити затримки та покращити швидкість доставки.

Також невід'ємною частиною програмного продукту, що створюється, є конфіденційність. Так як для нашої системи є необхідним використання фотографій, особистих даних та інше, витік такої інформації матиме негативні наслідки. Через це передача конфіденційної інформації стороннім сервісам та особам не є можливою.

Отже, за результатами порівнянь різних вебсайтів, було обрано варіант створення власного вебдодатку, так як основними критеріями є зручність, стабільність додатку й недорога собівартість.

1.3 Вибір архітектури обміну інформацією

Застосунок складається з компонентів клієнтської та серверної частин. Серверна частина, яка являє собою API, за допомогою якої ми можемо отримати данні для клієнтської частини.

API (Application Programming Interface) - це набір правил і протоколів, які дозволяють програмам взаємодіяти одна з одною. API визначає, які функції програми можуть бути викликані, які дані можуть бути передані та який формат цих даних.

При використанні API, створюється запит до сервера, що містить певну функціональність. Запит може бути зроблений за допомогою HTTP (або іншого протоколу), і сервер надсилає відповідь у вигляді даних, які ви можете використовувати у своїй програмі.

Основні елементи API:

Endpoint (точка доступу): Це URL, за яким ви можете звернутися до сервера. Endpoint визначає, яка конкретна функція або ресурс сервера буде викликана.

HTTP методи: Найпоширеніші методи це GET, POST, PUT, DELETE. GET[5] використовується для отримання даних з сервера, POST - для відправки даних на сервер, PUT - для оновлення існуючих даних на сервері, DELETE - для видалення даних.

Параметри запиту: Це додаткові дані, які можна передати разом із запитом до сервера. Наприклад, ви можете передати ідентифікатор користувача або фільтр для отримання певного виду даних.

Формати даних: API може підтримувати різні формати даних для обміну інформацією, такі як JSON [6] (найпоширеніший) та XML[7].

Автентифікація: Деякі API можуть вимагати автентифікації для забезпечення безпеки та контролю доступу. Це може включати використання токенів, ключів API або інших методів ідентифікації.

Основними структурними елементами (рис. 1.4) даної архітектури є:

- клієнтська частина (Client) – відповідає за відображення інформації;
- контролер (Controller) - відповідає за обробку запитів, які надходять з клієнтської частини додатку;
- API (Api) – метод, який відповідає за отримання інформації з бази даних;

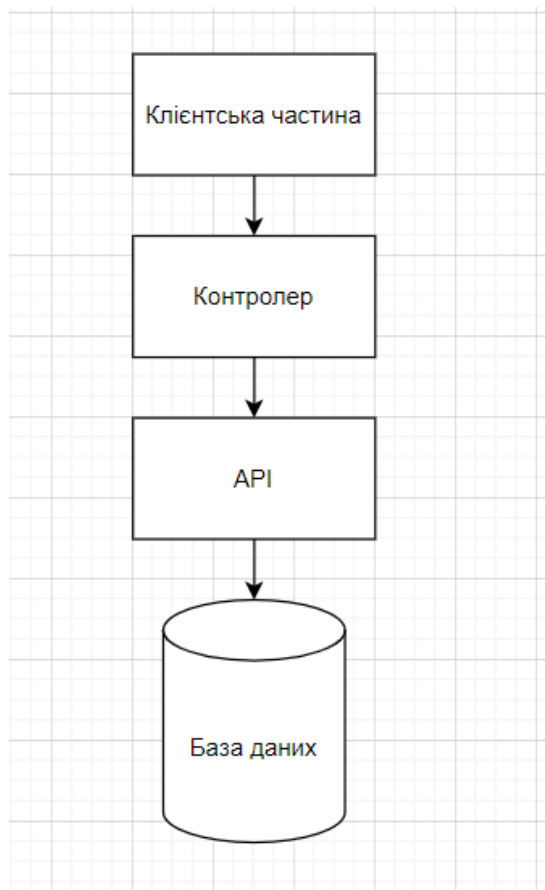


Рисунок 1.4 - Взаємодія компонентів з БД

При виборі архітектурного рішення було проаналізовано переваги та недоліки різних підходів. Для вебсервісу, що створюється, можна обрати одну з наступних архітектур:

- архітектура з використанням компонентів;
- архітектура з використанням контейнерів і компонентів;
- архітектура з використанням Redux або інших стейт-менеджерів.

Архітектура з використанням компонентів:

- компоненти React [8] відповідають за відображення і взаємодію з користувачем;
- компоненти можуть робити запити до API для отримання необхідних даних;
- компоненти можуть також відправляти дані на сервер через API, наприклад, при створенні або оновленні ресурсів.

Архітектура з використанням контейнерів і компонентів:

- контейнери є компонентами, які відповідають за обробку логіки взаємодії з API;
- контейнери отримують дані з API і передають їх у вигляді властивостей (props) до презентаційних компонентів;
- презентаційні компоненти не мають власного стану і просто відображають дані, що отримують від контейнерів.

Архітектура з використанням Redux або інших стейт-менеджерів:

- Redux (або інші стейт-менеджери, такі як MobX[9]) використовується для централізованого управління станом додатка;
- запити до API та отримані дані зберігаються в Redux Store;
- компоненти React підписуються на зміни стану в Store і оновлюють свої дані відповідно;

Після проведення порівнянь згаданих варіантів та аналізу вимог додатку було обрано архітектура з використанням контейнерів і компонентів.

Основною ідеєю контейнерних компонентів є відокремлення бізнес-логіки від представлення даних. Контейнерні компоненти відповідають за отримання даних з зовнішніх джерел (таких як API), обробку цих даних і передачу їх до презентаційних компонентів.

Наприклад, ви можете створити контейнерний компонент, який виконує запит до API за допомогою бібліотеки, такої як Fetch [10], і отримує дані. Після отримання даних контейнерний компонент може виконати додаткову обробку (фільтрацію, сортування тощо) і передати ці дані до презентаційних компонентів у вигляді властивостей (props [11]).

Презентаційні компоненти (наприклад, компоненти React) відповідають за відображення даних і взаємодію з користувачем. Вони отримують дані від контейнерних компонентів через властивості (props) і використовують їх для відображення відповідного інтерфейсу.

Так як дана архітектура базується на використанні React компонентів, додаток набуває додаткових переваг за допомогою цієї бібліотеки:

- більш простий та зрозумілий програмний код.
- можливість перевикористовувати створені компоненти в різних частинах додатку, що дає змогу пришвидшити розробку, полегшує розуміння проекту та покращує його якість.

Архітектура з використанням контейнерів і компонентів (рис. 1.5), вона має свої переваги й позитивно впливає на швидкодію та надійність додатків.

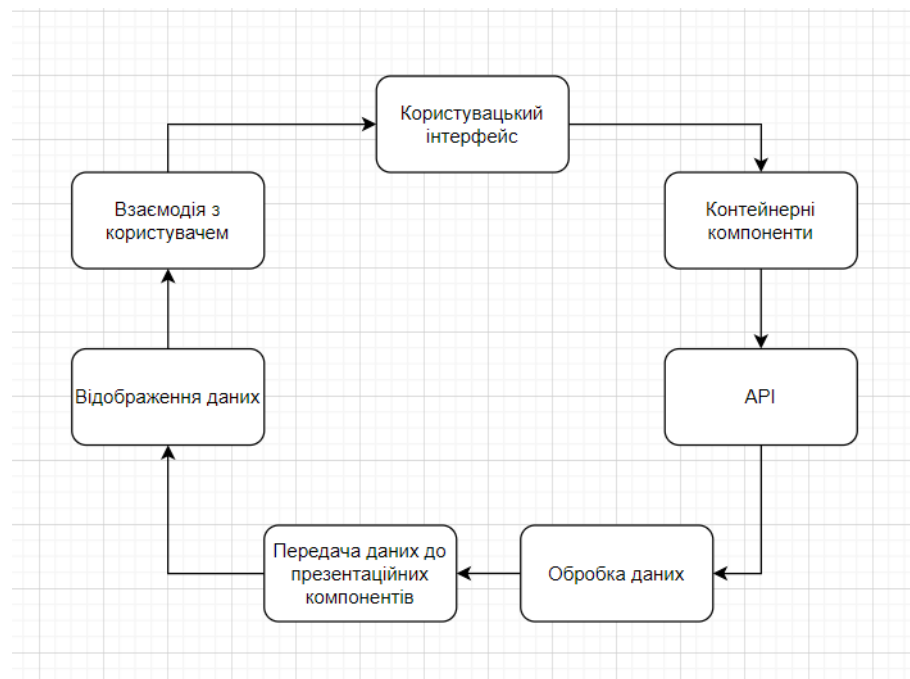


Рисунок 1.5 - Архітектура з використанням контейнерів і компонентів

Процес передачі даних являє собою взаємодію наступних частин:

- користувацький інтерфейс: Створення презентаційних компонентів, які відповідають за відображення даних та взаємодію з користувачем;
- контейнерні компоненти: Створення контейнерних компонентів, які відповідають за отримання даних з API, їх обробку та передачу до презентаційних компонентів;
- API: Взаємодія з зовнішнім API для отримання необхідних даних. Для цього можна використовувати бібліотеки, такі як Axios або Fetch, для виконання HTTP-запитів до API сервера;
- обробка даних: Контейнерні компоненти обробляють отримані дані з API, наприклад, фільтрують, сортують або групують їх відповідно до потреб застосунку;

- передача даних до презентаційних компонентів: Оброблені дані передаються з контейнерних компонентів до презентаційних компонентів через властивості (props);

- відображення даних: Презентаційні компоненти відповідають за відображення даних, які вони отримали від контейнерних компонентів. Вони створюють необхідні інтерфейси для відображення даних користувачу;

- взаємодія з користувачем: Презентаційні компоненти також відповідають за взаємодію з користувачем, наприклад, обробку подій, таких як кліки на кнопки, введення тексту в поля вводу, вибір пунктів зі списку та інші дії, які користувач може виконувати на інтерфейсі. Коли відбувається взаємодія, презентаційні компоненти можуть викликати відповідні функції чи методи, які обробляють ці події і здійснюють необхідні дії.

Архітектура з використанням контейнерів і компонентів (рис. 1.6) представляє собою набір послідовно з'єднаних модулів:

- користувач взаємодіє з додатком, ініціює події;
- контейнерний компонент отримує ці події та викликає відповідні функції або методи;
- контейнерний компонент здійснює запит до зовнішнього API за допомогою відповідних бібліотек;
- API обробляє запит та повертає відповідь з необхідними даними;
- контейнерний компонент оновлює стан та передає отримані дані презентаційним компонентам;
- презентаційні компоненти отримують дані та відображають їх на екрані для користувача.

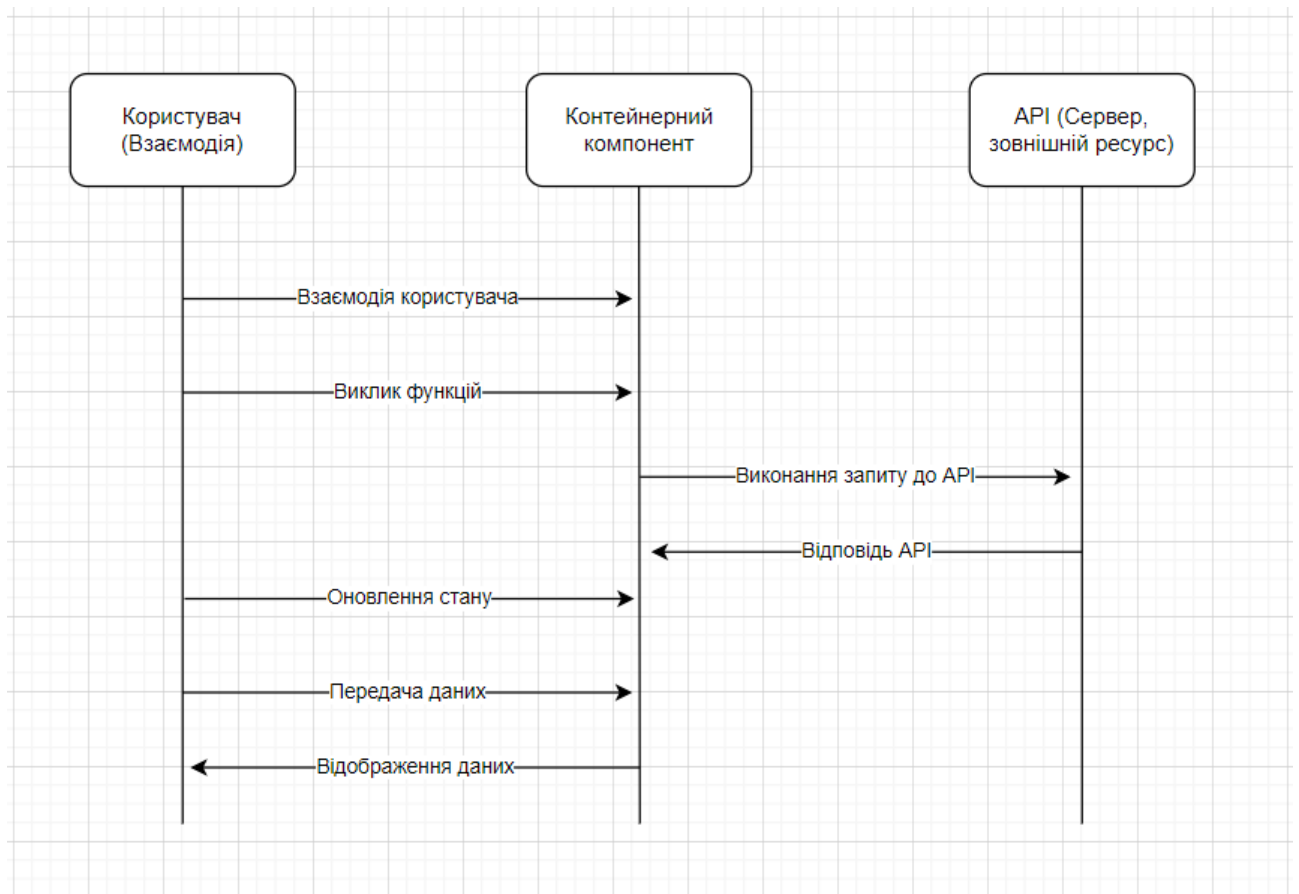


Рисунок 1.6 - Архітектура з використанням контейнерів і компонентів взаємодія з користувачем

Всі описані компоненти: вебсервер, клієнтський додаток (React додаток) та база даних сполучаються між собою за допомогою двонаправленого зв'язку між вебсервером та іншими частинами системи (рис. 1.7).

