ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Програмно-апаратна архітектура системи електронної комерції для інтернет-магазину автозапчастин»

на здобуття освітнього ступеня бакалавра
зі спеціальності 123 Комп'ютерної інженерії
(код, найменування спеціальності)
освітньо-професійної програми Комп'ютерна інженерія
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Олександр ГРИЩЕНКО
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконав: здобувач вищої освіти гр. КІД-42
Олександр ГРИЩЕНКО
Ім'я, ПРІЗВИЩЕ
Керівник: Наталія ЛАЩЕВСЬКА
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ
Рецензент:
науковий ступінь, вчене звання Ім'я, ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут Інформаційних технологій

Кафедра 123 Комп'ютерної інженерії

Ступінь вищої освіти бакалавр

Спеціальність 123 Комп'ютерної інженерії

Освітньо-професійна програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри
Комп'ютерної інженерії
_____ Наталія ЛАЩЕВСЬКА
_____
«_____» _____ 2026р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Грищенко Олександр Олександрович
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Програмно-апаратна архітектура системи електронної комерції для інтернет-магазину автозапчастин
керівник кваліфікаційної роботи Наталія ЛАЩЕВСЬКА
(Ім'я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» лютого 2026р. № 51
2. Строк подання кваліфікаційної роботи «8» травня 2026р.
3. Вихідні дані до кваліфікаційної роботи: науково-технічна література; науково-технічна література; платформа розробки Microsoft .NET Framework 4.7.2; мова програмування C#; технологія WPF (Windows Presentation Foundation); ORM-бібліотека Entity Framework 6.4.4; СУБД Microsoft SQL Server 2019; архітектурний патерн MVVM; принципи побудови систем електронної комерції; технології управління каталогом товарів та замовленнями.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Теоретичні основи систем електронної комерції та інтернет-торгівлі
2. Проєктування програмно-апаратної архітектури системи інтернет-магазину автозапчастин
3. Розробка застосунку інтернет-магазину автозапчастин
4. Тестування та верифікація розробленої системи
5. Перелік ілюстративного матеріалу: *презентація*
6. Дата видачі завдання «20» лютого 2026р

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Збір даних	25.02-10.03.2026	Виконано
2	Аналіз літератури та сучасних технологій електронної комерції	11.03-20.03.2026	Виконано
3	Дослідження принципів побудови програмно-апаратної архітектури системи	21.03-05.04.2026	Виконано
4	Проєктування архітектури застосунку інтернет-магазину автозапчастин	06.04-20.04.2026	Виконано
5	Розробка та тестування застосунку інтернет-магазину автозапчастин	21.04-05.05.2026	Виконано
6	Оформлення пояснювальної записки	06.05-12.05.2026	Виконано
7	Розробка презентації	13.05-20.05.2026	Виконано
8	Передзахист	21.05-29.05.2026	Виконано
9	Здача в деканат	02.06-05.06.2026	Виконано

7. Здобувач вищої освіти _____ Олександр ГРИЩЕНКО
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

8. Керівник кваліфікаційної роботи _____ Наталія ЛАЩЕВСЬКА
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 98 сторінок, 73 рисунків, 3 таблиці, 30 джерела, 3 додатки.

Об’єкт дослідження – розробка застосунку для інтернет-магазину автозапчастин.

Мета роботи – аналіз існуючих застосунків на тему інтернет-магазину автозапчастин, та на основі аналізу розробити застосунок для інтернет-магазину автозапчастин.

Метод дослідження – аналіз сучасних інтернет ресурсів, спрямованих на тему інтернет-магазину та купівлі автозапчастин.

У кваліфікаційній роботі розглянуто роботу сучасних застосунків, спрямованих на тему інтернет-магазину автозапчастин. Також розглянуті існуючі рішення інтернет-магазинів автозапчастин. Спроектовано застосунок з реалізацією інтернет-магазину автозапчастин.

Результатом роботи є спроектований застосунок інтернет-магазину автозапчастин.

Результати кваліфікаційної роботи можна використовувати для подальшої розробки застосунку для інтернет-магазину автозапчастин.

ЗАСТОСУНОК, БАЗА ДАНИХ, ІНТЕРНЕТ-МАГАЗИН, АВТОЗАПЧАСТИНИ, ЗАМОВЛЕННЯ.

ABSTRACT

The explanatory note to the diploma abstract: 98 pages, 73 figures, 3 tables, 30 sources, 3 appendices.

The object of the research is the development of an application for an online auto parts store.

The purpose of the work is to analyze existing applications on the topic of an online auto parts store, and based on the analysis to develop an application for an online auto parts store.

The research method is the analysis of modern Internet resources aimed at the topic of an online store and the purchase of auto parts.

In the qualification work, the work of modern applications aimed at the topic of the online auto parts store is considered. Existing solutions of auto parts online stores are also considered. An application with the implementation of an online auto parts store has been designed.

The result of the work is the designed application of the auto parts online store.

The results of the qualification work can be used for further development of the application for the online auto parts store.

APP, DATABASE, ONLINE STORE, AUTO PARTS, ORDER.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД — база даних;

ІТ — Інформаційні технології;

ІКТ — інфокомунікаційні технології;

СУБД — система управління базами даних;

DOM — Document Object Model, об'єктна модель документа;

ECTS — European Credit Transfer and Accumulation System, Європейська система перекладу та накопичення балів;

URL — Uniform Resource Locator, уніфікований покажчик ресурсу;

WPF — Інтерфейс Windows Presentation Foundation

MsSQL — Microsoft SQL

MVVM — Архітектура MODEL-VIEW-VIEWMODEL

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 6
ВСТУП 9
1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ 11
1.1 Огляд та аналіз існуючих платформ електронної комерції для автозапчастин 11
1.2 Комплексне порівняння архітектурних рішень існуючих платформ 16
1.3 Постановка задачі дослідження 18
Висновок до розділу 19
2 ПРОЕКТУВАННЯ СИСТЕМИ ІНТЕРНЕТ-МАГАЗИНУ АВТОЗАПЧАСТИН 20
2.1 Засоби розробки та мови програмування 20
2.2 Інструментальні засоби розробки 21
2.3 Алгоритми функціонування системи 22
2.4 UML-діаграма варіантів використання 25
2.5 Діаграма послідовності дій 27
Висновок до розділу 28
3 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ ІНТЕРНЕТ-МАГАЗИНУ 29
3.1 Архітектурне рішення застосунку 29
3.2 Опис структури та елементів застосунку 31
3.3 Реалізація інтерфейсу застосунку 33
3.4 Опис функціонування застосунку 42
Висновок до розділу 50
4 ТЕСТУВАННЯ ТА ВЕРИФІКАЦІЯ ЗАСТОСУНКУ 51
4.1 Характеристика методів, що застосовуються при перевірці застосунку 51
a. Поняття верифікації та валідації програмного забезпечення 51
b. Тестування застосунку 56
Висновок до розділу 60
ВИСНОВКИ 61
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 62

ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ 65
ДОДАТОК Б. ІНСТРУКЦІЯ ДЛЯ КОРИСТУВАЧА 70
ДОДАТОК В. ВИХІДНИЙ КОД ПРОГРАМИ 87
ДОДАТОК Г. ПРЕЗЕНТАЦІЯ 93

ВСТУП

Стрімкий розвиток інформаційних технологій (ІТ) суттєво змінює підходи до організації торгівлі та надання послуг. Електронна комерція перетворилася на один із ключових напрямів цифрової економіки, охоплюючи дедалі більшу кількість галузей — від роздрібної торгівлі до спеціалізованих ринків. Однією з таких галузей є ринок автомобільних запчастин, де онлайн-торгівля набуває особливого значення.

Сьогодні автомобілі є невід'ємною частиною повсякденного життя мільйонів людей. Зростання кількості транспортних засобів закономірно породжує постійний попит на автомобільні запчастини та аксесуари. Разом із тим сучасні власники автомобілів дедалі частіше обирають онлайн-канали для пошуку та придбання необхідних деталей — це зручніше, швидше та нерідко дешевше, ніж звернення до традиційних офлайн-магазинів.

Побудова ефективної системи електронної комерції для інтернет-магазину автозапчастин пов'язана з вирішенням низки технічних та архітектурних завдань, зокрема:

- проектування масштабованої програмно-апаратної архітектури;
- організація надійного зберігання та обробки даних про товари, замовлення та користувачів;
- забезпечення безпеки транзакцій та захисту персональних даних;
- інтеграція клієнтської та серверної частин системи з підтримкою сучасних веб-стандартів.

Вирішення цих завдань потребує комплексного підходу — від вибору відповідних технологій і фреймворків до проектування архітектури серверної та клієнтської частин, організації бази даних і вибору апаратної інфраструктури. Саме тому тема програмно-апаратної архітектури системи електронної комерції є актуальною та науково значущою.

Ринок автозапчастин в Україні та світі характеризується високою конкуренцією та специфічними вимогами до каталогізації товарів: кожна деталь має відповідати конкретній марці, моделі та року випуску автомобіля. Це ускладнює організацію бази даних і пошуку, що висуває особливі вимоги до архітектурних рішень. Онлайн-платформи для продажу автозапчастин повинні забезпечувати не лише зручний інтерфейс, а й складну логіку підбору деталей за VIN-кодом або параметрами автомобіля.

Ринок онлайн-торгівлі автозапчастинами з кожним роком зростає. Автовласники дедалі частіше воліють купувати запчастини онлайн завдяки широкому асортименту, зручному пошуку за параметрами та можливості доставки. Це стимулює розвиток спеціалізованих платформ електронної комерції з розширеним функціоналом.

У зв’язку з цим темою кваліфікаційної роботи обрано проектування програмно-апаратної архітектури системи електронної комерції для інтернет-магазину автозапчастин — рішення, що охоплює як серверну та клієнтську програмну частину, так і апаратну інфраструктуру, що забезпечує його функціонування.

1 ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ТА ОБҐРУНТУВАННЯ ВИБОРУ ТЕХНОЛОГІЙ

1.1 Огляд та аналіз існуючих платформ електронної комерції для автозапчастин

На сьогоднішній день існує чимало платформ електронної комерції для продажу автозапчастин — як вузькоспеціалізованих, так і універсальних. Розглянемо та проаналізуємо найбільш відомі з них з точки зору програмно-апаратної архітектури та функціональності.

Українська платформа електронної комерції автозапчастин

Exist.ua — найбільший спеціалізований маркетплейс автозапчастин в Україні. З архітектурної точки зору платформа реалізована як багаторівнева веб-система з серверним рендерингом, мікросервісним бекендом та реляційною СУБД.
Станом на 2023 рік, Exist.ua є найбільш відвідуваним спеціалізованим сайтом з продажу автозапчастин в Україні [1].

Exist.ua не зберігає власних товарних залишків — постачальники публікують залишки через API. Продаж реалізований за принципом маркетплейсу: замовлення прив’язується до конкретного постачальника з фіксованою ціною та строком доставки (рисунок 1.1).

Переваги:

- мікросервісна архітектура з високою масштабованістю
- API-інтеграція з постачальниками

Недоліки:

- складна підтримка та оновлення монолітної частини системи
- відсутність мобільного застосунку
- висока навантаженість серверної частини у пікові години

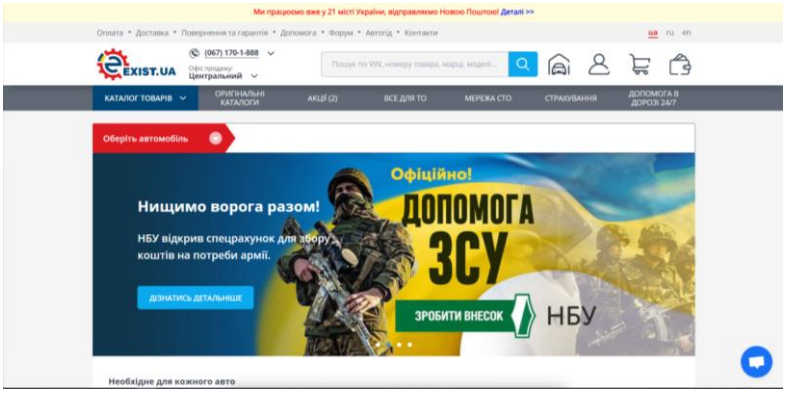


Рисунок 1.1 – Інтерфейс веб-сайту «Exist.ua»

Українська платформа агрегації пропозицій автозапчастин

Autoklad.ua — спеціалізований інтернет-магазин автозапчастин з власним складом та REST API-інтеграціями – один із провідних інтернет-магазинів автозапчастин в Україні. Архітектура системи орієнтована на зберігання власних товарних залишків з підтримкою логістики та доставки. (рис. 1.2) [2].

Переваги:

– наявність власного складу з реальними залишками товарів

– власна логістика та служба доставки

Недоліки:

– монолітна архітектура з обмеженими можливостями масштабування

– відсутність мобільного застосунку та API для партнерів

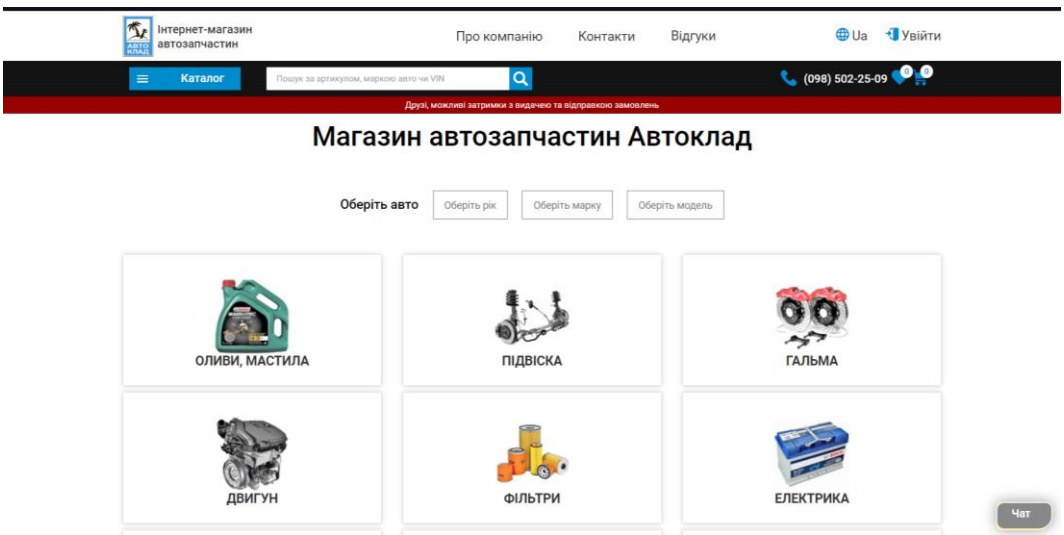


Рисунок 1.2 – Інтерфейс веб-сайту «Autoklad.ua»

Український інтернет-магазин який спеціалізується на продажі автозапчастин

Avto.pro — агрегатор пропозицій автозапчастин для іномарок та вітчизняних автомобілів. Архітектурно платформа реалізована як пошуковий агрегатор з централізованим каталогом та розподіленими вузлами даних постачальників, інтегрованими через RESTful API. платформу для взаємодії, яка дозволить вивести ринок на якісно кращий рівень, оптимізуючи процеси, надаючи інформацію та послуги всім учасникам ринку [3].

Принцип роботи платформи: Покупці знаходять найкращі пропозиції серед тисяч продавців та укладають угоди напряму — без посередницької комісії платформи.

Продавці отримують платформу для розміщення своїх пропозицій та ознайомленого з умовами покупця, що скорочує операційні витрати на залучення клієнтів. (рис. 1.3).

Переваги:

– агрегація пропозицій від багатьох продавців

– зручний пошук та порівняння цін

Недоліки:

– застарілий інтерфейс

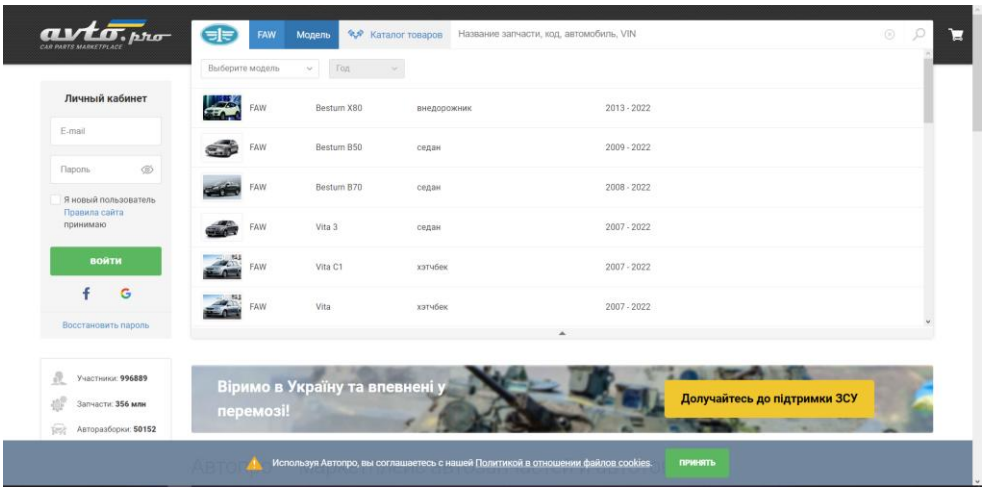


Рисунок 1.3 – Інтерфейс веб-сайту «avto.pro»

Європейська платформа електронної комерції оригінальних автозапчастин

Mapco.com — великий європейський B2B та B2C маркетплейс оригінальних запчастин для автомобілів. Платформа побудована на мікросервісній архітектурі з підтримкою пошуку за VIN-кодом та каталогізатором OEM-номерів деталей. (рис. 1.4) [4].

Переваги:

– зручний пошук

– підтримка пошуку за VIN-кодом та OEM-номерами

Недоліки:

– застарілий інтерфейс

– орієнтованість переважно на європейський ринок

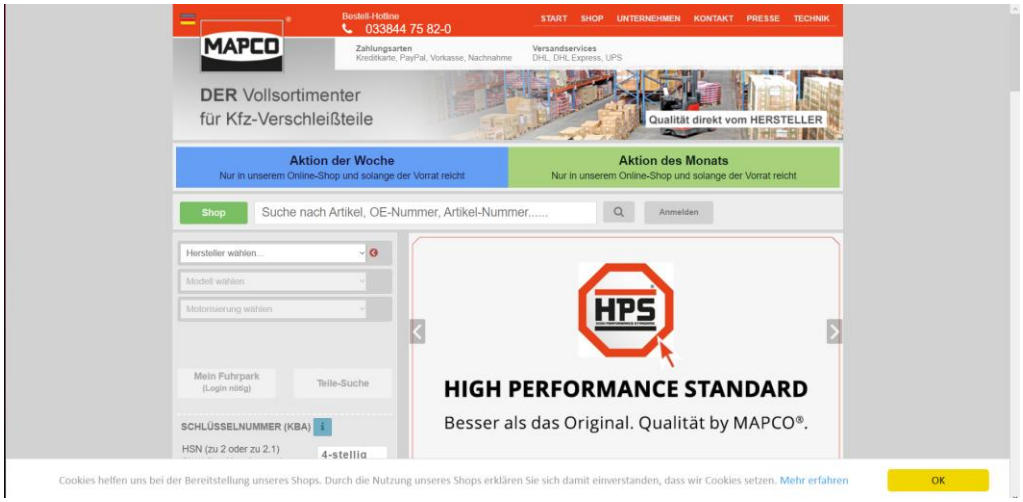


Рисунок 1.4 – Інтерфейс веб-сайту « mapco.com »

Японська універсальна платформа електронної комерції з розділом автозапчастин

rakuten.com — великий японський e-commerce маркетплейс, що реалізує запчастини для автомобілів. Платформа базується на хмарній інфраструктурі AWS з мікросервісною архітектурою та підтримкою великого навантаження. (рис. 1.5) [5].

Переваги:

– зручний пошук

– хмарна інфраструктура AWS з високою доступністю

Недоліки:

– застарілий інтерфейс

– інтерфейс та документація переважно японською мовою

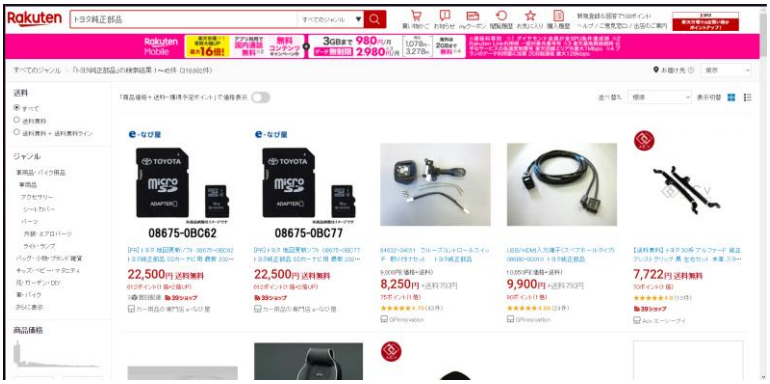


Рисунок 1.5 – Інтерфейс веб-сайту « rakuten.com»

Американський спеціалізований маркетплейс автозапчастин

Imsautoparts.com — великий американський спеціалізований маркетплейс запчастин. Побудований на n-рівневій архітектурі з використанням хмарного хостингу. Підтримує каталог з мільйонами позицій та розширений пошук за OEM-номерами.ілів (рис. 1.5) [6].

Переваги:

– зручний пошук

– зручний фільтр товарів

Недоліки:

– обмежена географія доставки (переважно США)

– відсутність локалізації та підтримки міжнародних платежів



Рисунок 1.6 – Інтерфейс веб-сайту « imsautoparts.com »

1.2 Комплексне порівняння архітектурних рішень існуючих платформ

Провівши аналіз кожної платформи з точки зору програмно-апаратної архітектури, проведемо порівняльну оцінку за ключовими технічними та функціональними критеріями.

Визначено показники та форму оцінки кожного з них (табл. № 1.1).

При порівнянні виникає проблема стандартизації, оскільки критерії оцінювання архітектурних рішень можуть відрізнятися залежно від регіону та вимог ринку.

Платформи електронної комерції автозапчастин обслуговують різні групи користувачів — від кінцевих покупців до B2B-компаній. Архітектурні рішення таких систем часто є транснаціональними.

Визначено 4 основних критерії оцінювання архітектурних рішень платформ за шкалою, аналогічною до Європейської системи перекладу та накопичення балів (ECTS) [14]:

- A – функція відмінно працює;
- B – функція працює вище середнього, але з деякими недоліками;
- C – функція працює, але не відповідає поставленій задачі;
- D – функція не відповідає стандарту та поставленій задачі, взагалі не працює або не має.

Таблиця № 1.1 – Порівняння архітектурних рішень платформ електронної комерції автозапчастин

Показники	Платформи					
	Exist.ua	autoklad.ua	avto.pro	mapco.com	rakuten.co	imsautoparts.com
Масштабованість архітектури	B	B	A	A	A	C
Якість API та інтеграцій	B	B	A	B	B	B
Безпека та захист даних	D	A	A	D	D	D
Мобільна підтримка	B	B	B	D	D	C
Пошук та фільтрація каталогу	B	B	A	A	A	B
Якість бази даних	A	B	A	A	B	C
Авторизація та автентифікація	A	B	A	B	A	B
Хмарна/серверна інфраструктура	D	A	D	D	A	D
Швидкодія та продуктивність	A	A	B	B	B	A
Технічна документація та підтримка	A	D	D	B	A	B
Надійність та відмовостійкість	B	A	A	A	A	B
Загальна оцінка архітектурного рішення	B	C	B	B	B	C

1.3 Постановка задачі дослідження

Проаналізувавши існуючі платформи та їхні архітектурні підходи, було визначено основне завдання та вимоги до проектованої системи.

Основним завданням кваліфікаційної роботи є проектування програмно-апаратної архітектури системи електронної комерції для інтернет-магазину автозапчастин. Система повинна забезпечувати авторизацію та реєстрацію користувачів, управління каталогом запчастин, оформлення замовлень. Адміністратору системи надається можливість додавати та редагувати товари у каталозі, управляти постачальниками та переглядати замовлення.

Клієнт повинен мати можливість переглядати каталог запчастин з фільтрацією за параметрами автомобіля, додавати товари до кошика та оформляти замовлення з відстеженням статусу.

Система має забезпечити єдиний цифровий майданчик для замовлення автозапчастин з відстеженням статусу замовлення. Адміністратори отримують повний функціонал управління каталогом: додавання, редагування та видалення товарів і постачальників, перегляд та управління замовленнями.

Для успішного виконання кваліфікаційної роботи необхідно:

- провести аналіз існуючих архітектурних рішень систем електронної комерції;
- дослідити сучасні технологічні стеки та інфраструктурні рішення для веб-платформ;
- визначити функціональні та нефункціональні вимоги до системи;
- спроектувати архітектуру програмної та апаратної частин системи;
- розробити структуру бази даних та схему взаємодії компонентів;
- реалізувати серверну та клієнтську частини системи;
- провести тестування і верифікацію розробленої системи;
- підготувати технічну документацію та керівництво користувача.

Висновок до розділу

Огляд існуючих платформ виявив ключові підходи до побудови архітектури систем електронної комерції для автозапчастин.

Аналіз підтвердив актуальність обраної теми та визначив архітектурні вимоги до проектованої системи.

Спроектована система повинна поєднувати надійну серверну інфраструктуру, ефективну базу даних з підтримкою каталогізації запчастин за параметрами автомобіля та зручний клієнтський веб-інтерфейс.

2 ПРОЕКТУВАННЯ СИСТЕМИ ІНТЕРНЕТ-МАГАЗИНУ АВТОЗАПЧАСТИН

2.1 Засоби розробки та мови програмування

У процесі створення програмного забезпечення застосовувались такі інструменти та технології:

- Середовище розробки Microsoft Visual Studio 2019;
- Платформа розробки .NET Framework версії 4.7.2;
- Об’єктно-орієнтована мова програмування C#;
- Декларативна мова розмітки XAML для опису інтерфейсу;
- Технологія побудови графічного інтерфейсу WPF;
- ORM-бібліотека Entity Framework версії 6.4.4;
- Система управління базами даних Microsoft SQL Server 2019.

Станом на сьогодні у світі налічується близько десяти тисяч мов програмування. Кожна з них має свою сферу застосування: одні орієнтовані на розробку ігор, інші — на медичне або промислове програмне забезпечення. C# — це об’єктно-орієнтована мова програмування з підтримкою компонентного підходу. Вона надає розробнику зручні мовні конструкції для роботи з компонентами та забезпечує підтримку сучасних парадигм розробки ПЗ. Мова постійно вдосконалюється і набуває нових можливостей, що відповідають актуальним вимогам індустрії. Сьогодні C# входить до числа найпоширеніших мов програмування у світі та широко застосовується для розробки настільних застосунків, складних веб-серверів і мобільних рішень. Виникнувши як частина екосистеми Microsoft .NET, ця мова з часом здобула визнання серед розробників по всьому світу.

Серед доступних технологій для побудови настільних застосунків варто виділити WF та WPF. Для реалізації графічного інтерфейсу обрано Windows Presentation Foundation (WPF) — розширений API-фреймворк, призначений для

побудови настільних програм із візуально насиченим користувацьким інтерфейсом. Графічна підсистема WPF побудована на базі DirectX, що дозволяє задіяти апаратне прискорення відеокарти.

Саме це є ключовою відмінністю WPF від попередньої технології Windows Forms, яка не використовувала DirectX для рендерингу елементів інтерфейсу.

Програмування логіки застосунку в середовищі WPF здійснювалося мовою C# — об’єктно-орієнтованою мовою з C-подібним синтаксисом, що є невід’ємною складовою платформи Microsoft .NET Framework.

Для організації взаємодії між застосунком на C# та базою даних використовується проміжний рівень — Entity Framework, об’єктно-реляційний відображувач (ORM) для платформи .NET Framework, що перетворює операції з об’єктами коду на відповідні SQL-запити до БД.

Entity Framework забезпечує виконання запитів до БД через LINQ to Entities. Основною одиницею роботи є сутність (entity) — об’єкт, що відповідає певному запису предметної галузі. Завдяки цьому розробник оперує не таблицями, а типізованими об’єктами та їх колекціями, що суттєво спрощує код доступу до даних.

2.2 Інструментальні засоби розробки

Основним середовищем розробки для даного проєкту обрано Microsoft Visual Studio — найбільш зручний та функціональний інструмент для написання WPF-застосунків.

Роль системи управління базами даних (СУБД) виконує Microsoft SQL Server 2019. Серед розглянутих альтернатив — Oracle СУБД з розширеними можливостями обробки великих масивів даних, кросплатформна MySQL та MongoDB з підтримкою нереляційних сховищ. Перевагою Microsoft SQL Server є відсутність необхідності попередньої реєстрації для повноцінної роботи, наявність зручного графічного інтерфейсу і вбудованих інструментів генерації скриптів

міграції, що загалом забезпечує стабільну продуктивність при роботі з базою даних застосунку [23].

2.3 Алгоритми функціонування системи

Незважаючи на розмаїття існуючих архітектурних підходів, у даній роботі були спроєктовані власні алгоритми роботи системи. Наведені нижче схеми описують ключові функції застосунку на концептуальному рівні, без прив'язки до конкретних деталей реалізації та кодових конструкцій (зображено на рис. 2.1–2.4).

Запуск застосунку відбувається шляхом відкриття виконуваного файлу з розширенням .exe, після чого подальша робота системи повністю визначається діями користувача. Насамперед розглянемо алгоритм роботи головної сторінки застосунку (зображено на рис. 2.1).

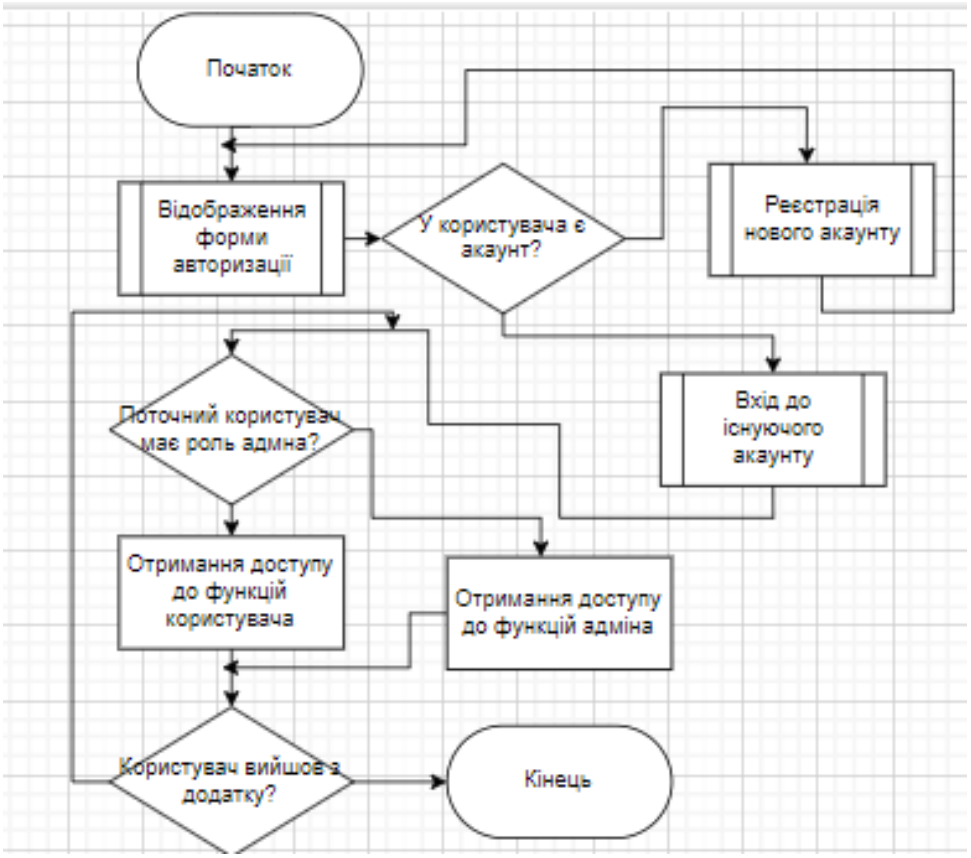


Рисунок 2.1 – Алгоритм роботи основної сторінки системи

Після запуску застосунку користувач опиняється на формі авторизації. Якщо облікового запису ще немає, передбачена можливість реєстрації — для цього

достатньо натиснути відповідну кнопку, що відкриє форму створення нового акаунту. Подальший сценарій роботи системи залежить від вибору користувача, тому окремі гілки алгоритму розглядаються далі.

Однією з основних функцій системи, що є доступною для всіх ролей користувачів, виступає можливість додавання товару до кошика. Алгоритм роботи відповідного модуля описано далі (зображено на рис. 2.2).

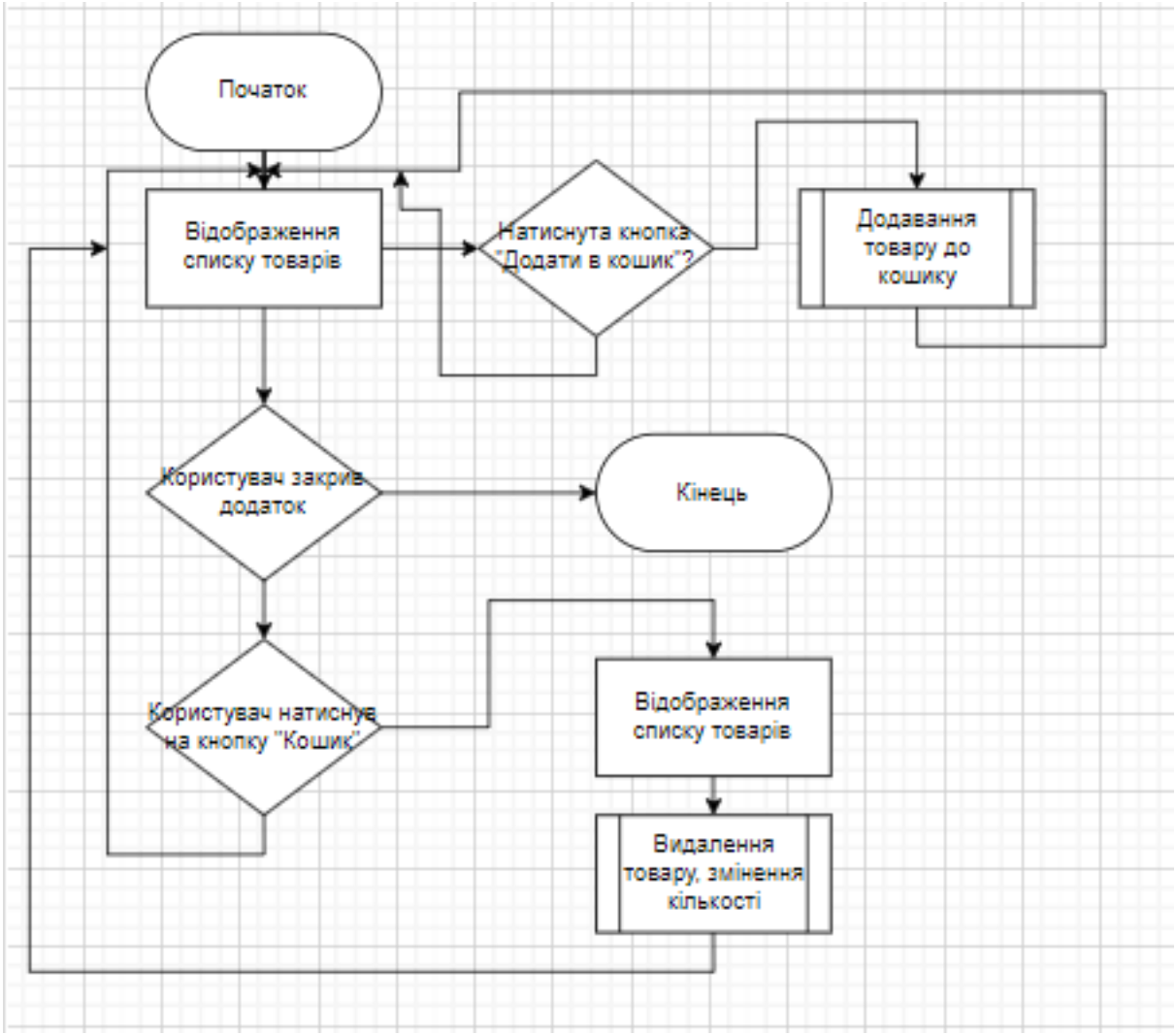


Рисунок 2.2 – Алгоритм роботи системи відображення списку товарів, та товарів, які було додано в кошик

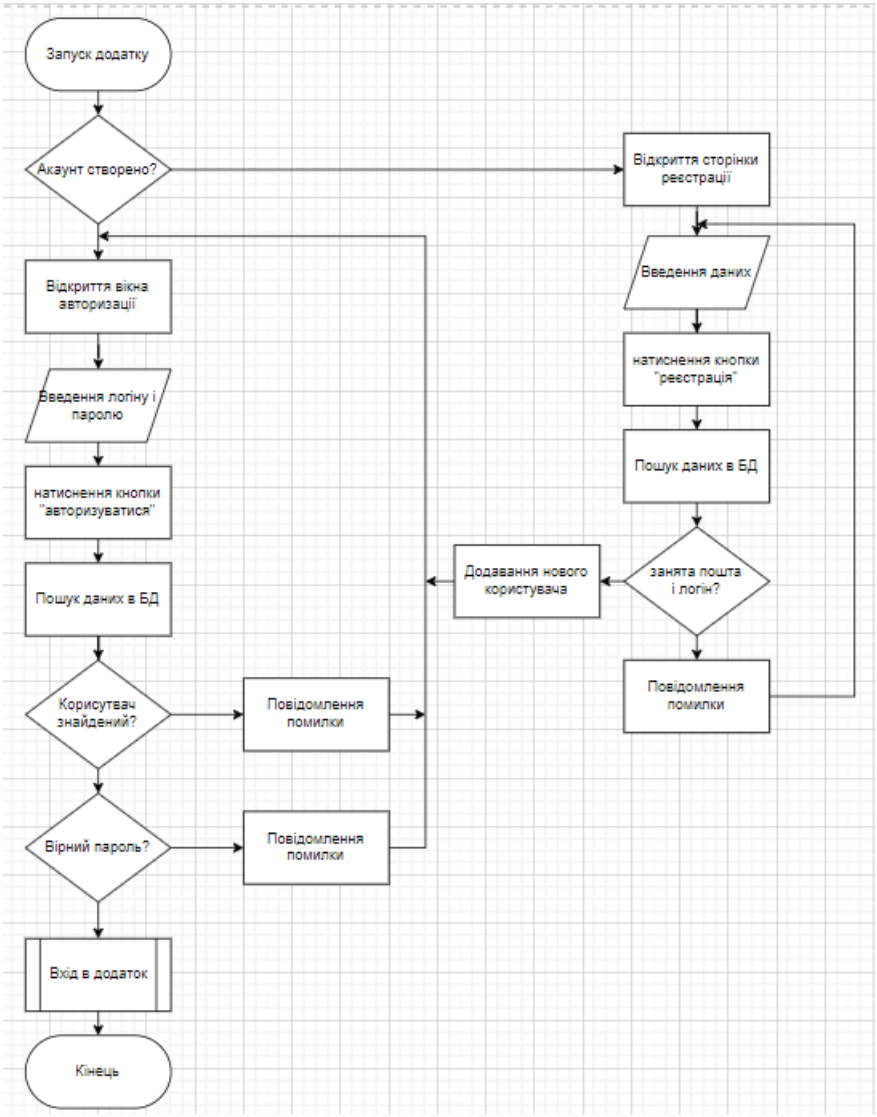


Рисунок 2.3 – Алгоритм роботи системи авторизації користувача

У межах цього алгоритму користувач вводить логін і пароль у відповідні поля форми авторизації. Введені значення проходять перевірку коректності, і за умови їх валідності передаються до бази даних для порівняння зі збереженими обліковими даними. Якщо відповідний запис знайдено — користувач успішно входить до свого акаунту, отримує доступ до персонального функціоналу і бачить підтверджувальне повідомлення.

