МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Web-орієнтована система обліку сервісного обслуговування транспортного засобу»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ-61 Ляшенко Максим Віталійович

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____ «__» _____ 2020 р.

Науковий керівник _____ к.т.н., доц., Марченко А. В.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д. М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2020

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«__» _____ 2020 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Ляшенко Максим Віталійович

1 Тема роботи *Web-орієнтована система обліку сервісного обслуговування транспортного засобу*

керівник роботи *Марченко Анна Вікторівна, к.т.н., доцент*,

затверджені наказом по університету від « 14 » травня 2020 р. № 0576-III

2 Строк подання студентом роботи «1» червня 2020 р.

3 Вхідні дані до роботи технічне завдання на розробку web-орієнтованої системи.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз предметної області, моделювання та проектування web-орієнтованої системи, розробка web-орієнтованої системи обліку сервісного обслуговування транспортного засобу.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність проекту, поставка задачі, аналіз сайтів-аналогів, порівняння сайтів-аналогів, функціональні вимоги до додатку, методи розробки, моделювання роботи додатку – IDEF0, моделювання роботи додатку – IDEF1, діаграми варіантів використання, архітектура додатку, ER діаграма бази даних, розроблений дизайн додатку, конференція IMA, висновки.

6. Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7.Дата видачі завдання_____01.10.2019_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Оформлення планування робіт	До 10.03.2020	
2.	Оформлення технічного завдання	До 20.03.2020	
3.	Проведення аналізу предметної області	До 24.03.2020	
4.	Моделювання та проектування	До 15.04.2020	
5.	Розробка додатку	До 15.05.2020	
6.	Тестування додатку	До 17.05.2020	
7.	Здача пояснювальної записки та файлів розробленого проекту	До 01.06.2020	

Студент _____ Ляшенко М.В.
(підпис)

Керівник роботи _____ к.т.н., доц. Марченко А.В.
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «Web-орієнтована система обліку сервісного обслуговування транспортного засобу».

Пояснювальна записка складається зі вступу, 3 розділів, висновків, списку використаних джерел із 12 найменувань, додатків. Загальний обсяг роботи − 86 сторінок, у тому числі 56 сторінок основного тексту, 1 сторінка списку використаних джерел, 29 сторінок додатків.

Кваліфікаційну роботу бакалавра присвячено розробці Web-орієнтованої системи обліку сервісного обслуговування транспортного засобу.

В роботі проведено аналіз актуальності розробки, аналіз сайтів-аналогів, моделювання та проектування Web-орієнтованої системи.

У роботі виконано розробку Web-орієнтованої системи, яка складається з таких модулей:

- Модуль авторизації;
- Модуль реєстрації;
- Модуль додавання автомобіля;
- Модуль редагування автомобіля;
- Модуль додавання сервісної події;
- Модуль редагування сервісної події;
- Модуль редагування профілю користувача;
- Модуль статистичних даних;
- Модуль фільтрованого пошуку.

Результатом проведеної роботи є розроблена Web-орієнтована система обліку сервісного обслуговування транспортного засобу

Ключові слова: автомобіль, сервісне обслуговування, статистика, web-додаток, пошук.

ЗМІСТ

ВСТУП

Рухатись вперед в технологічному розвитку в наш час є одним із найактуальніших пріоритетів сучасного суспільства. Бажання автоматизувати, полегшити будь-яку задачу завжди цікавила людей.

З кожним роком все важче зустріти людину, яка б не користувалася сучасними гаджетами, адже використання смартфону чи комп'ютеру значно полегшує життя в плані спілкування, пошуку інформації та її збереження, прослуховування музики чи перегляді відео матеріалів.

Кожен свідомий автомобіліст слідкує за своїм автомобілем, адже з ним від проводить велику кількість часу. Автомобіль повинен працювати в любу годину та пору року, але для цього за ним потрібно слідкувати, виконувати обслуговування. Тому, актуальною є розробка web-орієнтованої системи для сервісного обслуговування авто, адже додати інформацію можна як з смартфону так і комп'ютера чи планшета з доступом до мережі Інтернет.

Досить актуальна тема з оптимізації часу, його правильного планування для різноманітних задач. Однією з важливих задач для автомобіліста стає обслуговування їхнього автомобіля, потрібно тримати в пам'яті дату останнього технічного огляду чи сервісної події, які операції виконувались і навіть в яку ціну вони обійшлись. Людині властиво забувати й це стосується догляду за авто, використовуючи web-орієнтовану інформаційну систему, сервісна історія буде надійно збережена в хмарному сервісі й немає сенсу запам'ятовувати інформацію. Використання web-додатку значно спрощує життя автомобілістів й ризик забути про технічний огляд – мінімізується.

1. Аналіз предметної області

1.1. Огляд останніх досліджень і публікацій

З кожним роком автовласників стає все більше, й кожен може мати декілька автомобілів. Як зазвичай, вся техніка потребує обслуговування, автомобілі не виключення. Зазвичай, це проведення технічного огляду(ТО), заміна агрегатів автомобіля, які вийшли з ладу. Ці всі зміни потрібно фіксувати, адже ТО проводять через певний інтервал часу або відповідно проїханого шляху автомобілем, таку інформацію досить важко тримати в пам’яті, особливо коли володієш двома і більше автомобілями.

Кожен автомобіль за замовчуванням має спеціальну сервісну книгу, яка повинна завжди лежати в бардачку. Вона повинна містити всю інформацію про обслуговування автомобіля. Але це не зовсім надійно та зручно, особливо якщо декілька автомобілів, й книжку досить легко зіпсувати чи зовсім загубити.

В сучасному світі, де книги переходять в електронний вигляд, доцільно було б вести електронну сервісну книгу, до якої з легкістю можна отримати доступ в любому місці та в будь-який час, інформація буде надійно зберігатися й навіть після продажу автомобіля.

Продукт проекту – WEB-орієнтована система обліку сервісного обслуговування транспортного засобу.

У соціально-економічному аспекті розроблене рішення підвищить продуктивність людей, які займаються підбором автомобілів за допомогою фільтрованого пошуку.

У технічному аспекті додаток дозволяє отримувати доступ до інформації в любий час та з любої точки світу на будь-якій платформі.

У комерційному, економічному та фінансовому аспекті розроблене рішення підвищить впевненість автомобілістів в правильно обраному авто, яке вони зможуть повноцінно обслуговувати та знатимуть всі його недоліки.

Формалізація цінності – властивості WEB-орієнтованої системи будуть нести цінність для працівників логістичних центрів для автомобільних парків, звичайних автомобілістів та майбутніх власників авто.

Життєсталість – залежить від кількості поломок авто та їх характер чи звичайного технічного обслуговування. Також важливу роль відіграють технології та алгоритми розрахунку застосовані під час розробки WEB-орієнтованої системи, доки вони будуть актуальні життєсталість продукту буде високою.

1.2. Аналіз програмних продуктів – аналогів

Провівши дослідження, було знайдено декілька аналогів для майбутнього додатку, це web-сайт drive2.ru (рис. 1.1) та web-сайт drivernotes.net (рис. 1.2).

Drive2.ru – це інтернаціональна спілка автомобілістів, де висвітлюються новини автомобільного світу та інформація від самих користувачів. Функціонал досить простий, ведення бортового журналу в вільному стилі. Збір статистичних даних виконаних робіт не проводиться, тобто опис події не структурований певним чином, користувач описує свої дії в вільному стилі. Сайт має сучасний дизайн та досить швидко працює. Передбачає функціонал для спілкування між автомобілістами, свого роду соціальна мережа, де можна підписатися на користувача і поставити позначку «like». Має влаштовану рекламу та інтеграцію з іншими web-ресурсами.

DriverNotes.net – це також інтернаціональний сервіс, призначений для регулювання витрат утримання автомобіля, кількості використаного палива. Має структурований запис події, дані якої будуть відображені на статистичному графіку, дозволяє переглянути історію сервісного обслуговування моделі та марки автомобіля.

Сайт працює дуже повільно і має застарілий дизайн, деякі блоки сайту, іноді, зовсім не працюють.

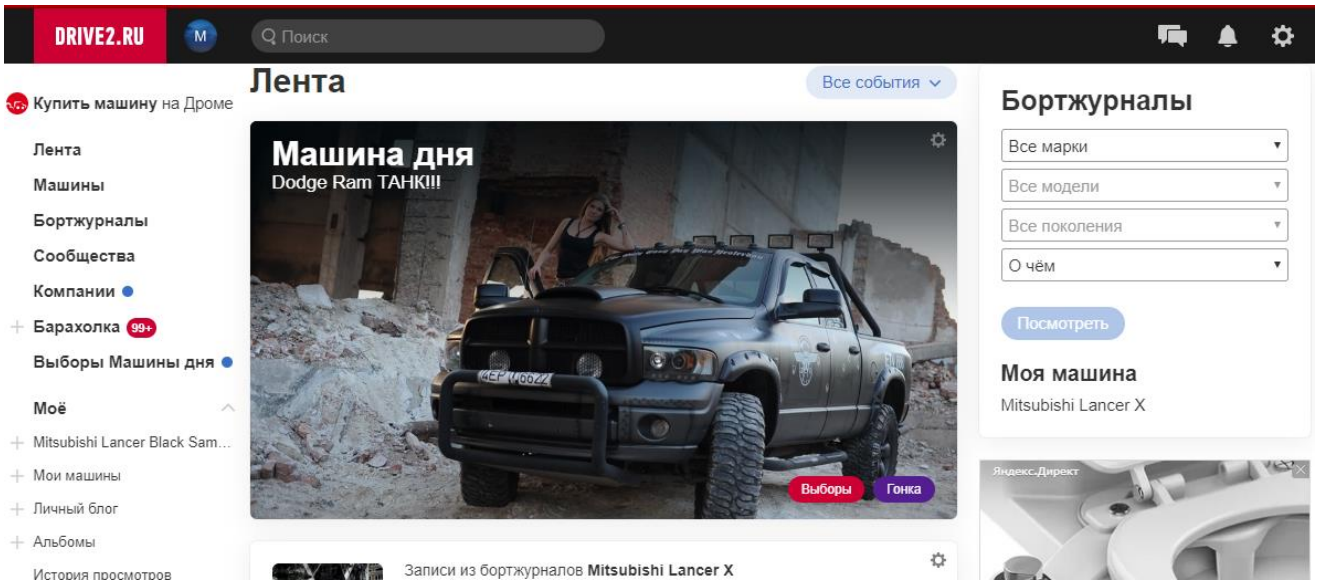

Рисунок 1.1 – Головна сторіка сайту drive2.ru

Рисунок 1.2 – Головна сторінка сайту drivernotes.net

Для фінального аналізу аналогів, було створено порівняльну характеристику, яка відобразить всі плюси та мінуси кожного аналогу й створюваного додатку також. Порівняльна характеристика відображена в таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика аналогів

Характеристики	**Drive2.ru**	**Drivernotes.net**	**Створюваний додаток**
Сучасний інтерфейс	+	-	+
Статистичні дані	-	+	+
Швидка робота сайту	+	-	+
Наявність реклами	+	+	-
Інтеграція з сторонніми сервісами	+	-	-
Швидкий зворотній зв’язок	+	-	+

1.3. Постановка задачі

Метою виконання даної роботи є розробка інформаційної системи для обліку сервісного обслуговування транспортного засобу. WEB-орієнтована система повинна виконувати ряд задач:

– збереження, фільтрація та аналіз сервісної історії автомобіля;

– відображення фільтрованої інформації для користувача без автомобіля;

– постійний та швидкий доступ до додатку, без необхідності встановлення додаткового ПЗ;

– постійний та швидкий доступ до додатку, незалежно від типу пристрою та операційної системи користувача.

Для реалізації поставленого завдання були виділені наступні задачі дослідження:

- аналіз аналогів;
- розробка додатку;
- розробка інтерфейсу;
- розробка модулю фільтрованого пошуку;
- розробка модулю адмін-панелі;
- розробка модулю авторизації та реєстрації;
- розробка користувальницького інтерфейсу;
- відладка та тестування;
- оформлення супроводжуючої документації.

2. Моделювання та проектування

2.1 Проектування WEB-орієнтованої системи

Проектування інформаційної системи проводиться за допомогою технік моделювання IDEF(Integrated DEFinition) – це комбінація графічних і мовних символів та правил, які були розроблені для фіксації процесів і структур в організаціях. В даному розділі представлені декілька типів діаграм IDEF0 – функціональне моделювання бізнес-процесів та IDEF1 – методологія моделювання інформаційних потоків в системі.

На рисунку 2.1 відображено контекстну діаграму 0-го рівня.

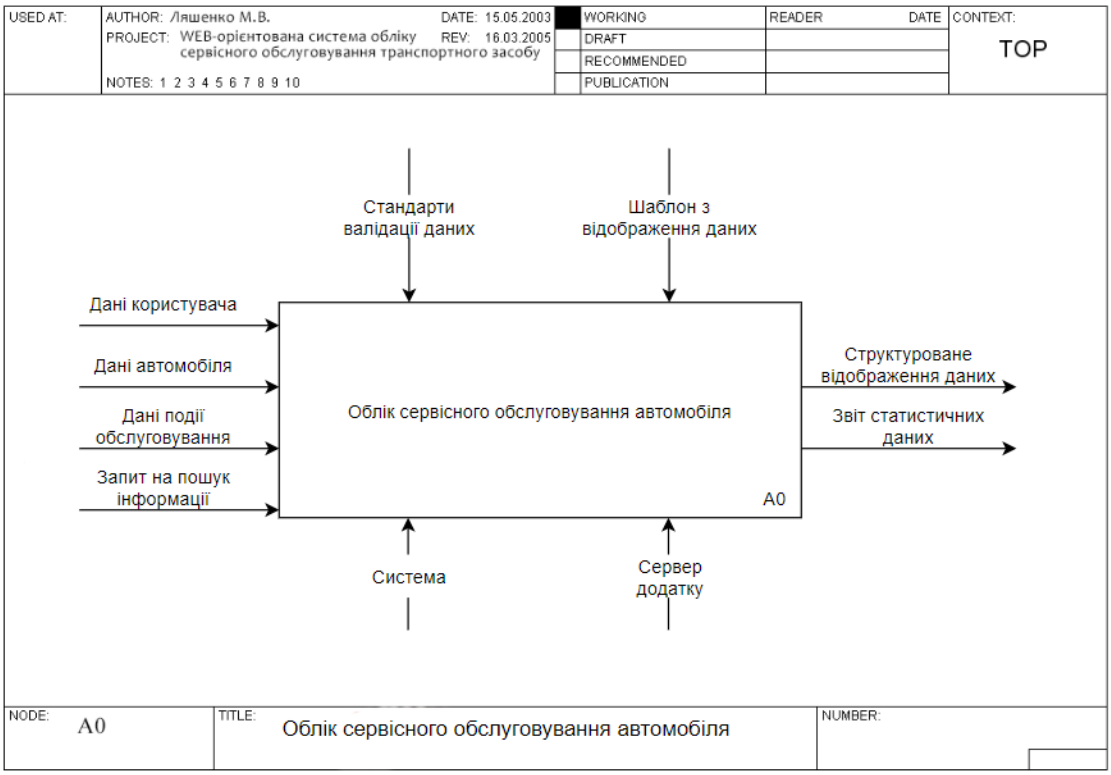

Рисунок 2.1 - Контекстна діаграма 0-го рівня, IDEF0

На рисунку 2.2 відображено діаграма декомпозиції 1-го рівня для процесу «Облік сервісного обслуговування автомобіля».

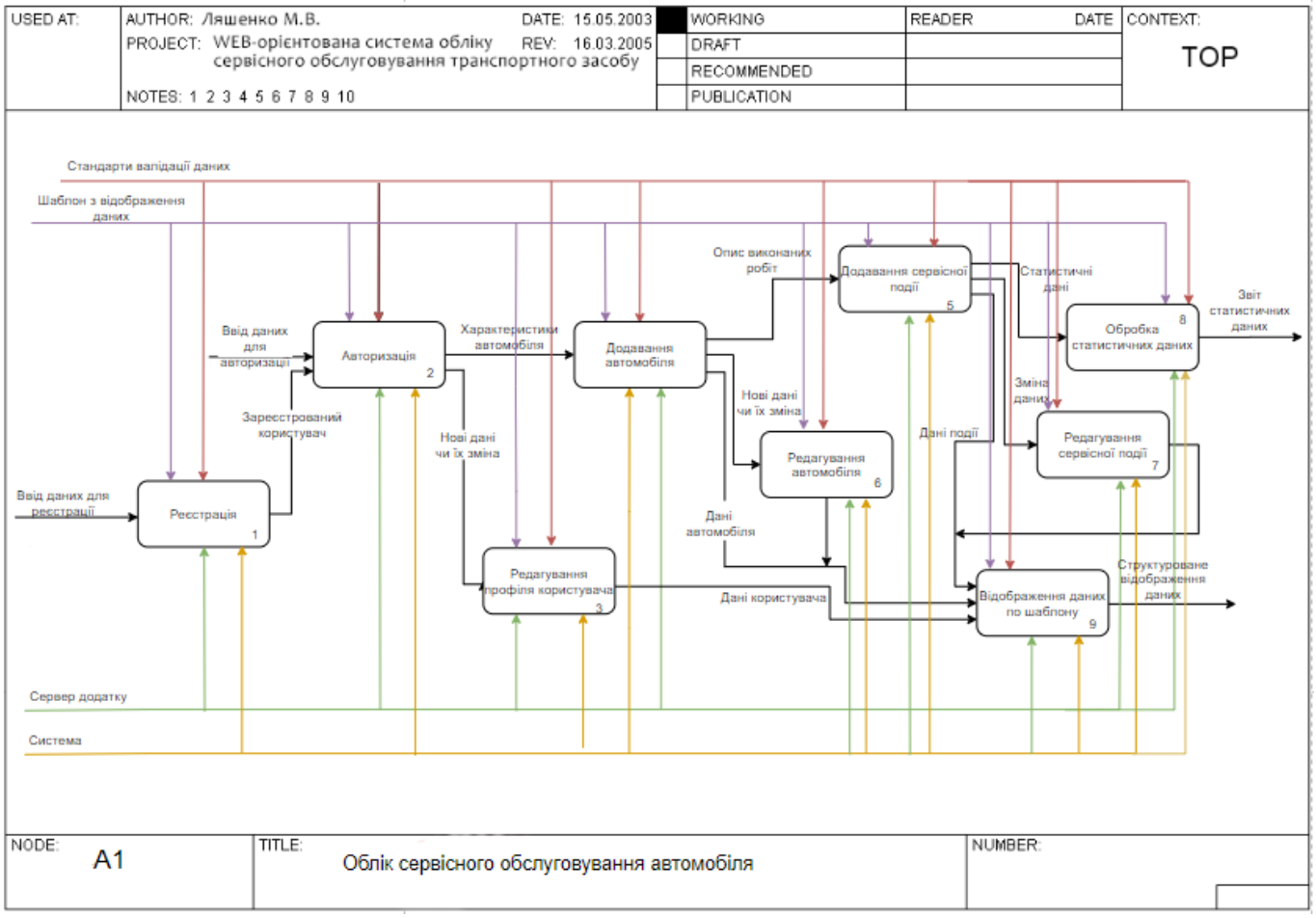

Рисунок 2.2 – Діаграма декомпозиція 1-го рівня, IDEF1

Для наглядного відображення варіантів використання WEB-орієнтованої системи, було створено діаграма варіантів використання – зображує відношення між акторами та прецендентами в системі. На рисунку 2.3 відображено діаграму варіантів використання.

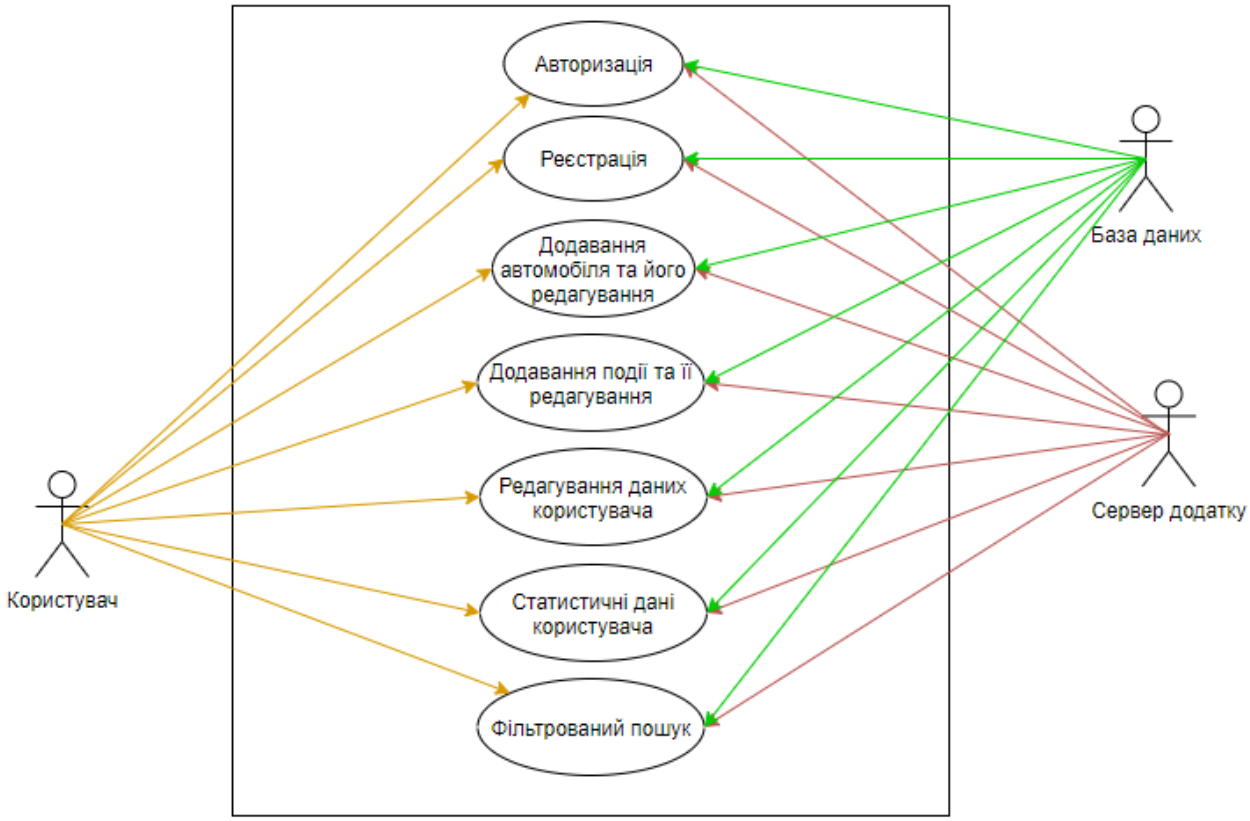

Рисунок 2.3 – Діаграма варіантів використання

2.2 Проектування моделі бази даних

В процесі моделювання бази даних, необхідно сформувати інформаційні сутності та необхідно визначити атрибути і їх параметри. Сформовані сутності відобразити за допомогою ER-діаграми – мета-моделі даних, де відобразити зв’язки між сутностями. Сутності представляють собою абстракції реального об’єкта чи деякого уявлення про об’єкт.

Для повноцінного функціонування web-додатку потрібно виділити такі сутності:

– Користувачі;

- Автомобілі;
- Записи сервісної події;
- Автомобільні бренди;
- Автомобільні марки.

Відповідно для кожної з представлених сутностей є відповідні атрибути, які наведені у таблиці 2.1.

Таблиця 2.1 – Таблиця з опису атрибутів сутностей

Назва сутності	Опис сутності	Назва поля	Опис атрибуту
users	Персональна інформація користувача	user_id	Унікальний ідентифікатор користувача
		first_name	Ім’я користувача
		last_name	Фамілія користувача
		gender	Стать користувача
		user_image	Фотокартка користувача
		description	Опис від користувача
		driving_experience	Водійський стаж
		e_mail	Електронна адреса
		password	Пароль до облікового запису
		level	Рівень доступу
		date	Дата реєстрації
		location	Місце проживання
cars	Інформація про автомобіль та його технічні характеристики	car_id	Унікальний ідентифікатор автомобіля
		user_id	Унікальний ідентифікатор користувача/власника авто
		brand_id	Унікальний ідентифікатор бренду автомобіля
		model_id	Унікальний ідентифікатор марки автомобіля
		year_of_issue	Рік випуску авто
		year_of_purchase	Рік покупки авто
		car_image	Зображення автомобіля
		description	Опис автомобіля
		engine_type	Тип двигуна
		engine_volume	Об’єм двигуна
		type_of_transmission	Тип трансмісії
		engine_power	Потужність двигуна
		type_of_drive	Тип приводу

Таблиця 2.1 – Продовження таблиці з опису атрибутів сутностей

Назва сутності	Опис сутності	Назва поля	Опис атрибуту
		car_name	Ім’я автомобіля
		add_date	Дата реєстрації авто
notes	Інформація про сервісну подію	note_id	Унікальний ідентифікатор запису
		car_id	Унікальний ідентифікатор автомобіля
		title	Заголовок запису
		description	Опис запису
		price	Ціна виконаних робіт
		date	Дата проведення
		operation_type	Тип операції
		km_count	Значення на одометрі авто
		location_name	Назва місця проведення робіт
		lat_lng	Координати локації
Car_brand	Список доступних брендів автомобілів	brand_id	Унікальний ідентифікатор бренду автомобіля
		brand	Назва бренду
Car_model	Список з моделей автомобіля та їх характеристик	model_id	Унікальний ідентифікатор марки автомобіля
		brand_id	Унікальний ідентифікатор бренду автомобіля
		model	Назва марки
		start_production	Початок виробництва
		end_production	Кінець виробництва

Відповідно до сформованих сутностей та їх атрибутів, була створена діаграма по типу «сутність-зв’язок», яка відображена на рисунку 2.4.