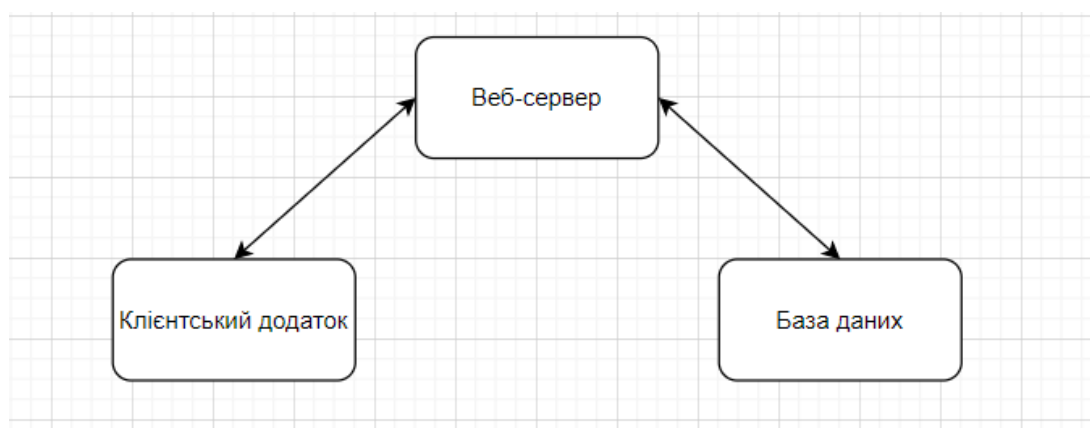


Рисунок 1.7 - Взаємодія компонентів

1.4 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

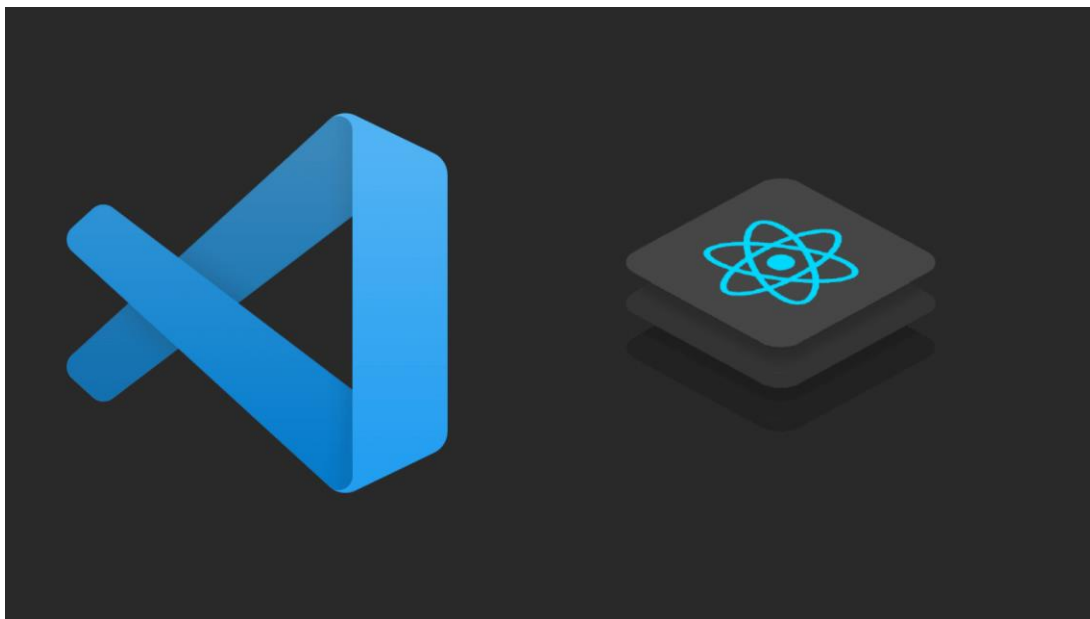


Рисунок 1.8 – Середовище розробки та бібліотека React

Розробка кожного компонента додатку вимагає набору певних інструментальних засобів.

Розробка клієнтського додатку відбуватиметься у інтегрованому середовищі розробки – Visual Studio Code[12]. Для розробки клієнтської частини були обрані наступні засоби:

1. React – це JavaScript-бібліотека для розробки інтерфейсів користувача [10].

2. React Router DOM[13] - це розширення бібліотеки React, яке надає можливості маршрутизації (routing) для вебдодатків, розроблених з використанням React. Воно дозволяє створювати односторінкові додатки (Single-Page Applications - SPA) з багатосторінковою навігацією, де вміст сторінки змінюється динамічно без перезавантаження сторінки.

3. Google-map-react [14] – бібліотека Material Design розроблена для React. Використовується для стилізації користувацького інтерфейсу.

4. @supabase/auth-helpers-react [15] - це бібліотека, що надає набір утиліт та компонентів для автентифікації в React-додатках з використанням Supabase.

Обрані засоби реалізації дозволяють створити зручний та сучасний клієнтський додаток з використанням новітніх підходів розробки, що забезпечують якість програмного продукту та надійність.

Серверна частина додатку виконувалася в Visual Studio за допомогою локального сервера.

Всі інші запити обробляє база даних Supabase:

Supabase [16] - це відкрите й просте в розгортанні API для створення та управління базами даних. Ця платформа надає інфраструктуру для розробки функціональних і безпечних серверних додатків швидко та зручно.

З метою гарної працездатності системи, забезпечення швидкодії та дотримання ефективного використання ресурсів апаратного забезпечення були сформовані певні вимоги:

1. Вимоги до локального серверу:
 - 1.1. Оперативна пам'ять - 8192 MB
 - 1.2. Жорсткий диск - 512 GB
 - 1.3. Процесор з частотою від 2 GHz
 - 1.4. Браузер
2. Вимоги до робочої станції розробника:
 - 2.1. Оперативна пам'ять - 8192 MB
 - 2.2. Операційна система – Windows 8/10
 - 2.3. Процесор з частотою від 2 GHz
 - 2.4. Браузер
 - 2.5. Жорсткий диск (бажано ssd або M2) - 256 GB
3. Вимоги до робочої станції користувача:
 - 3.1. Гаджет з браузером, який відповідає наступним вимогам:
 - 3.1.1. Chrome 60+
 - 3.1.2. Safari 10+ / iOS Safari 10+
 - 3.1.3. Edge 12+
 - 3.1.4. Firefox ESR+
 - 3.1.5. Internet Explorer 11

3.1.6. Opera

Час реакції програми на дії користувача (маніпуляції з пристроями введення даних) не повинен перевищувати 0,3 с.

Час виконання команд меню не більше 4 с.

Відображення даних за запитом не більше 10 секунд.

Доступність БД – 90% цілодобово.

Операції з'єднання з БД не більше 15 секунд.

Програма повинна нормально функціонувати при безперебійній роботі ПК.

Висновки до першого розділу.

Перш за все, використання React дозволяє побудувати інтерфейс додатку, який є гнучким, ефективним та легко розширюваним. React використовує компонентний підхід, що дозволяє розбити інтерфейс на незалежні компоненти, що полегшує управління та підтримку коду. Крім того, React має широку спільноту розробників, тому легко знайти документацію та приклади розв'язків для багатьох завдань.

API (Application Programming Interface) є ключовим елементом будь-якого сервісу. Використання API дозволяє забезпечити комунікацію між фронтендом (React) та бекендом (серверною частиною). Це дозволяє отримувати та передавати дані, керувати замовленнями.

SupaBase є відкритою, готовою до використання базою даних. Його використання спрощує управління базою даних та забезпечує швидку реалізацію запитів та маніпуляцій з даними. SupaBase також надає рішення для автентифікації та авторизації користувачів, що є важливим аспектом для вебдодатку з обмеженим доступом до деяких функцій або конфіденційної інформації.

В цілому, створення архітектури застосунку з використанням React, API та SupaBase дозволяє створити потужний та масштабований магазин з гнучким та

					ІПЗ.КР.Б-121-23-ПЗ	
						22

ефективним інтерфейсом, а також надає зручні інструменти для управління базою даних та взаємодії з зовнішніми сервісами. За допомогою цих технологій можна створити сучасний та конкурентоспроможний додаток, який задовольнятиме потреби як розробників, так і кінцевих користувачів.

РОЗДІЛ 2: ПРОЕКТУВАННЯ ДОДАТКУ З ВИРІШЕННЯ ЛОГІСТИЧНИХ ПРОБЛЕМ НА ПРИКЛАДІ КУПІВЛІ ТА ДОСТАВКИ АВТОМОБІЛЕЙ З ЄВРОПЕЙСЬКОГО СОЮЗУ

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

Додаток поєднує в собі взаємодію користувачів та користувачів з правами адміністраторів в одній системі. З метою наочного представлення обміну інформації між елементами системи було створено діаграму варіантів використання (рис. 2.1).

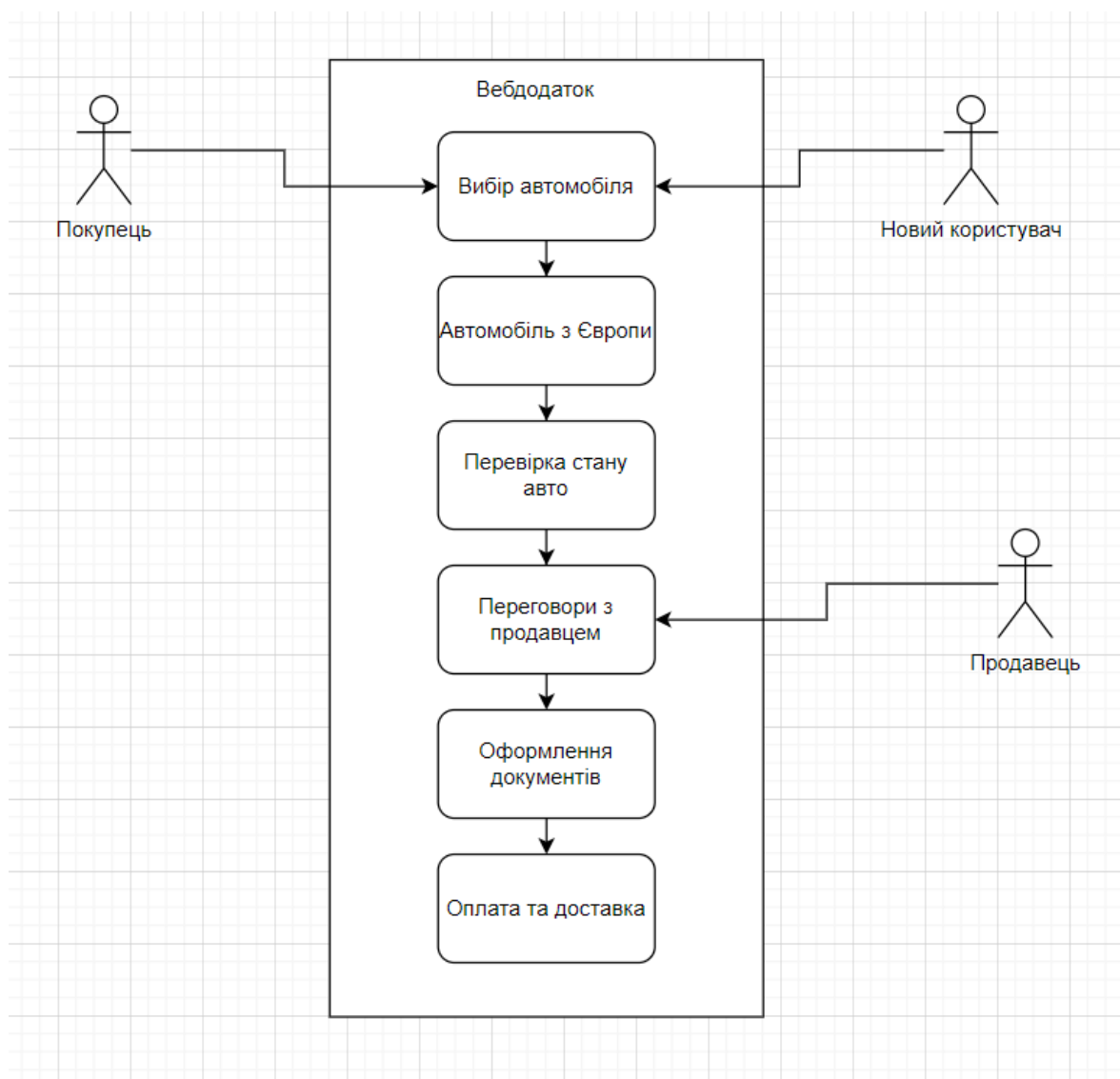


Рисунок 2.1 - Діаграма варіантів використання

Опис кожного прецеденту наведено у таблиці (табл. 2.1 – 2.12).

Таблиця 2.1

Назва варіанту використання	Вибір автомобіля
Основні актори	Покупець, вебдодаток
Короткий опис	Покупець розглядає різні варіанти автомобілів з країн Європи

Таблиця 2.2

Назва варіанту використання	Автомобіль з Європи
Основний актор	Покупець, вебдодаток
Короткий опис	Кожен варіант автомобіля представляє окремий актор інформації про автомобілі з певної країни ЄС.

Таблиця 2.3

Назва варіанту використання	Перевірка стану авто
Основний актор	Покупець, вебдодаток
Короткий опис	Покупець проводить перевірку технічного стану автомобіля, включаючи огляд, тест-драйв та перевірку історії обслуговування.

Таблиця 2.4

Назва варіанту використання	Переговори з продавцем
Основний актор	Покупець, вебдодаток, продавець
Короткий опис	Покупець вступає в переговори з продавцем щодо ціни, умов продажу, можливості торгу тощо.

Таблиця 2.5

Назва варіанту використання	Оформлення документів
-----------------------------	-----------------------

Основні актори	Покупець, вебдодаток, продавець
Короткий опис	Після досягнення угоди з продавцем, покупець займається оформленням необхідних документів для легалізації автомобіля.

Таблиця 2.6

Назва варіанту використання	Оплата та доставка
Основні актори	Покупець, вебдодаток, продавець
Короткий опис	Покупець здійснює оплату автомобіля та організовує його доставку з Європи.

Ця діаграма варіантів демонструє послідовність кроків, які покупець здійснює при придбанні автомобіля з Європи, починаючи з вибору автомобіля до оплати та доставки. Кожен крок є окремим актором, що виконує певну роль у процесі продажу. Задля того, щоб створена система задовольняла необхідні потреби, для нашого продукту були сформовані відповідні вимоги.

Бізнес-вимоги:

1. Основні цілі: додаток створюється з метою полегшити процес придбання або продажу автомобіля
2. Консультації: по використанню додатку та з інших питань будуть надаватися гарячі лінії
3. Представлення проекту: проект буде реалізовано у вигляді вебсайту

Вимоги користувачів:

1. Користувачі:
 - 1.1. Перегляд оголошень автомобілів
 - 1.2. Можливість фільтрувати за різними критеріями
 - 1.3. Можливість відстежувати авто за VIN [17] номером
 - 1.4. Користувач може скористуватися калькулятором для підрахунку загальної суми розмитнення авто

1.5. Є можливість скористатися гарячою лінією та ознайомитись з інформацією про компанію

2. Користувачі з правами адміна:

2.1. Перегляд оголошень автомобілів.

2.2. Можливість фільтрувати за різними критеріями.

2.3. Можливість відстежувати авто за Vin номером.

2.4. Користувач може скористуватися калькулятором для підрахунку загальної суми розмитнення авто.

2.5. Є можливість скористатися гарячою лінією та ознайомитись з інформацією про компанію.

2.6. Має можливість додавати оголошення ввівши код у поле надання цієї послуги.