Відповідальність за наповнення та редагування даних у системі покладено на адміністратора. Нижче розглядається алгоритм роботи адміністративного модуля (зображено на рис. 2.4).

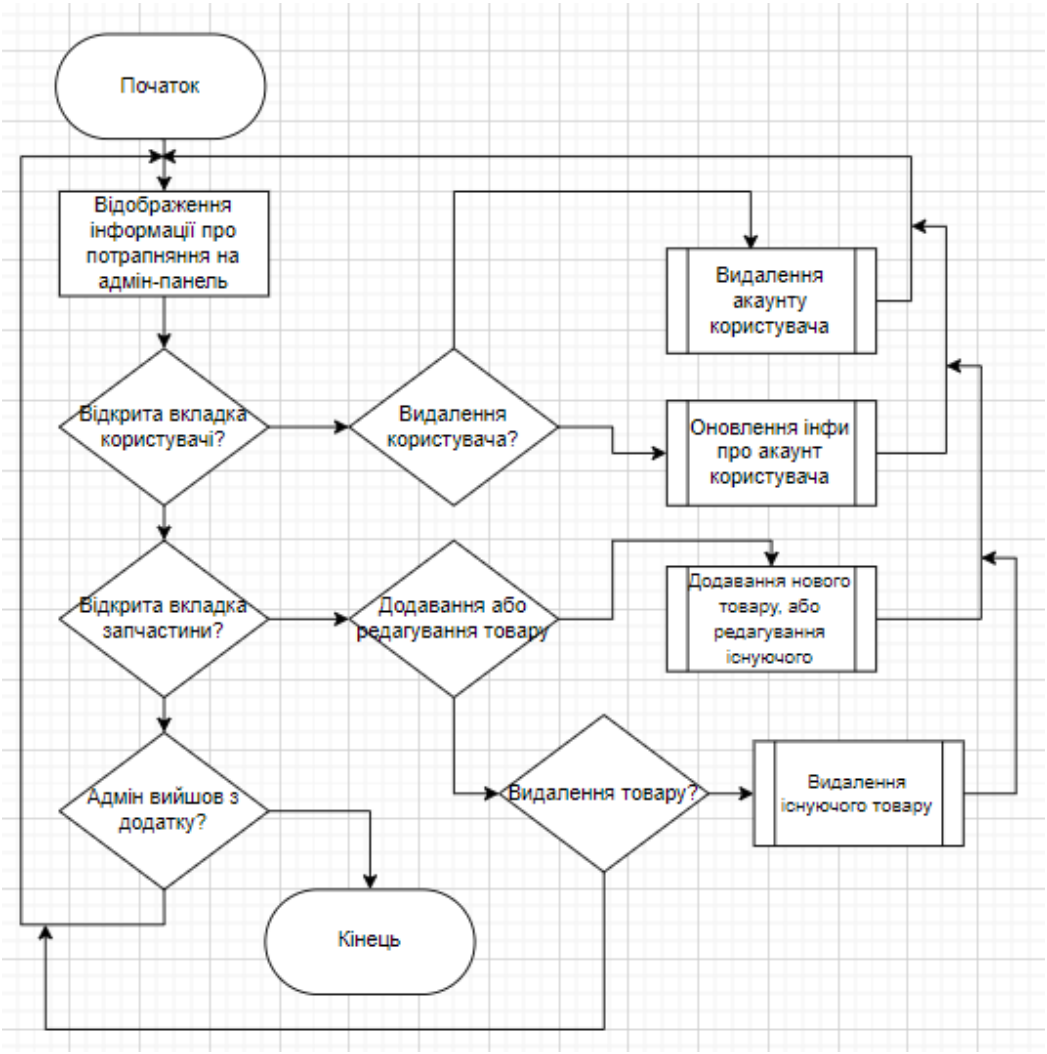


Рисунок 2.4 – Алгоритм роботи системи адміністраторських функцій

Початком роботи алгоритму є відображення системного повідомлення про вхід з правами адміністратора. Подальші кроки визначаються його діями: адміністратор може перейти до управління обліковими записами — видалити запис або відредагувати його дані зі збереженням змін. Також доступний розділ управління товарним каталогом, де можна додавати нові позиції, оновлювати наявні відомості про товари або видаляти застарілі записи.

2.4 UML-діаграма варіантів використання

Аналіз функціоналу застосунку дозволяє виокремити дві основні ролі — це безпосередньо користувачі системи з можливістю авторизації у власний обліковий

запис, та адміністратор, який здійснює модерацію і відповідає за наповнення системи актуальною інформацією (зображено на рис. 2.5).

Користувач може увійти до наявного облікового запису або зареєструватися для створення нового. Після авторизації зареєстрований користувач отримує доступ до розширеного функціоналу: перегляду та управління кошиком, оформлення замовлень, вибору способу доставки, а також редагування власних персональних даних.

При застосуванні фільтрів у каталозі користувач може одночасно задати кілька критеріїв пошуку, після чого система сформує відповідний перелік автозапчастин.

Після введення адміністратором даних нового товару запис зберігається у базі даних, а позиція одразу стає видимою у загальному каталозі.

Адміністратор системи має доступ до управління обліковими записами: може переглядати дані користувачів, вносити до них зміни або повністю видаляти записи у разі необхідності.

Крім того, адміністратор контролює актуальність контенту в системі та має можливість редагувати дані про товари, категорії, варіанти доставки та облікові записи користувачів.

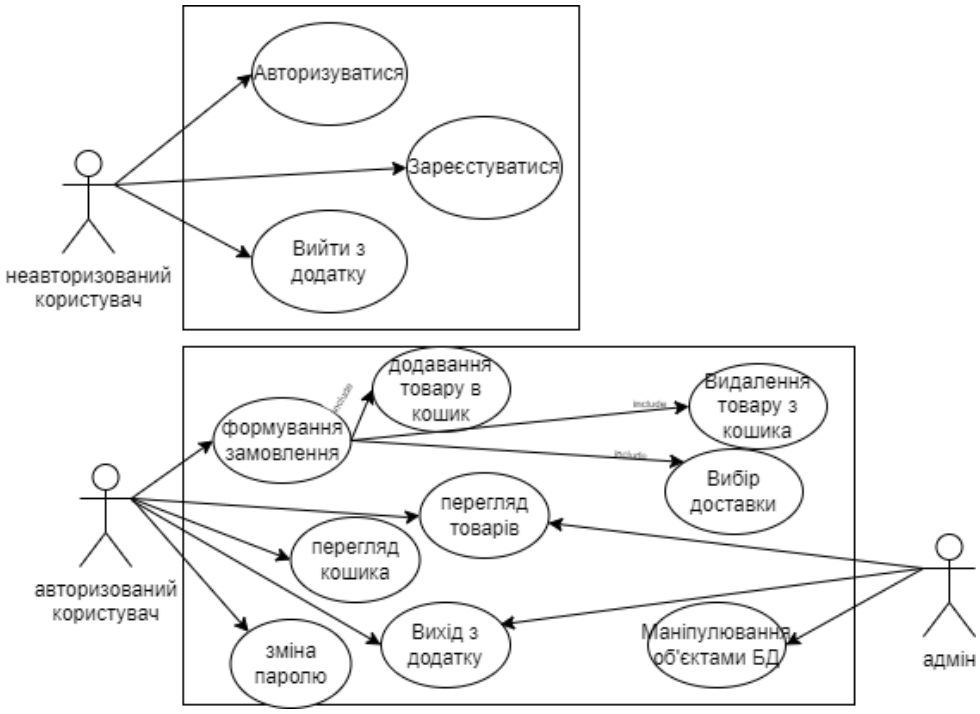


Рисунок 2.5 – UML діаграма використання системи

Таблиця № 2.1 – Опис варіантів використання для незареєстрованого(а) і зареєстрованого(б) користувача

а)

Назва прецеденту	Опис
Авторизація в системі	Надає можливість увійти до наявного облікового запису
Реєстрація нового акаунту	Дозволяє створити новий обліковий запис у системі.
Завершення роботи із застосунком	Забезпечує коректне закриття застосунку

б)

Назва прецеденту	Опис
Перегляд вмісту кошика	Дає змогу переглянути перелік товарів, доданих до кошика
Оформлення замовлення	Дозволяє оформити замовлення на обрані товари
Додавання товару до кошика	Дозволяє обрати позицію з каталогу та перемістити її до кошика
Видалення позиції з кошика	Надає змогу прибрати непотрібний товар із кошика
Вибір способу доставки	Дозволяє обрати зручний спосіб отримання замовлення
Зміна пароля акаунту	Дозволяє оновити пароль облікового запису
Перегляд каталогу товарів	Надає доступ до перегляду каталогу автозапчастин
Управління об’єктами бази даних	Надає адміністратору змогу додавати, редагувати та видаляти записи у базі даних.

2.5 Діаграма послідовності дій

На діаграмі послідовності дій (зображено на рис. 2.6) відображено взаємодію користувача із застосуном: користувач взаємодіє з елементами керування інтерфейсу застосунку, а БД зберігає відповідні дані. Коли користувач активує функцію «Фільтрація товарів», система формує вибірку з бази даних згідно з обраними параметрами і виводить результат в інтерфейсі застосунку. Під час авторизації користувач вводить облікові дані через інтерфейс, після чого вони

перевіряються по базі даних і користувач отримує повідомлення про успішний вхід і здобуває доступ до розширеного функціоналу. Коли адміністратор обирає операцію “Робота з товарами” та вносить дані нової позиції, ця інформація зберігається у базі даних, після чого оновлений список товарів відображається в інтерфейсі застосунку.

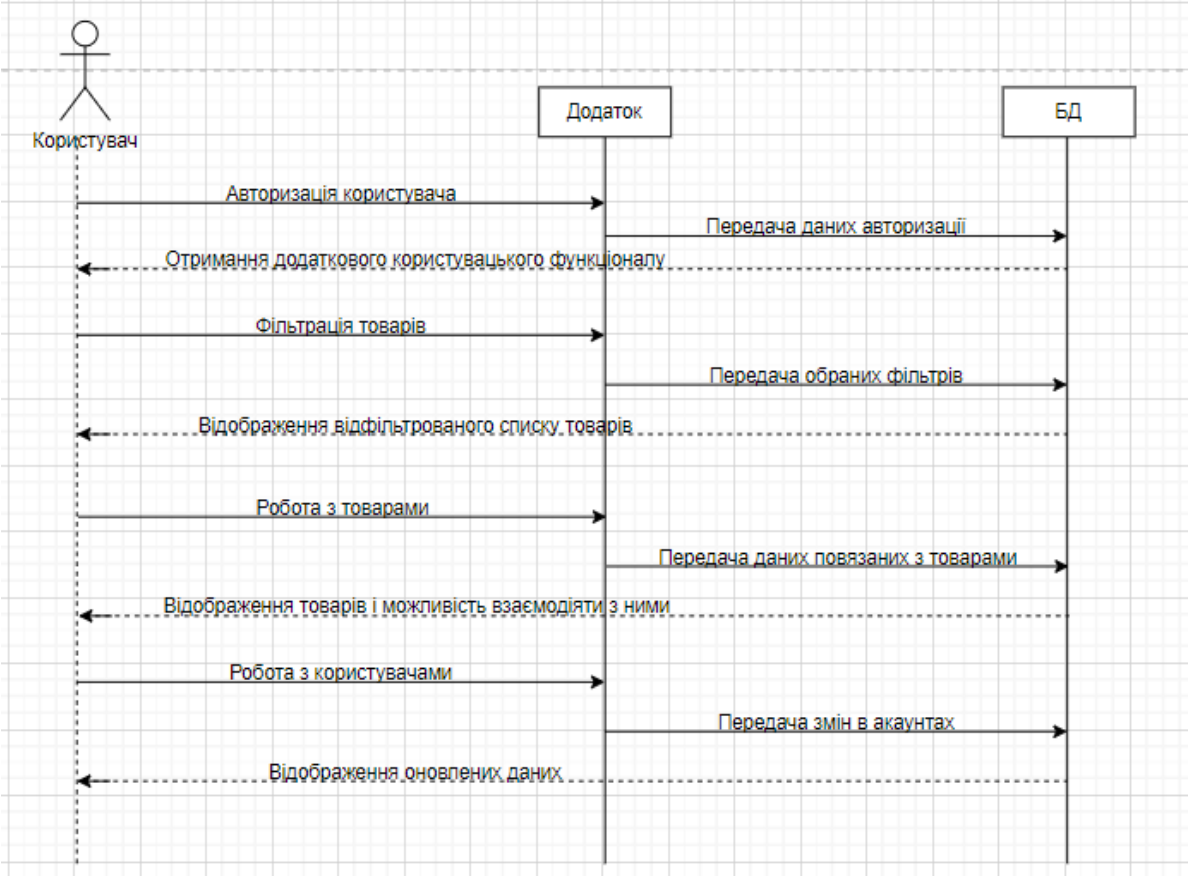


Рисунок 2.6 – Діаграма послідовності дій

Висновок до розділу

У цьому розділі визначено мову програмування, технологічний стек та допоміжні інструментальні засоби, що використовуються при розробці застосунку. Спроєктовано алгоритми роботи ключових функціональних модулів системи, побудовано UML-діаграму варіантів використання, що відображає доступний функціонал для кожної ролі. Детальна послідовність взаємодії між користувачем та системою представлена на діаграмі послідовності дій.

3 РЕАЛІЗАЦІЯ ЗАСТОСУНКУ ІНТЕРНЕТ-МАГАЗИНУ

3.1 Архітектурне рішення застосунку

В даному застосунку застосовано архітектурний патерн MVVM (Model-View-ViewModel), що забезпечує чітке розмежування між бізнес-логікою та візуальним представленням. Патерн складається з трьох компонентів:

– Модель відповідає за опис та зберігання даних, що використовуються в застосунку.

– Подання формує візуальний інтерфейс, через який здійснюється взаємодія із застосунком.

– Модель подання слугує сполучною ланкою між Моделлю та Поданням, реалізуючи механізм прив’язки даних.

ViewModel також відповідає за отримання даних з моделі та їх подальшу передачу до подання для відображення.

На рисунку 3.1 представлено діаграму, що ілюструє загальну структуру програми в рамках патерну MVVM.



Рисунок 3.1 – Загальна архітектура застосунку

Для зберігання даних застосунок використовує базу даних, реалізовану на платформі Microsoft SQL Server. Перелік таблиць, необхідних для функціонування системи (зображено на рис. 3.2):

- MigrationHistory — журнал міграцій бази даних
- Parts — перелік автозапчастин
- Carts — дані про вміст кошиків користувачів
- Cards — збережені банківські картки

– Providers — інформація про постачальників

– Users — облікові записи користувачів

– Orders — оформлені замовлення

– Categories — категорії автозапчастин

– Deliveries — способи та дані доставки

– OrderedParts — деталізовані позиції замовлень

– Marks — марки автомобілів

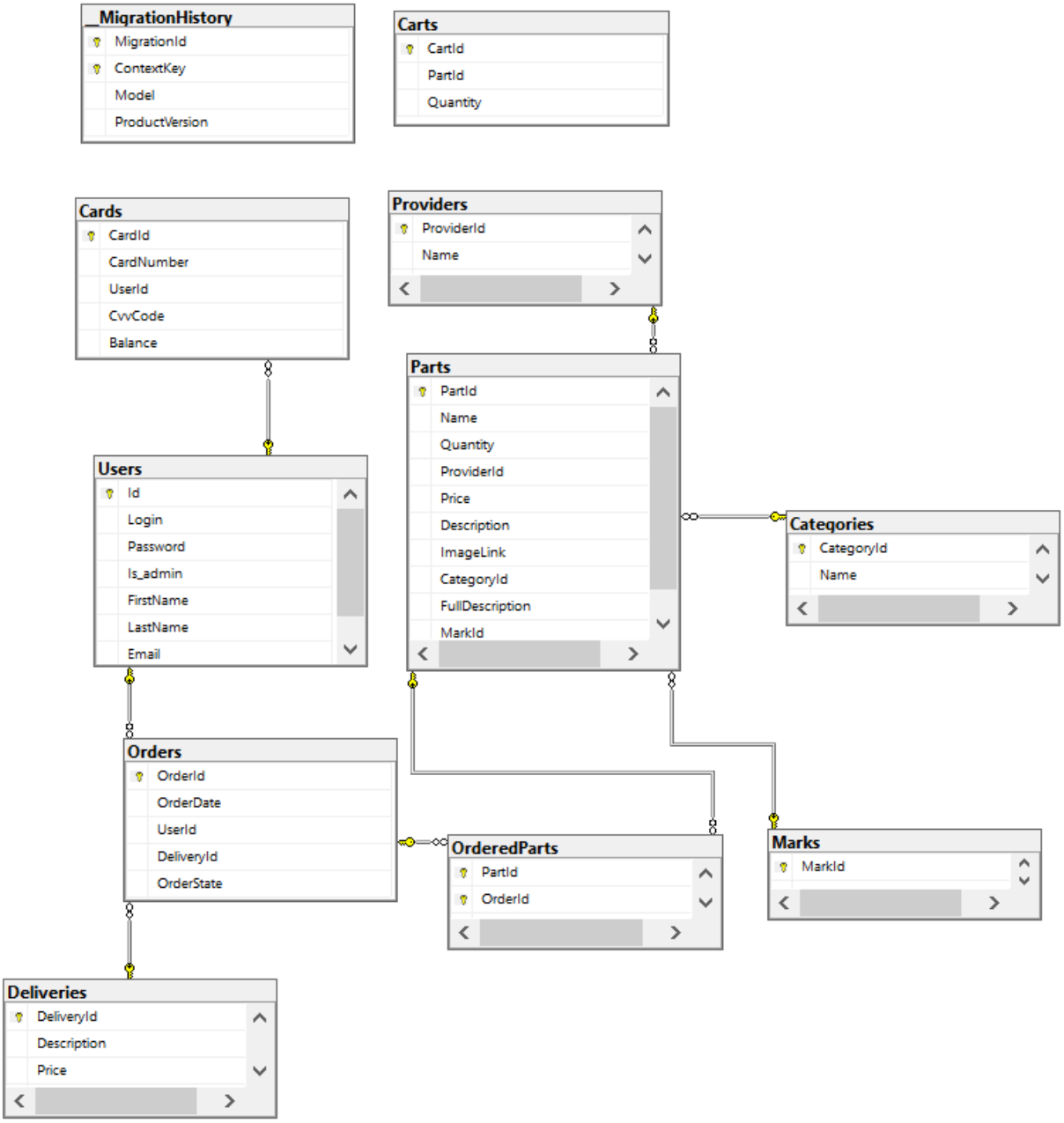


Рисунок 3.2 – структура бази даних

3.2 Опис структури та елементів застосунку

У системі передбачено три ролі: звичайний користувач, адміністратор застосунку та адміністратор бази даних. Кожна роль має власний набір доступних функцій (зображено на рис. 3.3).

Функціонал звичайного користувача охоплює перегляд списку автозапчастин та вмісту кошика. Крім того, користувач може фільтрувати товари за потрібними параметрами, додавати позиції до кошика або видаляти їх. Для оформлення замовлення користувач має змогу прив’язати банківську карту та обрати зручний спосіб доставки.

Адміністратор застосунку має розширені права: управління обліковими записами користувачів (редагування даних, видалення, зміна прав доступу), а також повноцінне керування каталогом товарів — додавання нових позицій, їх видалення та оновлення інформації про наявні записи.

Адміністратор бази даних відповідає за управління операторами БД: додавання нових облікових записів, налаштування прав доступу та видалення операторів. До його обов’язків також входить безпосереднє адміністрування: розгортання, розробка та оновлення структури БД, резервне копіювання та відновлення для забезпечення надійності роботи. Крім того, доступна функція підключення до бази даних, що включає авторизацію, отримання необхідних прав та завершення сеансу.

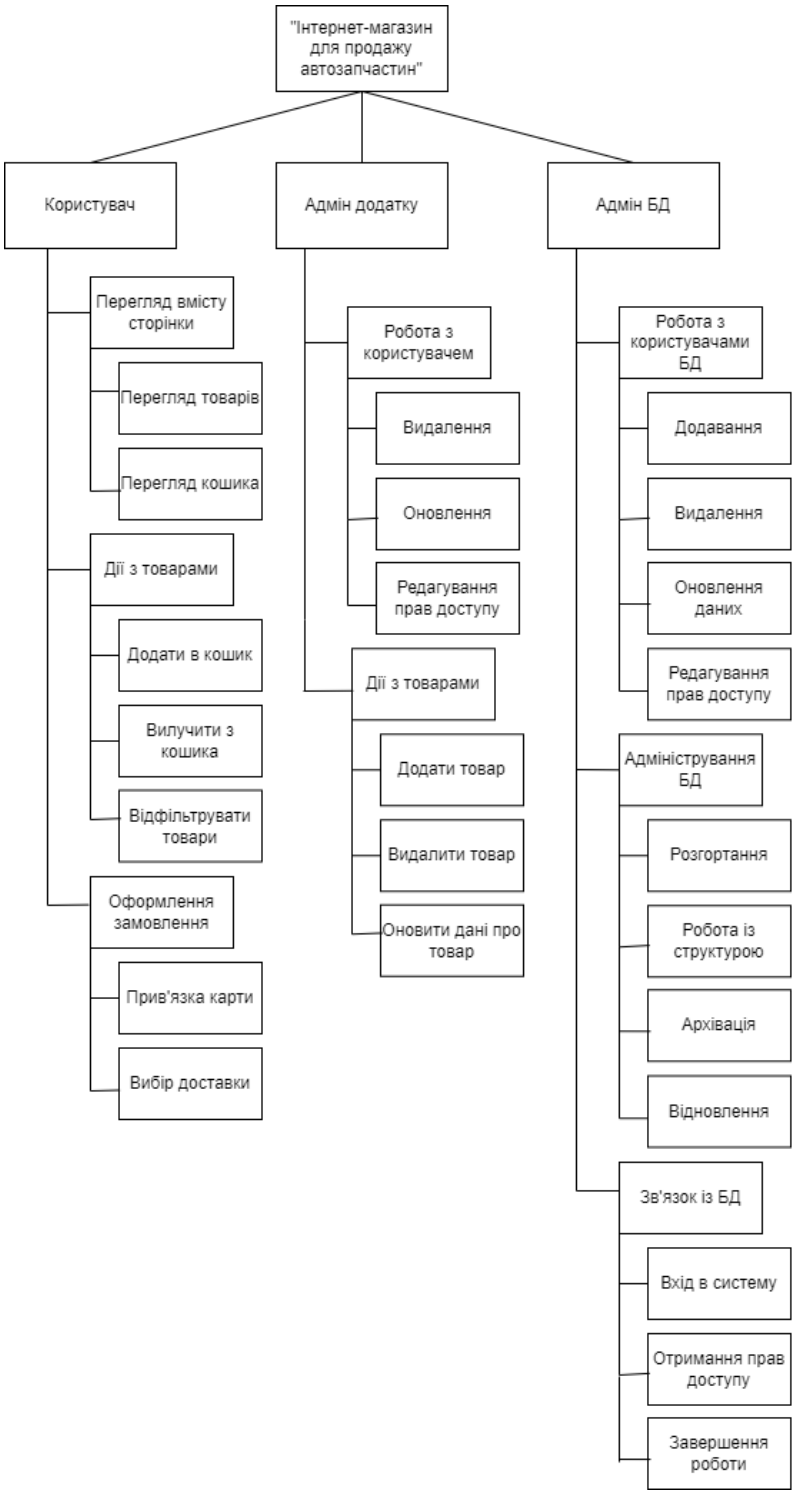


Рисунок 3.3 – Структура елементів застосунку

Під час запуску програми відображається форма авторизації. За відсутності облікового запису передбачена можливість реєстрації. Без попередньої авторизації доступ до функціоналу закрито. Після успішного входу користувач переходить на головну сторінку та отримує доступ до всіх доступних функцій. (зображено на рис. 3.4).

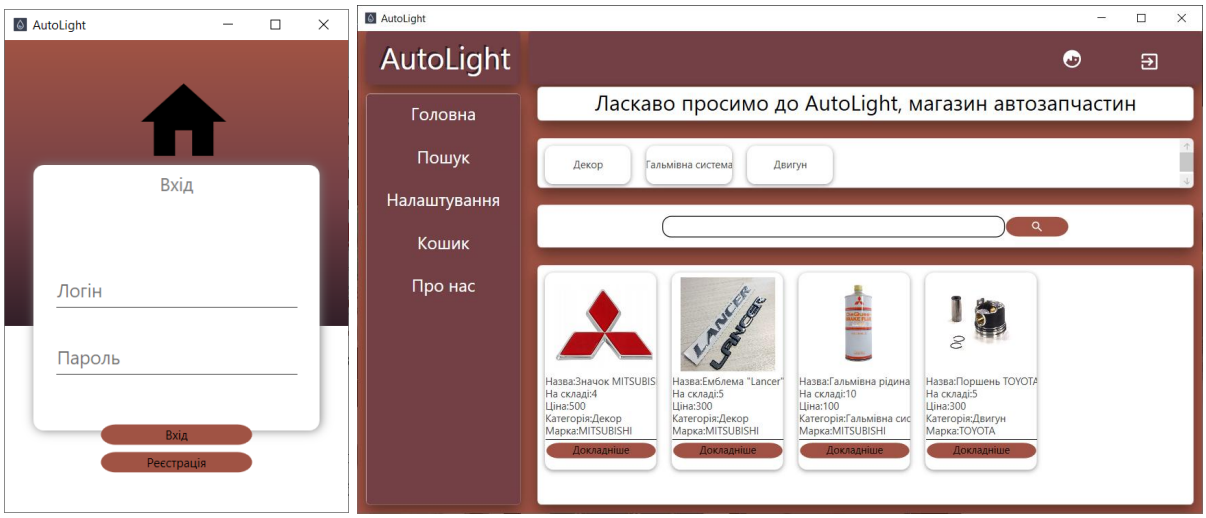


Рисунок 3.4 – Основні сторінки застосунку з переходами між ними

3.3 Реалізація інтерфейсу застосунку

Після запуску застосунку на екрані з’являється форма авторизації (зображено на рис. 3.5), на якій буде зображено поле для вводу логіну, поля для вводу пароля і кнопки: «Вхід» і «реєстрація», якщо у користувача ще немає власного акаунту.

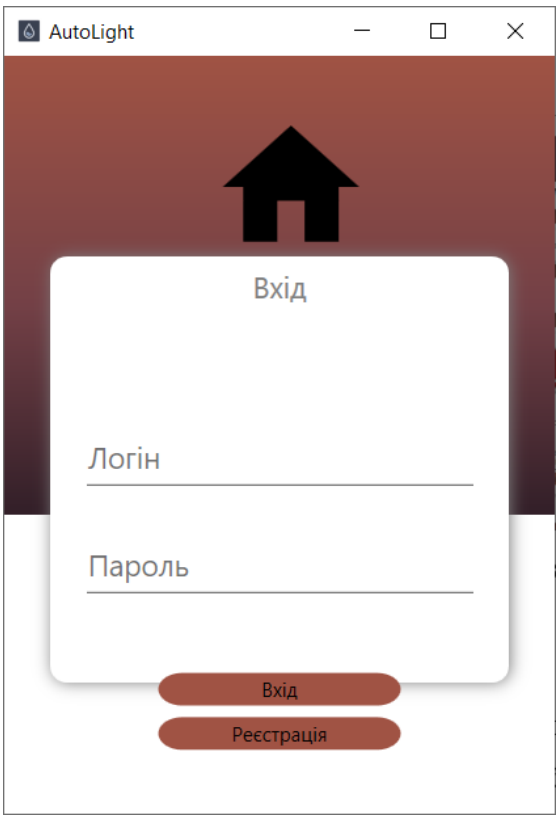


Рисунок 3.5 – Інтерфейс при завантаженні застосунку

За відсутності облікового запису користувач натискає «Реєстрація» і переходить до форми створення нового акаунту (зображено на рис. 3.6), на якій користувач може ввести дані для реєстрації в застосунку. Якщо дані пройшли валідацію, то користувачу буде створено новий власний акаунт. Після чого користувач знову перейде на форму для авторизації (рис. 3.6)

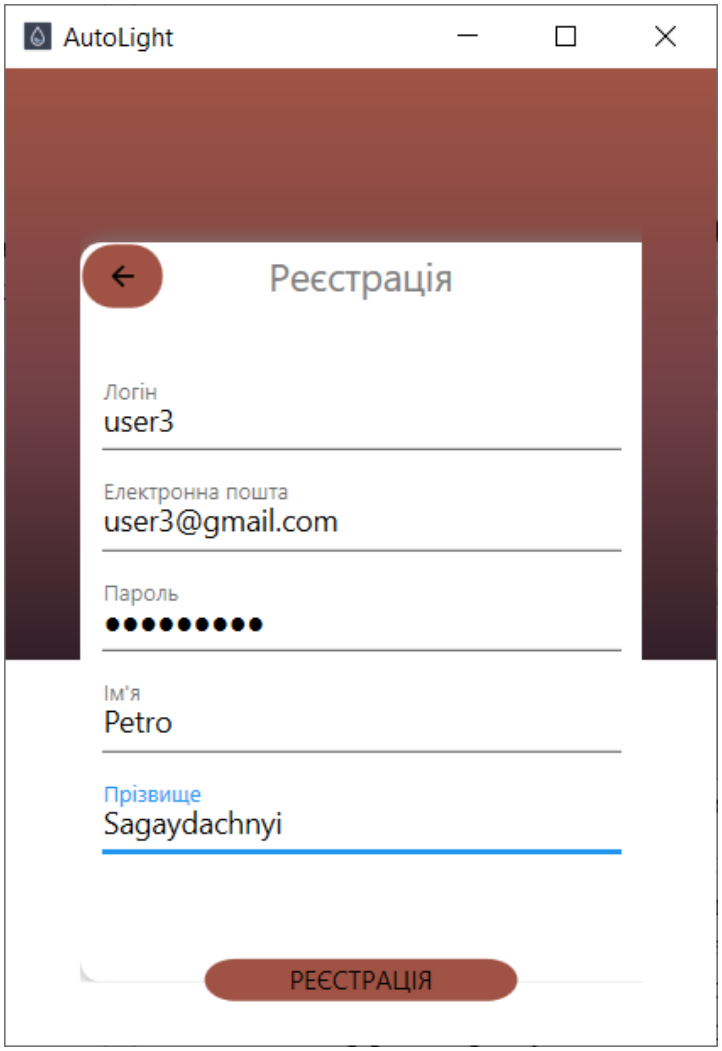


Рисунок 3.6 – Готова до роботи головна сторінка застосунку

Після успішної авторизації користувач потрапляє на головну сторінку магазину (інтерфейс зображено на рис. 3.7), де розміщено рядок пошуку, фільтр за категоріями та каталог автозапчастин. Звідси доступні розділи «Пошук» (рис. 3.8), «Налаштування» (рис. 3.9), «Кошик» (рис. 3.10), «Про нас» (інтерфейс зображено на рис. 3.11), також кнопка «Профіль» (інтерфейс зображено на рис. 3.12), і кнопка «Вихід з акаунту».

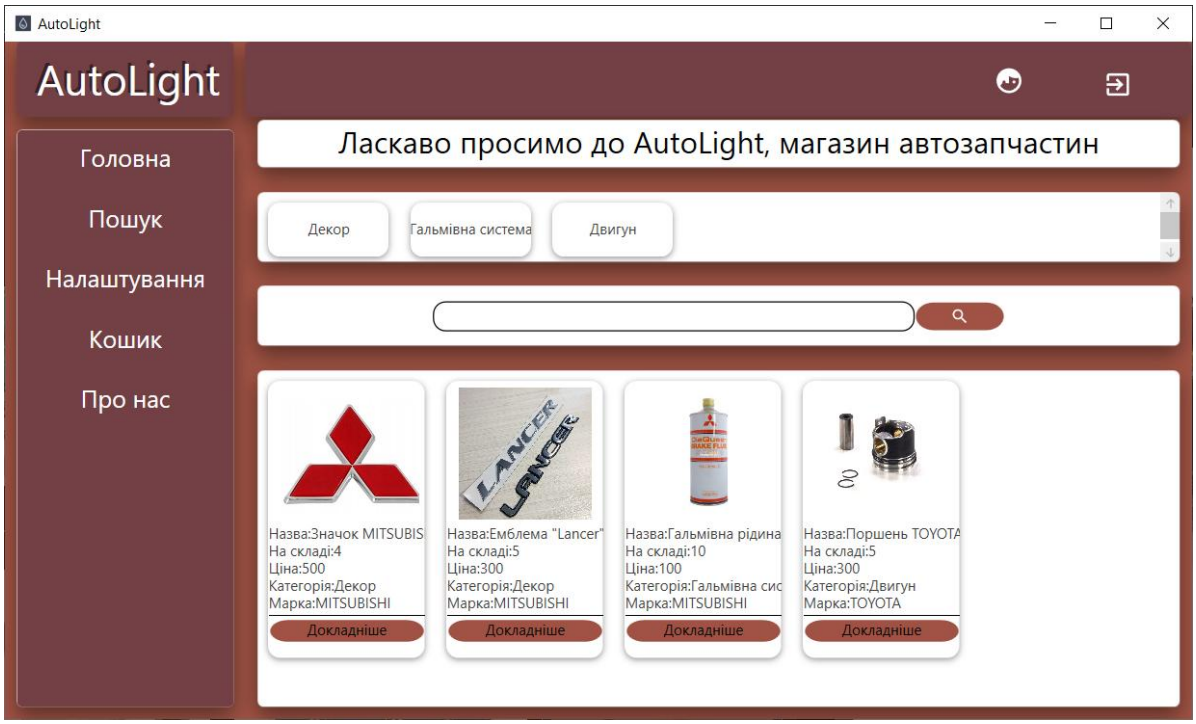


Рисунок 3.7 – Головна сторінка

Вкладка «Пошук» надає користувачу доступ до розширеного пошуку, що дозволяє швидко знайти потрібну автозапчастину за різними параметрами.

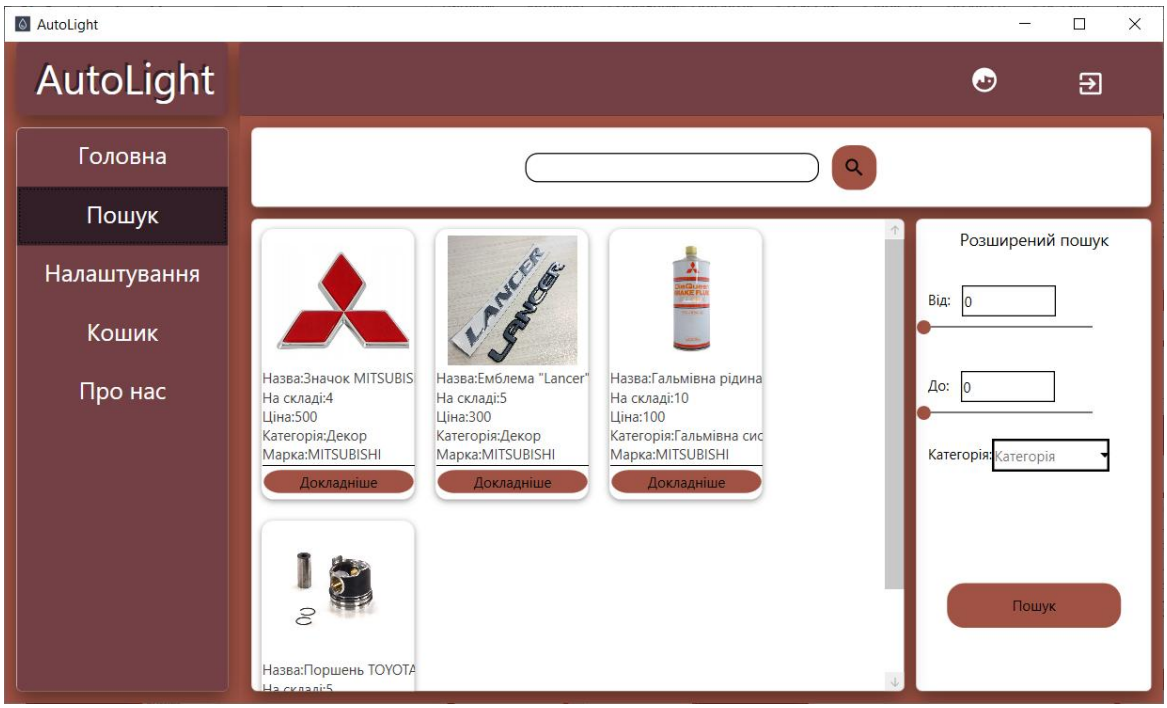


Рисунок 3.8 – Інтерфейс сторінки «Пошук»

На вкладці «Налаштування» користувач може змінити мову інтерфейсу. Застосунок підтримує дві мови: українську та англійську.

