

Projekt wypożyczalni samochodowej

Data oddania:

25.06.2025

Agata Skalska	416306	Inżynieria Mechatroniczna
Miłosz Stec	418076	Inżynieria Mechatroniczna
Adam Puc	417818	Inżynieria Mechatroniczna

Spis treści

Opis struktury projektu	3
Windows Presentation Foundation	3
Środowisko Rhapsody	5
Opis interfejsu użytkownika	6
Ekran Logowania.....	6
Okno użytkownika (pracownika)	7
Lista samochodów	7
Historia	7
Klienci	8
Okno Administratora	9
Cennik.....	9
Zarządzanie Flotą.....	9
Użytkownicy.....	10
Przyciski	11
Opis działania aplikacji	13
Logowanie i autoryzacja	13
Zarządzanie danymi (Repozytoria + baza SQL)	13
Wyliczenie ceny i scenariusz wypożyczenia.....	13
User Interface i obsługa zdarzeń	13
Algorytmika i techniczne rozwiązania	14
Fragmenty kodu.....	15
Implementacja dziedziczenia – ViewModelBase.....	15
Dziedziczenie w LoginViewModel	15
Obsługa zdarzeń	16
Walidacja danych przed logowaniem	16
Mechanizm autoryzacji logowania	17
Bezpieczne przechowywanie hasła	18
Obsługa zdarzeń w oknie – Login_Screen.xaml.cs.....	18
Testowanie i przykładowe scenariusze działania.....	19
Scenariusz dla Administratora	19
Scenariusz dla Użytkownika.....	23
Wnioski i dalszy rozwój.....	29

Wprowadzenie

Celem projektu jest zbudowanie aplikacji okienkowej za pomocą WPF (C#). Aplikacja podzielona jest na dwie funkcje dostępu -okno administratora oraz użytkownika (pracownika). Głównymi założeniami aplikacji są wypożyczanie aut oraz ich odbiór elektroniczny. W zależności od roli dodane zostały również funkcje umożliwiające:

- Logowanie
- Wyświetlanie listy wszystkich aut
- Historie wypożyczeni
- Dodania, usuwania oraz modyfikacji pojazdów
- Dodania, usuwania użytkowników

Aplikacja została stworzona do obsługi od strony kadry pracowniczej (nie przez użytkownika końcowego).

Opis struktury projektu

Windows Presentation Foundation

Projekt jest przykładem systemu typu MVVM (Model-View-ViewModel), zaimplementowanego przy użyciu technologii WPF (Windows Presentation Foundation)

Struktura Warstw:

- Model - Reprezentuje dane i logikę domenową
- ViewModel - Pośredniczy między View a Modelem, zawiera logikę
- View - Interfejs użytkownika

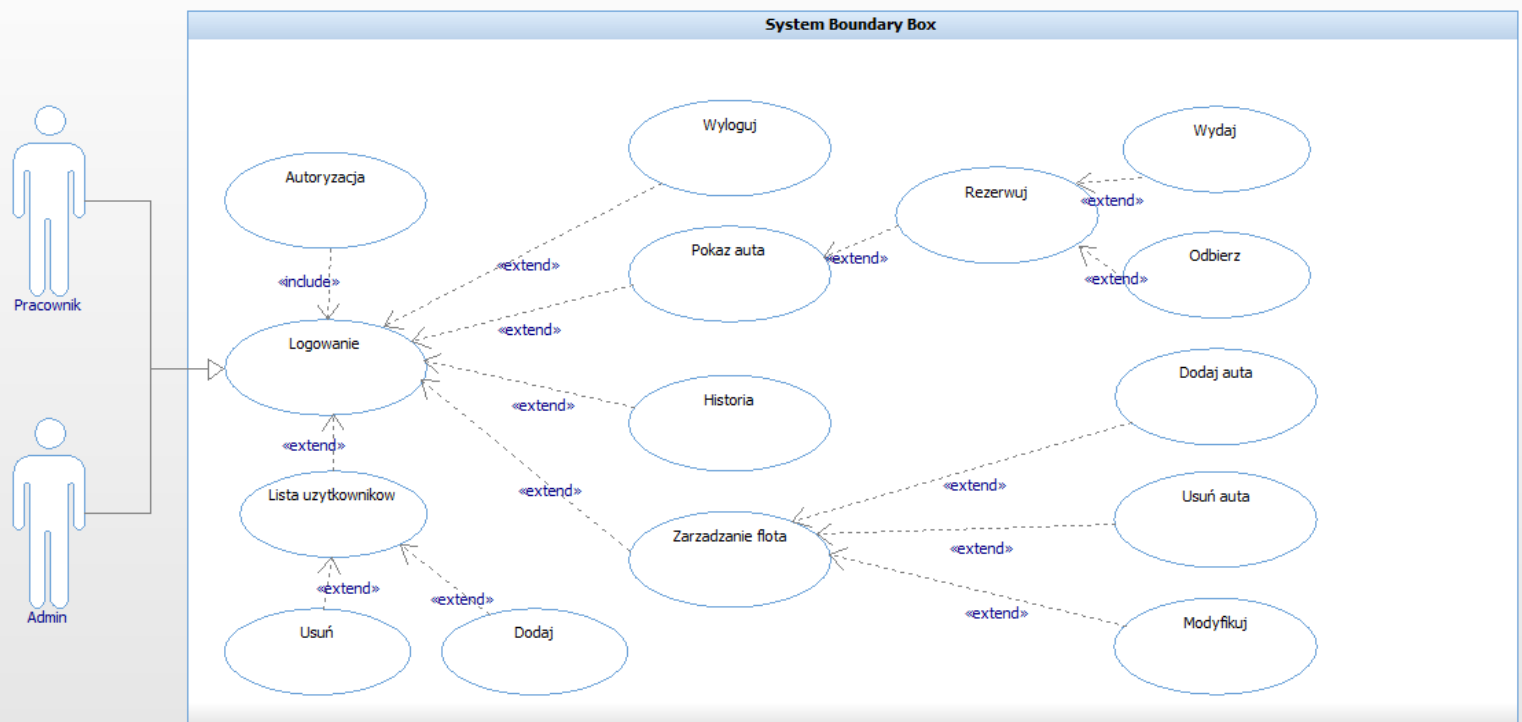
Opis powiązań najważniejszych relacji między klasami:

- LoginViewModel - Korzysta z UserRepository do uwierzytelniania, dziedziczy z ViewModelBase
- UserRepository - udostępnia metody typu GetByUsername, AuthenticateUser, zwraca UserModel
- UserModel - reprezentuje dane użytkownika: UserId, Username, Access, PasswordHash
- MainWindow.xaml - powiązany z LoginViewModel przez DataContext, reaguje na komendy
- App.xaml.cs - przechowuje aktualnego użytkownika (App.CurrentUser)
- UserSession - singleton dla danych sesji: ID, login, uprawnienia

Architektura:

- Logika biznesowa (np. logowanie) jest odseparowana od interfejsu
- Repozytoria oddzielają dane od logiki aplikacji – można łatwo zmienić źródło danych
- Komendy (ICommand) są używane zamiast zdarzeń przycisków
- Model danych (np. UserModel) jest łatwy w rozszerzeniu o kolejne funkcje

Środowisko Rhapsody



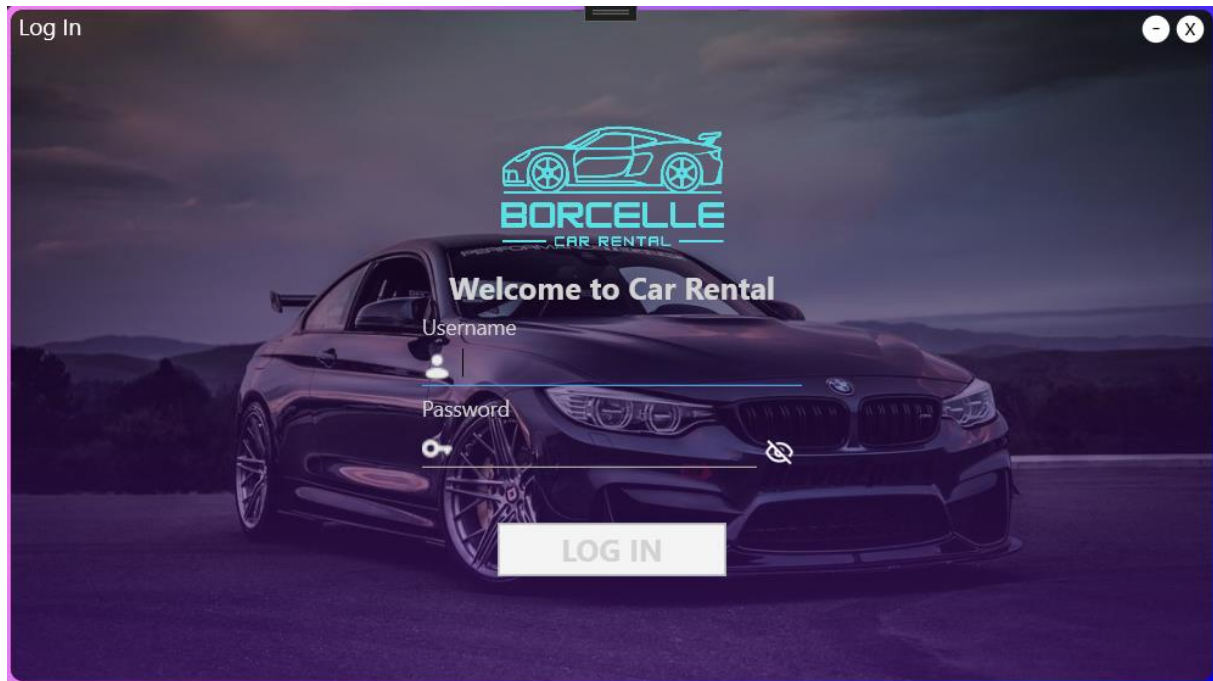
Pracownik (User) – osoba zatrudniona przez firmę operującą oprogramowaniem, która ma za zadanie wydawać i odbierać auta od klientów końcowych.