2.7. Редагувати додані оголошення та видаляти.

Вхідні дані: дані про авто, інформація щодо користувача та зображення авто.

Нефункціональні вимоги:

1. Сприйняття:

1.1. Час для ознайомлення з додатком для звичайних користувачів – 15 хвилин години, а для досвідчених – 10 хвилин.

1.2. Час відповіді системи в більшості випадків не повинен перевищувати 5 секунди, а в випадку складних запитів – 7 секунд

1.3. Інтерфейс повинен бути зручним та сучасним

2. Надійність

2.1. Максимальна норма помилок та дефектів в роботі системи – 1 помилка на 1000 запитів користувача.

3. Продуктивність

3.1. Система має працювати без перебоїв при одночасному використанні користувачами.

За основу об'єктно-орієнтованої структури системи було обрано багатoshарову архітектуру. Основними структурними елементами є: моделі-сутності, моделі-прив'язки, менеджери, API, допоміжні методи та контролери.

Діаграма класів, яка описує спосіб роботи додатку наведена на (рис. 2.2).

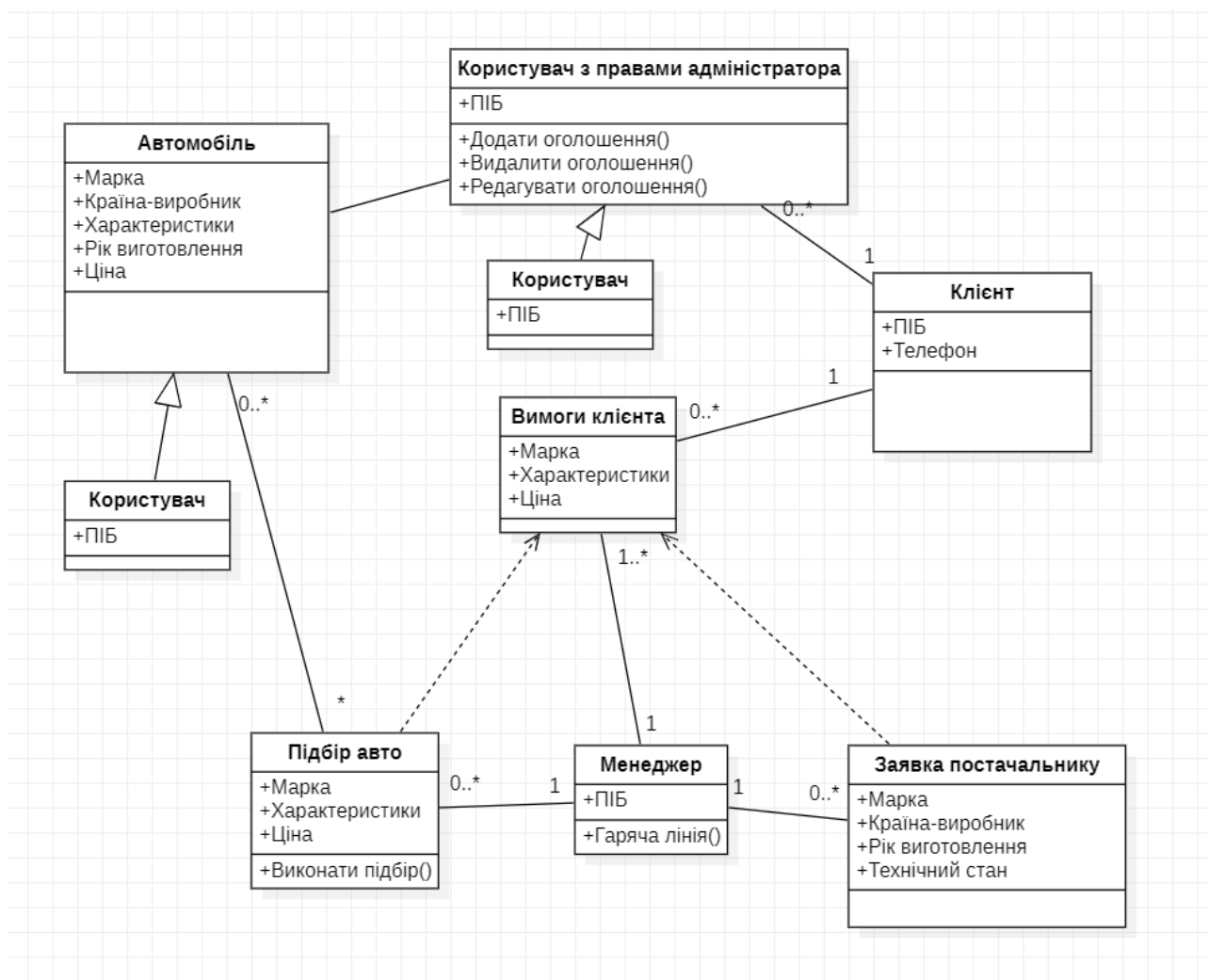


Рисунок 2.2 - Діаграма класів моделей-сутностей

Автомобіль – всі оголошення авто, які зібрані в базі даних та відображаються користувачеві при перегляді наявних авто. До кожного автомобіля приєднаний продавець, який являється користувачем додатку.

Користувач та користувач з правами адміністратора – майже один і той же користувач, відмінність полягає в тому, що у користувача з правами адміністратора є можливість додавати оголошення, редагувати та видаляти їх.

Клієнт – пред'являє вимоги щодо пошуку авто та очікує зв'язку від компанії.

Підбір авто – відбувається при фільтруванні всіх оголошень автомобілей.

Менеджер – відбувається зв'язок через гарячу лінію.

Заявка постачальнику – надсилається через менеджера.

2.2 Розробка бази даних системи

Для збереження сутностей та взаємодії з даними системи контролю відвідуваності була розроблена база даних. Для нашого вебдодатку було обрано SupraBase. Відповідно до описаної архітектури, взаємодія між вебдодатком та БД.

База даних системи складається з 2 таблиць:

1. Cars – таблиця, яка зберігає дані про пости в системі.
2. Images – таблиця, яка зберігає дані зображень

Нижче наведений опис кожної з наведених таблиць бази даних (див. табл. 2.13 – 2.14)

Таблиця 2.7

Cars		
Поле	Тип	Призначення
Id	int	Id оголошення
created_at	text	Дата та час створеного оголошення
car_price	text	Ціна автомобіля
car_description	text	Опис
type_of_transport	text	Тип транспорту
origin	text	Походження авто
city	text	Місто перебування
brand	text	Бренд
model	text	Модель
year	text	Рік виготовлення
color	text	Колір
injured	text	Пошкодження
customs_cleared	text	Розмитнення
wheel_drive	text	Привід автомобіля
transmission	text	Коробка передач
security	text	Захист

multimedia	text	Мультимедія
comfort	text	Комплектація
fuel	text	Паливо
engine	text	Ємність двигуна
range	text	Пробіг
img_link	text	Посилання на фото
vin_num	text	Vin номер
user_id	text	Id користувача

Таблиця 2.8

Images		
Поле	Тип	Призначення
user_id	text	Id користувача
Images_1	text	Посилання на завантажене фото користувачем
Images_2	text	Посилання на завантажене фото користувачем
Images_3	text	Посилання на завантажене фото користувачем
...	...	...

2.3 Проектування та реалізація алгоритмів роботи

Так як наш вебдодаток складається з трьох компонентів, дуже важливо ефективно реалізувати їх взаємодію. На рисунку 2.10 наведено діаграму активності, яка наочно представляє взаємодію між клієнтським додатком, сервером та БД й складається вона з наступних кроків:

1. Продавець або менеджер можуть створити оголошення.
2. Після заповнення дані валідуються на стороні клієнтського додатку.
3. За коректність введених даних слідкує адміністратор.

4. Дані валідуються на стороні серверу й повертається помилка у разі їх неправильності.
5. Авто зберігається у БД.
6. Результат виконання операції відображається користувачу у вигляді створеної пропозиції.

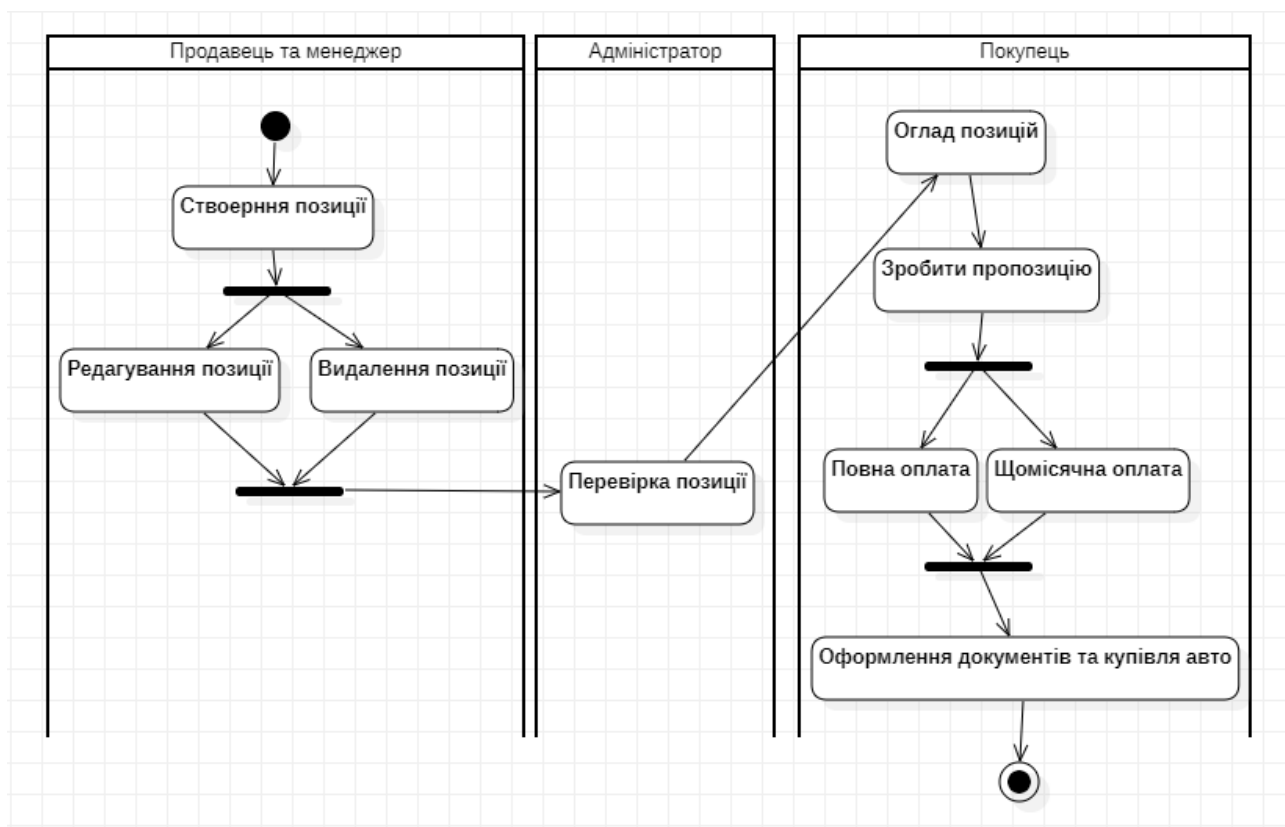


Рисунок 2.3 - Діаграма активності взаємодія користувача з вебдодатком

Отже, в результаті виконання описаних кроків взаємодія всіх компонентів системи дозволяє досягти автоматизації процесу фільтрування, пошуку та замовлення авто.

2.4 Реалізація функціоналу в додаток

Розробка даного додатка вимагає інструментів пошуку, а також створення нових постів.

Визначення фільтраційних параметрів: Спочатку необхідно визначити, які

параметри автомобілів будуть використовуватися для фільтрації. Це можуть бути такі характеристики, як марка, модель, рік випуску, ціновий діапазон, тип кузова, об'єм двигуна, тип палива, пробіг та інші важливі параметри.

Розроблення інтерфейсу фільтра: Створений інтерфейс, де користувачі зможуть встановлювати параметри фільтрації. Це може бути у вигляді випадających списків, перемикачів або полів для введення значень. Наприклад, користувач може встановити марку "BMW", ціновий діапазон від 10000 до 20000 євро та рік випуску після 2015 року.

Застосування фільтра: Після того, як користувач встановив параметри фільтра, додаток повинен виконати функцію фільтрації на основі введених значень. Це означає, що потрібно виконати запит до бази даних або іншого джерела і відфільтрувати автомобілі, які задовольняють встановлені критерії.

Відображення результатів: Після застосування фільтра, додаток повинен відобразити користувачеві результати, які відповідають встановленим параметрам. Це може бути у вигляді списку автомобілів з відображенням їх основних характеристик або відображенням на карті.

Можливість скидання фільтра: Додайте можливість скидання фільтра, щоб користувачі могли легко скинути всі встановлені параметри і переглянути усі доступні автомобілі без обмежень.

Розширені функції фільтрації: Залежно від потреб вашої аудиторії та особливостей ринку, ви можете розширити функціонал фільтрації. Наприклад, додати фільтрацію за країною походження автомобіля, системою безпеки, наявністю спеціальних функцій (навігація, камера заднього виду тощо), кольором кузова та іншими параметрами.

Функціонал по відстеженню за Vin номером:

Інтеграція з базою даних VIN: Перш за все, необхідно мати доступ до бази даних, яка зберігає інформацію про автомобілі та їх VIN-номери. Ця база даних може бути створена самостійно або отримана від стороннього постачальника, який надає такі дані.

Введення VIN-номера: Розроблений інтерфейс, який дозволяє користувачам ввести VIN-номер автомобіля. Це може бути текстове поле або спеціальний блок для введення цифр і літер VIN.

Перевірка VIN-номера: Після введення VIN-номера, додаток повинен перевірити його на правильність формату та перевірити наявність в базі даних. Це можна зробити шляхом виконання запиту до бази даних з використанням введеного VIN-номера.

Відображення результатів: Після отримання результатів запиту, додаток може відобразити інформацію, пов'язану з введеним VIN-номером. Це може включати такі дані, як марка автомобіля, модель, рік випуску, технічні характеристики тощо.

Функціонал по підрахунку загальної суми розмитнення:

Розрахунок акцизного збору: Акцизний збір залежить від різних факторів, таких як тип палива та об'єм двигуна. Використовуючи введені параметри авто, додаток може застосувати відповідні формули або таблиці для розрахунку акцизного збору.

Розрахунок митних зборів: Митні збори можуть базуватися на вартості автомобіля або його вагі. Ви можете використовувати формули або таблиці для розрахунку митних зборів відповідно до правил вашої країни.

Сумування вартості розмитнення: Додаток має здійснити сумування акцизного збору та митних зборів, щоб отримати загальну вартість розмитнення авто.

Відображення результатів: Після розрахунку вартості розмитнення, додаток повинен відобразити результати користувачеві. Це може бути сумарна вартість розмитнення або деталізована інформація про складові витрат.

Висновки до другого розділу

Запровадження функціоналу в додаток для авто з Європи має значний потенціал для покращення користувацького досвіду та зручності у процесі пошуку та розмитнення автомобілів з Європи. Розглянуті функції, такі як пошук

					ІПЗ.КР.Б-121-23-ПЗ	
						33

автомобілів за різними параметрами, пошук за VIN номером та калькулятор розмитнення, пропонують користувачам більш широкі можливості та точність у виборі та оцінці автомобілів з Європи.

