

RozliczenieApp

Temat i cel biznesowy aplikacji

RozliczenieApp to aplikacja desktopowa wspomagająca zarządzanie pracą kierowców. Głównym celem biznesowym jest umożliwienie efektywnego rejestrowania i rozliczania kursów wykonywanych przez kierowców, w tym:

- Przechowywania danych kierowców, ich pojazdów oraz wykonanych kursów.
- Automatycznego obliczania zarobków kierowców na podstawie zarejestrowanych danych.
- Wspierania logistyki i księgowości w małych firmach transportowych.

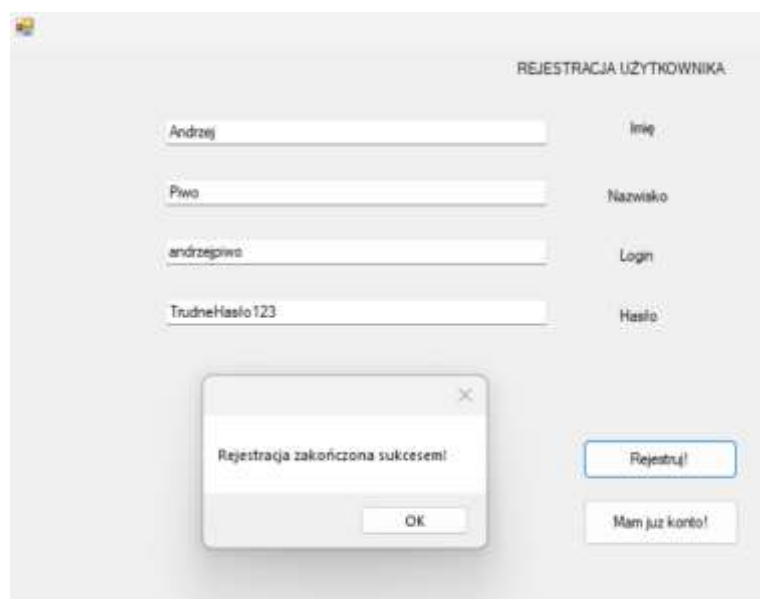
Użyte technologie

- **Język programowania:** C# (.NET Framework)
- **Baza danych:** SQLite
- **Zarządzanie ORM:** Entity Framework
- **IDE:** Visual Studio
- **Biblioteki:**
 - System.Data.SQLite
 - EntityFramework

Lista funkcjonalności

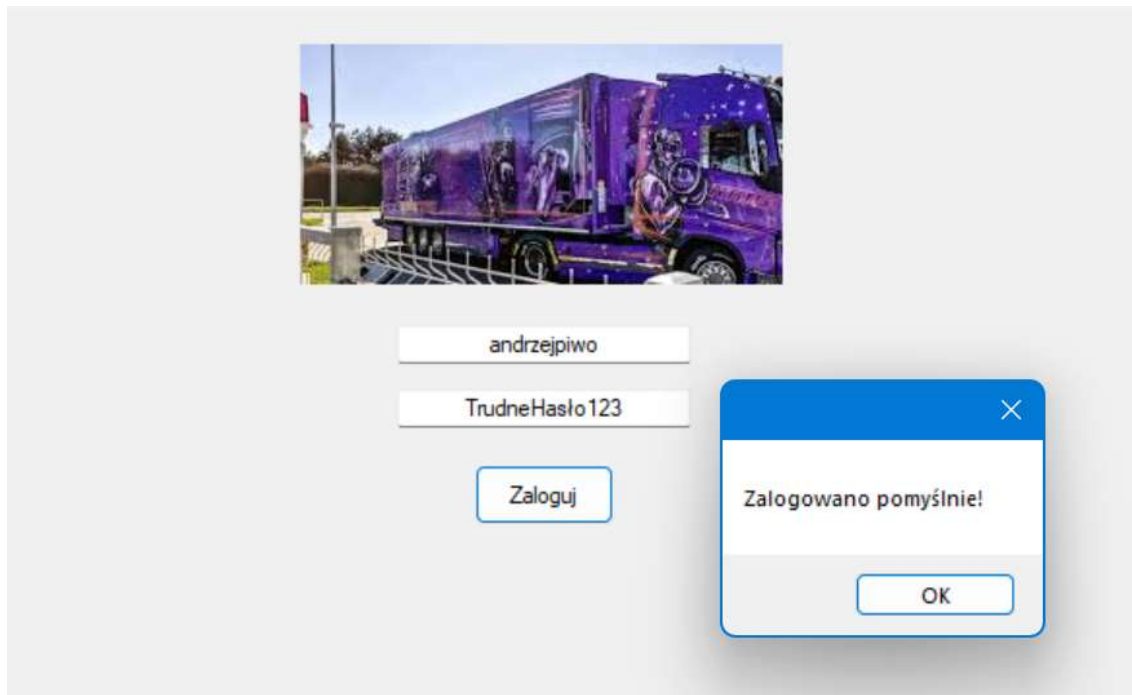
1. Zarządzanie użytkownikami

- Rejestracja nowych użytkowników.