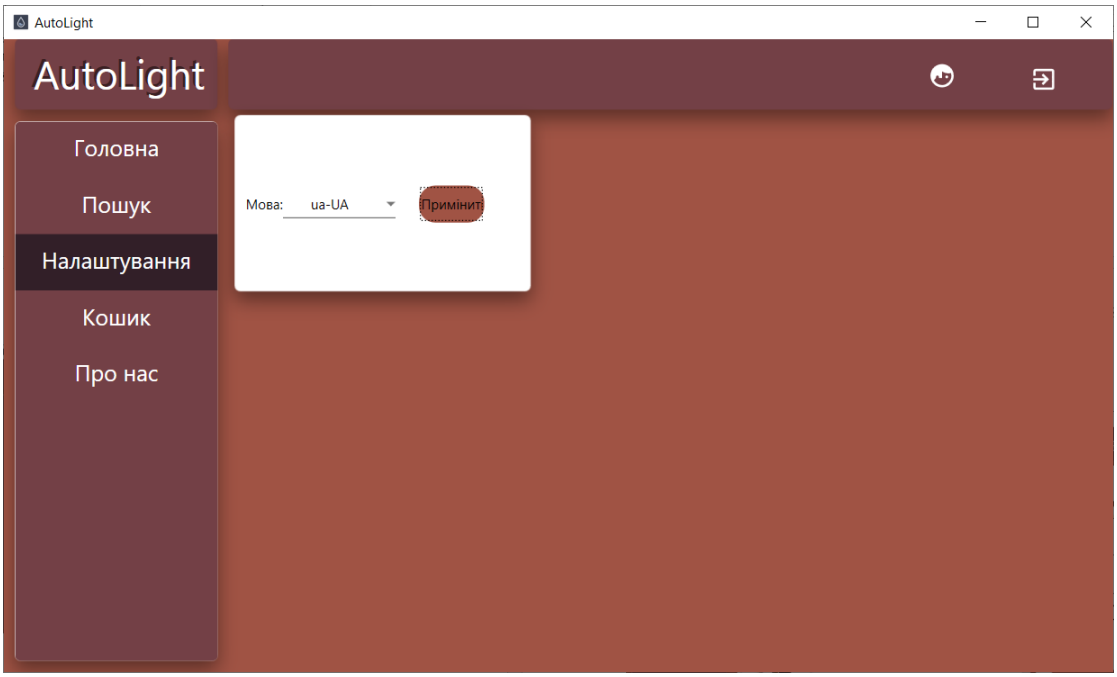


Рисунок 3.9 – Інтерфейс сторінки «Налаштування»

Розділ «Кошик» відображає перелік товарів, доданих користувачем раніше. Тут можна переглянути вміст кошика, видалити зайві позиції або перейти до оформлення замовлення.

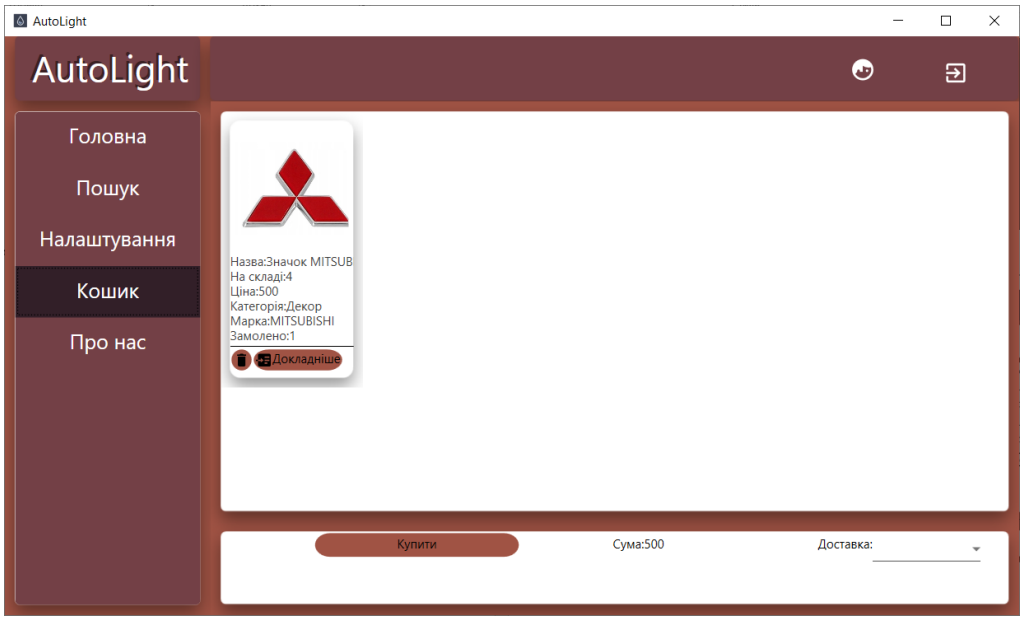


Рисунок 3.10 – Інтерфейс сторінки «Кошик»

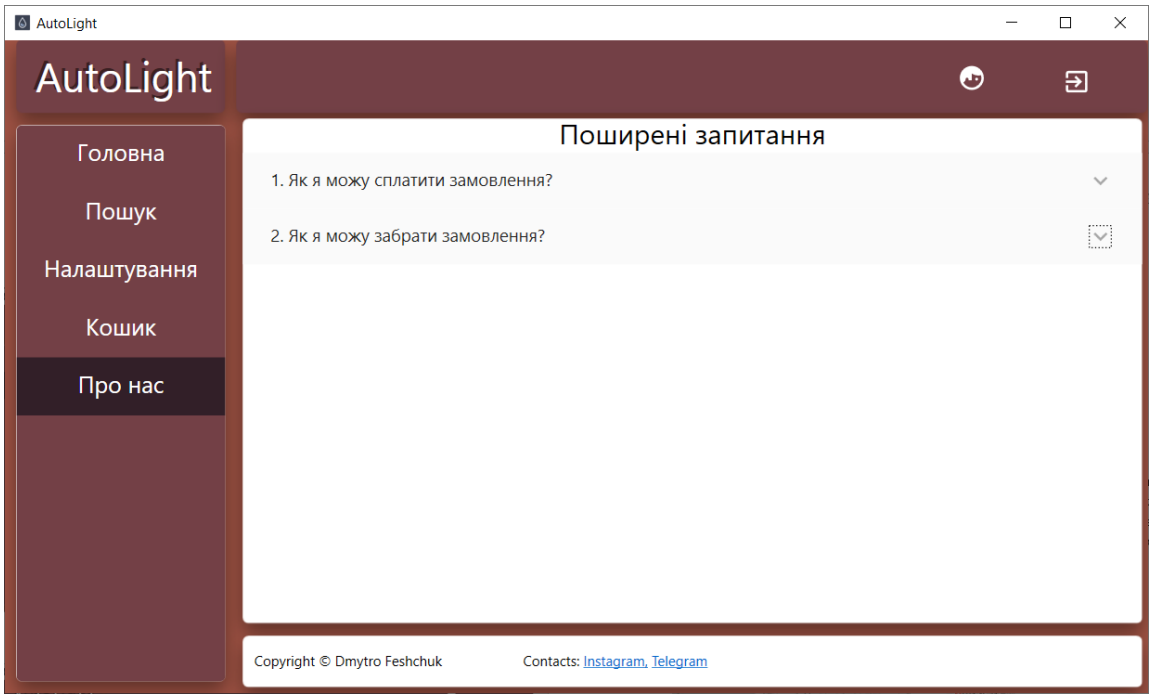


Рисунок 3.11 – Інтерфейс сторінки «Про нас»

У розділі «Профіль» користувач може переглянути власні персональні дані, прив’язати банківську карту, відстежувати оформлені замовлення та скасовувати їх за потреби. Також доступна функція зміни пароля.

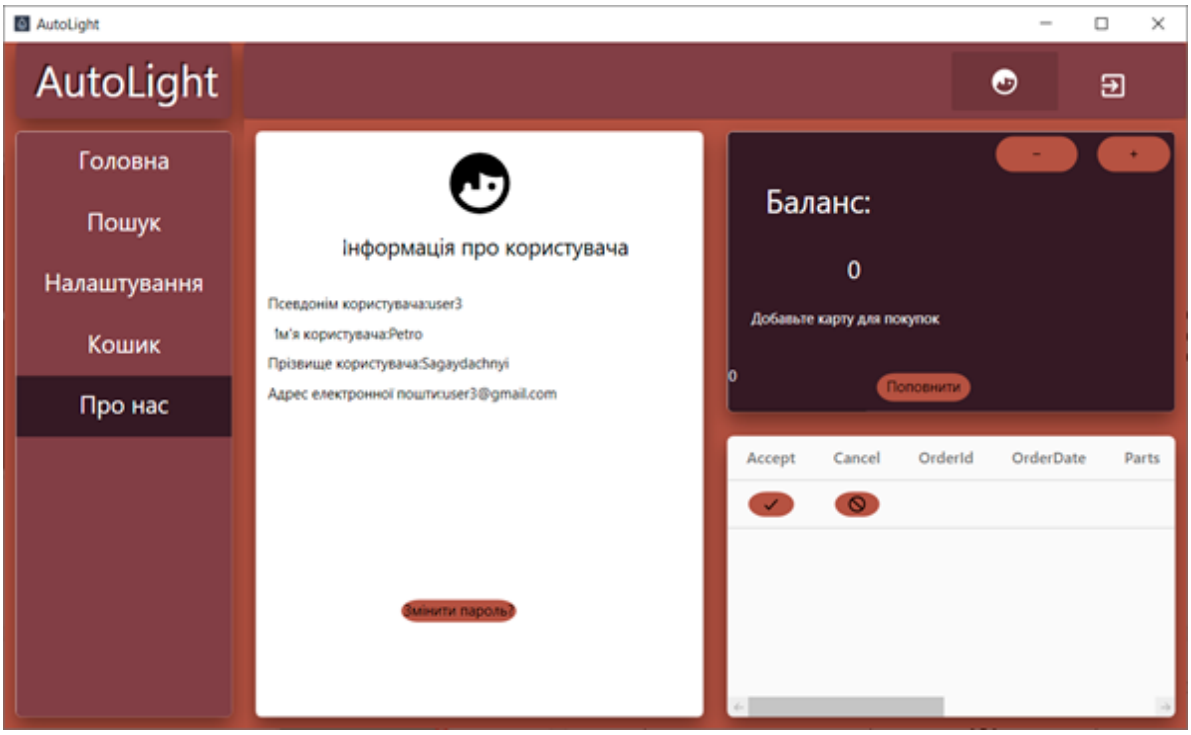


Рисунок 3.12 – Інтерфейс сторінки «Профіль»

Якщо обліковий запис має права адміністратора, інтерфейс застосунку відрізняється від стандартного. Після входу адміністратора відкривається спеціальна стартова форма (рис. 3.13).

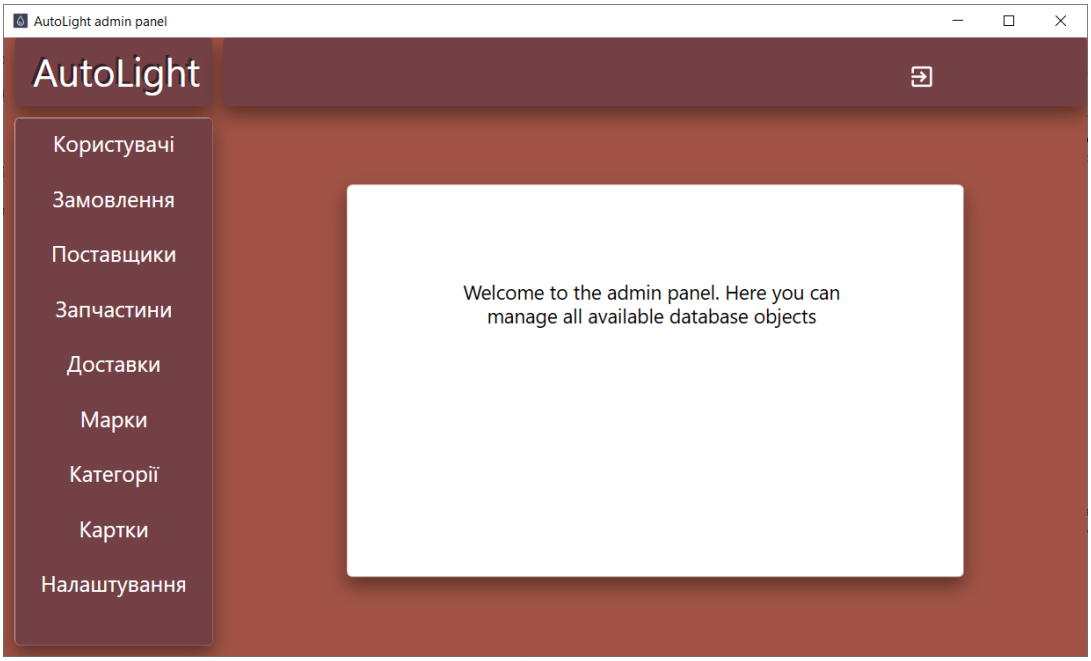


Рисунок 3.13 – Інтерфейс стартової сторінки адміна

Адміністратору доступні спеціальні розділи керування: користувачі (рис. 3.14), замовлення (рис. 3.15), постачальники (рис. 3.16), запчастини (рис. 3.17), доставки (рис. 3.18), марки автомобілів (рис. 3.19), категорії товарів (рис. 3.20), банківські картки (рис. 3.21) та налаштування. У кожному з розділів адміністратор може додавати нові записи, редагувати наявні або видаляти їх.

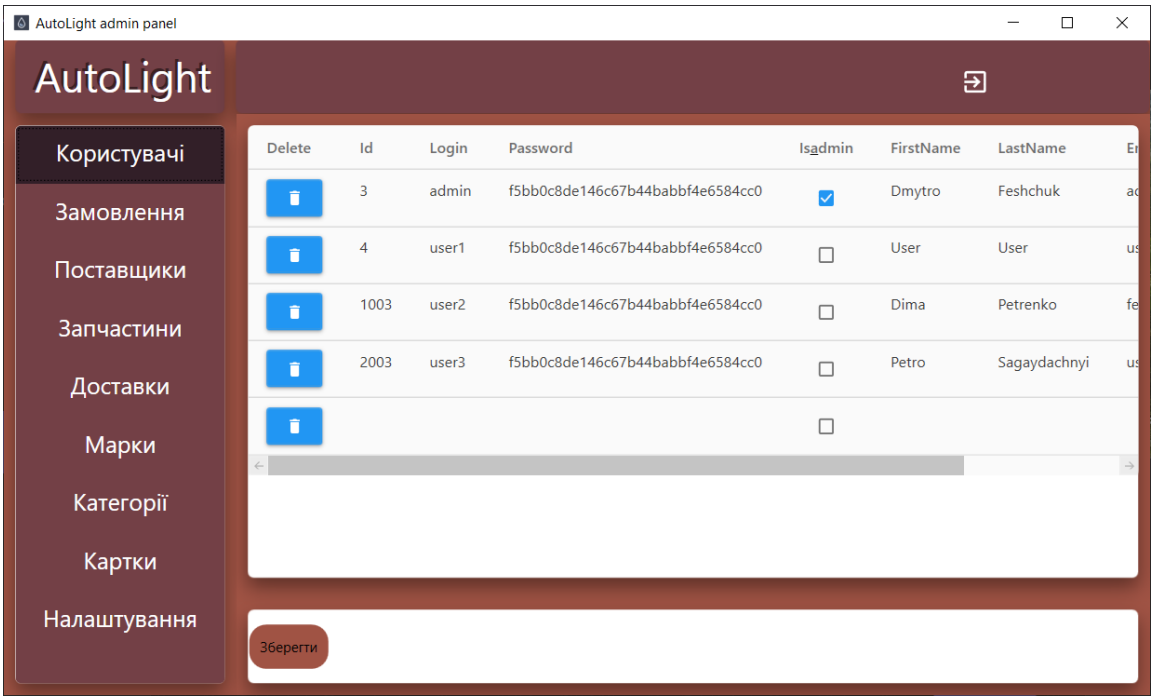


Рисунок 3.14 – Інтерфейс сторінки «Користувачі»

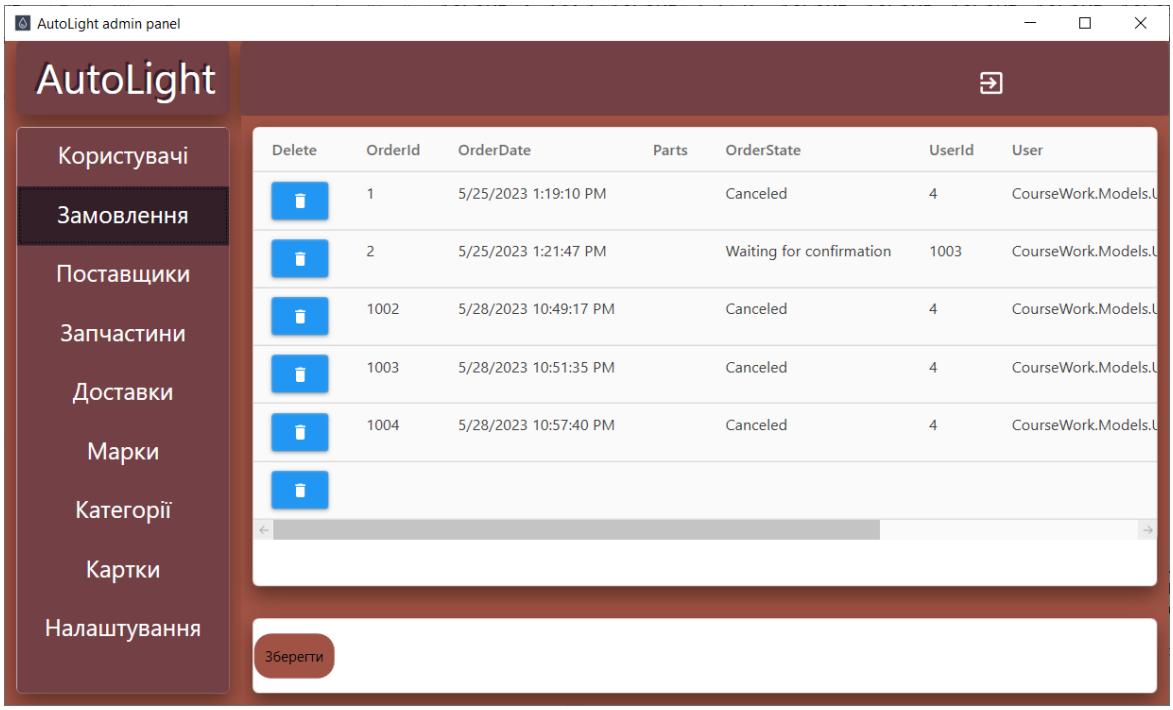


Рисунок 3.15 – Інтерфейс сторінки «Замовлення»

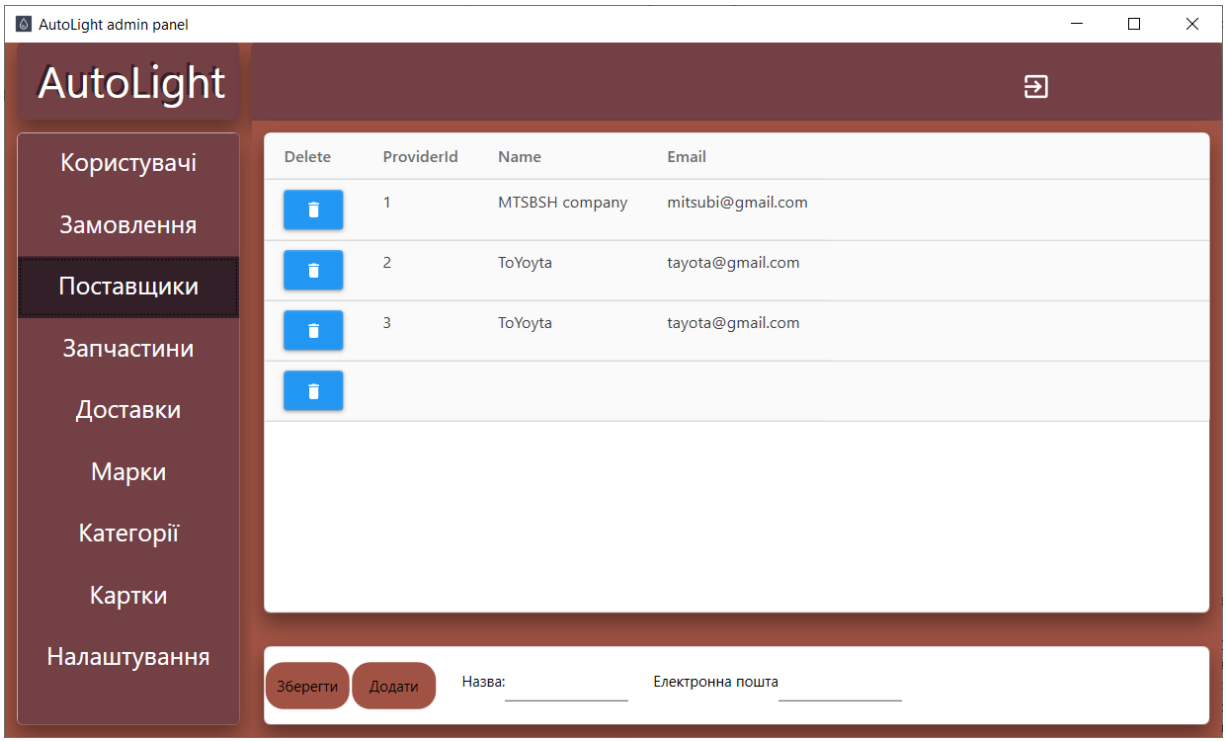


Рисунок 3.16 – Інтерфейс сторінки «Поставщики»

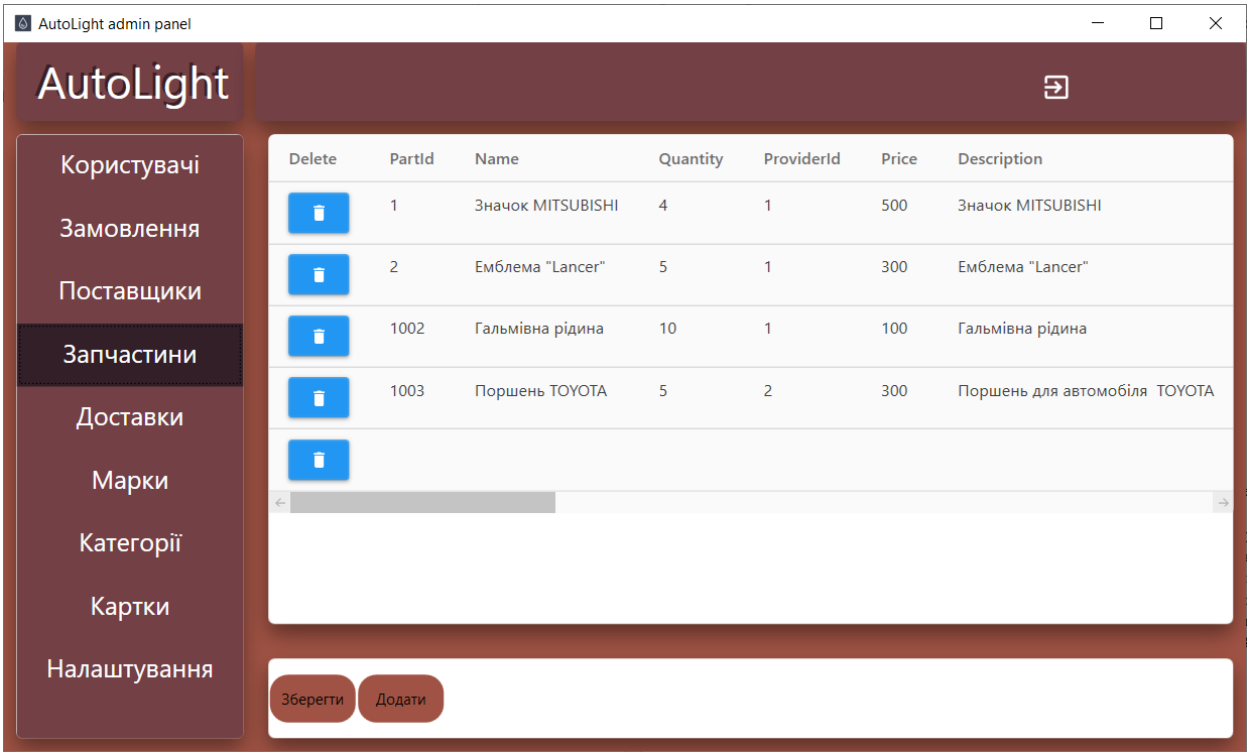


Рисунок 3.17 – Інтерфейс сторінки «Запчастини»

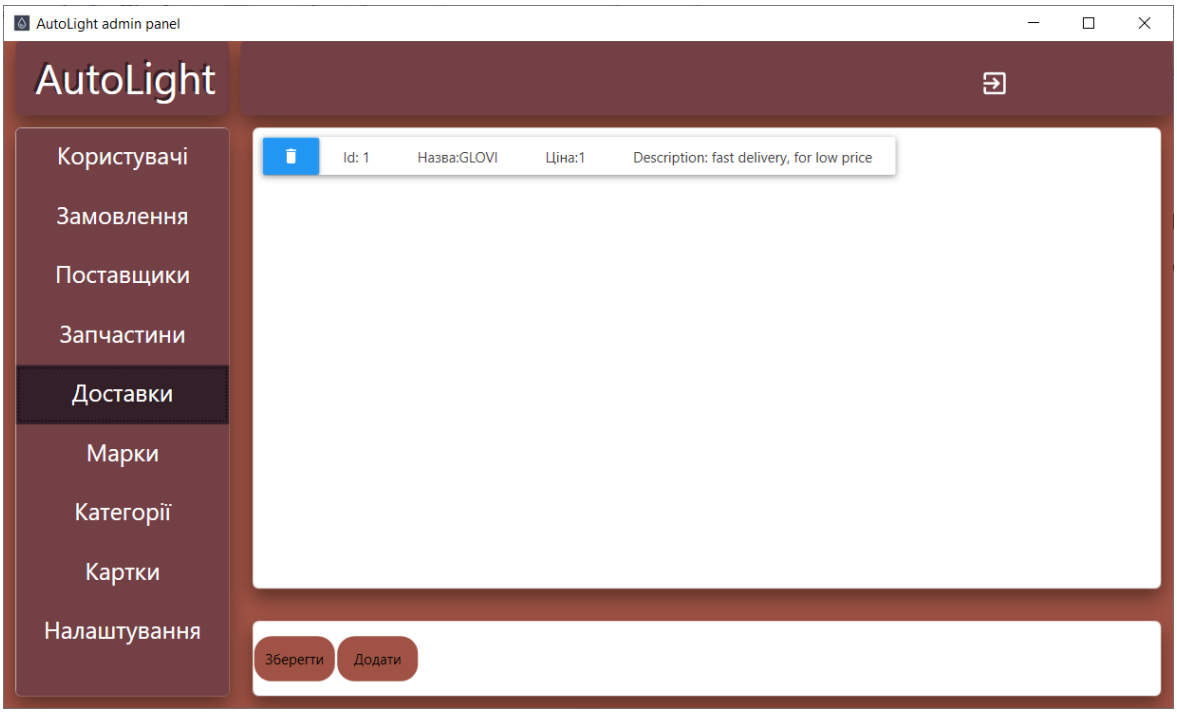


Рисунок 3.18 – Інтерфейс сторінки «Доставки»

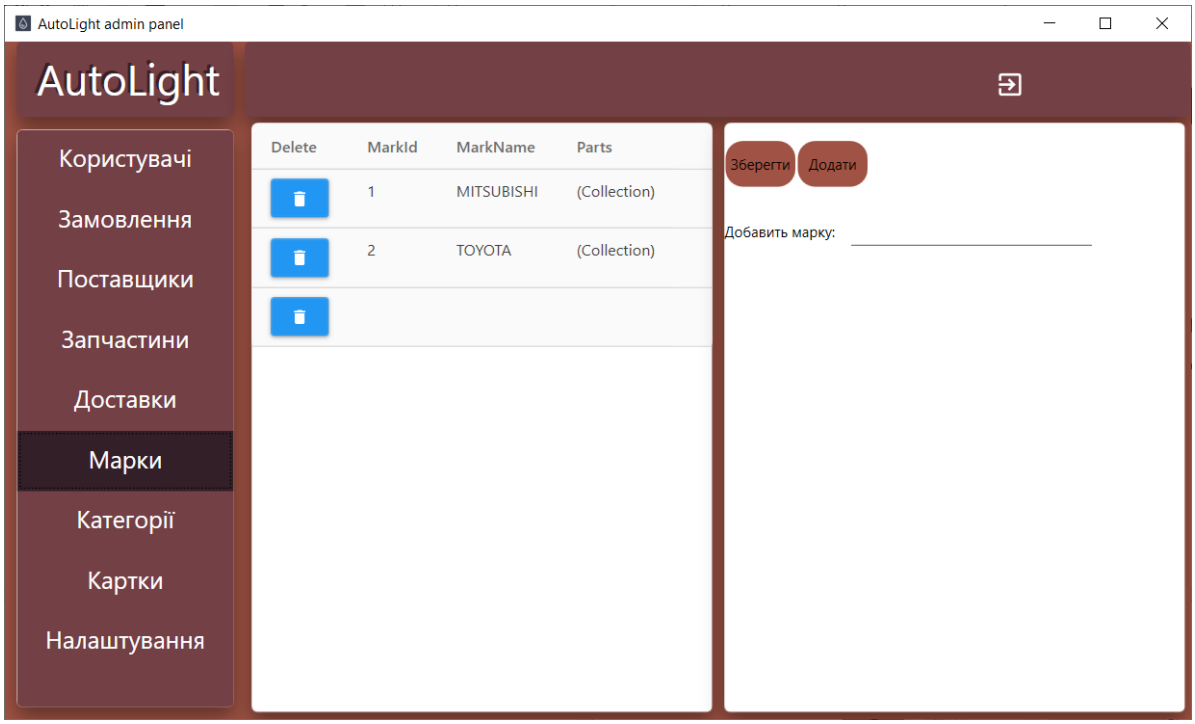


Рисунок 3.19 – Інтерфейс сторінки «Марки»

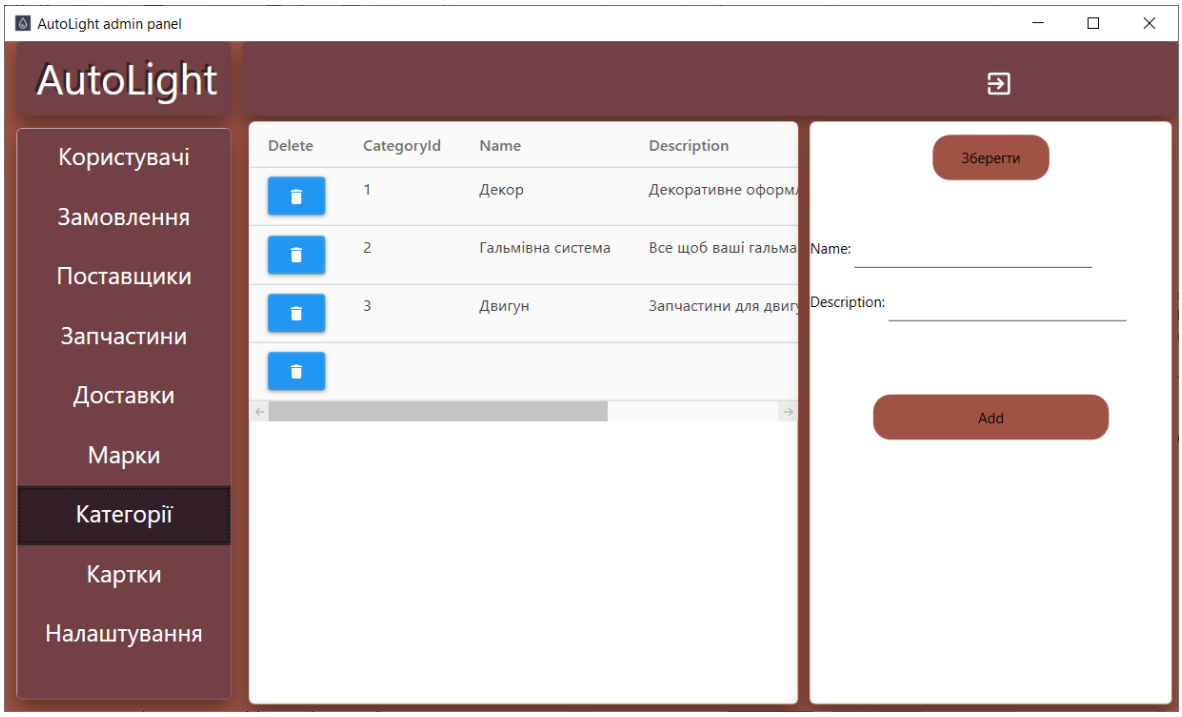


Рисунок 3.20 – Інтерфейс сторінки «Категорії»

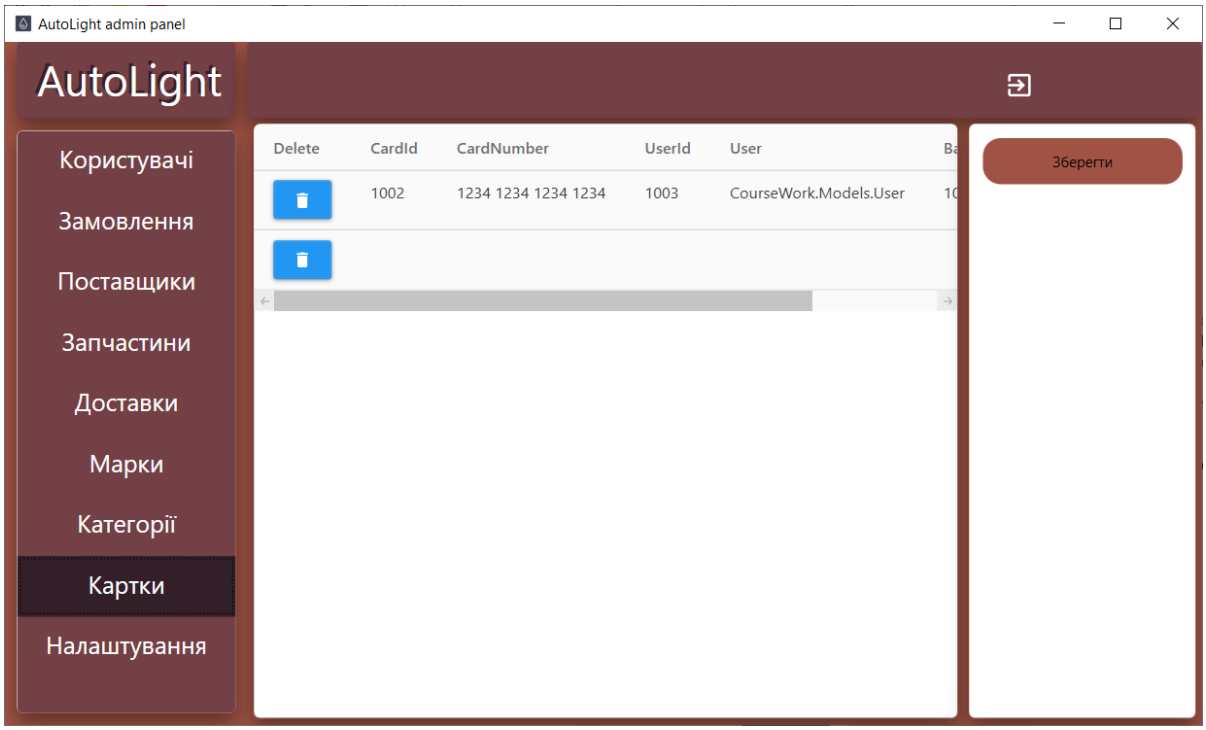


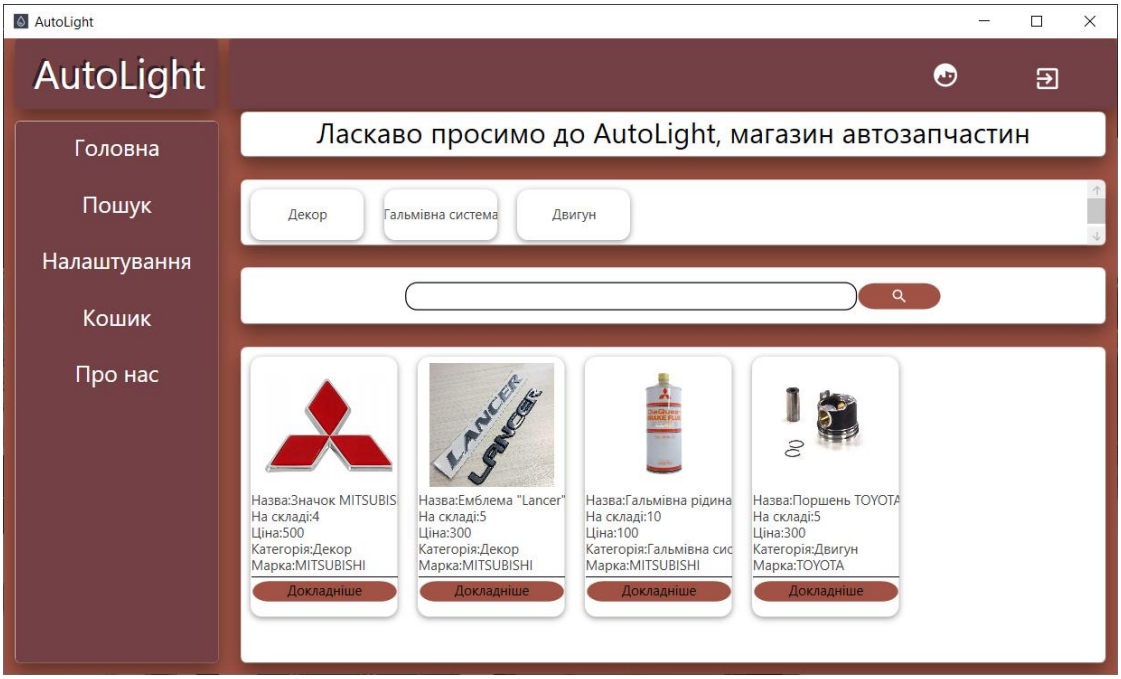
Рисунок 3.21 – Інтерфейс сторінки «Картки»

3.4 Опис функціонування застосунку

Під час запуску застосунок отримує з бази даних актуальну інформацію про кількість товарів, їхню вартість та інші дані (зображено на рис. 3.22 а). Якщо це

стосується акаунту з правами адміністратора — додатково завантажуються відомості про всіх зареєстрованих користувачів та прив’язані до їхніх акаунтів дані (зображено на рис. 3.22 б).

