Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

РОЗРОБКА ТА МОДЕЛЮВАННЯ ВЕБСАЙТУ З ПРОДАЖУ ЕЛЕКТРОМОБІЛІВ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-2
Мітьков М.Є.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник доц. Шафроненко А.Ю.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Мітькову Максиму Євгеновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та моделювання вебсайту з продажу електромобілів

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії 24 грудня 2023 р.

3. Вихідні дані до роботи актуальні методи розробки вебзастосунків, перелік використовуваних програмних засобів: теоретичні відомості про методи моделювання вебсайтів.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів розробки вебсайтів.
2. Розробка дизайну, моделювання бази даних та структури вебзастосунку.
3. Програмна реалізація вебсайту з продажу електромобілів.
4. Аналіз результатів роботи та перспектив подальшого розвитку проєкту.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми розробки вебсайтів, постановка задачі, тестові зображення, таблиці баз даних, зображення структури дизайну проєкту, опис перспектив розвитку проєкту.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	04.11.23-05.11.23	
3	Аналіз літератури з досліджуваної проблеми	06.11.23-08.11.23	
4	Аналіз методів розробки вебсайтів	09.11.23-10.11.23	
5	Розробка вебзастосунку	12.12.23-16.12.23	
6	Програмна реалізація	17.12.23-19.12.23	
7	Оформлення пояснювальної записки	20.12.23-21.12.23	
8	Перевірка на плагіат	22.12.2023	
9	Рецензування	23.12.2023	
10	Підготовка презентації та доповіді	25.12.2023	
11	Занесення роботи в електронний архів	27.12.2023	
12	Попередній захист кваліфікаційної роботи	04.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ доц. Шафроненко А. Ю.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 52 с., 4 табл., 23 рис., 50 джерел.

КЛІЄНТ-СЕРВЕРНИЙ ЗАСТОСУНОК, ВЕБСАЙТ, ЕЛЕКТРОМОБІЛІ, SPRING FRAMEWORK, BOOTSTRAP, VISUAL STUDIO CODE, POSTGRESQL, МОВА ЗАПИТІВ SQL, HTML, CSS, JAVASCRIPT.

Об’єктом дослідження є вебсайт з продажу електромобілів.

Метою дослідження є реалізація клієнт-серверного застосунку для огляду та продажу електромобілів.

Для розробки Front-end частини проєкту використовувались HTML, CSS, JavaScript та фреймворк BootStrap. Для розробки Back-end частини проєкту використано мову програмування Java та такі вебтехнології як Java EE та Spring Framework, а саме Spring Core, Spring MVC, Spring JDBC, Spring Security. У якості СУБД було використано PostgreSQL. Застосунок має 3 рівня доступу: незареєстрований користувач, зареєстрований користувач та адміністратор.

У результаті дослідження здійснена програмна реалізація системи інтернет-магазину.

CLIENT SERVER APPLICATION, WEB-SITE, ECOLOGICAL PROBLEMS, JAVA, SPRING FRAMEWORK, BOOTSTRAP, VISUAL STUDIO CODE, POSTGRESQL, SQL, HTML, CSS, JAVASCRIPT.

The object of the research is a website for the sale of electric cars.

The purpose of the research is the implementation of a client-server application for the review and sale of electric cars.

HTML, CSS, JavaScript and the BootStrap framework were used to develop the Front-end part of the project. The Java programming language and web technologies such as Java EE and Spring Framework, namely Spring Core, Spring MVC, Spring JDBC, Spring Security, were used to develop the Back-end part of the project. PostgreSQL was used as a database. The application has 3 levels of access: unregistered user, registered user and administrator.

As a result, the software implementation of the online store system was implemented.

ВСТУП

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

WEB – World Wide Web (всесвітня павутина)

SQL – Structured Query Language (мова структурованих запитів)

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту)

HTML – HyperText Markup Language (мова гіпертекстової розмітки)

URI – Uniform Resource Identifier (універсальний ідентифікатор ресурсу)

UML – Unified Modeling Language (уніфікована мова моделювання)

IDE – Integrated Development Environment (інтегроване середовище розробки)

JDBC – Java DataBase Connectivity (з'єднання з базами даних на Java)

JTA – Java Transaction API (API для підтримки транзакцій, що входить до стандарту серверної платформи для мови програмування Java - Jakarta EE)

JDO – Java Data Objects (об'єкти даних Java)

POJO – Plain Old Java Object («старий добрий Java-об'єкт», простий Java-об'єкт, не успадкований від якогось специфічного об'єкта і не реалізує ніяких службових інтерфейсів понад ті, які потрібні для бізнес-моделі)

ВСТУП

Сучасний світ не можна уявити без сайтів, оскільки вони стали важливою складовою навчання, роботи та розваг для багатьох людей. Сайти є джерелом корисної інформації, місцем для розважання та спілкування в соціальних мережах, а також джерелом розважального контенту, такого як фільми, серіали та музика. Хоча сайти стали невід'ємною частиною нашого життя, варто відзначити, що їх поява в історії відбулася досить недавно, в 1990 році. Перші сайти були простими сторінками з гіперпосиланнями на інші сторінки, але з часом вони стали невід'ємною частиною нашої повсякденності, де ми проводимо багато часу.

У даній кваліфікаційній роботі мова йде про моделювання та розробку клієнт-серверного вебзастосунку, який являє собою інтернет-магазин з продажу електромобілів.

Метою дослідження є реалізація клієнт-серверного застосунку для огляду та продажу електромобілів.

Актуальність теми полягає в тому, що зараз електромобілі здобувають все більшої популярності завдяки своїй екологічності та технологічності, тому було вирішено провести інформаційний огляд електромобілів та розробити сервіс з їх продажу.

1 ТЕОРЕТИЧНИЙ ОГЛЯД РОЗРОБКИ ВЕБСАЙТІВ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Історія вебсайтів

Перший в історії сайт побачив світ 6 серпня 1991 року info.cern.ch. Ініціатором його створення був Тім Бернерс-Лі, який розмістив на цьому вебсайті опис нової технології під назвою World Wide Web. Зазначена технологія базується на використанні протоколу передачі даних HTTP, системи адресації URI та мови гіпертекстової розмітки HTML.

На сайті info.cern.ch були опубліковані важливі відомості про принципи функціонування серверів і браузерів, а також принципи роботи World Wide Web. Цей вебсайт став першим в світі інтернет-каталогом завдяки доданню на нього списку посилань на інші вебсайти.

Слід зазначити, що всі необхідні інструменти для створення першого вебсайту були підготовлені Тімом Бернерсом-Лі ще наприкінці 1990 року. Ці інструменти включали в себе перший гіпертекстовий браузер під назвою WorldWideWeb, який об’єднував у собі функціонал вебредактора, а також перший вебсервер, побудований на базі комп’ютера NeXTcube, та перші вебсторінки [1].

Тім Бернерс-Лі, вважається «батьком» вебу, і він втілив свою ідею в життя, переконаний, що гіпертекст може бути основою для обміну даними через мережу. Вже у 1980 році він розробив гіпертекстове програмне забезпечення Enquire, яке використовувало випадкові асоціації для зберігання даних. Пізніше, працюючи в Європейському центрі ядерних досліджень в Женеві (CERN), він пропонував колегам публікувати гіпертекстові документи з гіперпосиланнями між ними. Бернерс-Лі продемонстрував можливості гіпертекстового доступу до внутрішніх

документів пошукача та новинних ресурсів Інтернету. Ця робота призвела до затвердження стандарту WWW в CERN в травні 1991 року [2].

Тім Бернерс-Лі є ключовою постаттю, яка стояла за основними технологіями вебу, такими як HTTP, URI/URL та HTML, хоча теоретичні основи цих ідей були викладені раніше. У 1940-х роках Венівар Буш пропонував ідеї розширення пам’яті людини за допомогою технічних засобів та індексації накопиченої інформації для швидкого пошуку. Теодор Нельсон та Даг Енгельбарт розробили технологію гіпертексту, яка надавала читачеві можливість різних варіантів читання. Проєкт Xanadu, незавершена гіпертекстова система Нельсона, була спроєктована для зберігання та пошуку текстів із взаємозв’язками та «вікнами». Нельсон мріяв про з’єднання всіх текстів, створених людством, за допомогою перехресних посилань [3].

На сьогоднішній день Тім Бернерс-Лі очолює Консорціум Всесвітньої павутини (World Wide Web Consortium), який працює над розробкою та впровадженням стандартів для Інтернету.

1.2 Огляд типів сайтів

Певний клас вебресурсів іноді називають інтернет-представництвом для індивідів чи організацій. У якості прикладу такого представництва може бути сторінка-візитка на повнофункціональному вебсайті або порталі. Коли говорять про «свою сторінку в Інтернеті», це може включати в себе весь вебсайт або особисту сторінку, розміщену на іншому вебсайті чи порталі. Крім звичайних вебсайтів, існують також WAP-сайти, призначені для перегляду на мобільних телефонах [4].

Спочатку вебсайти складалися з набору статичних документів, наприклад, сторінок-візиток. З розвитком засобів комунікації кількість внутрішніх та зовнішніх посилань збільшилася. Вебсайт перестав виконувати роль простої інформаційної сторінки і став функціональним офісом,

новинним чи медійним центром. На сьогоднішній день більшість вебсайтів є динамічними та інтерактивними. Для таких випадків фахівці використовують термін «вебзастосунок» – готовий програмний комплекс для вирішення завдань на вебсайті [5].

Вебзастосунок входить до складу вебсайту, але без даних вебсайту він лише технічна оболонка. Для його функціонування необхідно заповнити його контентом та активувати.

Зазвичай, у Інтернеті кожен вебсайт має своє унікальне доменне ім'я, яке використовується для його ідентифікації в глобальній мережі. Однак існують варіанти, коли кілька доменів вказують на один і той же сайт або коли один домен асоціюється з різними сайтами.

Зазвичай, великі вебпортали використовують кілька доменів для логічного відокремлення різних видів послуг, які вони надають. Наприклад, «mail.google.com», «news.google.com» та «maps.google.com» – це різні піддомени на сайті Google, кожен з яких надає свої послуги [6].

Іншим варіантом є виділення окремих доменів для різних країн або мов. Наприклад, «google.ru» та «google.fr» представляють собою сайти Google на різних мовах, але технічно це різні вебсайти.

Безкоштовні хостинги часто об'єднують кілька вебсайтів під одним доменом. Іноді для ідентифікації конкретного сайту в адресі використовують тильду та ім'я сайту, наприклад, «example.com/~my-site-name/».

Також бувають ситуації, коли один і той же вебсайт може бути доступний за різними адресами та розміщуватися на різних серверах. У таких випадках копія оригінального сайту називається дзеркалом. Офлайнова версія сайту – це копія сайту, яку можна переглянути на будь-якому комп'ютері без підключення до мережі та використання серверного програмного забезпечення [7].

У процесі розробки вебсайту, особливо при створенні великих проєктів, важливим є тестування і налагодження в офлайновій версії. Це робиться з метою уникнення помилок та недоліків, які можуть виникнути під

час демонстрації проєкту в Інтернеті. Зазвичай для цього запрошують досвідчених тестерів і проводять тестування в корпоративній мережі або на окремих серверах з обмеженим доступом за паролем.

Розробка та обслуговування вебсайту включає в себе важливу роль адміністраторів, які керують стратегічними завданнями та інформаційним наповненням сайту або порталу. Вони співпрацюють з командою учасників проєкту та визначають стратегію та спрямування розвитку проєкту [8].

Сучасні технології, зокрема PHP, дозволяють розробляти різноманітні програми і інструменти для вебсайтів, але це також підвищує вимоги до кваліфікації учасників проєкту через багатофункціональність та складність завдань.

Відновлення сторінок (наприклад, сайту-візитки) може бути виконано секретарем-референтом, але розробка та підтримка великих сайтів та порталів вимагає високої кваліфікації фахівців. Для активного спілкування на сайті важливий директор напрямку та служба супроводу, які відповідають за листування, оперативну підтримку та інші аспекти [9].

Деякі сайти оновлюються дуже часто, навіть більше одного разу на день, особливо це стосується новинних сайтів, де актуальність інформації вельми важлива. Інтернет-магазини оновлюють інформацію залежно від наявності товарів і руху товару.

1.3 Огляд типів інтернет-ресурсів

За доступності сервісів [10]:

- відкриті – всі сервіси повністю доступні для будь-яких відвідувачів і користувачів;
- напіввідкриті – для доступу необхідно зареєструватися (зазвичай безкоштовно);

− це повністю недоступні для широкого загалу службові вебресурси організацій, включаючи корпоративні сайти, а також особисті сайти приватних осіб. Такі вебресурси доступні лише для обмеженого кола користувачів, а доступ новим користувачам зазвичай надається за допомогою запрошень, відомих як інвайти.

За фізичним розташуванням [11]:

− загальнодоступні сайти мережі Інтернет;

− локальні сайти – це вебресурси, які доступні лише в межах конкретної локальної мережі. Це можуть бути корпоративні сайти організацій або особисті вебресурси, які функціонують в локальній мережі провайдера.

1.4 Огляд основних методів розробки вебсайтів

В даний час існує безліч різних способів і засобів створити свій сайт, але будь-який із цих способів можна віднести до однієї з трьох категорій [12]:

− розробка у конструкторі сайтів;

− розробка на CMS (особливо популярна WordPress);

− самостійна розробка, у тому числі з використанням популярних інструментів та фреймворків (Laravel, Django, Spring та ін.).

1.4.1 Конструктор

Конструктор для вебсайтів є програмним рішенням, часто доступним онлайн, яке дозволяє створювати сайти за допомогою модульного підходу. Розробник обирає та комбінує готові «кубики», надані конструктором, для створення структури сайту. Цей підхід дозволяє створити сайт навіть без глибоких знань веброзробки та інших спеціалізованих навичок. Такий

інструмент часто використовується для створення простих особистих сайтів або сайтів компаній, де не вимагається високої якості програмного коду або максимальної швидкості роботи. Особливо популярні конструктори для створення простих лендінгів у рамках рекламних кампаній. Проте, варто враховувати, що вик[ористовувати конструктор для створення складних корпоративних рішень або інтернет-магазинів недоцільно [13].