Admin – kierownik oddziału firmy operującej oprogramowaniem, która ma za zadanie nadzorować działanie floty pojazdów oraz ma możliwości pracownika.

Obie role mają dostęp do okna logowania. W zależności od przypisanych funkcji do danej roli, mają one dostęp do danych funkcjonalności aplikacji. Administrator ma te same funkcjonalności co User

Opis interfejsu użytkownika

Ekran Logowania



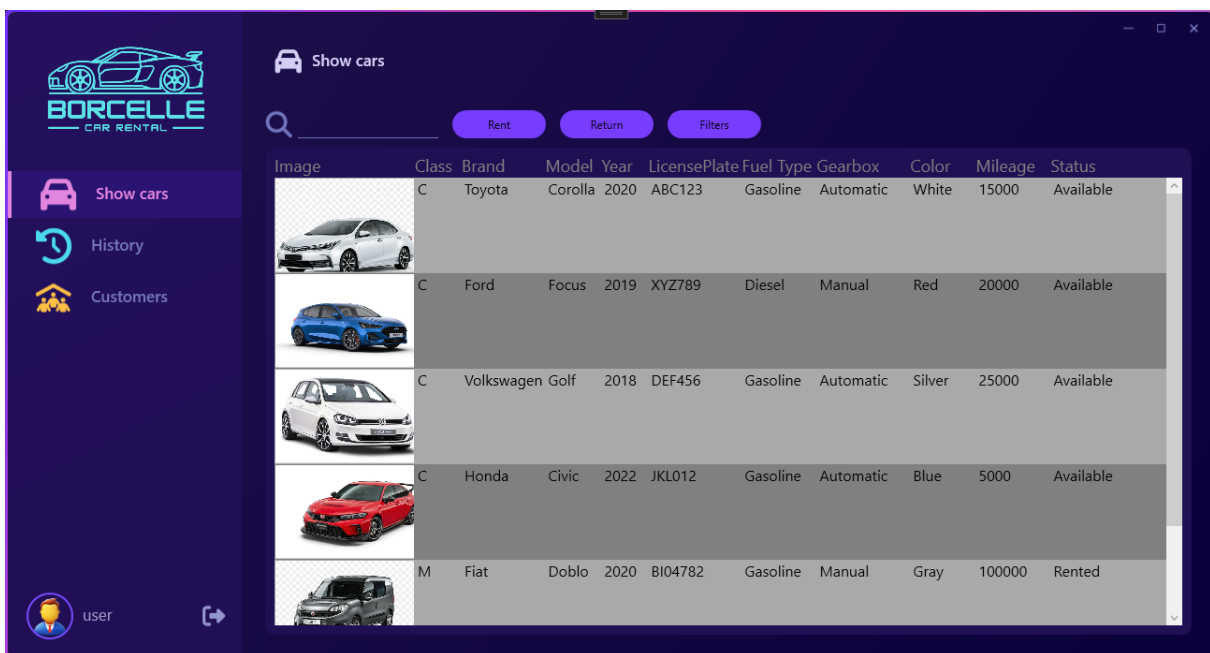
Okno logowania zawiera dwa pola tekstowe do wpisywania danych do logowania (username oraz password), a także przycisk zatwierdzający te dane, przenoszący do kolejnych okien. W prawym górnym rogu znajdują się przyciski zamykające oraz minimalizujące okno.

Okno użytkownika (pracownika)

Okno pracownika składa się z menu bocznego oraz ekranu głównego.
Funkcje dostępne dla użytkownika:


Lista samochodów




Pracownik firmy może wyświetlać, wypożyczać oraz odbierać samochody




Historia

Dla użytkownika dostępna jest historia wypożyczonych samochodów wraz z danymi klientów, datą wypożyczenia oraz zwrotu, statusem oraz ceną



 Show cars
  History
  Customers


 user




History


Edit
Delete
Service History

ID	Plate	User	Customer	PESEL	Start Date	End Date	Status	Total Price
10	JKL012	admin	Anna Nowak	98765432109	6/23/2025 12:00:00 AM	6/25/2025 12:00:00 AM	Finished	120
11	JKL012	admin	Anna Nowak	98765432109	6/27/2025 12:00:00 AM	6/29/2025 12:00:00 AM	Finished	120
12	JKL012	admin	Jan Kowalski	12345678901	6/23/2025 12:00:00 AM	6/25/2025 12:00:00 AM	Finished	120
13	JKL012	admin	Jan Kowalski	12345678901	6/25/2025 12:00:00 AM	6/29/2025 12:00:00 AM	Finished	220
14	JKL012	admin	Jan Kowalski	12345678901	6/30/2025 12:00:00 AM	7/2/2025 12:00:00 AM	Finished	120
15	JKL012	admin	Anna Nowak	98765432109	7/2/2025 12:00:00 AM	7/5/2025 12:00:00 AM	Finished	180
16	JKL012	admin	Jan Kowalski	12345678901	7/10/2025 12:00:00 AM	7/19/2025 12:00:00 AM	Finished	450
17	JKL012	admin	Jan Kowalski	12345678901	9/2/2025 12:00:00 AM	10/5/2025 12:00:00 AM	Finished	1320
18	JKL012	admin	Anna Nowak	98765432109	7/28/2025 12:00:00 AM	7/29/2025 12:00:00 AM	Finished	60
19	JKL012	admin	Anna Nowak	98765432109	7/30/2025 12:00:00 AM	7/31/2025 12:00:00 AM	Finished	60
20	JKL012	admin	Anna Nowak	98765432109	7/1/2025 12:00:00 AM	7/3/2025 12:00:00 AM	Finished	120
21	JKL012	admin	Anna Nowak	98765432109	7/2/2025 12:00:00 AM	7/6/2025 12:00:00 AM	Finished	220
22	JKL012	admin	Jan Kowalski	12345678901	7/1/2025 12:00:00 AM	7/2/2025 12:00:00 AM	Finished	60
23	JKL012	admin	Anna Nowak	98765432109	6/24/2025 12:00:00 AM	6/25/2025 12:00:00 AM	Finished	60
24	JKL012	admin	Jan Kowalski	12345678901	6/26/2025 12:00:00 AM	6/27/2025 12:00:00 AM	Finished	60
25	JKL012	admin	Jan Kowalski	12345678901	6/24/2025 12:00:00 AM	6/26/2025 12:00:00 AM	Finished	120
26	JKL012	admin	Anna Nowak	98765432109	7/1/2025 12:00:00 AM	7/6/2025 12:00:00 AM	Finished	275
27	JKL012	admin	Anna Nowak	98765432109	7/1/2025 12:00:00 AM	7/4/2025 12:00:00 AM	Finished	180
28	JKL012	admin	Jan Kowalski	12345678901	6/24/2025 12:00:00 AM	6/25/2025 12:00:00 AM	Finished	60
29	JKL012	admin	Anna Nowak	98765432109	7/1/2025 12:00:00 AM	7/2/2025 12:00:00 AM	Finished	60