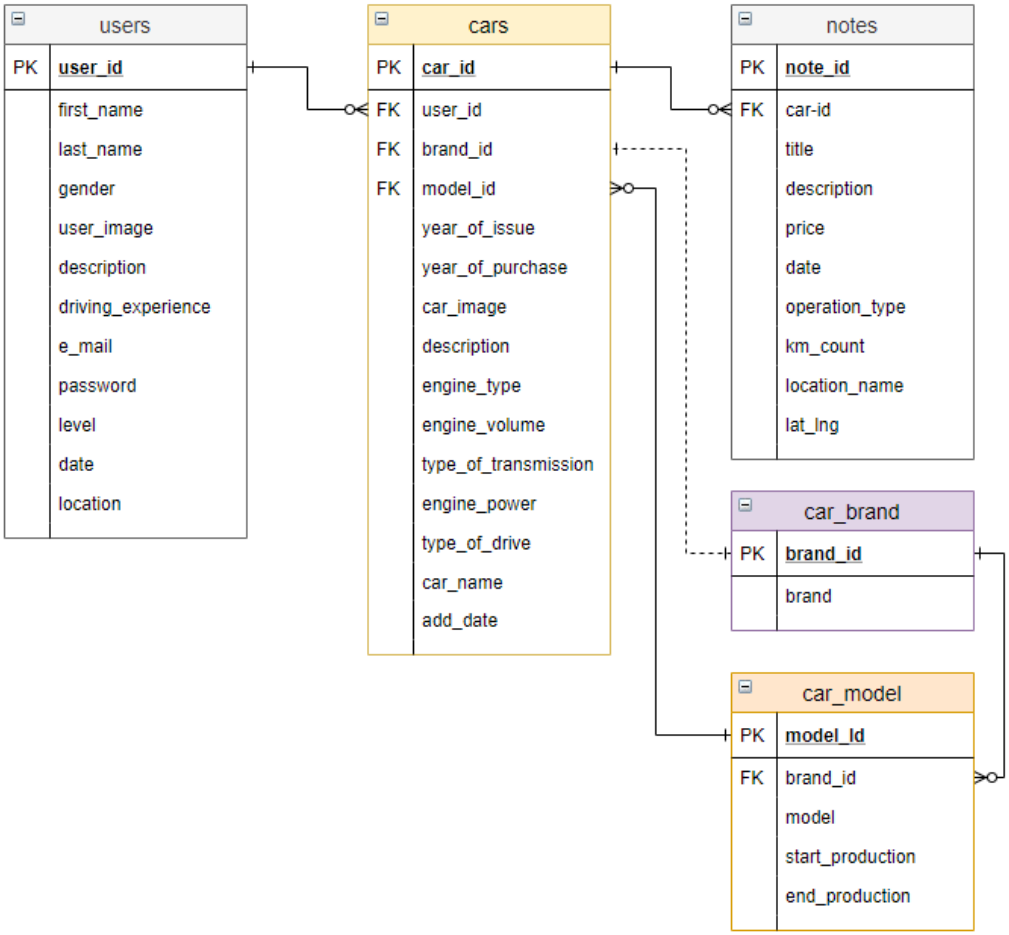

Рисунок 2.4 – ER-діаграма

На рисунку 2.4 відображені зв’язки між сутностями та ці зв’язки відображують залежності. Для сутності «users» та «cars» позначений зв’язок один-до-багатьох, тобто у користувача може бути не обмежена кількість автомобілів, в свою чергу, у автомобіля може бути тільки один власник. Схожий зв’язок (один-до-багатьох) мають такі сутності: «cars» та «notes» (автомобіль має багато записів, але один запис відноситься тільки до одного автомобіля), «car_brand» та «car_model» (у одного бренду може бути декілька моделей але у однієї моделі тільки один бренд), «car_model» та «cars» (у одного автомобіля може бути декілька моделей одного бренду), «cars» та «car_brand» (у одного автомобіля може бути тільки один бренд, в свою чергу бренд може відповідати багатьом автомобілям).

3. Розробка Web-додатку

3.1 Архітектура Web-додатку

Web-додаток будується на архітектурі «клієнт-сервер», адже являє собою особливий тип програми. Адже Web-додаток знаходиться та виконується на стороні сервера, де клієнт отримує тільки результат роботи. Вся працездатність додатку базується на основі запитів, якими обмінюються клієнт та сервер, обробка яких відбувається через мережу Інтернет. Схему архітектури створеного web-додатку можна переглянути на рисунку 3.1.

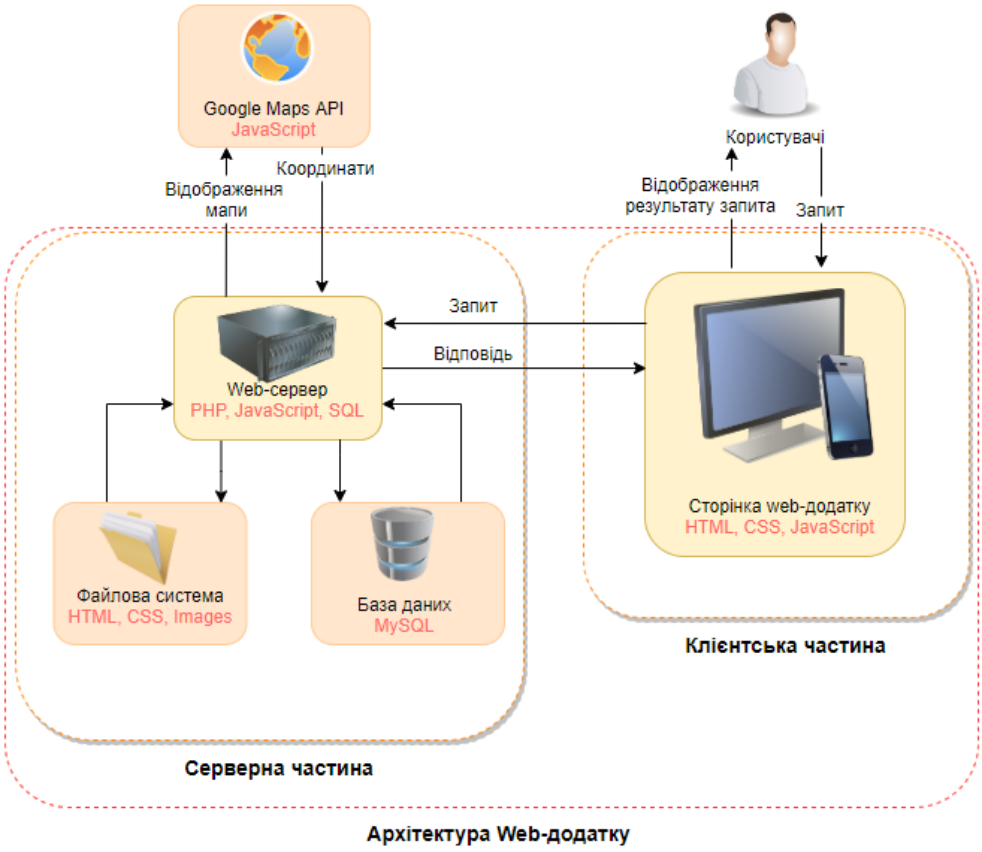

Рисунок 3.1 – Схема архітектури Web-додатку

Додаток орієнтований на роботу з користувачами, тому кожен його запит повинен бути опрацьований і відображено результат запиту, навіть якщо запит не повернув жодної інформації або закінчився з помилкою. Переглянувши схему, стає зрозуміло, що додаток складається з клієнтської частини та серверної.

Клієнтська частина – відповідає за коректне відображення сторінок в браузерах та інших девайсах, їх адаптивність. Використовує для цього HTML, CSS, JS.

Серверна частина – має великий функціонал та багато компонентів, які взаємодіють між собою. Основою є web-сервер, який займається обробкою запитів та відповідає на них, для він використовує PHP, JS, SQL. Файлова система відповідає за роботу з шаблонами документів для відображення інформації, файлами каскадних стилів для придання сторінці структурованого і стилізованого вигляду, фотографіями користувачів або фотографіями, які потребує система, також для збереження фото. База даних містить в собі всю інформацію, яку вводить користувач, тобто персональні дані, дані про автомобілі та їх сервісні події. Під час запиту від web-серверу відповідає масивом значень й після чого сервер аналізує отриману інформацію.

Для роботи з мапою використаний зовнішній компонент – Google Maps API, який постійно взаємодіє з додатком, спочатку виконує відображення мапи в певній точці та заданому масштабі й в результаті роботи користувача відправляє до сервера обрані користувачем координати мітки та назву місця проведення, де сервер аналізує цю інформацію та відправляє до бази даних.

3.2 Програмна реалізація

Для розробки web-додатку існує багато технологій, які дозволяють створити сучасний, адаптивний, кросплатформний та кросбраузерний додаток. За основу були обрані:

– HTML (HyperText Markup Language) – мова гіпертекстової розмітки, за допомогою якої створюється каркас сторінок, розміщуються блоки та створюється повноцінна структура сторінки;

– CSS (Cascading Style Sheets) – це формальна мова, призначена для опису стилю сторінок, їх зовнішнього вигляду. За її допомоги налаштовується адаптивність сайту та його кольорова гамма;

– JavaScript – об’єктно-орієнтована, прототипна, динамічна мова програмування, має великий функціонал, слугує для придання сторінці інтерактивності;

– PHP (Hypertext Preprocessor) – скриптова мова програмування загального призначення, використовується для розробки web-додатків.

– SQL (Structured Query Language) – мова програмування, призначена для керування даними в реляційних базах даних, тобто: відображення, додавання та редагування.

– MySQL – безкоштовна реляційна система для керування базами даних.

Також, були застосовані сторонні сервіси:

– Bootstrap – фреймворк від творців Twitter, знаходиться в вільному доступі, представляє собою дистрибутив з набору класів та стилів, за допомогою яких можна в лічені години створити структуру сайту.

– Font Awesome – сервіс з графічних іконок, частина іконок знаходиться в вільному доступі, були використані тільки безкоштовні іконки й слугують для візуального ефекту;

– Google Fonts – безкоштовний сервіс шрифтів від компанії Google;

– Google Maps – сервіс для роботи з мапою, цей сервіс є платним з можливістю оформлення пробного періоду в 12 місяців.

Відповідно до технічного завдання, були виділені основні функції додатку, його структура, наявність модулів та їх структуру і залежності. Сформована структура додатку зображена на рисунку 3.2.

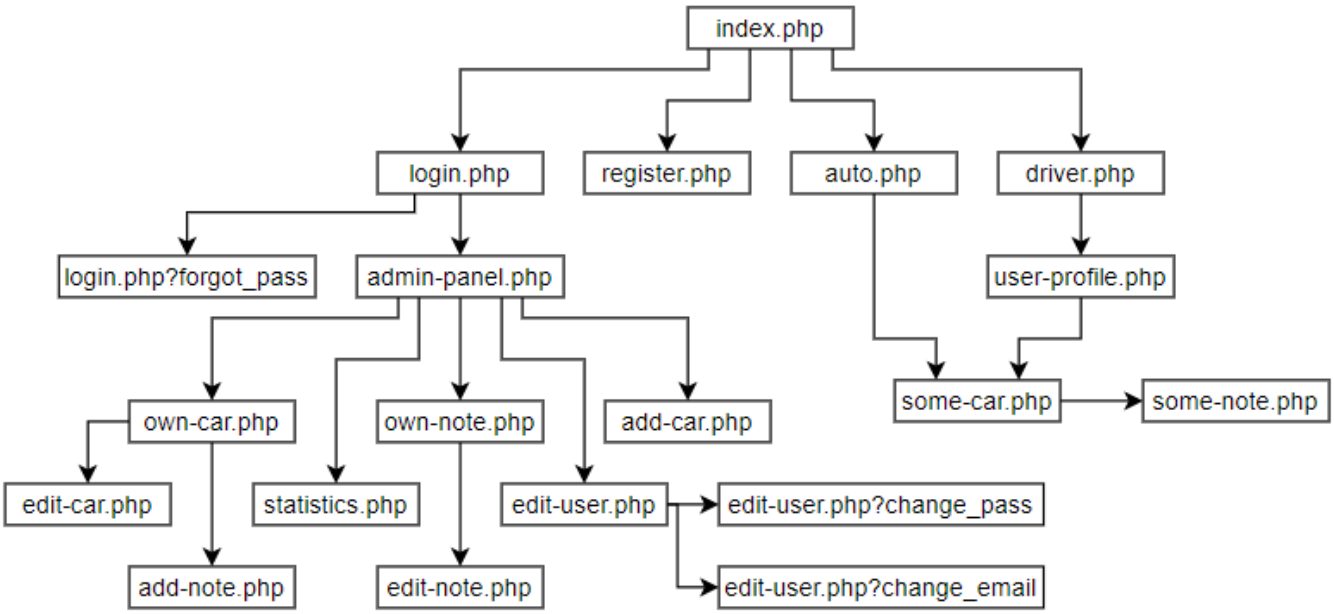

Рисунок 3.2 – Структура web-додатку

Основні модулі додатку: реєстрація, авторизація, додавання та відображення інформації, її редагування. Модулі з додавання та редагування інформації містять перевірку даних тобто валідацію. Також присутній модуль з відображення навігації, адже для зареєстрованого користувача – вона змінюється.

Після повністю спланованої бази даних, підготовлюється sql запит на створення бази даних в реляційній системі баз даних MySQL. Фізичну реалізацію бази даних можна переглянути на рисунку 3.3.

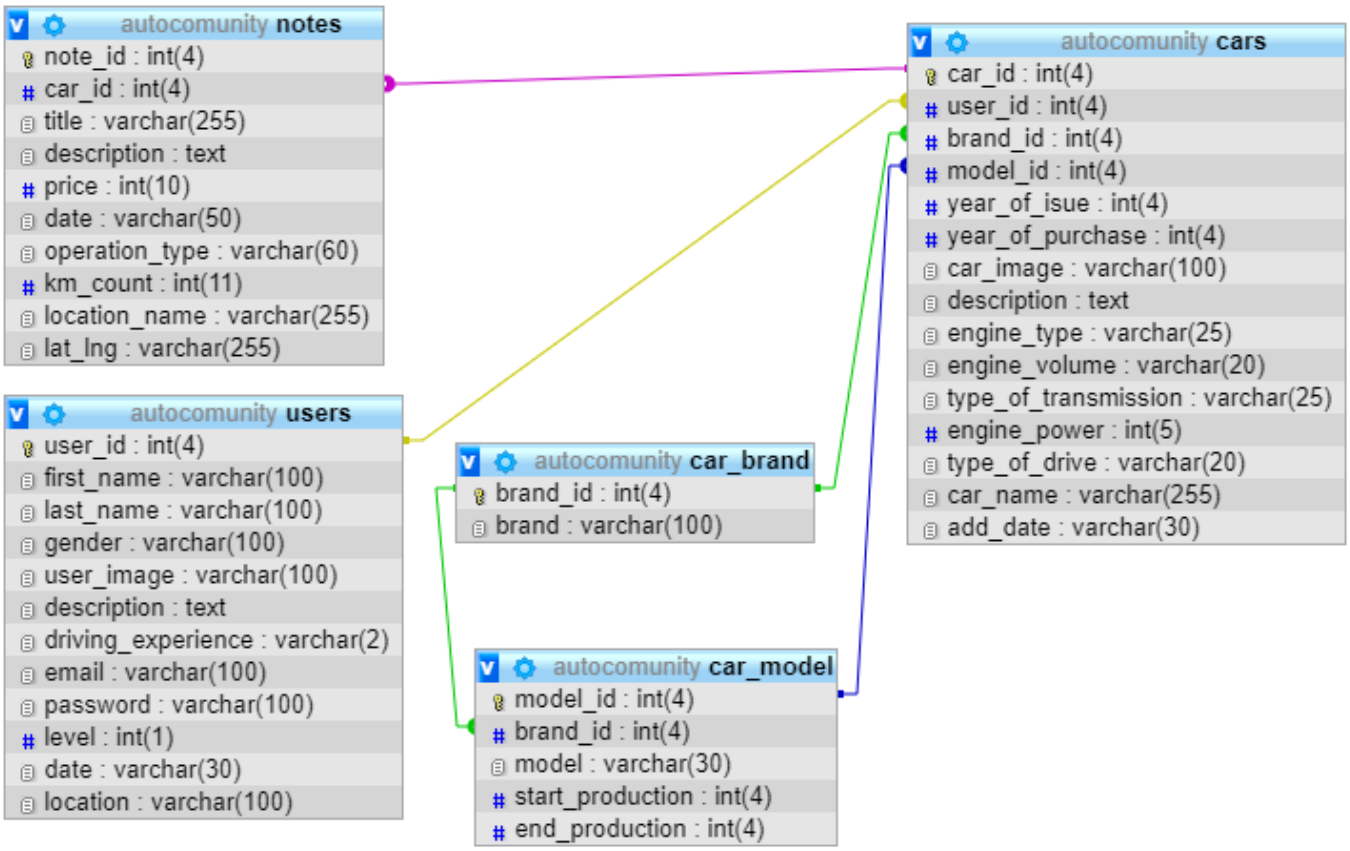

Рисунок 3.3 –Фізична реалізація бази даних

3.2.1 Модуль реєстрації

Модуль реєстрації представлений з п’яти полів: ім’я, фамілія, email, пароль та повтор введеного паролю. Інформацію ж кожного поля зберігається з змінну та оброблюється в подальшому, фрагмент з модулем реєстрації зображений на рисунку 3.4.

Основні змінні:

- $r_name – зберігає введене ім’я користувача з методу POST;
- $r_surname – зберігає введену фамілію користувача з методу POST;
- $r_mail – зберігає введений email користувача з методу POST;
- $r_password – зберігає введений пароль користувача з методу POST;
- $r_password2 – зберігає повторно введений пароль користувача з методу POST.

Наступний крок – перевірка даних на дублікат в базі даних, за це відповідає змінна $test_email, використовуючи дані зі змінної $r_mail перевіряється наявність такої ж електронної пошти, якщо вже введена електронна адреса існує в мережі то відображується повідомлення «Усп! Такий e-mail додано в систему!».

Виконується перевірка паролів, якщо паролі не співпадають – користувачу відображується повідомлення «Упс! Паролі не співпадають.». Пройшовши перевірку паролів, змінна $query формує sql запит для додавання інформації до бази даних й змінна $result відправляє його до бази даних. Доцільно виконати перевірку на успішне виконання змінної, користувач отримує повідомлення про успішно виконану реєстрацію й посилання до сторінки авторизації.

```php
<?php
if(isset($_POST["r_name"])){ $r_name = $_POST["r_name"]; }
if(isset($_POST["r_surname"])){ $r_surname = $_POST["r_surname"]; }
if(isset($_POST["r_mail"])){ $r_mail = $_POST["r_mail"]; }
if(isset($_POST["r_password"])){ $r_password = md5($_POST["r_password"]); }
if(isset($_POST["r_password2"])){ $r_password2 = md5($_POST["r_password2"]); }
if(isset($_POST["r_send"])){
    $test_email = mysql_query("SELECT `email` FROM `users` WHERE `email` = '$r_mail';");
    if(mysql_num_rows($test_email) != 0){
        echo '
            <div class="alert alert-danger alert-dismissible fade show" role="alert">
              <strong>Усп!</strong> Такий e-mail додано в систему!
              <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                <span aria-hidden="true">&times;</span>
              </button>
            </div>';
    }else if(mysql_num_rows($test_email) == 0){
        if($r_password !=$r_password2){
            echo '<div class="alert alert-danger alert-dismissible fade show" role="alert">
              <strong>Усп!</strong> Паролі не співпадають.
              <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                <span aria-hidden="true">&times;</span>
              </button>
            </div>';
        }else{
    $date = date("m.d.y");
    $query = " INSERT INTO `users` (`first_name`, `last_name`, `email`,`password`,`level`, `date`,
    `location`, `user_image`) VALUES ('$r_name', '$r_surname', '$r_mail','$r_password','2', '$date',
    'Відсутня', 'user.png');";
    $result = mysql_query($query) or die ( "Error : ".mysql_error() );
    if($result){
        echo '<div class="alert alert-info" role="alert">
              Реєстрація виконана успішно. Пройти <a href="login.php">авторизацію</a>
            </div>';
        exit();
    }}
    }
}
?>
```

Рисунок 3.4 – Модуль реєстрації

3.2.2 Модуль авторизації

Сторінка з авторизацією представлена у вигляді форми, яка містить декілька полів для вводу електронної адреси на паролю, посилання на сторінку з відновлення паролю та сторінку реєстрації. Модуль має декілька головних змінних:

– $login - зберігає введену електронну адресу користувача з методу POST й перед зберіганням проходить перевірку допустимості символів за допомогою методу mysql_real_escape_string – метод екранує недопустимі символи і тим самим перешкоджає виконанню скриптів та вірусного коду;

– $password - зберігає введений пароль користувача з методу POST, перед збереженням виконується шифрування паролю за допомогою методу md5 – представляє собою строку з 16 символів.

Процес авторизації банально простий, дані зі змінних, описаних вище використовуються для перевірки їх на наявність в базі даних за допомогою змінних $sql_log – формує запит до бази даних та $row_log – відправляє запит до бази даних. Якщо дані не були знайдені, користувач буде проінформований повідомленням «Упс! Користувач з таким Email чи паролем не знайдено.», для підвищення безпеки, яке саме поле було не правильним – не уточнюється. В іншому випадку авторизація проходить успішно, в сесію записується унікальний ідентифікатор користувача та його ім’я і користувача перенаправляє на сторінку адміністративної панелі. Лістинг коду форми авторизації представлений на рисунку 3.5.

```php
if (isset($_POST['send']))
 {
    $login = mysql_real_escape_string($_POST['email']);
    $password = md5($_POST['password']);
    $sql_log = mysql_query("SELECT `user_id`, `level`, `first_name` FROM `users`
    WHERE `email`='$login' AND `password`='$password'
    LIMIT 1",$link) or die(mysql_error());
    $row_log = mysql_fetch_array($sql_log);
    $id_profile = $row_log['user_id'];
    $_SESSION['user_level'] = $row_log['level'];
    if (mysql_num_rows($sql_log) == 1) {
        if($row_log['level']==1) {
           $_SESSION['user_name'] = 'Администратор';
           $_SESSION['user_id'] = $row_log['user_id'];
           echo  '<meta http-equiv="refresh" content="0; url=url=admin-panel.php?
           user_id='.$row_log['user_id'].'">';
        }else if($row_log['level']==2){
           $_SESSION['user_name'] = $row_log['first_name'];
           $_SESSION['user_id'] = $row_log['user_id'];
           echo  '<meta http-equiv="refresh" content="0; url=admin-panel.php?
           user_id='.$row_log['user_id'].'">';
        }
    }
    else {
    echo '<div class="alert alert-danger alert-dismissible fade show" role="alert">
          <strong>Усп!</strong> Користувача з таким Email чи паролем не знайдено.
          <button type="button" class="close" data-dismiss="alert" aria-label="Close">
            <span aria-hidden="true">&times;</span>
          </button>
        </div>';
    }
}
```

Рисунок 3.5 – Модуль авторизації

3.2.3 Модуль для відновлення паролю

Відновлення паролю – досить таки важливий функціонал і він повинен бути в обов’язковому порядку. Адже при втраті паролю користувачем, він повинен мати можливість відновити його. Даний модуль представлений одним полем для вводу електронної адреси, після успішної перевірки пошти на її наявність в базі даних, відправляється новий згенерований пароль і відображується повідомлення про успішно виконану операцію та посилання на сторінку авторизації. В іншому випадку, користувач отримає повідомлення про відсутність даної електронної адреси в системі. Лістинг коду модуля з відновлення паролю представлений на рисунку 3.6.

```php
if(isset($_POST['send_email'])){
      $login = mysql_real_escape_string($_POST['email']);
      $sql_log = mysql_query("SELECT `first_name`, `last_name` FROM `users`
      WHERE `email`='$login' LIMIT 1",$link) or die(mysql_error());
      $row_log = mysql_fetch_array($sql_log);
      $first_name = $row_log['first_name'];
      $last_name = $row_log['last_name'];
      if (mysql_num_rows($sql_log) == 1) {
          $pass_array = array ("92", "83", "7", "66", "45", "4", "36", "22", "1", "0",
             "k", "l", "m", "n", "o", "p", "q", "1r", "3s", "a", "b", "c", "d", "5e", "f", "g",
             "h", "i", "j6", "t", "u", "v9", "w", "x5", "6y", "z5");
          for ($i = 0; $i < 8; $i++)
          {
            shuffle ($pass_array);
            $string = $string.$pass_array[1];

          }
          $new_pass = md5($string);
          $refresh_pass = mysql_query("UPDATE `users` SET  `password`='{$new_pass}' WHERE
          `email`='{$login}'",$link) or die(mysql_error());
          mail($login, "Запрос на восстановление пароля", "Привет, $last_name $first_name. Твой
          новый пароль: $string");
          echo  '<div class="alert alert-success alert-dismissible fade show" role="alert">
                <strong>Круто!</strong> Новий пароль було відправлено. Переходь до <a
                href="login.php">авторизації</a>
                <button type="button" class="close" data-dismiss="alert" aria-label="Close">
                  <span aria-hidden="true">&times;</span>
                </button>
              </div>';
      }else{
          echo '<div class="alert alert-danger alert-dismissible fade show" role="alert">
            <strong>Усп!</strong> Користувача з даним Email не знайдено.
            <button type="button" class="close" data-dismiss="alert" aria-label="Close">
              <span aria-hidden="true">&times;</span>
            </button>
          </div>';
      }
  }
```

Рисунок 3.6 – Модуль для відновлення паролю

3.2.4 Модуль з редагування персональної інформації користувача

В додатку, представлена швидка реєстрація, представлено всього п’ять полів для вводу мінімальної інформації про користувача. Додаток не змушує вводити детальну інформацію про користувача, але за бажанням – користувач може це зробити після реєстрації. Модуль з редагування профілю представлений з декількох частин: видалення фотокартки, видалення профілю та редагування профілю.