Завдяки функції пошуку автомобілів, користувачі можуть легко знайти автомобілі, що відповідають їхнім вимогам і пріоритетам, з використанням фільтрів за маркою, моделлю, роком випуску та іншими параметрами. Це допомагає ефективно вибрати найбільш підходящий автомобіль з великого асортименту.

Функція пошуку за VIN номером надає можливість швидко знайти конкретний автомобіль, якщо користувач вже має VIN номер. Це особливо корисно, коли користувач має інформацію про певний автомобіль, як, наприклад, відомості про модель чи історію транспортного засобу.

Калькулятор розмитнення дозволяє користувачам отримати оцінку вартості розмитнення автомобіля з Європи. Враховуючи різні фактори, такі як вік автомобіля, вартість, об'єм двигуна та інші податкові та митні ставки, цей функціонал допомагає користувачам отримати уявлення про потенційні витрати на розмитнення перед придбанням автомобіля.

Усі ці функції допомагають покупцям автомобілів з Європи бути краще інформованими та зробити обґрунтований вибір. Вони забезпечують зручність, точність та швидкість у процесі пошуку, оцінки та планування витрат, сприяючи успішному завершенню операцій з автомобілями з Європи.

РОЗДІЛ 3: ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ДОДАТКОМ

3.1 Порядок встановлення та налаштування параметрів системи

Встановлення та налаштування системи вимагає розгортання всіх компонентів системи та налаштування як кожного елементу окремо так і загальної системи в цілому.

Для опису архітектури програмного комплексу та розгортання було створено діаграму розгортання (рис. 3.1).

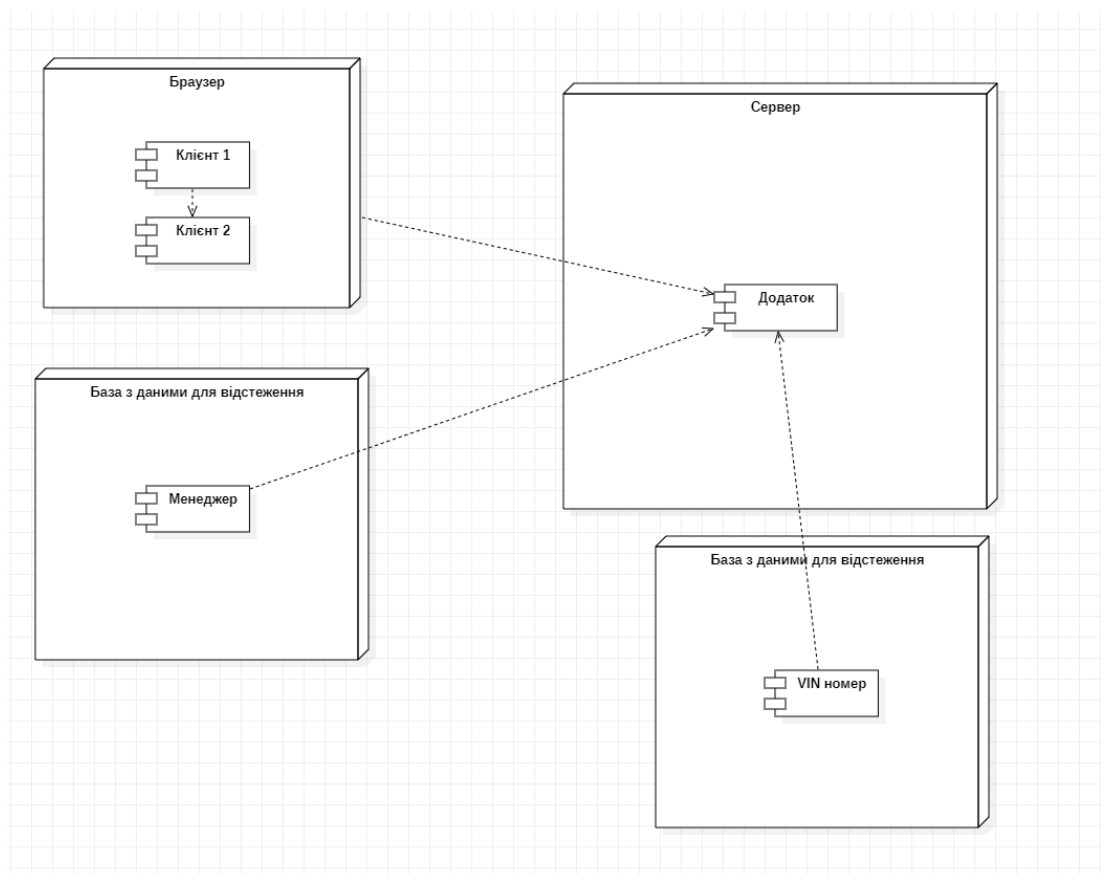


Рисунок 3.1 - Діаграма розгортання додатку

За допомогою змодельованої діаграми можна детально описати програмний комплекс компонентів й процес розгортання.

Вебсервер - частина додатку, яка є вебсервером. Розгортання проекту здійснюється через локальний сервер.

Браузер – представляє собою будь-який сучасний браузер на пристрої користувача. На нього завантажується клієнтська частина додатку (React application) з серверу.

База даних – представляє собою таблицю з інформацією, яка розгортається в форматі Json.

3.2 Структура інтерфейсу

Взаємодія користувача з системою відбувається за допомогою клієнтського додатку. При відкритті сервісу у браузері, користувач бачить перед собою головну сторінку (рис. 3.2).

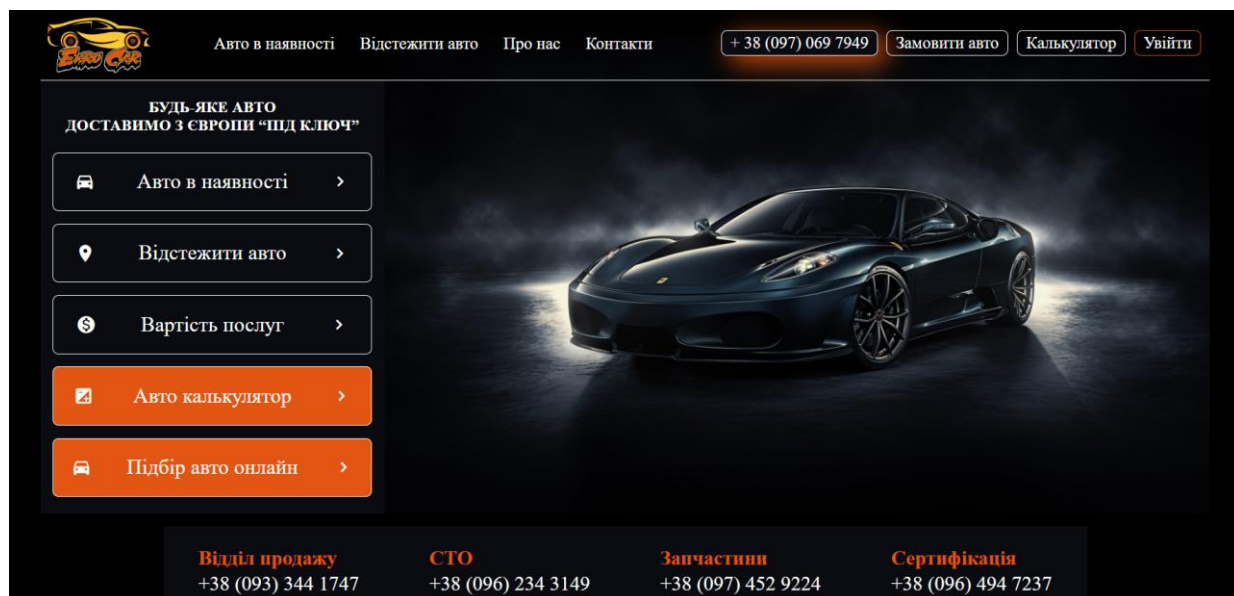


Рисунок 3.2 - Головна сторінка

Після ознайомлення з інформацією користувач може скористатися послугами які не потребують авторизації. Такі як калькулятор, відстеження авто по VIN номеру, фільтрація наявних автомобілей тощо. (рис. 3.3 - рис.3.5)

Калькулятор розрахунку авто з Європи

Наш калькулятор допоможе зорієнтуватися, який автомобіль з Європи вибрати для доставки.

Під час доставки з країн Америки, наприклад, береться умовний каталог, де ціни завищені і не враховують невеликого зносу машини. Так ціна розмитнення завищується, і вам неприємно мати справу з такими операціями. Більше й не потрібно – доставка з Європи буде прозорою та простою, тим більше, ми все зробимо самі.

Калькулятор

Ціна авто (USD)

Рік випуску

Тип двигуна

Об'єм двигуна (см3)/Ємність батареї (кВтг)

Ціна

Ціна авто

Ввізне мито

Акшиз

ПДВ

Загальна вартість

Зателефонувати

Даний розрахунок є орієнтовним

Рисунок 3.3 – Калькулятор розмитнення авто

На цій сторінці можна ознайомитись з інформацією щодо компанії (рис. 3.6 – 3.7).

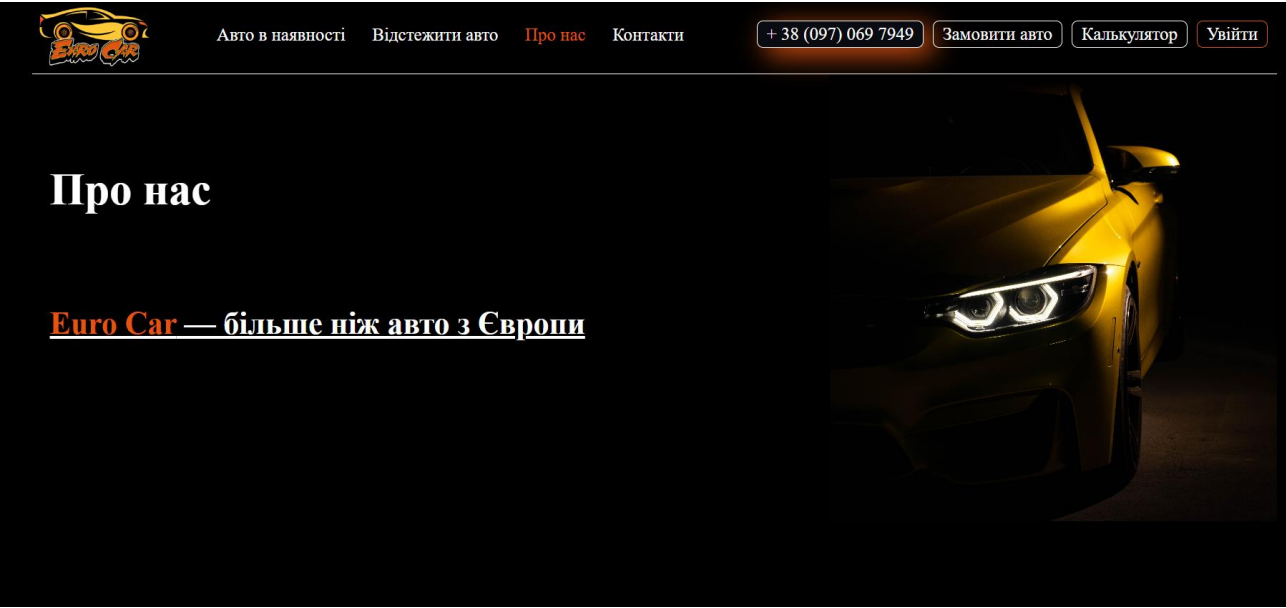


Рисунок 3.7 – Сторінка про компанію

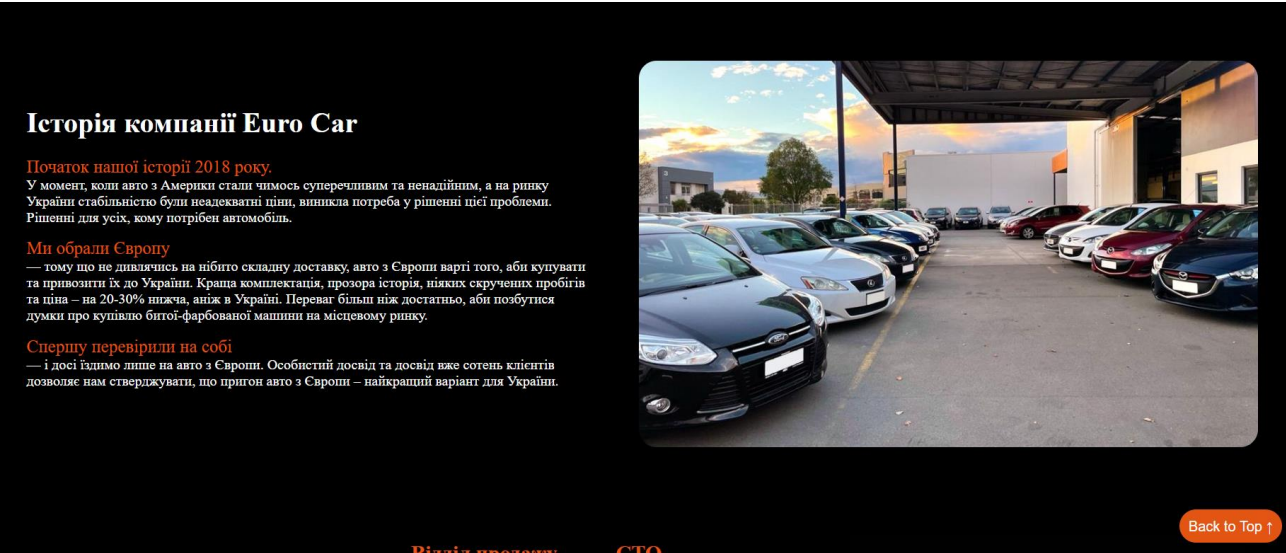


Рисунок 3.8 – Сторінка про компанію

Для того, щоб відкрити детальну інформацію про авто, користувач повинен обрати відповідний елемент списку та натиснути “Детальніше про авто”. Після цього з’явиться сторінка з інформацією про автомобіль (рис. 3.8 – 3.9).



[Авто в наявності](#)
[Відстежити авто](#)
[Про нас](#)
[Контакти](#)

[+ 38 \(097\) 069 7949](#)
[Замовити авто](#)
[Калькулятор](#)
[Увійти](#)



Chevrolet Camaro

20000 \$

70т. км
Бензинбензин
Об'єм 2.4л
2020

механічна
передній привід

Опис:

Виробництво нового Сатаго розпочалося 16 березня 2009 року після кількох років перерви з попереднім поколінням, виробництво якого закінчилося в 2002 році. Сатаго V надійшов у продаж в 2009 році. Автомобіль пропонується з двома модифікаціями двигунів: на Сатаго LT — V6 304 к.с., Сатаго SS — V8 на 400 та 427 к.с.