Klienci – lista klientów zarejestrowanych w serwisie



 Show cars
  History
  Customers

 user

Customer management

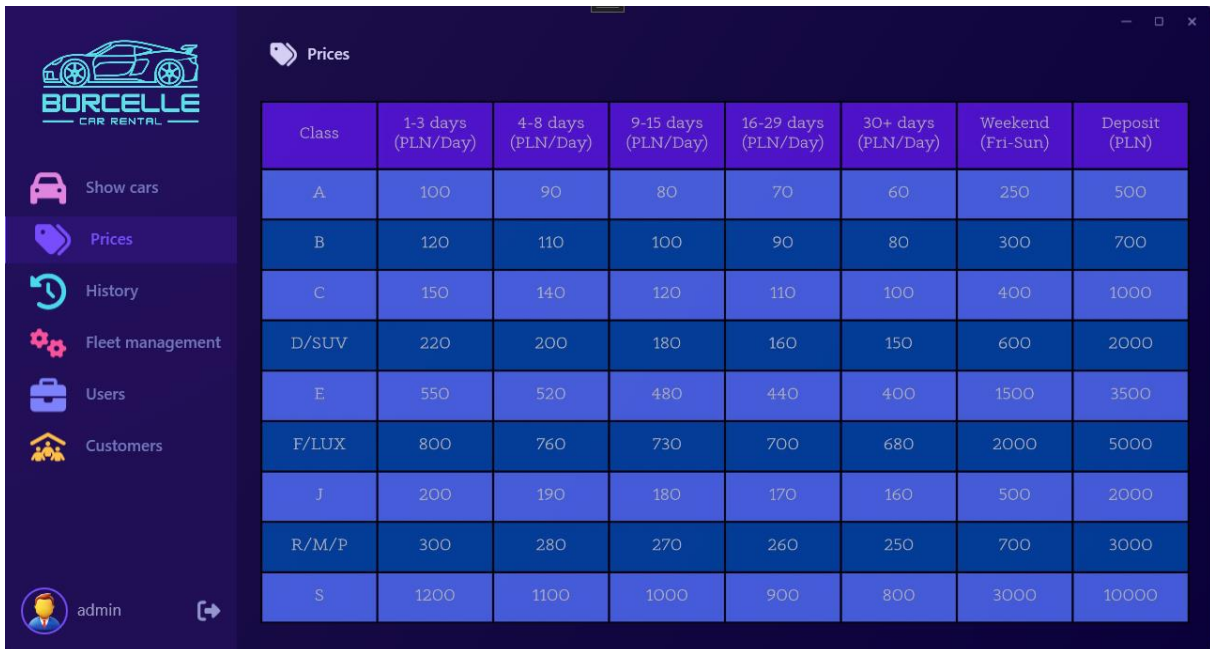
Add
Edit
Remove

Name	Surname	Email	Phone	Pesel	License number	Date of birth	Street
Jan	Kowalski	jan.kowalski@example.com	123456789	12345678901	B1234567	7/10/1985 12:00:00 AM	ul. Warszawsk
Anna	Nowak	anna.nowak@example.com	987654321	98765432109	A7654321	5/21/1990 12:00:00 AM	ul. Krakowska
Adam	Puc	abc	123	987	xyz	6/4/2025 12:00:00 AM	asd

Okno Administratora

Administrator posiada wszystkie funkcje, które posiada pracownik, a dodatkowo:

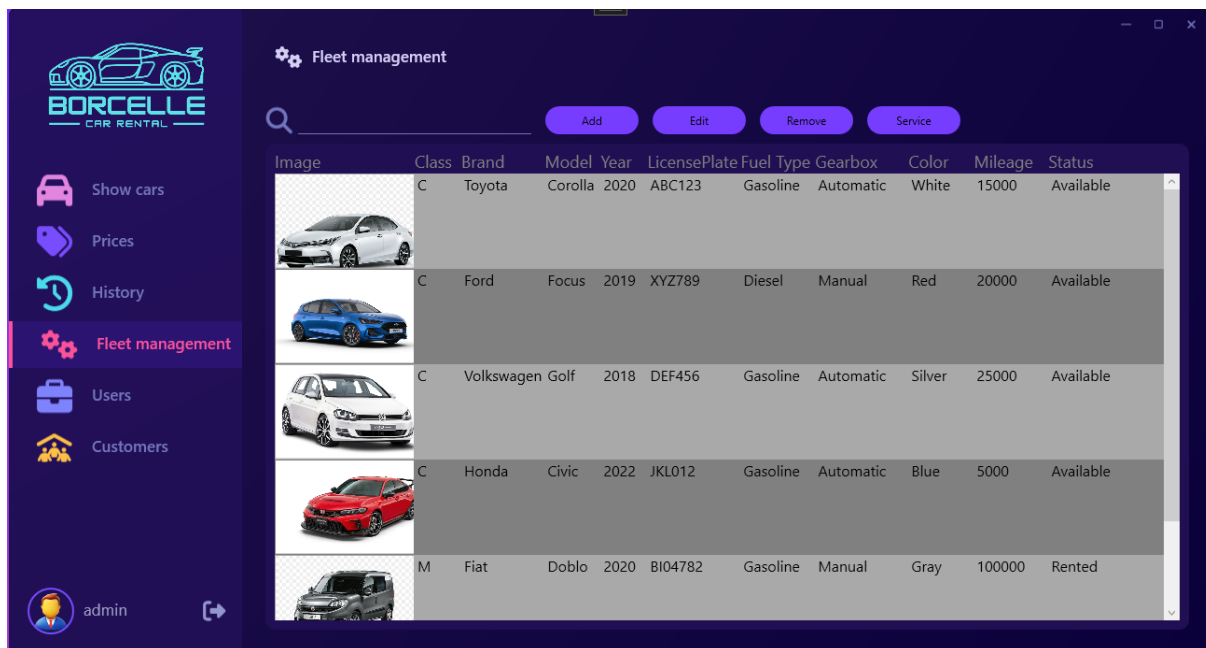
Cennik – Administrator ma dostęp do cennika wypożyczeni samochodów



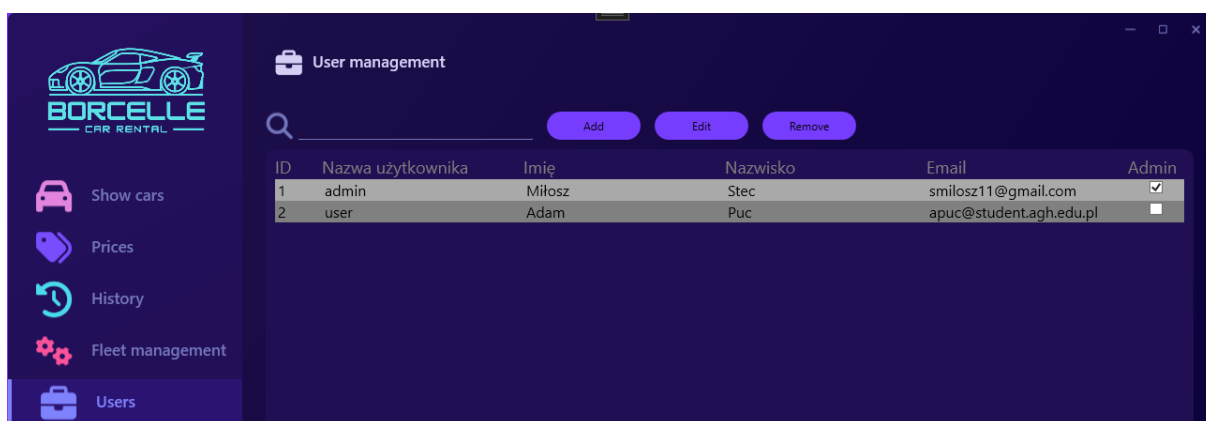
The screenshot shows the 'Prices' section of the Borcelle Car Rental Admin interface. On the left is a sidebar with navigation links: Show cars, Prices (selected), History, Fleet management, Users, and Customers. At the bottom of the sidebar is a user profile for 'admin'. The main area displays a table with car rental prices.

Class	1-3 days (PLN/Day)	4-8 days (PLN/Day)	9-15 days (PLN/Day)	16-29 days (PLN/Day)	30+ days (PLN/Day)	Weekend (Fri-Sun)	Deposit (PLN)
A	100	90	80	70	60	250	500
B	120	110	100	90	80	300	700
C	150	140	120	110	100	400	1000
D/SUV	220	200	180	160	150	600	2000
E	550	520	480	440	400	1500	3500
F/LUX	800	760	730	700	680	2000	5000
J	200	190	180	170	160	500	2000
R/M/P	300	280	270	260	250	700	3000
S	1200	1100	1000	900	800	3000	10000

Zarządzanie Flotą – administrator może dodawać, usuwać oraz edytować samochody znajdujące się w wypożyczalni. Dodatkowo może dać pojazd do serwisu



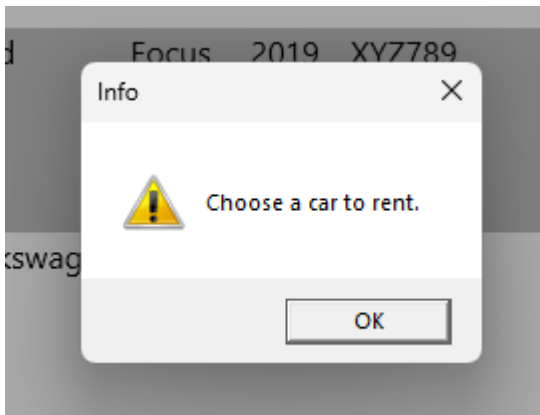
Użytkownicy – Administrator może dodawać, usuwać i edytować użytkowników uprawnionych do dostępu do aplikacji oraz nadawać im odpowiednie role