Видалення фотокартки стає доступним користувачу з завантаженим зображенням, якщо ж встановлене стандартне зображення то можливість видалили фотокартки – не доступна. Лістинг коду представлений на рисунку 3.7.

```php
if (isset($_POST['del_photo'])){
   $sql_refresh = mysql_query("UPDATE `users` SET `user_image` = 'user.png' WHERE
   `user_id`=".$_GET['red_id'], $link);
    echo '<meta http-equiv="Refresh" content="0; URL="admin-panel.php?
    user_id='.$_GET['red_id'].'">';
}
```

Рисунок 3.7 – Модуль з видалення фотокартки

Редагування основної інформації представлений формою з полями, де наявна інформація завантажується в форму для її зміни або відображується пусті поля, якщо інформація відсутня. Наприклад, система зрозуміє, якщо фотокартка була додана й виконає її завантаження на сервер, в іншому випадку – залишається стандартне зображення. Основний алгоритм роботи модулю: зібрати введену інформацію з полів за допомогою методу POST і додати її в sql запит до змінної $sql_refresh й після чого запит відправляється до бази даних. Лістинг коду представлений на рисунку 3.8.

```php
if (isset($_POST['refresh_submit'])){
    if($_FILES['red_image']['name'] != ''){
    $path = 'images/';
      $tmp_path = 'tmp/';
    // Массив допустимых значений типа файла
    $types = array('image/gif', 'image/png', 'image/jpeg');
    // Максимальный размер файла
    $size = 3050000;
    $extension_img = strtolower(substr(strrchr($_FILES['red_image']['name'], '.'), 1));
    $name_img =rand(1, 1000).'-'.md5($_FILES['red_image']['name']). '.' . $extension_img;
    if ($_SERVER['REQUEST_METHOD'] == 'POST')
    {
        // Проверяем тип файла
        if (!in_array($_FILES['red_image']['type'], $types))
            die('Запрещённый тип файла. <a href="?">Попробовать другой файл?</a>
            '.$_FILES['red_image']['type']);
        // Проверяем размер файла
        if ($_FILES['red_image']['size'] > $size)
            die('Слишком большой размер файла. <a href="?">Попробовать другой файл?</a>');
        // Загрузка файла и вывод сообщения
        if (!@copy($_FILES['red_image']['tmp_name'], $path . $name_img)){}
            echo 'Что-то пошло не так';
    }}else {$name_img = $result['user_image'];}
    $sql_refresh = mysql_query("UPDATE `users` SET `first_name` = '".$_POST['red_name']."',
    `last_name` = '".$_POST['red_surname']."', `gender` = '".$_POST['red_gender']."',
    `user_image` = '".$name_img."', `description` = '".$_POST['red_description']."',
    `driving_experience` = '".$_POST['red_expirience']."', `location` =
    '".$_POST['red_location']."' WHERE `user_id` = ".$_GET['red_id']);
    if(!$sql_refresh){
    echo '<p style="text-align:center; font-size:18px; color:red;">Ошибка! ' .
    mysql_error().'</p></br>';
    }else{
        echo '<meta http-equiv="Refresh" content="0; URL="admin-panel.php?
        user_id='.$_GET['red_id'].'">';
        }
}
```

Рисунок 3.8 – Модуль редагування персональної інформації користувача

3.2.5 Модуль додавання автомобіля

Модуль містить поля для вводу інформації про автомобіль: марку, модель, рік випуску, рік покупки, можливість завантажити фотографію автомобіля, опис автомобіля від власника та технічні характеристики. Так як кількість бреднів автомобілів велика кількість й кожен бренд має не визначену кількість моделей, вибір моделі був реалізований за допомогою асинхронного JavaScript (AJAX). Опис: користувач обирає з випадаючого списку бренд автомобіля й після чого ajax відображує поле з доступними моделями автомобілів відповідно до обраної марки, лістинг коду представлений на рисунку 3.9. Функціонал з додавання фотографії автомобіля реалізовано схожим чином як і для редагування персональної інформації користувача та відправка даних до бази даних також.

```
$(document).ready(function(){
    $('#inputModel').change(function(){
    $.ajax({
    type: "POST",
    url: "show.php",
    data: "inputModel="+$("#inputModel").val(),
    success: function(html){
    $("#content_model").html(html);
    }
    });
    return false;
    });
});
```

Рисунок 3.9 – Функція з обробки обраної марки автомобіля

3.2.6 Модуль додавання запису

Модуль із додавання записів розроблений схожим чином як і інші модулі для додавання інформації. Також містяться поля для вводу інформації, інформація зберігається в змінні з методу POST й потім формується запит і відправляється до бази даних. Модуль маж особливість, вибір гео-локації місця проведення, реалізовано за допомогою стороннього сервісу Google Map API, користувач обирає місце на карті та ставить маркер, після чого потрібно натиснути на маркер і ввести назву місця проведення. Координати та назва передаються з JS до PHP й в подальшому

передаються до бази даних. Лістинг коду обробки гео-локації представлений на рисунку 3.10.

```
var marker;
function initialize() {
if (GBrowserIsCompatible()) {
var map = new GMap2(document.getElementById("map_canvas"));
        var map = new GMap2(document.getElementById("map_canvas"));
        map.setCenter(new GLatLng(56.32811,44.0), 15);
              map.addControl(new GLargeMapControl());
         map.addControl(new GMapTypeControl());

        GEvent.addListener(map, "click", function(overlay, latlng) {
          if (latlng) {
            marker = new GMarker(latlng, {draggable:true});
            GEvent.addListener(marker, "click", function() {
              var html = "<table><form>" +
                         "<tr><td>Назва:</td> <td><input type='text' id='name'/> </td> </tr>" +
                         "<tr><td></td><td><input type='button' value='Зберегти місце'
                         onclick='saveData()'/></td></tr></table>";
              marker.openInfoWindow(html);
            });
                map.addOverlay(marker);
          }
        });
      }
    }
    function saveData() {
      var name = document.getElementById("name").value;
      var latlng = marker.getLatLng();
      var lat = latlng.lat();
      var lng = latlng.lng();
      document.getElementById("name_").value = name;
      document.getElementById("lat_").value = lat;
      document.getElementById("lng_").value = lng;
      document.getElementById("message").innerHTML = "Гео-локація успішно додана.";
    }
```

Рисунок 3.10 – Функція initialize() та saveData() для обробки гео-локації

3.2.7 Модуль з відображення статистичних даних

Модуль розділений на декілька частин: відображення таблиці зі стислою інформацією про кожну подію (тип події, заголовок, автомобіль, ціна виконаних робіт та дата проведення), відображення грошових витрат по кожному автомобілю й подальше розбиття витрат на типи події (технічний огляд, ремонт або тюнинг).

Відображення даних в таблиці відбувається для всіх автомобілів користувача з присутнім посиланням на подію та автомобіль. Лістинг коду для виводу інформації представлений на рисунку 3.11.

```php
<?php
   $sql_notes = mysql_query("SELECT * FROM `notes` n join `cars` c
   on(n.`car_id`=c.`car_id`) join `users` u on(c.`user_id` = u.`user_id`) join `car_model`
   m on(c.`model_id` = m.`model_id`) join `car_brand` b on(c.`brand_id` = b.`brand_id`)
   WHERE u.`user_id` = ".$_GET["user_id"], $link);
   while ($result_notes = mysql_fetch_array($sql_notes)) {
     echo '
     <tr><td>'.$result_notes["operation_type"].'</td>'.
     '<td><a href="own-note.php?
     note_id='.$result_notes["note_id"].'">'.$result_notes["title"].'</a></td>'.
     '<td><a href="own-car.php?
     car_id='.$result_notes["car_id"].'">'.$result_notes["model"].'
     '.$result_notes["brand"].' '.$result_notes["year_of_isue"].'</a></td>'.
     '<td>'.$result_notes["price"].' грн.</td>'.
     '<td>'.$result_notes["date"].'</td>';
     }
 ?>
```

Рисунок 3.11 – Модуль з виводу інформації про події

Грошові витрати по кожному автомобілю підраховуються загально та відображаються першими й після чого відбувається розподіл загальної суми витрат по типам записів. Лістинг коду з відображення витрат зображений на рисунку 3.12.

```
$sql_cars = mysql_query("SELECT SUM(n.`price`) as `sum`, c.`car_id` as `car_id`,
m.`model` as `model`, b.`brand` as `brand` FROM `notes` n join `cars` c
on(n.`car_id`=c.`car_id`) join `users` u on(c.`user_id` = u.`user_id`) join `car_model`
m on(c.`model_id` = m.`model_id`) join `car_brand` b on(c.`brand_id` = b.`brand_id`)
WHERE u.`user_id` = ".$_GET["user_id"]." group by c.`car_id`;", $link);
while ($result_cars = mysql_fetch_array($sql_cars)) {
  echo '
  <div class="col-6">
            <span class="stat-btn stat-btn-first">
                Автомобіль: <span class="badge badge-light"><a href="own-car.php?
                car_id='.$result_cars["car_id"].'">'.$result_cars["model"].'
                '.$result_cars["brand"].'</a></span>
            </span>
            <span class="stat-btn stat-btn-second">
                Загальна сума: <span class="badge badge-light">'.$result_cars["sum"].'
                грн</span>
            </span>';
 $sql_to = mysql_query("SELECT SUM(`price`) as `sum` FROM `notes` where car_id =
 ".$result_cars["car_id"]." and operation_type = 'Технічний огляд'", $link);
 $result_to = mysql_fetch_array($sql_to);
    if(!empty($result_to["sum"])){
       echo '<span class="stat-btn stat-btn-third">
                Технічний огляд: <span class="badge badge-light">'.$result_to["sum"].'
                грн</span>
            </span>
      ';
    }
 $sql_fix = mysql_query("SELECT SUM(`price`) as `sum` FROM `notes` where car_id =
 ".$result_cars["car_id"]." and operation_type = 'Ремонт'", $link);
 $result_fix = mysql_fetch_array($sql_fix);
    if(!empty($result_fix["sum"])){
       echo '<span class="stat-btn stat-btn-third">
                Ремонт: <span class="badge badge-light">'.$result_fix["sum"].'
                грн</span>
            </span></br>
      ';
    }
 $sql_upgrade = mysql_query("SELECT SUM(`price`) as `sum` FROM `notes` where car_id =
 ".$result_cars["car_id"]." and operation_type = 'Тюнинг'", $link);
 $result_upgrade = mysql_fetch_array($sql_upgrade);
    if(!empty($result_upgrade["sum"])){
       echo '<span class="stat-btn stat-btn-third">
                Тюнинг: <span class="badge badge-light">'.$result_upgrade["sum"].'
                грн</span>
            </span>
      ' ;
    }
echo '</div>';
  }
```

Рисунок 3.12 – Модуль з відображення витрат по автомобілю

3.3 Використання програмного додатку

Для навігації по додатку розроблене навігаційне меню (рис. 3.13) яке може бути трьох видів: для не зареєстрованого користувача, авторизованого та в адміністративній панелі.

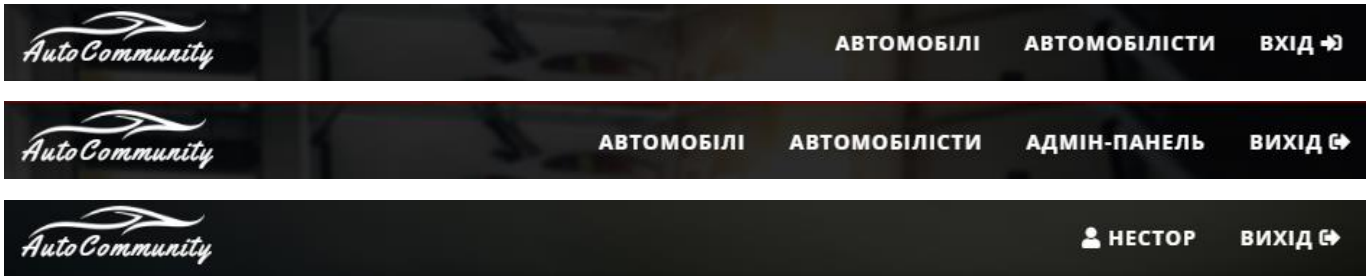

Рисунок 3.13 – Навігаційне меню

Навігація, прив’язана до верхівки браузеру й при прокручуванні сторінки – навігація зменшує свій розмір. Містить банер та блок з закликом до реєстрації на сервісі (рис. 3.14).

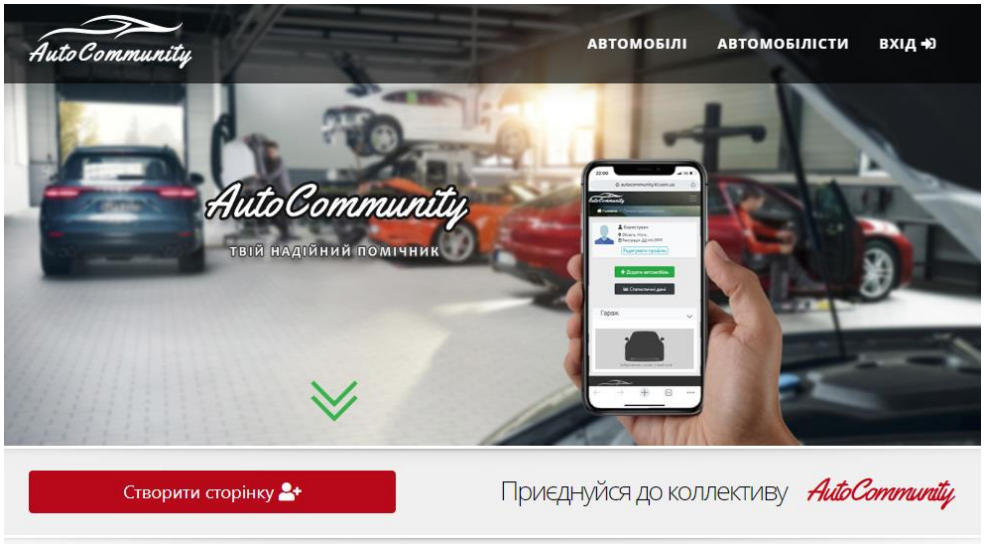

Рисунок 3.14 – Навігація сайту, банер та блок для реєстрації

Наступний блок містить короткий опис сервісу, тобто основний функціонал додатку та його можливості (рис. 3.15).

Блок «Переваги сервісу» відображує основні переваги з використання, чому слід користуватися й чого очікувати, зображено на рисунку 3.16.

Блок «Останні події» складається з блоків, які відображають останні записи користувачів (рис. 3.17): заголовок події, користувач та його фотокартка, ціна виконаних робіт, автомобіль та дату. Де заголовок, користувач і автомобіль відображені як посилання й є можливість переглянути як подію повністю, так і користувача чи його автомобіль. Кількість блоків регламентується довільно.

Рисунок 3.15 – Блок к описом функцій додатку

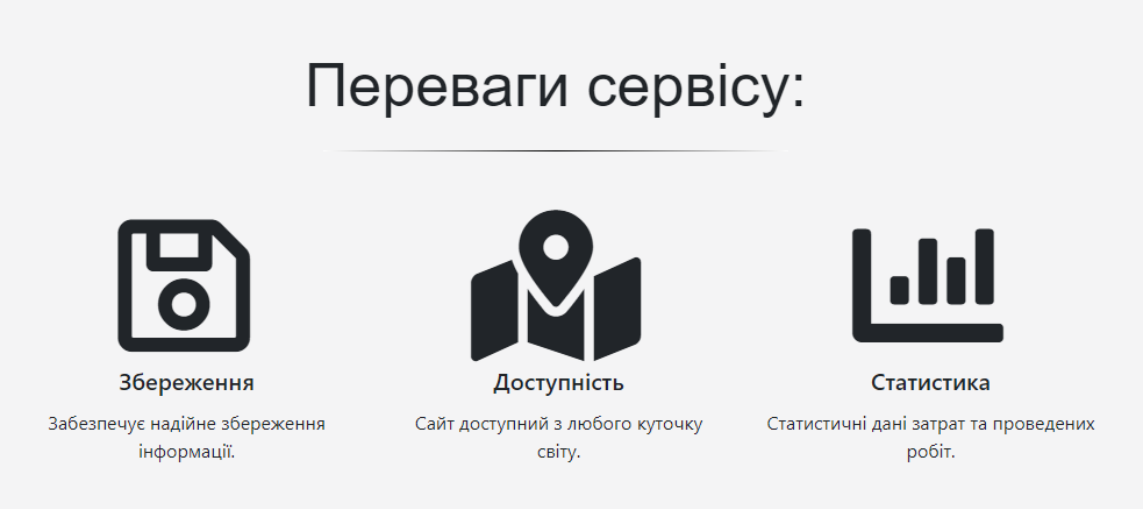

Рисунок 3.16 – Блок к описом переваг сервісу

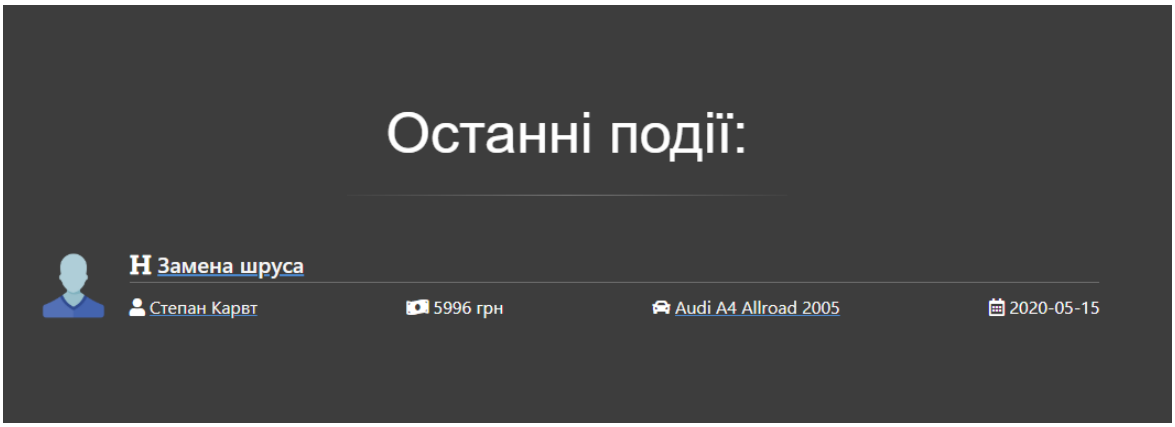

Рисунок 3.17 – Блок останніх доданих події до додатку

Останній блок – футер, містить в собі системну інформацію, та навігацію (рис. 3.18).

Рисунок 3.18 – Футер сайту

Для користувача надана можливість пройти реєстрацію, представлено окремою сторінкою з формою, яка містить поля для вводу інформації, кнопку та посилання на сторінку авторизації. Також передбачені діалогові вікна з помилками, та вікно з повідомленням по успішну реєстрацію з посиланням на сторінку авторизації. Дана форма з діалоговими вікнами представлена на рисунку 3.19.

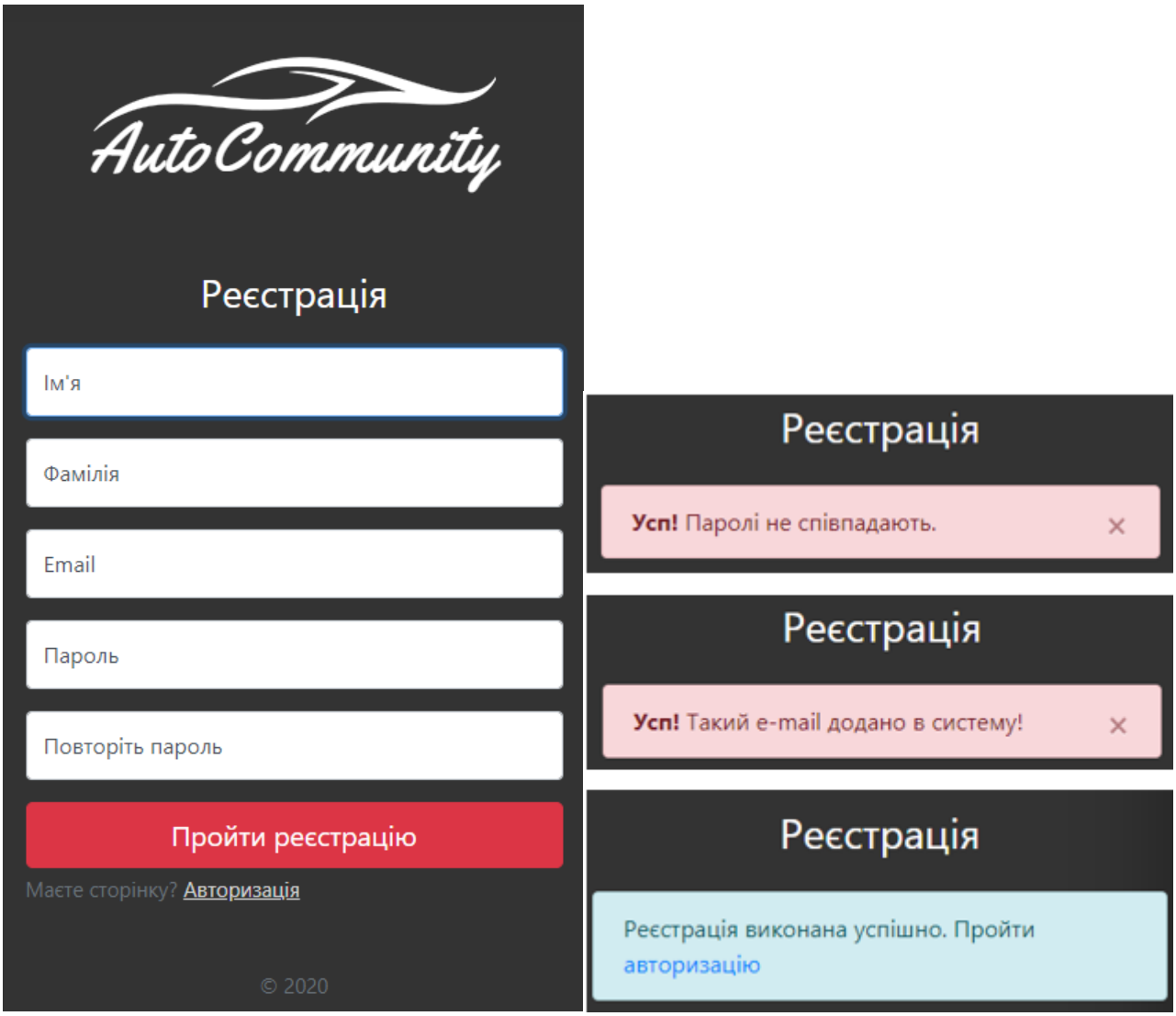

Рисунок 3.19 – Форма реєстрації та діалогові вікна

Форма авторизації представлена в схожому вигляді з реєстрацією, але має іншу структуру й функціонал в цілому. Містить поля для вводу даних, кнопку для авторизації та посилання для відновлення паролю і переходу до сторінки реєстраці.

Також розроблені діалогові вікна з відображенням помилок. Форма представлена на рисунку 3.20.

Сторінка з відновлення паролю є стратегічно важливим функціоналом при авторизації, адже людский фактор завжди присутній, можна як забути пароль або втратити записи з паролем. Сторінка представлена в вигляді форми з одним полем та кнопкой, де в поле потрібно ввести e-mail адресу, на яку був зареєстрований профіль. Передбачені діалогові вікна з відображенням помилок та інформацією про віправлений пароль на почту. Форма відображена на рисунку 3.21.