[Отримати консультацію](#)

Рисунок 3.9 - Детальна інформація про авто

[Отримати консультацію](#)

ТЕХНІЧНИЙ ПАСПОРТ:

Походження:	Європа
Місто:	Київ
Марка:	Chevrolet
Модель:	Сатаго
Рік:	2020
Колір:	чорний
Пошкодження:	Ні
Розмитнений:	Так
Привід:	передній привід
Коробка:	механічна
Паливо:	бензин

КОМПЛЕКТАЦІЯ:

Безпека:	датчик руху
Мультимедіа:	дисплей заднього виду
Комфорт:	клімат-контроль



Euro Car

Відділ продажу
СТО

+38 (093) 344 1747
+38 (096) 234 3149

[Приєднуйся:](#)

[Back to Top ↑](#)

Рисунок 3.10 - Детальна інформація про авто

Сторінка з контактною інформацією містить телефонні номери та місцерозташування компанії. Також, користувач може зв'язатися з компанією через електронну пошту, щоб не очікувати на лінії. Для зручності було сформовано активні посилання, для того щоб здійснювалась переадресація користувача на пошту або в контакти. (рис. 3.10 – 3.11).

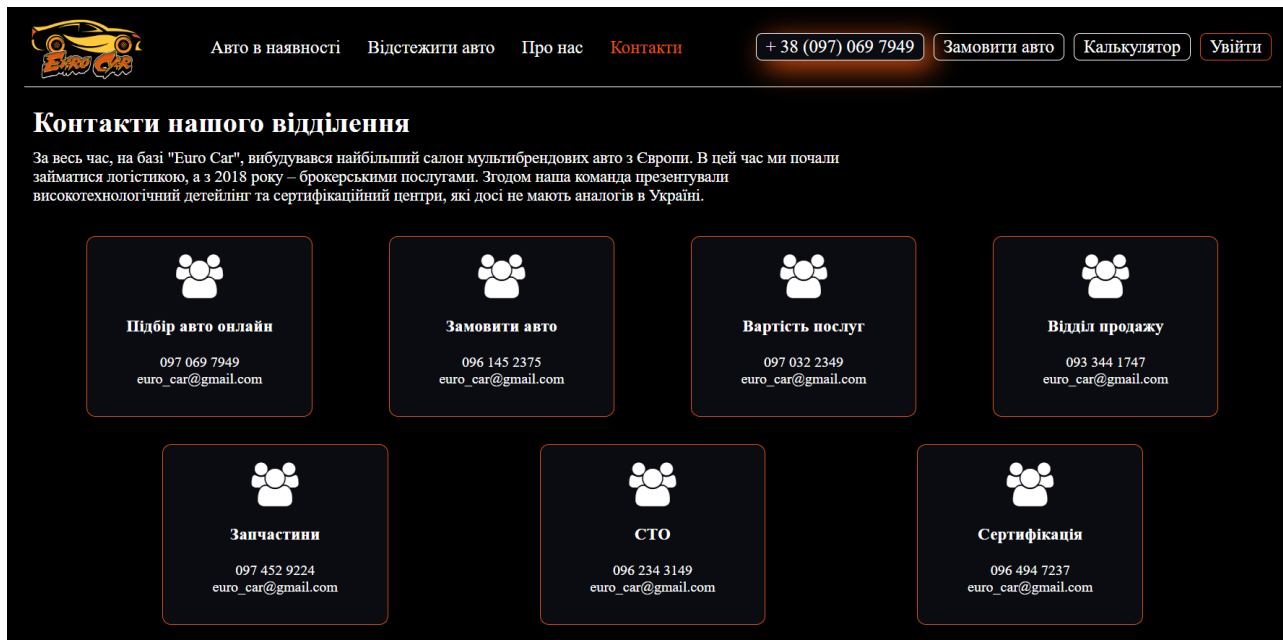


Рисунок 3.11 – Сторінка з контактною інформацією

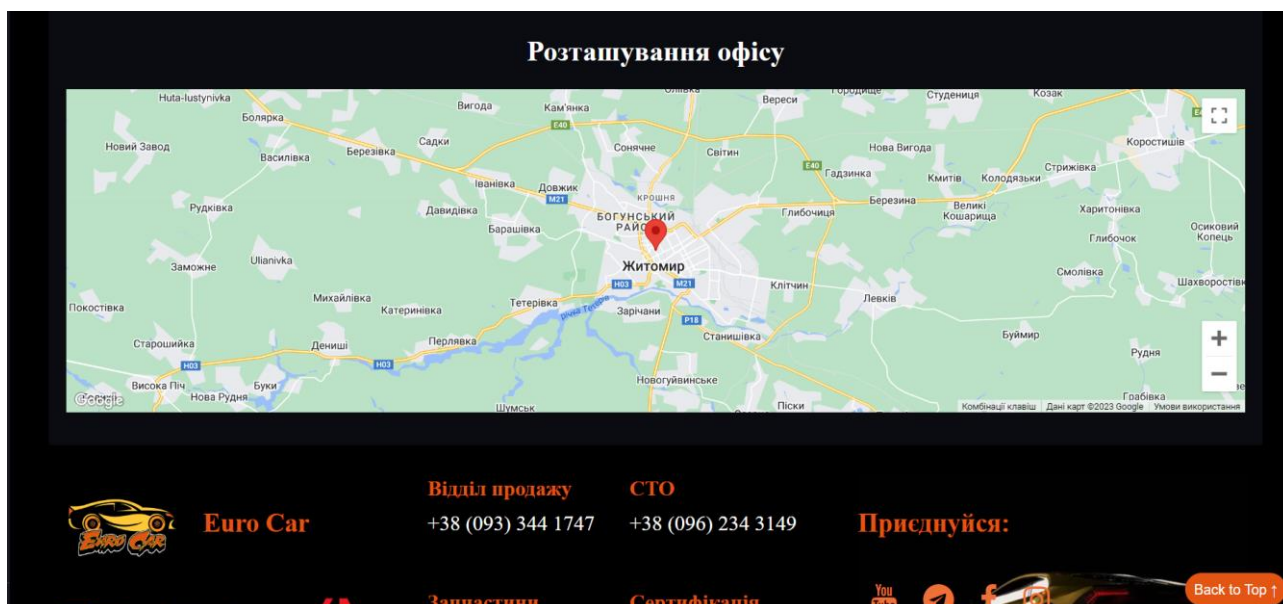


Рисунок 3.12 – Сторінка з контактною інформацією

У випадку якщо користувач хоче ознайомитись з детальною інформацією щодо оформлення замовлення, потрібно буде авторизуватися та увійти до особистого кабінету. Авторизація здійснюється через стандартну сторінку авторизації (рис. 3.12 – 3.13). Якщо користувач не має створеного облікового запису, тоді йому потрібно буде пройти реєстрацію (рис. 3.14 – 3.15). Також дані сторінки мають підказки.

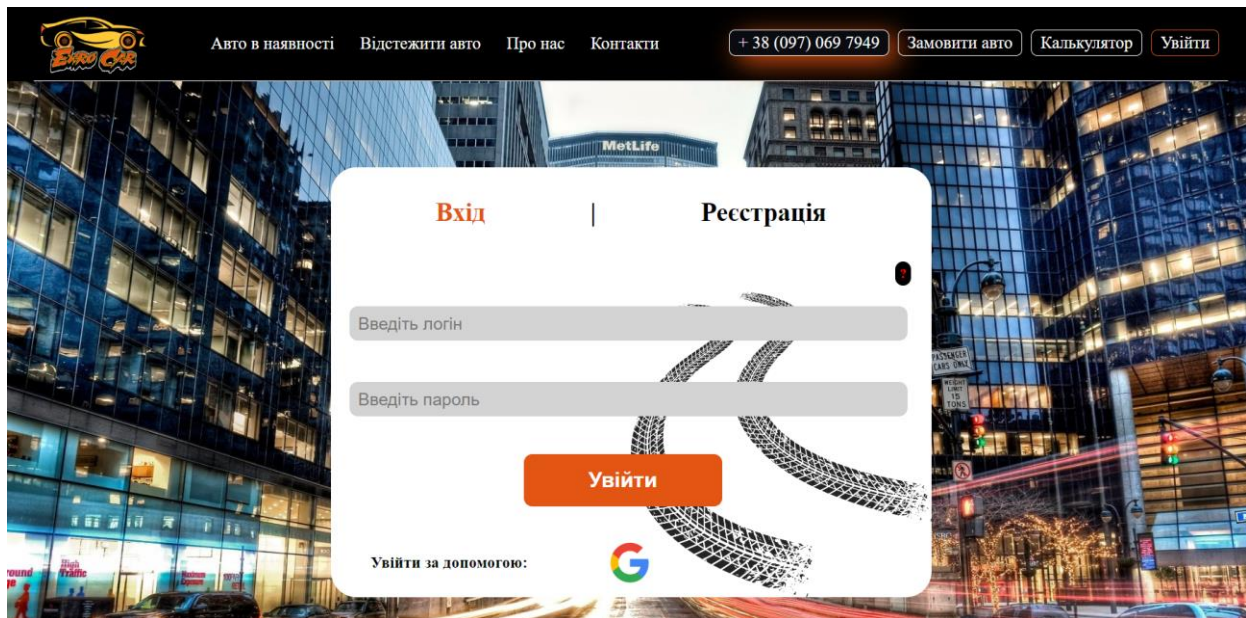


Рисунок 3.13 – Сторінка авторизації

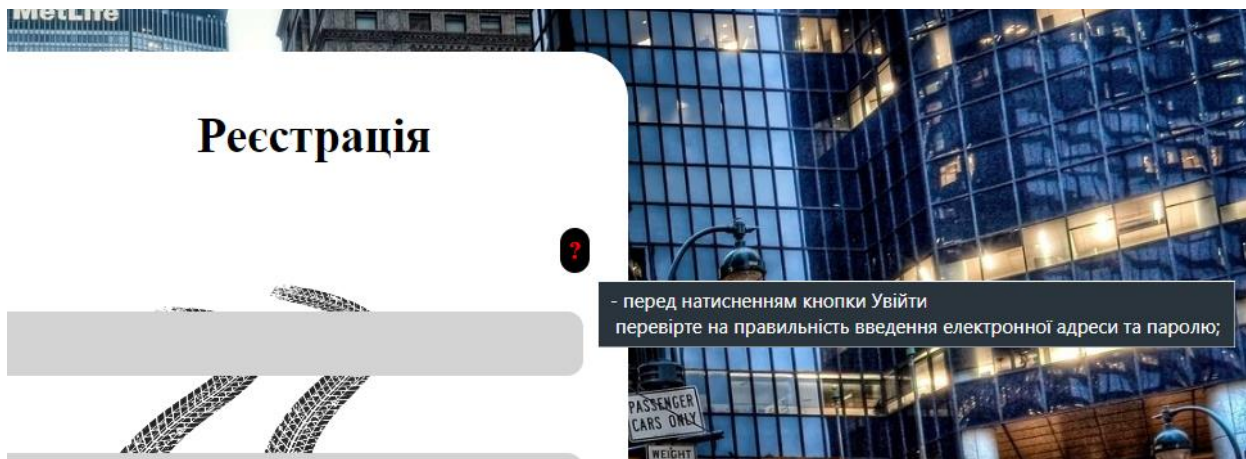


Рисунок 3.14 – Сторінка авторизації з підказкою для користувача

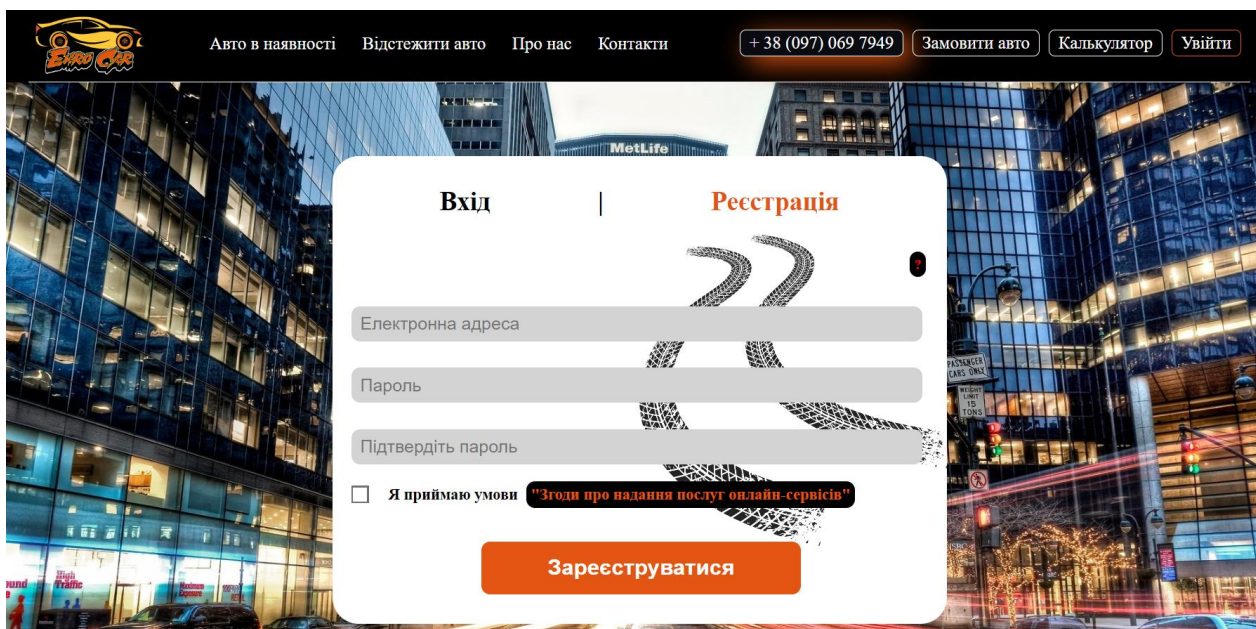


Рисунок 3.15 – Сторінка реєстрації



Рисунок 3.16 – Сторінка реєстрації з підказкою для користувача

Також додаток дозволяє скористатися авторизацією через Google. На сьогодні ця функція є дуже зручною та користується популярністю (рис. 3.17 – 3.18).