Плюси:

− низька ціна. Майже всі конструктори спочатку безкоштовні, а вартість модулів, що підключаються, дуже низька;

− простота використання. Щоб створити свій сайт, достатньо вміти користуватися браузером комп'ютера і мати почуття смаку;

− уся рутинна робота виконується за допомогою конструктора. Великі програмні скрипти, підключення модулів, розміщення на хостингу та інші складні процеси виконуються за декілька кліків кнопок у зрозумілій панелі керування [14].

Мінуси:

− приховані витрати. За виглядом низької вартості на початку часто приховані додаткові, іноді значущі витрати, такі як розміщення на хостингу, придбання домену другого рівня (наприклад, example.ru), налаштування електронної пошти з використанням імені домену та інші витрати;

− домен третього чи вищого рівня. При використанні безкоштовних облікових записів у конструкторах, клієнт отримує можливість розміщення свого сайту лише на домені, який знаходиться на рівні не нижче третього, наприклад, sitename.constructor.com. Такі домени не завойовують великої довіри в інтернет-спільноті, а домени другого рівня (наприклад, sitename.com) часто коштують більше, ніж при їхньому придбанні у реєстраторів доменів безпосередньо [15];

− важкість сайту. Сайт, створений за допомогою конструктора, завжди буде завантажуватися повільніше, ніж аналогічний сайт, розроблений на CMS або створений самостійно. Це пояснюється тим, що конструктор

включає в себе велику кількість програмного коду, який не стосується безпосередньо вашого сайту, але необхідний для побудови його окремих елементів та оформлення;

− відсутність SEO. Навіть при заявках будь-якого конструктора сайтів, повноцінна оптимізація для пошукових систем (SEO) на сайті, створеному конструктором, часто виявляється неможливою. SEO включає різноманітні заходи, такі як оптимізація коду, індексація в пошукових системах та створення структури сайту, для чого потрібний доступ до програмного коду та розмітки, що зазвичай недоступно в інтерфейсі конструктора [16].

1.4.2 CMS

У цьому випадку розглянемо найпопулярну систему управління контентом (CMS) – WordPress, але існують і інші аналогічні CMS, такі як Joomla, Drupal та інші. Вони можуть не бути такими відомими, але надають схожий набір інструментів та можливостей [17].

CMS – це набір програмних інструментів для керування вмістом вебсайту. Простими словами, це фундамент та набір додаткових інструментів та доповнень, які дозволяють не лише створити вебсайт або вебдодаток, але й управляти їхньою роботою, оновлювати вміст та взаємодіяти з користувачами. Усі CMS мають панель управління зі зручним інтерфейсом. Основна мова програмування, яку використовують в CMS – це PHP. Будь-яка CMS дозволяє створювати складні рішення, такі як інтернет-магазини або великі корпоративні сайти з глибокою структурою сторінок, але вони мають свої особливості [18].

Плюси:

− безкоштовний доступ. Багато CMS (крім 1С-Бітрікс) спочатку безкоштовні, а крім того, в мережі існує безліч готових шаблонів сайтів під

них (особливо під WordPress). Берете будь-який, що сподобався, міняєте дизайн під себе і сайт готовий;

− зручне керування контентом. За допомогою панелі керування керувати сайтом легко та просто. При використанні CMS не потрібні особливі навички для керування контентом;

− безліч готових рішень. У мережі існує маса модулів, плагінів, доповнень для різних завдань (від слайдерів для картинок до систем seo-оптимізації та супроводу користувача).

Мінуси:

− вразливість сайту. Одним з головних недоліків широко використовуваних CMS є їхня вразливість до злому. Навіть порівняно з конструкторами, іноді сайти, створені на CMS, можуть мати менш ефективний рівень захисту від несанкціонованого доступу. Навіть платна CMS, така як 1С-Бітрікс, не завжди вважається найбільш надійною та захищеною від атак і неправомірних проникнень;

− вимоги до знань. Розробка сайту на CMS вже вимагає від клієнта базових знань з верстки та програмування (в основному мовою PHP), що вже додає складнощів при створенні сайту. Хоча, варто зауважити, що знання можуть знадобитися тоді, коли клієнт хоче додати до наявного шаблону новий функціонал або створити повністю з нуля власний проєкт [19];

− складнощі з перенесенням. Хоча сучасні популярні CMS автоматизували процес установки на практично будь-якому хостингу, виникнення труднощів можливе при потребі у перенесенні сайту або керуванні його станом. Це може виникнути через необхідність повторно виконати весь процес установки у випадку перенесення або потреби у виправленні конфігурації сайту;

− витрати на додатковий контент. Так само, як і в разі конструкторів, використання додаткових модулів та розширень для сайту на CMS є витратним заходом, але в цьому випадку зазвичай витрати є ще значнішими, ніж для конструкторів. Практично будь-який модуль, який використовується

регулярно, вимагає витрат або на початковому етапі, або у формі щомісячної плати за підписку;

− великий сайт – великі витрати. Створення складного та обширного проєкту за допомогою CMS може виявитися не дешевшим, а в деяких випадках навіть вартішим, ніж розробка за допомогою чистих мов програмування або з використанням фреймворків. Поширене уявлення, що використання CMS дозволяє зекономити на вартості створення сайту порівняно з професійними розробниками, часто є помилковим і може призвести до додаткових витрат для клієнта.

1.4.3 Самостійна розробка

Створення власного вебсайту або вебпрограми – це творчий і вільний, але водночас робоціний процес, схожий на виготовлення одягу на замовлення. Це вимагає великих знань, не лише в області програмування, але й у розумінні архітектури, бізнес-процесів клієнта та багатьох інших аспектів [20].

При розробці вебпродукту «з нуля,» клієнт отримує унікальний і налаштований під свої потреби продукт, який вирішує його завдання, не витрачаючи час на зайві операції.

Самостійна розробка вебпродукту дозволяє створювати проєкти будь-якої складності та враховувати всі побажання клієнта.

Плюси:

− свобода вибору. Існує можливість замовити повний комплекс послуг для ефективного вирішення бізнес-завдань. У цьому випадку весь функціонал буде розроблений з урахуванням конкретних потреб користувача та не буде адаптований з будь-якого готового шаблону;

− широкі можливості просування. На відміну від CMS та конструкторів, просунути у природному пошуку самостійно розроблений сайт набагато легше;

− індивідуальний дизайн. Лише бездоганний код може сприяти створенню продукту, який відповідає очікуванням та бажанням користувача. Особливо важливо відзначити, що ефективний підхід до дизайну і користувацького досвіду можливий лише в разі використання чистого коду, оскільки при використанні CMS клієнт може бути обмежений використанням готових рішень, які не завжди відповідають його усім потребам [21].

Мінуси:

− ціна. Обговорюваний, але потенційний недолік чистого коду. У випадку великих проєктів вибір розробки без використання конструкторів і CMS може бути найбільш ефективним і, в деяких випадках, економічно вигідним рішенням. Це може уникнути ситуацій, коли потрібно змінювати результат через невідповідність вихідного продукту потребам, та в майбутньому уникає переробки при кожній зміні вимог;

− наявність знань. Самостійна розробка вебзастосунків вимагає глибоких знань у сферах програмування, створення архітектури вебдодатків, алгоритмів та структур даних, бізнес-процесів та інших аспектів. Це призводить до необхідності звертатися до найманих фахівців або вебстудій для успішної реалізації проєкту;

− витрати часу. Навіть найвміліше володіння кодом не завжди дозволяє створити простий сайт так швидко, як використання CMS чи конструкторів. У випадках, коли час важливий фактор, використання цих інструментів стає вибором на користь ефективності, не витрачаючи час на написання коду з нуля [22].

1.5 Постановка проблеми

З ростом усвідомлення екологічних питань та підвищенням інтересу до сталої та екологічно чистої транспортної системи зростає попит на електромобілі. Із збільшенням обсягів виробництва та розповсюдженням електромобілів на ринку, постає необхідність у зручних та ефективних засобах продажу та обслуговування цих авто. Відсутність зручного онлайн-ресурсу для вибору, порівняння та придбання електромобілів створює необхідність у розробці вебсайту для продажу електромобілів на технології Java Spring Framework.

Проблема полягає у тому, що на сучасному ринку відсутні повноцінні онлайн-ресурси, які надають користувачам можливість ознайомитися із різними моделями електромобілів, порівняти їх характеристики, оцінити вартість володіння та операційні витрати. Такий вебсайт може сприяти поширенню інформації про переваги електромобілів, спонукати користувачів робити інформований вибір та сприяти переходу до більш сталого виду транспорту.

Отже, кваліфікаційна робота має на меті розробити вебсайт для продажу електромобілів на технології Java Spring, що дозволить забезпечити споживачів зручним інструментом для вибору та придбання екологічно чистих транспортних засобів, а також сприятиме збільшенню обсягів використання електромобілів для зменшення негативного впливу на довкілля.

1.6 Постановка задачі дослідження

Об’єктом розробки є вебсайт з продажу електромобілів.

Метою дослідження є реалізація клієнт-серверного застосунку для огляду та продажу електромобілів.

Таким чином, необхідно розробити методи і алгоритм створення динамічного вебсайту. Для цього необхідно вирішити наступні задачі:

− ознайомитися з сучасними Інтернет-технологіями і використовувати їх у своїй розробці;

− вивчити програмний інструментарій, застосовуваний для розробки і створення вебсайтів;

− виявити і врахувати методи і способи подання на вебсторінках різних видів інформації, що не перешкоджають їх доступності;

− ознайомитися з основними правилами і рекомендаціями по розробці і створенню вебсайтів і неухильно дотримуватися їх у своїй практиці;

− вибрати стратегію розробки та створення вебсайту.

Для даної предметної області було обрано наступні технології:

− для розробки Front-end частини застосунку будуть використовуватися HTML, CSS, JavaScript та фреймворк BootStrap;

− для розробки Back-end частини застосунку будуть використовуватися технології Java EE та Spring Framework, а саме Spring Core, Spring MVC, Spring Data/Hibernate та Spring Security;

− для моделювання та реалізації бази даних вебзастосунку обрано СУБД PostgreSQL.

2 МОДЕЛЮВАННЯ БАЗИ ДАНИХ, СТРУКТУРИ ТА ДИЗАЙНУ ВЕБСАЙТУ

2.1 Загальний огляд концепції роботи застосунку

Основною ідеєю вебсайту є клієнт-серверний вебзастосунок, який поєднує інформаційну частину з оглядом електромобілів компанії Tesla та реалізацію інтернет-магазину електромобілів різних компаній.

У межах цієї програми користувач може мати три ролі: незареєстрований користувач, зареєстрований користувач і адміністратор. Кожна роль має свій відповідний перелік можливостей і функціоналу.

Незареєстрований користувач має повне право на вивчення інформаційної частини вебсайту, пограти в міні-гру, а також зареєструватися.

Зареєстрований користувач (клієнт) має всі привілеї незареєстрованого користувача, а також має можливість користуватися магазином, робити замовлення та продивлятися список своїх зроблених замовлень.

Адміністратор програми має доступ до таблиць користувачів, товарів та угод. Він має додавати нові товари, редагувати та видаляти вже існуючі, а також може видаляти оформлені користувачами замовлення.

2.2 Розробка структури дизайну вебсайту з продажу електромобілів

Дизайн сайту буде збудований методом секційної верстки (рис. 2.1-2.3). Для цього необхідно виділити основні секції документа, такі як header, footer, та декілька необхідних функціональних блоків.