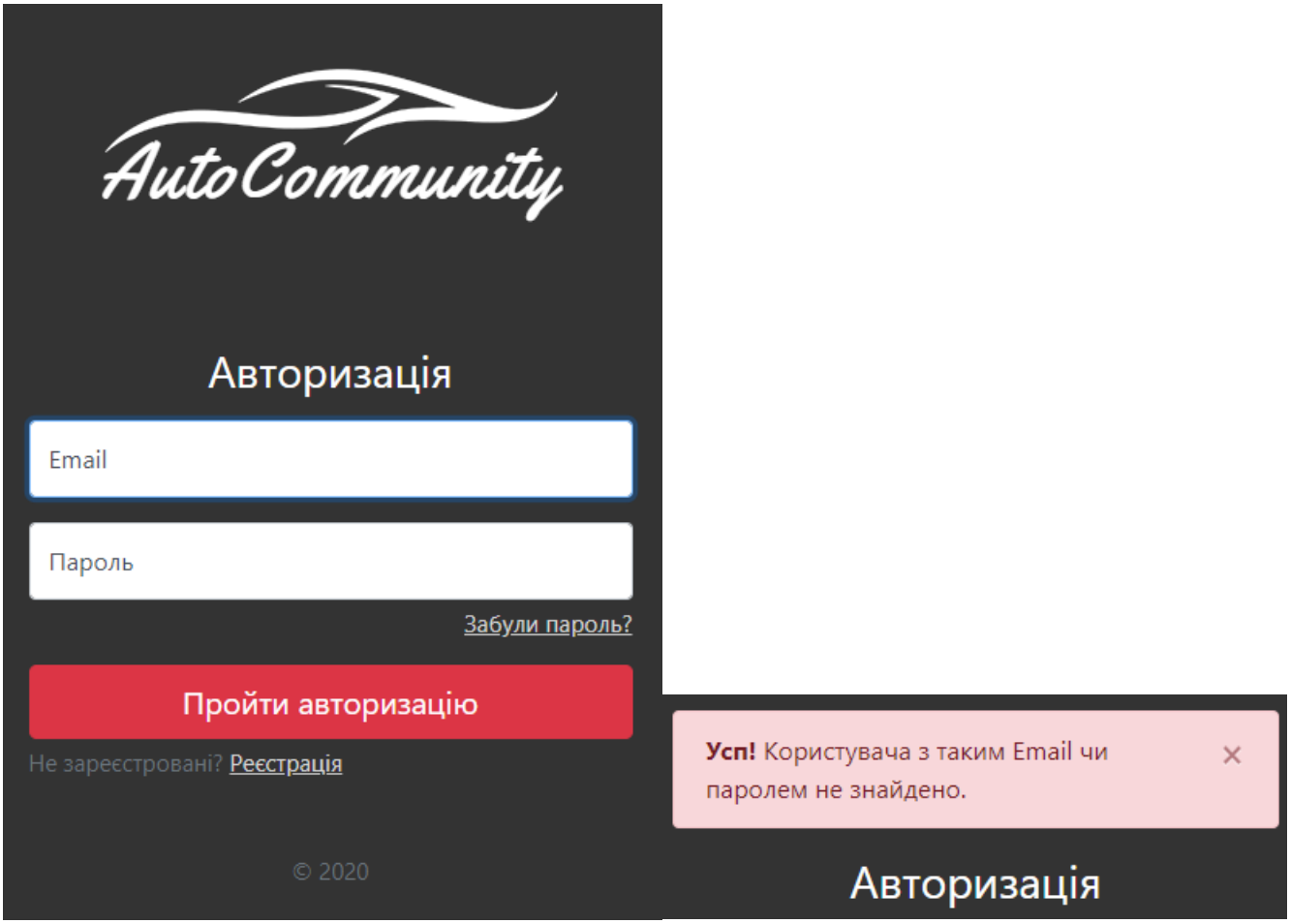

Рисунок 3.20 – Форма реєстрації та діалогові вікна

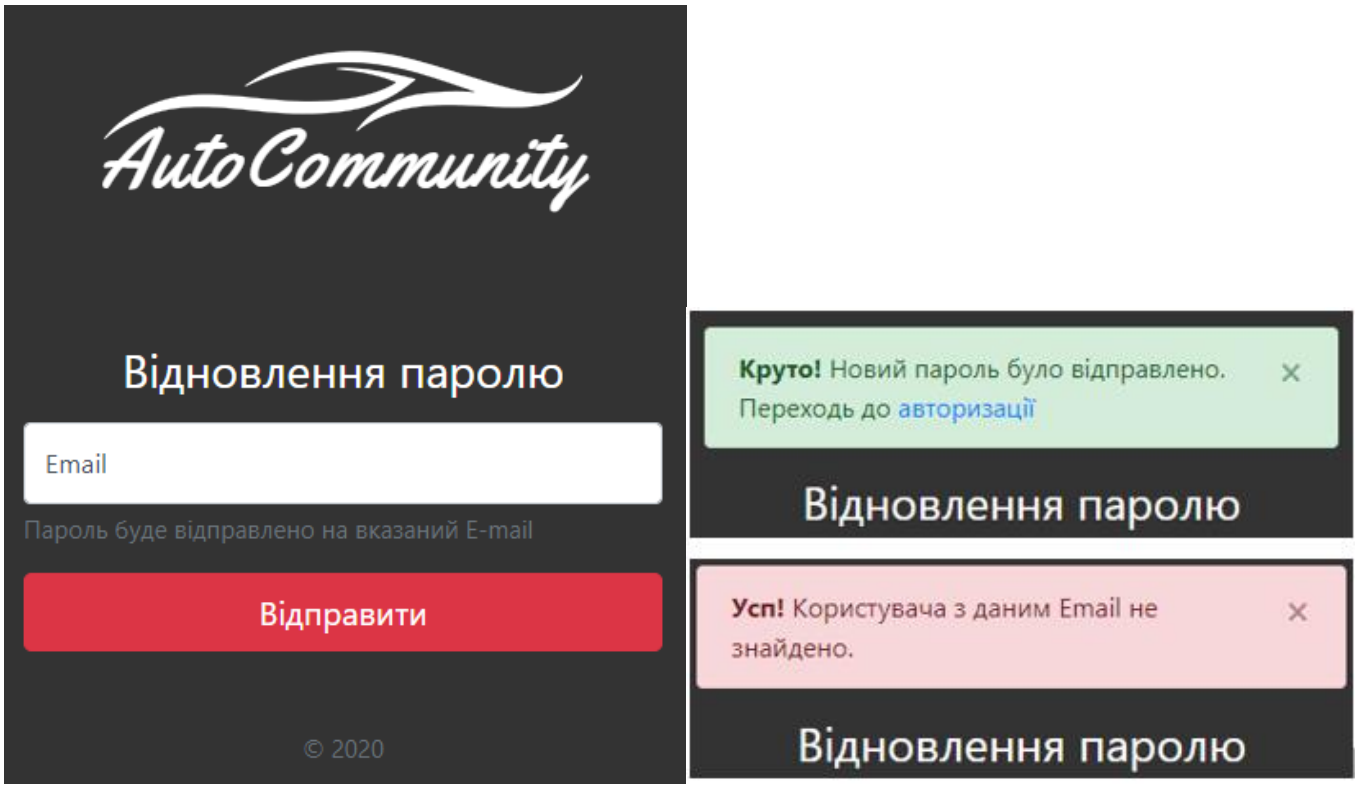

Рисунок 3.21 – Форма реєстрації та діалогові вікна

Після успішної авторизації користувача перенаправляє до адміністративної панелі (рис. 3.22), де користувачу відображується гараж з його автомобілів, бортовий журнал з його сервісними подіями для всіх автомобілів. Доступні посилання в вигляді кнопки з редагування профілю, додавання автомобіля та статистичні дані. Навігація адміністративної панелі відображує ім'я користувача та кнопку для виходу з профілю, під навігацією знаходиться банер з навігаційною панелью.

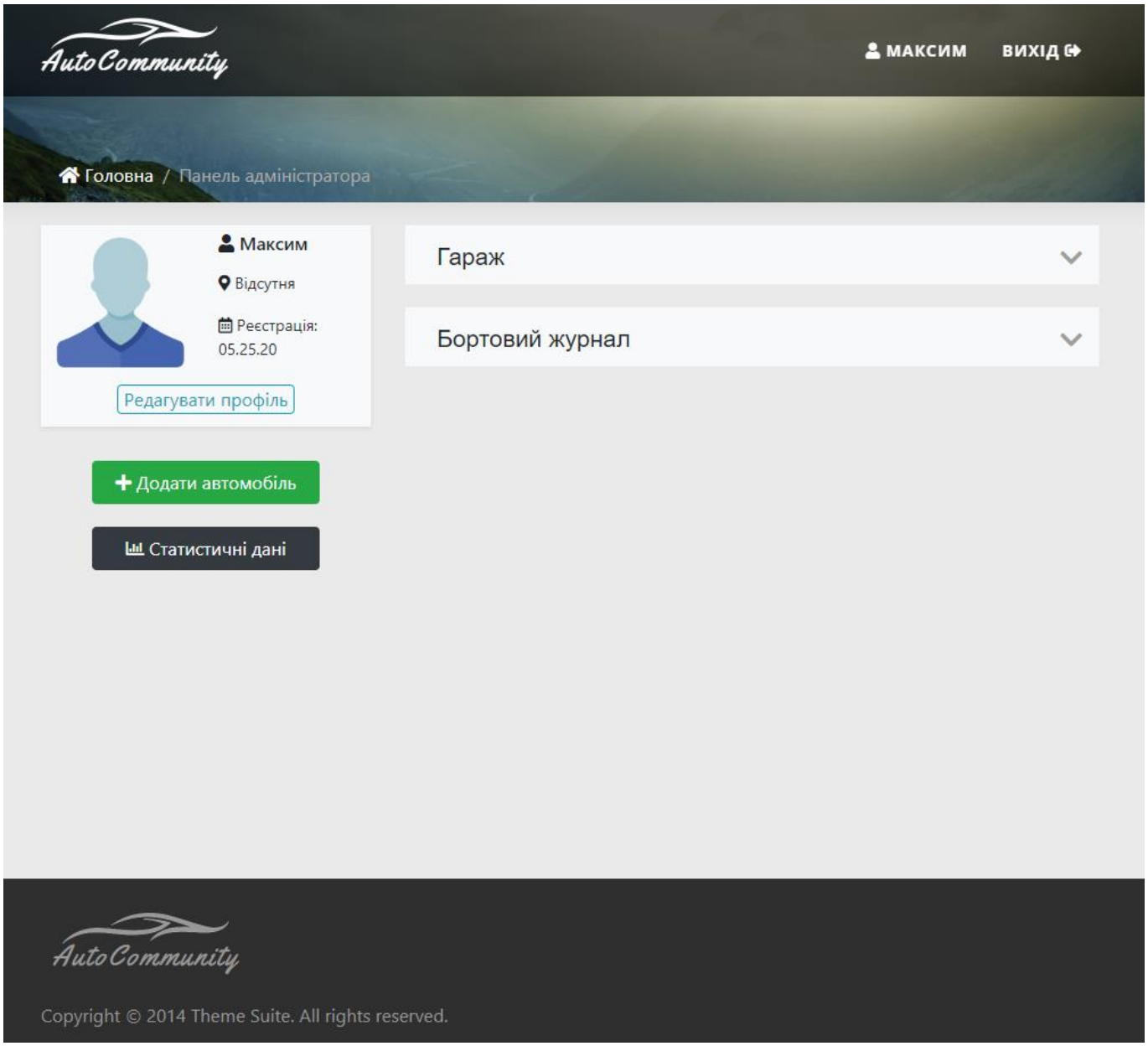

Рисунок 3.22 – Інтерфейс адміністративної панелі

Якщо натиснути на кнопку «Редагувати профіль», відбувається перехід до сторінки з формою для редагування профілю (рис. 3.23). Доступні поля для редагування інуючої інформації чи додавання нової. Є можливість обрати місто проживання, стать, завантажити власну фотографію розповісти про себе та вказати водійський стаж. Для зміни паролю чи email адреси передбачені сторінки з формами.

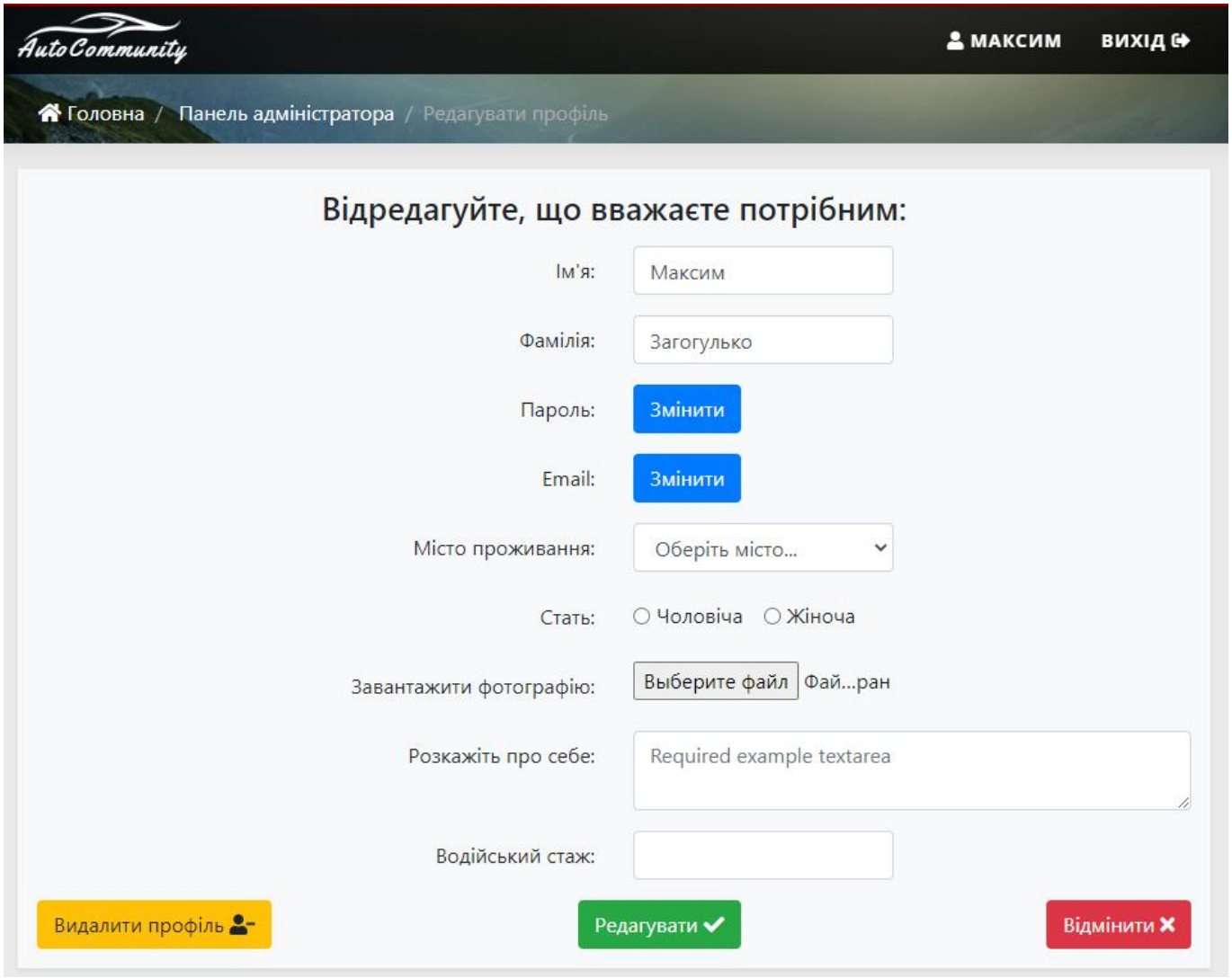

Рисунок 3.23 – Форма з редагування профілю

Якщо користувач не додавав власну фотокартку, то використовується зображення за замовчуванням, після доданої фотокартки, вона відображується в формі зображеній на рисунку 3.23, й з’являється кнопка для видалення фотокартки, після чого, користувачу присвоюється зображення за замовчуванням. Відображена форма з доданою фотокарткою зображена на рисунку 3.24.

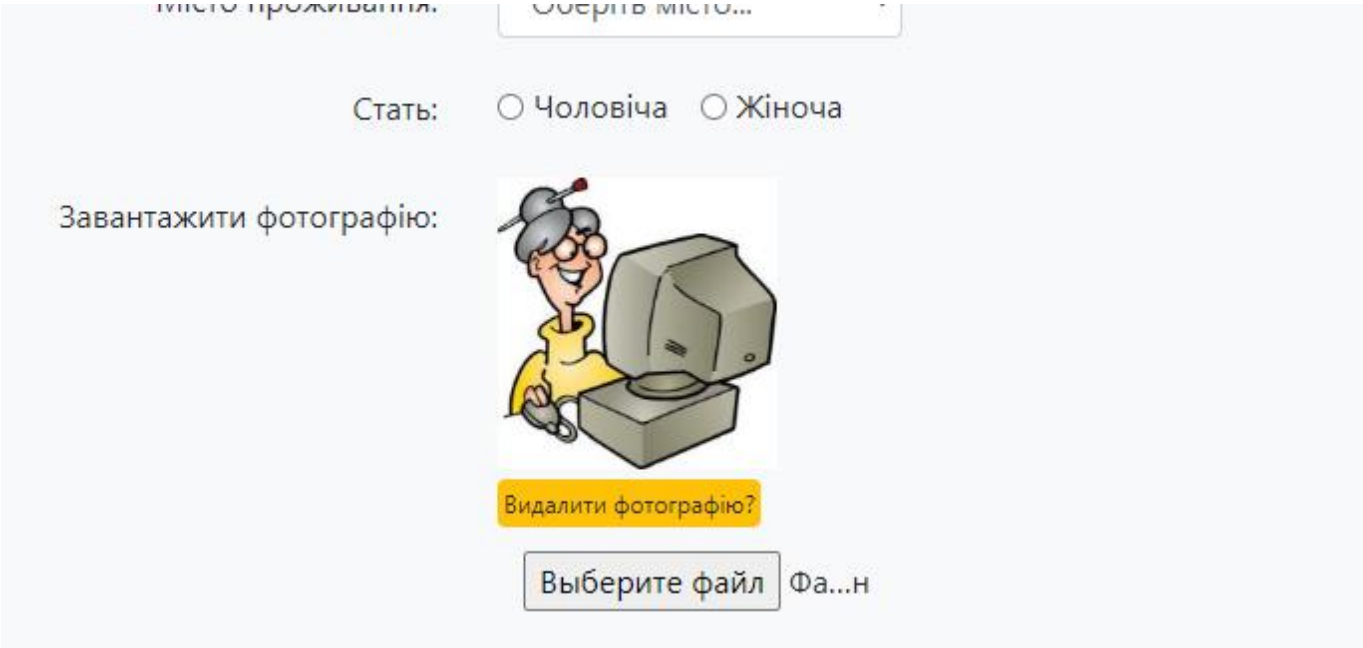

Рисунок 3.24 – Функціонал для редагування персональної фотокартки

Як зазначалося раніше, для редагування паролю призначена окрема сторінка з формою (рис. 3.25), де відображено три поля для вводу старого паролю та нового паролю й поле для дублювання паролю. Передбачені діалогові вікна (рис. 3.26) для відображення помилок: якщо поточний пароль не співпадає, якщо введений пароль не співпадає з дублем введеного нового паролю та інформаційне вікно з повідомленням про успішну зміну паролю.

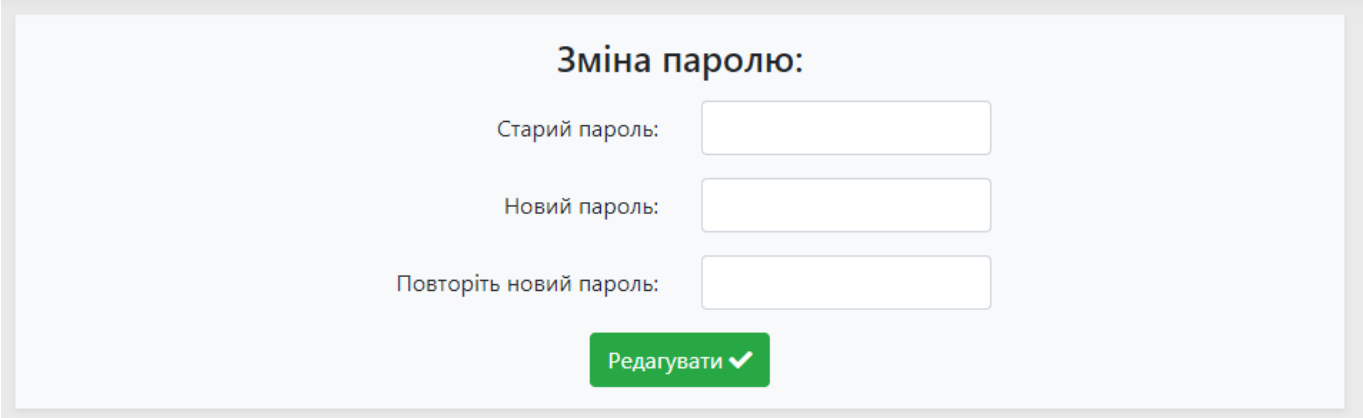

Рисунок 3.25 – Форма зміни паролю

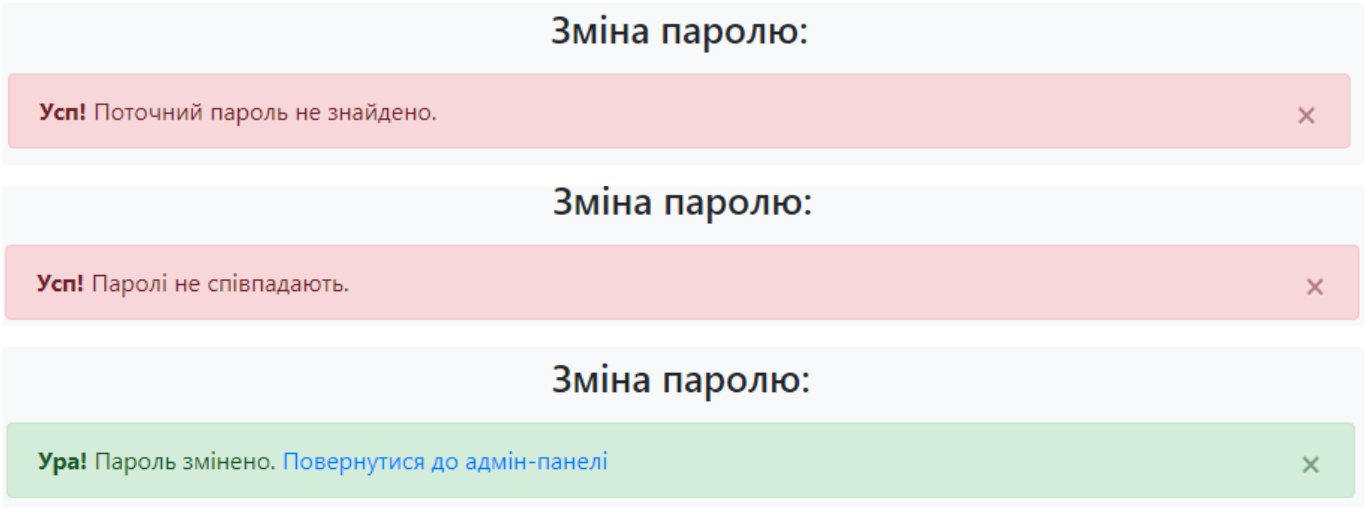

Рисунок 3.26 – Діалогові вікна при зміні паролю

Редагування email адреси передбачає схожий функціонал зі зміною паролю. Містить поле з відображенням поточної адреси, поля для вводу нової адреси та поточного паролю (рис. 3.27). Також передбачені діалогові вікна (рис. 3.28) для відображення помилок: якщо адреса вже існує в системі та поточний пароль не співпадає з введеним і інформаційне вікно, яке повідомляє про успішну зміну адреси.

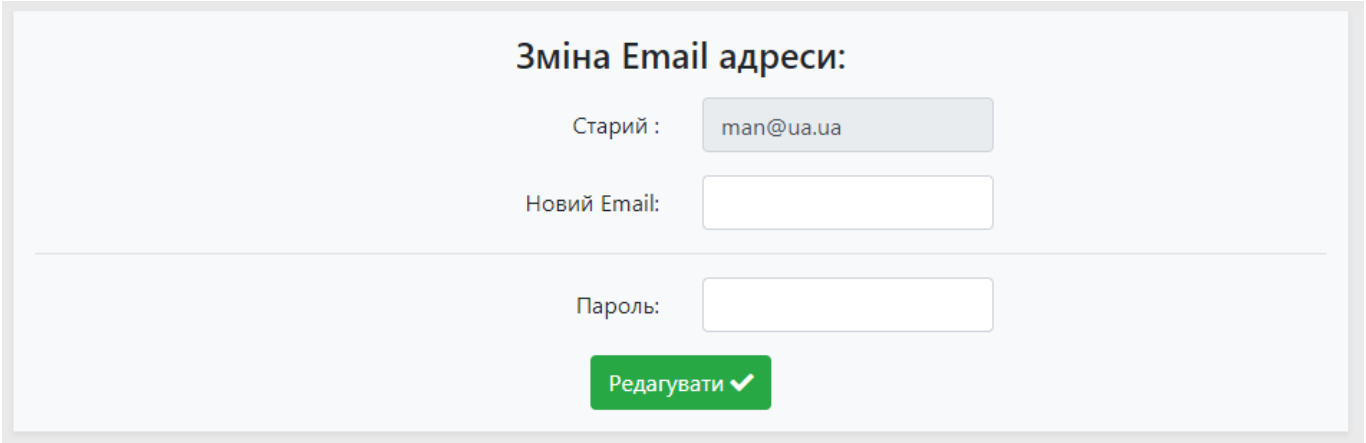

Рисунок 3.27 – Форма зміни email адреси

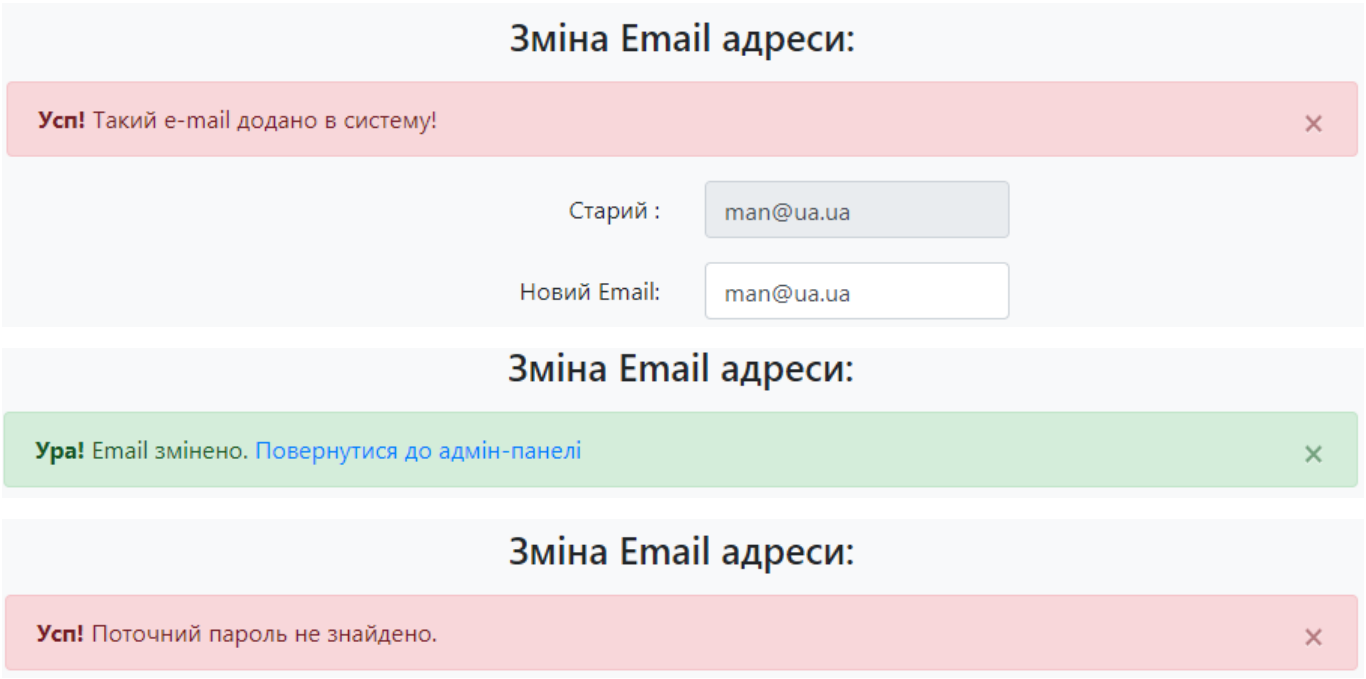

Рисунок 3.28 – Діалогові вікна при зміні email адреси

Щоб додати автомобіль до системи, потрібно натиснути на кнопку «Додати автомобіль» й після чого відкриється сторінка з формою (рис. 3.29). Містить стандартний набір: поля для вводу інформації та завантаження фотографії автомобіля. Якщо фотографія не була завантажена, то використовується стандартне зображення (рис. 3.30).