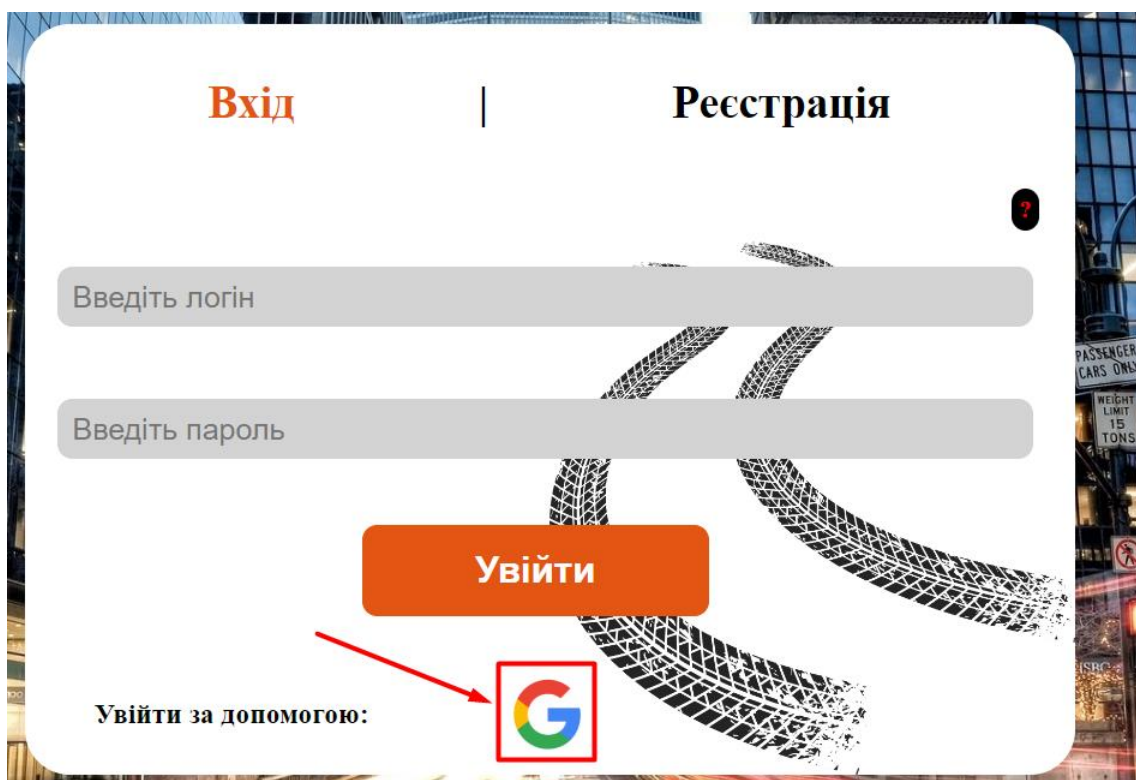


Рисунок 3.17 – Авторизація через Google

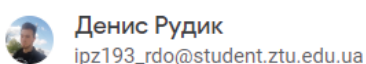


Рисунок 3.18 – Авторизація через Google

Після авторизації в вебдодатку буде відображатися аватар у верхньому правому куті (рис. 3.19).

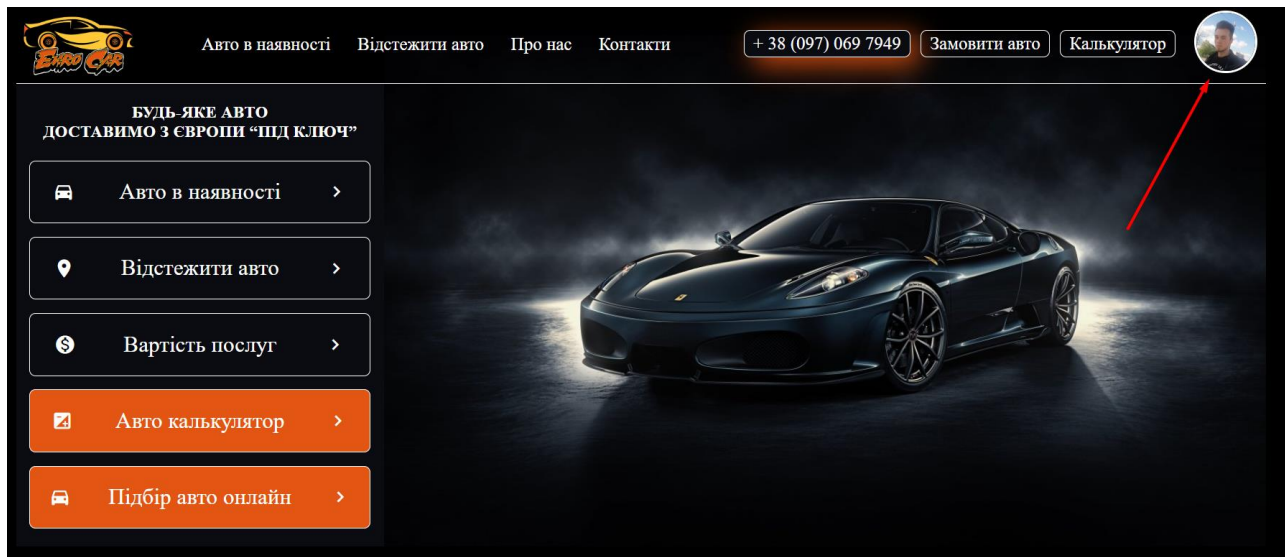


Рисунок 3.19 – Відображення аватара після авторизації

Після успішної авторизації користувач може перейти до особистого кабінету(рис. 3.20)

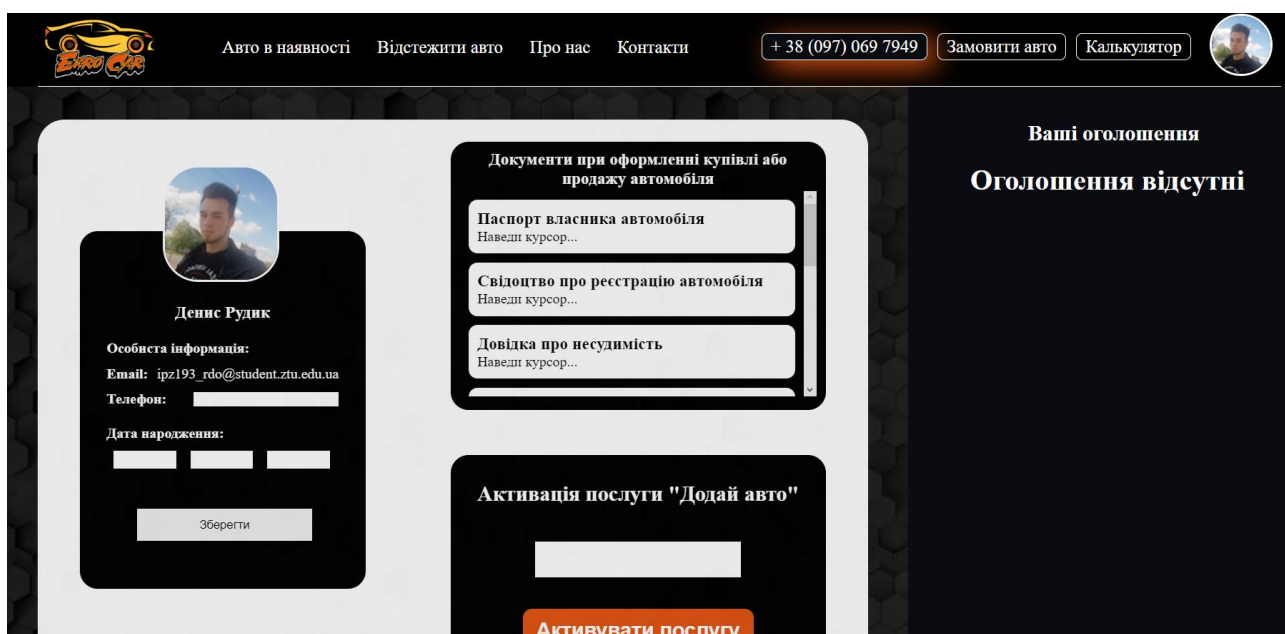


Рисунок 3.20 – Особистий кабінет

В особистому кабінеті можна ознайомитись з інформацією щодо купівлі або продажу авто (рис. 3.21).

Рисунок 3.21 – Інформація щодо купівлі або продажу авто

Процес створення оголошення відбувається після активування послуги “Додай авто”. Щоб скористатися цією послугою користувач повинен мати код доступу (рис. 3.22). Після активації загориться кнопка “Додати авто” (рис. 3.23-3.24).

Рисунок 3.22 – Доступ до послуги

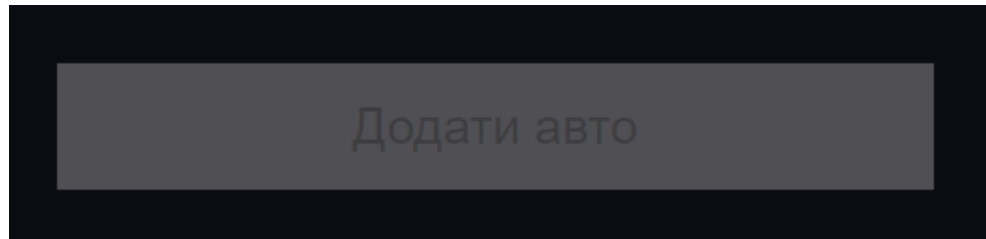


Рисунок 3.23 – Кнопка до активації

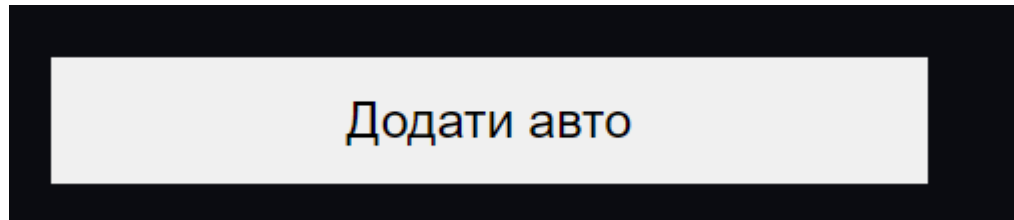


Рисунок 3.24 – Кнопка після активації

Таким чином користувач отримує права адміністратора та може додавати свої оголошення. Для створення та розміщення оголошення переходимо на стрінку натиснувши кнопку “Додати авто” (рис. 3.25 – 3.26)

The screenshot shows the 'Stvoriti oghoshennia' (Create advertisement) form on the Euro Car website. The form is titled 'Створити оголошення' with a subtitle 'Опишіть у подробицях'. It features a green button labeled 'Виберіть файл'. The form fields are: 'Ціна' (Price) with the value '20000', 'Опис' (Description) with the placeholder 'Опишіть своє авто', 'Категорія' (Category) with a dropdown menu showing '--- Оберіть категорію ---', and 'Походження авто' (Origin of the car). The website header includes a logo, navigation links ('Авто в наявності', 'Відстежити авто', 'Про нас', 'Контакти'), a phone number '+ 38 (097) 069 7949', and buttons for 'Замовити авто' and 'Калькулятор'. A user profile picture is also visible in the top right corner.

Рисунок 3.25 – Форма для заповнення

Походження авто
--- Оберіть походження ---

Місто перебування
--- Оберіть місто ---

Марка
--- Оберіть марку ---

Модель
[Input field]

Рік виготовлення
--- Оберіть рік ---

Колір
--- Оберіть колір ---

Пошкодження
--- Авто перебувало в ДТП ---

Розмитнено
--- Розмитнене авто ---

Привід
[Input field]

Back to Top ↑

Рисунок 3.26 – Форма для заповнення

Після створення оголошення здійснюється переадресація на профіль користувача. Можна помітити що створене оголошення з'явилося (рис. 3.27).

Документи при оформленні купівлі або продажу автомобіля

Паспорт власника автомобіля
Наведи курсор...

Свідоцтво про реєстрацію автомобіля
Наведи курсор...

Довідка про несудимість
Наведи курсор...

Активація послуги "Додай авто"

[Input field]

Активувати послугу

Ваші оголошення

[Image of a car]

Редагувати Видалити

Рисунок 3.26 – Відображення створеного оголошення

Далі продавець може відредагувати або видалити створене оголошення, якщо воно стало не актуальним. Редагування відбувається наступним чином:

- Натискаємо на “Редагувати” та переходимо на сторінку редагування (рис. 3.27).
- Після цього педредаговується потрібні поля та зберігається

The screenshot shows the 'Оновити оголошення' (Update Ad) form on the 'Expo Car' website. The form is titled 'Оновити оголошення' with a subtitle 'Опишіть у деталях'. It features a green button 'Выберите файл' (Choose file). The form includes the following fields:

- Ціна:** 25000 (Price)
- Опис:** Якийсь опис... (Description: Some description...)
- Категорія:** Автобуси (Category: Buses)
- Походження авто:** Європа (Car origin: Europe)

Рисунок 3. 27– Редагування оголошення

3.3 Тестування роботи програмного комплексу

Важливим етапом розробки програмного забезпечення є процес його тестування та виправлення дефектів. Для перевірки коректності роботи системи виконаємо мануальне тестування.

1. Вхід у обліковий запис:

передумови: користувач знаходиться на формі авторизації;

кроки: введення облікових даних, натиснення на кнопку ”Увійти”

очікуваний результат: користувач буде перенаправлений на головну сторінку;

фактичний результат: користувач був перенаправлений на головну сторінку

2. Перехід на сторінку створення оголошення:

передумови: користувач аутентифікований у системі;

кроки: перехід на сторінку створення оголошення за допомогою вибору пункту “Додати авто”

очікуваний результат: користувачу відкривається форма створення оголошення;

фактичний результат: користувачу відкрилася форма створення оголошення;

3. Перегляд оголошення:

передумови: користувач створив оголошення та знаходиться на сторінці “Авто в наявності”;

кроки: користувач знаходить створене оголошення та натискає на нього

очікуваний результат: відкривається інформація про оголошення.

фактичний результат: : відкрилася інформація про оголошення.

4. Редагування оголошення:

передумови: користувач відредагував оголошення та знаходиться на сторінці “Авто в наявності”;

кроки: користувач відкриває особисту сторінку та натискає “Редагувати” в обраному оголошенні.

очікуваний результат: користувачу відкривається форма оновлення оголошення.

фактичний результат: користувачу відкривається форма оновлення оголошення.

5. Фільтрування оголошень:

передумови: користувач перейшов на сторінку оголошень;

кроки: користувач вводить дані для фільтрації;

очікуваний результат: оголошення відфільтрується за запитом користувача;

фактичний результат: оголошення відфільтрувалось за запитом користувача;

Висновки до третього розділу

В даному розділі було наведено схему будови та процесу розгортання кожного компонента додатку. Був проведений опис кожного структурного елемента та його взаємодію з іншими частинами системи.

Проведено опис здійснення редагування та видалення оголошень. Даний функціонал є важливою функціональністю системи автомобілів з Європи. Це дозволяє користувачам змінювати та оновлювати інформацію про автомобілі, яку вони розміщують на сайті.

Фільтрування оголошень є корисною можливістю для покупців, оскільки вони можуть швидко та ефективно звузити вибір автомобілів за такими параметрами, як ціновий діапазон, бренд, модель, розмір двигуна та інші характеристики. Це спрощує пошук та допомагає знайти автомобіль, який відповідає потребам покупця.

Можливість відстежувати автомобілі за VIN номером є цінним інструментом для перевірки історії автомобіля. Це дозволяє покупцям отримати доступ до інформації. Відстеження за VIN номером допомагає забезпечити прозорість і довіру у процесі купівлі автомобіля.

Компанія, яка веде даний додаток, є ключовим фактором успіху. Важливо мати довіру до компанії, що має добру репутацію, забезпечує якісний сервіс та надає надійну інформацію про автомобілі.

Наявність гарячої лінії є важливим засобом забезпечення взаємодії з клієнтами. Гаряча лінія дозволяє покупцям задавати запитання, отримувати консультацію, роз'яснення щодо процесу купівлі, послуг, умов гарантії тощо. Це створює сприятливу спілкувальну платформу між клієнтами та сервісом.