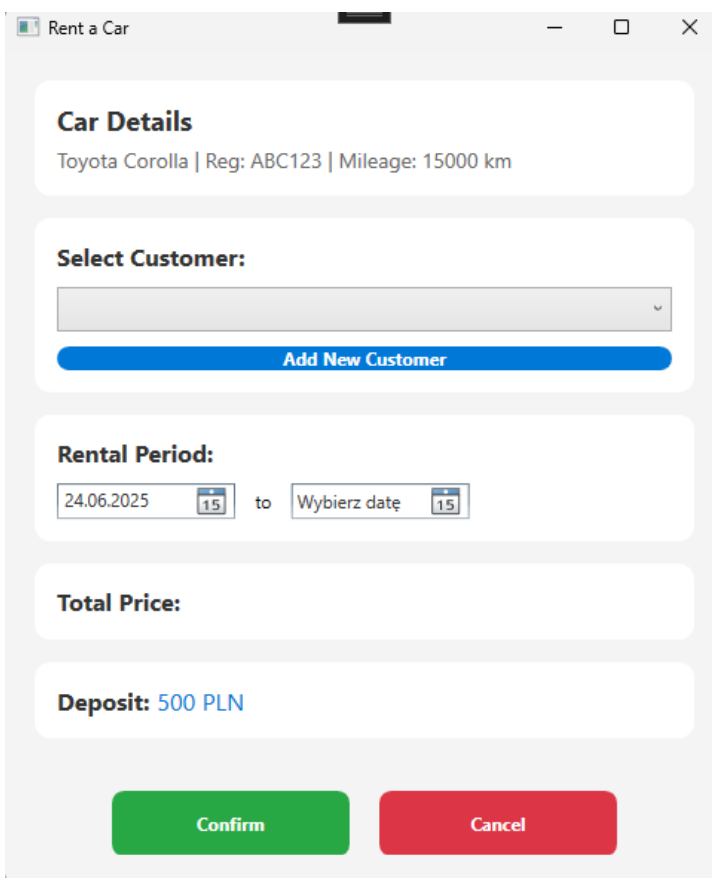
Przyciski

Każdy przycisk przenosi do osobnego okna, w zależności od funkcji.
Przykładowe okna funkcji „rent” w „Show Cars” dla Administratora”

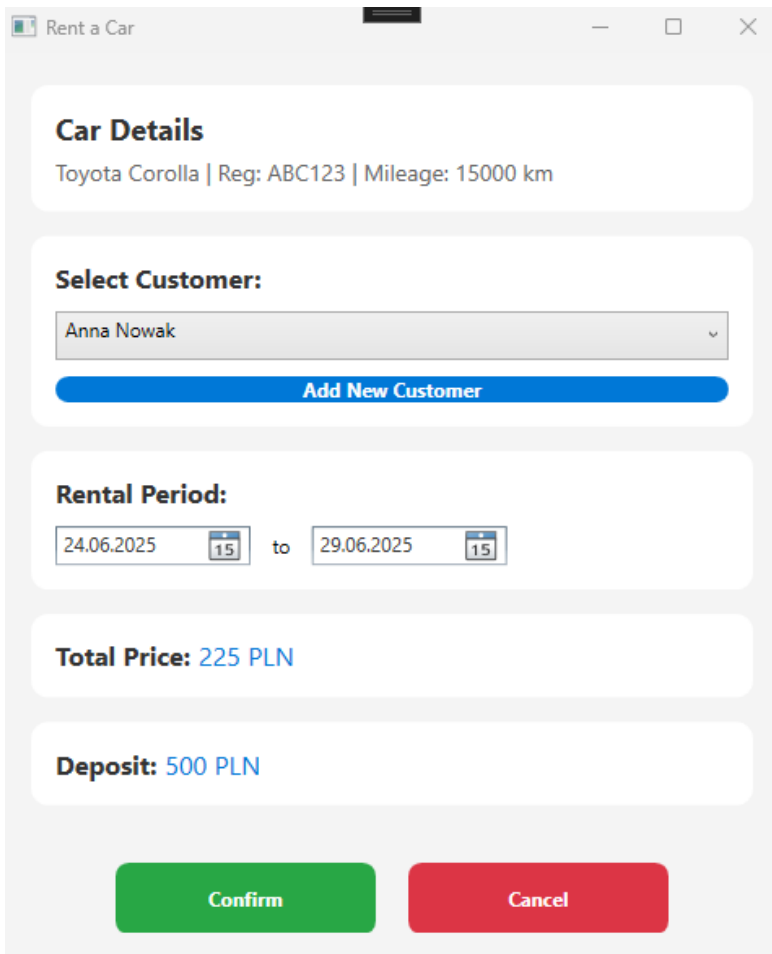
Jeżeli nie został wybrany samochód do wypożyczenia, program wyświetli błąd:



Po wybraniu pojazdu pokazuje się następujące okno:

A screenshot of a "Rent a Car" form. The form is titled "Rent a Car" and has a close button (X). It contains several sections: "Car Details" with "Toyota Corolla | Reg: ABC123 | Mileage: 15000 km"; "Select Customer:" with a dropdown menu and an "Add New Customer" button; "Rental Period:" with a date picker showing "24.06.2025" and a "Wybierz datę" (Choose date) button; "Total Price:"; and "Deposit: 500 PLN". At the bottom, there are two buttons: "Confirm" (green) and "Cancel" (red).

Możliwe jest wybranie klienta, okresu wypożyczenia, po czym wyświetlana jest cena. Operację można potwierdzić, albo anulować



The screenshot shows a web application window titled "Rent a Car". It contains several sections: "Car Details" showing "Toyota Corolla | Reg: ABC123 | Mileage: 15000 km"; "Select Customer:" with a dropdown menu showing "Anna Nowak" and a blue "Add New Customer" button; "Rental Period:" with date pickers for "24.06.2025" and "29.06.2025" (both with "15" in a small box) separated by "to"; "Total Price: 225 PLN"; and "Deposit: 500 PLN". At the bottom are two buttons: a green "Confirm" button and a red "Cancel" button.

Pozostałe funkcje działają na podobnej logice

Opis działania aplikacji

Logowanie i autoryzacja

- Użytkownik wpisuje login/hasło; hasło przechowywane jako `SecureString`.
- `LoginViewModel` waliduje długość wpisu (min. 3 znaki), następnie wywołuje `UserRepository.AuthenticateUser()`.
- Po pomyślnej autoryzacji zapisuje sesję w singletonie `UserSession` i otwiera odpowiednie okno (rola Admin lub Pracownik).

Zarządzanie danymi (Repozytoria + baza SQL)

- Dane przechowywane są w relacyjnej bazie
- Dostęp do danych realizowany jest w warstwie `Repositories` – oddziela logikę dostępu od `ViewModeli`;
- Operacje `Create`, `Read`, `Update`, `Delete` na samochodach, klientach i użytkownikach.

Wyliczanie ceny i scenariusz wypożyczenia

- Wybór auta i klienta, daty wypożyczenia i zwrotu.
- Obliczenie ceny na podstawie jednodniowej stawki $\text{Car.DailyRate} * \text{liczba dni} + \text{ewentualne opłaty dodatkowe (np. serwisowe)}$.
- Zapis transakcji do tabeli `Rentals` z nadaniem statusu.

User Interface i obsługa zdarzeń

- Menu boczne zmienia zawartość głównego obszaru (`UserControl`).
- `Converters` zamieniają np. `enum` statusu na kolor/tekst w UI.

- CustomControls standaryzują wygląd przycisków, data pickera itp.
- Obsługa błędów i wyświetlanie komunikatów (brak wybranego auta, nieprawidłowe dane).

Algorytmika i techniczne rozwiązania

- Data Binding – automatyczne odświeżanie UI przy zmianie właściwości w ViewModelu.
- ICommand – zamiast zdarzeń przycisków, co pozwala na testowanie logiki niezależnie od UI.
- SecureString – bezpieczne przechowywanie hasła w pamięci.
- Singleton – UserSession trzyma stan zalogowanego użytkownika.
- Repository Pattern – enkapsuluje logikę dostępu do bazy.
- Walidacja Inputu – CanExecute komend sprawdza poprawność danych przed wykonaniem operacji.
- Obliczenia cen – prosta arytmetyka, możliwość rozszerzenia o promocje czy cenniki sezonowe.

Fragmenty kodu

Zastosowane funkcje i technologie na przykładzie funkcji logowania do aplikacji:

Implementacja dziedziczenia – ViewModelBase

```
Odwolania: 11
public abstract class ViewModelBase : INotifyPropertyChanged
{
    public event PropertyChangedEventHandler PropertyChanged;

    Odwołania: 22
    public void OnPropertyChanged(string propertyName)
    {
        PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
    }
}
```

ViewModelBase – Klasa bazowa dla wszystkich ViewModels (w naszym przypadku LoginViewModel). Umożliwia powiadamianie widoku o zmianie danych (NotifyPropertyChanged) – kluczowe dla data bindingu w WPF.

Dziedziczenie w LoginViewModel

```
public class LoginViewModel : ViewModelBase
{
    // Fields
    private string _username;
    private SecureString _password;
    private string _errorMessage;
    private bool _isViewVisible = true;
    private bool _isAccess;

    private IUserRepository userRepository;

    // Properties
    Odwołania: 6
    public string Username
    {
        get { return _username; }
        set
        {
            _username = value;
            OnPropertyChanged(nameof(Username));
        }
    }
}
```

LoginViewModel dziedziczy z ViewModelBase, co umożliwia korzystanie z OnPropertyChanged(...) i automatyczne odświeżanie UI po zmianie właściwości, np. public string Username

Obsługa zdarzeń

```
1 odwołanie
public ICommand LoginCommand { get; }
1 odwołanie
public ICommand RecoverPasswordCommand { get; }
Odwołania: 0
public ICommand ShowPasswordCommand { get; }
Odwołania: 0
public ICommand RememberPasswordCommand { get; }

// Constructor
Odwołania: 0
public LoginViewModel()
{
    userRepository = new UserRepository();
    LoginCommand = new ViewModelCommand(ExecuteLoginCommand, CanExecuteLoginCommand);
    RecoverPasswordCommand = new ViewModelCommand(p => ExecuteRecoverPassCommand("", ""));
}
```

Komenda LoginCommand wiązana w widoku do przycisku „Zaloguj”. ViewModelCommand to klasa implementująca ICommand, umożliwiającą wykonywanie logiki po kliknięciu przycisku.