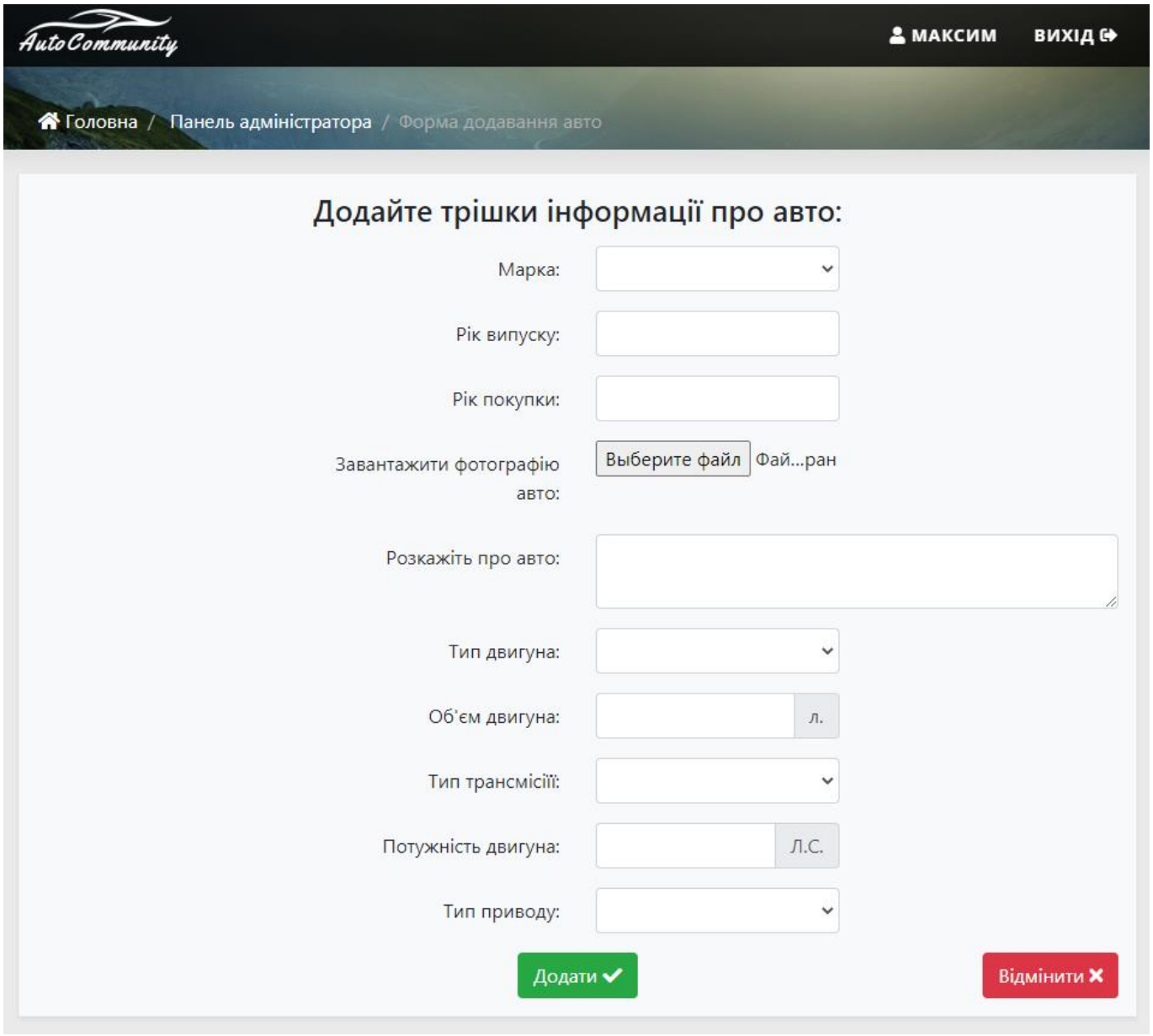

Рисунок 3.29 – Форма для додавання автомобіля

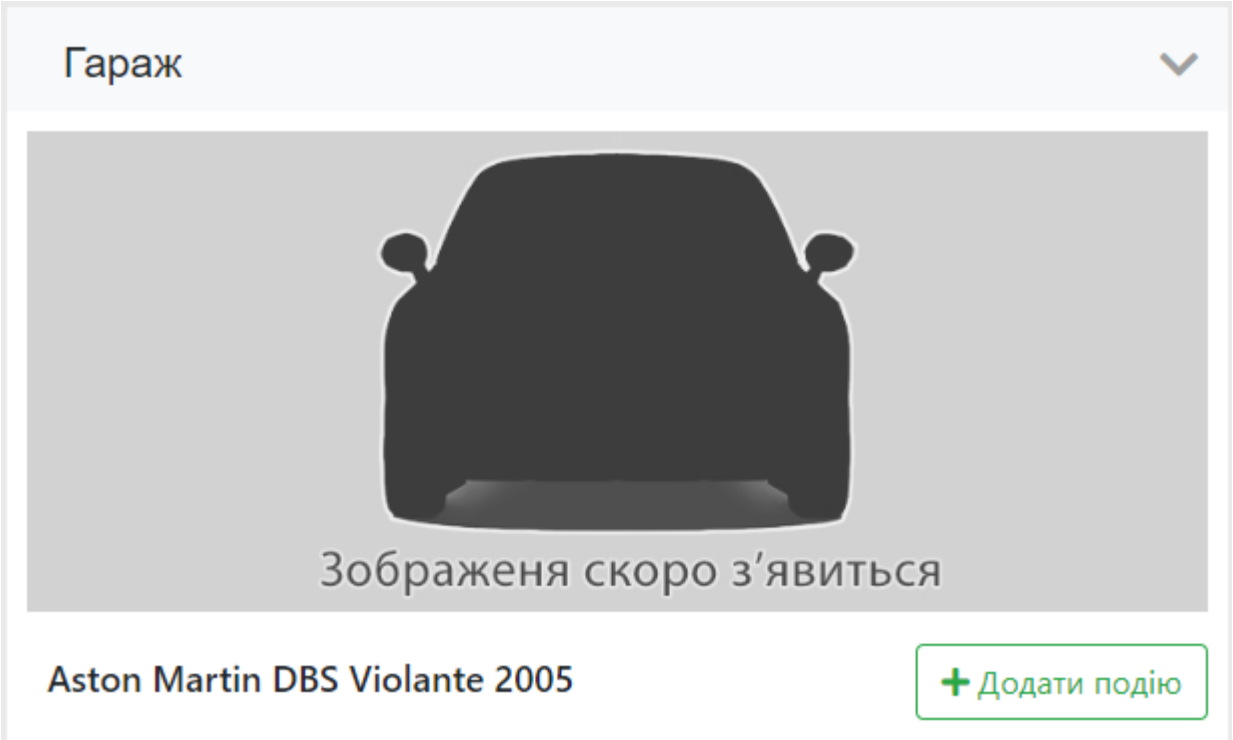

Рисунок 3.30 – Автомобіль, доданий в гараж зі стандартним зображенням

Для перегляду інформації про автомобіль, його бортовий журнал передбачена сторінка (рис. 3.31), яка відображує ці данні та надає можливість редагувати або зовсім видалити автомобіль. Щоб виконати редагування інформації про автомобіль, слід натиснути на кнопку «Редагувати» й після чого відбудеться перехід до сторінки з формою для редагування (рис. 3.32). Форма містить стандартний набір: поля для вводу інформації та можливість завантаження зображення автомобіля.

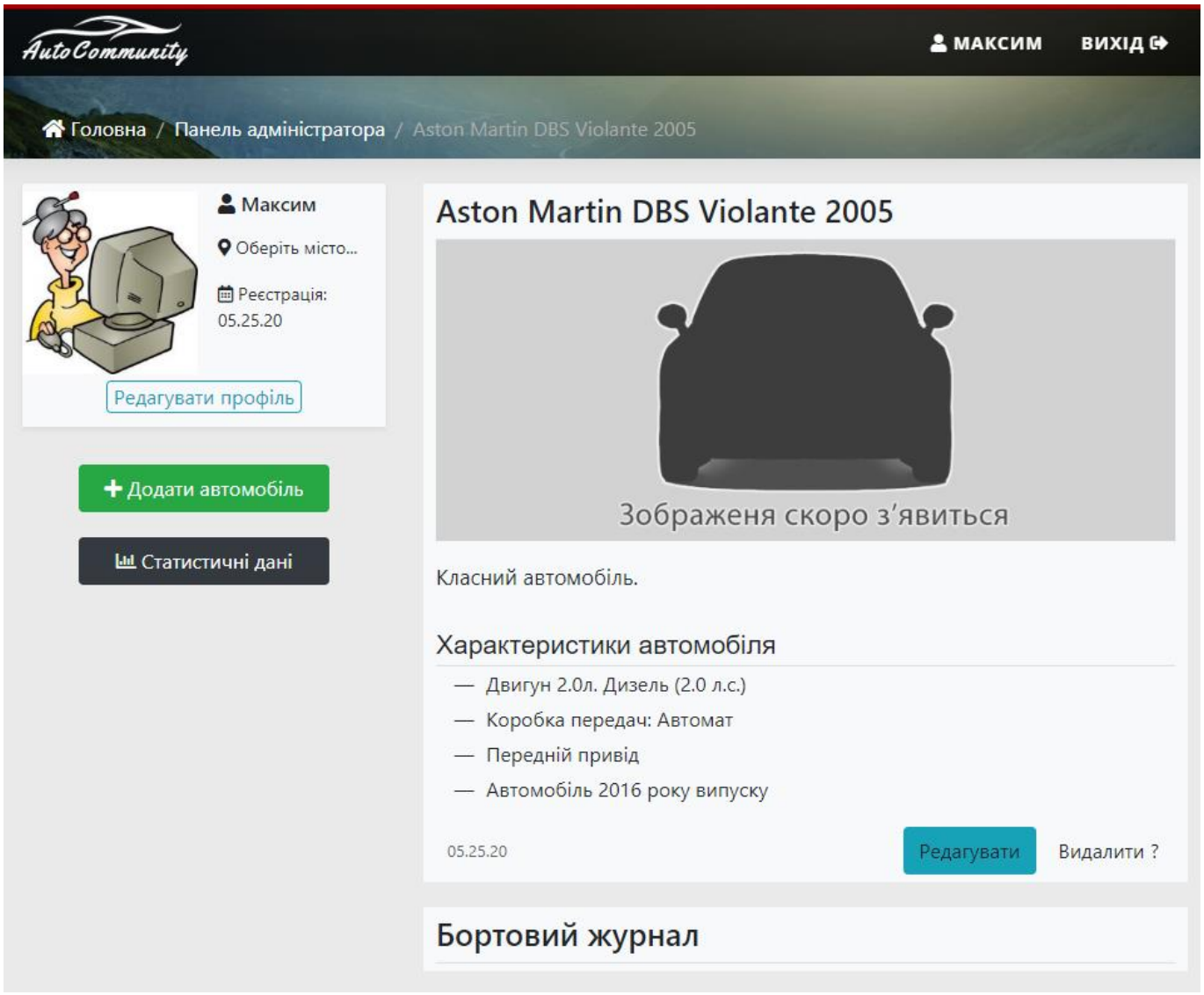

Рисунок 3.31 – Сторінка з інформацією про автомобіль та елементами керування

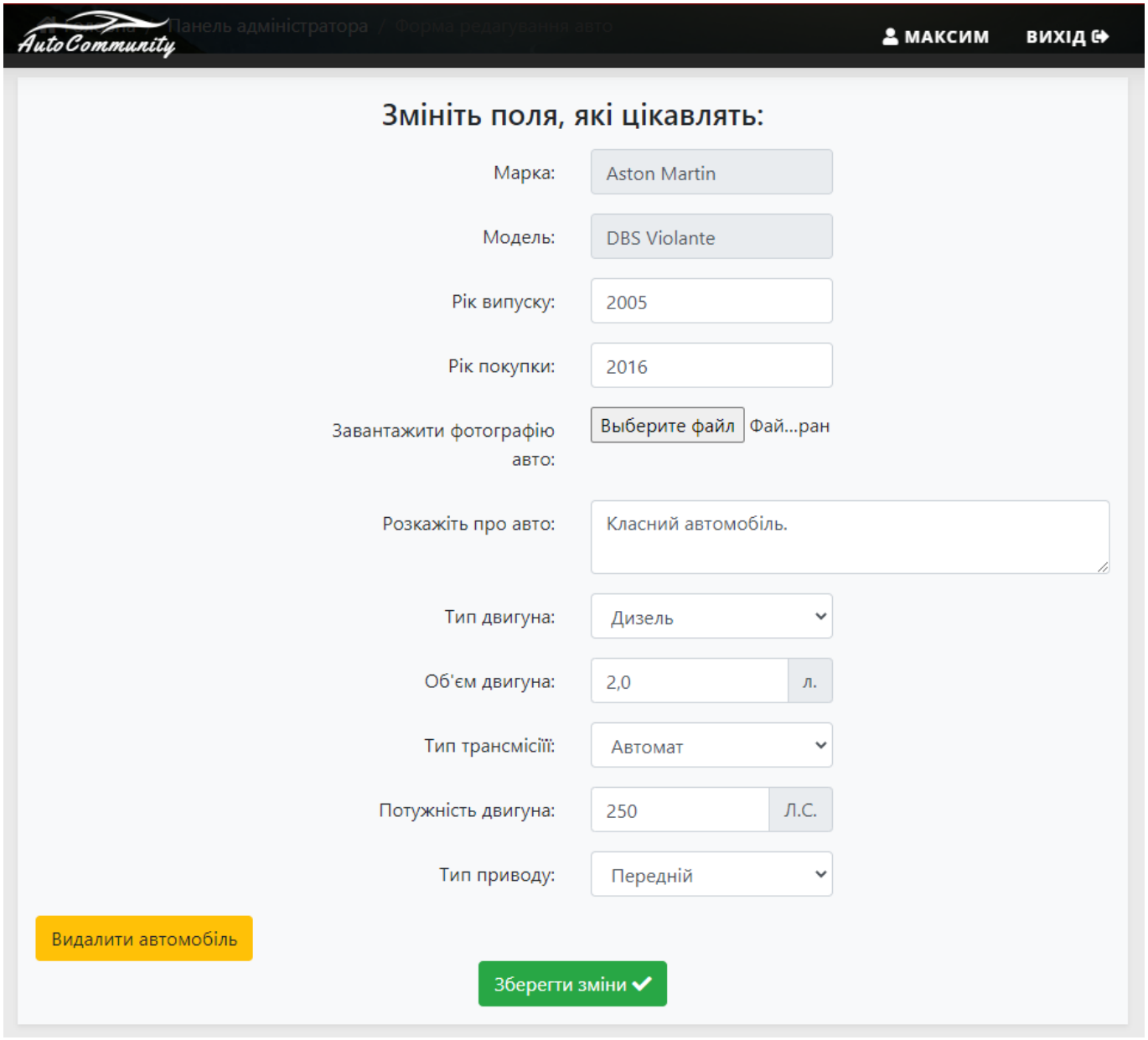

Рисунок 3.32 – Форма редагування інформації про автомобіль

Функціонал з додавання події реалізований таким чином, тільки на доданий автомобіль можна додати подію, як показано на рисунку 3.30, натиснувши на кнопку «Додати подію», відбувається перехід на сторінку з формою (рис. 3.33) для вводу типу події, її опису та локацію місця проведення – реалізовано за допомогою Google Maps API.

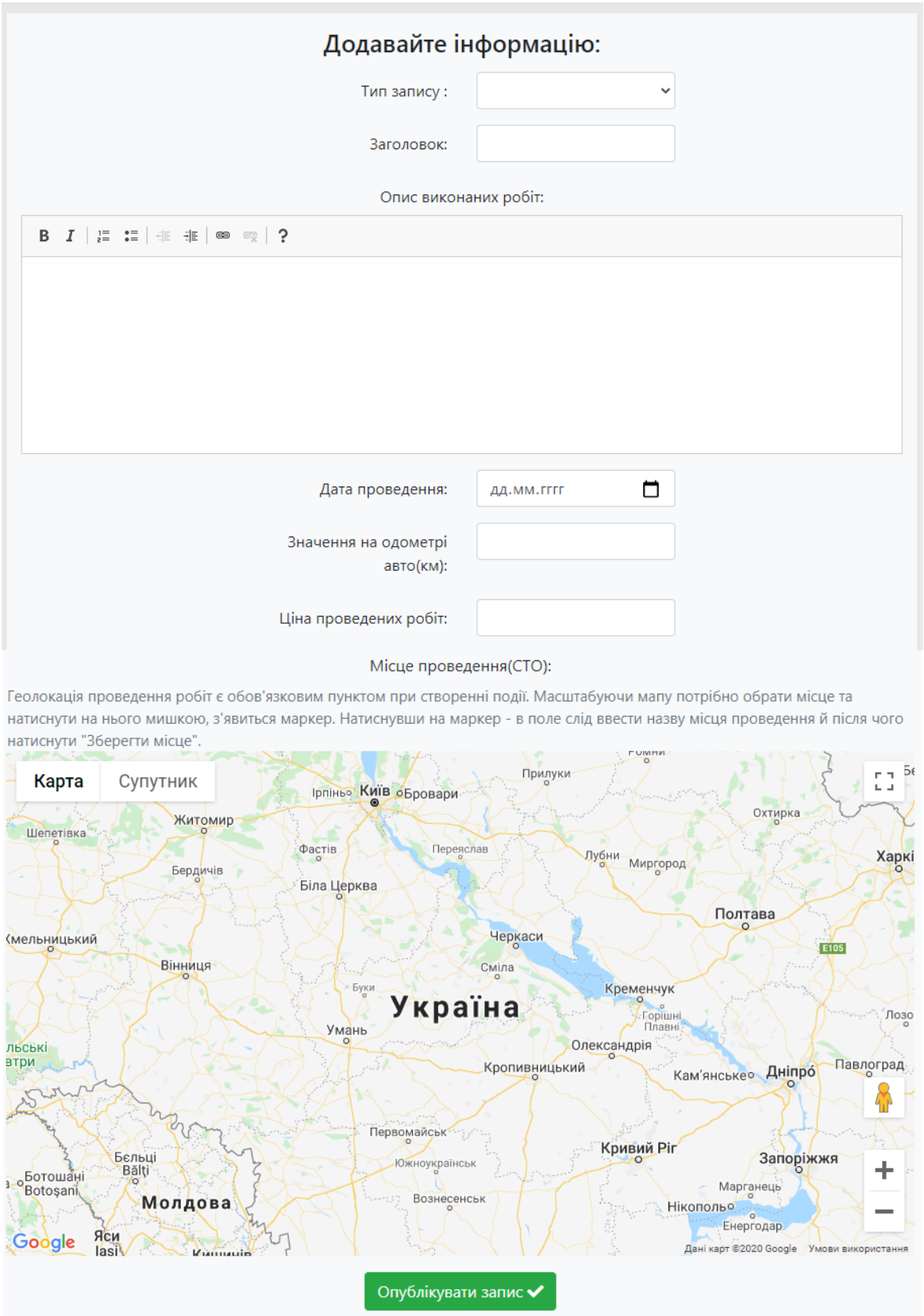

Рисунок 3.33 – Форма створення події обслуговування автомобіля

Створену подію можна переглянути як з головної сторінки адміністративної панелі (рис. 3.22) в блоці «Бортовий журнал», але на головній сторінці відображуються всі події користувача для всіх автомобілів користувача. Якщо потрібно переглянути події конкретного автомобіля, то слід перейти до сторінки з інформацією про нього (рис. 3.31) і також в блоці «Бортовий журнал» переглянути події але тільки одного автомобіля. Сторінка з описом події представлена на рисунку 3.34, де також є можливість редагувати або зовсім видалити запис. Редагування запису здійснюється стандартним чином, натиснувши на кнопку «Редагувати» відбувається перехід до сторінки з формою (рис. 3.35).

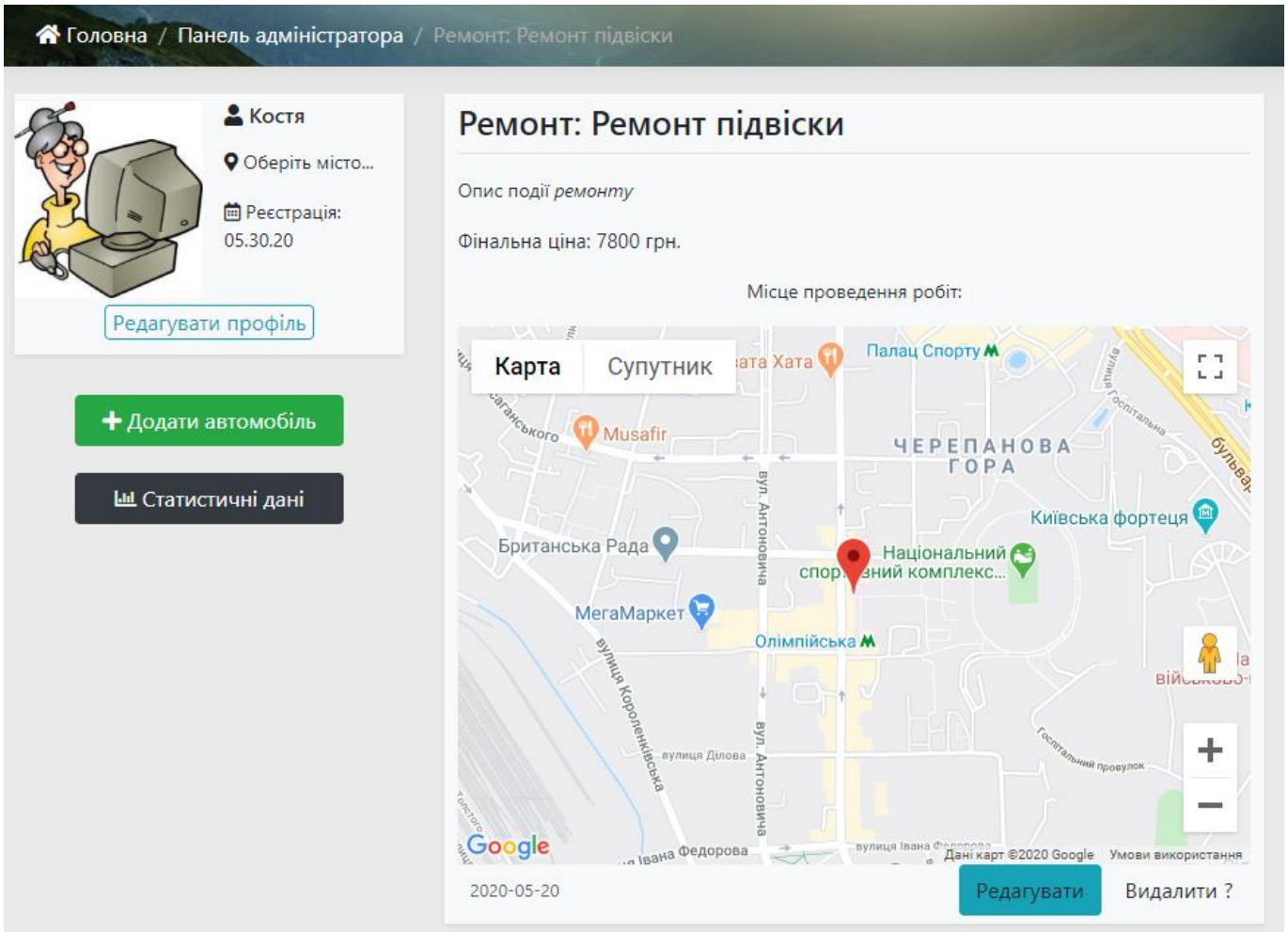

Рисунок 3.34 – Сторінка з описом доданої події

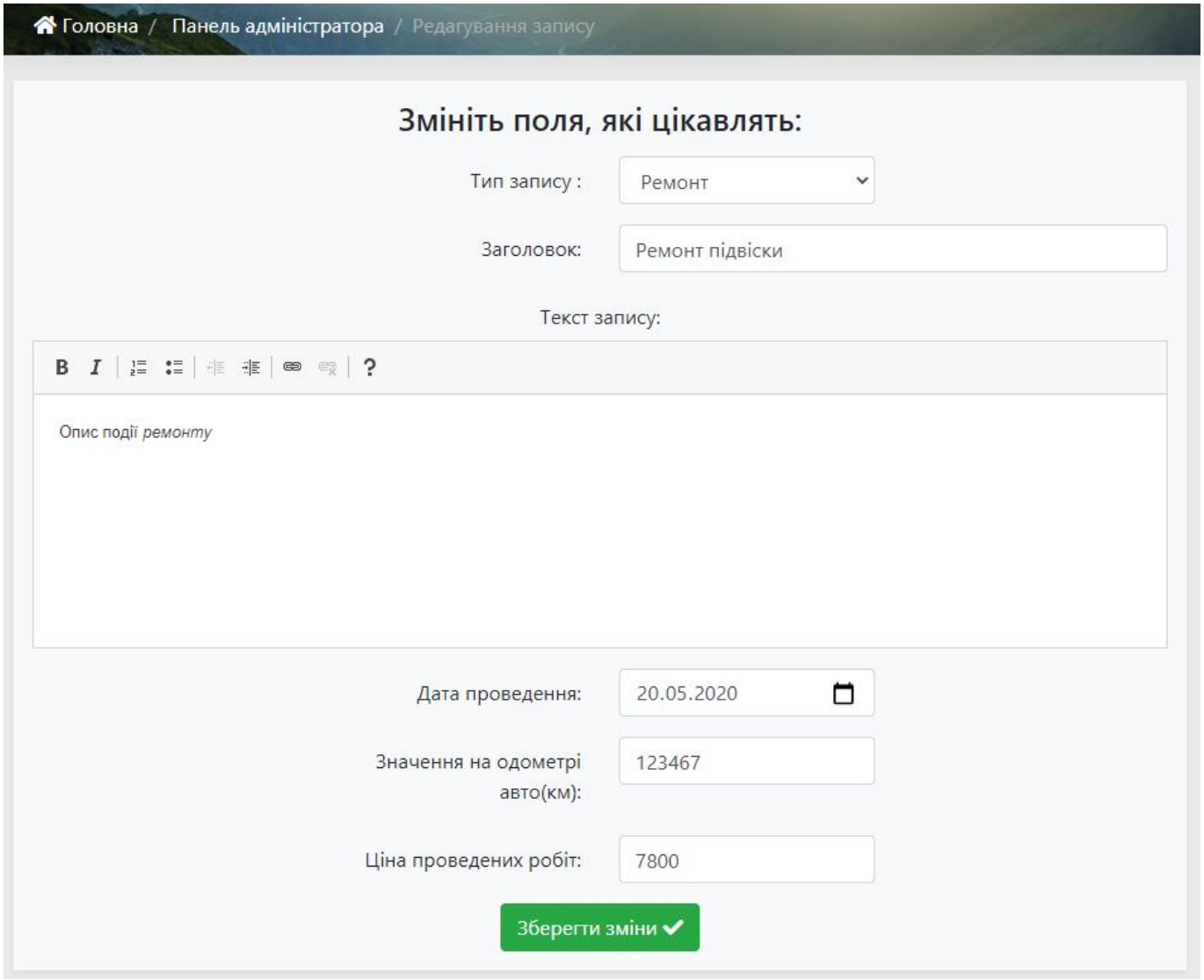

Рисунок 3.35 – Форма з редагування запису

Функціонал з статистичними даними реалізовано на окремій сторінці (рис. 3.36), яка представлена блоком «Дані з обслуговування автопарку», де відображується таблиця з доданих подій для всіх автомобілів, їх ціна та інші супутні дані. Після блоку розміщується інформація про кожен автомобіль окремо, тобто вираховується загальна сума витрат на авто й після чого сума розділяється та типи подій, тобто: сервісне обслуговування, технічний огляд або тюнинг.