а)



б)

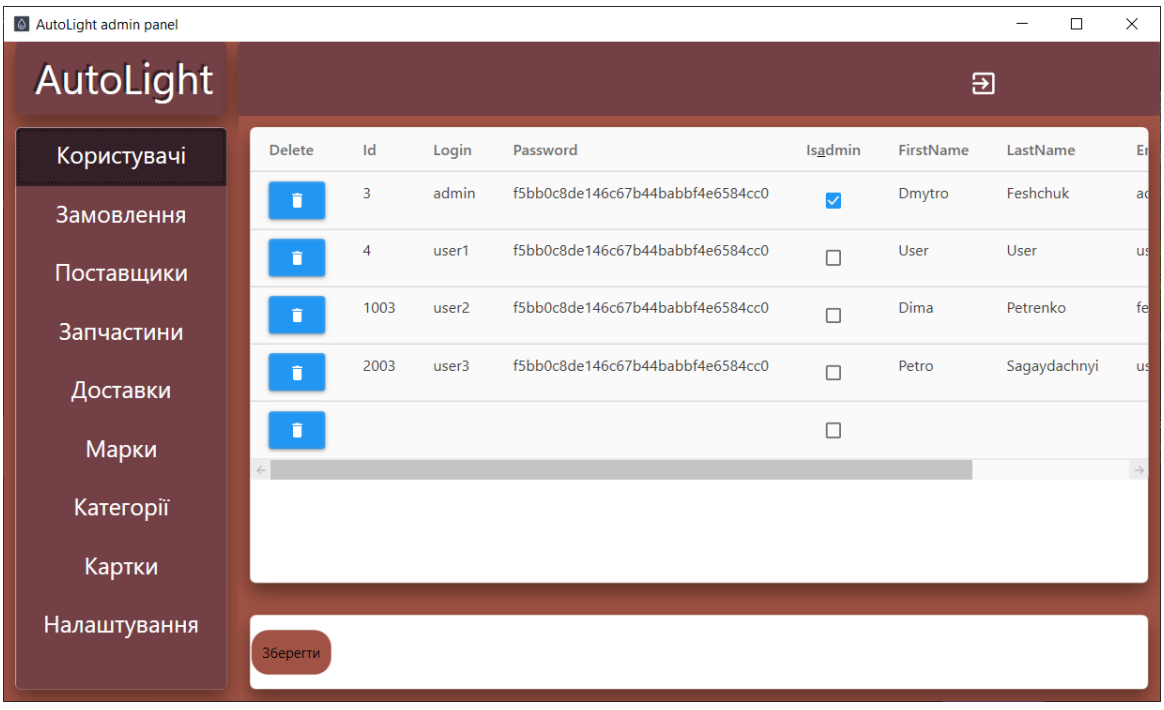


Рисунок 3.22 – а) Вид з акаунту користувача; б)Вид з акаунту адміна

Навігація між розділами здійснюється натисканням відповідних кнопок. Наприклад, кнопка «Кошик» відкриває список товарів, заздалегідь доданих

користувачем. (зображено на рис. 3.23 а). Адміністратору доступні додаткові кнопки переходу до сторінок управління магазином (зображено на рис. 3.23 б)

а)



б)

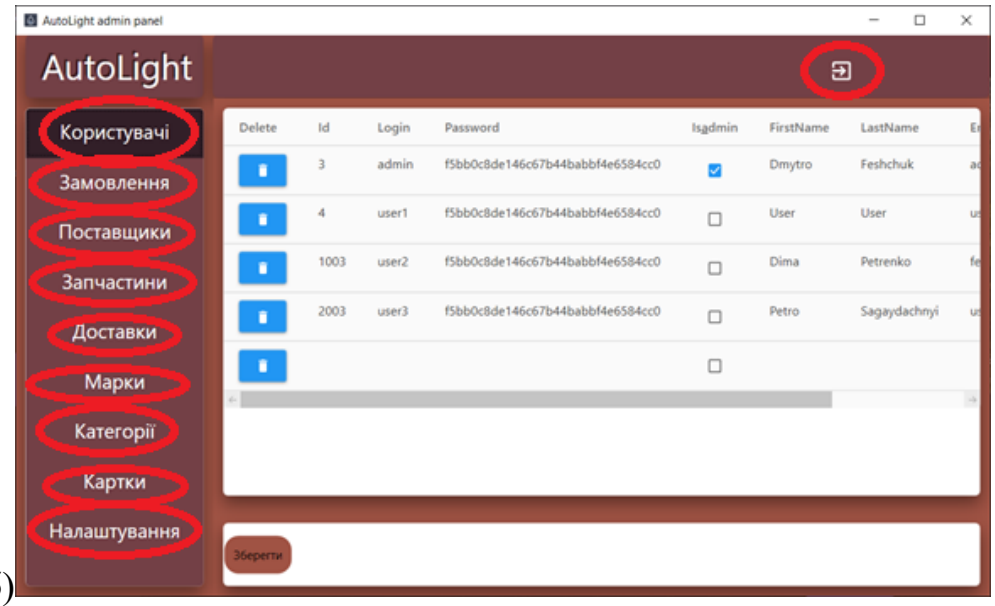
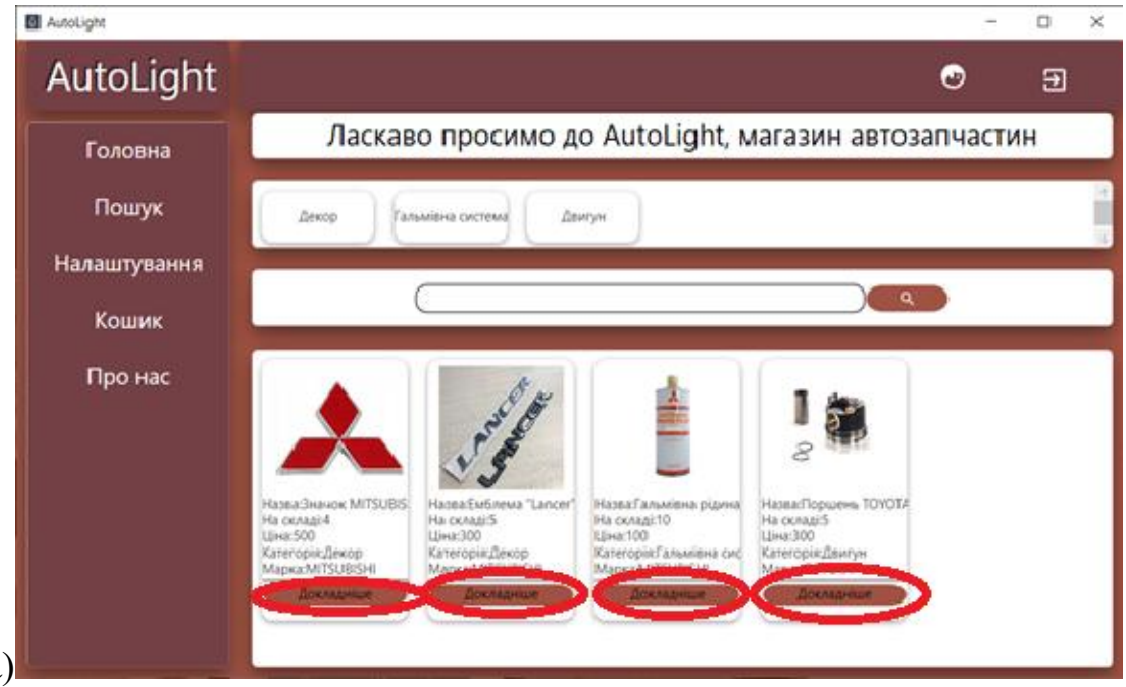


Рисунок 3.23 – а) Доступні вкладки без адмін статусу; б) Доступні вкладки з адмін статусом.

З головної сторінки, а також з інших розділів, користувач може відкрити детальний опис обраної автозапчастини (зображено на рис. 3.24 а) та додати її до кошика (зображено на рис. 3.24 б).

а)



б)

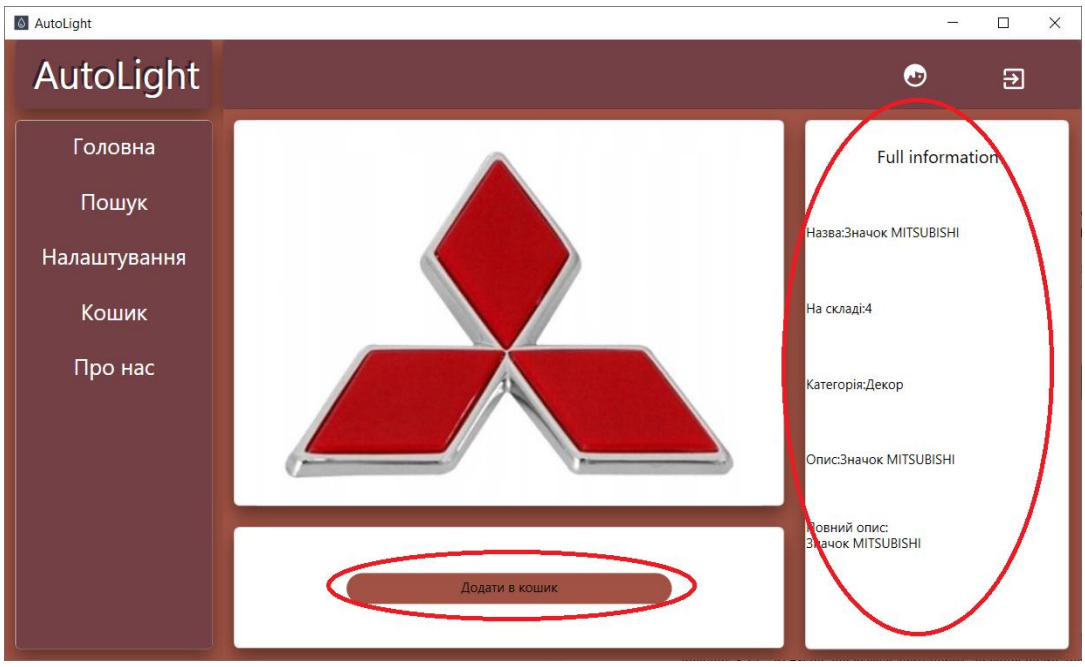
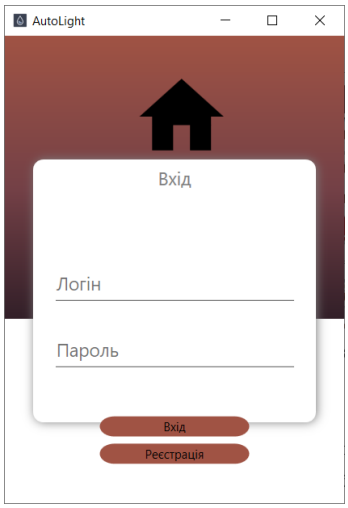
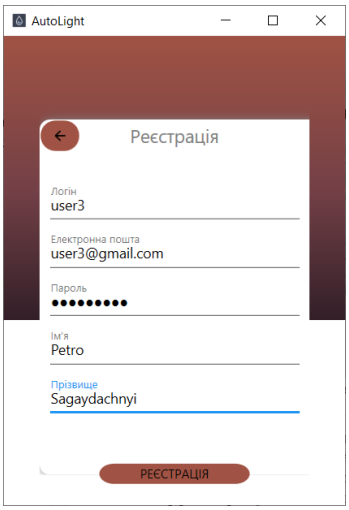


Рисунок 3.24 – а) Список товарів; б) Детальний опис обраного товару

Модуль авторизації дозволяє увійти до існуючого акаунту або пройти реєстрацію. Для входу потрібно натиснути відповідну кнопку та ввести облікові дані (зображено на рис. 3.25 а). При реєстрації відкривається форма, де необхідно заповнити поля та підтвердити створення акаунту (зображено на рис. 3.25 б). Після цього користувач авторизується. Є можливість вийти з акаунту, і перейти до форми авторизації. (зображено на рис. 3.25 в)

а) 

б) 

в) 

Рисунок 3.25 – а) Кнопки авторизації користувача; б) Вікно реєстрації нового користувача; в)Кнопка для виходу з акаунту.

Для зручності пошуку автозапчастин користувачу доступні функції фільтрації та повнотекстового пошуку по каталогу (зображено на рис. 3.26).

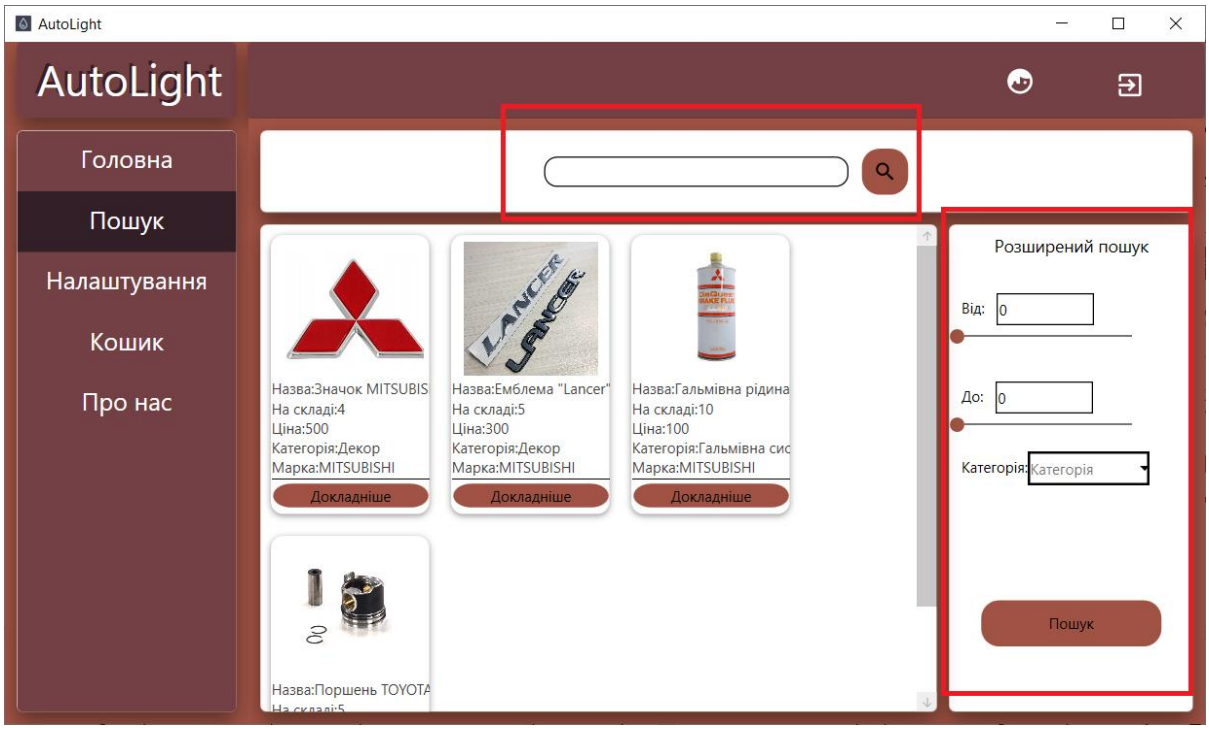
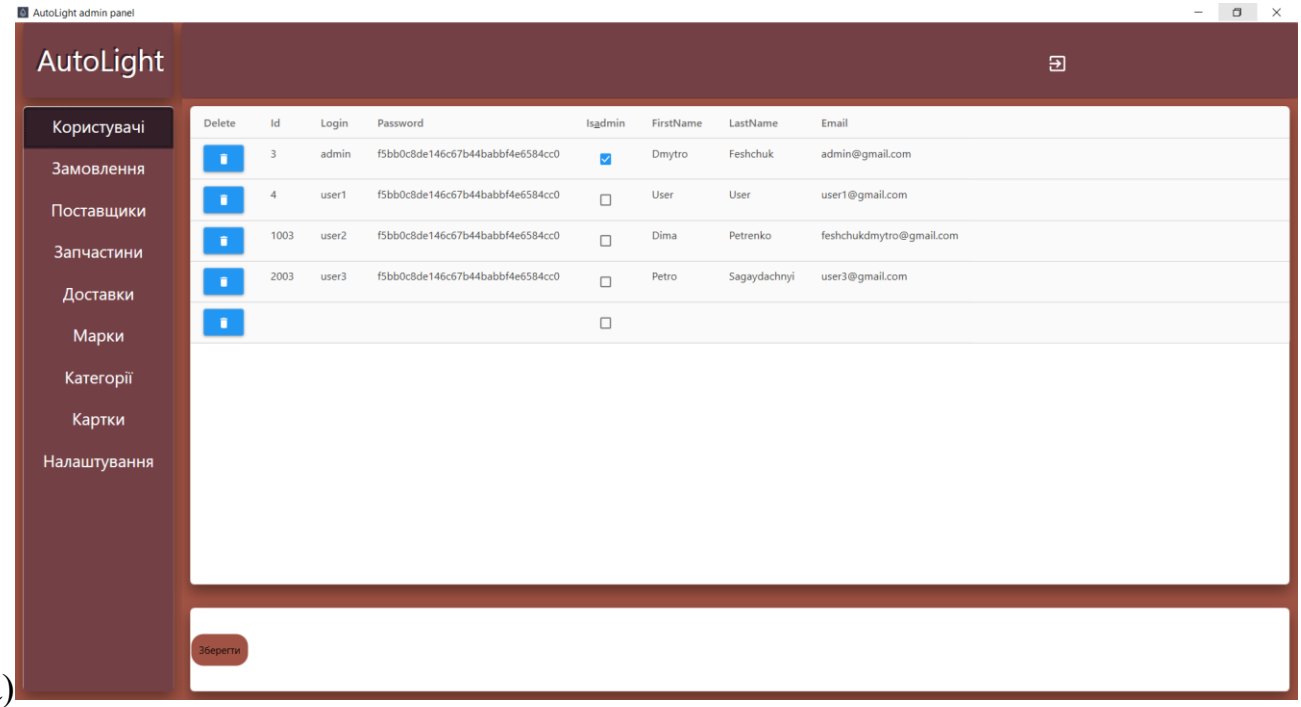


Рисунок 3.26 – Фільтрація і пошук товарів

Функція додавання товарів до каталогу доступна виключно користувачам з правами адміністратора. Адміністратор також може додавати постачальників,

способи доставки тощо (зображено на рис. 3.27 а), редагувати будь-які дані в системі та призначати іншим користувачам статус адміністратора (зображено на рис. 3.27 б).

а)



б)

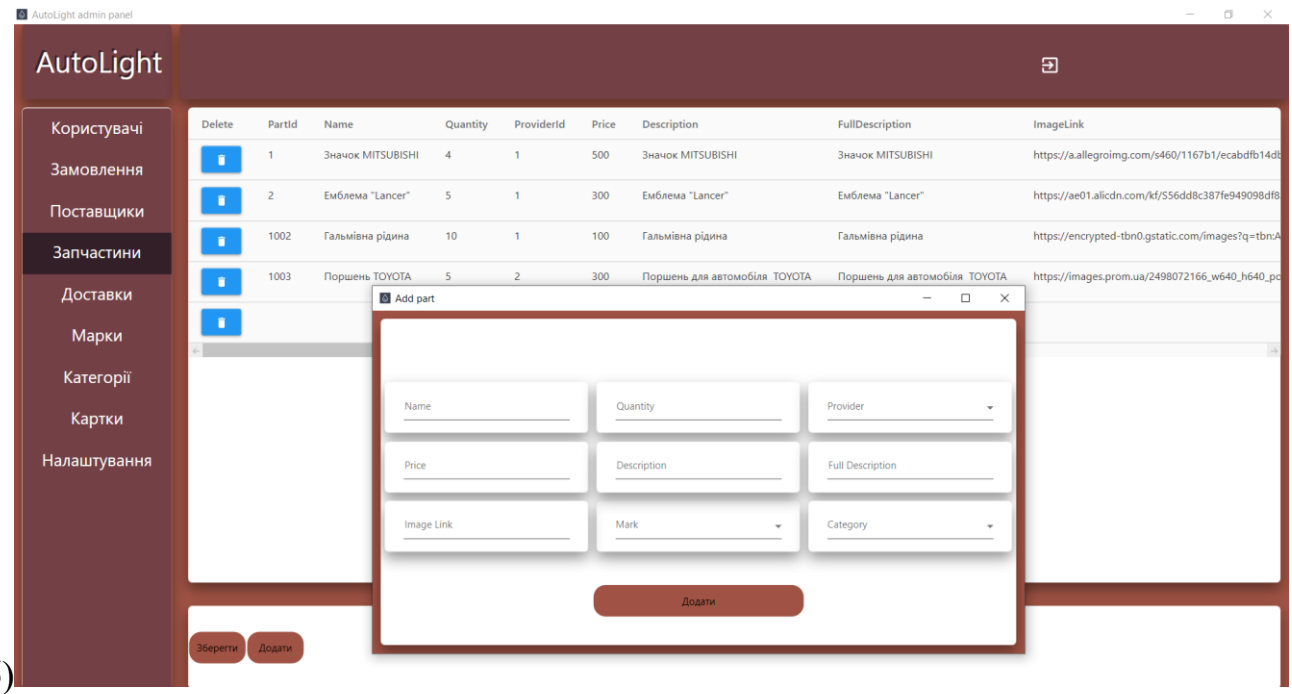
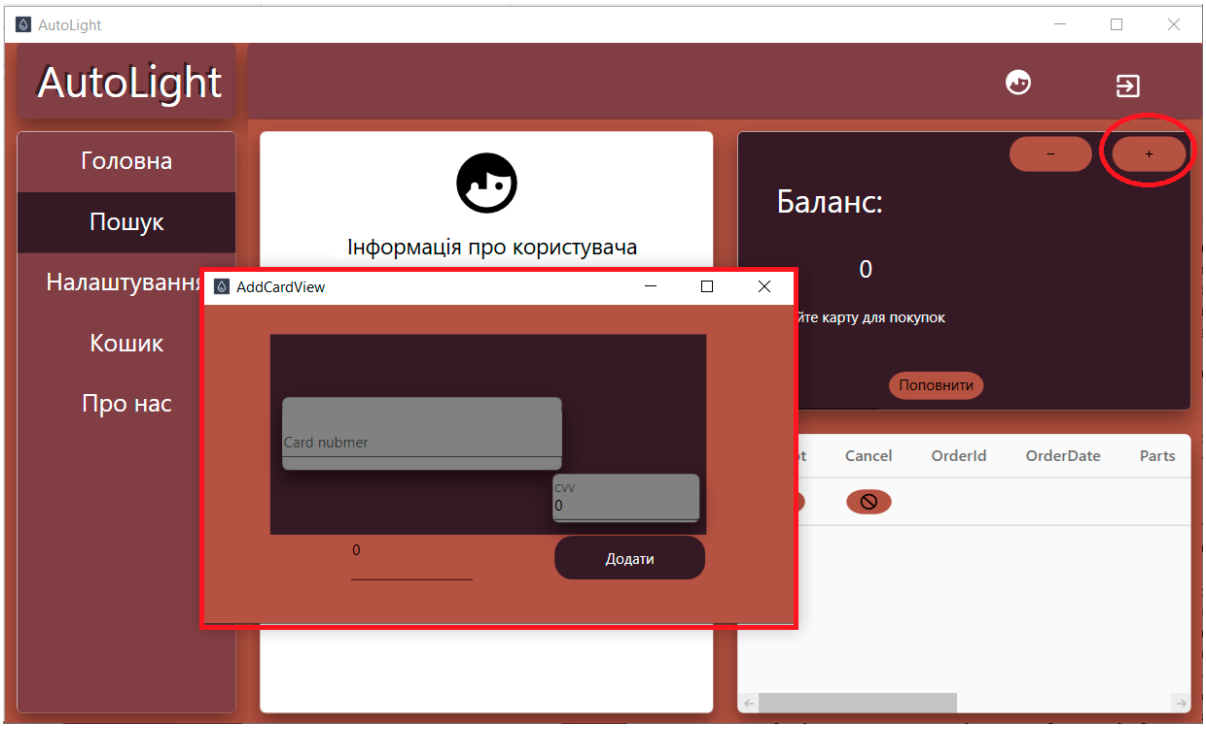


Рисунок 3.27 – а) вигляд вікон для додавання та редагування інформації про користувачів, б)вигляд вікон для додавання та редагування інформації про товари

Реалізовано функцію прив’язки банківської карти. Для цього слід натиснути відповідну кнопку (зображено на рис. 3.28 а), після чого з’явиться форма введення реквізитів. Якщо дані успішно пройдуть валідацію, карта буде прив’язана до акаунту (зображено на рис. 3.28 б). За потреби користувач може відв’язати карту.

а)



б)

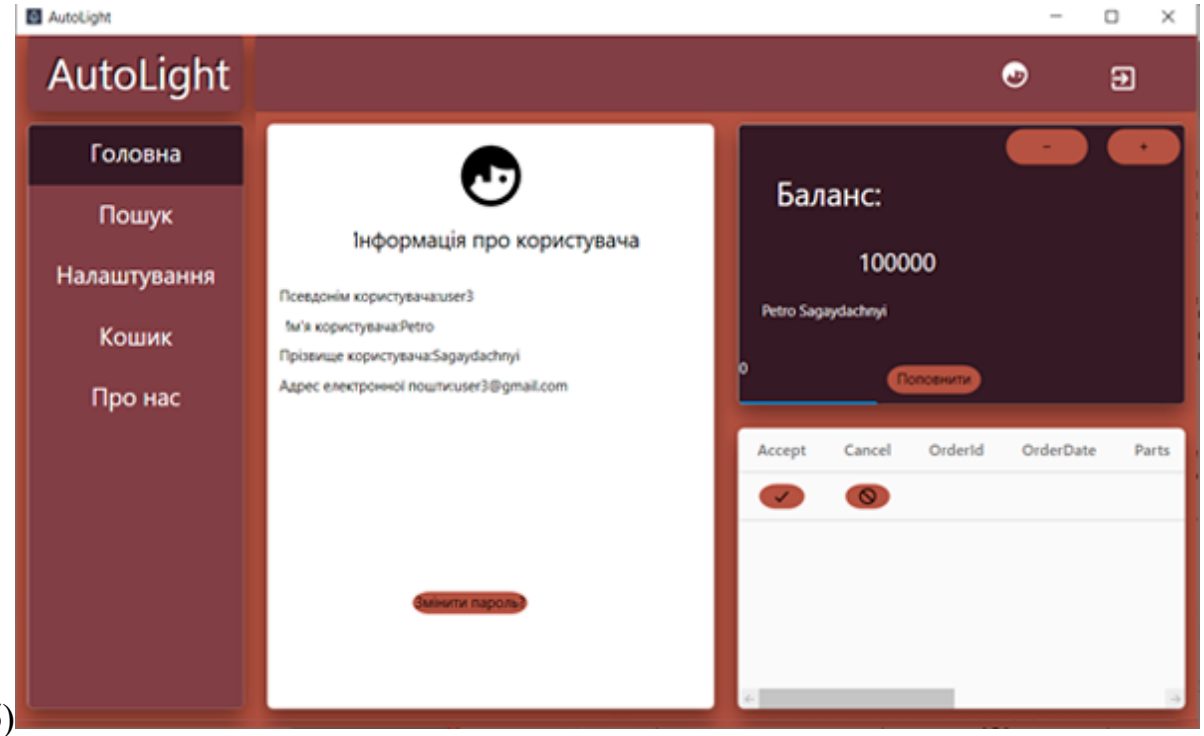
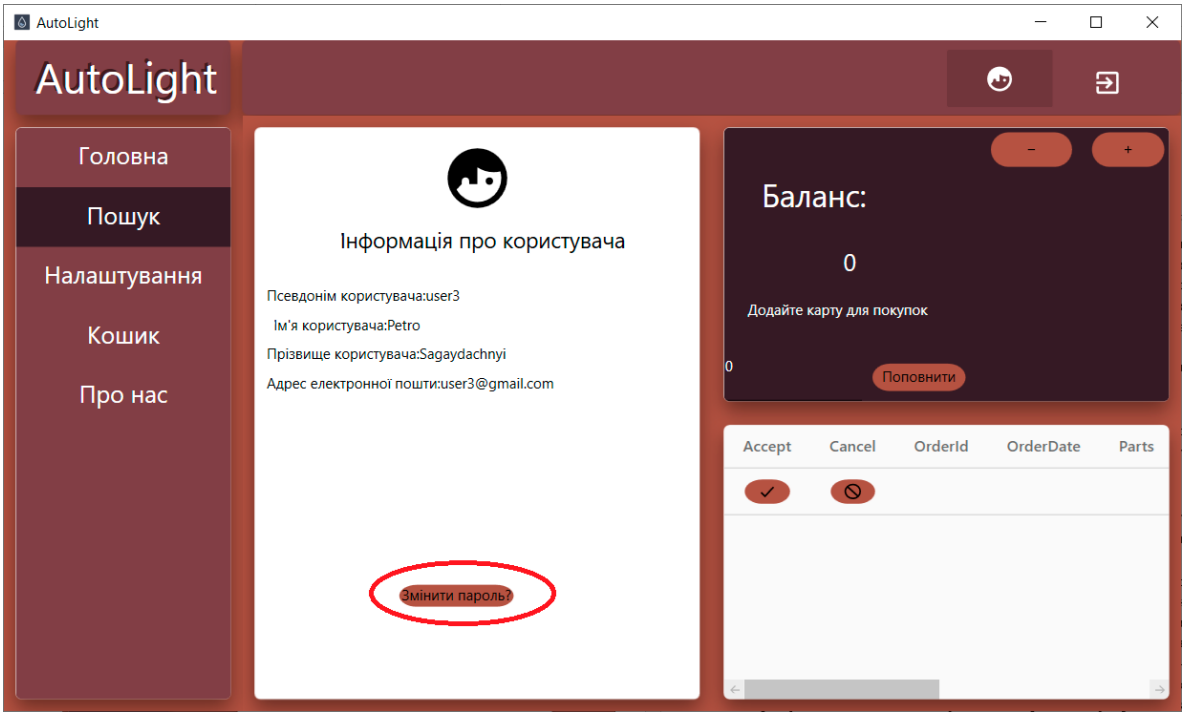


Рисунок 3.28 – а)процес прив’язки банківської карти, б)карта успішно прив’язана

Користувач має можливість змінити пароль свого акаунту (зображено на рис. 3.29 а). Для цього слід натиснути відповідну кнопку, після чого система запропонує два поля для введення та підтвердження нового пароля (зображено на рис. 3.29 б).

а)



б)

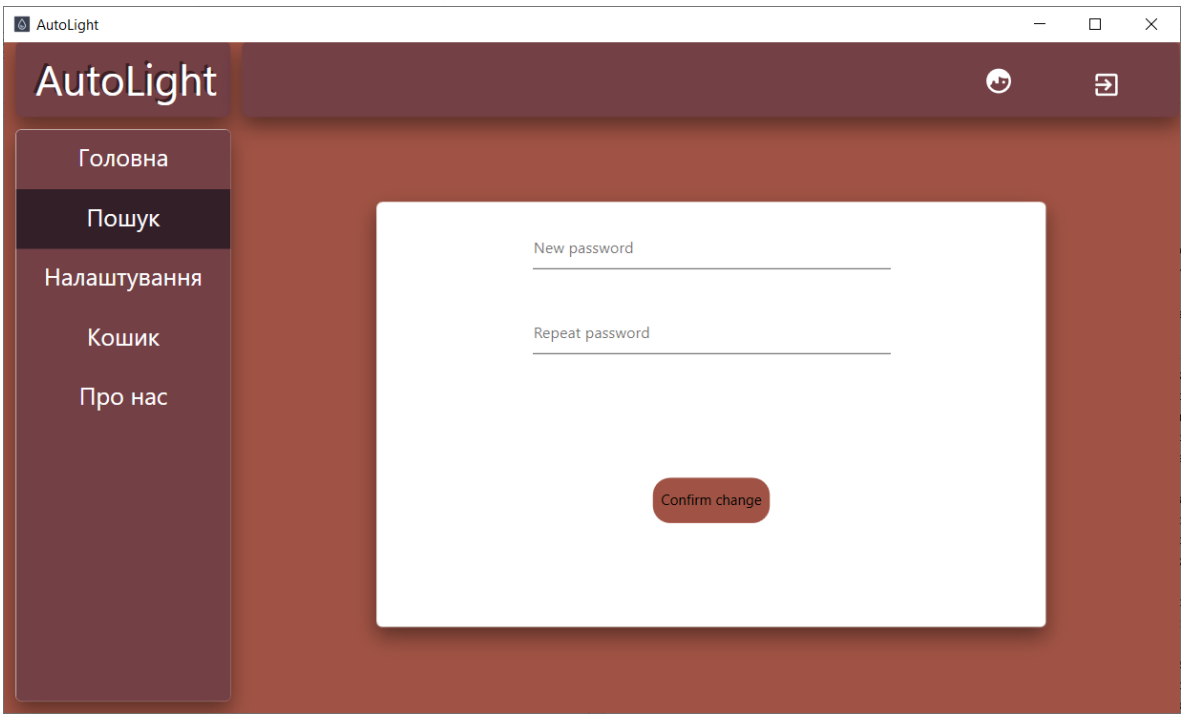


Рисунок 3.29 – а)кнопка для переходу в форму зміни паролю б) форма зміни паролю

Висновок до розділу

У даному розділі розглянуто загальну архітектуру застосунку, що дає уявлення про принцип взаємодії між компонентами системи та організацію програмного коду за патерном MVVM.

Описано структуру застосунку та визначено набір функціональних можливостей для кожної ролі. Схема основних сторінок та переходів між ними закладає основу для розроблення інтерфейсу системи в цілому.

Реалізований інтерфейс застосунку демонструє зовнішній вигляд ключових екранів та вікон. Наведено опис кожного розділу з відповідними скриншотами.

У підрозділі функціонування застосунку розглянуто основні операції, доступні користувачам, порядок їх виконання та очікувані результати.

4 ТЕСТУВАННЯ ТА ВЕРИФІКАЦІЯ ЗАСТОСУНКУ

4.1 Характеристика методів, що застосовуються при перевірці застосунку

Тестування, як завершальний етап розробки застосунку, відіграє життєво важливу роль у процесі створення якісного програмного забезпечення. Чим складніше застосунок, тим більше часу потрібно на його перевірку та налаштування і налагодження [27].

Починається все з підготовчих робіт – тестувальник вивчає отриману документацію: аналізує вимоги технічного завдання, опрацьовує фінальні макети застосунку і складає план тесту для подальшого тестування) [28].

Функціональне тестування є найбільш трудомістким етапом перевірки застосунку. Його суть цього процесу полягає у перевірці всього описаного функціоналу:

- перевірки роботи всіх обов'язкових функцій застосунку;
- тестування працездатності користувальницьких форм в застосунку ;
- перевірки роботи фільтрів і пошуку;
- перевірки навігаційних посилань, пошук неробочих посилань;
- перегляд на відповідність вмісту вкладок застосунку вихідного контенту, наданого замовником.

Usability тестування – проводиться для оцінки зручності продукту у використанні, що базується на залученні реальних користувачів як тестувальників та аналізі отриманих результатів[29].

Тестування безпеки – на даній стадії тестування перевіряється чи немає у користувачів доступу до службових чи закритих сторінок, а також проводиться перевірка захисту всіх критично важливих сторінок (наприклад, розділу адміністрування застосунку) від зовнішнього впливу.

a. Поняття верифікації та валідації програмного забезпечення

Верифікація та валідація — це взаємопов’язані, але різні за природою види діяльності із забезпечення якості програмного забезпечення. Попри спільну мету

— виявлення помилок — вони відрізняються джерелами перевірюваних властивостей і правил. Верифікація встановлює відповідність між артефактами, що створюються в процесі розробки та супроводу ПЗ, попереднім артефактам або вихідним даним, а також відповідність цих артефактів прийнятим правилам і стандартам. Зокрема, вона перевіряє узгодженість між вимогами стандартів, технічним завданням, проектними рішеннями, вихідним кодом, користувацькою документацією та реальною поведінкою програми [30].

Валідація, своєю чергою, підтверджує відповідність створених або використаних у процесі розробки артефактів реальним потребам користувачів та замовників ПЗ — з урахуванням специфіки предметної галузі та обмежень середовища використання. Ці потреби нерідко залишаються незафіксованими документально, а при їх формалізації перетворюються на опис вимог чи інший артефакт процесу розробки. Співвідношення між валідацією та верифікацією ілюструє рисунок 4.1.

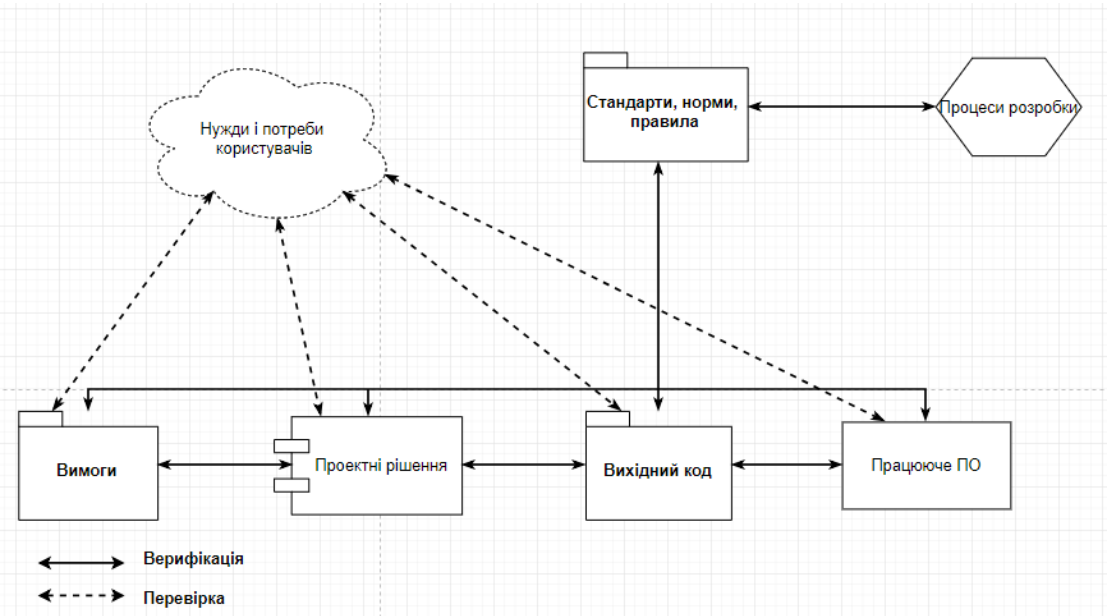


Рисунок 4.1 – Співвідношення валідації та верифікації

Тестування є одним із методів верифікації та полягає у виконанні програми з метою виявлення наявних дефектів і помилок.