Walidacja danych przed logowaniem

```
private bool CanExecuteLoginCommand(object obj)
{
    bool validData = !string.IsNullOrEmpty(Username) && Username.Length >= 3 &&
        Password != null && Password.Length >= 3;
    return validData;
}
```

Zabezpieczenie – przycisk logowania aktywny tylko wtedy, gdy dane wejściowe są poprawne (minimum 3 znaki).

Mechanizm autoryzacji logowania

```
private void ExecuteLoginCommand(object obj)
{
    var password = Password;

    var isValidUser = userRepository.AuthenticateUser(new NetworkCredential(Username, password));

    if (isValidUser)
    {
        // Ustawienie aktualnego użytkownika
        App.CurrentUser = userRepository.GetByUsername(Username);

        // Ustawienie danych sesji
        UserSession.Username = App.CurrentUser.Username;
        UserSession.UserId = App.CurrentUser.UserId;

        // Ustawienie tożsamości
        Thread.CurrentPrincipal = new System.Security.Principal.GenericPrincipal(
            new System.Security.Principal.GenericIdentity(Username), null);

        IsAccess = App.CurrentUser?.Access ?? false;

        if (IsAccess)
        {
            var adminMenu = new Admin_Menu();
            adminMenu.Show();
        }
        else
        {
            var userMenu = new User_Menu();
            userMenu.Show();
        }

        IsViewVisible = false;
    }
    else
    {
        ErrorMessage = "* Invalid username or password";
    }
}
```

Funkcja odpowiada za: Weryfikację danych użytkownika (AuthenticateUser), ustawienie aktualnego użytkownika i jego roli, utworzenie odpowiedniego widoku (Admin lub User), oraz ukrycie okna logowania (IsViewVisible = false).

Bezpieczne przechowywanie hasła

```
public SecureString Password
{
    get { return _password; }
    set
    {
        _password = value;
        OnPropertyChanged(nameof(Password));
    }
}
```

Hasło przechowywane jako SecureString, co zwiększa bezpieczeństwo aplikacji – minimalizuje ryzyko przechwycenia danych z pamięci.

Obsługa zdarzeń w oknie – Login_Screen.xaml.cs

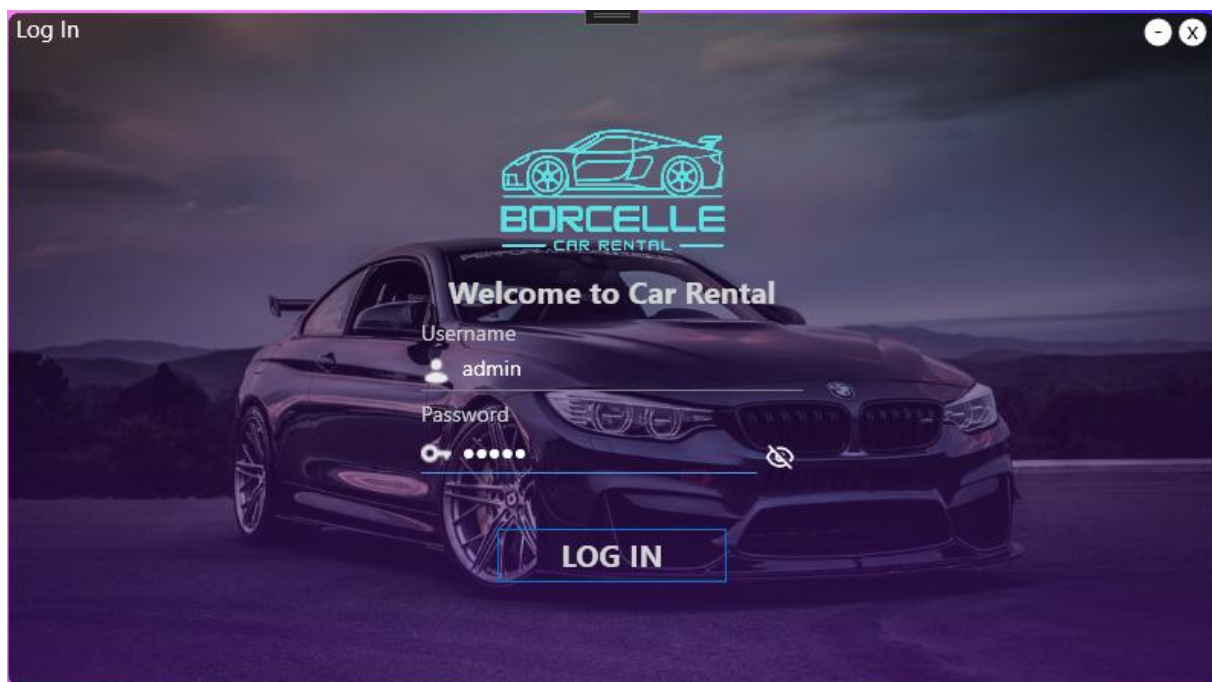
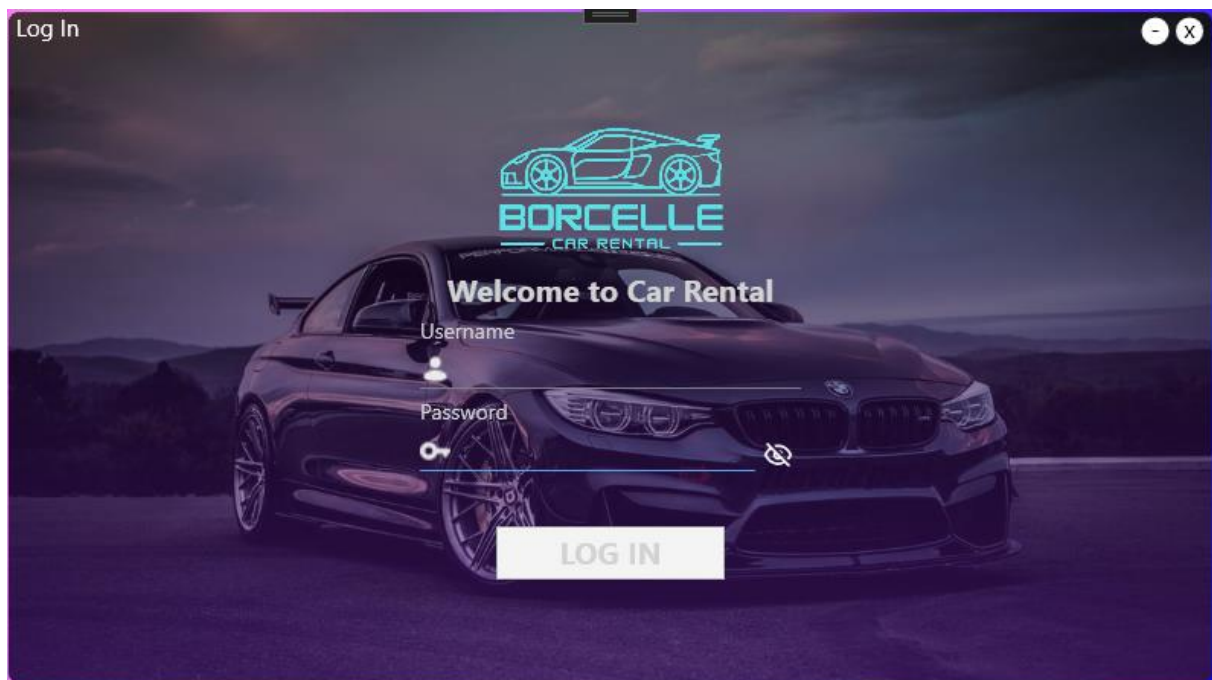
```
private void Login_Screen_Mouse_Down(object sender, MouseButtonEventArgs e)
{
    if (e.ChangedButton == MouseButton.Left)
    {
        DragMove();
    }
}
```

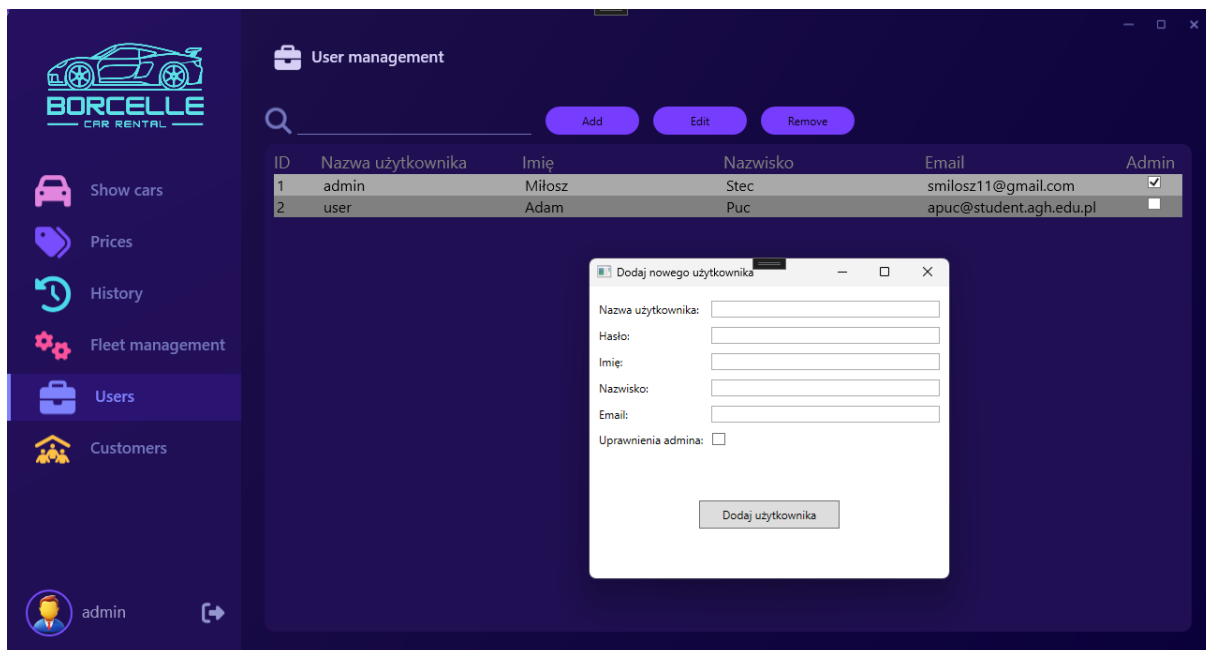
Prosty event pozwalający użytkownikowi przeciągać okno