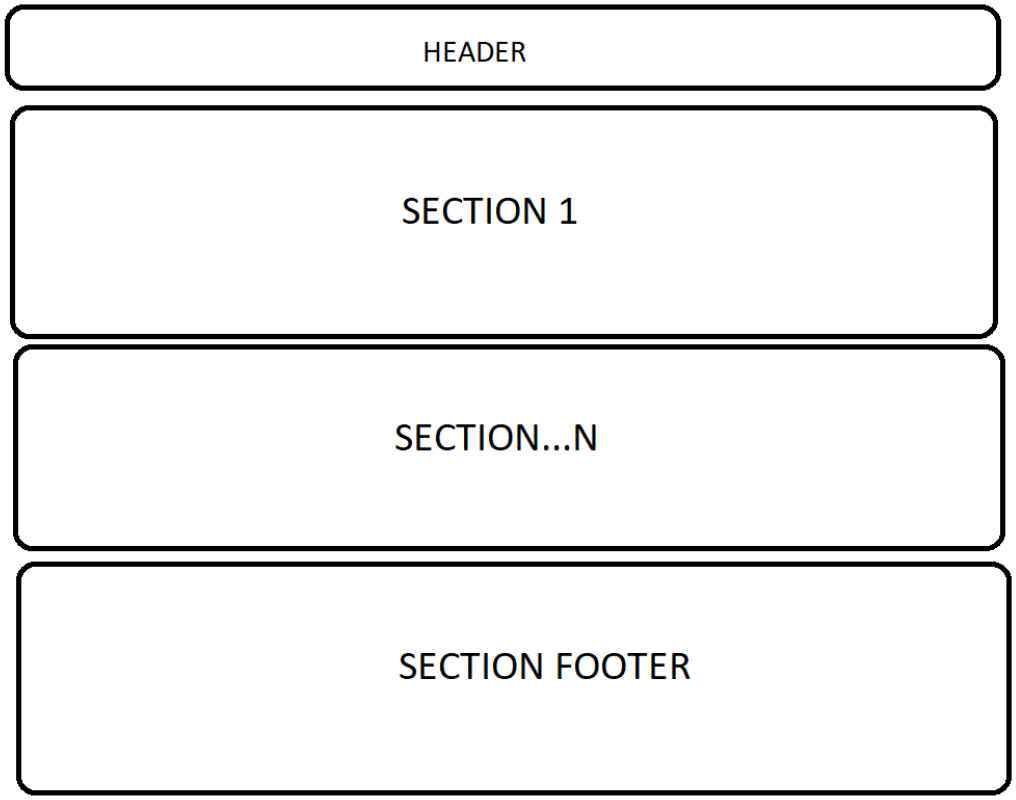

Рисунок 2.1 – Макет головної інформаційної сторінки вебсайту

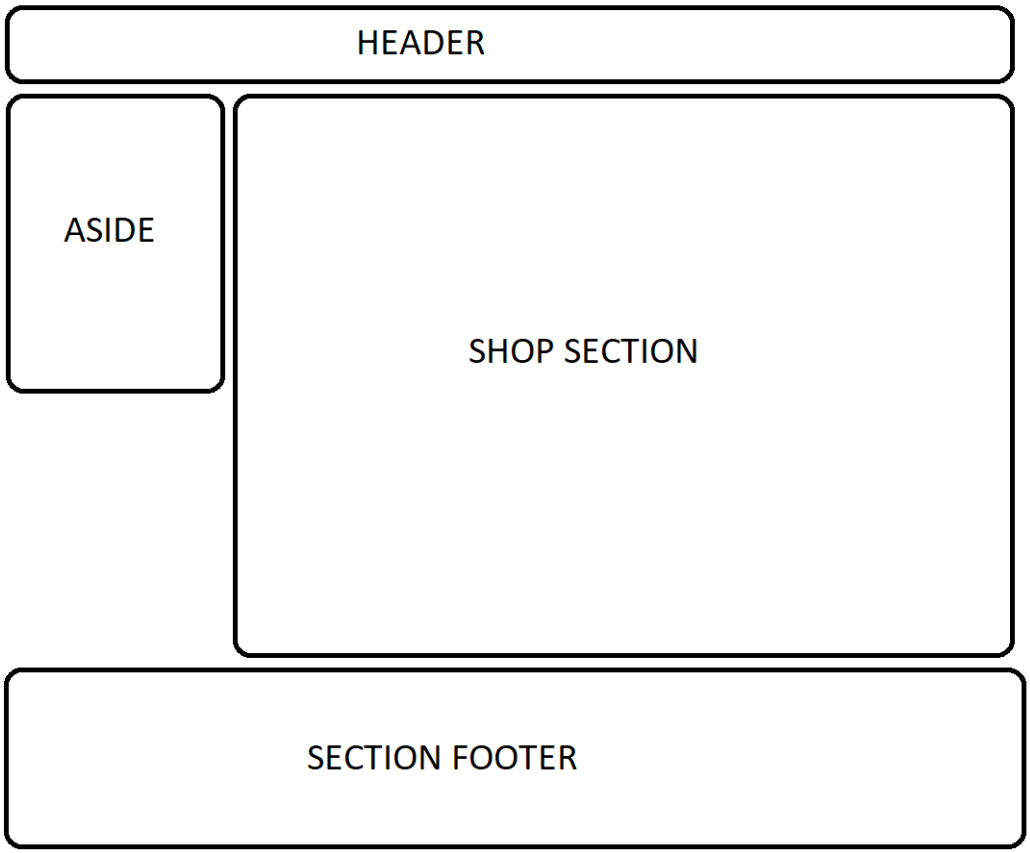

Рисунок 2.2 – Макет сторінки магазину

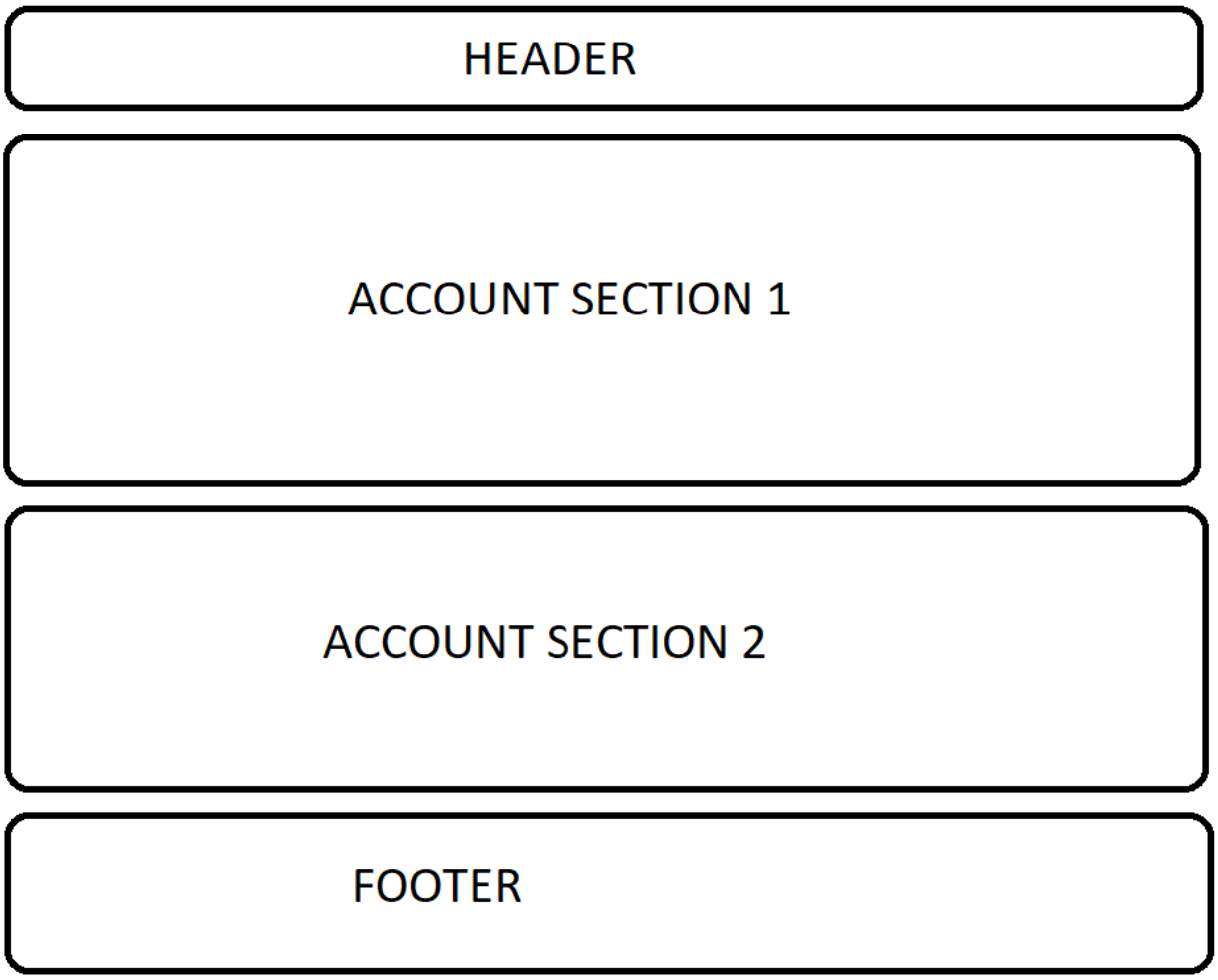

Рисунок 2.3 – Макет сторінки акаунту користувача

2.3 Розробка бізнес-правил застосунку

Враховуючи концепцію проєкту, необхідно побудувати бізнес-правила до нього:

- незареєстрований користувач може користуватися інформаційною частиною вебсайту;
- незареєстрований користувач не може оформити замовлення;
- зареєстрований користувач може користуватися інтернет-магазином електромобілів;
- зареєстрований користувач може оформити замовлення товарів;

– адміністратор може додавати нові товари в систему, а також редагувати і видаляти вже існуючі;

– адміністратор може видаляти оформлені клієнтами замовлення.

2.4 Моделювання структури прав кожної ролі у системі

За визначеними бізнес-правилами побудуємо UML-діаграму прецедентів вебсайту з продажу електромобілів (рис. 2.4).

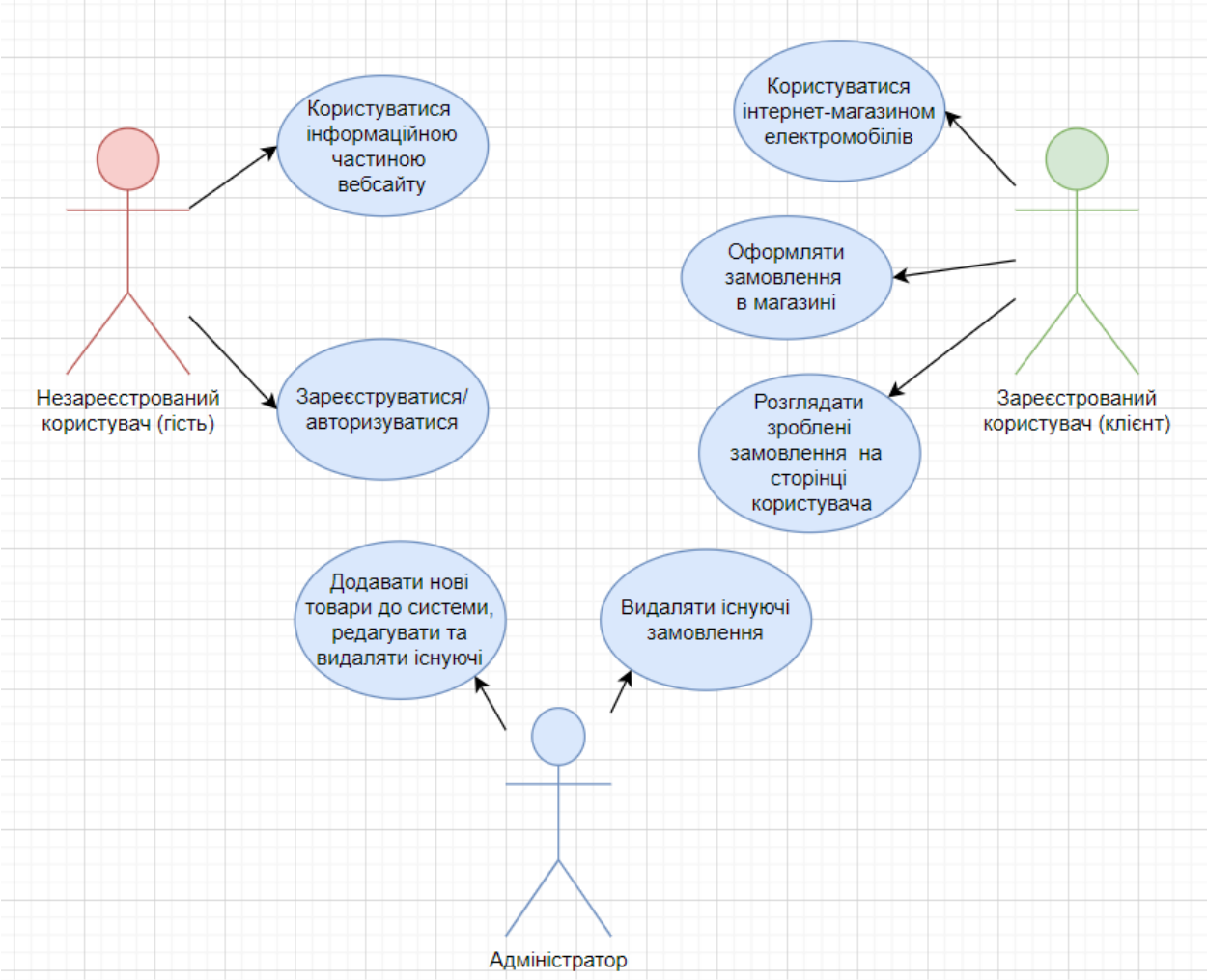

Рисунок 2.4 – UML-діаграма прецедентів вебсайту

2.5 Розробка бази даних застосунку з врахуванням розроблених бізнес-правил

2.5.1 Визначення необхідних таблиць та полів бази даних

Для розробки бази даних програми необхідно визначити, які сутності будуть використовуватися в роботі.

Так як у нас буде присутній інтернет-магазин електромобілів, нам необхідно їх десь зберігати. Для цього створимо таблицю product з наступними полями:

- product_id (ідентифікатор товарів);
- product_brand (бренд);
- product_name (назва моделі);
- product_year_of_release (рік випуску електромобіля);
- product_price (ціна товару).

Оскільки в системі будуть присутні користувачі різних ролей, необхідно створити для них відповідну таблицю – person, за наступними полями:

- person_id (ідентифікатор користувача);
- person_full_name (ім’я користувача);
- person_username (логін користувача);
- person_phone (номер телефону користувача);
- person_email (email користувача);
- person_password (пароль користувача);
- person_date_of_birth (дата народження користувача);
- person_role (роль користувача).

Інформація про замовлення зберігається у таблиці orders з наступними полями:

- order_id (ідентифікатор замовлення);
- order_created_at (дата створення замовлення);
- person_id (зовнішній ключ користувача, власника замовлення);

− product_id (зовнішній ключ товару, який був куплений в рамках цього замовлення).

На основі цієї інформації побудуємо таблицю зв’язків між сутностями (табл. 2.1).

Таблиця 2.1 – Типи зв’язків між сутностями

Назва сутності	**Назва сутності**	**Зв’язок**
Person	Order	1:M
Product	Order	1:M

2.5.2 Моделювання бази даних

На основі визначених таблиць можна приступити до моделювання бази даних. Для цього чудово підійде ER-діаграма.

Діаграма прив’язки сутностей (ER) – це графічне зображення, яке ілюструє, як різні «сутності,» такі як люди, об’єкти чи концепції, пов’язані між собою в системі. Цей тип діаграми, ER-діаграма, найчастіше використовується для проєктування та оптимізації реляційних баз даних у сферах програмного забезпечення, бізнес-інформаційних систем та наукових досліджень. Для відображення взаємозв’язків між об’єктами, відношень та їх атрибутів на ER-діаграмі використовуються геометричні символи, такі як прямокутники, ромби, овали та лінії [23].