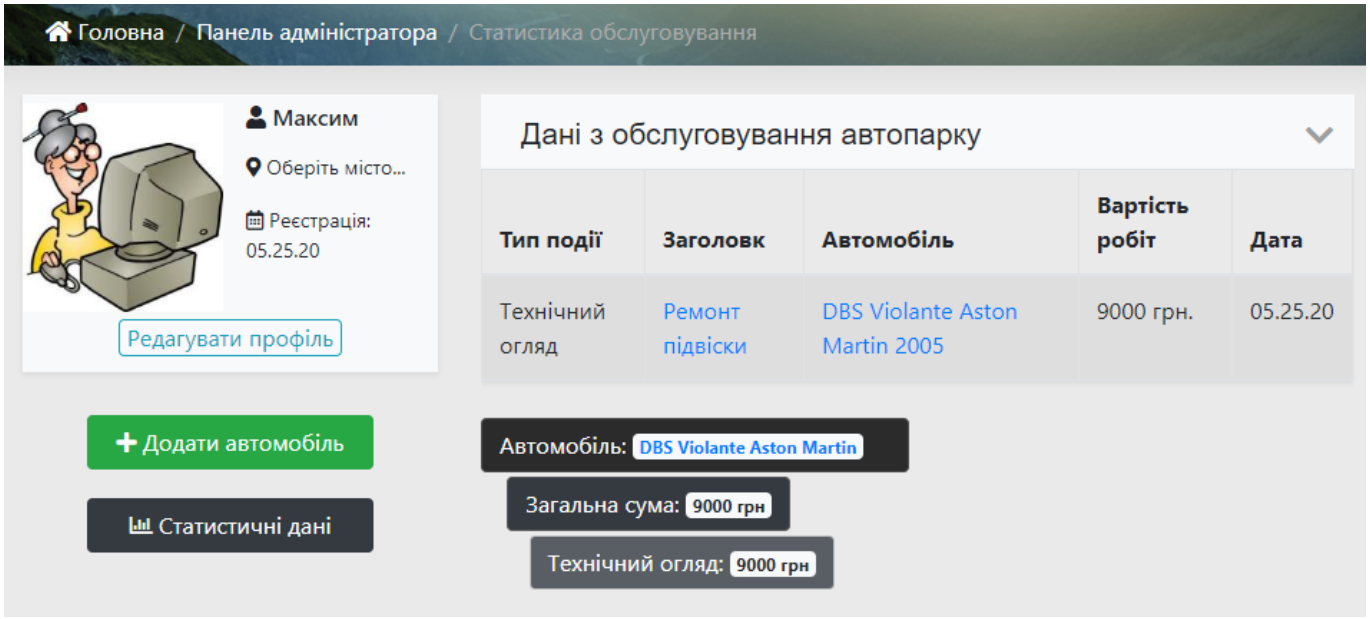

Рисунок 3.36 – Статистичні дані автомобілів

Так як інформація знаходиться в відкритому доступі, кожен охочий може переглянути інформацію про користувача або його автомобіль з доданими до нього подіями, й для цього не потрібно мати обліковий запис. Навігація на головній сторінці web-додатку (рис. 3.13), містить посилання «Автомобілі», «Автомобілісти».

Посилання «Автомобілі» (рис. 3.37) відображує всі наявні автомобілі в системі, також має функціонал для пошуку по марці або бренду автомобіля та року виробництва. Кожен автомобіль представлений у вигляді окремого блоку, при натисканні на який, відбувається перехід до сторінки з інформацією про автомобіль та його бортовий журнал (рис. 3.38). Також є можливість переглянути записи в бортовому журналі (рис. 3.39).

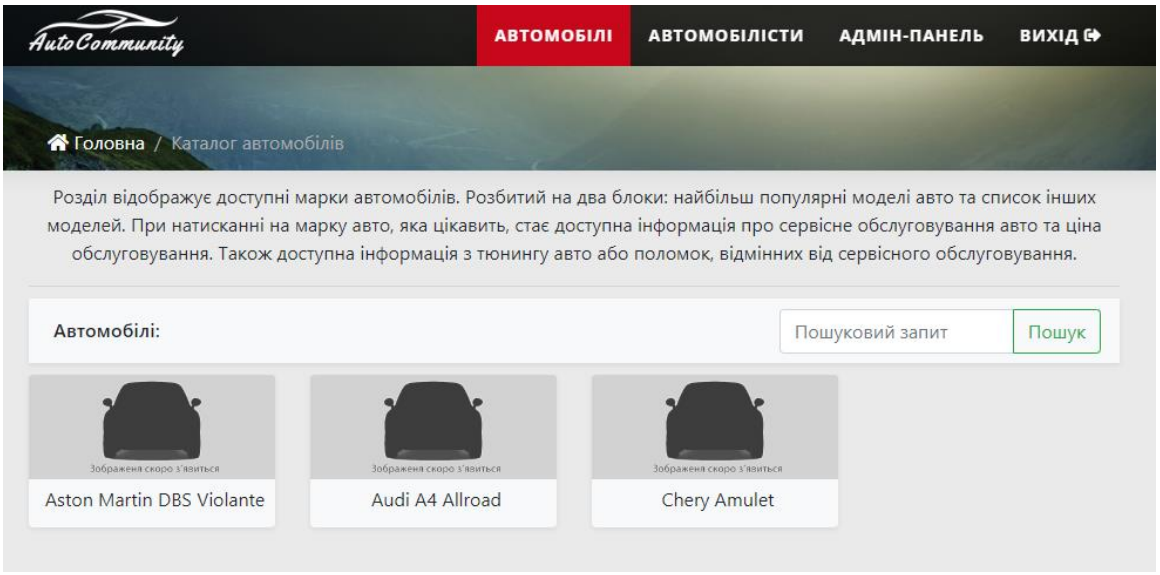

Рисунок 3.37 – Перелік наявних автомобілів в системі

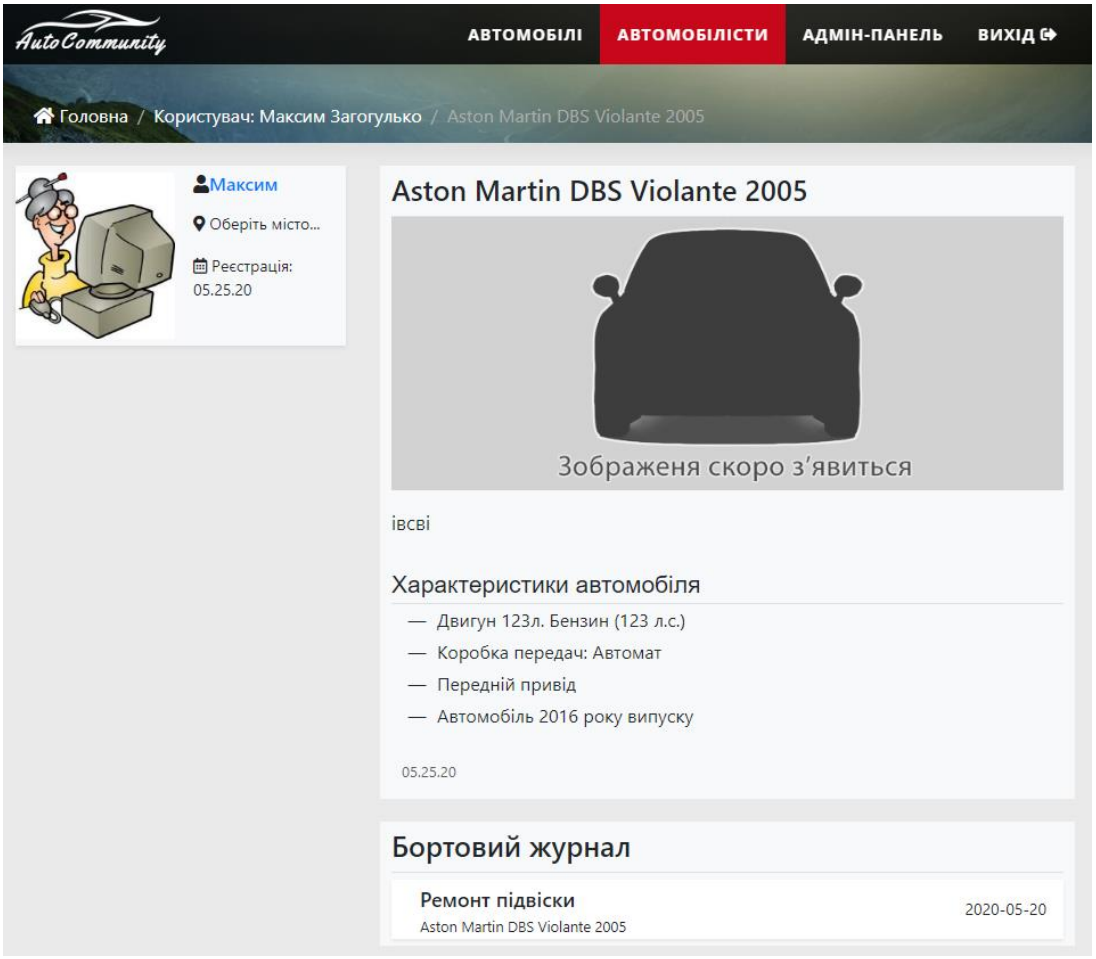

Рисунок 3.38 – Сторінка з характеристиками автомобіля та його бортовий журнал

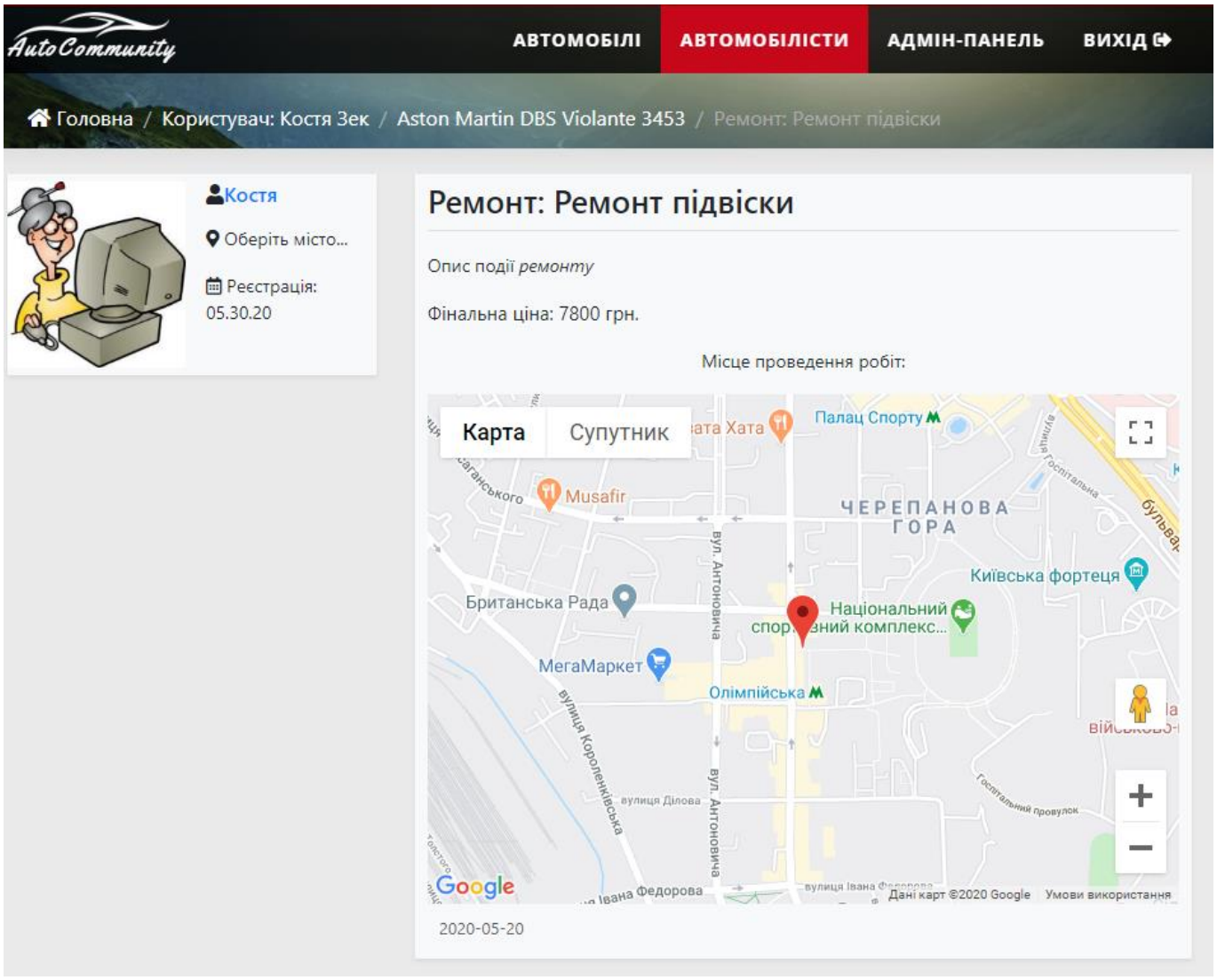

Рисунок 3.39 – Відображення інформації запису з бортового журналу

Посилання «Автомобілісти» відображує всіх зареєстрованих користувачів в системі та має функціонал з пошуку за ім’ям та фамілією чи за локацією (рис. 3.40). Кожен користувач відображується в окремому блоці, блок містить фотокартку користувача, його ім’я та фамілію, місце проживання, його автомобілі та дату реєстрації. Фото картка та ім’я з фамілією представлені у вигляді гіперпосилань, при натисканні на які, відкривається сторінка з їх персональною інформацією та гараж автомобілів (рис. 3.41).

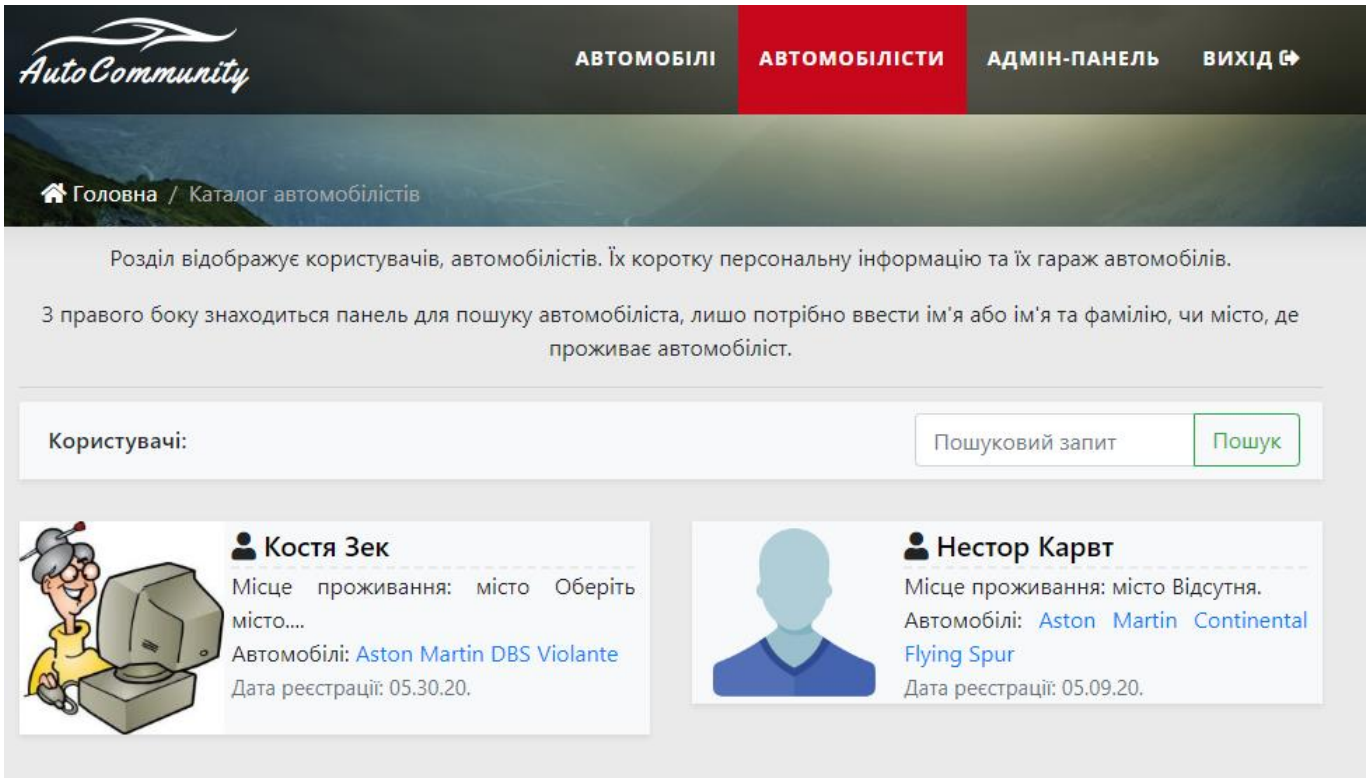

Рисунок 3.40 – Перелік наявних користувачів в системі

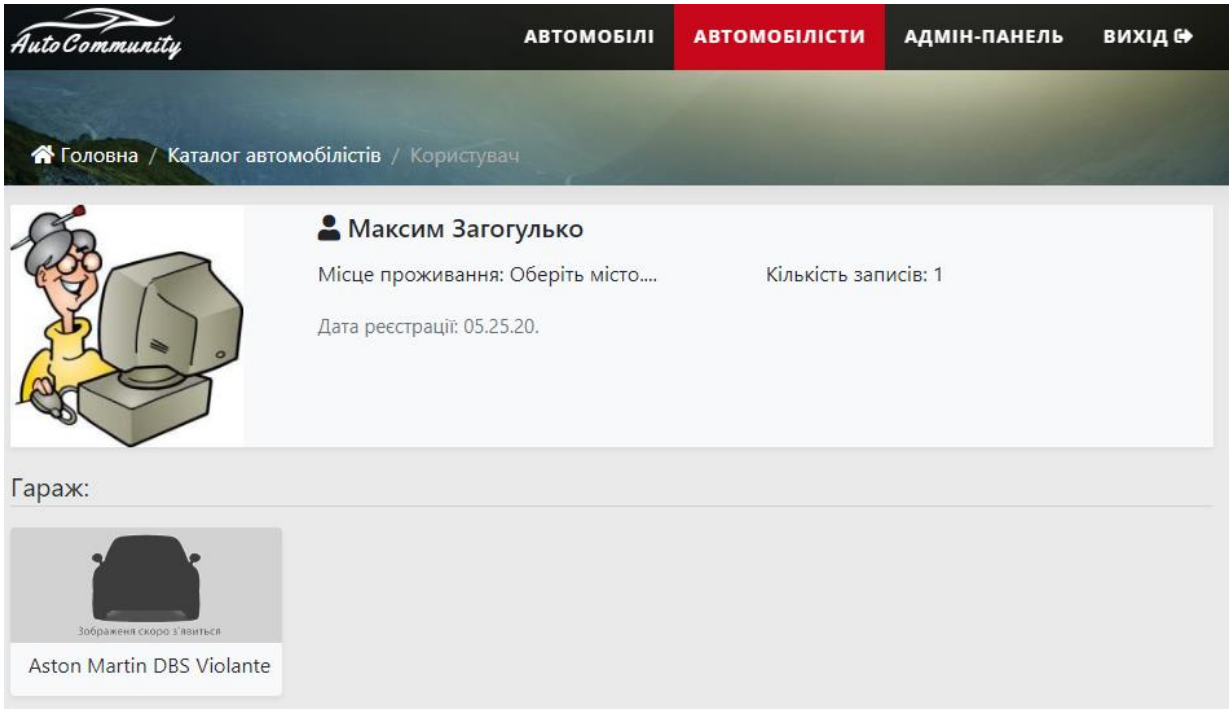

Рисунок 3.41 – Персональна інформація користувача

Висновки

В ході виконання кваліфікаційної роботи бакалавра було досліджено та проаналізовано всі фактори, які відіграють важливу роль в розробці web-орієнтованої системи для збереження сервісної історії автомобіля.

Розробка web-орієнтованої системи актуальна, як ніколи, кількість автомобілів стрімко збільшується й відносно кількість автовласників – також. Автомобіль потребує обслуговування, події обслуговування потребую збереження й структурованого відображення з грошових витрат на його утримання.

За час виконання кваліфікаційної роботи, були вирішені наступні задачі:

- аналіз аналогів, декілька web-сайтів, які мають схожий функціонал, для визначення переваг та недоліків з створюваним додатком;
- розробка інтерфейсу, створення гнучкого й сучасного дизайну з інтуїтивно зрозумілим розміщенням блоків та елементів керування системи;
- розробка діаграм бізнес процесів в нотації IDEF, які допомагають зрозуміти структуру створюваного додатку та його принцип дії й можливості в цілому;
- розробка діаграми варіантів використання, для відображення основних функцій додатку та розподіл прав на використання тих чи інших функцій.

Використання web-орієнтованої системи допомагає автомобілістам слідкувати за своїм автомобілем та гаманцем в цілому, виконувати технічний огляд своєчасно й ділитись своїм досвідом.

Результати роботи над web-орієнтованою системою були апробовані на науково-практичній конференції ІМА 2020 в Сумському державному університеті.

Список використаних джерел

1. Современный учебник JavaScript: [Електронний ресурс]. URL: https://learn.javascript.ru/

2. W3Schools Online Web Tutorials: [Електронний ресурс]. URL: https://www.w3schools.com/

3. HTML Academy: интерактивные онлайн-курсы по HTML, CSS и JavaScript: [Електронний ресурс]. URL: https://htmlacademy.ru/

4. Курс «Основы CSS» — HTML Academy: [Електронний ресурс]. URL: https://htmlacademy.ru/courses/basic-css

5. API Picker | Google Maps Platform | Google Developers: [Електронний ресурс]. URL: https://developers.google.com/maps/documentation/api-picker?hl=ru

6. PHP, MySQL и другие веб-технологии: [Електронний ресурс]. URL: http://www.php.su/

7. Google Fonts: [Електронний ресурс]. URL: https://fonts.google.com/

8. Font Awesome: [Електронний ресурс]. URL: https://fontawesome.com/

9. Bootstrap · The most popular HTML, CSS, and JS library in the world: [Електронний ресурс]. URL: https://getbootstrap.com/

10. Кваліфікаційна робота бакалавра : Зміст: [Електронний ресурс]. URL: https://mix.sumdu.edu.ua/textbooks/12635/index.html

11. jQuery: [Електронний ресурс]. URL: https://jquery.com/

12. Smart WYSIWYG HTML editor | CKEditor 4: [Електронний ресурс]. URL: https://ckeditor.com/ckeditor-4/

Додаток А. Технічне завдання

ТЕХНІЧНЕ ЗАВДАННЯ

на розробку програмного продукту «Web-орієнтована система обліку сервісного обслуговування транспортного засобу»

Суми 2020

1 Призначення й мета створення Web-орієнтованої системи

1.1 Призначення

Web-орієнтована система надає можливість для збереження та фільтрованого відображення інформації з сервісного обслуговування автомобіля.

1.2 Мета створення

Спрощення обслуговування автомобіля та слідкування за його технічним станом.