Жоден із сучасних методів тестування ПЗ не дозволяє абсолютно повно виявити всі дефекти або з абсолютною впевненістю підтвердити коректність

роботи програми. Тому всі наявні підходи до тестування функціонують у межах формалізованого процесу перевірки досліджуваного або розроблюваного ПЗ.

Формальна перевірка або верифікація здатна довести лише відсутність дефектів у межах застосованого методу. Повністю гарантувати відсутність помилок у програмному продукті неможливо, оскільки людський фактор присутній на всіх етапах його життєвого циклу.

Існує широкий спектр підходів до тестування та верифікації ПЗ, однак ефективна перевірка складних програмних продуктів — це процес, що не зводиться лише до виконання жорстко регламентованих процедур або їх розробки.

Модульне (unit) тестування — перевіряється найменший можливий компонент системи: окремий клас або функція. Як правило, цей вид тестування виконується безпосередньо розробниками ПЗ.

Системне тестування — перевіряється цілісна інтегрована система на відповідність усім висунутим до неї вимогам.

Одним із ключових підходів у даному контексті є стратегія «чорного ящика» — тестування з управлінням за даними або за принципом вхід-вихід. За цієї стратегії програма розглядається як закрита система без доступу до її внутрішньої реалізації. Метою є виявлення ситуацій, за яких поведінка програми відхиляється від її специфікації. Тестові дані формуються виключно на основі специфікації, без урахування внутрішньої структури програми.

За такого підходу критерієм вичерпного вхідного тестування вважається виявлення усіх можливих помилок. Оскільки досягнення повного покриття практично неможливе, метою стає максимізація ефективності тестування — тобто виявлення якомога більшої кількості дефектів одним тестом.

Оскільки застосунок інтернет-магазину автозапчастин не належить до систем критично важливої інфраструктури, до нього не висуваються підвищені вимоги щодо надійності та відмовостійкості, а також не потрібне додаткове тестування всіх можливих нормальних і аварійних сценаріїв. Основним завданням верифікації є підтвердження виконання функціональних вимог. Переважну більшість вимог

можна перевірити методом функціонального тестування за принципом «чорного ящика».

Верифікація виконується за допомогою чек-листа, складеного на основі вимог технічного завдання. Чек-лист містить стислі коментарі щодо способу реалізації кожної вимоги в застосунку та методу її перевірки. Форму чек-листа наведено в таблиці 4.1, де відображено опис реалізації кожної вимоги та висновок щодо її виконання.

Таблиця №4.1 – Чек-лист

№ п/п	Вимога з технічного завдання	Реалізація вимоги в системі	Метод верифікації	Виконання
1	Запуск застосунку на комп’ютері	Запуск застосунку на комп’ютері	Функціональне тестування	True
2	Відображення доступних товарів	Відобразилися всі доступні товари	Функціональне тестування	True
3	Фільтрування(відображення) товарів за обраними категоріями	Відображаються лише товари обраних категорій	Функціональне тестування	True
4	Відкриття в застосунку вікна «Про нас»	Відкрилось вікно «Про нас»	Функціональне тестування	True

Продовження таблиці № 4.2

5	Використання аналітичного фільтру ціни	Відображаються товари в діапазоні	Функціональне тестування	True
6	Реєстрація акаунту нового користувача	Відкрилось вікно з реєстрацією, після успішної реєстрації нового користувача додано до системи	Функціональне тестування	True
7	Авторизація в акаунті користувача	При зупуску програми відкрилось вікно авторизації.Після успішної авторизації користувач потрапив у свій акаунт	Функціональне тестування	True
8	Додавання товарів в кошик	Вибрані товари відображаються в кошику	Функціональне тестування	True
9	Видалення товару з кошика	Товар було успішно видалено з кошика	Функціональне тестування	True
10	Спроба додати товар без заповнення необхідної інформації про неї	У вікні додавання товару отримана помилка заповнення усіх необхідних даних	Функціональне тестування	False
11	Спроба зайти у неіснуючий акаунт користувача	У вікні авторизації отримана помилка входу	Функціональне тестування	False

b. Тестування застосунку

Розроблений застосунок пройшов комплексне тестування, що включало: перевірку навігаційних посилань, перевірку відповідності контенту на вкладках, перевірку відповідності застосунку індивідуальному завданню,перевірка на зрозумілістьінтерфейсу та коректність ведення даних.

Перевірка інформаційного наповнення підтвердила відповідність текстового контенту заданій тематиці, відсутність орфографічних, стилістичних та пунктуаційних помилок.

За результатами usability-тестування встановлено, що застосунок є зручним у використанні,має просте та інтуїтивно зрозуміле меню, приємну кольорову гаму та нескладну логічну структуру.

Тестування запуску застосунку (зображено на рисунку 4.3).

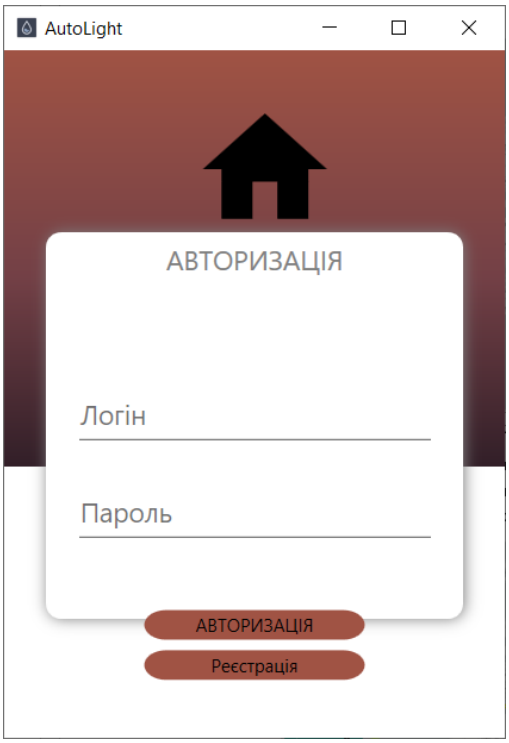


Рисунок 4.2 – Тестування запуску застосунку

Перевірено роботу фільтрації товарів за категоріями. Відфільтровані позиції коректно відображаються в інтерфейсі, що підтверджує рисунок 4.3.

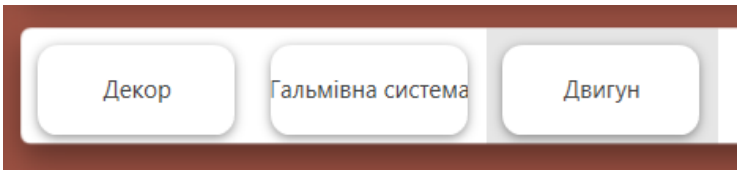




Рисунок 4.3 – Тестування фільтрації товарів за категоріями

Додатково протестовано роботу цінового фільтру. В результаті в каталозі відображаються лише ті товари, що відповідають заданому ціновому діапазону (зображено на рис. 4.4).

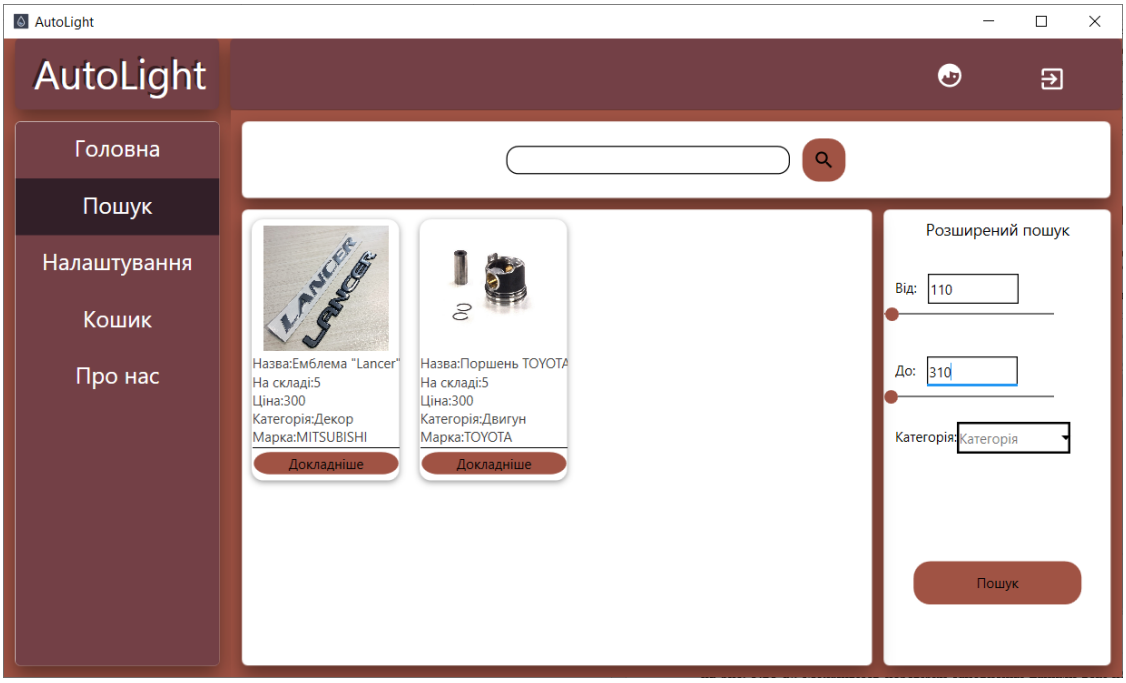


Рисунок 4.4 – Тестування цінового фільтру каталогу автозапчастин

Протестовано процес реєстрації нового користувача (зображено на рис. 4.5 а), перевірку валідації обов'язкових полів форми (зображено на рис. 4.5 б), а також поведінку системи при введенні некоректних або неіснуючих облікових даних під час авторизації (зображено на рис. 4.5 в).

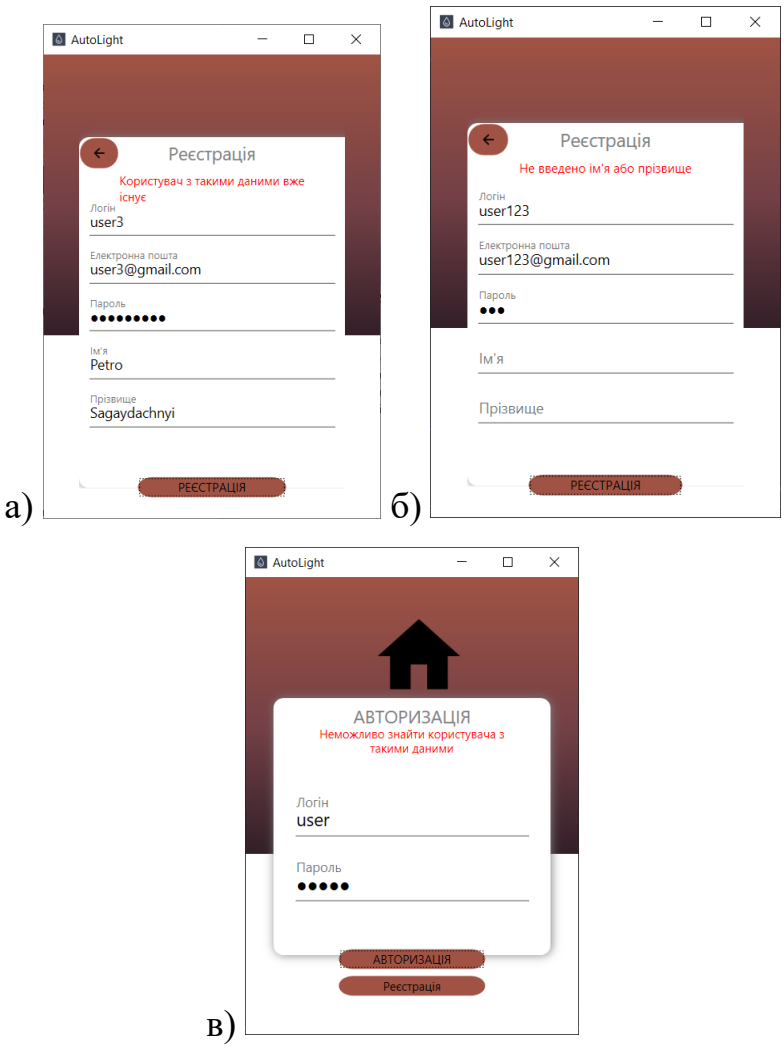
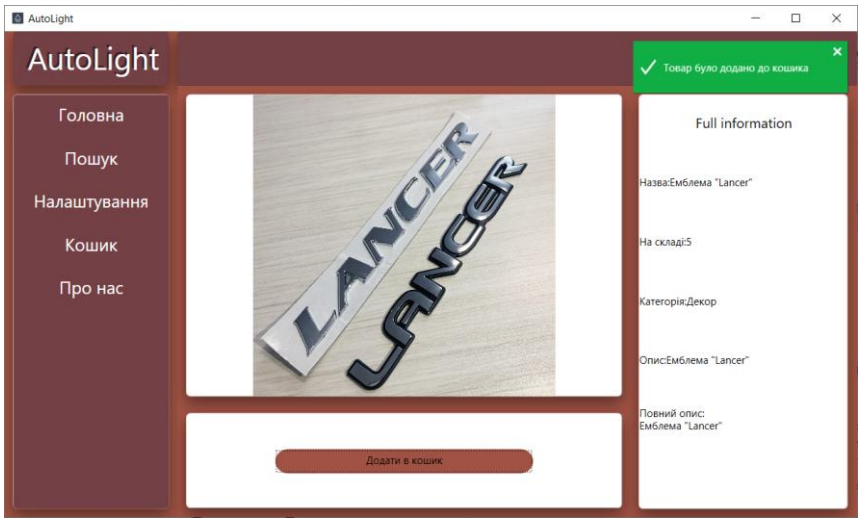


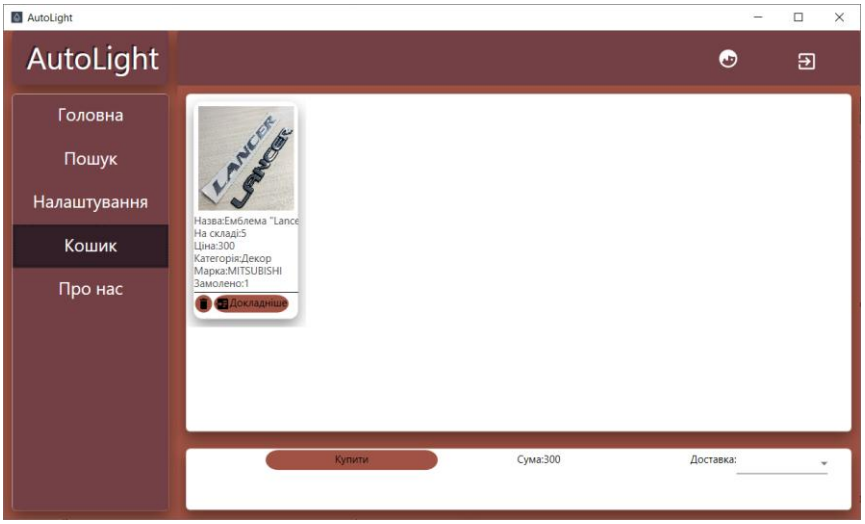
Рисунок 4.5 – а) Тестування реєстрації нового користувача; б) Тестування валідації заповнення усіх необхідних полів; в) Тестування входу у акаунт, використовуючи некоректні данні

Протестовано функції додавання та видалення товару з кошика. Для додавання необхідно натиснути кнопку «Додати в кошик» (зображено на рис. 4.6 а), для видалення — скористатися відповідною іконкою (зображено на рис. 4.6 б).

а)



б)



в)

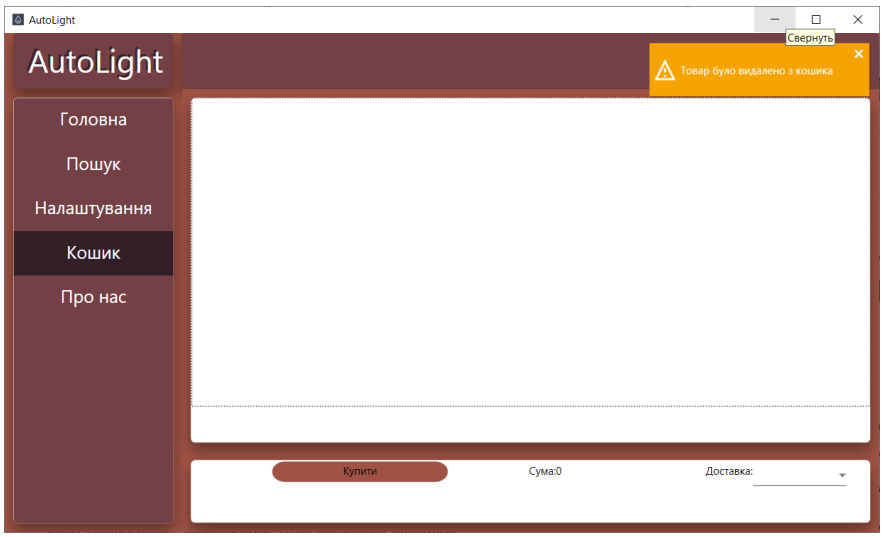


Рисунок 4.6 – а) Додавання товару в кошик; б) Перевірка наявності товару в кошику; в) Видалення товару з кошика

На рис. 4.7 зображено результат спроби адміністратора зберегти товар без заповнення всіх обов’язкових полів..

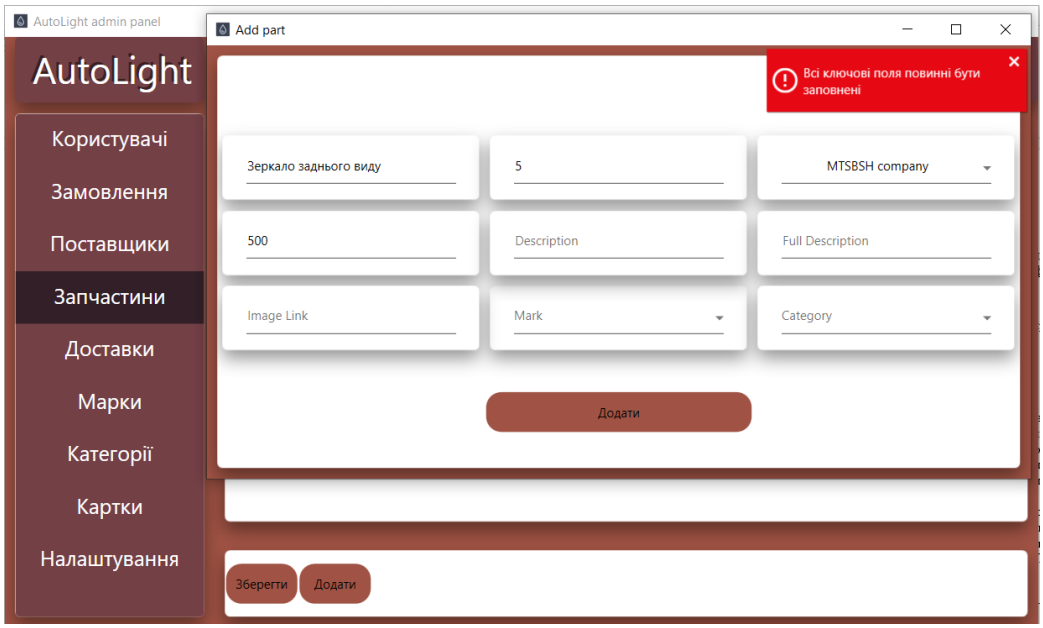


Рисунок 4.7– Спроба додати товар без заповнення всієї необхідної інформації

Під час перевірки навігаційних посилань не виявлено ніяких помилок у роботі навігаційних посилань. Всі гіперпосилання ведуть на потрібні сторінки. Перевірка вертикального меню, та інших кнопок з навігації коректно працюють.

Тестування застосунків призначено для того, щоб зробити процес користування ними приємніше і облегчити процес купівлі – отже, тестування має велике значення. Воно являє собою поєднання людського управління і можливостей комп'ютера. Тестування дозволить вам створити якісний і оптимізований додаток.

Висновок до розділу

У розділі проведено тестування та верифікацію застосунку, охарактеризовано основні методи перевірки. За результатами тестування підтверджено працездатність ключових функцій застосунку, інтуїтивність та зручність та зручність інтерфейсу, а також коректне відображення застосунку в найбільш популярних браузерах. Після проведених тестувань можна зробити висновок, що загалом застосунок працює коректно та відповідає поставленим до нього вимогам.

ВИСНОВКИ

У даній роботі проведено аналіз існуючих рішень у сфері інтернет-торгівлі автозапчастинами, зокрема таких платформ як: Exist.ua та Autoclad.ua. Результатом аналізу стало порівняння зазначених систем за визначеними критеріями та загальними показниками (табл. № 1.1).

У роботі також визначено та обрано засоби розробки: мову програмування C#, технологію WPF, платформу .NET Framework, бібліотеку Entity Framework та СУБД Microsoft SQL Server, необхідні для реалізації застосунку інтернет-магазину автозапчастин.

Спроєктовано алгоритми функціонування системи як в цілому, так і окремих її ключових компонентів. Побудовано UML-діаграму варіантів використання, де визначено ролі в системі та доступний їм функціонал. Для детального опису роботи основних функцій розроблено діаграму послідовності дій та таблицю варіантів використання.

Реалізовано архітектуру застосунку на основі патерну MVVM, розроблено інтерфейси основних вікон та елементів. Описана структура застосунку дає чітке розуміння функціоналу, доступного для кожної ролі користувача.

Розглянуто функціонування застосунку з точки зору кінцевого користувача: описано дії, необхідні для отримання доступу до різних функцій, та очікувані результати їх використання.

Проведено тестування та верифікацію застосунку із застосуванням методу «чорного ящика». Результати підтвердили працездатність основних функцій, інтуїтивність та зручність та зручність інтерфейсу для користувача, а також коректне відображення в різних середовищах. За результатами тестування можна стверджувати, що застосунок працює коректно та відповідає поставленим до нього вимогам.

У подальшому планується розширення функціоналу застосунку в частині обробки замовлень, зокрема реалізація надсилання підтверджень замовлень на електронну пошту користувача.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Exist.ua, український сайт на якому продаються автозапчастини. URL- https://exist.ua/uk/ (дата звернення 26.02.2026)
2. Autoclad.ua, український сайт на якому продаються автозапчастини . URL - https://www.autoklad.ua/ua/ (дата звернення 27.02.2026)
3. Avto.pro, український сайт на якому продаються автозапчастини. URL - https://avto.pro (дата звернення 28.02.2026)
4. Mapco.com, німецький сайт на якому продаються автозапчастини. URL - https://www.mapco.com (дата звернення 01.03.2026)
5. Racuten.com, японський сайт на якому продаються автозапчастини. URL - https://www.rakuten.com/ (дата звернення 03.03.2026)
6. Imsautoparts.com, , американський сайт на якому продаються автозапчастини. URL - https://imsautoparts.com/ (дата звернення 05.03.2026)
7. Посилання на статтю: «C#: що за мова, та де її використовують». URL- https://robotdreams.cc/uk/blog/284-s-chto-eto-za-yazyk-i-gde-ego-ispolzuyut (дата звернення 11.03.2026)
8. Посилання на статтю «Посібник по WPF»- https://metanit.com/sharp/wpf/ (дата звернення 12.03.2026)
9. Посилання на сайт статтю «Початок роботи з Visual Studio» .URL - https://visualstudio.microsoft.com/ru/vs/getting-started/(дата звернення 13.03.2026)
10. Посилання на статтю «Кращі IDE для програмування на C#» URL - https://deveducation.com/blog/luchshie-ide-dlya-razrabotki-na-c/ (дата звернення 14.03.2026)
11. Посилання на статтю «Welcome to VisualStudio IDE|C#». URL - https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/visual-studio-ide?view=vs-2022 (дата звернення 15.03.2026)

12. Посилання на статтю «Створення застосунку WPF на C#». URL - https://learn.microsoft.com/ru-ru/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2022 (дата звернення 16.03.2026)
13. Посилання на статтю «Основні відомості про бази даних ». URL - https://support.microsoft.com/uk-ua/office/основні-відомості-про-бази-даних-a849ac16-07c7-4a31-9948-3c8c94a7c204 (дата звернення 17.03.2026)
14. Посилання на статтю «5 найкращих систем управління базами даних ». 2015. URL - https://cloud.itstep.org/blog_3/5-best-database-management-systems-in-2022(дата звернення 18.03.2026)
15. Сайт Microsoft SQL . URL - https://www.microsoft.com/ru-ru/sql-server/sql-server-2019 (дата звернення 19.03.2026)
16. Посилання на статтю «Типи баз даних: особливості, відмінності ». URL - https://dou.ua/lenta/articles/types-of-databases/ (дата звернення 20.03.2026)
17. Посилання на статтю «Взаємодія між формами WPF » .URL - https://metanit.com/sharp/wpf/20.2.php (дата звернення 24.03.2026)
18. Посилання на статтю «Підключення бази даних MSSQL ». URL https://metanit.com/sharp/mvc/5.2.php (дата звернення 28.03.2026)
19. Посилання на статтю « Валідація даних WPF ». URL - https://metanit.com/sharp/wpf/14.php/ (дата звернення 01.04.2026)
20. Посилання на статтю « Керівництво користувача.Локалізація застосунку». URL - https://learn.microsoft.com/ru-ru/dotnet/desktop/wpf/advanced/how-to-localize-an-application?view=netframeworkdesktop-4.8 (дата звернення 03.04.2026)
21. Посилання на статтю «Форма для реэстрації WPF» URL - https://itproger.com/ua/course/csharp-app/5/ (дата звернення 05.04.2026)
22. Visual studio code, code editing. URL - https://code.visualstudio.com/ (дата звернення 07.04.2026)
23. СУБД Microsoft SQL Server. URL - https://www.microsoft.com/en-us/sql-server (дата звернення 10.04.2026)

24. Стаття на тему “Тестування та валідація”. URL - https://qalight.ua/baza-znaniy/verifikatsiya-ta-validatsiya/ (дата звернення 21.04.2026)
25. Стаття “як проходить тестування веб-сайту”. URL - https://qberry.com.ua /ru/blog/kak-prohodit-testirovanie-veb-sayta (дата звернення 22.04.2026)
26. Стаття на тему “Правильне тестування веб-сайту”. URL - https://freshdesign.ua/amp/blog/technology/website-testing (дата звернення 23.04.2026)
27. Usability testing. Чекліст тестування. URL - https://training.qatestlab.com/blog/technical-articles/usability-testing-check-list-testing/ (дата звернення 24.04.2026)
28. IEEE 1012-2004 Standard for Software Verification and Validation. IEEE, 2005.
29. 10 ways to make money as a Web Developer. URL - https://www.coderchamp.com/make-money-as-a-web-developer/(дата звернення 28.04.2026)
30. Project Cost Management: Definition & Best Practices. URL - https://www.forbes.com/advisor/business/project-cost-management(дата звернення 29.04.2026)

ДОДАТОК А. ТЕХНІЧНЕ ЗАВДАННЯ

1 Вступ

1.1 Найменування, шифр роботи та підстави для виконання роботи

Найменування – «Застосунок для інтернет-магазину автозапчастин» (далі по тексту – ПЗ).

1.2 Коротка характеристика області застосування

Дане ПЗ призначене для забезпечення онлайн-продажу автозапчастин через зручний інтерфейс каталогу з можливістю пошуку, фільтрації та оформлення замовлень.

2 Підстава для розробки

2.1 Підстава для проведення розробки

Підстава для розробки ПЗ – завдання на кваліфікаційну роботу кафедри «Комп'ютерної інженерії» Державного університету інформаційно-комунікаційних технологій на тему «Програмно-апаратна архітектура системи електронної комерції для інтернет-магазину автозапчастин»

2.2 Назва та умовне позначення розробки

Назва теми розробки – «Застосунок для інтернет-магазину автозапчастин». Умовне позначення розробки – «APECS». (Auto Parts E-Commerce System)

3 Призначення розробки

3.1 Функціональне призначення розробки

Основним призначенням даного ПЗ є: відображення інформації у зручному графічному форматі, накопичення та зберігання інформації про товари, марки, категорії та іншу інформацію що стосується інтернет-магазину.

3.2 Експлуатаційне призначення

ПЗ розроблене з метою полегшення процесу купівлі запчастин в інтернеті.

Програмний компонент ПЗ поставляється у вигляді застосунку, що включає в себе БД та користувацькі частини, що розміщуються.

4 Вимоги до системи

4.1 Вимоги до функціональних характеристик

4.1.1 Вимоги до складу виконуваних функцій

ПЗ повинна виконувати наступні функції:

- відображення списку товарів при авторизації користувача;
- надавати користувачу опис обраного товару;
- фільтрувати товари за категоріями;
- фільтрувати товари за ціною;
- додавання адміном товарів в БД;
- додання та видалення товарів з кошика;
- редагування всієї інформації в БД адміністратором системи.

4.1.2 Вимоги до організації вхідних даних

Вхідними даними може бути інформація, про товар який додається.

4.1.3 Вимоги до організації вихідних даних

Вихідними даними буде інформація про товари, зображена у вигляді картинок з назвою та описом товарів на формі застосунку. Товари зберігаються у базі даних..

4.1.4 Вимоги до часових характеристик

Вимоги до часових характеристик не пред’являються.

4.2 Вимоги до надійності

ПЗ повинне зберігати працездатність і забезпечувати відновлення своїх функцій при виникненні наступних позаштатних ситуацій:

– при перебоях в системі електропостачання апаратної частини, що призводять до перезавантаження операційної системи, відновлення програмної частини має відбуватися після перезапуску серверу, а відповідної апаратної частини одразу після відновлення електроживлення;

– при помилках в роботі апаратних засобів (крім носіїв даних і програм) відновлення функції системи покладається на операційну систему;

– при помилках, пов’язаних з програмним забезпеченням (операційна система та драйвери пристроїв), відновлення працездатності покладається на операційну систему.

4.2.1 Вимоги до забезпечення сталого функціонування ПЗ

Стійке функціонування ПЗ має бути забезпечене виконанням замовником сукупності організаційно-технічних заходів:

– комплектація технічних засобів джерелом безперебійного ПЗ;

– регулярне виконання вимог нормативних документів, що стосуються захисту інформації шляхом випробування операційної системи на наявність комп’ютерних вірусів;

– організація політики розмежування доступу до ПЗ та її компонентів;

– попереднє навчання користувачів і обслуговуючого персоналу (адміністраторів);

– проведення комплексу заходів налагодження, пошуку та виключення помилок, наприклад, за рахунок ведення системних повідомлень і помилок для подальшого аналізу та зміни конфігурації.

4.2.2 Відмови через некоректні дії оператора

Відмови ПЗ можливі внаслідок некоректних дій адміністратора при взаємодії з операційною системою. Щоб уникнути виникнення відмов програмної частини ПЗ за вказаною вище причиною слід забезпечити попереднє навчання адміністратора та обов’язково організувати політику розмежування доступу до ПЗ, її компонентів та системних файлів.

4.3 Умови експлуатації

4.3.1 Кліматичні умови експлуатації

Кліматичні умови експлуатації, при яких повинні забезпечуватися задані характеристики програми, повинні задовольняти вимогам, що пред’являються до технічних засобів в частині умов їх експлуатації.

ПЗ призначене для експлуатації в виділених приміщеннях (серверних та офісних) умовах.

4.3.2 Вимоги до видів обслуговування

Потребує оновлення адміном інформації про товар, якщо вона змінюється.Це стосується опису товару, назви, зображення товару.

4.3.3 Вимоги до чисельності та кваліфікації персоналу

Кінцевий користувач не повинен володіти спеціальними технічними навичками.

4.4 Вимоги до складу і параметрів технічних засобів

До складу технічних засобів ПЗ повинно входити:

- Database Server (MS-SQL);

4.5 Вимоги до інформаційної та програмної сумісності

4.5.1 Вимоги до інформаційних структур і методів розв’язання

Вимоги до інформаційних структур на вході та виході, а також до методів рішення не пред’являються.

4.5.2 Вимоги до вихідного коду і мов програмування

Для розробки програми в якості мови програмування повинна використовуватися мова програмування для розробки ПЗ, наприклад мова C#.

4.5.3 Вимоги до програмних засобів, які використовуються в ПЗ

Програмна частина ПЗ підключається до бази даних MSSQL.

4.5.4 Вимоги до захисту інформації та програм

Програма не потребує захисту, так як працює на локальній БД.

4.6 Спеціальні вимоги

Система не потребує спеціальних вимог.

5 Вимоги до програмної документації

5.1 Попередній склад програмної документації

В результаті розробки ПЗ повинно бути представлено наступну документацію:

– технічне завдання;
– схеми алгоритмів;
– код програмної частини;
– результати тестування;
– пояснювальна записка;
– інструкція користувача.

6 Стадії та етапи розробки

6.1 Стадії розробки

Розробка повинна проводитися за такими стадіями:
– розробка технічного завдання;
– робоче проектування;
– отримання результатів.

6.2 Етапи розробки

На стадії розробки технічного завдання повинен бути виконаний етап розробки, узгодження та затвердження цього технічного завдання.

На стадії робочого проектування повинні бути виконані перераховані нижче види робіт:
– обґрунтування архітектури ПЗ;
– розробка елементів, та інтерфейсів ПЗ;
– розробка програмної частини;
– тестування та верифікація.

На етапі отримання результатів повинні бути проведені відповідні тести та зроблені висновки про виконану роботу та її результати.

ДОДАТОК Б. ІНСТРУКЦІЯ ДЛЯ КОРИСТУВАЧА

Застосунок призначений для реалізації інтернет магазину автозапчастин. При завантаженні застосунку користувач побачить форму для авторизації (зображено на рис. Б.1), на якій буде зображено поле для вводу логіну, поля для вводу пароля і кнопки: «Вхід» і «реєстрація», якщо у користувача ще немає власного акаунту.

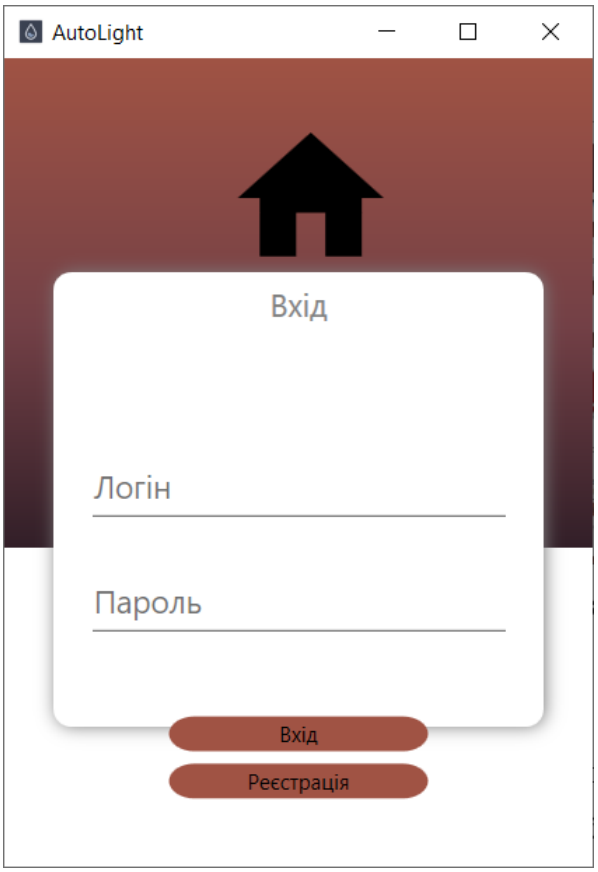


Рисунок Б.1 – Інтерфейс при завантаженні застосунку

Якщо у користувача нема акаунту, то він натискає на «реєстрація» і перехрдить на форму для реєстрації нового акаунту (зображено на рис. Б.2), на якій користувач може ввести дані для реєстрації в застосунку. Якщо дані пройшли валідацію, то користувачу буде створено новий власний акаунт. Після чого користувач знову перейде на форму для авторизації (рис. Б.2)

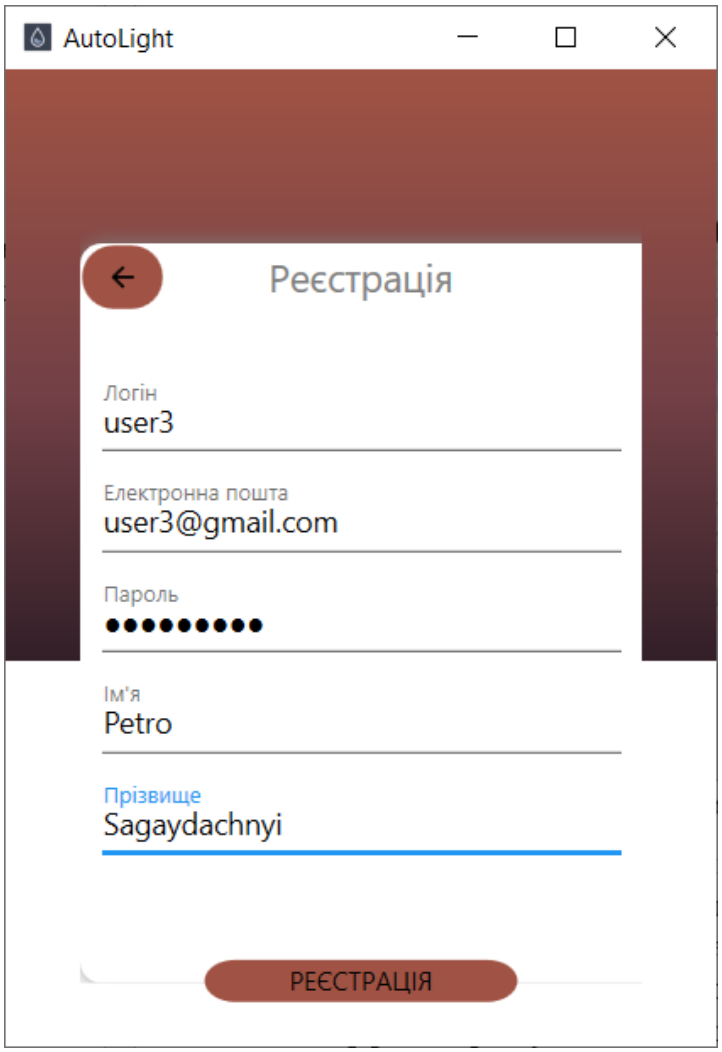


Рисунок Б.2 – Готова до роботи головна сторінка застосунку

Якщо користувач ввів правильні дані свого акаунту, він потрапляє на головну сторінку магазину(інтерфейс зображено на рис. Б.3), де є пошук товару, вибір категорії товару, а також доступні товари. Також користувач може перейти на форму «Пошук» (інтерфейс зображено на рис. Б.4), «Налаштування» (інтерфейс зображено на рис. Б.5), «Кошик» (інтерфейс зображено на рис. Б.6), і «Про нас» (інтерфейс зображено на рис. Б.7), також кнопка «Профіль» (інтерфейс зображено на рис. Б.8), і кнопка «Вихід з акаунту».