Пітер Чен, викладач університету Карнегі-Меллона в Піттсбурзі, приписується розробка цього виду діаграми ER. Він почав використовувати її для проєктування баз даних у 70-х роках минулого століття. Під час свого робочого перебування як помічник професора в MIT Sloan School of Management, у 1976 році він опублікував статтю під назвою «Модель відношення сутностей для одночасного представлення даних». В історичному контексті, ідеї, що лягли в основу зображення взаємозв’язків

між об’єктами, мають свої коріння у давньогрецькій філософії, і можна знайти в працях великих думкопрохідців, таких як Аристотель, Сократ і Платон.Про них згадується у роботах філософів-логіків пізнішого історичного періоду – Чарльза Сандерс Пірса та Готлофа Фреге. До 1960-х і 1970-х років Чарльз Бахман з однодумцями працювали над теоретичною моделлю Чена. Бахман розробив діаграму структури даних, яку він назвав. Браун опублікував роботи з моделювання реальних систем [24].

Джеймс Мартін вніс певні уточнення до діаграми ER, а його робота, а також спільна робота з іншими дослідниками, включаючи Чена, Бахмана, Брауна та інших, призвели до створення уніфікованої мови моделювання (UML), яка широко використовується у розробці програмного забезпечення.

Цей метод ER-діаграм використовується для моделювання та розробки реляційних баз даних. Він особливо популярний при створенні логічної моделі та визначенні конкретних технологій, які будуть використовуватися в фізичній моделі даних. У сфері розробки програмного забезпечення ER-діаграма часто виступає як початковий етап для встановлення вимог до проєкту інформаційних систем. Реляційна база даних відображається в ER-діаграмі у вигляді еквівалентної реляційної таблиці і використовується для виразу відповідних вимог. Області застосування:

− виправлення дефектів у базі даних включає в себе використання діаграм ER для аналізу існуючих баз даних з метою виявлення та усунення проблем в логіці або структурі розгортання. Створення діаграми спрямовано на наглядне відображення місць, де ці проблеми виникають [25];

− системи бізнес-інформації. Діаграми використовуються для розробки та аналізу баз даних, які застосовуються в бізнес-процесах. Будь-який бізнес-процес, який використовує полі дані, включаючи сутності, дії та взаємодію, може потенційно здобути користь від використання реляційної бази даних. Це може призвести до оптимізації процесів, спрощення обробки інформації та покращення результатів;

− реорганізація бізнес-процесів (BPR). Моделі діаграм ER допомагають у аналізі баз даних, що використовуються під час реорганізації бізнес-процесів [26];

− освіта. Сучасна база даних використовується як засіб для зберігання реляційної інформації з метою освітніх завдань та подальшого пошуку, тому використання діаграм ER може бути важливим при проєктуванні цих структур;

− дослідження. Оскільки так багато досліджень зосереджено на структурованих фактів, ER може відігравати ключову роль у створенні корисних баз даних для їх аналізу.

Діаграми ER баз даних є вельми корисним інструментом для створення та управління великими обсягами даних. По-перше, вони легко зрозумілі, що робить їх ідеальними для спілкування між дизайнерами, розробниками, клієнтами та кінцевими користувачами, незалежно від рівня їхньої експертизи в галузі IT [27].

Ці діаграми легко перетворюються в реляційні таблиці, які можна використовувати для швидкого створення баз даних. Крім того, розробники баз даних можуть використовувати ER-діаграми як плани для імплементації даних у конкретні програмні продукти. Ці діаграми можуть бути корисними і в інших контекстах, таких як пояснення взаємозв’язків і операцій всередині організації. Діаграма ER дуже популярна, оскільки має безліч переваг:

− ефективна комунікація дозволяє читачам легко розуміти відносини між різними предметними галузями діаграми ER;

− символи використовуються для ефективного представлення інформації, а також допомагають розуміти роботу бази даних;

− візуальне представлення діаграми потоків даних разом із діаграмами ER можуть ефективно використовуватися для візуального представлення макету [28];

− просте розуміння дизайну за допомогою діаграм ER;

– висока гнучкість. Ефективне використання діаграм ER полягає в установленні взаємозв'язків із наявними системами. Для цього можуть використовуватися математичні формули та реляційні таблиці для виконання цієї операції [29].

Побудуємо ER-діаграму (рис. 2.5) для бази даних (табл. 2.2-2.4).

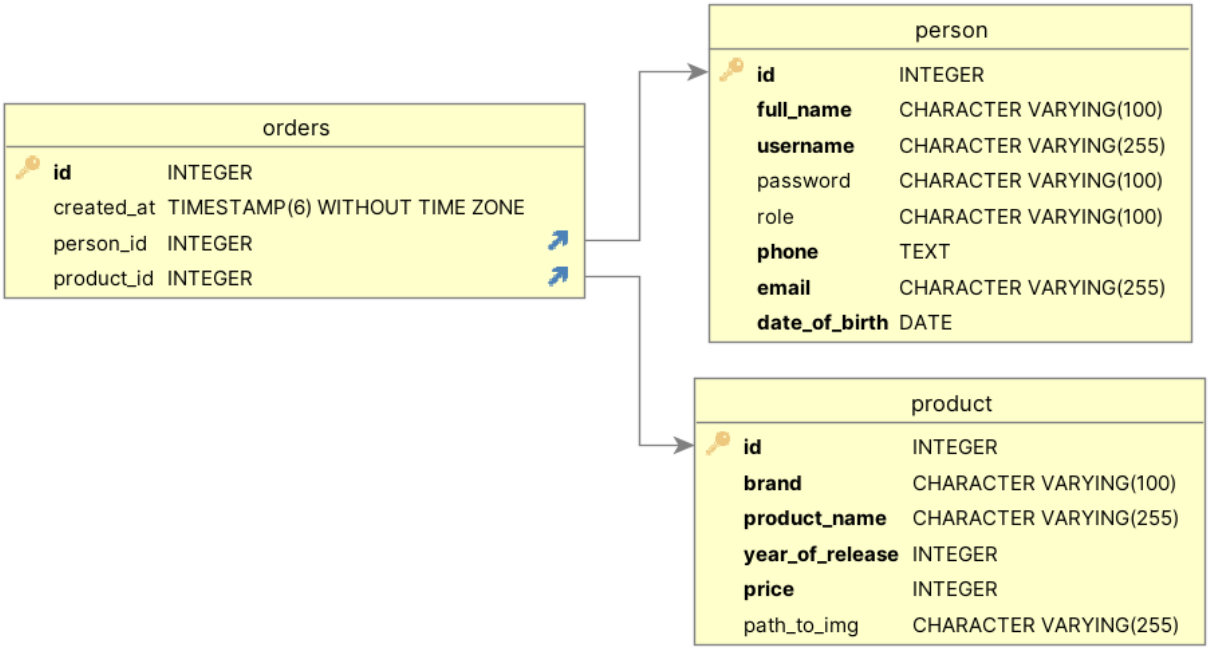

Рисунок 2.5 – Згенерована в DbVisualizer (PostgreSQL) модель бази даних

Таблиця 2.2 – Опис структури таблиці «Product»

Ключ	**Ім'я поля**	**Тип даних**	**Розмір поля**	**Опис**
PK	Product_id	INT		Ідентифікатор товарів
	Product_brand	VARCHAR	255	Бренд
	Product_name	VARCHAR	255	Назва моделі
	Product_year_of_release	INT		Рік випуску електромобіля
	Product_price	INT		Ціна

Таблиця 2.3 – Опис структури таблиці «Person»

Ключ	**Ім’я поля**	**Тип даних**	**Розмір поля**	**Опис**
PK	person_id	INT		Ідентифікатор користувача
	person_full_name	VARCHAR	255	Ім’я користувача
	person_username	VARCHAR	255	Логін користувача
	person_mobile	VARCHAR	100	Номер телефону користувача
	person_email	VARCHAR	255	Email користувача
	person_password	VARCHAR	100	Пароль користувача
	person_date_of_birth	DATETIME		Дата народження користувача
	person_role	VARCHAR	50	Роль користувача

Таблиця 2.4 – Опис структури таблиці «Orders»

Ключ	**Ім’я поля**	**Тип даних**	**Розмір поля**	**Опис**
PK	order_id	INT		Ідентифікатор замовлення
	order_created_at	DATETIME		Дата створення замовлення

Продовження таблиці 2.4

FK	person_id	INT		Зовнішній ключ користувача, власника замовлення
FK	product_id	INT		Зовнішній ключ товару замовлення

2.5.3 Створення фізичної моделі бази даних

На основі визначених раніше моделей створимо структуру бази даних застосунку (лістинг 2.1-2.3).

Лістинг 2.1 Створення таблиці «Products»:

```
CREATE TABLE Product (
    id int GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    brand varchar(100) NOT NULL,
    product_name varchar(255) NOT NULL,
    year_of_release int NOT NULL,
    price int NOT NULL,
    path_to_img varchar(255)
);
```

Лістинг 2.2 Створення таблиці «Users»:

```
CREATE TABLE Person (
    id int GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
```

```
    full_name varchar(100) NOT NULL,
    username varchar(255) NOT NULL UNIQUE,
    password varchar(100),
    role varchar(100),
    phone text NOT NULL,
    email varchar(255) NOT NULL,
    date_of_birth DATE NOT NULL
);
```

Лістинг 2.3 Створення таблиці «Orders»:

```
CREATE TABLE Orders (
    id int GENERATED BY DEFAULT AS IDENTITY PRIMARY KEY,
    created_at timestamp,
    person_id int REFERENCES Person(id) ON DELETE CASCADE,
    product_id int REFERENCES Product(id) ON DELETE CASCADE
);
```

3 РОЗРОБКА ВЕБСАЙТУ З ПРОДАЖУ ЕЛЕКТРОМОБІЛІВ

3.1 Обґрунтування вибору середовища програмної реалізації

Для реалізації вебзастосунку в якості IDE були обрані IntelliJ IDEA та Visual Studio Code. В якості бази даних та СУБД було обрано PostgreSQL та PgAdmin. Для розробки Front-end частини були використані наступні технології: HTML, CSS, JavaScript та фреймворк BootStrap. Для розробки Back-end частини використовувалися технології Java EE та Spring Framework.

3.1.1 IntelliJ IDEA

Середовище розробки JetBrains IntelliJ IDEA є передовим інструментарієм для швидкої розробки додатків на Java. IntelliJ IDEA – це високотехнологічне рішення, яке об’єднує в собі інтелектуальний редактор вихідних текстів з розширеними можливостями автоматизації. Воно також включає потужні інструменти для рефакторингу коду, інтеграцію з технологіями J2EE, підтримку платформ Ant/JUnit для тестування, і системи управління версіями. JetBrains IntelliJ IDEA пропонує унікальні можливості, такі як Code Inspection для оптимізації коду і інноваційний візуальний конструктор графічних інтерфейсів. Важливо відзначити, що IntelliJ IDEA звільняє розробників від рутинної роботи, допомагає вчасно виявляти помилки і підвищує якість коду, піднімаючи продуктивність розробників на новий рівень [30].

Хоча середовище спочатку створювалося для максимальної оптимізації Java-розробки, зараз у ній є опції для роботи з більшістю затребуваних мов програмування, причому деякі інструменти використовують технологію

машинного навчання. IDE надає інтелектуальну допомогу під час написання коду [31]:

- виконує глибокий аналіз та створює віртуальну карту проєкту;
- виявляє помилки та пропонує варіанти виправлення;
- автоматично доповнює код з огляду на контекст;
- проводить валідацію (перевірку на відповідність стандартам) коду;
- виконує рефакторинг коду – робить його простішим і зрозумілішим;
- підтримує роботу із вставками, написаними іншими мовами програмування [32];
- дозволяє використовувати шаблони для вставки фрагментів коду, що повторюються;
- пропонує оптимізацію за допомогою профілювача – інструмента, який аналізує продуктивність коду та оцінює навантаження на процесор та оперативну пам’ять.

3.1.2 Visual Studio Code

Visual Studio Code (VS Code) – це текстовий редактор вихідного коду, який був розроблений компанією Microsoft і призначений для використання на операційних системах Windows, Linux та macOS. Він відомий своєю легкістю та високою продуктивністю та призначений для кросплатформеного розробки веб та хмарних додатків [33].

VS Code володіє вражаючим набором функцій, що включають в себе вбудований відладчик, інтеграцію з Git для керування версіями коду, автоматичне підсвічування синтаксису, IntelliSense для автоматичного завершення коду та засоби для виконання рефакторингу, що сприяють поліпшенню структури коду [34].

Крім того, VS Code славиться своєю гнучкістю та можливостями налаштування. Користувачі можуть змінювати теми оформлення,

налаштовувати комбінації клавіш та створювати власні файли конфігурації для оптимізації редактора під свої потреби.

Особливістю VS Code є те, що він є вільно розповсюджуваним програмним забезпеченням з відкритим вихідним кодом. Це дозволяє розробникам спільноти активно співпрацювати над покращенням редактора та створенням розширень для вирішення конкретних завдань. У результаті VS Code завжди вдосконалюється та підтримується широким колом розробників [35].

3.1.3 BootStrap

Bootstrap – це відкритий та безкоштовний HTML, CSS та JS фреймворк, який використовується веброзробниками для швидкої верстки адаптивних дизайнів сайтів та вебзастосунків [36].