Testowanie i przykładowe scenariusze działania

Scenariusz dla Administratora

Logowanie, dodanie pracownika do listy, następnie usunięcie jednego pojazdu z floty





Dodaj nowego użytkownika

Nazwa użytkownika:

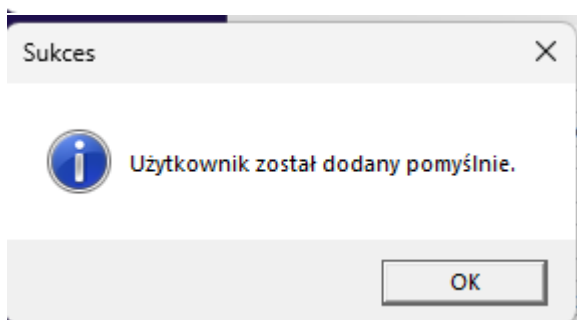
Hasło:

Imię:

Nazwisko:


Email:

Uprawnienia admina: ☐



ID	Nazwa użytkownika	Imię	Nazwisko	Email	Admin
1	admin	Miłosz	Stec	smilosz11@gmail.com	<input checked="" type="checkbox"/>
2	user	Adam	Puc	apuc@student.agh.edu.pl	<input type="checkbox"/>
9	User	Piotr	Nowak	pnowak@gmail.com	<input type="checkbox"/>

Przypadek źle wpisanych danych:

 Dodaj nowego użytkownika

Nazwa użytkownika:

Hasło:


Imię:

Nazwisko:

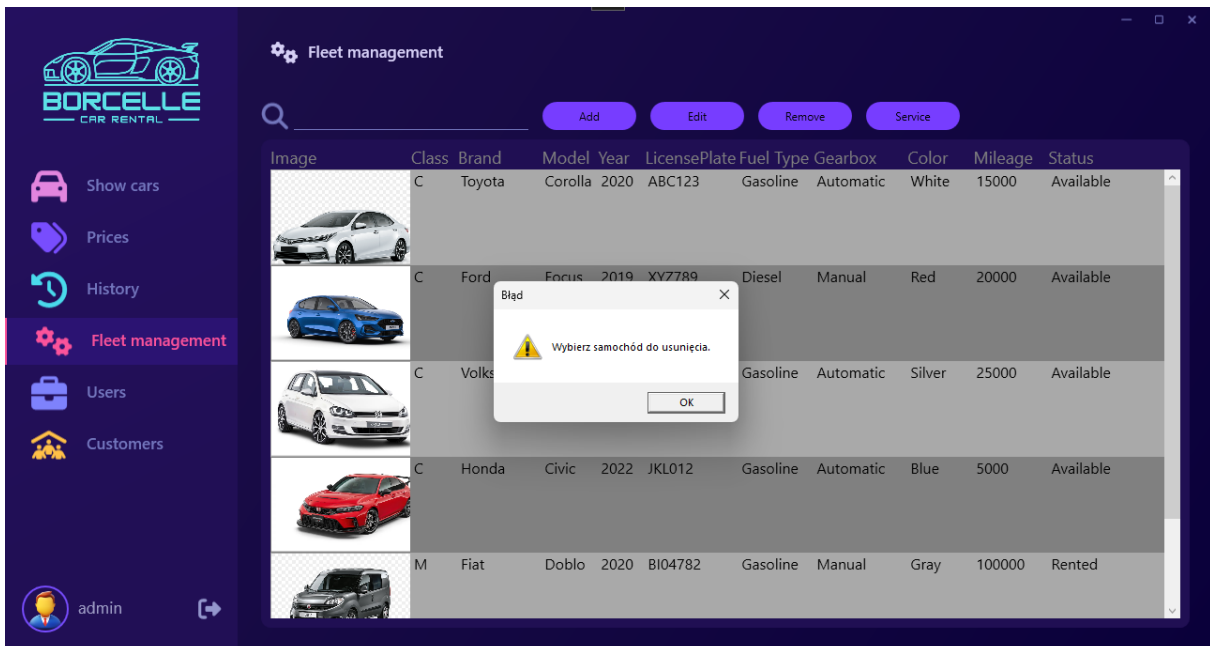
Email:

Uprawnienia admina: ☒

Błąd





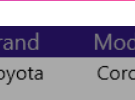
 Proszę uzupełnić wszystkie pola.

Edycja Floty – brak wybranego pojazdu do usunięcia:



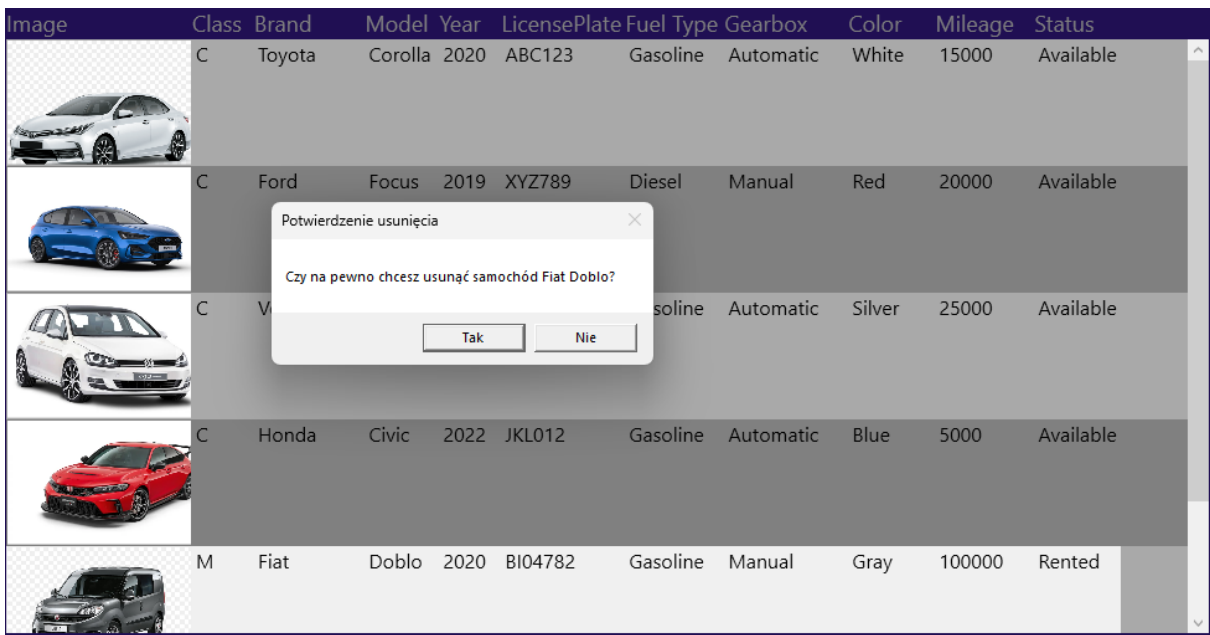
The screenshot shows the 'Fleet management' interface with a sidebar menu and a table of cars. An error dialog box is displayed over the table.

Table Data:

Image	Class	Brand	Model	Year	LicensePlate	Fuel Type	Gearbox	Color	Mileage	Status
	C	Toyota	Corolla	2020	ABC123	Gasoline	Automatic	White	15000	Available
	C	Ford	Focus	2019	XYZ789	Diesel	Manual	Red	20000	Available
	C	Volkswagen	Golf	2018	DEF456	Gasoline	Automatic	Silver	25000	Available
	C	Honda	Civic	2022	JKL012	Gasoline	Automatic	Blue	5000	Available
	M	Fiat	Doblo	2020	BI04782	Gasoline	Manual	Gray	100000	Rented






Error Dialog Box:

Błąd
Wybierz samochód do usunięcia.
OK







The screenshot shows the 'Fleet management' interface with a table of cars. A confirmation dialog box is displayed over the table.