1.3 Цільова аудиторія

До цільової аудиторії Web-орієнтованої системи відносяться:

- Автомобілісти – зберігають інформацію про сервісне обслуговування свого автомобіля;
- Звичайні користувачі або майбутні автомобілісти – перегляд сервісної історії автомобіля, яка відображується за допомгою фільтрованого пошуку;
- Адміністратор – контролює працездатність сервісу.

2 Вимоги до Web-орієнтованої системи

2.1 Загальні вимоги до додатку

Додаток повинен бути реалізований у вигляді web-сайту. Під час розміщення в мережі Інтернет повинен мати міжрегіональну локацію. Створення додатку повинно складатися із взаємнозалежних модулів з чітко розділеними функціями. Дизайн повинен бути простий і доступний, щоб користувач міг орієнтуватися інтуїтивно, кольорова гамма бути приємною для сприйняття користувачем. Присутня кросбраузерність та кросплатформеність.

2.2 Вимогу до функцій додатку

Однією з головних функції є забезпечення доступу до додавання та редагування інформації. Додавати та редагувати інформацію повинні зареєстровані користувачі, тому повинна бути присутня авторизація та реєстрація.

Для зареєстрованого користувача буде доступна його індивідуальна адмін-панель, за допомогою якої виконується контроль додавання та редагування

інформації. Для ролі адміністратора будуть доступні всі дані користувачів та можливість їх редагувати чи навіть видаляти.

Не зареєстрований користувач має можливість переглядати інформацію за допомогою пошуку, тільки після реєстрації йому будуть доступні функції додавання інформації.

Пошук на сайті повинен бути фільтрований й виконувати фільтрування по марці авто її моделі або просто пошук по ключовим словам.

Користувач має можливість ввести повну характеристику власного автомобіля, включаючи об'єм двигуна, модифікацію, одне фото автомобіля та власний опис авто в довільній формі.

Фіксація події сервісного обслуговування автомобіля включає: тип події, дату події, пробіг автомобіля, ціну виконаних робіт і опис від власника.

2.3 Візуальне представлення додатку

2.3.1 Відображення головної сторінки

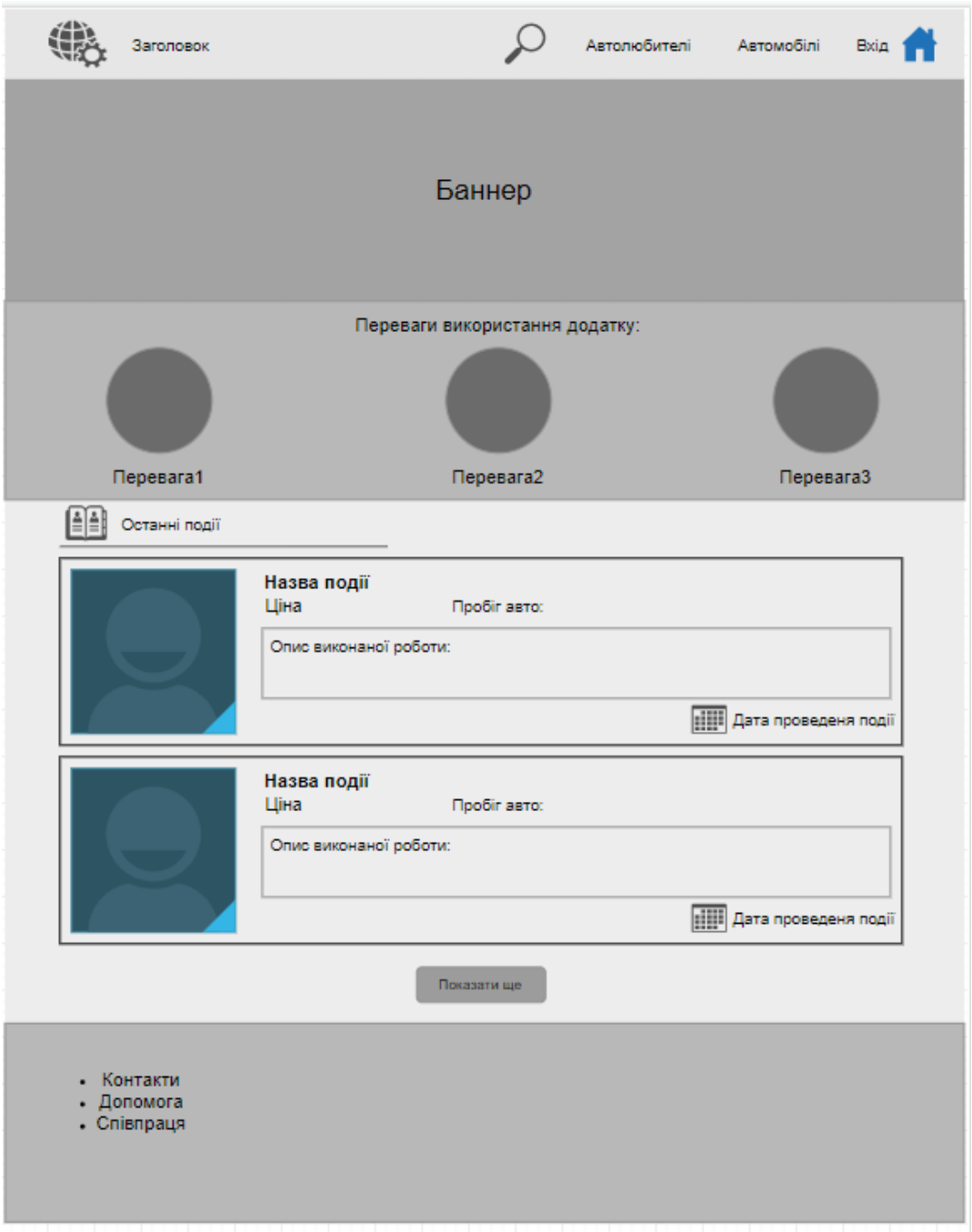

Рисунок 2.3.1.1 – Дизайн головної сторінки

2.3.2 Відображення форми авторизації та реєстрації

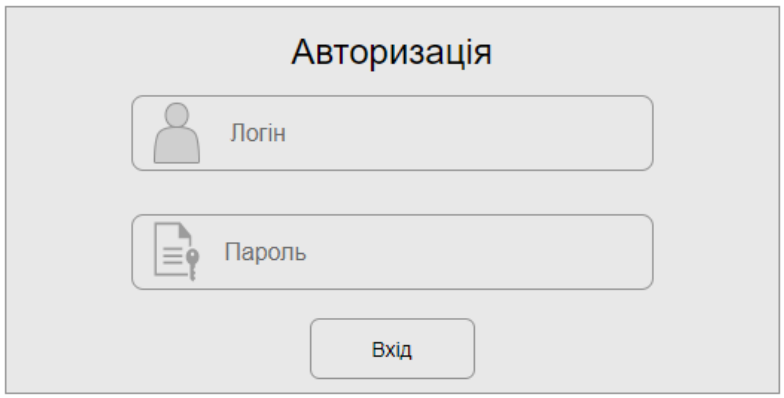

Рисунок 2.3.2.1 – Дизайн форми авторизації

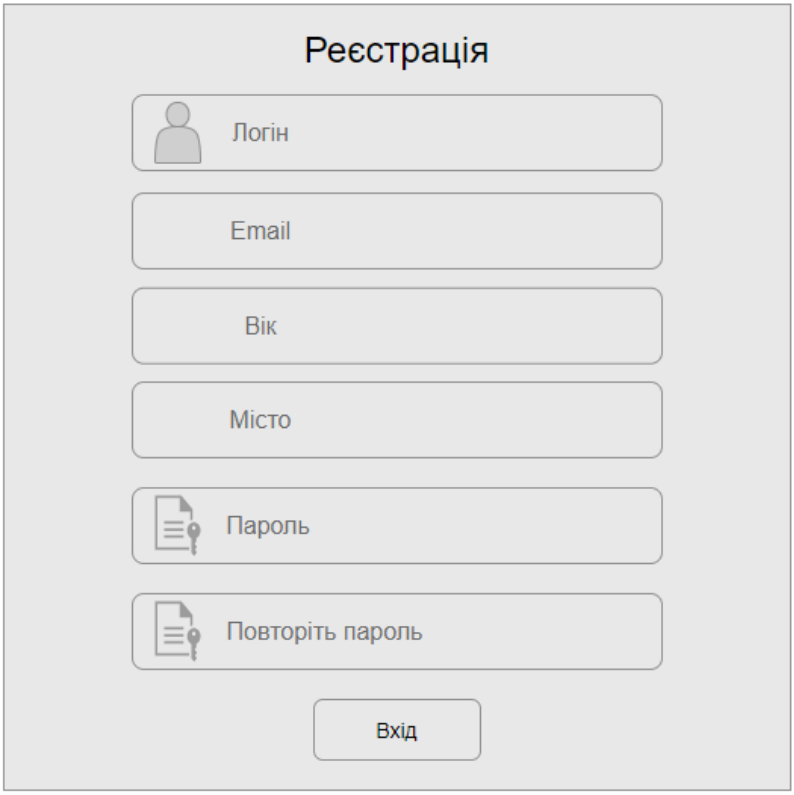

Рисунок 2.3.2.2 – Дизайн форми реєстрації

2.3.3 Відображення адмін-панелі

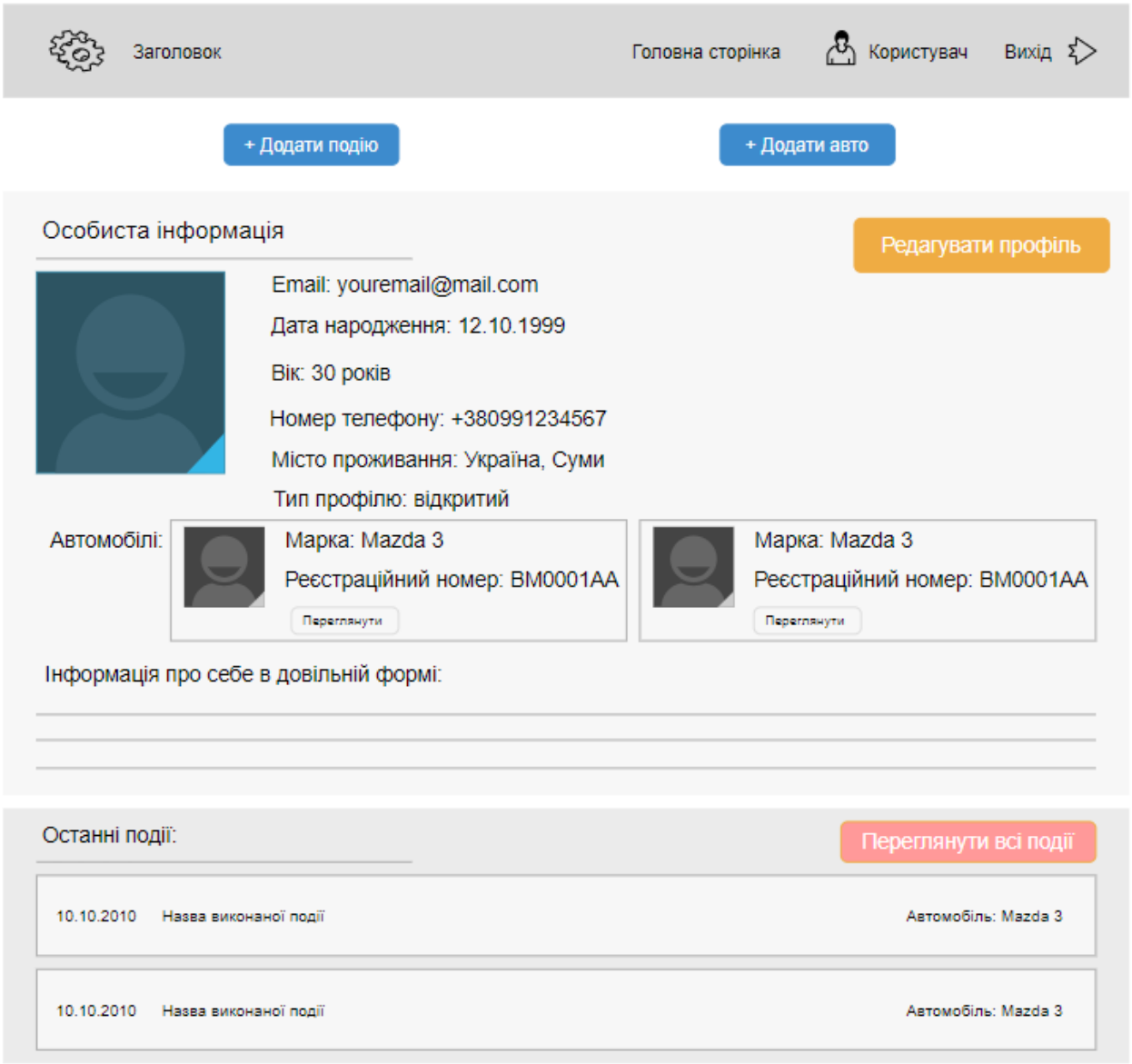

Рисунок 2.3.3.1 – Дизайн адмін-панелі

2.3.4 Відображення форми додавання події сервісного обслуговування

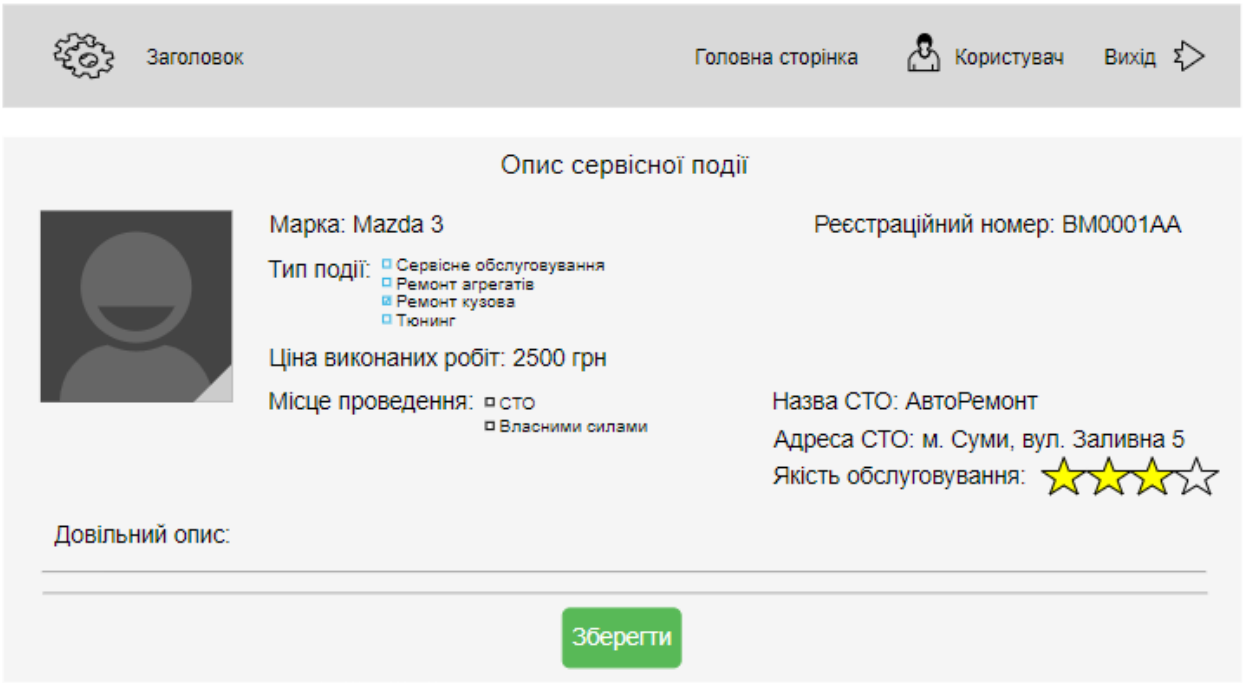

Рисунок 2.3.4.1 – Дизайн форми додавання події сервісного обслуговування

3 Склад і зміст робіт зі створення додатку

Перелік етапів роботи зі створення web-додатку наведені в таблиці 3.1.

Таблиця 3.1 – Етапи створення web-додатку

№	Склад і зміст робіт	Строк розробки
1	Проектування: Проектування структури додатку, структури бази даних.	5 днів
2	Розробка інтерфейсу: Попереднє розташування елементів додатку, розробка всіх вікон.	4 дні
3	Програмування функціональних модулів: • Модуль головної сторінки; • Модуль авторизації; • Модуль реєстрації; • Модуль адмін-панелі; • Модуль додавання автомобіля та події сервісного обслуговування; • Модуль фільтрованого пошуку.	 2 дні 1 день 1 день 4 дні 3 дні 2 дні
4	Тестування кросплатформенності та кросбраузерності.	2 дні
5	Тестування функціональних модулів.	5 днів
	Загальна тривалість робіт	**29 днів**

Додаток Б. Планування робіт

1 Ідентифікація ідеї проекту

Кожен свідомий автомобіліст слідкує за своїм автомобілем, адже з ним від проводить велику кількість часу. Автомобіль повинен працювати в любу годину та пору року, але для цього за ним потрібно слідкувати, виконувати обслуговування.

Автомобіль потребує обов’язкове технічне обслуговування, яке виконується через визначений час чи відповідно до пройденого шляху. Власнику авто досить важко запам’ятовувати всю інформацію й для цього створені сервісні книжки, де все це фіксується.

Web-додаток буде фіксувати сервісну історію автомобіліста, та виконувати її аналіз і фільтрацію. Адже автомобіліст може одночасно володіти декількома автомобілями, тому вести й відображати сервісну історію стане простіше.

Також web-додаток орієнтований на майбутніх автомобілістів, адже вся сервісна історія буде в загальному доступі. Майбутній власник авто зможе переглянути найчастіші поломки певного автомобіля за допомогою фільтрованого пошуку.

2 Деталізація мети методом SMART

S. Розробити WEB-орієнтовану система обліку сервісного обслуговування транспортного засобу.

M. Результатом виконання проекту, має бути Web-додаток, отримати доступ до якого, можна ввівши адресу сайту до адресного рядку, в свою чергу, Web-додаток мусить бути розміщений на хостингу та пов’язані з ним бази даних(БД), форми та модулі мають бути вірно налаштовані та під’єднані.

A. Проект потребує розробника із знанням HTML, CSS, мови програмування JavaScript, PHP, MySQL та об’єктно-орієнтованого програмування і програмного забезпечення (ПЗ), яке використовується для розроблення даного додатку.

R. Мету реально досягнути, так як розробка WEB-додатку за допомогою сучасних Web-технологій, не вимагає надзвичайно складних дій або велику кількість ресурсів.

T. Обмеженість в часі зумовлена рішенням замовника, щоб отримати програмне забезпечення як можна швидше.

3 Опис функціонування продукту

На основі встановленої мети проекту було встановлено рекомендації до функціоналу додатку які мають найбільше значення:

– збереження, фільтрація та аналіз сервісної історії автомобіля;

– відображення фільтрованої інформації для користувача без автомобіля;

– постійний та швидкий доступ до додатку, без необхідності встановлення додаткового ПЗ;

– постійний та швидкий доступ до додатку, незалежно від типу пристрою та операційної системи користувача.

4 Опис IT-проекту на фазі розробки

4.1 Планування змісту робіт

Структурна декомпозиція робіт (work breakdown structure, WBS) - це ієрархічна структура робіт, побудована з метою логічного розподілу усіх робіт з виконання проекту і подана у графічному вигляді. Це сукупність декількох рівнів, кожний з яких формується в результаті розподілу роботи попереднього рівня на її складові. Елементом найнижчого рівня є група робіт, або так званий робочий пакет (work package).

WBS представлена на рисунку 4.1.

Рисунок 4.1 – WBS структура проекту

4.2 Планування структури виконавців

Наступним кроком розробки структури проекту є визначення організаційної структури (OBS) проекту.

Організаційна структура проекту (OBS) – є графічним відображенням учасників проекту (фізичних та юридичних осіб) та їхніх відповідальних осіб, залучених до реалізації проекту. На верхньому рівні OBS проекту знаходиться керівник та команда управління проектом; на наступному рівні – виконавці. Останнім рівнем OBS-структури є відповідальні особи виконавців. Це не обов'язково повинні бути керівники, а ті співробітники, яким доручено безпосередньо організовувати і відповідати перед виконавцем за виконання конкретного елемента WBS-структури.

OBS структура представлена на рисунку 4.2.

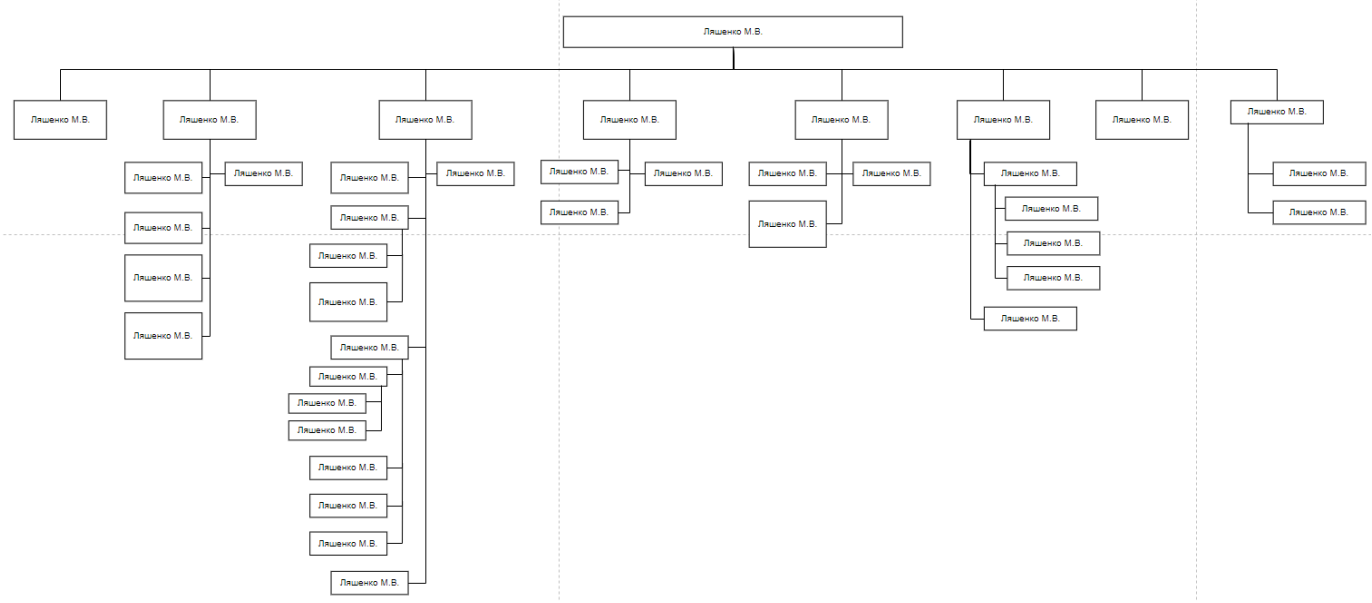

Рисунок 4.2 – OBS структура

4.3 Побудова матриці відповідальності

Матриця відповідальності (Responsibility Assignment Matrix) забезпечує опис і узгодження структури відповідальності за виконання пакетів робіт. Вона являє собою форму опису розподілу відповідальності за реалізацію робіт проекту із зазначенням ролі кожного з виконавців. Будується на основі WBS та OBS.

У таблиці 4.1 показано матрицю відповідальності проекту.

Таблиця 4.1 – RAM

№	Duty	Ляшенко Максим	Марченко Анна	Шендрик Віра	Парфененко Юлія
1	Постановка вимог до проекту	A	R	A	A
2	Пошук по марці авто	R			
3	Пошук по марці та моделі авто	R			
4	Пошук відповідно року виробництва авто	R			
5	Пошук відповідно комплектуючих авто	R			
7	Тестування модуля для фільтрованого пошуку	R			
8	Особиста інформація автолюбителя	R			
9	Вибір марки та моделі авто	R			
10	Ходові характеристики автомобіля	R			
11	Адреса місця ремонту	R			

Продовження таблиці 4.1

№	Duty	Ляшенко Максим	Марченко Анна	Шендрик Віра	Парфененко Юлія
12	Відгук про місце ремонту	R	I		
13	Дата ремонту	R	I		
14	Що було відремонтовано	R	I		
15	Ціна ремонту	R	I		
16	Статистичні дані ремонту авто	R	I		
17	Тестування модуля адмін-панелі	R	I		
18	Валідація введених даних при реєстрації	R	I		
19	Підтвердження email	R	I		
20	Тестування модуля реєстрації	R	I		
21	Валідація введених даних при авторизації	R	I		
22	Створення сессії авторизованого користувача	R	I		
23	Тестування модуля авторизації	R	I		
24	Підбір кольорової гамми	R	I		

Продовження таблиці 4.1

№	Duty	Ляшенко Максим	Марченко Анна	Шендрик Віра	Парфененко Юлія
25	Підбір шрифтів від google fonts	R	I		
26	Підбір іконок від font awesome	R	I		
27	Реалізація дизайну	R	I		
28	Розробка інструкції з користування	R	I		
29	Системне тестування	R	I		
30	Юзабіліті тестування	R	I		
31	Запуск робочої версії	R	I	A	A
32	Перевірка працездатності системи	I	I	R	A
33	Перевірка документації проекту	I	A	A	R

4.4 Побудова календарного графіку

Діаграма Ганта - це популярний вид діаграми (придуманий Генрі Гант), який використовується для планування і контролю виконання проекту. Такий інтерактивний мережевий графік присутній практично у всіх системах управління проектами.

На діаграмі відображаються завдання і стадії проекту з урахуванням їх часу виконання. Завдання на діаграмі можуть бути залежними один від одного (наприклад, одна задача може починатися тільки після завершення другого). Крім того, може показуватися відсоток виконання кожного завдання і відповідальний за її виконання.