Загалом, додаток автомобілів з Європи, який надає можливість редагування та видалення оголошень, фільтрування оголошень, відстеження за VIN номером, відображає надійність та є зручним вибором для покупців.

					ІПЗ.КР.Б-121-23-ПЗ	
						49

ВИСНОВКИ

Під час виконання випускної роботи бакалавра було спроектовано та реалізовано систему з вирішення логістичних проблем на прикладі купівлі та продажі автомобілів з Європейського Союзу. Для цього було вирішено наступні завдання:

1. Проведений аналіз предметної області дозволив визначити основні вимоги до додатку. На основі аналізу було визначено набір можливостей програмного продукту для забезпечення автоматизації процесу продажу авто.

2. Так як розробка конкурентоспроможного вебдодатку неможлива без аналізу його конкурентів, було проведено порівняння додатків, які найбільш наближені до нашого. Як результат, був сформований набір важливих функцій для нашого продукту.

3. Система складається з декількох компонентів, її якість залежить від якості кожної складової. В результаті для кожного компоненту системи була обрана та архітектура, яка найбільше відповідає його особливостям.

4. Відповідно до обраної архітектури та поставлених вимог був обраний той чи інший засіб для реалізації проекту.

5. Був описаний алгоритм взаємодії компонентів системи та логіки виконання операцій з даними.

6. Також був проаналізовано алгоритм реалізації системи. Проведений аналіз проблем поставленого завдання й знайдено відповідне архітектурне рішення. Також описано методи обробки даних з метою покращення ефективності системи.

7. Наведено детальний опис схеми системи, їх призначення та процес розгортання продукту.

8. Проведено опис інтерфейсу додатку, його структурних елементів, функціональних можливостей у відповідності до ролі користувача та інструменти роботи з даними.

9. Було проведено тестування додатку з метою перевірки правильності роботи системи.

					ІПЗ.КР.Б-121-23-ПЗ	
						50

Реалізований програмний комплекс готовий до використання. Система може бути використана для продажу автомобілів з ЄС.

В подальшому даний програмний продукт може удосконалюватися шляхом підтримки історії пошуку, автомобілі які знаходяться в тренді тощо.

					ІІЗ.КР.Б-121-23-ІІЗ	
						51

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. “AJAX and APIs”, Reactjs.org, 2023 [Електроний ресурс]. Доступ за посиланням: <https://legacy.reactjs.org/docs/faq-ajax.html#gatsby-focus-wrapper> [Дата звернення: 07.04.2023]
2. “Boss Auto”, Bossautoukraine.com.ua, 2023 [Електроний ресурс]. Доступ за посиланням: <https://bossautoukraine.com.ua> [Дата звернення: 09.04.2023]
3. “Auto Terminal”, Bossautoukraine.com.ua, 2023 [Електроний ресурс]. Доступ за посиланням: <https://aterminal.com.ua> [Дата звернення: 10.04.2023]
4. “Komis Auto”, Komis.com.ua, 2023 [Електроний ресурс]. Доступ за посиланням: <https://www.komis.com.ua> [Дата звернення: 11.04.2023]
5. “Найкращі практики проєктування REST API”, Codeguida.com, 2023 [Електроний ресурс]. Доступ за посиланням: <https://codeguida.com/post/2462> [Дата звернення: 14.04.2023]
6. “AJAX та запити до API”, Reactjs.org, 2023 [Електроний ресурс]. Доступ за посиланням: <https://uk.legacy.reactjs.org/docs/faq-ajax.html> [Дата звернення: 17.04.2023]
7. “Посібник: знайомство з React”, Reactjs.org, 2023 [Електроний ресурс]. Доступ за посиланнями: <https://uk.legacy.reactjs.org/tutorial/tutorial.html> [Дата звернення: 19.04.2023]
8. “React JavaScript-бібліотека для створення користувацьких інтерфейсів”, Reactjs.org, 2023 [Електроний ресурс]. Доступ за посиланням: <https://uk.legacy.reactjs.org> [Дата звернення: 21.04.2023]
9. “MobX”, 2023 Доступ за посиланням: <https://mobx.js.org/README.html> [Дата звернення: 22.04.2023]
10. “How to Fetch Data in React: Cheat Sheet + Examples”, Reencodecamp.org, 2023 [Електроний ресурс]. Доступ за посиланнями: <https://www.freecodecamp.org/news/fetch-data-react/> [Дата звернення: 23.04.2023]

11. “Components and Props”, Reactjs.org, 2023 [Електроний ресурс]. Доступ за посиланням: <https://legacy.reactjs.org/docs/components-and-props.html> [Дата звернення: 02.04.2023]
12. “Code editing. Redefined.”, Visualstudio.com, 2023 Доступ за посиланням: <https://code.visualstudio.com> [Дата звернення: 27.04.2023]
13. “React Router DOM”, Reactrouter.com, 2023 [Електроний ресурс]. Доступ за посиланням: <https://reactrouter.com/en/main> [Дата звернення: 30.04.2023]
14. “Google-map-react”, Npmjs.com, 2023 [Електроний ресурс]. Доступ за посиланням: <https://www.npmjs.com/package/google-map-react> [Дата звернення: 2.05.2023]
15. “@supabase/auth-helpers-react”, Npmjs.com, 2023 [Електроний ресурс]. Доступ за посиланням: <https://www.npmjs.com/package/@supabase/auth-helpers-nextjs> [Дата звернення: 04.05.2023]
16. “Supabase is an open source Firebase alternative for building secure and performant Postgres backends with minimal configuration.”, Supabase.com, 2023 [Електроний ресурс]. Доступ за посиланням: <https://supabase.com> [Дата звернення: 07.05.2023]
17. “Що таке VIN номер і як він допоможе врятуватися від автошахраїв”, Youcontrol.com.ua, 2023 [Електроний ресурс]. Доступ за посиланням: <https://youcontrol.com.ua/topics/shcho-take-vin-nomer-iyak-vin-dopomozhe-vriatuvatysia-vid-avtoshakhrayiv/> [Дата звернення: 10.05.2023]

ДОДАТКИ

					ІІЗ.КР.Б-121-23-ІІЗ	
						54

1. Index.jsx:

```

2. import React from "react";
3. import "./index.css";
4. import MainCar from "../../assets/1579155590_31-39.jpg";
5. import ShortInfo from "../../assets/short_info_img.jpg";
6. import BanerCars from "../../assets/1579155590_31-39.jpg";
7. import { NavLink, useNavigate } from "react-router-dom";
8.
9. import MultiCarousel from "../../components/MultiCarousel/MultiCarousel";
10.
11. const Index = () => {
12.   const navigate = useNavigate();
13.   return (
14.     <>
15.       <section>
16.         <div className="container">
17.           <header>
18.             <div className="left-main_menu">
19.               <h3>
20.                 БУДЬ-ЯКЕ АВТО
21.               <br />
22.                 ДОСТАВИМО З ЄВРОПИ “ПІД КЛЮЧ”
23.             </h3>
24.             <ul>
25.               <a href="#">
26.                 <NavLink to="/stock">
27.                   <li>
28.                     <i className="large material-
29. icons">directions_car</i>Авто
30.                     в наявності
31.                     <i className="large material-icons">chevron_right</i>
32.                   </li>
33.                 </NavLink>
34.               <a href="#">
35.                 <NavLink to="/tracking">
36.                   <li>
37.                     <i className="large material-icons">place</i>Відстежити
38.                     авто
39.                     <i className="large material-icons">chevron_right</i>
40.                   </li>
41.                 </NavLink>
42.               <a href="#">
43.                 <NavLink to="/contacts">
44.                   <li>
45.                     <i className="large material-icons">monetization_on</i>
46.                     Вартість послуг
47.

```

```

48.         <i className="large material-icons">chevron_right</i>
49.     </li>
50. </NavLink>
51. </a>
52. <a href="#">
53.     <NavLink to="/calculator">
54.         <li className="last_btn">
55.             <i className="large material-icons">exposure</i>Авто
56.             калькулятор
57.             <i className="large material-icons">chevron_right</i>
58.         </li>
59.     </NavLink>
60. </a>
61. <a href="#">
62.     <NavLink to="/contacts">
63.         <li className="last_btn">
64.             <i className="large material-icons">directions_car</i>
65.             Підбір авто онлайн
66.             <i className="large material-icons">chevron_right</i>
67.         </li>
68.     </NavLink>
69. </a>
70. </ul>
71. </div>
72. <img className="img_main-car" src={MainCar} />
73. </header>
74. <div className="contacts_header">
75.     <ul>
76.         <li>
77.             Відділ продажу
78.             <br />
79.             <span>+38 (093) 344 1747</span>
80.         </li>
81.         <li>
82.             СТО
83.             <br />
84.             <span>+38 (096) 234 3149</span>
85.         </li>
86.         <li>
87.             Запчастини
88.             <br />
89.             <span>+38 (097) 452 9224</span>
90.         </li>
91.         <li>
92.             Сертифікація
93.             <br />
94.             <span>+38 (096) 494 7237</span>
95.         </li>
96.     </ul>
97. </div>
98. </div>

```

```

99.         </section>
100.        <section>
101.            <div className="container">
102.                <div className="short_info">
103.                    <div className="choose_btn-div">
104.                        <h2>ТВОЯ МРІЯ СТАЛА ЩЕ ДОСТУПНІШОЮ!</h2>
105.                        <p>
106.                            АДЖЕ EURO CAR
107.                            <br />
108.                            ПОКРИВАЄ УВЕСЬ ДІАПАЗОН КУПІВЛІ,
109.                            <br />
110.                            ДОСТАВКИ ТА ОФОРМЛЕННЯ АВТО
111.                        </p>
112.                        <NavLink to="/contacts">
113.                            <button>Підібрати авто в один клік</button>
114.                        </NavLink>
115.                    </div>
116.                    <div className="short_info-li">
117.                        <ul>
118.                            <li>
119.                                АВТОМОБІЛІ З ЄВРОПИ
120.                                <span className="large material-icons">check</span>
121.                            </li>
122.                            <li>
123.                                ДОСТАВКА<span className="large material-
124.                                    icons">check</span>
125.                            </li>
126.                            <li>
127.                                РОЗМИТНЕННЯ<span className="large material-
128.                                    icons">check</span>
129.                            </li>
130.                            <li>
131.                                РЕМОНТ<span className="large material-
132.                                    icons">check</span>
133.                            </li>
134.                            <li>
135.                                СЕРТИФІКАЦІЯ
136.                                <span className="large material-icons">check</span>
137.                            </li>
138.                        </ul>
139.                    </div>
140.                </div>
141.                <img src={ShortInfo} alt="short_info_img" />
142.            </div>
143.        </section>
144.        <section>
145.            <div className="container">
146.                <div className="div_offers">
147.                    <div className="info_offers">
148.                        <h2 className="main_h2">
149.                            <span>EURO CAR</span>

```

```

147.      <br /> НАДІЙНА ДОСТАВКА АВТО З ЄВРОПИ ТА ДОСТАВКА У
        ВАШЕ МІСТО
148.      </h2>
149.      <div className="blocks_offers">
150.      <div className="offer">
151.      <h2 className="offer_h2_main">
152.      КУПИТИ АВТО
153.      <br />3 ЄС
154.      </h2>
155.      <p>ЗАМОВЛЯЮЧИ АВТО ЧЕРЕЗ EURO CAR, ВИ ЕКОНОМИТЕ
        25%.</p>
156.      <h2>У ВАРТІСТЬ ВКЛЮЧЕНО:</h2>
157.      <ul>
158.      <li>- АУКЦІОННА ЦІНА АВТО</li>
159.      <li>- ДОСТАВКА АВТО</li>
160.      <li>- РОЗМИТНЕННЯ ТА СЕРТИФІКАЦІЯ АВТО</li>
161.      <li>- РЕМОНТ АВТО ТА ХІМЧИСТКА</li>
162.      </ul>
163.      <button>Отримати консультацію</button>
164.      </div>
165.      <div className="offer">
166.      <h2 className="offer_h2_main">
167.      КУПИТИ РОЗМИТНЕНЕ
168.      <br />
169.      АВТО
170.      </h2>
171.      <p>
172.      ЗПРОДАЄМО РОЗМИТНЕНІ АВТО ІЗ ЄВРОПИ ЗА ДОСТУПНИМИ
        ЦІНАМИ.
173.      </p>
174.      <h2>У ВАРТІСТЬ ВКЛЮЧЕНО:</h2>
175.      <ul>
176.      <li>- АУКЦІОННА ЦІНА АВТО</li>
177.      <li>- ДОСТАВКА АВТО</li>
178.      <li>- РОЗМИТНЕННЯ ТА СЕРТИФІКАЦІЯ АВТО</li>
179.      <li>- РЕМОНТ АВТО ТА ХІМЧИСТКА</li>
180.      </ul>
181.      <button>Отримати консультацію</button>
182.      </div>
183.      <div className="offer">
184.      <h2 className="offer_h2_main">
185.      ПРОДАЙ СВОЄ АВТО З
186.      <br />
187.      ЄВРОПИ
188.      </h2>
189.      <p>У ВАС Є МОЖЛИВІСТЬ ПРОДАТИ СВОЄ ПРИВЕЗЕНЕ АВТО З
        ЄС</p>
190.      <h2>У ВАРТІСТЬ ВКЛЮЧЕНО:</h2>
191.      <ul>
192.      <li>- ВАРТІСТЬ ПОЛУГ EURO CAR</li>
193.      <li>- ДОСТАВКА АВТО</li>

```

```

194.         <li>- РОЗМИТНЕННЯ ТА СЕРТИФІКАЦІЯ АВТО</li>
195.         <li>- РЕМОНТ АВТО ТА ХІМЧИСТКА</li>
196.     </ul>
197.     <button>Отримати консультацію</button>
198. </div>
199. </div>
200. </div>
201. </div>
202. </div>
203. </section>
204. <section>
205.     <div className="container">
206.         <div className="car_slider_box">
207.             <h2 className="main_h2">ОТРИМАЙ БАЖАНИЙ АВТОМОБІЛЬ В ОДИН
                КЛИК</h2>
208.             <div className="slider">
209.
210.                 <MultiCarousel />
211.
212.                 <button className="more_cars" onClick={() =>
                    navigate("/stock")}>
213.                     Більше авто{" "}
214.                 </button>
215.             </div>
216.         </div>
217.     </div>
218. </section>
219. </>
220. );
221. };
222.
223. export default Index;
224.