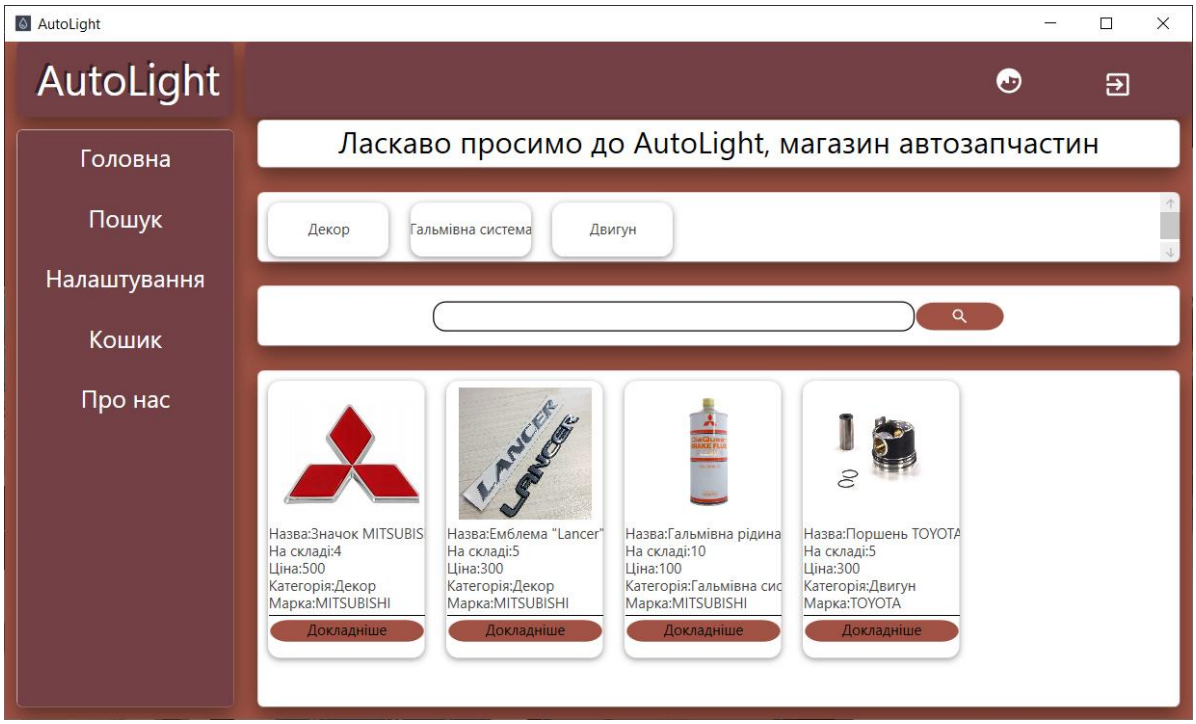


Рисунок Б.3 – Головна сторінка

На вкладці «Пошук» користувач має можливіть скористуватися розширеним пошуком, щоб швидше знайти товар, який йому потрібен.

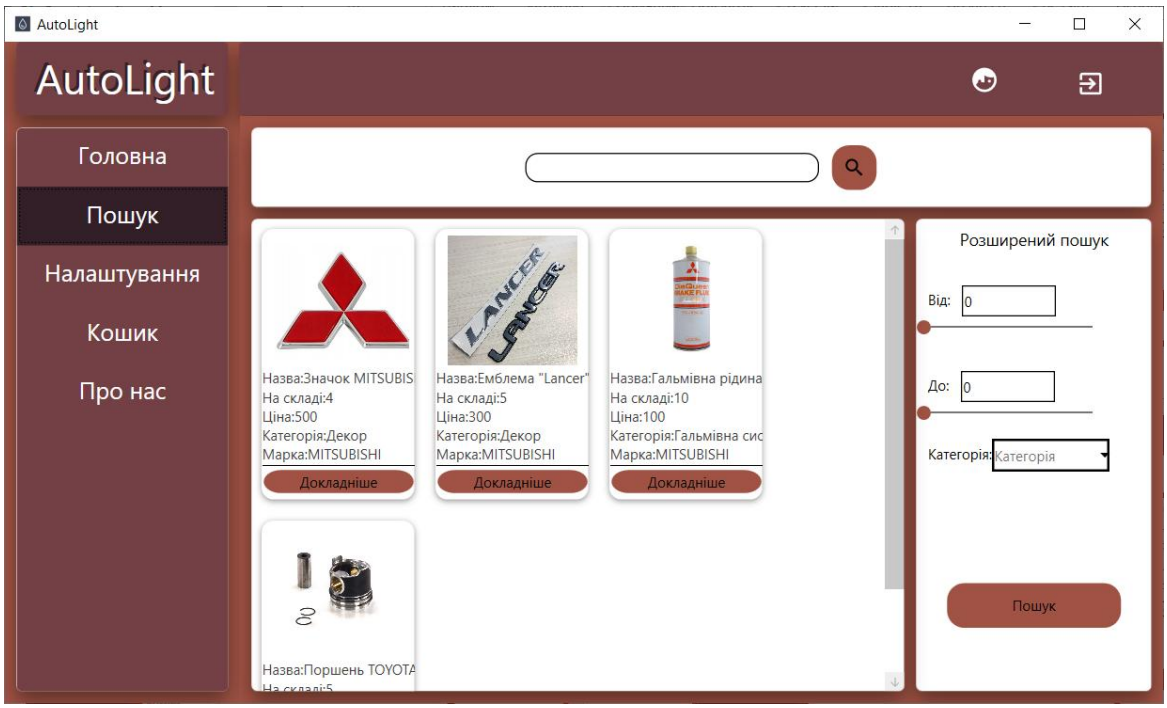


Рисунок Б.4 – Інтерфейс сторінки «Пошук»

На вкладці «Налаштування» користувач має можливіть змінити мову інтерфейсу застосунку. Серед доступних мов: українська і англійська.

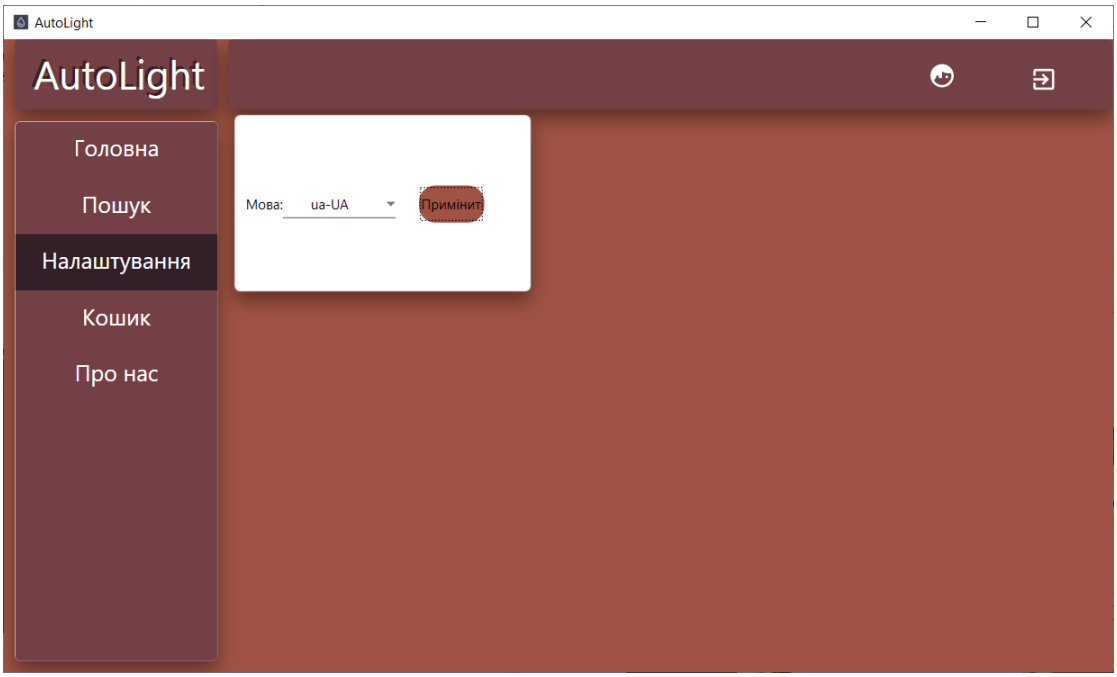


Рисунок Б.5 – Інтерфейс сторінки «Налаштування»

На вкладці «Кошик» користувач має можливіть переглянути товари, які він попередньо додав до нього, видалити товари, а також замовити вміст свого кошика.

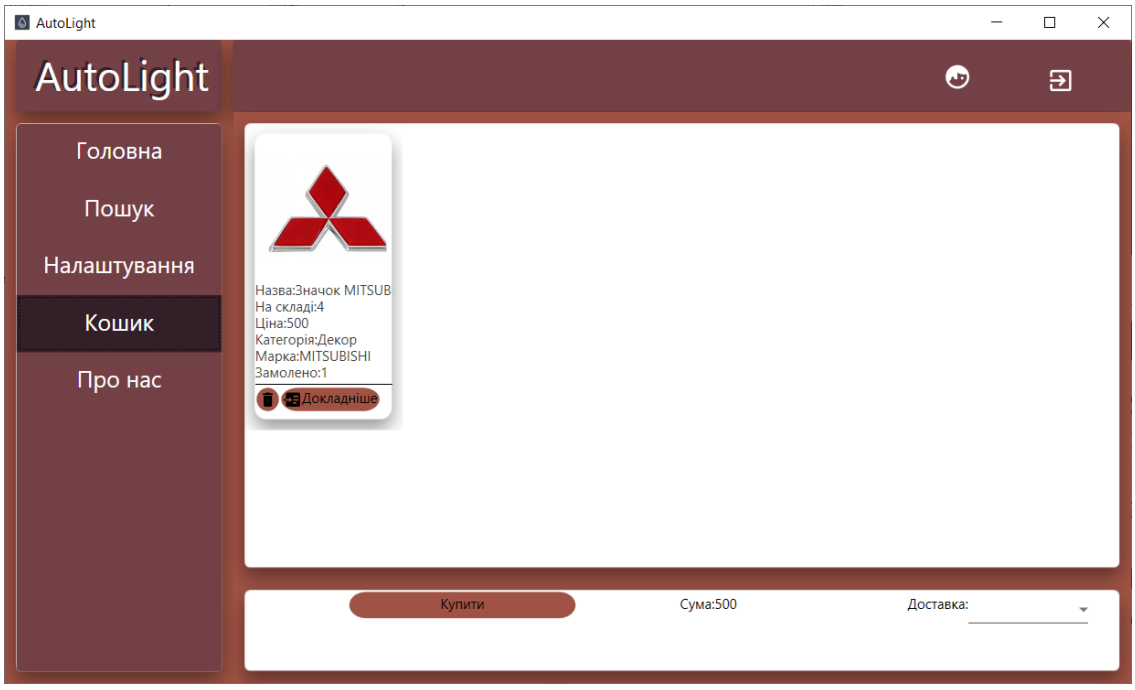


Рисунок Б.6 – Інтерфейс сторінки «Кошик»

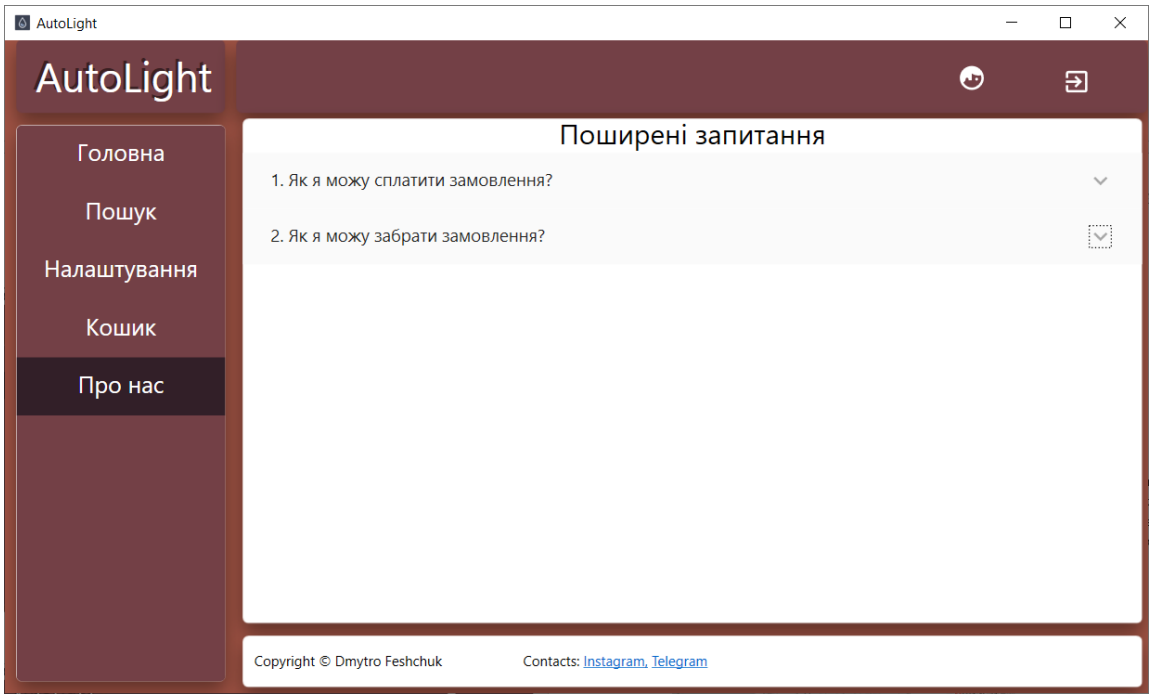


Рисунок Б.7 – Інтерфейс сторінки «Про нас»

На вкладці «Профіль» користувач має можливіть переглянути інформацію про себе, прив’язати банківську карту, відстежувати замовлення, також відміняти їх. Ще доступна функція зміни паролю.

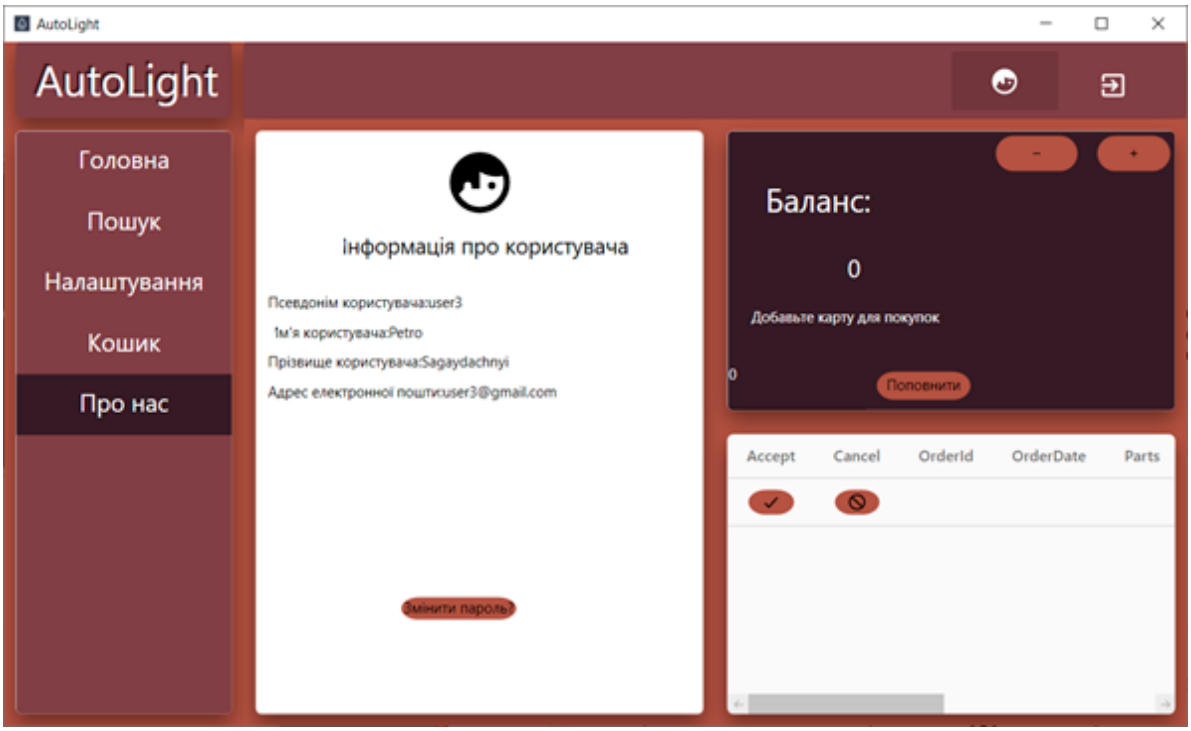


Рисунок Б.8 – Інтерфейс сторінки «Профіль»

Якщо користувач має права адміна, то його інтерфейс буде відрізнятися, від звичайного користувацького.Після входу в акаунт з правами адміністратора буде відкрита стартова форма(рис. Б.9)

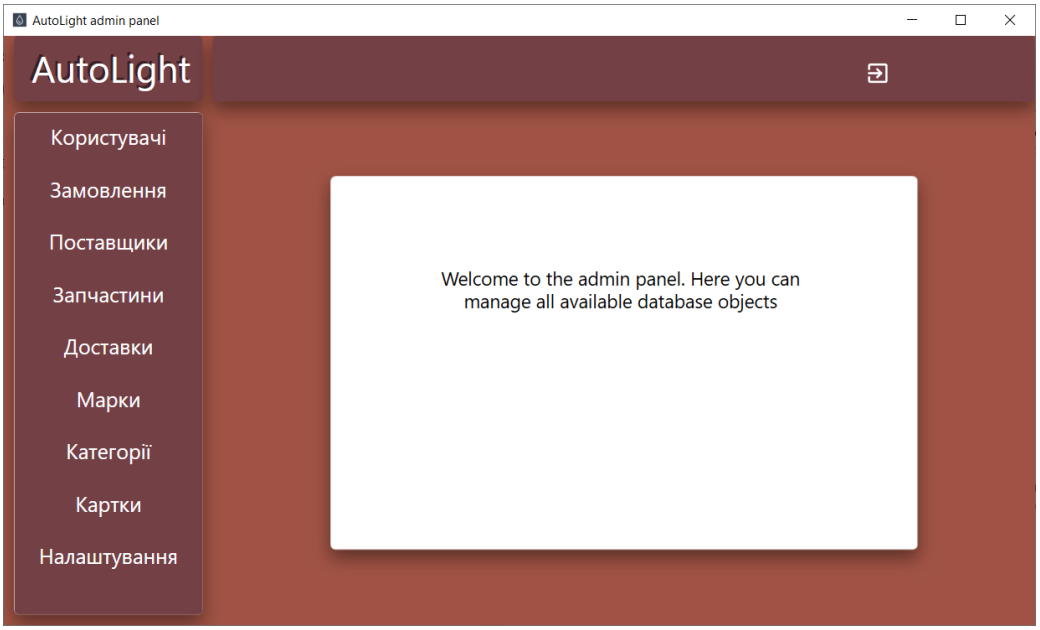


Рисунок Б.9 – Інтерфейс стартової торінки адміна

Користувачу з правами адміна доступні сторінки взаємодії з користувачами(рис Б.10), замовленнями(рис Б.11), поставщиками(рис Б.12), запчастинами(рис Б.13), доставками(рис Б.14), марками машин(рис Б.15), категоріями товарів(рис Б.16), банківськими картами(рис Б.17), і вкладка налаштування. На кожній із сторінок адмін має право редагувати інформацію, додавати, або видаляти.

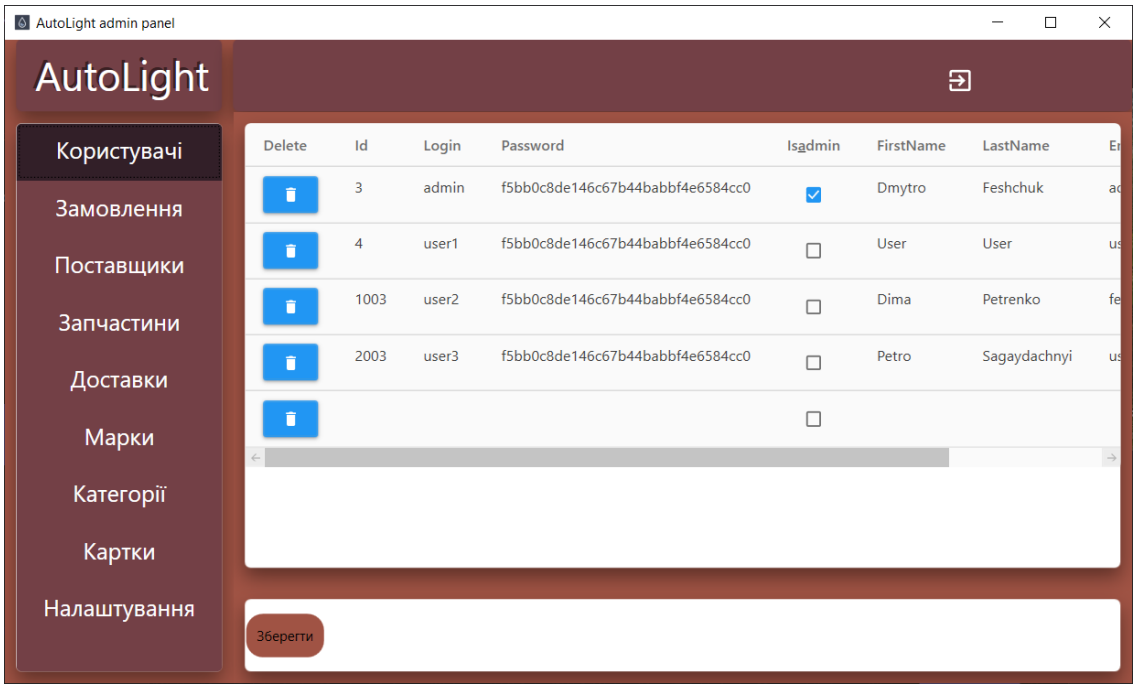


Рисунок Б.10– Інтерфейс сторінки «Користувачі»

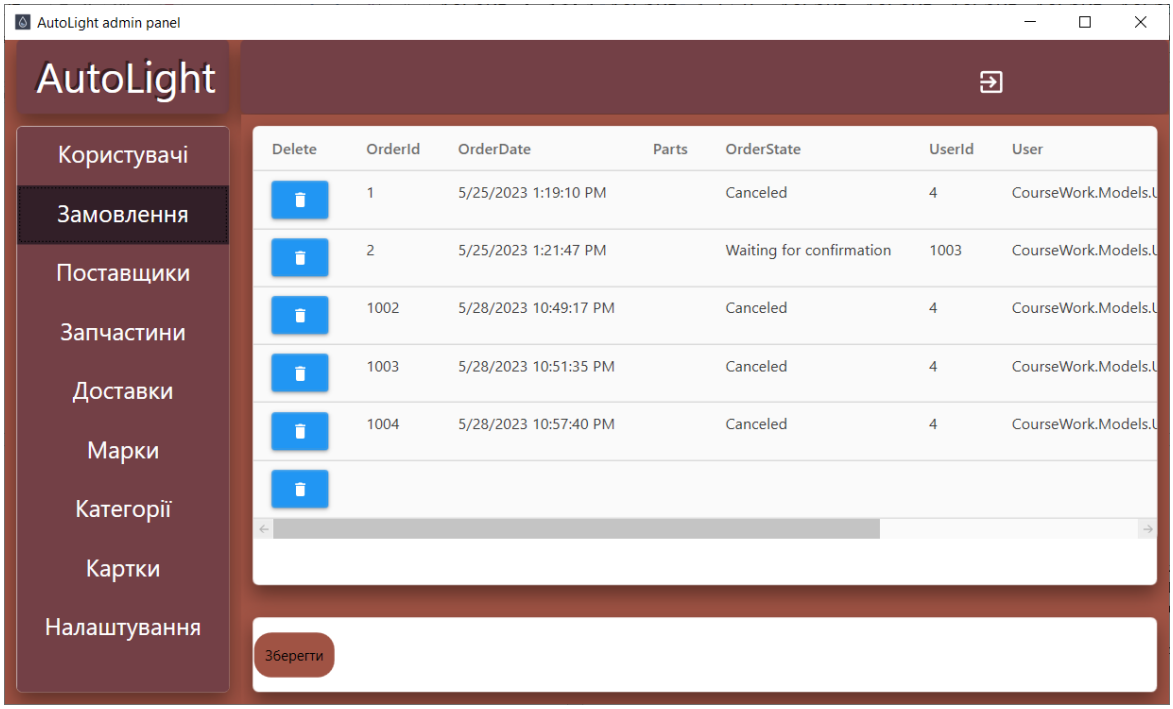


Рисунок Б.11– Інтерфейс сторінки «Замовлення»

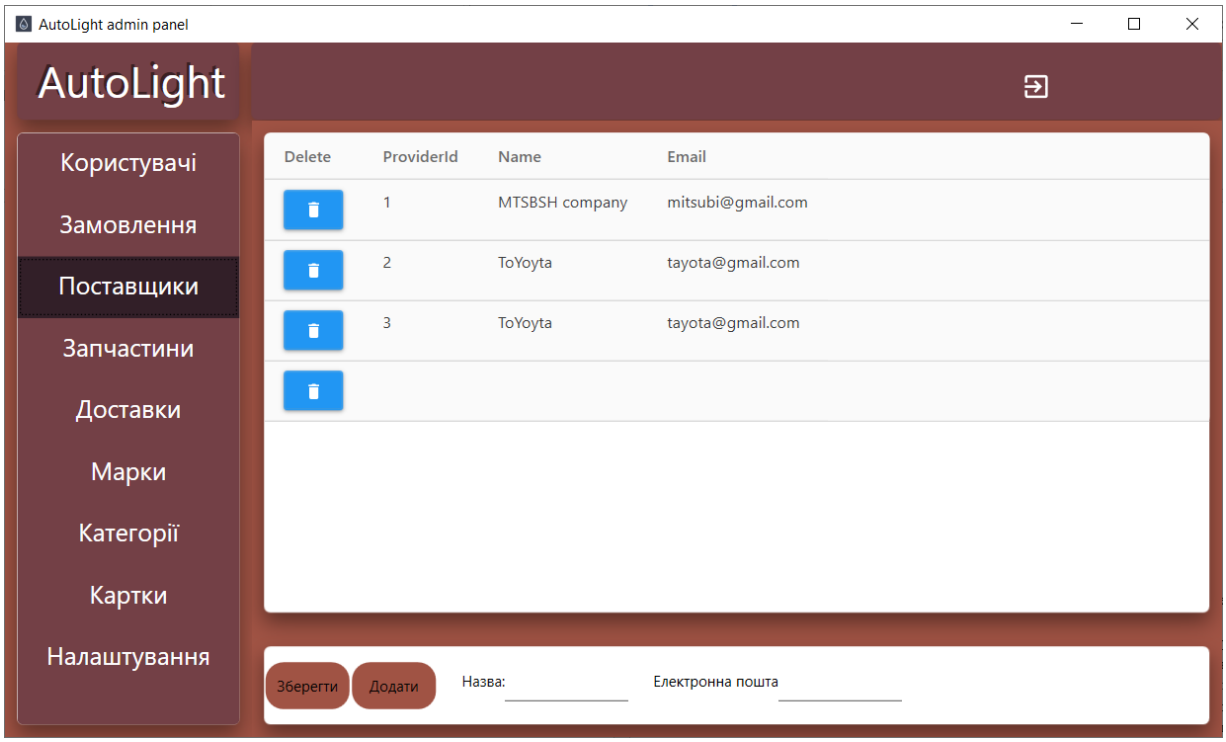


Рисунок Б.12– Інтерфейс сторінки «Поставщики»

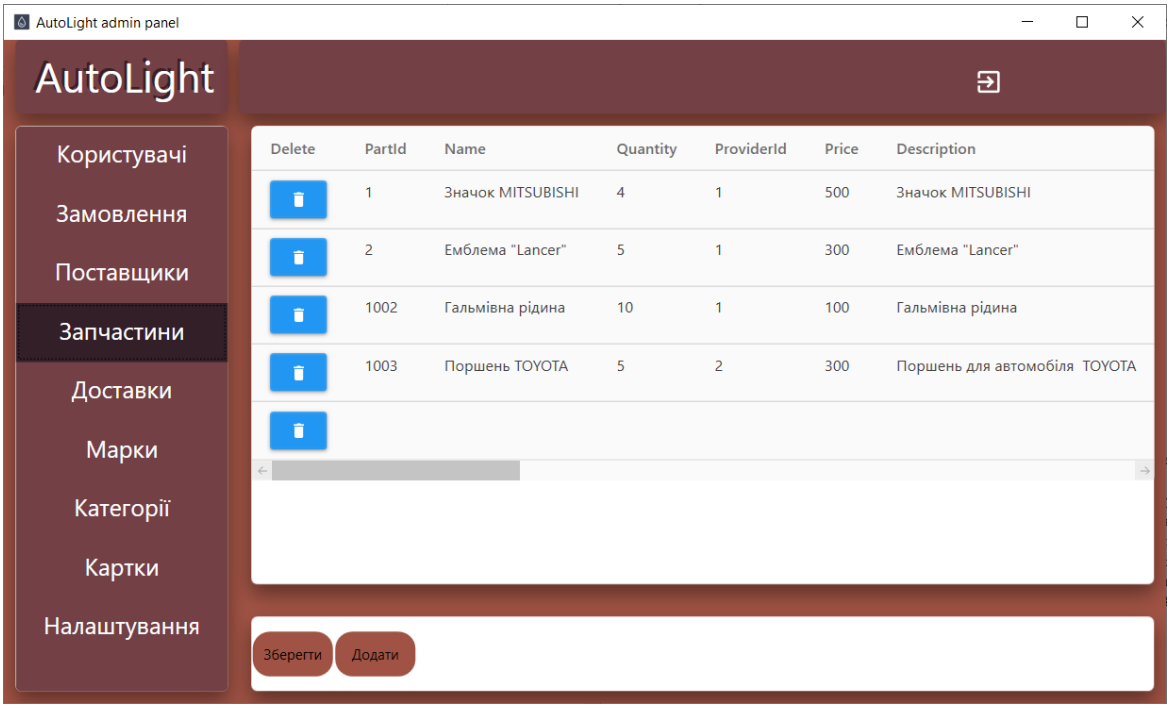


Рисунок Б.13– Інтерфейс сторінки «Запчастини»

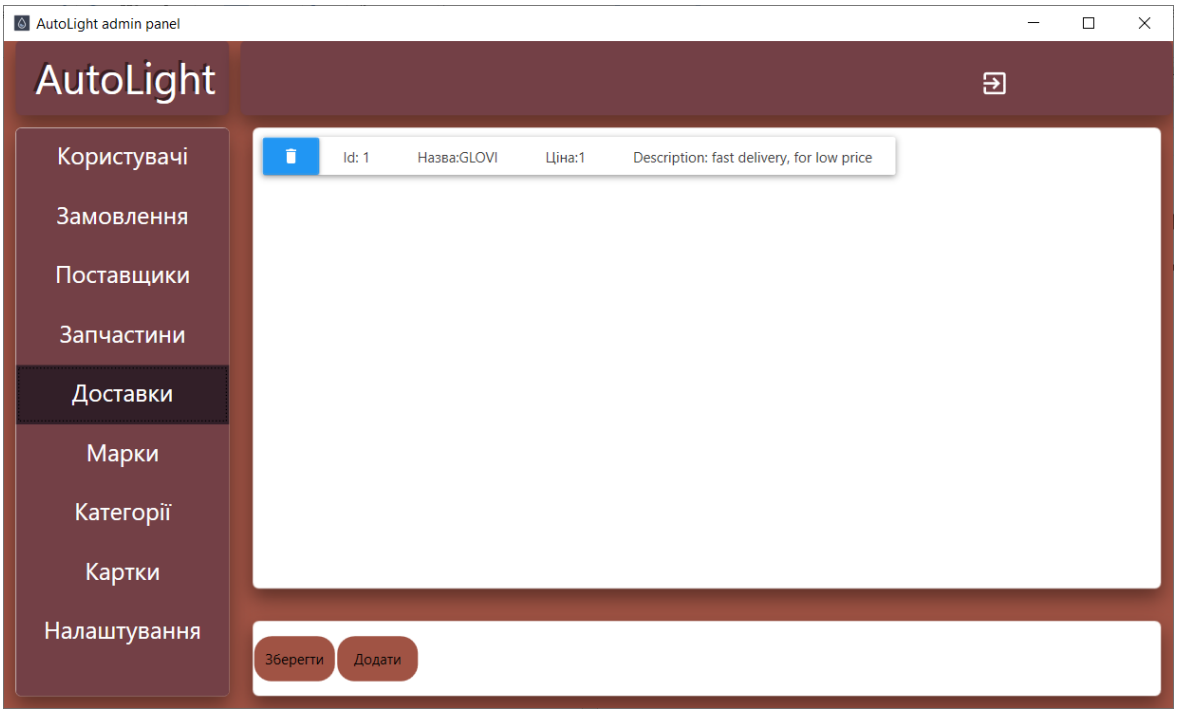


Рисунок Б.14– Інтерфейс сторінки «Доставки»

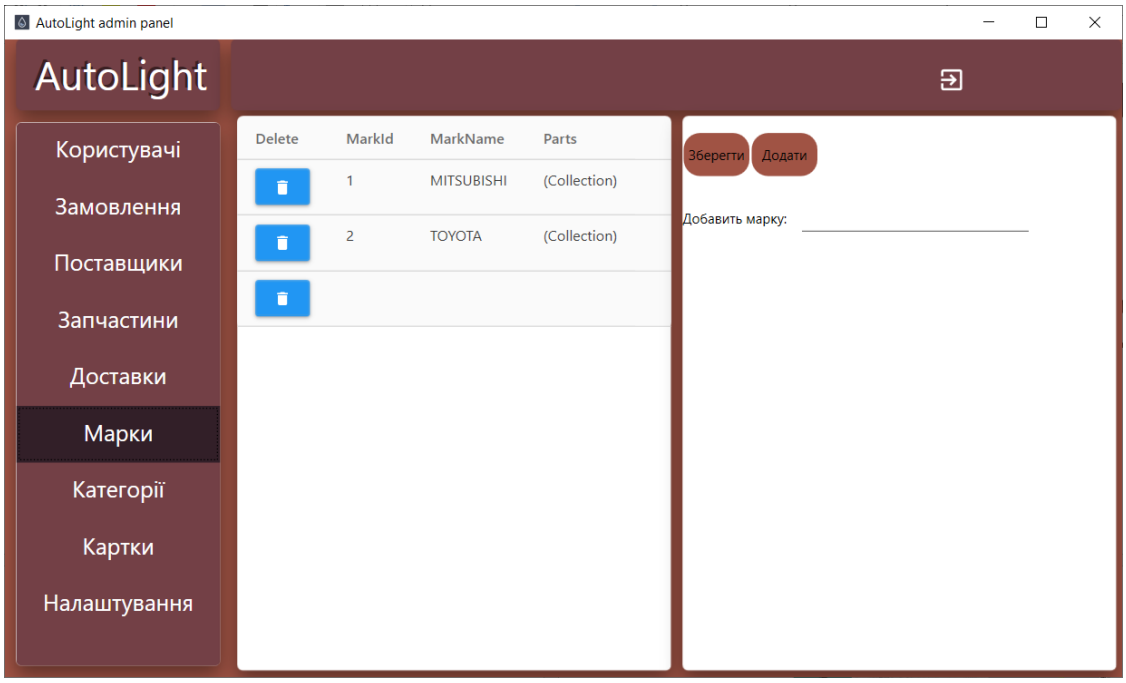


Рисунок Б.15– Інтерфейс сторінки «Марки»

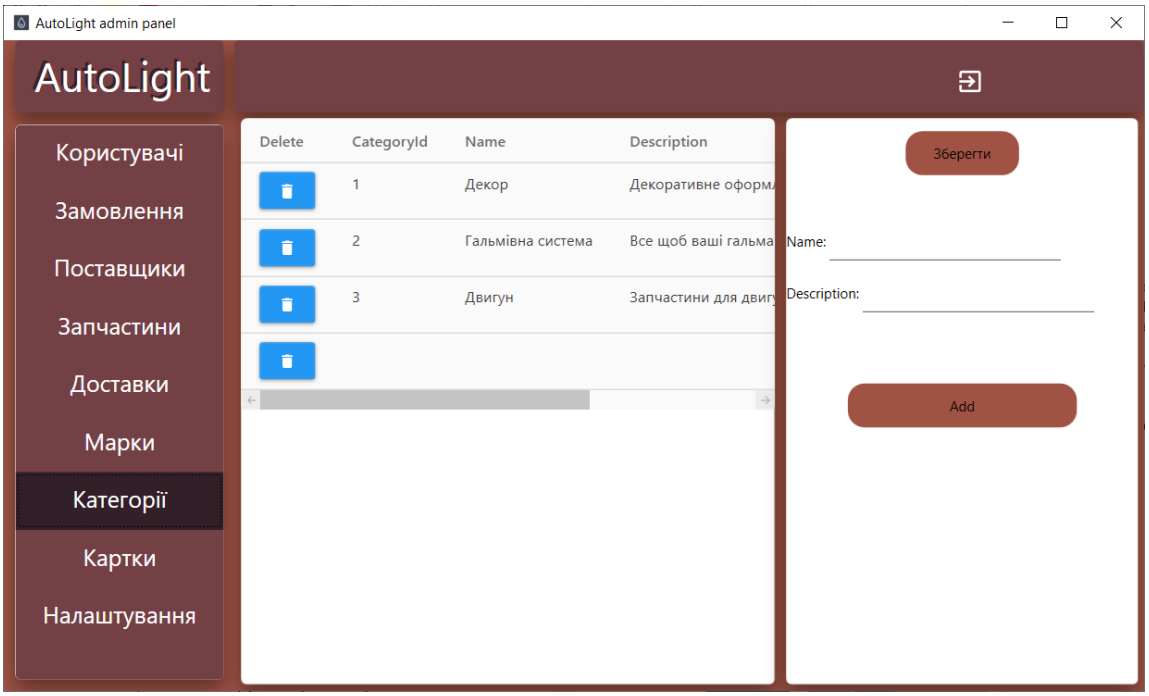


Рисунок Б.16– Інтерфейс сторінки «Категорії»