Bootstrap складається з:

− інструментів для створення макета (обгорткових контейнерів, потужної системи сіток, гнучких медіа-об’єктів, адаптивних утилітних класів) [37];

− класів для стилізації базового контенту: тексту, зображень, коду, таблиць та figure;

− готових компонентів: кнопок, форм, горизонтальних і вертикальних навігаційних панелей, слайдерів, списків, акордеонів, модальних вікон, спливаючих підказок та ін [38];

− утилітних класів для вирішення традиційних завдань, що найбільш часто виникають перед веброзробниками: вирівнювання тексту, відображення та приховування елементів, завдання кольору, фону, margin і padding відступів, і т.д [39].

Переваги Bootstrap при його використанні для frontend розробки сайтів:

− швидке створення високоякісної адаптивної верстки стає доступним навіть для початківців веброзробників за рахунок використання готових класів та компонентів, розроблених професіоналами [40];

− кросбраузерність та кросплатформенність (коректне відображення та робота сайту у всіх підтримуваних цим фреймворком браузерах та операційних системах);

− наявність великої кількості готових добре продуманих компонентів, протестованих величезним співтовариством веброзробників на різних пристроях [41];

− можливість індивідуального налаштування під свій проєкт досягається шляхом зміни змінних SCSS та використання міксинів. Це дозволяє налаштовувати різноманітні параметри, такі як кількість колонок, кольори, радіуси заокруглень, відступи між колонками і т.д;

− низький поріг входження; для роботи з фреймворком не обов’язково мати «глибокі» знання з HTML, CSS, JavaScript та jQuery (досить знати лише основи цих технологій) [42];

− однорідність дизайну та його узгодженість між різними компонентами (у Bootstrap всі компоненти виконані в єдиному стилі);

− наявність обширних спільнот та навчальних ресурсів створює унікальну можливість не лише глибоко освоїти фреймворк, але й знаходити відповіді на практично будь-які питання, які можуть виникнути у користувача.

3.1.4 PostgreSQL

PostgreSQL – це відкрита СКБД з високою надійністю та розширюваністю. Вона є безкоштовною та має відкритий вихідний код, що робить її доступною для комерційних та некомерційних проєктів. PostgreSQL підтримує транзакції, резервне копіювання та відновлення даних, що робить

її ідеальною для корпоративних застосувань. Її розширення дозволяють додавати нові функції та типи даних. PostgreSQL також підтримує реляційні дані та JSON-дані [43].

pgAdmin – це безкоштовний інструмент для адміністрування та розробки баз даних PostgreSQL. Він має графічний інтерфейс для створення та керування базами даних та таблицями. pgAdmin спрощує виконання SQL-запитів та підтримує резервне копіювання та відновлення даних. Ці інструменти дозволяють легко управляти великими обсягами даних і використовувати PostgreSQL для різних видів проєктів [44].

3.1.5 Spring Framework

Spring є надзвичайно популярним середовищем розробки програм для корпоративних застосунків Java. Мільйони розробників по всьому світу використовують Spring Framework для створення високопродуктивного, легко тестованого та повторно використовуваного коду [45].

Spring Framework – це платформа Java з відкритим вихідним кодом, яку спроєктував Родом Джонсон та випустив під ліцензією Apache 2.0 в червні 2003 року.

Ця платформа надає широкий спектр функцій, які можна використовувати для розробки будь-яких Java-додатків. Крім того, є розширення для створення вебдодатків на основі платформи Java EE. Spring спрямований на спрощення процесу розробки за допомогою J2EE та популяризацію кращих практик програмування, використовуючи модель програмування на основі POJO (Plain Old Java Object). Нижче наведено список деяких переваг використання Spring Framework [46]:

- Spring дозволяє розробникам створювати програми корпоративного рівня, використовуючи прості об’єкти Java (POJO). Однією з переваг використання лише POJO є те, що вам не потрібен контейнер EJB, такий як

сервер додатків. Вам надається можливість використовувати надійний контейнер сервлетів, такий як Tomcat, або комерційний продукт [47];

− Spring має модульну структуру. Незважаючи на значну кількість пакетів та класів, вам доведеться приділити увагу лише тим, які безпосередньо стосуються вашого проєкту, і проігнорувати всі інші;

− тестування програм, розроблених на Spring, виявляється дуже простим, оскільки код, що залежить від середовища, може бути легко переміщений у цю платформу. Крім того, використання JavaBean-style POJO спрощує впровадження залежностей для введення тестових даних;

− вебфреймворк Spring представляє собою добре спроєктований і популярний вебфреймворк, що становить відмінну альтернативу іншим вебфреймворкам, таким як Struts або менш популярним або переускладненим рішенням (рис. 3.1);

− Spring надає зручний API для перетворення винятків, специфічних для певної технології (наприклад, JDBC, Hibernate або JDO), у злагоджені, неперевірені винятки [48];

− контейнери IoC в Spring, як правило, є легкими, що робить їх особливо вигідними в порівнянні з важкими контейнерами, такими як контейнери EJB. Це особливо цінно для розробки та розгортання програм на пристроях із обмеженими ресурсами;

− Spring надає консистентний інтерфейс для керування транзакціями, який може масштабуватися від локальних транзакцій (наприклад, використовуючи одну базу даних) до глобальних транзакцій (наприклад, з використанням JTA) [49].

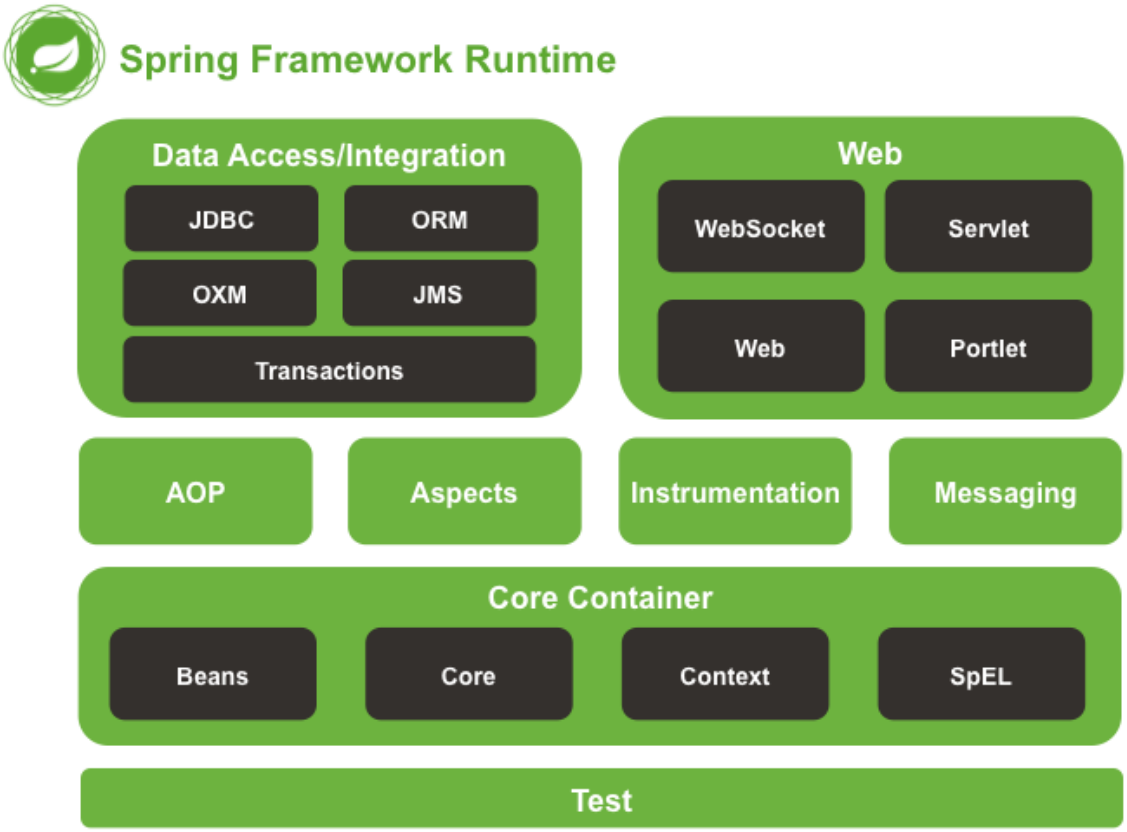

Рисунок 3.1 – Компоненти Spring Framework

3.2 Інструкція користувача та дорожня карта проєкту

При заході на початкову сторінку для незареєстрованого користувача є досить багато функціоналу. Він може вивчати всю інформацію на головній інформаційній сторінці та пограти на ній у міні-гру. Після цього користувачеві необхідно авторизуватися, або, якщо у нього ще немає облікового запису – зареєструватися. Зареєстрований користувач може продивлятися товари в інтернет-магазинці та робити замовлення в ньому. Існуючі замовлення можна відстежити на сторінці акаунту користувача. Також адміністратор може авторизуватися та потрапити на сторінку адміністратора, з відповідним функціоналом (рис. 3.2).

В рамках вебсайту, що розробляється, існують наступні сторінки:

- головна сторінка з інформацією;
- сторінка авторизації користувача;
- сторінка реєстрації користувача;
- сторінка адміністратора;
- сторінка інтернет-магазину;
- сторінка кошика;
- сторінка акаунту користувача.

Рисунок 3.2 – Дорожня карта вебсайту

3.3 Тестування розробленого вебсайту з продажу електромобілів

У цьому підрозділі наведено скріншоти (рис. 3.3 – 3.18), що ілюструють ключові етапи роботи вебзастосунку. Вони становлять важливий візуальний компонент, який сприяє кращому розумінню функціоналу та інтерфейсу користувача. Проаналізувавши ці знімки екрану, є можливість отримати детальне уявлення про взаємодію користувачів із системою, а також функціонал, доступний кінцевим користувачам.