```

4. AddNewPost.jsx:

```

5. import React, { useState } from "react";
6. import "./NewPost.css";
7.
8. import {
9.     carBrand,
10.    carCategory,
11.    cityCar,
12.    colorCar,
13.    comfortCar,
14.    fuelCar,
15.    multimediaCar,
16.    securityCar,
17.    transmissionCar,
18.    wheelDriveCar,

```

```

19.
20.   yearCar,
21. } from "../AddPostInfo";
22. import { supabase } from "../../components/SupabaseConf/Supabase";
23. import { useEffect } from "react";
24. import { useNavigate } from "react-router-dom";
25.
26. const AddNewPost = () => {
27.   // imgInfo__
28.   const navigate = useNavigate();
29.
30.   const [imgInfo, setImgInfo] = useState(null);
31.   const [post, setPost] = useState({
32.     car_price: "",
33.     car_description: "",
34.     type_of_transport: "",
35.     origin: "",
36.     city: "",
37.     brand: "",
38.     model: "",
39.     year: "",
40.     color: "",
41.     injured: "",
42.     customs_cleared: "",
43.     wheel_drive: "",
44.     transmission: "",
45.     security: "",
46.     multimedia: "",
47.     comfort: "",
48.     fuel: "",
49.     engine: "",
50.     range: "",
51.     img_url: "",
52.     img_link: "",
53.     user_id: "",
54.   });
55.
56.   useEffect(() => {
57.     setPost({ ...post, img_url: imgInfo?.target.files[0].name });
58.     imgLink(imgInfo); ////
59.   }, [imgInfo]);
60.
61.   async function uploadFile(event) {
62.     const {
63.       data: { user },
64.     } = await supabase.auth.getUser();
65.     // console.log(user.id);
66.
67.     const mainIMG = event.target.files[0];
68.     // console.log(avatarFile);
69.

```

```

70.     const { data, error } = await supabase.storage
71.       .from("images")
72.       .upload(user.id + "/" + mainIMG.name, mainIMG);
73.     // console.log(data);
74.   }
75.
76.   async function imgLink(event) {
77.     const {
78.       data: { user },
79.     } = await supabase.auth.getUser();
80.     const mainIMG = event?.target.files[0];
81.     const { data } = await supabase.storage
82.       .from("images")
83.       .getPublicUrl(user.id + "/" + mainIMG?.name);
84.     setPost(
85.       // { ...post, user_id: user.id },
86.       { ...post, img_link: data.publicUrl, user_id: user.id }
87.     );
88.     console.log(data.publicUrl);
89.   }
90.
91.   // end imgInfo__
92.
93.   async function InsertRows() {
94.     // console.log(post);
95.     uploadFile(imgInfo);
96.     const { data, error } = await supabase.from("cars").insert([post]);
97.     navigate("/profile");
98.   }
99.   console.log(post);
100.
101.   return (
102.     <>
103.     <div className="container">
104.       <div className="post-header">
105.         <div className="layout_post">
106.           <div className="h_post">
107.             <h1>Створити оголошення</h1>
108.             <p>Опишіть у подробицях</p>
109.           </div>
110.
111.           <div className="input-file-container">
112.             <input
113.               className="input-file"
114.               id="my-file"
115.               type="file"
116.               onChange={(e) => setImgInfo(e)}
117.             />
118.             <label tabIndex="0" for="my-file" className="input-file-
119.               trigger">

```

Виберіть файл

```

120.         </label>
121.     </div>
122.     <p className="file-return"></p>
123.
124.     <div className="price_post_car margin_post_car">
125.         <h3>Ціна </h3>
126.         <input
127.             type="number"
128.             placeholder="20000"
129.             value={post.car_price}
130.             onChange={(e) =>
131.                 setPost({ ...post, car_price: e.target.value })
132.             }
133.         />
134.     </div>
135.
136.     <div className="text_post_car margin_post_car">
137.         <h3>Опис</h3>
138.         <textarea
139.             value={post.car_description}
140.             placeholder="Опишіть своє авто"
141.             onChange={(e) =>
142.                 setPost({ ...post, car_description: e.target.value })
143.             }
144.         ></textarea>
145.     </div>
146.     <div className="category_post_car margin_post_car">
147.         <h3>Категорія</h3>
148.         <select
149.             value={post.type_of_transport}
150.             onChange={(e) =>
151.                 setPost({ ...post, type_of_transport: e.target.value
152.             })
153.             >
154.             <option value="" defaultValue hidden>
155.                 --- Оберіть категорію ---
156.             </option>
157.             {carCategory.map((type_of_transport) => (
158.                 <option>{type_of_transport}</option>
159.             ))}
160.         </select>
161.     </div>
162.     <div className="origin_post_car margin_post_car">
163.         <h3>Походження авто</h3>
164.         <select
165.             value={post.origin}
166.             onChange={(e) => setPost({ ...post, origin:
167.                 e.target.value })}
168.         >
169.         <option value="" defaultValue hidden>

```

```

169.         --- Оберіть походження ---
170.         </option>
171.         <option>Європа</option>
172.     </select>
173. </div>
174. <div className="city_post_car margin_post_car">
175.     <h3>Місто перебування </h3>
176.     <select
177.         value={post.city}
178.         onChange={(e) => setPost({ ...post, city:
179.             e.target.value })}
180.     >
181.         <option value="" defaultValue hidden>
182.             --- Оберіть місто ---
183.             </option>
184.             {cityCar.map((city) => (
185.                 <option>{city}</option>
186.             ))}
187.         </select>
188.     </div>
189. <div className="brand_post_car margin_post_car">
190.     <h3>Марка</h3>
191.     <select
192.         value={post.brand}
193.         onChange={(e) => setPost({ ...post, brand:
194.             e.target.value })}
195.     >
196.         <option value="" defaultValue hidden>
197.             --- Оберіть марку ---
198.             </option>
199.             {carBrand.map((brand) => (
200.                 <option>{brand}</option>
201.             ))}
202.         </select>
203.     </div>
204. <div className="model_post_car margin_post_car">
205.     <h3>Модель</h3>
206.     <input
207.         type="text"
208.         onChange={(e) => setPost({ ...post, model:
209.             e.target.value })}
210.     />
211. </div>
212. <div className="year_post_car margin_post_car">
213.     <h3>Рік виготовлення</h3>
214.     <select
215.         value={post.year}
216.         onChange={(e) => setPost({ ...post, year:
217.             e.target.value })}
218.     >

```

```

216.         <option value="" defaultValue hidden>
217.             --- Оберіть рік ---
218.         </option>
219.         {yearCar.map((year) => (
220.             <option>{year}</option>
221.         ))}
222.     </select>
223. </div>
224. <div className="color_post_car margin_post_car">
225.     <h3>Колір</h3>
226.     <select
227.         value={post.color}
228.         onChange={(e) => setPost({ ...post, color:
e.target.value })}
229.     >
230.         <option value="" defaultValue hidden>
231.             --- Оберіть колір ---
232.         </option>
233.         {colorCar.map((color) => (
234.             <option>{color}</option>
235.         ))}
236.     </select>
237. </div>
238. <div className="injured_post_car margin_post_car">
239.     <h3>Пошкодження</h3>
240.     <select
241.         value={post.injured}
242.         onChange={(e) => setPost({ ...post, injured:
e.target.value })}
243.     >
244.         <option value="" defaultValue hidden>
245.             --- Авто перебувало в ДТП ---
246.         </option>
247.         <option>Так</option>
248.         <option>Hi</option>
249.     </select>
250. </div>
251. <div className="customs_cleared_post_car margin_post_car">
252.     <h3>Розмитнено</h3>
253.     <select
254.         value={post.customs_cleared}
255.         onChange={(e) =>
256.             setPost({ ...post, customs_cleared: e.target.value })
257.         }
258.     >
259.         <option value="" defaultValue hidden>
260.             --- Розмитнене авто ---
261.         </option>
262.         <option>Так</option>
263.         <option>Hi</option>
264.     </select>

```

```

265.         </div>
266.         <div className="wheel_drive_post_car margin_post_car">
267.             <h3>Привід</h3>
268.             <select
269.                 value={post.wheel_drive}
270.                 onChange={(e) =>
271.                     setPost({ ...post, wheel_drive: e.target.value })
272.                 }
273.             >
274.                 <option value="" defaultValue hidden>
275.                     --- Оберіть тип приводу ---
276.                 </option>
277.                 {wheelDriveCar.map((wheel) => (
278.                     <option>{wheel}</option>
279.                 ))}
280.             </select>
281.         </div>
282.         <div className="transmission_post_car margin_post_car">
283.             <h3>Коробка передач</h3>
284.             <select
285.                 value={post.transmission}
286.                 onChange={(e) =>
287.                     setPost({ ...post, transmission: e.target.value })
288.                 }
289.             >
290.                 <option value="" defaultValue hidden>
291.                     --- Оберіть тип коробки передач ---
292.                 </option>
293.                 {transmissionCar.map((transmission) => (
294.                     <option>{transmission}</option>
295.                 ))}
296.             </select>
297.         </div>
298.         <div className="fuel_post_car margin_post_car">
299.             <h3>Паливо</h3>
300.             <select
301.                 value={post.fuel}
302.                 onChange={(e) => setPost({ ...post, fuel:
e.target.value })}
303.             >
304.                 <option value="" defaultValue hidden>
305.                     --- Оберіть тип палива ---
306.                 </option>
307.                 {fuelCar.map((fuel) => (
308.                     <option>{fuel}</option>
309.                 ))}
310.             </select>
311.         </div>
312.         <div className="comfort_post_car margin_post_car">
313.             <h3>Комфорт</h3>
314.             <select

```

```

315.         value={post.comfort}
316.         onChange={(e) => setPost({ ...post, comfort:
    e.target.value })}}
317.     >
318.         <option value="" defaultValue hidden>
319.             --- Оберіть комплектацію авто ---
320.         </option>
321.         {comfortCar.map((comfort) => (
322.             <option>{comfort}</option>
323.         ))}
324.     </select>
325. </div>
326. <div className="multimedia_post_car margin_post_car">
327.     <h3>Мультимедія</h3>
328.     <select
329.         value={post.multimedia}
330.         onChange={(e) =>
331.             setPost({ ...post, multimedia: e.target.value })
332.         }
333.     >
334.         <option value="" defaultValue hidden>
335.             --- Оберіть мультимедію ---
336.         </option>
337.         {multimediaCar.map((multimedia) => (
338.             <option>{multimedia}</option>
339.         ))}
340.     </select>
341. </div>
342. <div className="security_post_car margin_post_car">
343.     <h3>Сигналізація</h3>
344.     <select
345.         value={post.security}
346.         onChange={(e) => setPost({ ...post, security:
    e.target.value })}}
347.     >
348.         <option value="" defaultValue hidden>
349.             --- Оберіть тип сигналізації ---
350.         </option>
351.         {securityCar.map((security) => (
352.             <option>{security}</option>
353.         ))}
354.     </select>
355. </div>
356. <div className="engine_post_car margin_post_car">
357.     <h3>Об'єм двигуна (л.)</h3>
358.     <input
359.         type="number"
360.         placeholder="2"
361.         value={post.engine}
362.         onChange={(e) => setPost({ ...post, engine:
    e.target.value })}}

```

```

363.         />
364.     </div>
365.     <div className="range_post_car margin_post_car">
366.         <h3>Пробіг (тис.)</h3>
367.         <input
368.             type="number"
369.             placeholder="20000"
370.             value={post.range}
371.             onChange={(e) => setPost({ ...post, range:
    e.target.value })}
372.         />
373.     </div>
374.     <div className="btn_post_car">
375.         <button onClick={() => InsertRows()}>Створити</button>
376.     </div>
377. </div>
378. </div>
379. </div>
380. </>
381. );
382. };
383.
384. export default AddNewPost;
385.

```

3. FuncCalculator.jsx:

```

import React, { useState } from "react";
import { useEffect } from "react";

const infoCar = {
  years: [
    2023, 2022, 2021, 2020, 2019, 2018, 2017, 2016, 2015, 2014, 2013, 2012,
    2011, 2010, 2009, 2008, 2007, 2006, 2005, 2004, 2003, 2002, 2001, 2000,
    1999, 1998, 1997, 1996, 1995, 1994, 1993, 1992, 1991, 1990, 1989, 1988,
    1987, 1986, 1985, 1984, 1983, 1982, 1981, 1980, 1979, 1978, 1977, 1976,
    1975, 1974, 1973, 1972, 1971, 1970, 1969, 1968, 1967, 1966, 1965,
  ],
  typesCar: ["Бензин", "Дизель", "Електро"],
};

const FuncCalculator = () => {
  const [yearCar, setYearCar] = useState(2023);
  const [carType, setCarType] = useState("Бензин");
  const [engineCap, setEngineCap] = useState(0);
  const [price, setPrice] = useState(0);
  const [excise, setExcise] = useState(0);
  const [customs, setCustoms] = useState(0);
  const [pdv, setPDV] = useState(0);
  const [total, setTotal] = useState(0);

```

```

useEffect(() => {
  CalculateExcise();
  CalculateCustoms();
  CalculatePDV();
  totalPrice();
}, [price, carType, total, pdv, engineCap]);

function CalculateExcise() {

  let rate = 0;
  if (carType === "Бензин") {
    if (price < 3000) {
      rate = 50;
    } else {
      rate = 100;
    }
  } else if (carType === "Дизель") {
    if (price > 3500) {
      rate = 75;
    } else {
      rate = 100;
    }
  } else {
    rate = 1;
  }
  setExcise(rate * (engineCap / 1000) * (2024 - yearCar));
}

function CalculateCustoms() {
  let rate = 10;
  setCustoms((price / 100) * rate);
}

function CalculatePDV() {
  let rate = 20;
  setPDV(
    (price / 100) * rate + (excise / 100) * rate + (customs / 100) * rate
  );
}

function totalPrice() {
  setTotal(price + excise + customs + pdv);
}

return (
  <>
    <div className="calc_form_div">
      <div className="calc_calcForm ">
        <p className="calc_formLabel">Калькулятор</p>
        <div className="column">

```

```

<p className="calc_label">Ціна авто (USD)</p>
<input
  className="auto_res"
  onChange={(e) => setPrice(parseInt(e.target.value, 10))}
/>
</div>
<div className="column">
  <p className="calc_label">Рік випуску</p>
  <select
    className="auto_res"
    onChange={(e) => setYearCar(parseInt(e.target.value, 10))}
  >
    {infoCar.years.map((year) => (
      <option value={year}>{year}</option>
    ))}
  </select>
</div>
<div className="column">
  <p className="calc_label">Тип двигуна</p>
  <select
    className="auto_res"
    onChange={(e) => setCarType(e.target.value)}
  >
    {infoCar.typesCar.map((type_car) => (
      <option value={type_car}>{type_car}</option>
    ))}
  </select>
</div>
<div className="column">
  <p className="calc_label">
    Об'єм двигуна (см3)/Ємність батареї (кВт)
  </p>
  <input
    className="auto_res"
    onChange={(e) => setEngineCap(parseInt(e.target.value, 10))}
  />
</div>
</div>
<div className="calc_calcForm column">
  <p className="calc_formLabel">Ціна</p>
  <p className="calc_priceInfo">
    <span>Ціна авто</span>
    <b>{price}$</b>
  </p>
  <p className="calc_priceInfo">
    Ввізне мито<b>{customs}$</b>
  </p>
  <p className="calc_priceInfo">
    Акциз<b>{excise}$</b>
  </p>
  <p className="calc_priceInfo">

```

```

        ПДВ<b>{pdv}$</b>
    </p>
    <p className="calc_priceInfo">
        Загальна вартість<b>{total}$</b>
    </p>
    <a
        href="tel:+380 (66) 177 3992"
        className="button m-t-20 calc_button"
    >
        Зателефонувати
    </a>
    <p className="calc_details">Даний розрахунок є орієнтовним</p>
</div>
</div>
</>
);
};

export default FuncCalculator;

```