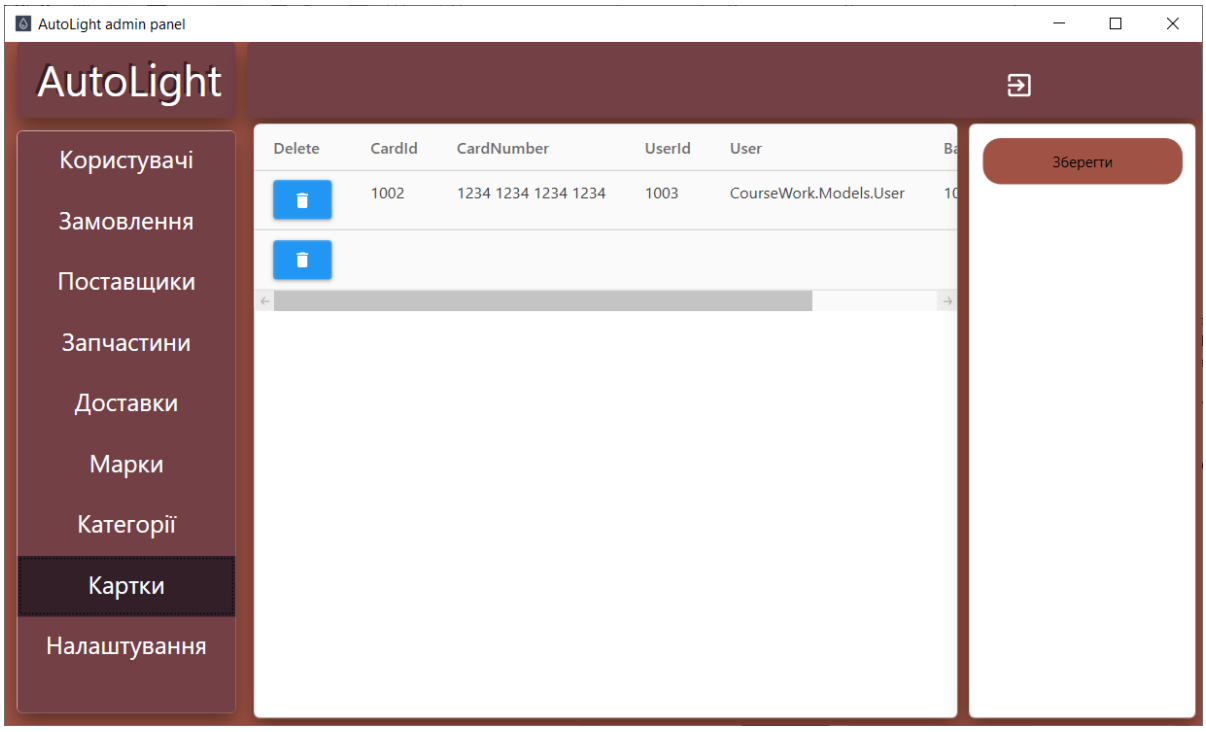
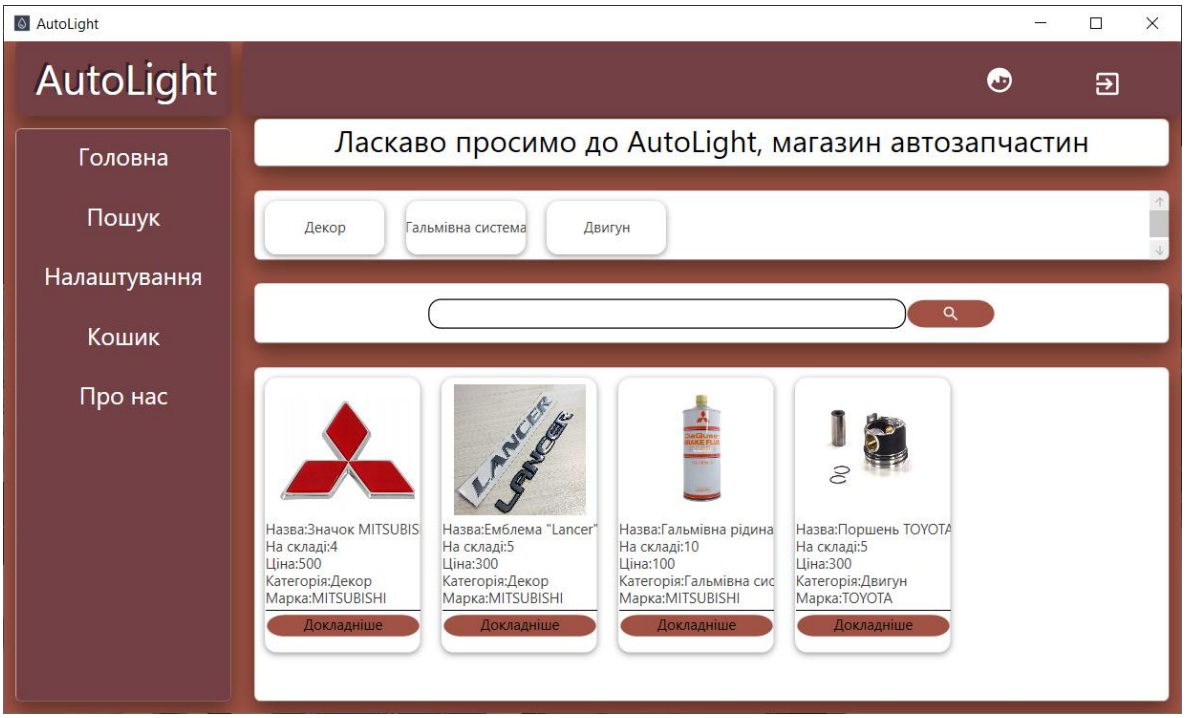


Рисунок Б.17 – Інтерфейс сторінки «Картки»

Застосунку завантажує з БД інформацію про кількість товарів, їхню вартість та іншу інформацію(зображено на рис. Б.18 а).Якщо це стосується акаунту з правами адміна - то також інформацію про всіх користувачів, прив’язані до акаунтів користувачів. (зображено на рис. Б.18 б).

а)



б)

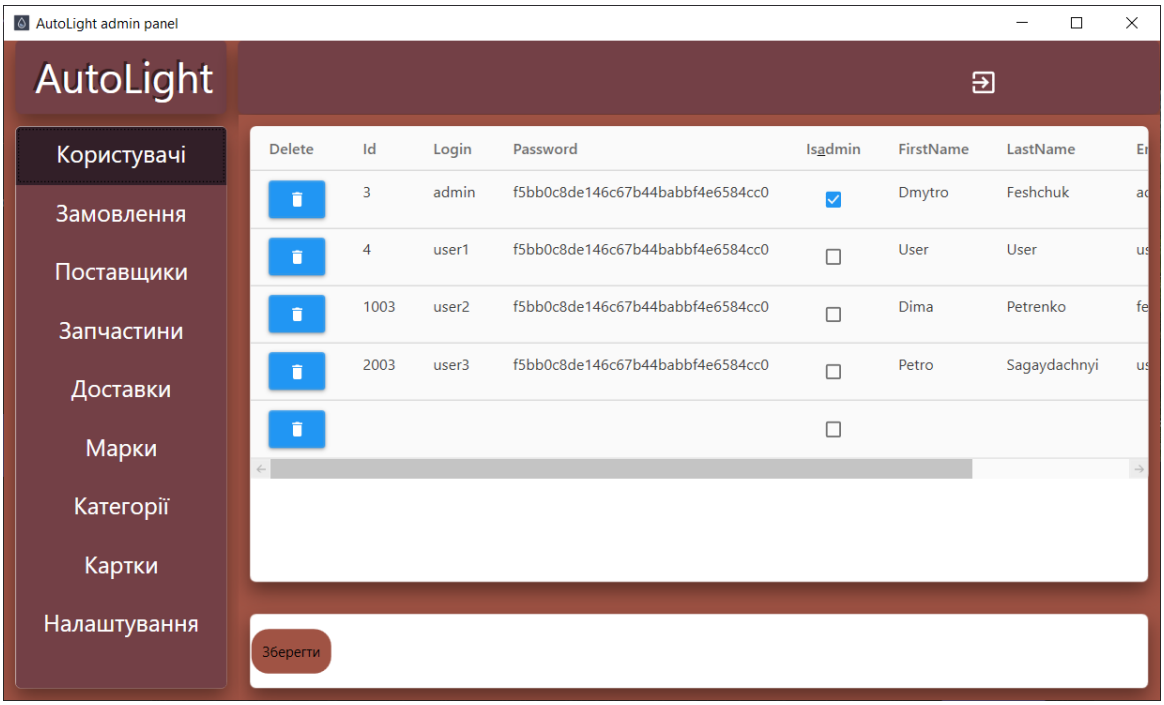


Рисунок Б.18 – а) Вид з акаунту користувача; б)Вид з акаунту адміна

Натискаючи на відповідні кнопки, користувач переміщається між вкладками застосунку. Так ,наприклад, при натисненні на кнопку «Кошик» користувач перейде до свого кошика з списком товарів, які він туди сам заздалегіть додав. (зображено на рис. Б.19 а). Також для адміна доступні інші кнопки, які перехордять на вкладки для управління магазином. (зображено на рис. Б.19 б)

а)



б)

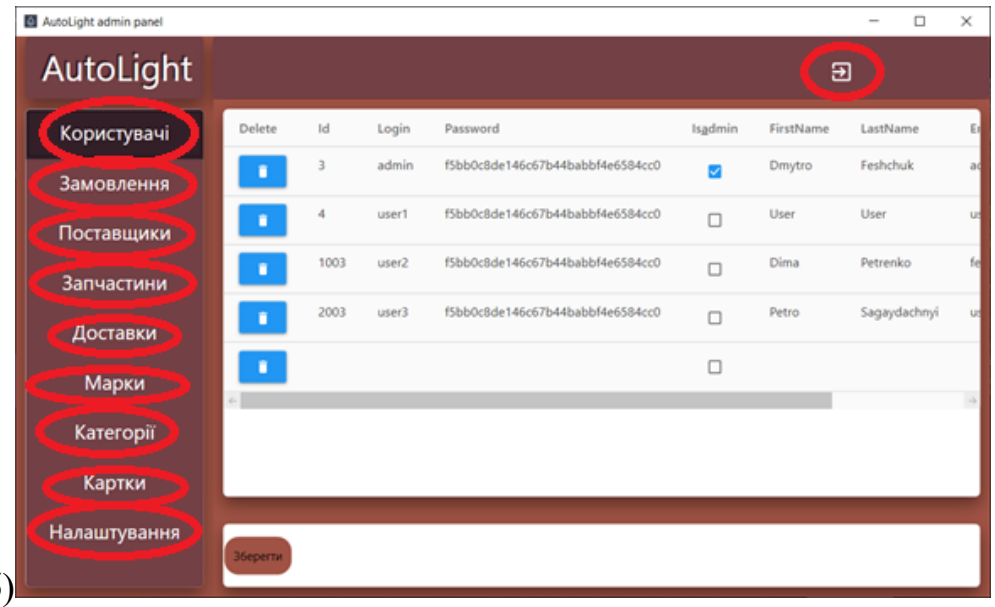
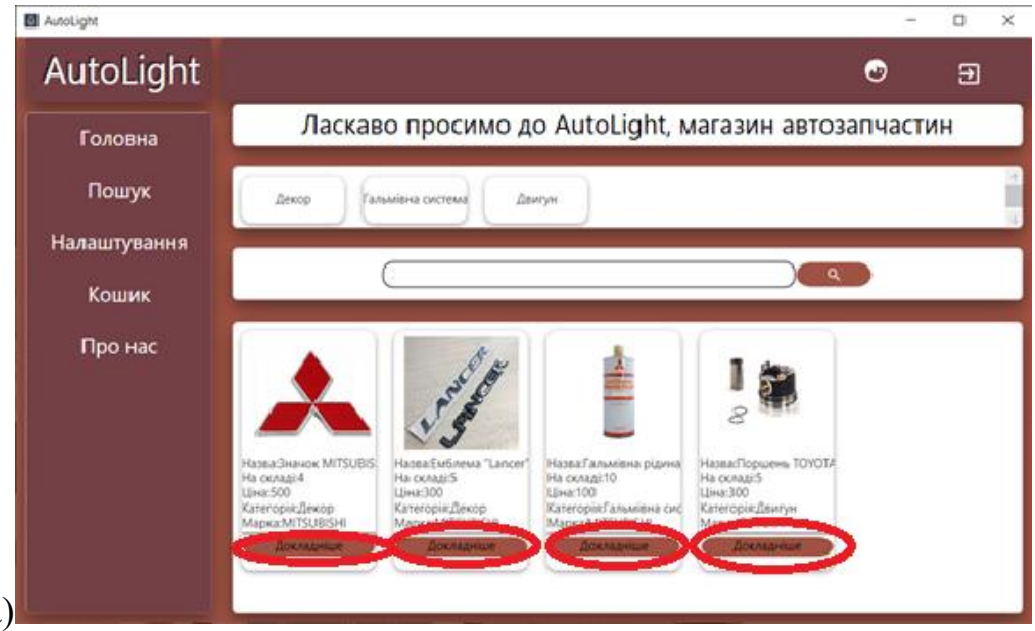


Рисунок Б.19 – а) Доступні вкладки без адмін статусу; б) Доступні вкладки з адмін статусом.

Користувач на головній сторінці, і не тільки може перейти до детального опису товару(зображено на рис. Б.20 а), після чого додати товар в кошик(зображено на рис. Б.20 б).

а)



б)

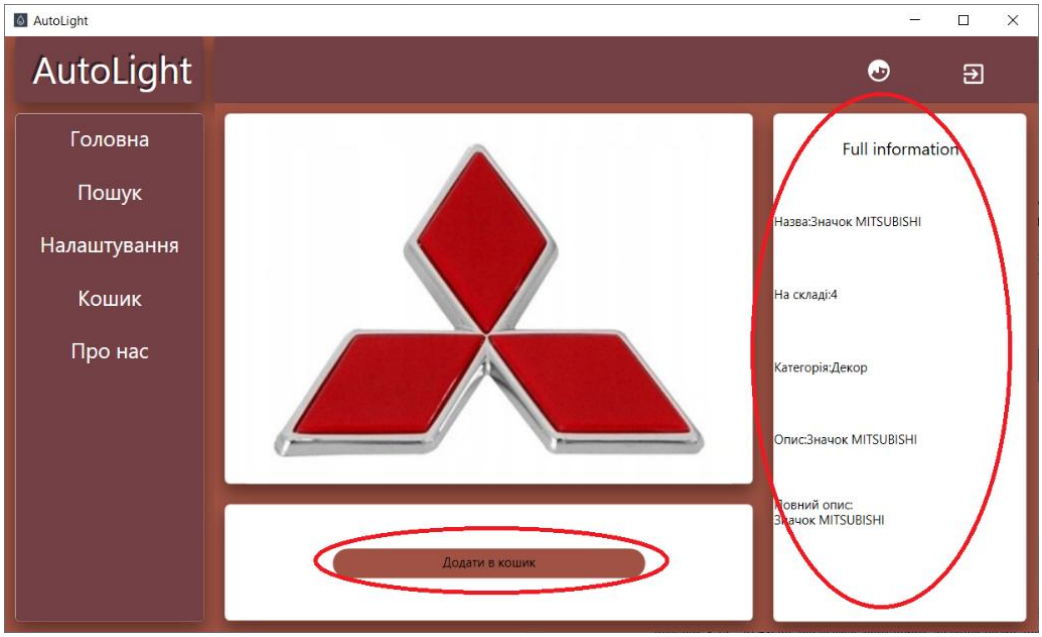
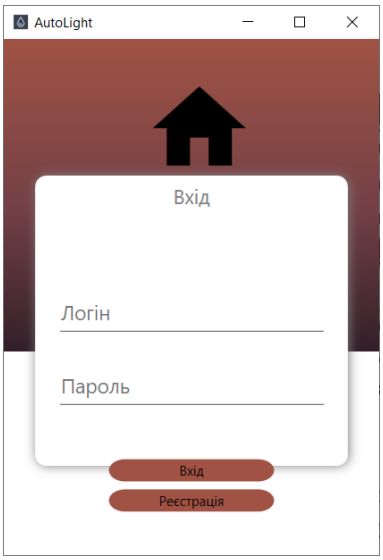
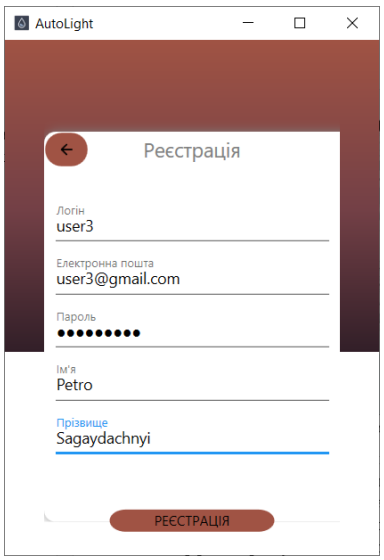


Рисунок Б.20– а) Список товарів; б) Детальний опис обраного товару

Функції авторизації дозволяють користувачу увійти у свій акаунт, чи створити новий акаунт. Для авторизації, користувачу потрібно натиснути на відповідну кнопку, після введення даних свого акаунту (зображено на рис. Б.21 а). Так при реєстрації акаунту користувачу відкривається вікно з реєстрацією, де необхідно ввести коректні дані та підтвердити реєстрацію (зображено на рис. Б.21 б). Після чого користувач повинен буде авторизуватися для входу в свій акаунт. Є

можливість вийти з акаунту, і перейти до форми авторизації. (зображено на рис. Б.21в)

а) 

б) 

в) 

Рисунок Б.21– а) Кнопки авторизації користувача; б) Вікно реєстрації нового користувача; в)Кнопка для виходу з акаунту.

Користувач може скористуватися фільтрацією товарів і пошуком.(зображено на рис. Б.22).

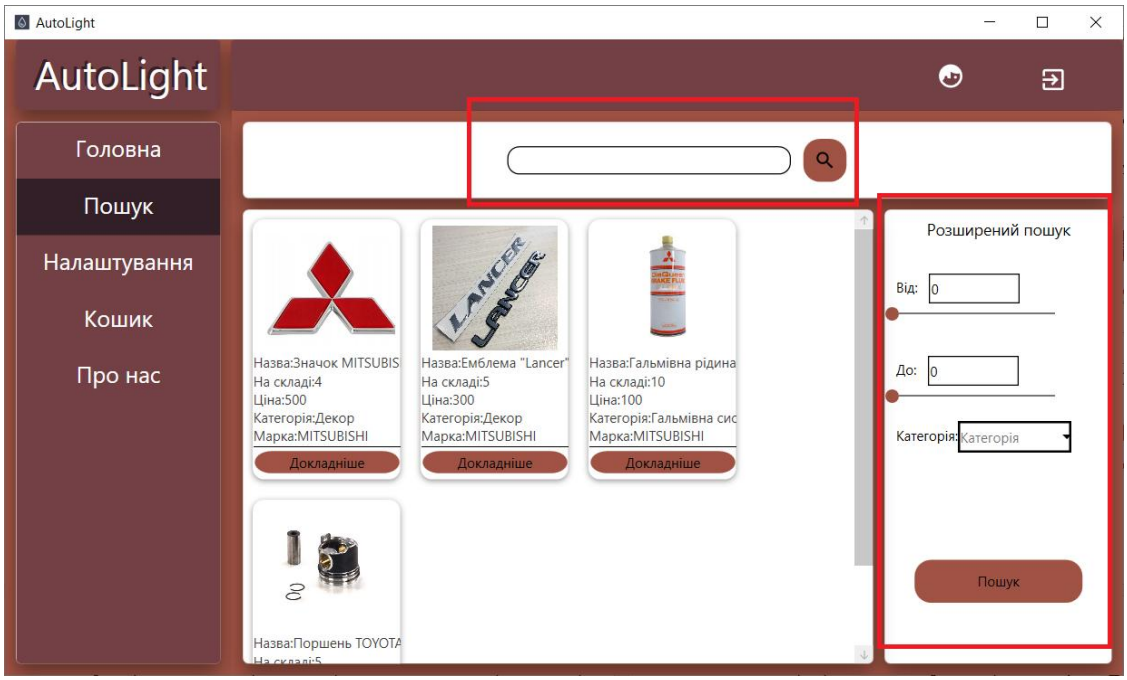
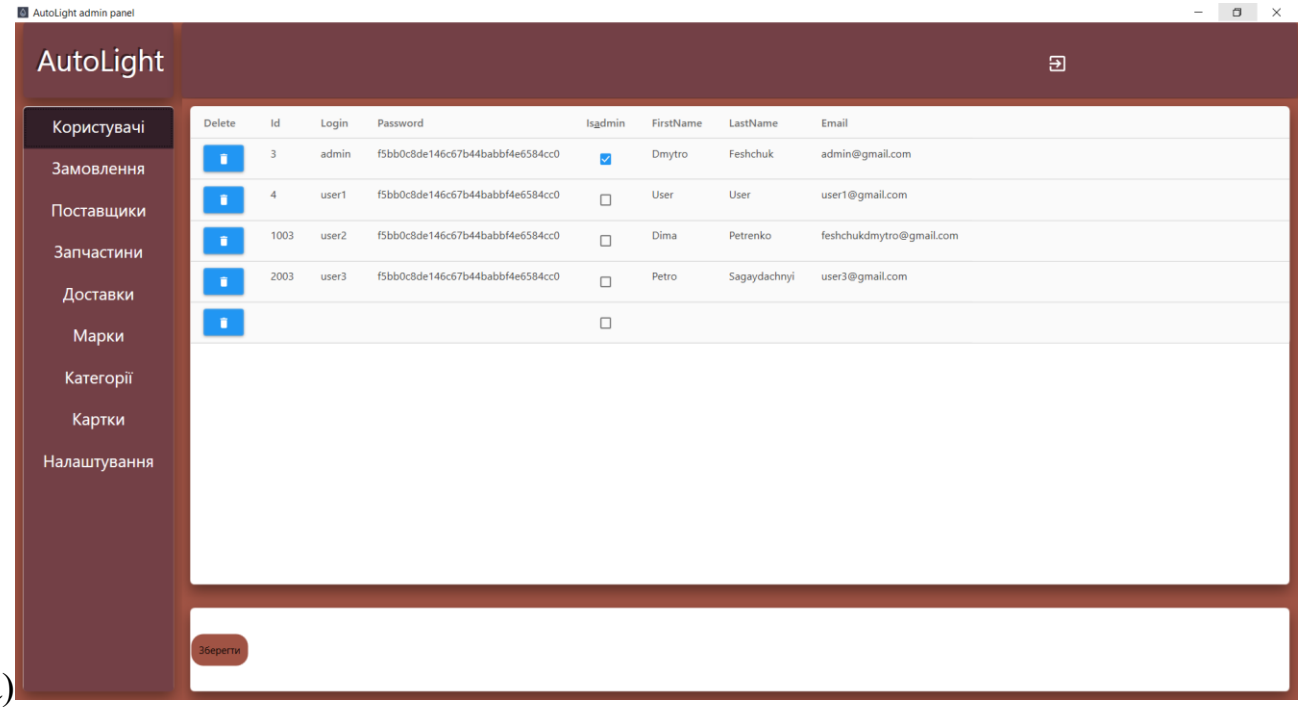


Рисунок Б.22– Фільтрація і пошук товарів

Функція додавання товарів доступна тільки корисутвачам з адмін статусом. А також додавання нових поставщиків, доставок і т.д. (зображено на рис. Б.23а) адмін також може редагувати всі дані. Також надавати статус адміна. (зображено на рис. Ю.23).

а)



б)

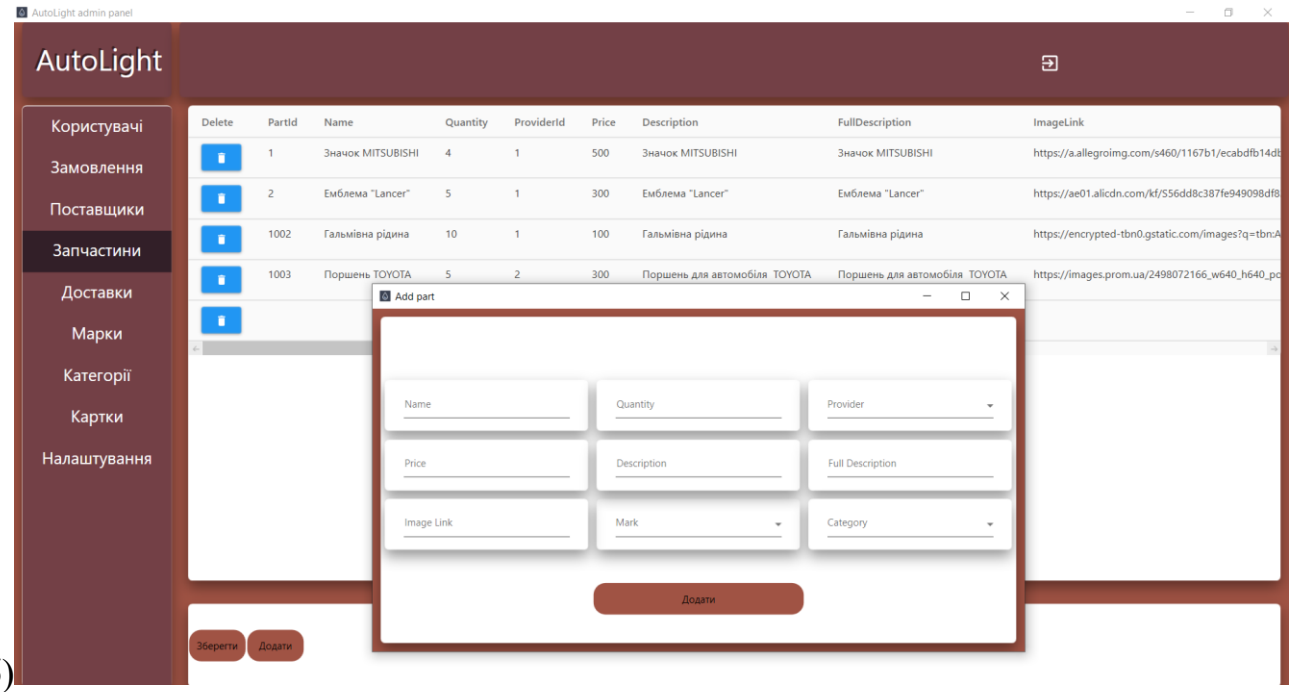
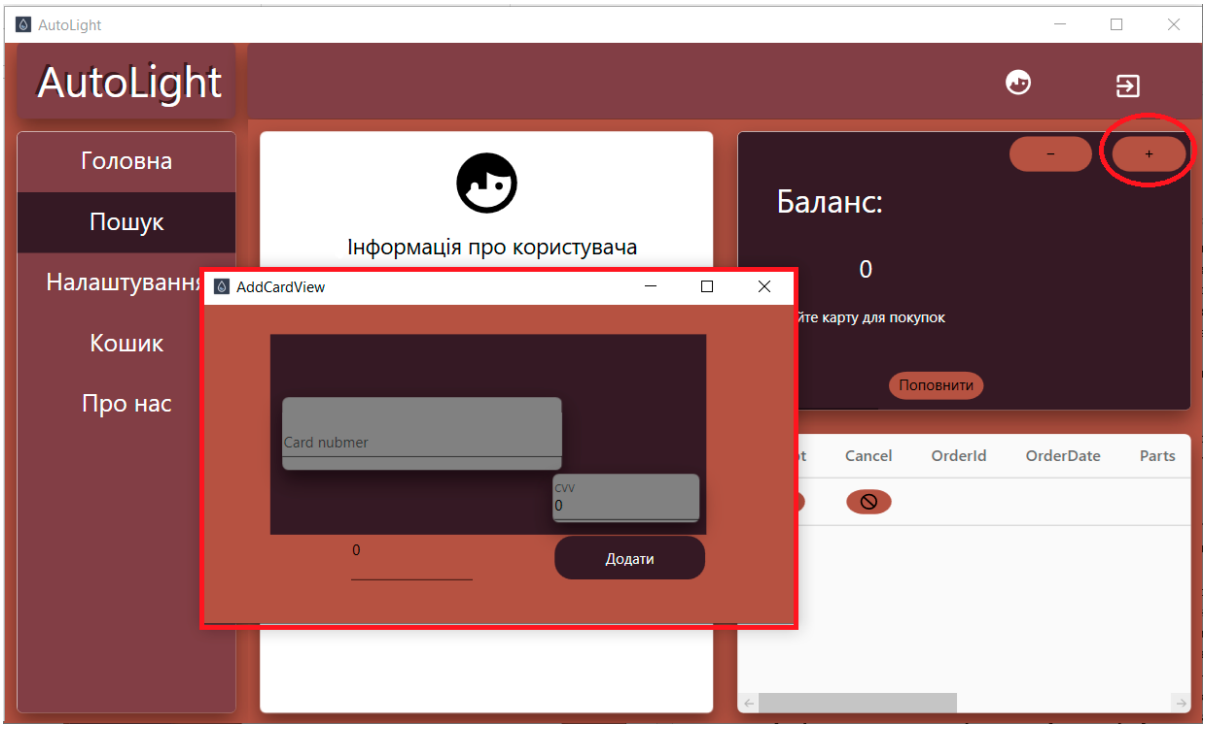


Рисунок Б.23 – а) вигляд вікон для додавання та редагування інформації про користувачів, б)вигляд вікон для додавання та редагування інформації про товари

Доступна прив’язка карти. Для цього користувачу потрібно написнути на відповідну кнопку(зображено на рис. Б.24а), після чого з’явиться форма для вводу даних карти, якщо дані пройдуть валідацію, то карта буде успішно прив’язана. (зображено на рис. Б.24б)при необхідності користувач може відв’язати карту.

а)



б)

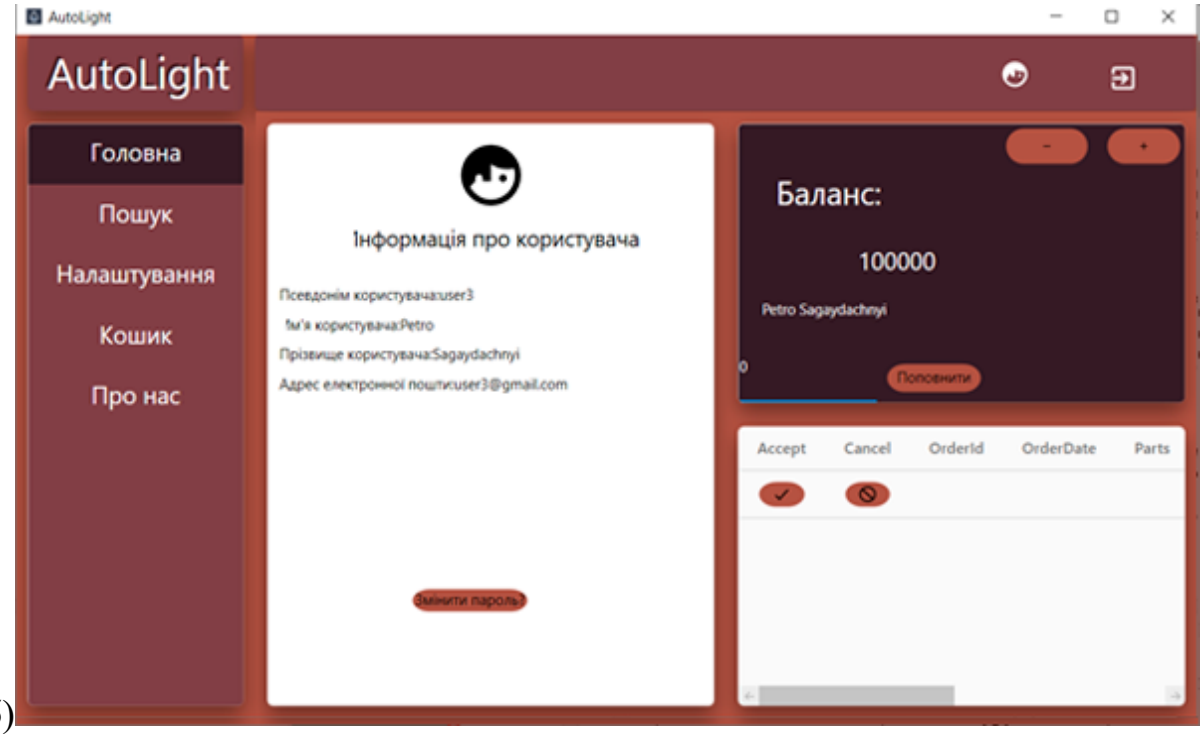
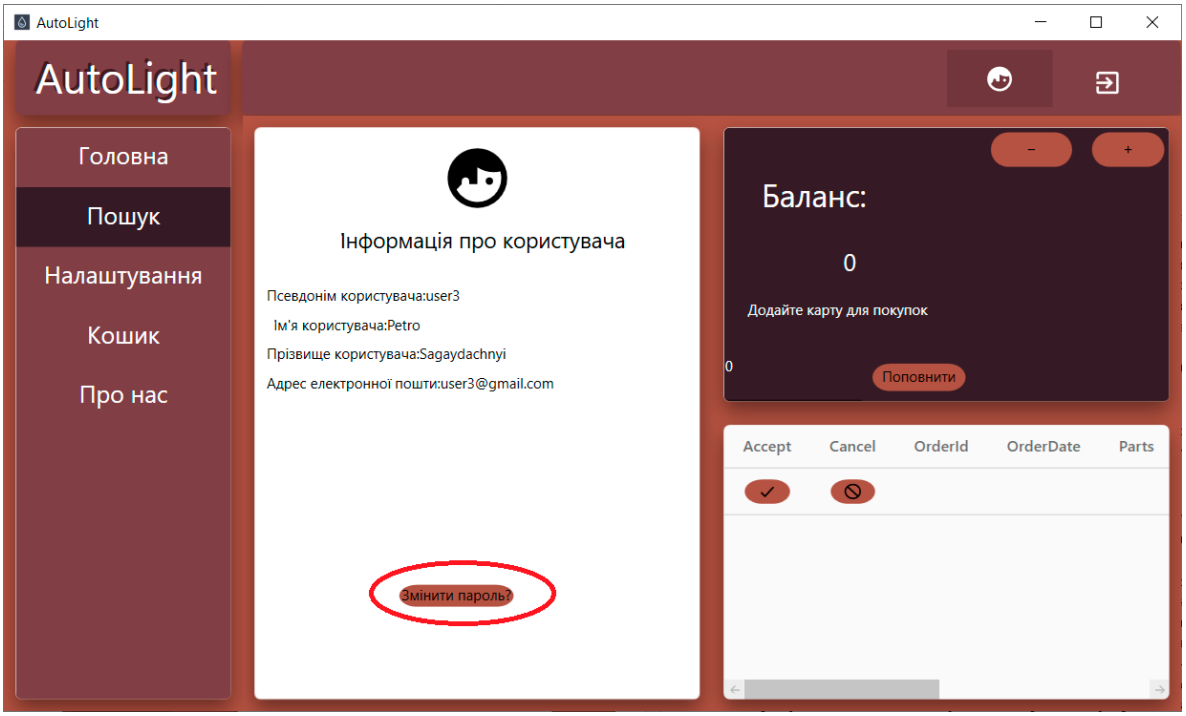


Рисунок Б.24 – а)процес прив’язки банківської карти, б)карта успішно прив’язана

Також користувач може змінювати пароль. (зображено на рис. Б.25а). Для цього портібно натиснути на відповідну кнопку, після чого буде запропоновано 2 поля для введення нового паролю. (зображено на рис. Б.25б).