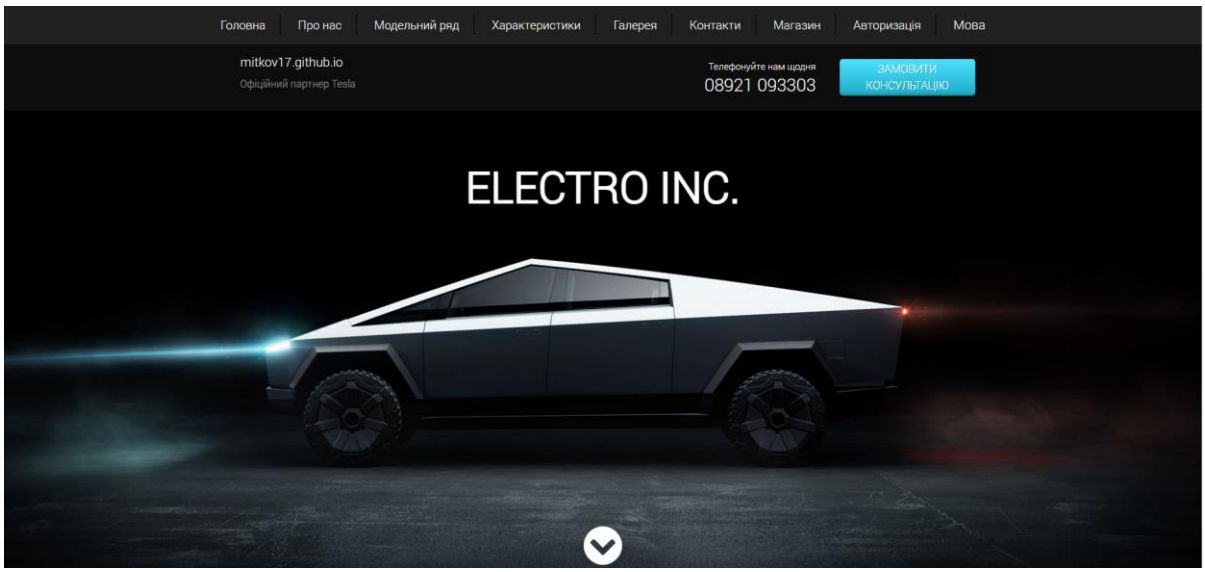

Рисунок 3.3 – Головний екран вебсайту з продажу електромобілів

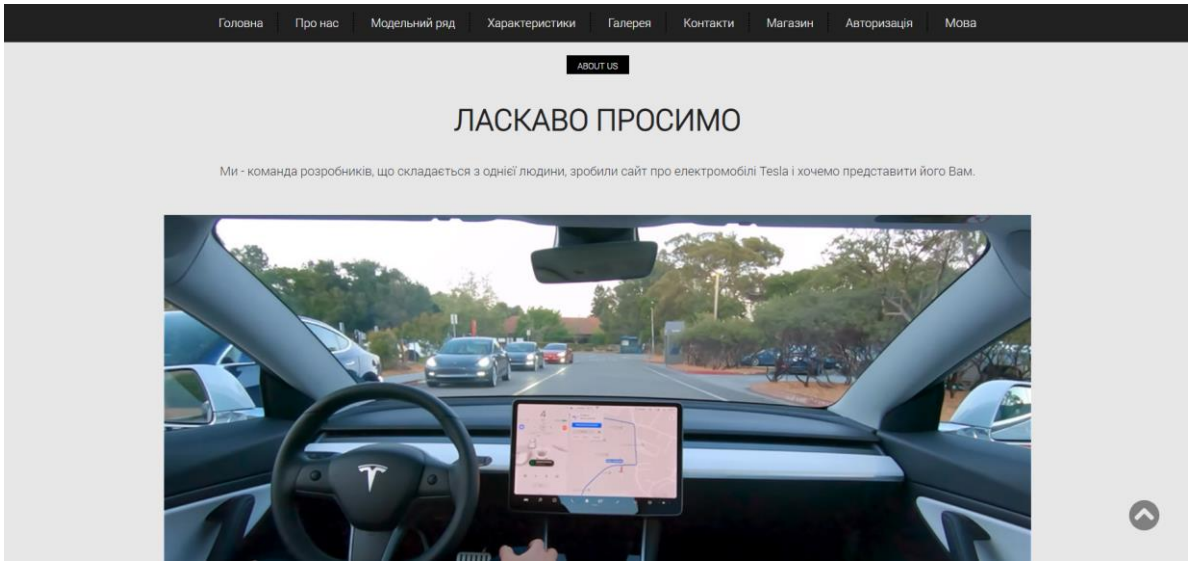

Рисунок 3.4 – Welcome-секція вебсайту

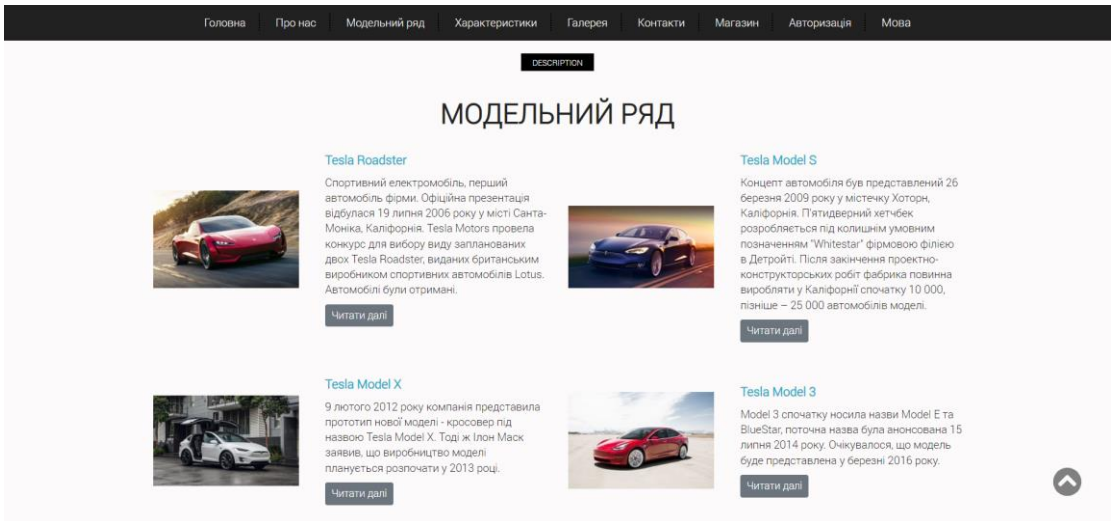

Рисунок 3.5 – Інформаційна секція з огляду електромобілів

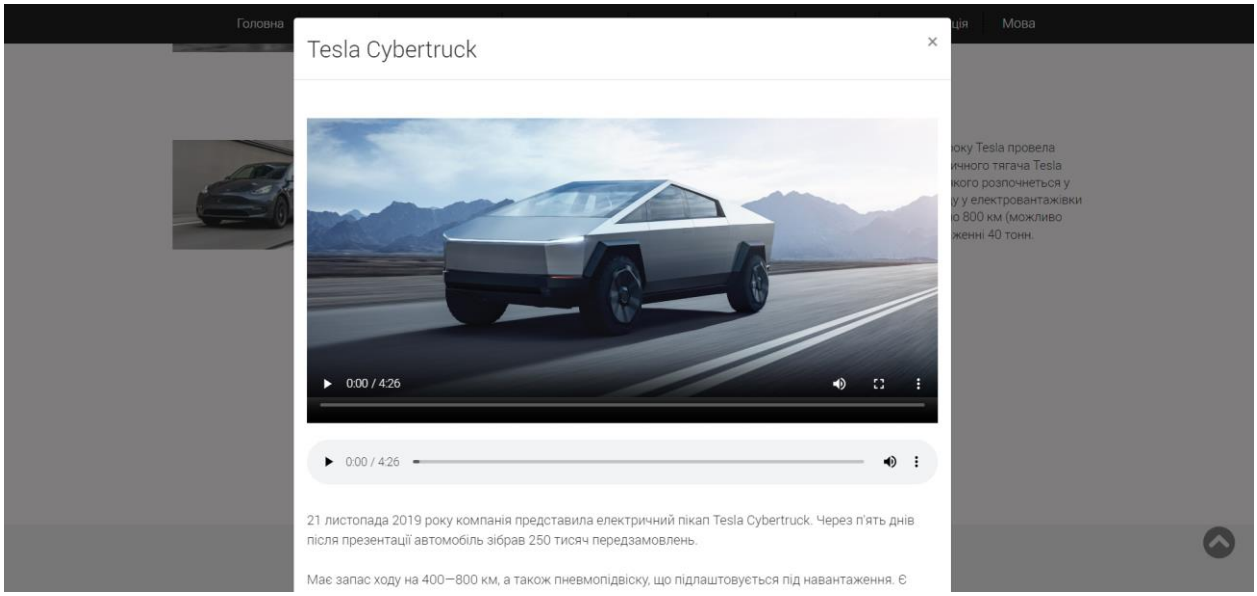

Рисунок 3.6 – Детальна інформація щодо кожного з електромобілів

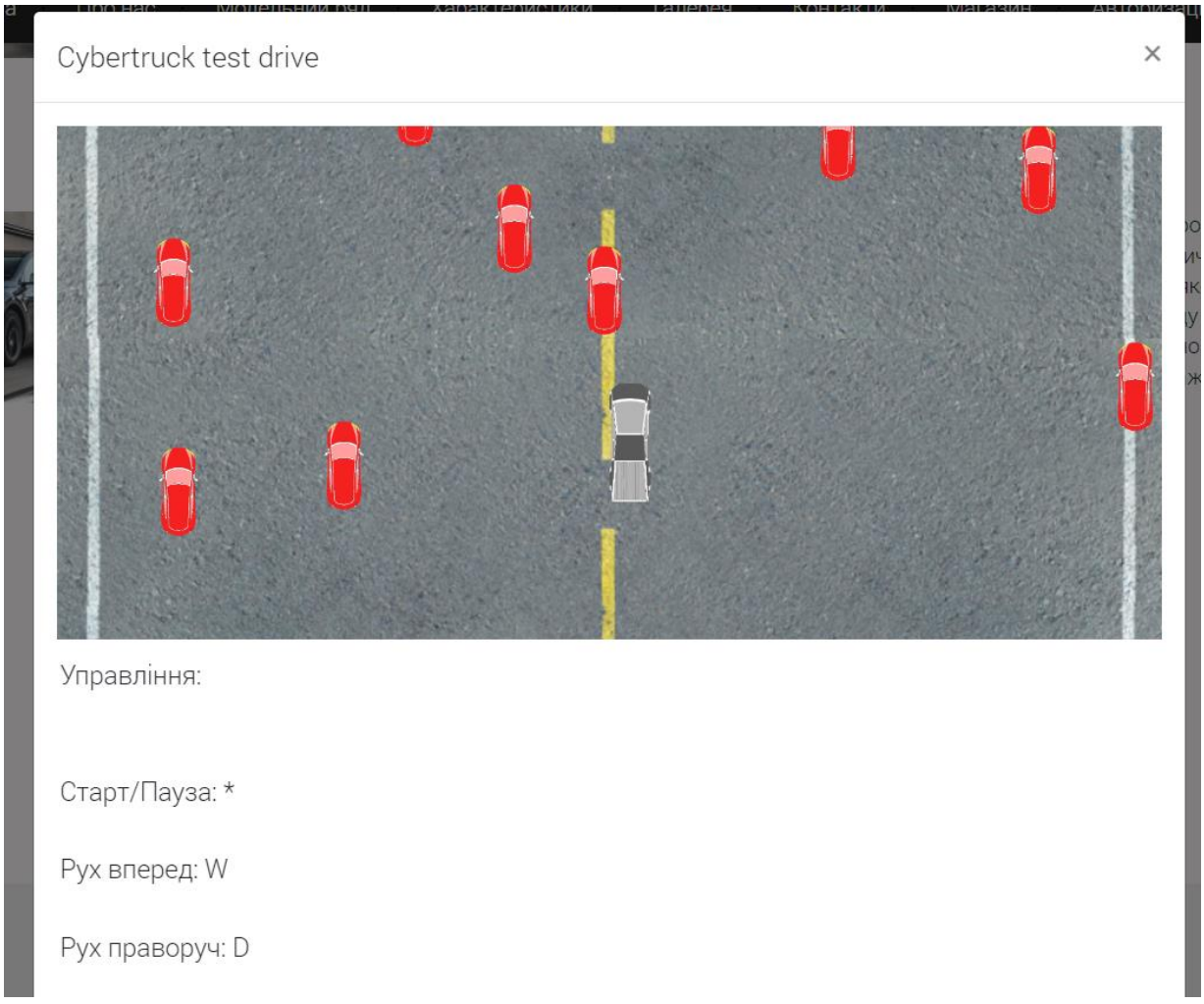

Рисунок 3.7 – Міні-гра «Тест-драйв Tesla Cybertruck»