Table Data:

Image	Class	Brand	Model	Year	LicensePlate	Fuel Type	Gearbox	Color	Mileage	Status
	C	Toyota	Corolla	2020	ABC123	Gasoline	Automatic	White	15000	Available
	C	Ford	Focus	2019	XYZ789	Diesel	Manual	Red	20000	Available
	C	Volkswagen	Golf	2018	DEF456	Gasoline	Automatic	Silver	25000	Available
	C	Honda	Civic	2022	JKL012	Gasoline	Automatic	Blue	5000	Available
	M	Fiat	Doblo	2020	BI04782	Gasoline	Manual	Gray	100000	Rented

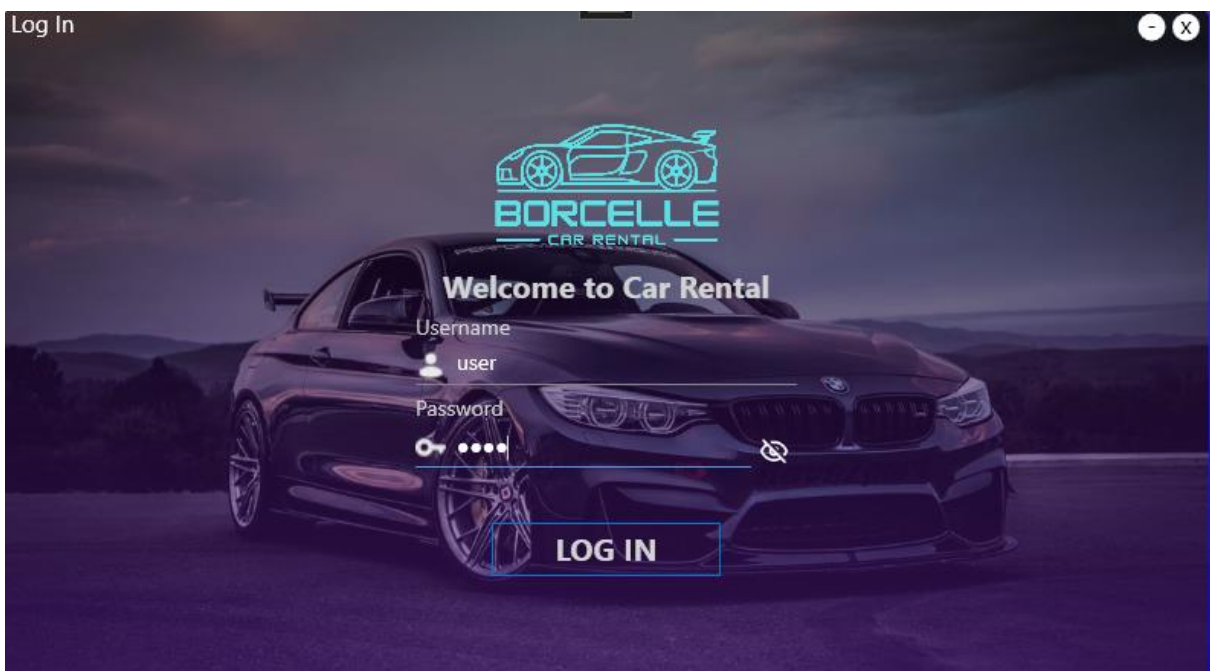
Confirmation Dialog Box:

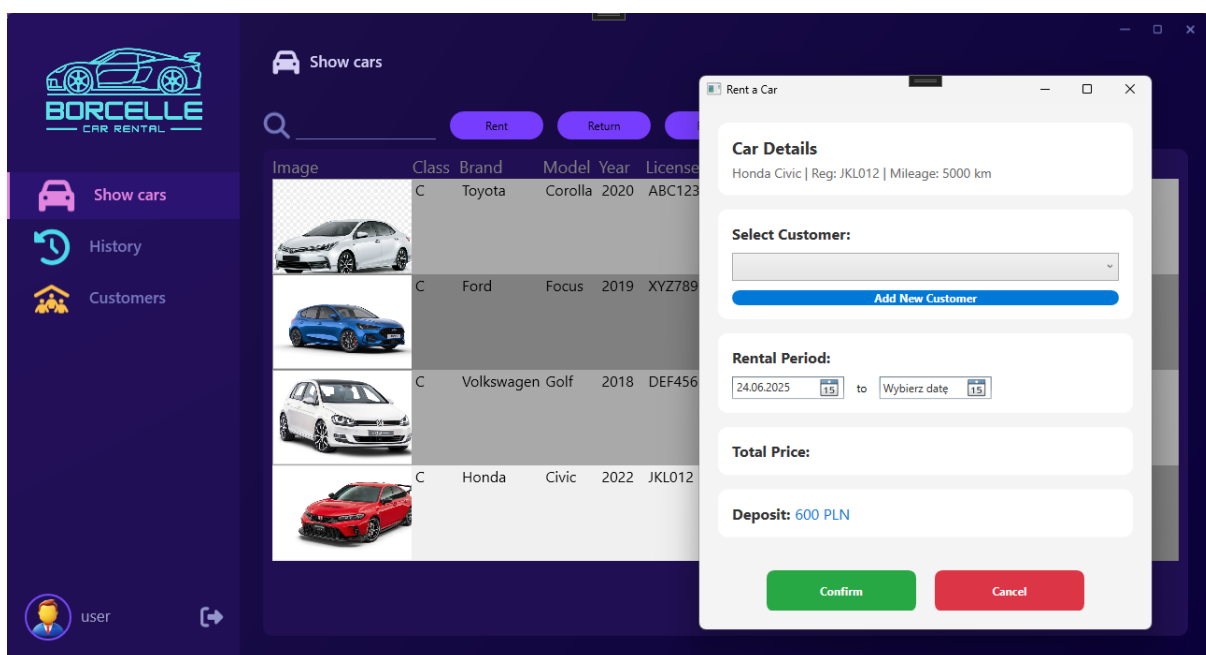
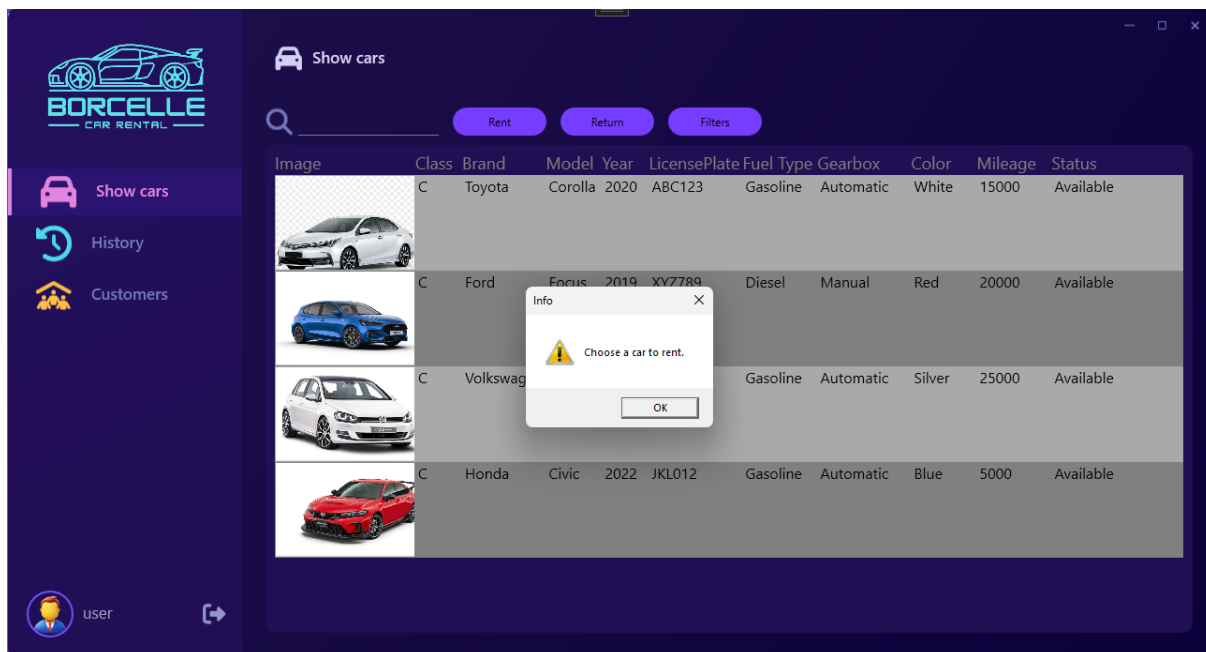
Potwierdzenie usunięcia
Czy na pewno chcesz usunąć samochód Fiat Doblo?
Tak Nie

Image	Class	Brand	Model	Year	LicensePlate	Fuel Type	Gearbox	Color	Mileage	Status
	C	Toyota	Corolla	2020	ABC123	Gasoline	Automatic	White	15000	Available
	C	Ford	Focus	2019	XYZ789	Diesel	Manual	Red	20000	Available
	C	Volkswagen	Golf	2018	DEF456	Gasoline	Automatic	Silver	25000	Available
	C	Honda	Civic	2022	JKL012	Gasoline	Automatic	Blue	5000	Available