а)



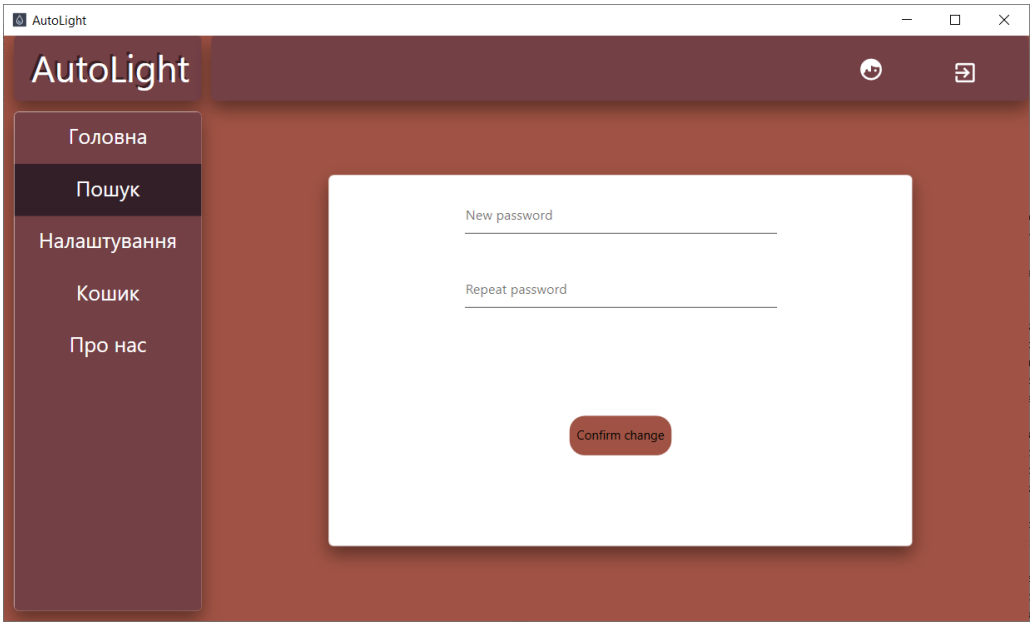


Рисунок Б.25 – а)кнопка для переходу в форму зміни паролю б) форма зміни паролю

ДОДАТОК В. ВИХІДНИЙ КОД ПРОГРАМИ

Лістинг основного коду програми:

HomeViewModel.cs

```
using CourseWork.Commands;
using CourseWork.Database;
using CourseWork.Models;
using CourseWork.SingletonView;
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Input;
using ToastNotifications;
using ToastNotifications.Lifetime;
using ToastNotifications.Messages;
using ToastNotifications.Position;

namespace CourseWork.ViewModels
{
    public class HomeViewModel : ViewModelBase
    {
        public ObservableCollection<Part> Parts { get; set; }
        public ObservableCollection<Part> PartsForSearch { get; set; }
        public Part part;
        public string textForSearch { get; set; }
        public ObservableCollection<Category> categories { get; set; }
        public ObservableCollection<Mark> Marks { get; set; }
        private Command addToCartCommand;
        private Command openFullInfoCommand;
        private Command findByCategory;
        private Command findByName;
        public int id { get; set; }
        private Part selectedPart;
        public Part SelectedPart
        {
            get { return selectedPart; }
            set
            {
```

```
                selectedPart = value;
                OnPropertyChanged("SelectedPart");
            }
        }
        private Category selectedCategory;
        public Category SelectedCategory
        {
            get { return selectedCategory; }
            set {
                selectedCategory = value;
                OnPropertyChanged("SelectedCategory");
            }
        }

        public HomeViewModel()
        {
            using (PartShopDbContext db = new PartShopDbContext())
            {
                try
                {
                    Parts = new ObservableCollection<Part>(db.Parts);
                    PartsForSearch = new ObservableCollection<Part>(db.Parts);
                    categories = new ObservableCollection<Category>(db.Categories);
                    Marks = new ObservableCollection<Mark>(db.Marks);
                }
                catch(Exception e)
                {
                    MessageBox.Show(e.Message);
                }
            }
        }
        public ICommand AddToCartCommand
        {
            get
            {
                return addToCartCommand ??
                  (addToCartCommand = new Command(obj =>
                    {
                        Singleton.getInstance(null).MainViewModel.CurrentViewModel = new
SearchViewModel(Parts);
                    }));
            }
        }
```

```
        public ICommand FindByCategory
        {
            get
            {
                return findByCategory ??
                  (findByCategory = new Command(obj =>
                  {
                      try
                      {
                          using (PartShopDbContext db = new PartShopDbContext())
                          {
                              Parts = new ObservableCollection<Part>(db.Parts.Where(x =>
x.CategoryId == SelectedCategory.CategoryId));
                              Singleton.getInstance(null).MainViewModel.CurrentViewModel
= new SearchViewModel(Parts);
                          }
                      }
                      catch(Exception e)
                      {
                          MessageBox.Show(e.Message);
                      }
                  }));
            }
        }
        public ICommand OpenFullInfo
        {
            get
            {
                return openFullInfoCommand ??
                  (openFullInfoCommand = new Command(obj =>
                  {
                      Singleton.getInstance(null).MainViewModel.CurrentViewModel = new
FullInfoViewModel(selectedPart);
                  }));
            }
        }
        public ICommand FindByName
        {
            get
            {
                return findByName ??
                  (findByName = new Command(obj =>
                  {
```

```
                    try
                    {
                        PartsForSearch = new
ObservableCollection<Part>(App.db.Parts.Where(x => x.Name.Contains(textForSearch)));
                        Singleton.getInstance(null).MainViewModel.CurrentViewModel =
new SearchViewModel(PartsForSearch);
                    }
                    catch(Exception e)
                    {
                        MessageBox.Show(e.Message);
                    }
                }));
            }
        }
        private Command testCommand;
        public ICommand TestCommand
        {
            get
            {
                return testCommand ??
                  (testCommand = new Command(obj =>
                  {
                      try
                      {
                          App.db.Parts.Where(x => x.PartId ==
13).FirstOrDefault().Quantity -= 3;
                          App.db.SaveChanges();
                      }
                      catch(Exception e)
                      {
                          MessageBox.Show(e.Message);
                      }
                  }));
            }
        }

    }
}
```

ConnectionBetweenViews.cs

```
using CourseWork.Models;
using System;
using System.Collections.Generic;
```

```
using System.Collections.ObjectModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace CourseWork.Database
{
    public static class ConnectionBetweenViews
    {
        public static ObservableCollection<Part> Parts = new
ObservableCollection<Part>();
        public static Part Part = new Part();
    }
}
```

PartShopDbContext.cs

```
using CourseWork.Models;
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

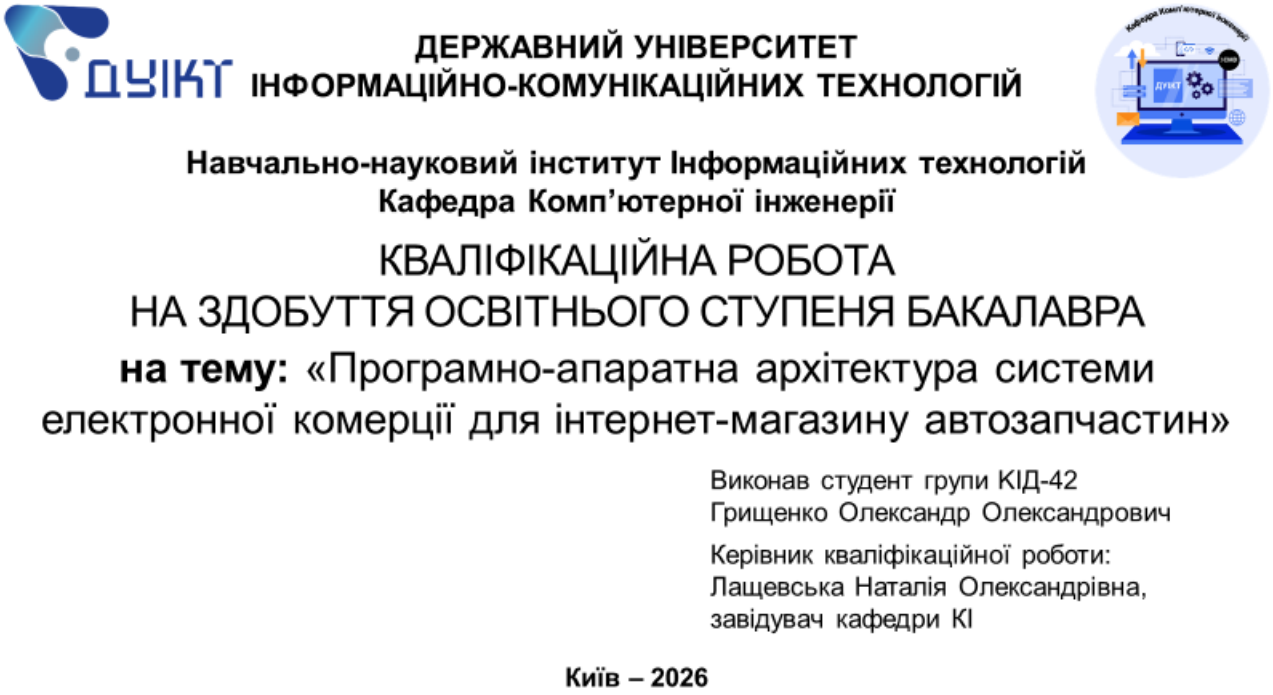
namespace CourseWork.Database
{
    public class PartShopDbContext : DbContext
    {
        public PartShopDbContext() : base("DBConnection") { }
        public DbSet<User> Users { get; set; }
        public DbSet<Cart> Carts { get; set; }
        public DbSet<Order> Orders { get; set; }
        public DbSet<Part> Parts { get; set; }
        public DbSet<Provider> Providers { get; set; }
        public DbSet<Delivery> Deliveries { get; set; }
        public DbSet<Category> Categories { get; set; }
        public DbSet<Card> Cards { get; set; }
        public DbSet<Mark> Marks { get; set; }
        public DbSet<OrderedParts> OrderedParts { get; set; }

        protected override void OnModelCreating(DbModelBuilder modelBuilder)
        {
            modelBuilder.Entity<OrderedParts>().HasKey(t => new { t.PartId, t.OrderId });
```

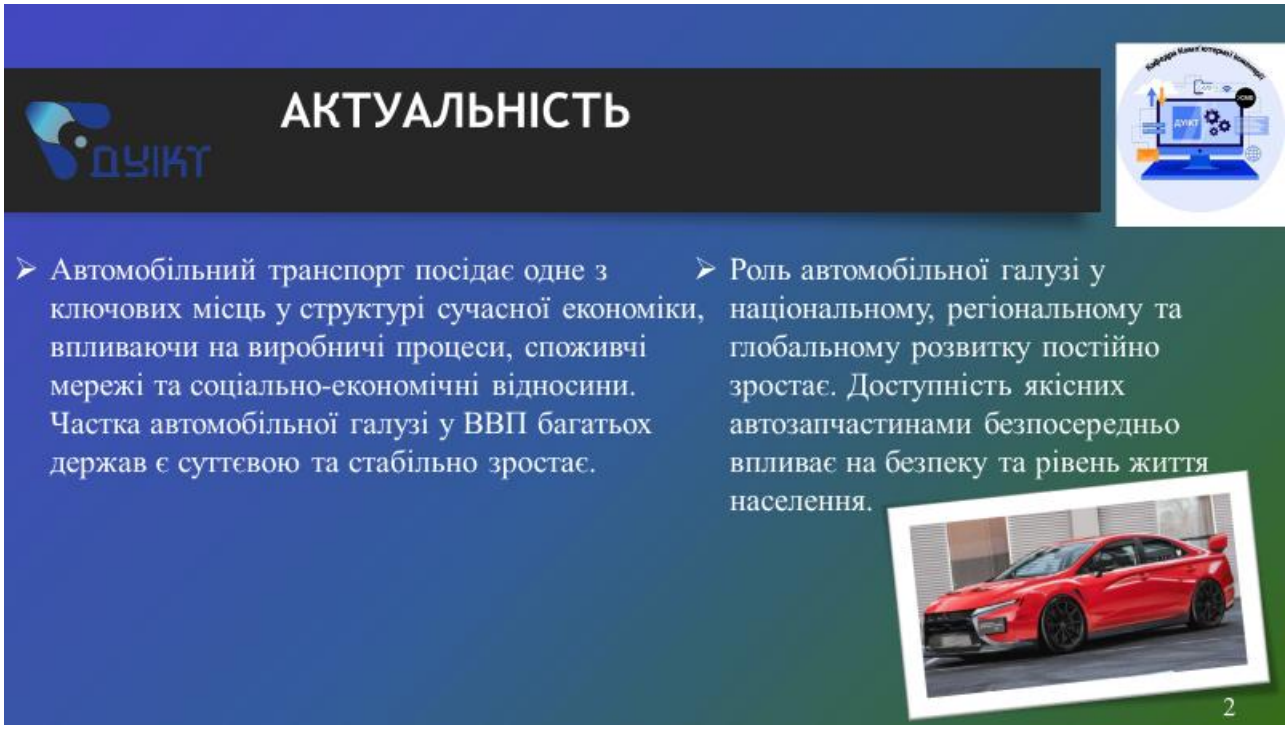
```
            modelBuilder.Entity<OrderedParts>()
                .HasRequired(u => u.Order)
                .WithMany(ur => ur.Parts)
                .HasForeignKey(ui => ui.PartId);

            modelBuilder.Entity<OrderedParts>()
                 .HasRequired(r => r.Order)
                 .WithMany(ur => ur.Parts)
                 .HasForeignKey(ri => ri.OrderId);
        }
    }
}
```

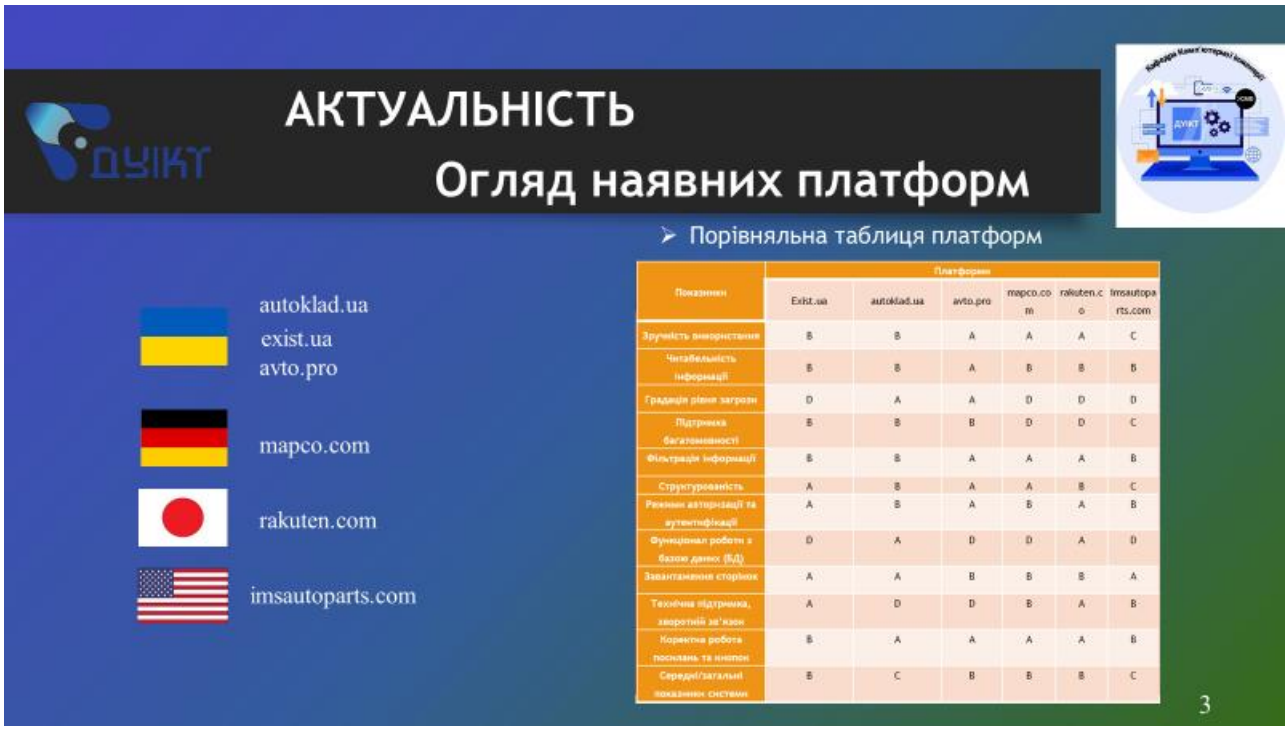
ДОДАТОК Г. ПРЕЗЕНТАЦІЯ



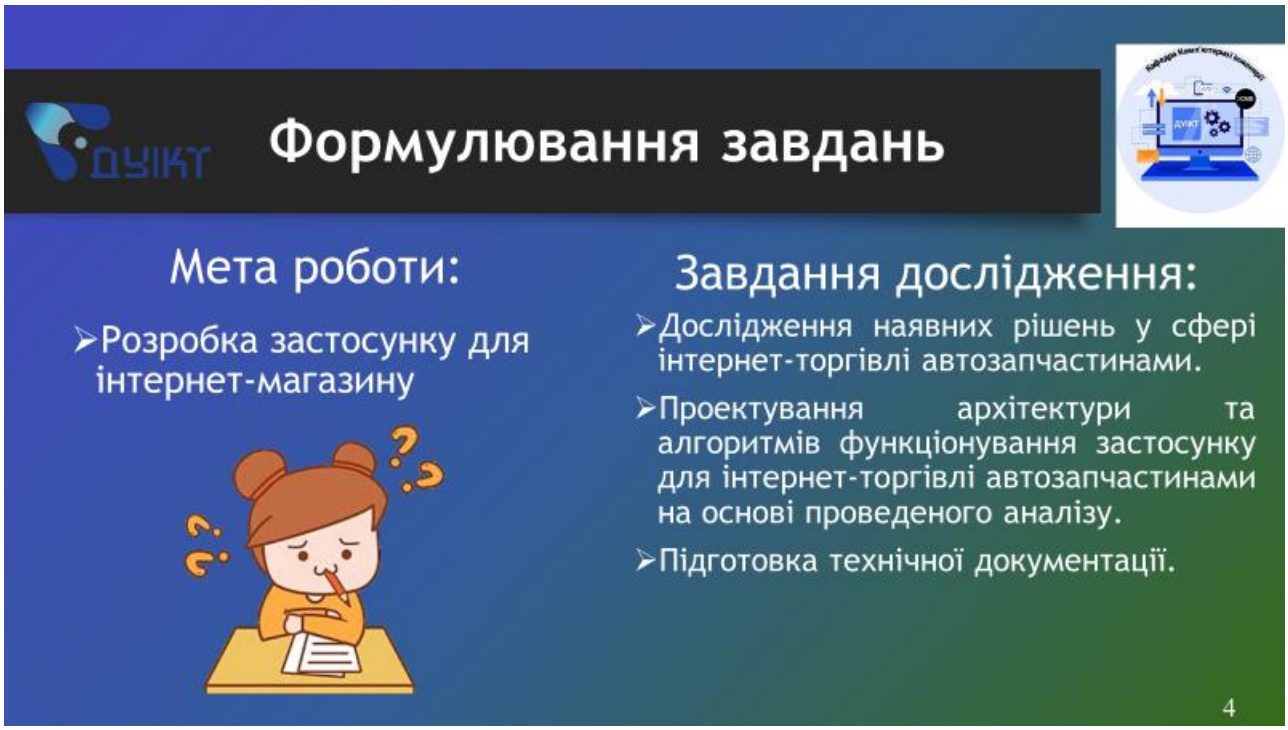
Слайд Г.1 – Слайд 1



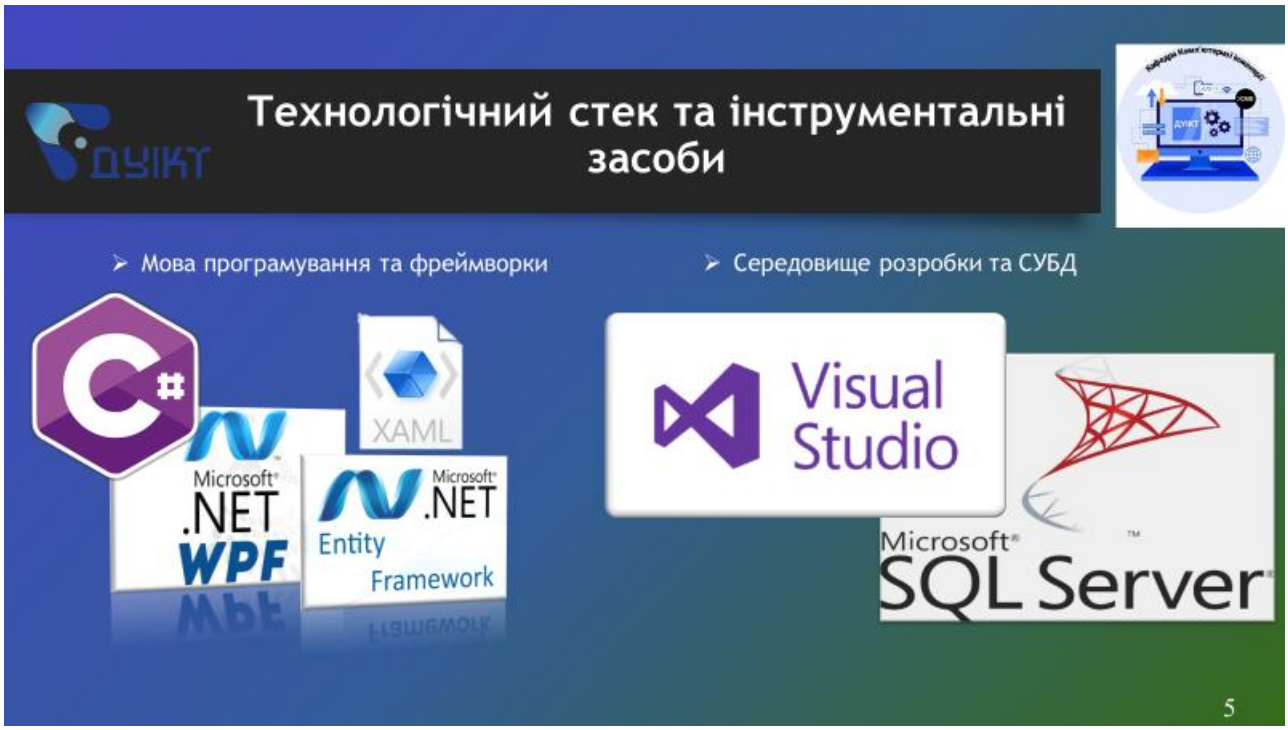
Слайд Г.2 – Слайд 2



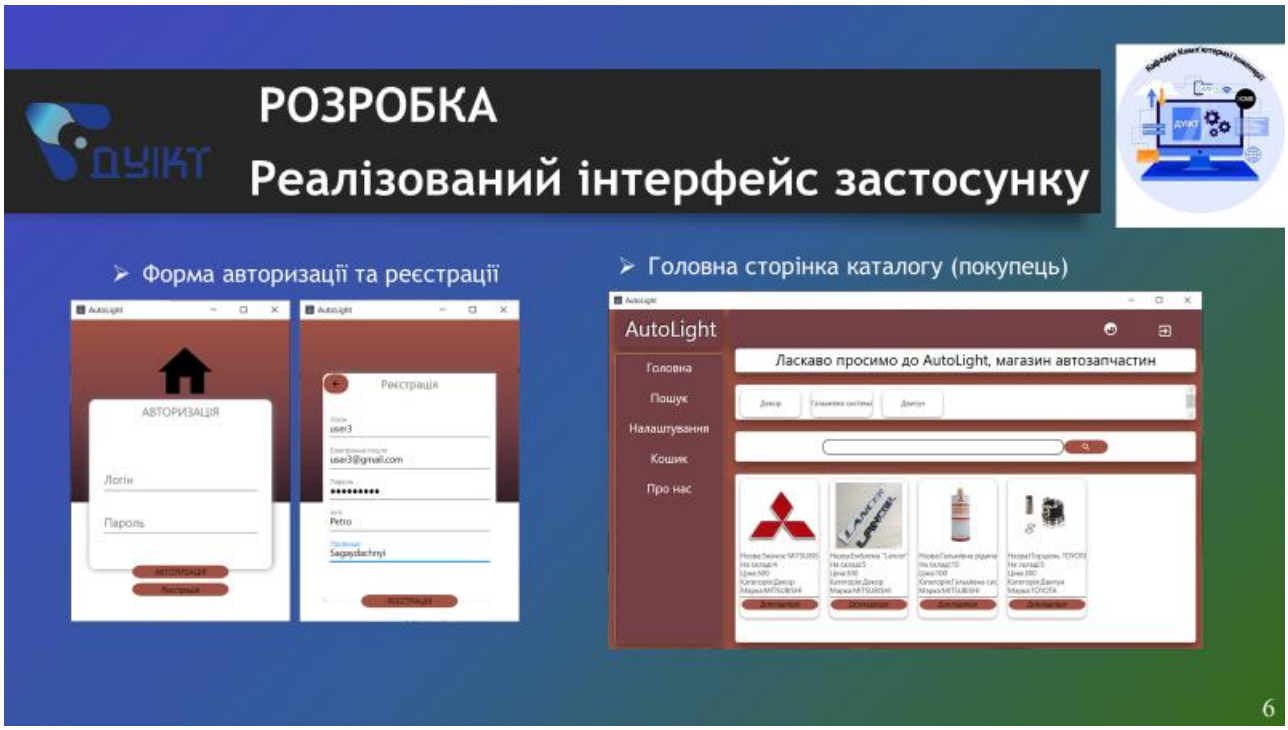
Слайд Г.3 – Слайд 3



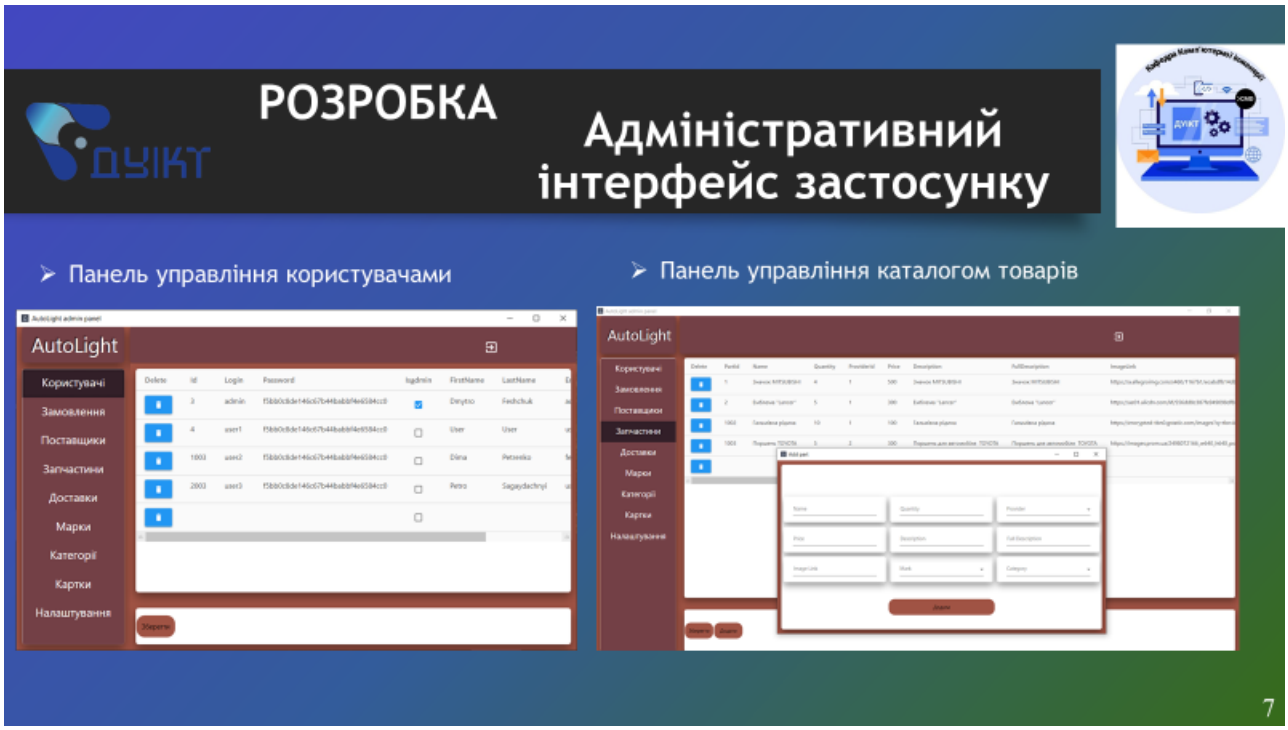
Слайд Г.4 – Слайд 4



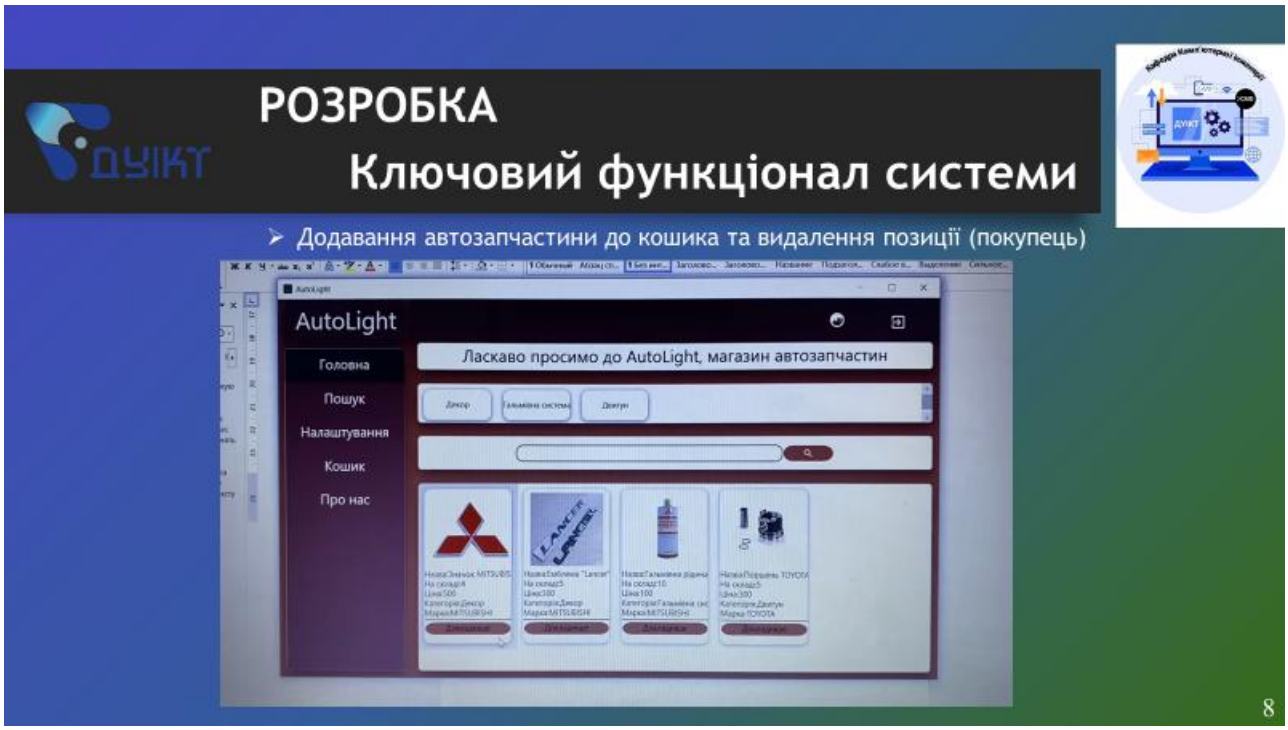
Слайд Г.5 – Слайд 5



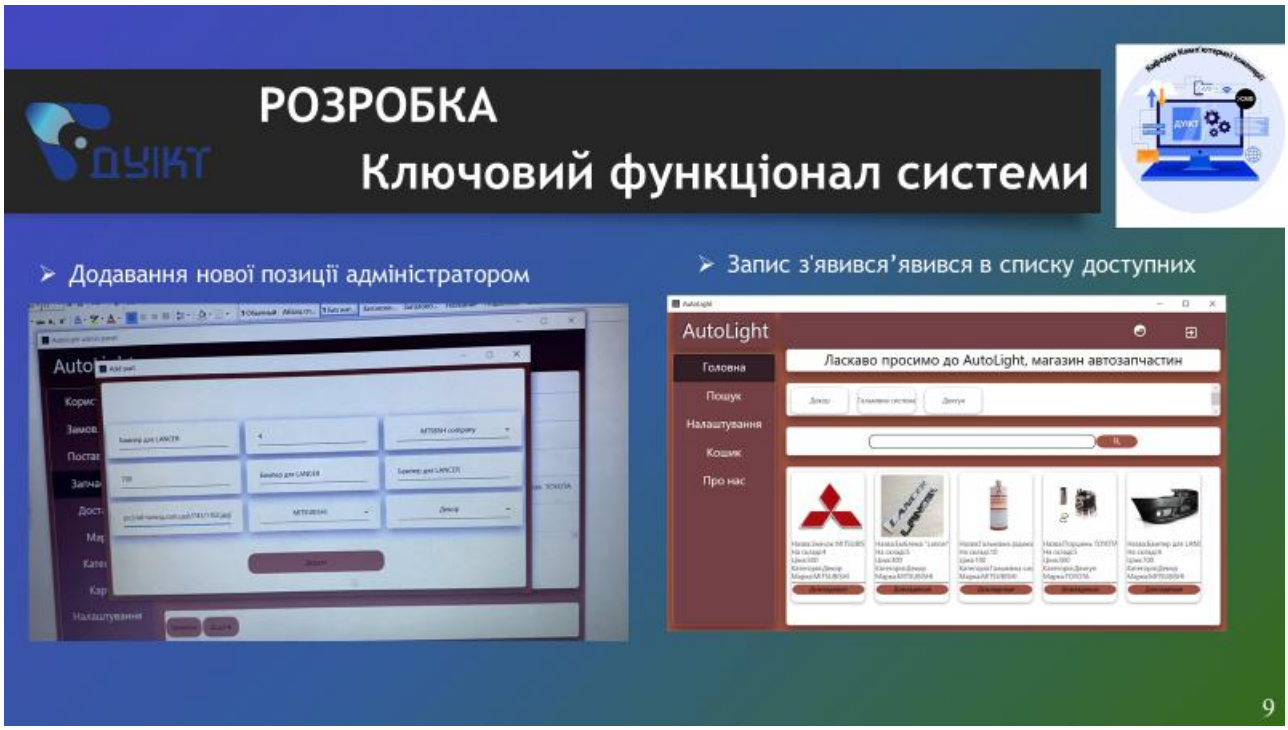
Слайд Г.6 – Слайд 6



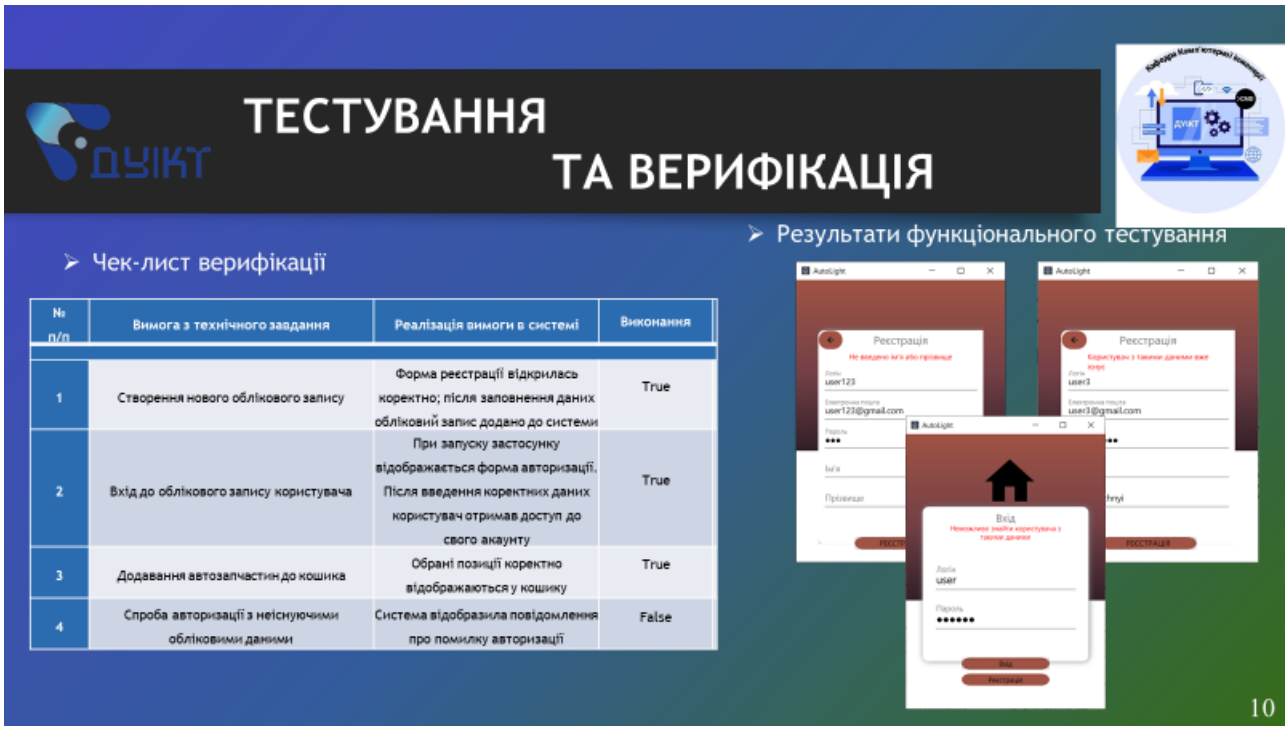
Слайд Г.7 – Слайд 7



Слайд Г.8 – Слайд 8

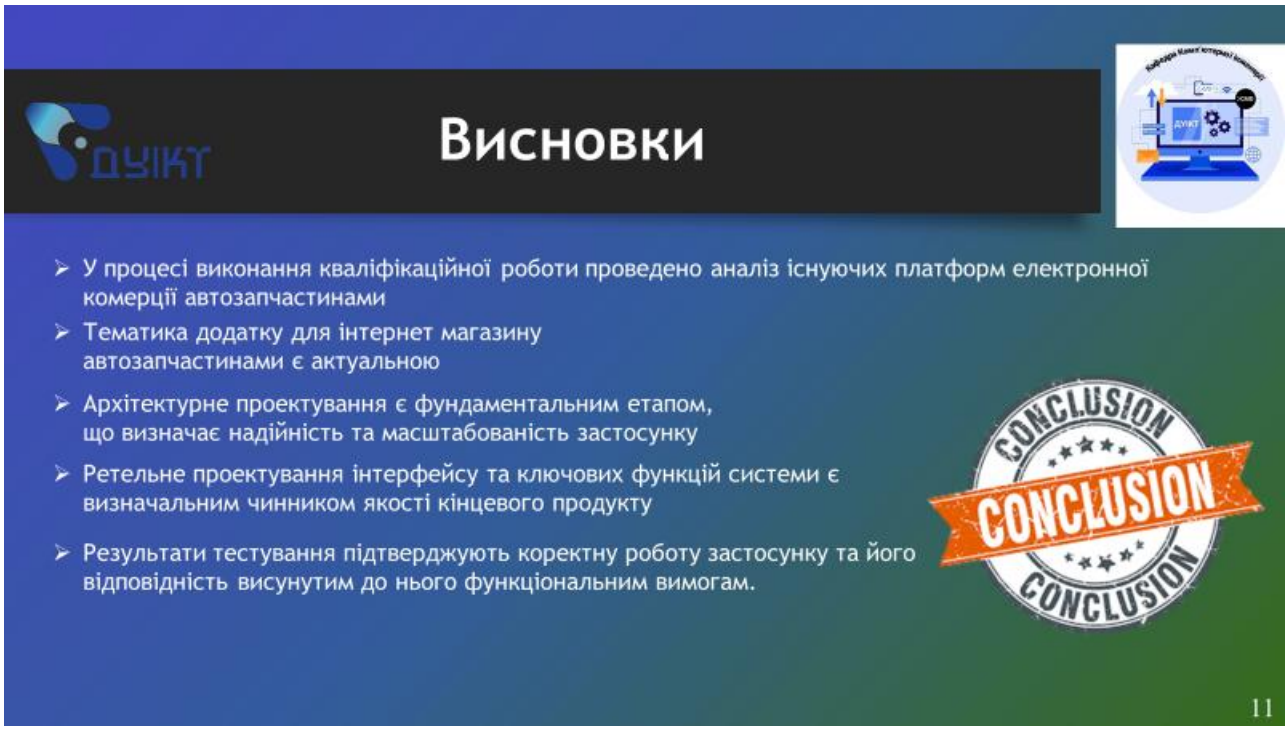


Слайд Г.9 – Слайд 9



№ п/п	Вимога з технічного завдання	Реалізація вимоги в системі	Виконання
1	Створення нового облікового запису	Форма реєстрації відкрилась коректно; після заповнення даних обліковий запис додано до системи	True
2	Вхід до облікового запису користувача	При запуску застосунку відображається форма авторизації. Після введення коректних даних користувач отримав доступ до свого акаунту	True
3	Додавання автозапчастин до кошика	Обрані позиції коректно відображаються у кошику	True
4	Спроба авторизації з неіснуючими обліковими даними	Система відобразила повідомлення про помилку авторизації	False

Слайд Г.10 – Слайд 10



Слайд Г.11 – Слайд 11