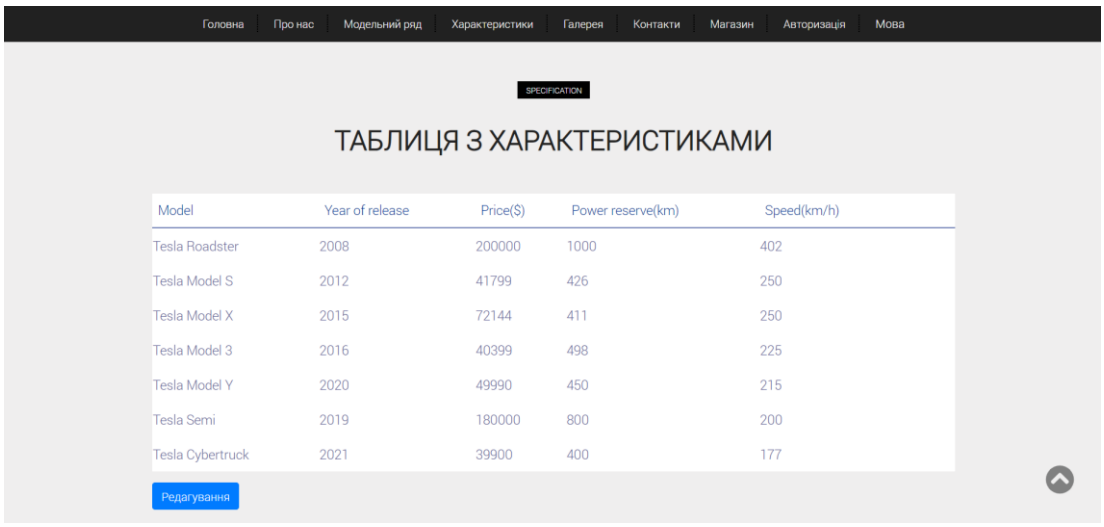

Рисунок 3.8 – Таблиця з характеристиками електромобілів Tesla

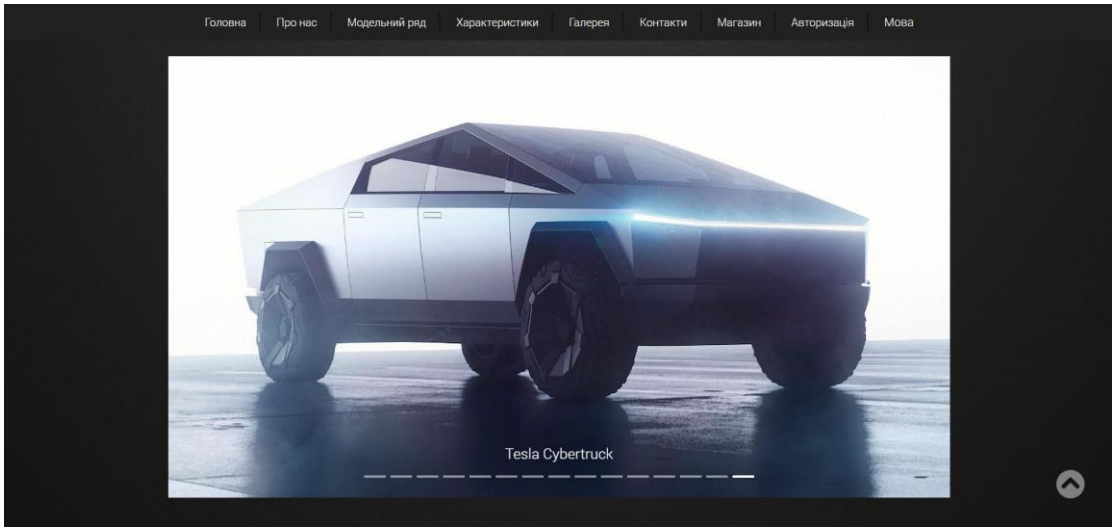

Рисунок 3.9 – Секція зі слайдером (Галерея) світлин з електромобілями Tesla

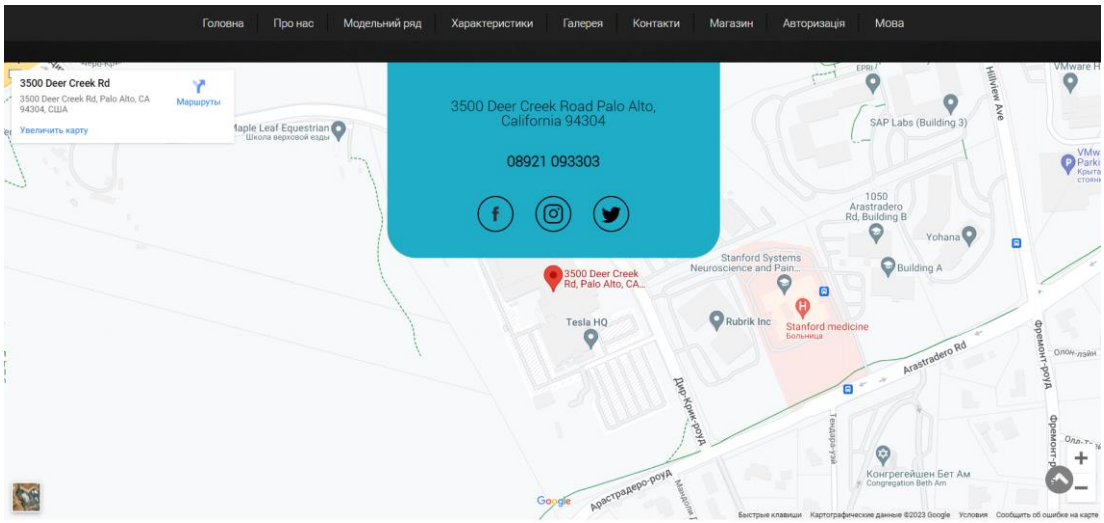

Рисунок 3.10 – Футер вебсайту

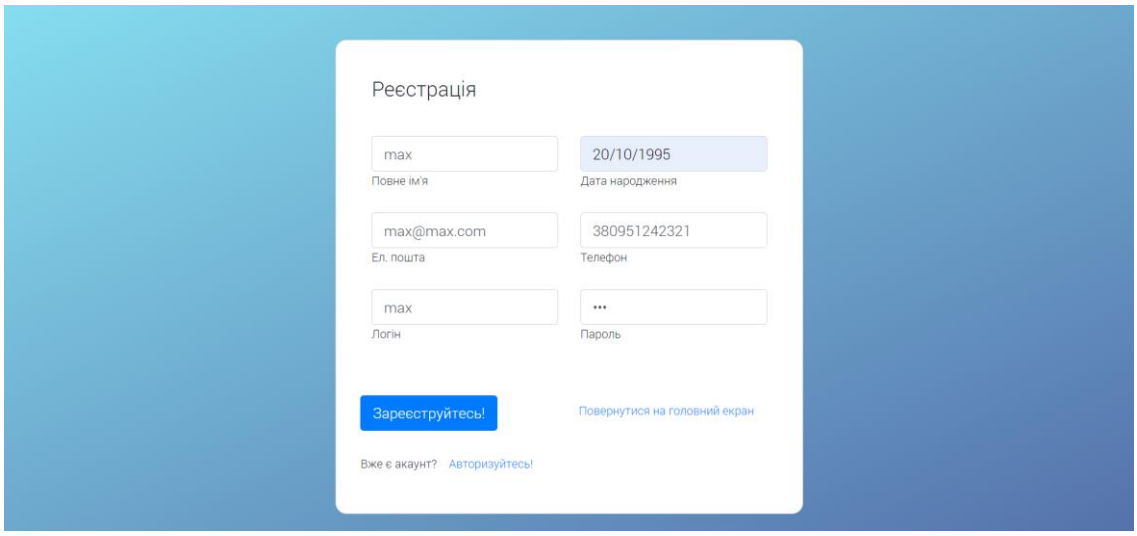

Рисунок 3.11 – Сторінка реєстрації

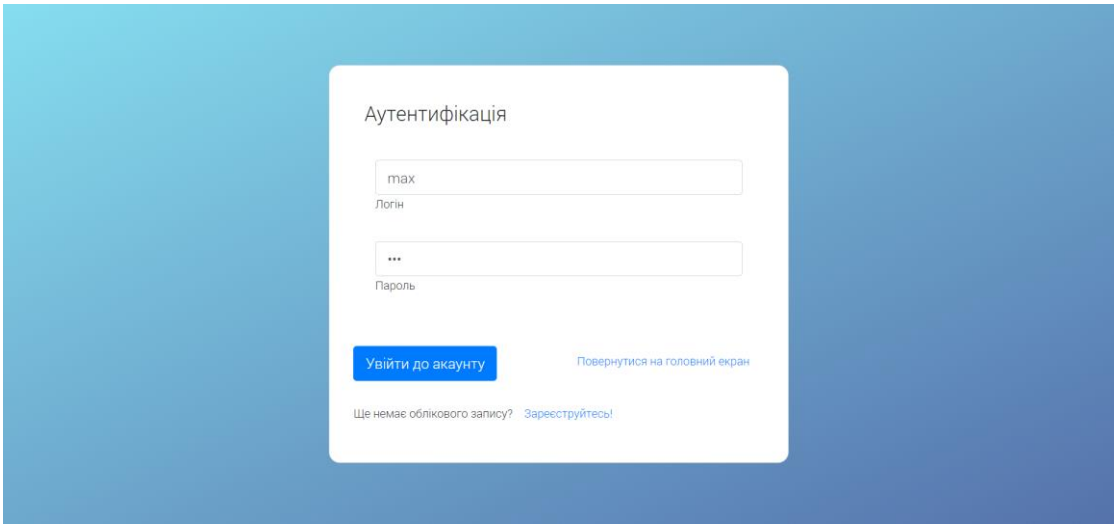

Рисунок 3.12 – Сторінка аутентифікації

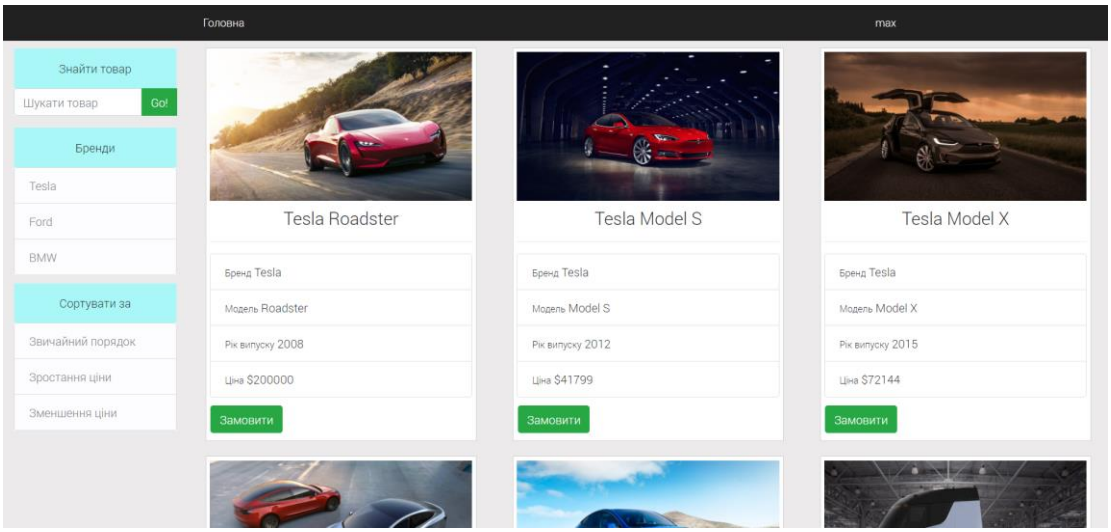

Рисунок 3.13 – Сторінка інтернет-магазину

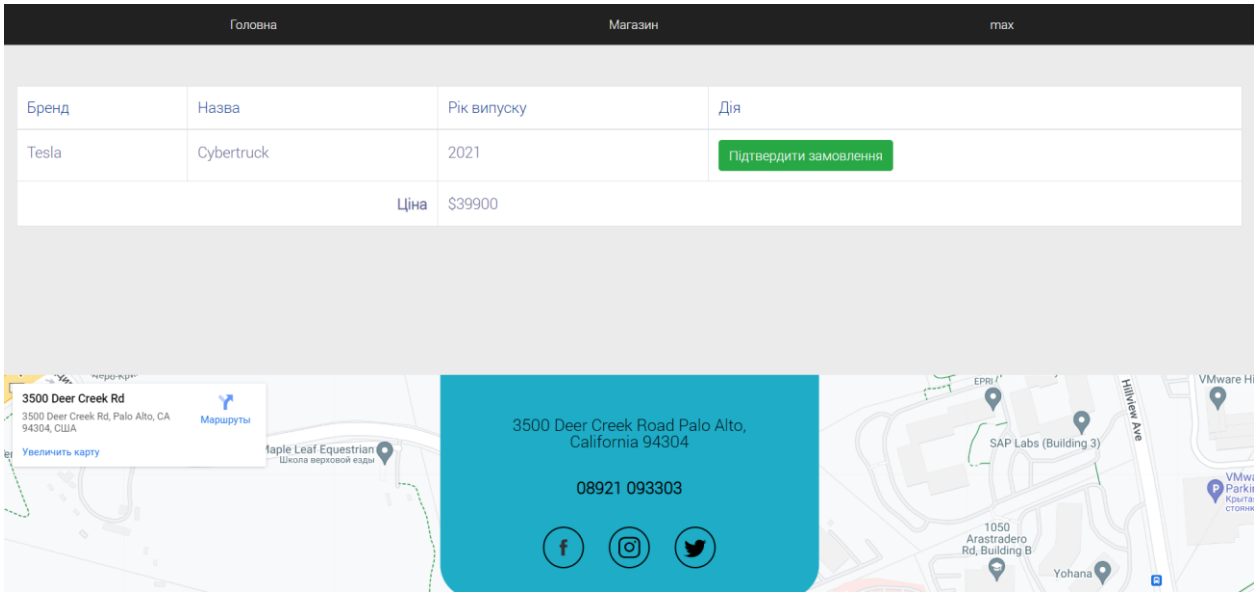

Рисунок 3.14 – Кошик

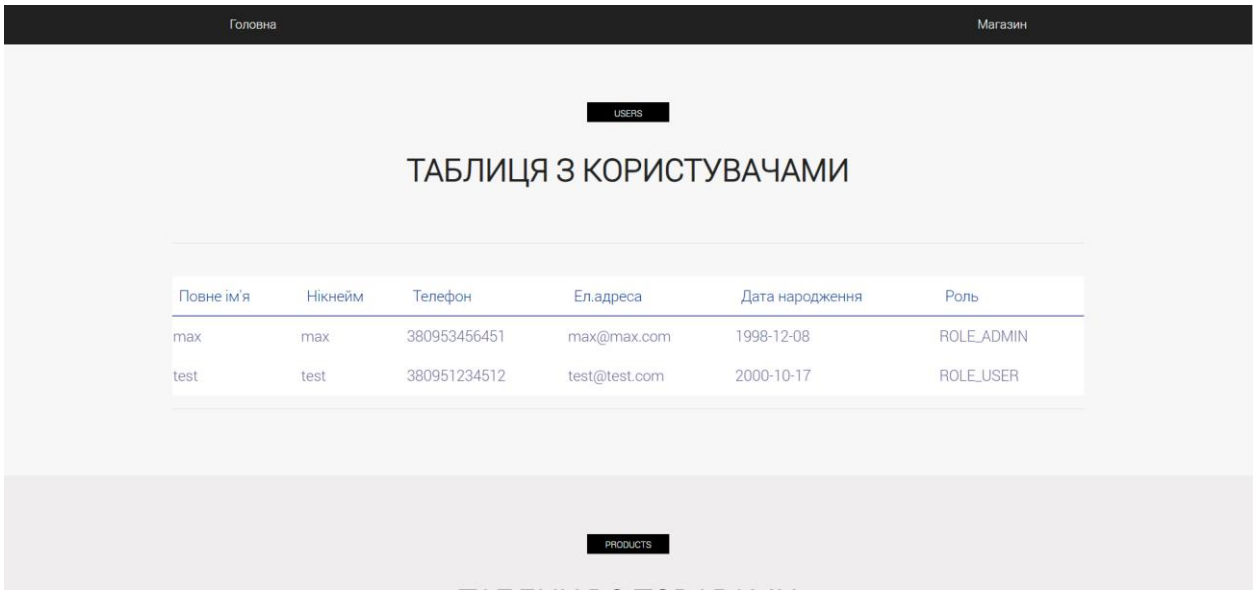

Рисунок 3.15 – Таблиця з користувачами на сторінці акаунту адміністратора

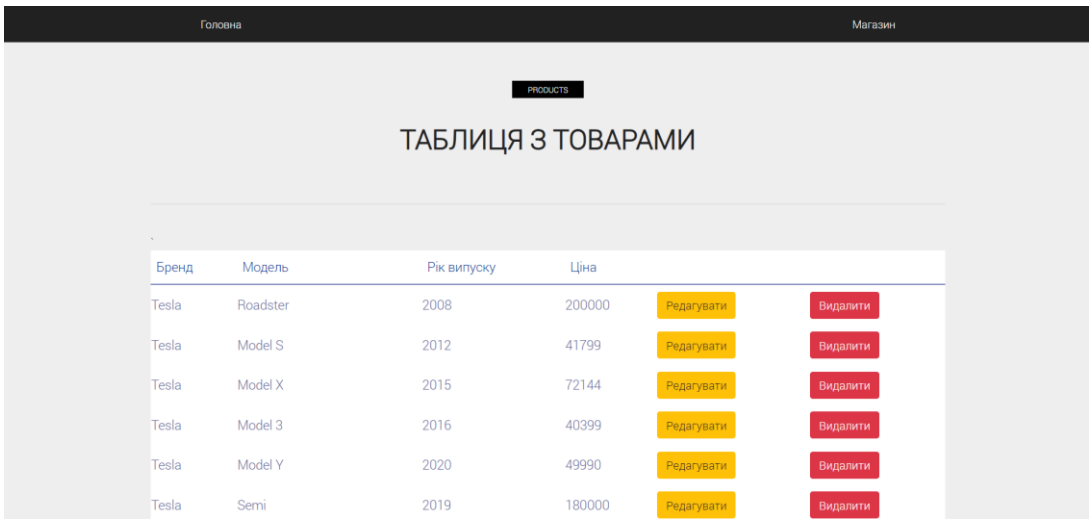

Рисунок 3.16 – Таблиця з товарами на сторінці акаунту адміністратора

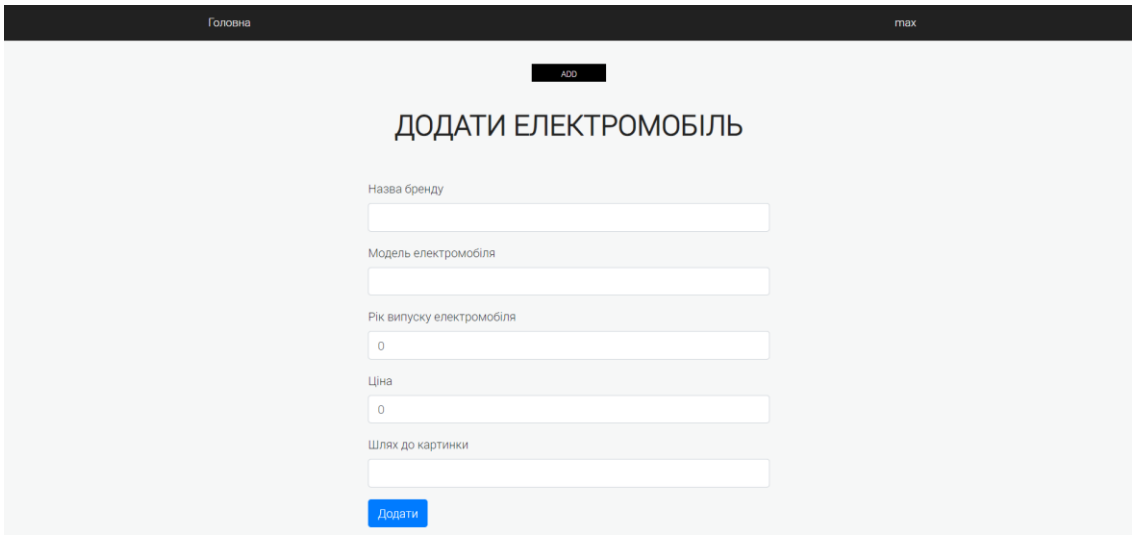

Рисунок 3.17 – Сторінка додавання нового електромобіля до системи

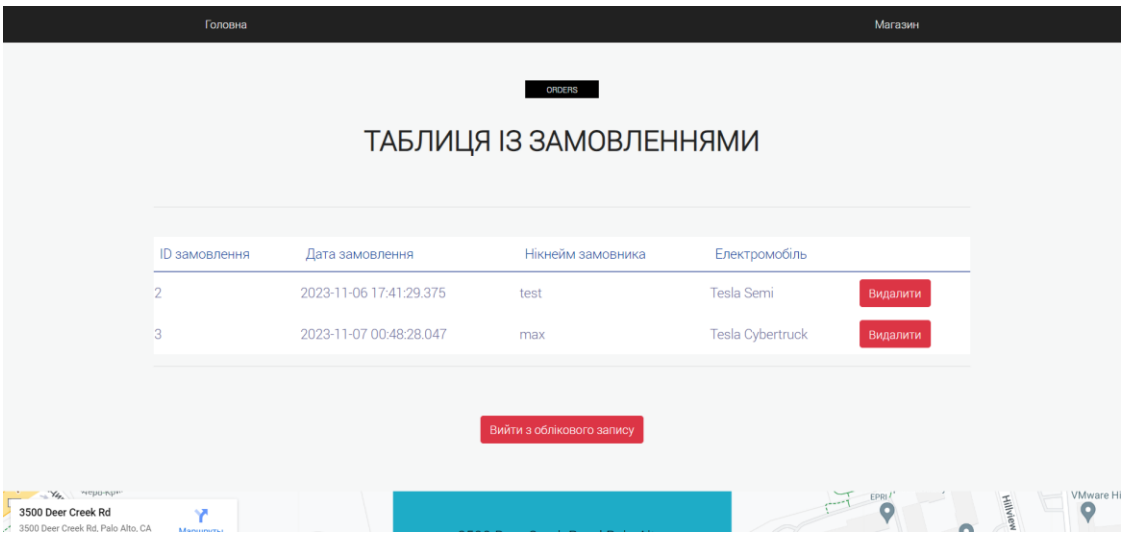

Рисунок 3.18 – Таблиця з усіма замовленнями в системі на сторінці адміністратора

3.4 Перспективи подальшої роботи

У процесі розробки вебсайту було виявлено варіанти для подальшого розвитку програми:

- додавання нових електромобілів у магазин;
- додати завантаження аватарок для користувачів;
- інтегрувати авторизацію у системі з соц. мережами;
- оптимізувати верстку для різних роздільних здатностей екрана;
- викласти сайт на реальний хостинг.

ВИСНОВКИ

В рамках кваліфікаційної роботи було змодельовано та реалізовано вебсайт з продажу електромобілів.

Під час розробки цього проєкту було засвоєно та успішно застосовано різноманітні технології для розробки клієнт-серверних додатків. Наша команда детально ознайомилася та впровадила у практику такі технології, як HTML, CSS, JavaScript і використала фреймворк BootStrap для створення вигляду та функціональності Front-end частини проєкту. Для реалізації Back-end частини були використані технології Java EE, зокрема Spring Framework, включаючи Spring Core, Spring MVC, Spring JDBC і Spring Security.

У процесі розробки також було вдосконалено знання з мови SQL та розглянуто різні системи управління реляційними базами даних. Наш вибір для проєкту впав на PostgreSQL та інструмент для адміністрування PgAdmin.

Розроблений вебзастосунок дозволяє користувачам дізнатися про сучасні електромобілі та за бажанням скористатися інтернет-магазином з продажу електромобілів.

Систему було протестовано, та за результатами тестування були виявлені варіанти та перспективи подальшої роботи над сайтом та його розвитку.

Результати дослідження апробовано у вигляді тез доповідей під час Міжнародного молодіжного форуму «New ways of creating scientific ideas for implementation» [50].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Bloch, J. (2008). Effective java. Addison-Wesley Professional.

2. Bloch, J. (2008). Effective java (the java series). Prentice Hall PTR.

3. Joy, B., Steele, G., Gosling, J., & Bracha, G. (2000). The Java language specification.

4. O'reilly, T. (2009). What is web 2.0. «O'Reilly Media, Inc.».

5. Murugesan, S. (2007). Understanding Web 2.0. IT professional, 9(4), 34-41.

6. Aghaei, S., Nematbakhsh, M. A., & Farsani, H. K. (2012). Evolution of the world wide web: From WEB 1.0 TO WEB 4.0. International Journal of Web & Semantic Technology, 3(1), 1-10.

7. Aghaei, S., Nematbakhsh, M. A., & Farsani, H. K. (2012). Evolution of the world wide web: From WEB 1.0 TO WEB 4.0. International Journal of Web & Semantic Technology, 3(1), 1-10.

8. Lyashenko, V., Mustafa, S. K., Tvoroshenko, I., & Ahmad, M. A. (2020). Methods of using fuzzy interval logic during processing of space states of complex biophysical objects.

9. Tvoroshenko, I., Ahmad, M. A., Mustafa, S. K., Lyashenko, V., & Alharbi, A. R. (2020). Modification of models intensive development ontologies by fuzzy logic.

10. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Tools for fast metric data search in structural methods for image classification. IEEE Access, 10, 124738-124746.

11. Гороховатський, В. О., & Творошенко, І. С. (2022). Аналіз багатовимірних даних за описом у формі множини компонент.

12. Ahmad, M. A., Tvoroshenko, I., Baker, J. H., & Lyashenko, V. (2019). Modeling the structure of intellectual means of decision-making using a system-oriented nfo approach.

13. Date, C. J. (1989). A Guide to the SQL Standard. Addison-Wesley Longman Publishing Co., Inc..

14. Melton, J., & Simon, A. R. (1993). Understanding the new SQL: a complete guide. Morgan Kaufmann.

15. Beaulieu, A. (2009). Learning SQL: master SQL fundamentals. « O'Reilly Media, Inc.».

16. Date, C. J. (2011). SQL and relational theory: how to write accurate SQL code. «O'Reilly Media, Inc.».

17. Walls, C. (2022). Spring in action. Simon and Schuster.

18. Mitchell, J. G., Gibbons, J. J., Hamilton, G., Kessler, P. B., Khalidi, Y. A., Kougiouris, P., ... & Radia, S. R. (1994, February). An overview of the Spring system. In Proceedings of COMPCON'94 (pp. 122-131). IEEE.

19. Johnson, R., Hoeller, J., Donald, K., Sampaleanu, C., Harrop, R., Risberg, T., ... & Webb, P. (2004). The spring framework-reference documentation. interface, 21, 27.

20. Johnson, R., Hoeller, J., Arendsen, A., & Thomas, R. (2009). Professional Java development with the Spring framework. John Wiley & Sons.

21. Bodyanskiy, Y., Shafronenko, A., & Mashtalir, S. (2020). Online robust fuzzy clustering of data with omissions using similarity measure of special type. In Lecture Notes in Computational Intelligence and Decision Making: Proceedings of the XV International Scientific Conference "Intellectual Systems of Decision Making and Problems of Computational Intelligence"(ISDMCI'2019), Ukraine, May 21–25, 2019 15 (pp. 637-646). Springer International Publishing.