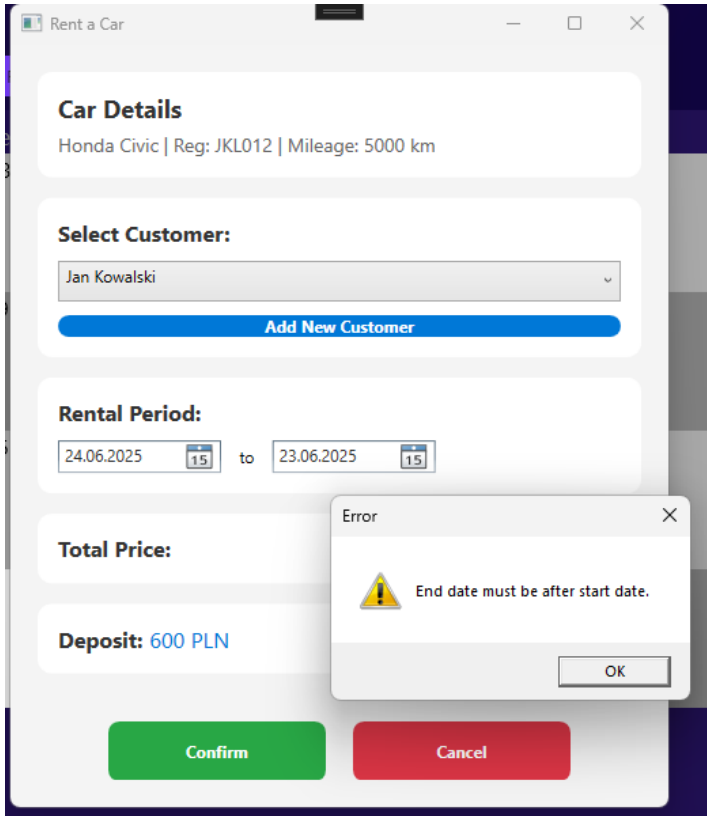
Scenariusz dla Użytkownika

Logowanie, wypożyczenie oraz odebranie samochodu, dodanie nowego klienta do bazy danych

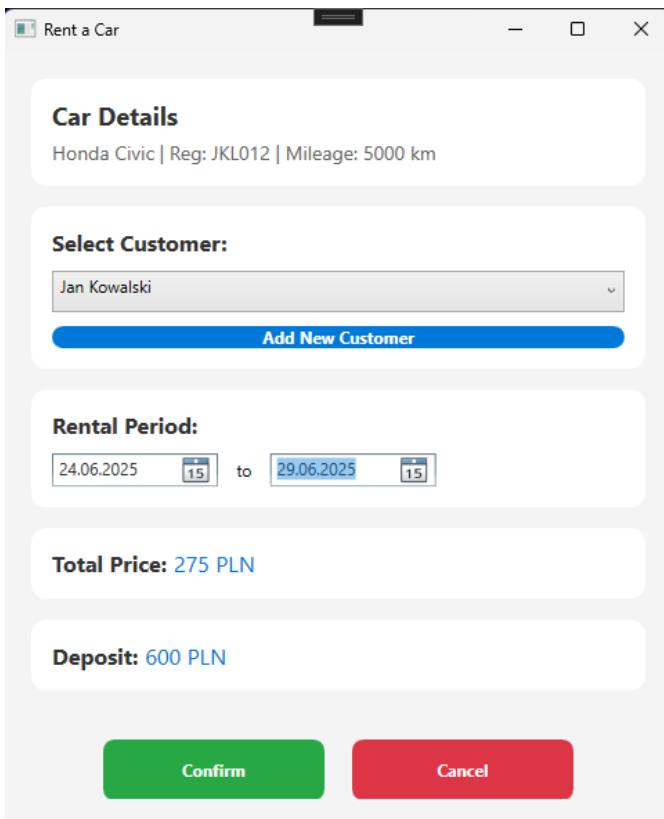




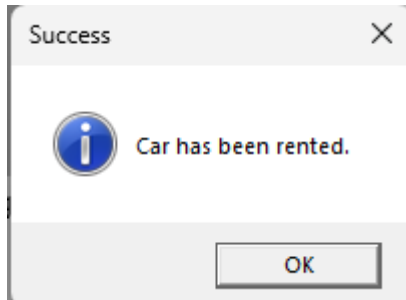
Data oddania jest nieprawidłowa – program pokazuje błąd:



The screenshot shows the 'Rent a Car' application window. The 'Car Details' section displays 'Honda Civic | Reg: JKL012 | Mileage: 5000 km'. The 'Select Customer:' dropdown is set to 'Jan Kowalski', with an 'Add New Customer' button below it. The 'Rental Period:' section shows a start date of '24.06.2025' and an end date of '23.06.2025'. The 'Total Price:' field is empty, and the 'Deposit:' is '600 PLN'. An error dialog box is overlaid on the form, displaying a yellow warning icon and the message 'End date must be after start date.' with an 'OK' button.



The screenshot shows the 'Rent a Car' application window after the error has been resolved. The 'Car Details' and 'Select Customer:' sections remain the same. The 'Rental Period:' section now shows a start date of '24.06.2025' and a corrected end date of '29.06.2025'. The 'Total Price:' field now displays '275 PLN', and the 'Deposit:' remains '600 PLN'. The 'Confirm' and 'Cancel' buttons are visible at the bottom.



47	JKL012	Unknown	Anna Nowak	98765432109	7/3/2025 12:00:00 AM	7/5/2025 12:00:00 AM	Active	120
48	JKL012	user	Jan Kowalski	12345678901	6/24/2025 12:00:00 AM	6/29/2025 12:00:00 AM	Active	275

Odbiór samochodu:

Return Car

Choose reservation to return:

#48 | 24.06.2025 → 29.06.2025 | Jan Kowalski | JKL012

Car details:
Marka: Honda, Model: Civic, Rok: 2022, Rejestracja: JKL012

Customer:
Klient: Jan Kowalski
PESEL: 12345678901

Return time:
2025-06-24 20:05:31

Mileage (km):
5000

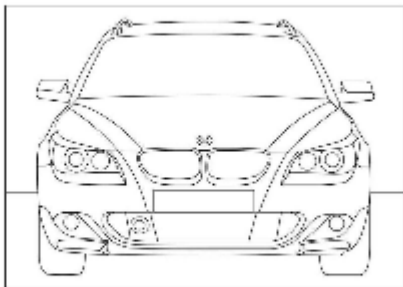
Damage marking (optional):

Front

Back

Left

Right



Confirm return

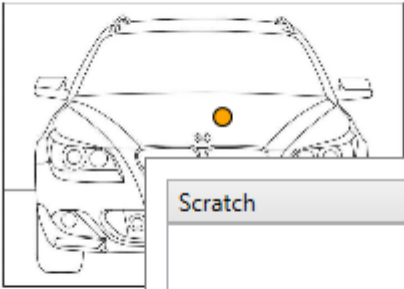
Damage marking (optional):

Front

Back

Left

Right




Scratch

Add

Confirm return

Success



Car returned successfully!

OK

ID	Vehicle	User	Name	Phone	Start Date	Start Time	End Date	End Time	Status	Score
47	JKL012	Unknown	Anna Nowak	98765432109	7/3/2025	12:00:00 AM	7/5/2025	12:00:00 AM	Active	120
48	JKL012	user	Jan Kowalski	12345678901	6/24/2025	12:00:00 AM	6/29/2025	12:00:00 AM	Finished	275

Dodanie Klienta:

Customer management

Search

Add

Edit

Remove

Name	Surname	Email	Phone	Pesel	License
Jan	Kowalski	jan.kowalski@example.com	123456789	12345678901	B12345
Anna	Nowak	anna.nowak@example.com	987654321	98765432109	A76543
Adam	Puc	abc	123	987	xyz

Add New Customer

First Name

Last Name

Email

Phone

PESEL

Driver License Number

Date of Birth

Wybierz datę

Street

City

Postal Code

Add

Cancel

Add New Customer

First Name

Monika

Last Name

Konieczna

Email

m.konieczna@gmail.com

Phone

123456789

PESEL

12345678910

Driver License Number

kk123

Date of Birth

31.03.2000

15

Street

Kowaliowa

City

Kraków

Postal Code

11-111

Add

Cancel

Name	Surname	Email	Phone	Pesel	License number	Date of birth	Street
Jan	Kowalski	jan.kowalski@example.com	123456789	12345678901	B1234567	7/10/1985 12:00:00 AM	ul. Warszawsk
Anna	Nowak	anna.nowak@example.com	987654321	98765432109	A7654321	5/21/1990 12:00:00 AM	ul. Krakowska
Adam	Puc	abc	123	987	xyz	6/4/2025 12:00:00 AM	asd
Monika	Konieczna	m.konieczna@gmail.com	123456789	12345678910	kk123	3/31/2000 12:00:00 AM	Kowaliowa

Wnioski i dalszy rozwój

Aplikacja znacząco ułatwia działanie wypożyczalni samochodowej. Podział na warstwy umożliwia łatwą implementację nowych funkcji jak i testowanie samego programu. Jest ona również przygotowana pod dalszą rozbudowę i integrację z innymi systemami. Wykorzystanie aplikacji okienkowej zapewnia użytkownikowi przyjemny odbiór platformy, a podział na mniejsze zadania ułatwia mu poruszanie się po aplikacji. Kolejnymi etapami rozwoju będzie dodanie dodatkowych zabezpieczeń (np. weryfikacja, czy klient posiada prawo jazdy), integracja z zewnętrznym systemem do płatności internetowych oraz projekt okna dla klienta.