Діаграма Ганта представлена на рисунку 4.3

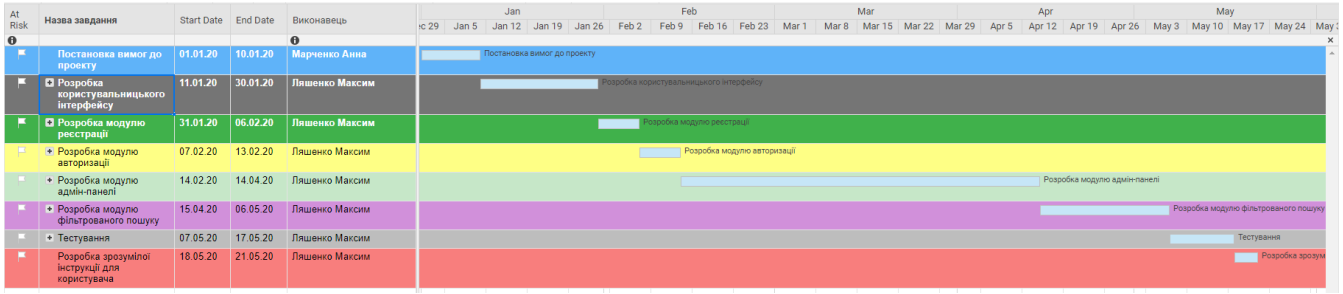

Рисунок 4.3 – Розроблена Gantt Chart

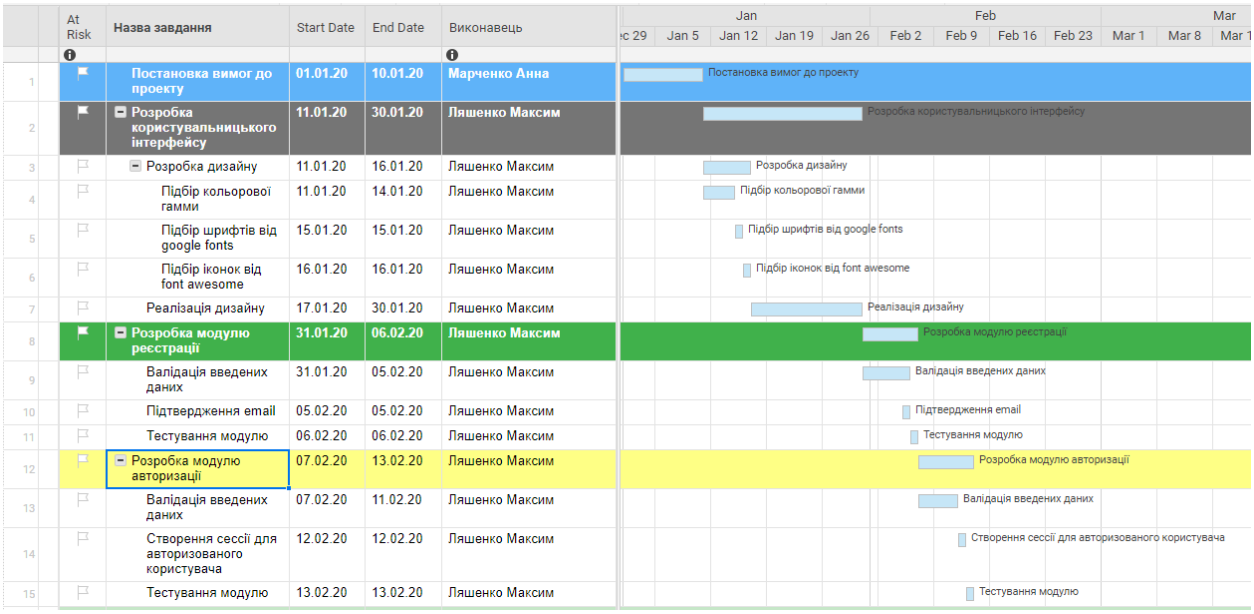

Рисунок 4.4 – Детальне відображення Gantt Chart

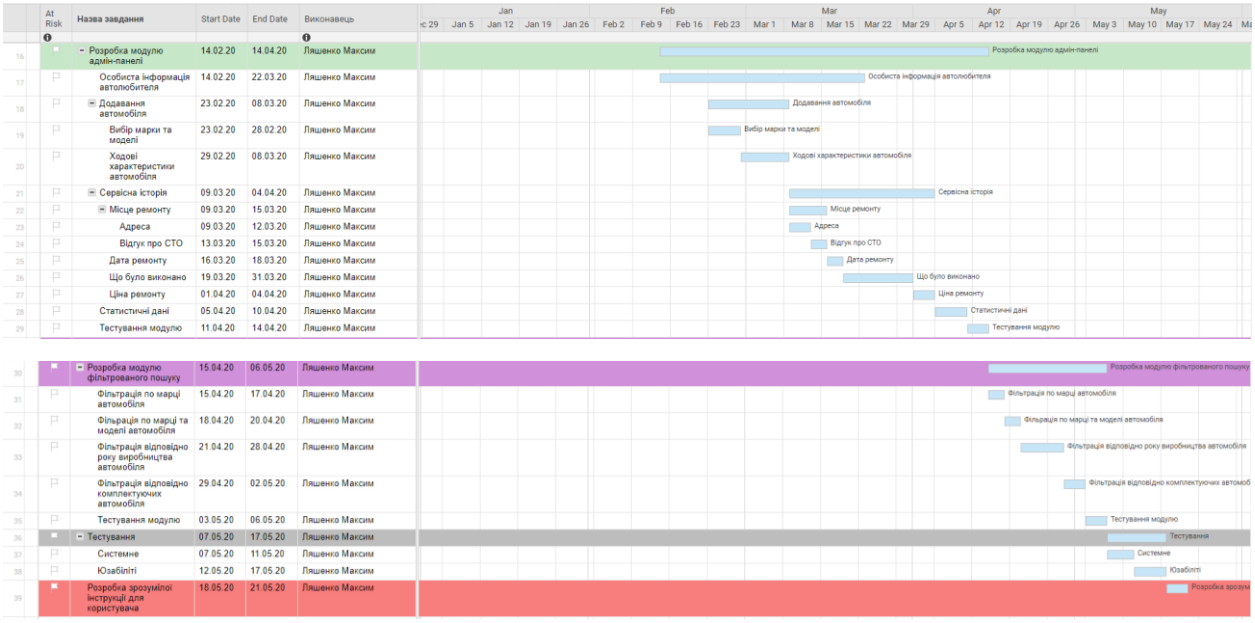

Рисунок 4.5 – Продовження до рисунку 4.4

4.5 Управління ризиками проекту

Ризик – це імовірнісна подія, яка у випадку своєї появи негативно або позитивно вплине на проект.

Процес управління ризиками включає наступні етапи:

1. ідентифікація
2. процес оцінювання ризиків, який включає в себе якісний, кількісний аналіз.
3. заходи реагування на ризики
4. моніторинг заходів і ризиків

Ризики проекту представлені у таблиці 4.2

Таблиця 4.2 – Risk Register

№	Risk description	Impact	Probability	RV	Mitigation
1	Project is not well-defined	4	2	M	Complete a business case if not already provided and ensure purpose is well defined on Project Charter and PID.
2	Unplanned work to be performed at a late stage of the project	4	3	M	1 Document all assumptions made in planning and communicate to the project manager before project kick off
3	Pressure to arbitrarily reduce task durations and or run tasks in parallel which would increase risk of errors.	3	1	L	Show project plan to interested parties and patiently explain that a reduction in lead time will lead to unavoidable errors and poor product quality.
4	Low customer involvement	2	1	L	Explain to the customer that the higher his involvement in the project, the better product he will receive.
5	Difficult or impossible tasks	5	3	H	Upgrade staff or replace / exclude a task

Рівні критичності:

1 Negligible;
2 Low;
3 Medium;
4 High;
5 Critical.

Таблиця 4.3 – Probability / Impact Matrix

5					
4					
3				2	5
2		4		1	
1			3		
Probability / Impact	1	2	3	4	5

Додаток В. Апробація

ІМА :: 2020

СЕКЦІЯ 2: Інформаційні технології проектування

WEB-орієнтована система обліку сервісного обслуговування транспортного засобу

Ляшенко М.В., *студент*; Марченко А.В., *доцент*
Сумський державний університет, м. Суми

З кожним роком, розвиток технологій вражає, збільшується кількість сервісів та можливостей для зберігання та структуризації інформації.

Остання в свою чергу стає більш доступною за рахунок зменшення кількості часу для пошуку та відображення.

Актуальною є розробка web-орієнтованої системи для сервісного обслуговування транспортного засобу, доступ до додатку може здійснюватися за допомогою будь-якого пристрою, який має доступ до мережі Інтернет.

Для досягнення мети проекту, розроблення web-орієнтованої системи, були реалізовані такі задачі: аналіз вимог до web-системи, проектування архітектури системи та бази даних, розробка інтерфейсу та реалізація бази даних, написання програмної частини і повне тестування створеної web-орієнтованої системи.

Базу даних було вирішено реалізувати за допомогою MySQL – це вільна система керування реляційними базами даних та компактний багатопотоковий сервер баз даних.

Володіє високою швидкістю, стійкістю і простотою використання. Підходить як для малих так і середніх розмірів проектів.

Для розробки web-додатку, необхідні технології розробки адаптивного та сучасного, швидкого та надійного додатку, який буде працювати без відмов.

Розробка інтерфейсу та програмної частини здійснена за стандартними технологіями HTML, CSS, JavaScript та PHP.

Розроблена web-система орієнтована на розміщення історії обслуговування транспортного засобу у вільному доступі, що дозволить швидко ознайомитись із поточним технічним станом засобу, допомагає слідкувати за станом автомобіля поточному авто автовласнику, і в свою чергу майбутнім автовласникам, які тільки оглядають ринок.

Додаток Г. Лістинг програмного коду

SQL запит на створення бази даних:

```
CREATE TABLE `cars` (
  `car_id` int(4) NOT NULL,
  `user_id` int(4) NOT NULL,
  `brand_id` int(4) NOT NULL,
  `model_id` int(4) NOT NULL,
  `year_of_isue` int(4) NOT NULL,
  `year_of_purchase` int(4) NOT NULL,
  `car_image` varchar(100) NOT NULL,
  `description` text NOT NULL,
  `engine_type` varchar(25) NOT NULL,
  `engine_volume` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `type_of_transmission` varchar(25) NOT NULL,
  `engine_power` int(5) NOT NULL,
  `type_of_drive` varchar(20) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `car_name` varchar(255) NOT NULL,
  `add_date` varchar(30) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `car_brand` (
  `brand_id` int(4) NOT NULL,
  `brand` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `car_model` (
  `model_id` int(4) NOT NULL,
  `brand_id` int(4) NOT NULL,
  `model` varchar(30) NOT NULL,
  `start_production` int(4) NOT NULL,
  `end_production` int(4) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `notes` (
  `note_id` int(4) NOT NULL,
  `car_id` int(4) NOT NULL,
  `title` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `description` text NOT NULL,
  `price` int(10) NOT NULL,
  `date` varchar(50) NOT NULL,
  `operation_type` varchar(60) NOT NULL,
  `km_count` int(11) NOT NULL,
  `location_name` varchar(255) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL,
  `lat_lng` varchar(255) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `users` (
  `user_id` int(4) NOT NULL,
```

```
  `first_name` varchar(100) DEFAULT NULL,
  `last_name` varchar(100) DEFAULT NULL,
  `gender` varchar(100) DEFAULT NULL,
  `user_image` varchar(100) DEFAULT NULL,
  `description` text,
  `driving_experience` varchar(2) DEFAULT NULL,
  `email` varchar(100) NOT NULL,
  `password` varchar(100) NOT NULL,
  `level` int(1) NOT NULL,
  `date` varchar(30) DEFAULT NULL,
  `location` varchar(100) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

ALTER TABLE `cars`
  ADD PRIMARY KEY (`car_id`),
  ADD KEY `car_user_id_fk` (`user_id`),
  ADD KEY `car_brand_id_fk` (`brand_id`),
  ADD KEY `car_model_id_fk` (`model_id`);

ALTER TABLE `car_brand`
  ADD PRIMARY KEY (`brand_id`);

ALTER TABLE `car_model`
  ADD PRIMARY KEY (`model_id`),
  ADD KEY `car_model_brand_id_fk` (`brand_id`);

ALTER TABLE `notes`
  ADD PRIMARY KEY (`note_id`),
  ADD KEY `note_car_id_fk` (`car_id`);

ALTER TABLE `users`
  ADD PRIMARY KEY (`user_id`);

ALTER TABLE `cars`
  MODIFY `car_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=17;

ALTER TABLE `car_brand`
  MODIFY `brand_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=83;

ALTER TABLE `car_model`
  MODIFY `model_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=83;

ALTER TABLE `notes`
  MODIFY `note_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;

ALTER TABLE `users`
  MODIFY `user_id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=9;

ALTER TABLE `cars`
  ADD CONSTRAINT `car_brand_id_fk` FOREIGN KEY (`brand_id`) REFERENCES
`car_brand` (`brand_id`),
```

```
        ADD CONSTRAINT `car_model_id_fk` FOREIGN KEY (`model_id`) REFERENCES `car_model` (`model_id`),
        ADD CONSTRAINT `car_user_id_fk` FOREIGN KEY (`user_id`) REFERENCES `users` (`user_id`);

        ALTER TABLE `car_model`
        ADD CONSTRAINT `car_model_brand_id_fk` FOREIGN KEY (`brand_id`) REFERENCES `car_brand` (`brand_id`);

        ALTER TABLE `notes`
        ADD CONSTRAINT `note_car_id_fk` FOREIGN KEY (`car_id`) REFERENCES `cars` (`car_id`);
        COMMIT;
```

Лістинг коду головної сторінки – index.php:

```
        <?php
        session_start();
        require 'scripts/connect.php';
        if (isset($_GET["is_exit"])) {
            if ($_GET["is_exit"] == 1) {
                unset($_SESSION['user_name']);
                session_destroy();
                header("Location: ?is_exit=0");
            }
        }?>
        <!doctype php>
        <html land="ru">
            <head>
                <meta charset="utf-8">
                <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
                <title></title>
                <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
                <link rel="stylesheet" href="css/style.css" >
                <link rel="stylesheet" type="text/css" href="http://fonts.googleapis.com/css?family=Yellowtail%7COpen%20Sans%3A400%2C300%2C600%2C700%2C800" media="screen" />
                <script src="https://kit.fontawesome.com/361785f0e7.js" crossorigin="anonymous"></script>
            </head>
            <body>
            <header>
            <nav id="top-nav" class="navbar fixed-top navbar-expand-lg navbar-dark bg-dark">
                <div class="container">
                 <a class="top-logo navbar-brand" href="index.php">
                    <img src="images/auto-trade-logo-retina.png" alt="">
                    <span>AutoCommunity</span>
                 </a>
```

```
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNavDropdown" aria-controls="navbarNavDropdown" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
      <div class="collapse navbar-collapse" id="navbarNavDropdown">
        <ul class="nav d-flex justify-content-end text-uppercase">
          <li class="nav-item">
            <a class="nav-link" href="auto.php"><span>Автомобілі</span></a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="driver.php"><span>Автомобілісти</span></a>
          </li>
           <?php
               if (isset($_SESSION['user_name'])) {
                    echo '
           <li class="nav-item">
            <a class="nav-link" href="admin-panel.php?user_id='.$_SESSION['user_id'].'"><span>Адмін-панель</span></a>
           </li>
           <li class="nav-item">
            <a class="nav-link" href="?is_exit=1"><span>Вихід</span> <i class="fas fa-sign-out-alt"></i></a>
           </li>
                    ';
          }else {
           echo '
           <li class="nav-item">
            <a class="nav-link" href="login.php"><span>Вхід</span> <i class="fas fa-sign-in-alt"></i></a>
          </li>';
          }
           ?>
        </ul>
          </div>
          </div>
      </nav>
          </header>
          <section id="top-image">
            <img src="images/banner.jpg" alt="...">
          </section>
          <section id="message-wrap">
            <div class="container">
              <div class="row">
                <div class="col-5"> <a type="button" href="register.php" class="btn btn-add-user">Створити сторінку <i class="fas fa-user-plus"></i></a> </div>
                <div class="col-7"><h2 >Приєднуйся до коллективу <span class="alternate-font">AutoCommunity</span></h2>
                </div>
              </div>
              </div>
```

```
    <div class="message-shadow"></div>
  </section>
  <section id="description">
    <h2 class="text-center">Короткий опис сервісу:</h2>
    <hr>
    <div class="container">
    <div class="media position-relative">
     <div class="row">
      <div class="col-md-4">
        <img src="images/about1.jpg" class="info-image" class="mr-3" alt="...">
      </div>
      <div class="col-md-8 d-flex">
      <div class="media-body media-middle">
        <h5 class="mt-0">Контроль технічного стану автомобіля та грошових витрат</h5>
        <p>Функціонал сервісу дозволяє додавати та зберігати інформацію про сервісне обслуговування автомобіля, де також зберігається дата та час, місце проведення, комплектуючі та тип виконаних робіт й скільки коштувала дата процедура.</p>
      </div>
      </div>
     </div>
    </div>
    <div class="media">
     <div class="row">
      <div class="col-md-8 d-flex">
      <div class="media-body media-middle">
        <h5 class="mt-0 mb-1">Статистичні дані як користувача так і марки в цілому</h5>
          <p>Для моніторингу витрат та контролю технічного стану авто, сервіс аналізує всі виконані дії, їх дату виконання та значення на одометрі авто та звичайно ціну виконання. Відображає дані за допомогою діаграми або просто таблиці.</p>
      </div>
      </div>
      <div class="col-md-4">
        <img src="images/about2.jpg" class="info-image float-right" class="ml-3" alt="...">
      </div>
    </div>
    </div>
  </section>
  <section id="advantages">
    <h2 class="text-center">Переваги сервісу:</h2>
    <hr>
    <div class="container">
    <div class="card-deck">
     <div class="card">
       <span>
          <i class="far fa-save"></i>
       </span>
       <div class="card-body">
         <h5 class="card-title">Збереження</h5>
         <p class="card-text text-center">Забезпечує надійне збереження інформації.</p>
       </div>
     </div>
```

```
                <div class="card">
                  <span>
                    <i class="fas fa-map-marked-alt"></i>
                  </span>
                  <div class="card-body">
                    <h5 class="card-title">Доступність</h5>
                    <p class="card-text text-center">Сайт доступний з любого куточку світу.</p>
                  </div>
                </div>
                <div class="card">
                  <span>
                    <i class="fas fa-chart-bar"></i>
                  </span>
                  <div class="card-body">
                    <h5 class="card-title">Статистика</h5>
                    <p class="card-text text-center">Статистичні дані затрат та проведених робіт.</p>
                  </div>
                </div>
              </div>
              </div>
            </section>
            <section id="event">
              <h2 class="text-center">Останні події:</h2>
              <hr class="line">
              <div class="container-fluide">
                <div class="row no-gutters justify-content-center">
                  <?php
                  $sql_note = mysql_query("SELECT `user_image`, `note_id`, `title`, `user_id`, `first_name`, `last_name`, `price`, `car_id`, `brand`, `model`, `year_of_isue`, notes.date as `note_date` FROM `notes` join `cars` using(`car_id`) join `users` USING(`user_id`) join `car_model` using(`model_id`) join `car_brand` on(cars.`brand_id`=car_brand.`brand_id`) Order by `note_id` DESC Limit 10");
                    while ($result_note = mysql_fetch_array($sql_note)) {
                      echo '
                      <div class="col-xl-8 col-lg-10 col-md-10 col-sm-11">
                      <ul class="list-unstyled">
                        <li class="media">
                          <img src="images/'.$result_note["user_image"].'" href="user-profile.php?user_id='.$result_note["user_id"].'" class="mr-3" alt="...">
                          <div class="media-body">
                            <h5 class="mt-0 mb-1"><i class="fas fa-heading"></i> <a href="some-note.php?note_id='.$result_note["note_id"].'">'.$result_note["title"].'</a></h5>
                            <hr/>
                            <div class="d-flex justify-content-between">
                              <p><i class="fas fa-user"></i> <a href="user-profile.php?user_id='.$result_note["user_id"].'">'.$result_note["first_name"].'
                              '.$result_note["last_name"].'</a> </p>
                              <p><i class="fas fa-money-bill-wave"></i>  '.$result_note["price"].' грн</p>
```

```
                                        <p><i class="fas fa-car"></i> <a href="some-car.php?car_id='.$result_note["car_id"].'">'.$result_note["brand"].' '.$result_note["model"].' '.$result_note["year_of_isue"].'</a></p>
                                        <p><i class="far fa-calendar-alt"></i> '.$result_note["note_date"].'</p>
                                    </div>
                                </div>
                            </li>
                        </ul>
                        </div>';
                    }
                ?>
            </div>
        </div>
    </section>
<?php include ("templates/footer.php");?>
```

Лістинг коду головної сторінки адміністративної панелі – admin-panel.php:

```
<?php
session_start();
require 'scripts/connect.php';
if (isset($_GET["is_exit"])) {
    if ($_GET["is_exit"] == 1) {
        unset($_SESSION['user_name']);
        session_destroy();
        header("Location: ?is_exit=0");
    }
}?>
<!doctype html>
<html land="ru">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
        <title>Адміністративна панель</title>
        <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">
        <link rel="stylesheet" href="css/style.css" >
        <link rel="stylesheet" type="text/css" href="http://fonts.googleapis.com/css?family=Yellowtail%7COpen%20Sans%3A400%2C300%2C600%2C700%2C800" media="screen" />
        <?php
        if (!isset($_SESSION['user_name']) || $_SESSION['user_id'] !=$_GET['user_id']) {
            echo '<meta http-equiv="Refresh" content="0; URL=index.php">';
        }?>
        <script src="https://kit.fontawesome.com/361785f0e7.js" crossorigin="anonymous"></script>
    </head>
    <body class="bg-grey">
```

```
<section id="resize-sec">
<?php include ("templates/admin-nav.php");?>
  <section id="top-image">
    <div class="card bg-dark text-white">
      <img src="images/dynamic-header-13.jpg" class="card-img" alt="...">
      <div class="own-card-img-overlay container">
        <nav aria-label="breadcrumb">
          <ol class="breadcrumb">
            <li class="breadcrumb-item"><a href="index.php"><i class="fas fa-home"></i> Головна</a></li>
            <li class="breadcrumb-item active">Панель адміністратора</li>
          </ol>
        </nav>
      </div>
    </div>
  </section>
  <section id="admin-container">
    <div class="container">
      <div class="row">
        <?php include ("templates/left-side-bar.php");?>
        <div class="col-md-8">
          <div class="accordion" id="accordionExample1">
            <div class="card">
              <div class="card-header" id="headingOne">
                <h2 class="mb-0">
                  <button class="btn btn-link" type="button" data-toggle="collapse" data-target="#collapseOne" aria-expanded="true" aria-controls="collapseOne">
                    Гараж
                  </button>
                  <i class="fas fa-chevron-down arrow-accordion"></i>
                </h2>
              </div>

              <div id="collapseOne" class="collapse show" aria-labelledby="headingOne" data-parent="#accordionExample1">
                <?php
                  $sql = mysql_query("SELECT * FROM `cars` join `car_brand` using(`brand_id`) join `car_model` using(`model_id`) WHERE `user_id` = ".$_SESSION['user_id'].";");
                  while ($result = mysql_fetch_array($sql)) {
                    echo '
                      <div class="card-body">
                        <div class="card mb-3 auto-card">
                          <figure>
                            <a href="own-car.php?user_id='.$_SESSION['user_id'].'&&car_id='.$result["car_id"].'"><img src="images/'.$result["car_image"].'" class="card-img-top" alt="..."></a>
                          </figure>
                          <div class="card-body">
                            <div class="d-flex justify-content-between">
```

```
                                        <h5 class="card-title"><a href="own-car.php?user_id='.$_SESSION['user_id'].'&&car_id='.$result["car_id"].'"> '.$result["brand"].' '. $result["model"].' '. $result["year_of_isue"].' </a></h5>
                                        <a type="button" href="add-note.php?user_id='.$_SESSION['user_id'].'&&car_id='.$result["car_id"].'" class="btn btn-outline-success"><i class="fas fa-plus"></i> Додати подію</a>
                                        </div>
                                      </div>
                                    </div>
                                  </div>';
                                }
                              ?>
                            </div>
                          </div>
                        </div>
                        <div class="accordion" id="accordionExample2">
                          <div class="card">
                            <div class="card-header" id="headingTwo">
                              <h2 class="mb-0">
                                <button class="btn btn-link" type="button" data-toggle="collapse" data-target="#collapseTwo" aria-expanded="true" aria-controls="collapseTwo">
                                  Бортовий журнал
                                </button>
                                <i class="fas fa-chevron-down arrow-accordion"></i>
                              </h2>
                            </div>

                            <div id="collapseTwo" class="collapse show" aria-labelledby="headingTwo" data-parent="#accordionExample2">
                              <?php
                                $sql_note = mysql_query("SELECT * FROM `notes` join `cars` using(`car_id`) join `car_brand` using(`brand_id`) join `car_model` using(`model_id`) WHERE `user_id` =".$_SESSION['user_id'].";");
                                while ($result_note = mysql_fetch_array($sql_note)) {
                                  echo '
                                    <div class="card-body">
                                      <div class="row admin-note-list">
                                        <div class="col-md-9 col-9">
                                          <h5 class="card-title"><a href="own-note.php?user_id='.$_SESSION['user_id'].'&&note_id='.$result_note["note_id"].'">'.$result_note["title"].'</a></h5>
                                          <small>'.$result_note["brand"].' '. $result_note["model"].' '. $result_note["year_of_isue"].' </small>
                                        </div>
                                        <div class="col-md-3 col-3">
                                        <p class="float-right note-date">'.$result_note["date"].'</p>
                                        </div>
                                      </div>
                                    </div>';
                                  }
                              ?>
```

```
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </section>
    </section>
    <?php include ("templates/footer.php");?>
```