22. Shafronenko, A., Dolotov, A., Bodyanskiy, Y., & Setlak, G. (2018, August). Fuzzy clustering of distorted observations based on optimal expansion using partial distances. In 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP) (pp. 327-330). IEEE.

23. Bodyanskiy, Y. V., Shafronenko, A., & Klymova, I. (2021, April). Adaptive Recovery of Distorted Data Based on Credibilistic Fuzzy Clustering Approach. In COLINS (pp. 6-15).

24. Hu, Z., Bodyanskiy, Y. V., Tyshchenko, O. K., & Shafronenko, A. (2019, July). Fuzzy clustering of incomplete data by means of similarity measures. In 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON) (pp. 957-960). IEEE.

25. Shafronenko, A., Bodyanskiy, Y. V., Klymova, I., & Holovin, O. (2020, May). Online credibilistic fuzzy clustering of data using membership functions of special type. In CMIS (pp. 744-753).

26. Tvoroshenko I., Pomazan V., Gorokhovatskyi V., and Kobylin O. (2023) Application of video data classification models using convolutional neural networks, International Journal of Academic and Applied Research, 7(11), pp. 134-145.

27. Netek, R., Brus, J., & Tomecka, O. (2019). Performance testing on marker clustering and heatmap visualization techniques: a comparative study on javascript mapping libraries. ISPRS international journal of geo-information, 8(8), 348.

28. Cherny, B. (2019). Programming TypeScript: making your JavaScript applications scale. O'Reilly Media.

29. Park, S., Xu, W., Yun, I., Jang, D., & Kim, T. (2020, May). Fuzzing javascript engines with aspect-preserving mutation. In 2020 IEEE Symposium on Security and Privacy (SP) (pp. 1629-1642). IEEE.

30. Malik, R. S., Patra, J., & Pradel, M. (2019, May). NL2Type: inferring JavaScript function types from natural language information. In 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE) (pp. 304-315). IEEE.

31. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., Gadetska S., and Al-Dhaifallah M. (2023) Statistical data analysis models for determining the relevance of structural image descriptions, IEEE Access, 11, pp. 126938-126949.

32. He, C. H., Amer, T. S., Tian, D., Abolila, A. F., & Galal, A. A. (2022). Controlling the kinematics of a spring-pendulum system using an energy

harvesting device. Journal of Low Frequency Noise, Vibration and Active Control, 41(3), 1234-1257.

33. Williams, W. C. (2019). Spring and all. Dover Publications.

34. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2022). Introduction to algorithms. MIT press.

35. Sarker, I. H. (2021). Machine learning: Algorithms, real-world applications and research directions. SN computer science, 2(3), 160.

36. Padilla, R., Netto, S. L., & Da Silva, E. A. (2020, July). A survey on performance metrics for object-detection algorithms. In 2020 international conference on systems, signals and image processing (IWSSIP) (pp. 237-242). IEEE.

37. Tsamados, A., Aggarwal, N., Cowls, J., Morley, J., Roberts, H., Taddeo, M., & Floridi, L. (2021). The ethics of algorithms: key problems and solutions. Ethics, Governance, and Policies in Artificial Intelligence, 97-123.

38. Kellogg, K. C., Valentine, M. A., & Christin, A. (2020). Algorithms at work: The new contested terrain of control. Academy of Management Annals, 14(1), 366-410.

39. Kochenderfer, M. J., & Wheeler, T. A. (2019). Algorithms for optimization. Mit Press.

40. Amoore, L. (2020). Cloud ethics: Algorithms and the attributes of ourselves and others. Duke University Press.

41. Daradkeh Y.I., Gorokhovatskyi V., Tvoroshenko I., and Zeghid M. (2022) Tools for fast metric data search in structural methods for image classification, IEEE Access, 10, pp. 124738-124746.

42. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, Advances in Electrical and Electronic Engineering, 21(1), pp. 19-27.

43. Гороховатський В.О., Творошенко І.С., Чмутов Ю.В. (2022) Застосування систем ортогональних функцій для формування простору ознак

у методах класифікації зображень, *Сучасні інформаційні системи*, 6(3), С. 5-12.

44. Гороховатський В., Передрій О., Творошенко І., Марков Т. (2023) Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, *Сучасні інформаційні системи*, 7(1), С. 5-13.

45. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Development of an application for recognizing emotions using convolutional neural networks, *International Journal of Academic Information Systems Research*, 7(7), pp. 25-36.

46. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Handwritten character recognition models based on convolutional neural networks, *International Journal of Academic Engineering Research*, 7(9), pp. 64-72.

47. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, *International Journal of Academic and Applied Research*, 7(9), pp. 57-70.

48. Gorokhovatskyi V., Tvoroshenko I. (2023) Identification of visual objects by the search request. *International scientific symposium «INTELLIGENT SOLUTIONS-S». Computational intelligence (results, problems and perspectives). Decision making theory: proceedings of the international symposium*, September 28, 2023, Kyiv-Uzhorod, Ukraine, pp. 25-27.

49. Yakovleva O., Kovač M., Ardasov V. & Yeremenko I. (2023). Study on adding functionality to the Zoom online conference system for monitoring the participant activities, *Public Administration and Regional Development*, 19(1), pp. 158-184.

50. Мітьков, М. (2023). Особливості предметної області розроблення вебсайту з продажу електромобілів.