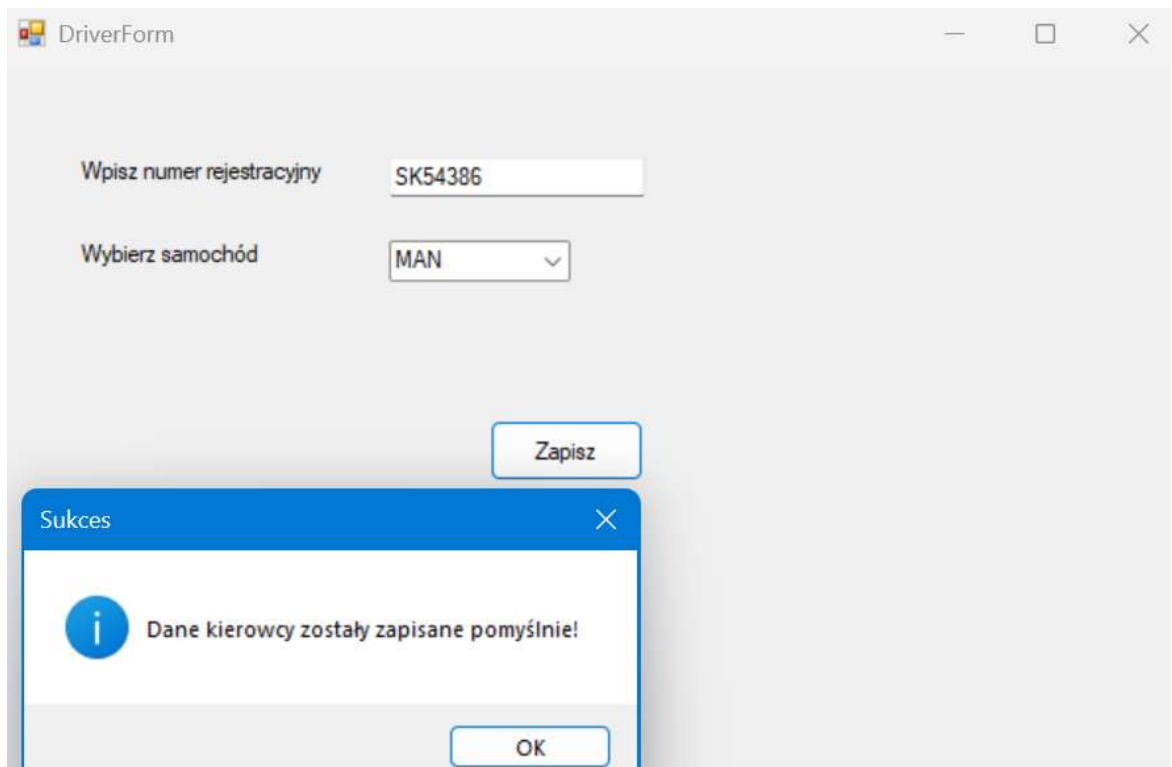
The screenshot shows a desktop application window titled "REJESTRACJA UŻYTKOWNIKA". It contains four input fields arranged in two columns. The left column has fields for "Imię" (First Name) with the value "Andrzej", "Nazwisko" (Last Name) with the value "Piwo", "Login" with the value "andrzejpiwo", and "Hasło" (Password) with the value "TrudneHaslo123". The right column has corresponding labels. Below the input fields, there is a "Rejestruj!" button and a "Mam już konto!" button. A small dialog box is open in the foreground, displaying the message "Rejestracja zakończona sukcesem!" (Registration completed successfully!) and an "OK" button.

- Logowanie istniejących użytkowników z walidacją danych.



2. Zarządzanie kierowcami

- Dodawanie nowych kierowców do systemu.



- Przechowywanie informacji o kierowcy, takich jak imię, nazwisko, numer rejestracyjny pojazdu i typ pojazdu.



3. Rejestrowanie kursów

- Dodawanie informacji o kursach, w tym daty, liczby kursów i zarobionej kwoty.
- Automatyczne obliczanie wynagrodzenia na podstawie stawki zależnej od dnia tygodnia.
- Wyświetlanie listy zarejestrowanych kursów dla danego kierowcy.

The screenshot shows a web application interface for registering courses. At the top, there are three fields: 'Imię i nazwisko: Andrzej Piwo', 'Numer rejestracyjny: SK54386', and 'Marka samochodu: MAN'. Below these, there are three input fields: 'Data kursu' (set to 'sobota, 18 stycznia 2025'), 'Liczba kursów', and 'Opis kursów'. A 'Dodaj kurs' button is at the bottom. On the right, a list of registered courses is displayed:

- Data: 18.01.2025, Opis: Tychy Lodowisko, Liczba kursów: 1, Kwota: 180.00 zł
- Data: 01.01.2025, Opis: Galeria Katowicka, Liczba kursów: 2, Kwota: 230.00 zł
- Data: 10.01.2025, Opis: Katowice Kopalnia Wujek, Liczba kursów: 1, Kwota: 230.00 zł
- Data: 07.01.2025, Opis: Katowice IKEA, Liczba kursów: 1, Kwota: 230.00 zł

4. Obsługa bazy danych SQLite

- Przechowywanie wszystkich danych (użytkownicy, kierowcy, kursy) w lekkiej bazie danych SQLite.

The screenshot shows a SQLite database viewer with the 'Courses' table selected. The table has 4 records. The columns are: Id, DriverUsername, Date, Description, TripsCount, and Amount.

Id	DriverUsername	Date	Description	TripsCount	Amount
1	andrzejpiwo	2025-01-18	Tychy Lodowisko	1	180
2	andrzejpiwo	2025-01-01	Galeria Katowicka	2	230
3	andrzejpiwo	2025-01-10	Katowice Kopalnia ...	1	230
4	andrzejpiwo	2025-01-07	Katowice IKEA	1	230

- Automatyczne tworzenie tabel przy pierwszym uruchomieniu aplikacji.

Instalacja i uruchomienie

Krok 1: Klonowanie repozytorium

Skopiuj kod z repozytorium GitHub:

```
git clone https://github.com/your-username/rozliczenieapp.git
```

```
cd rozliczenieapp
```

Krok 2: Przywrócenie pakietów NuGet

W Visual Studio otwórz projekt i uruchom:

```
nuget restore
```

Krok 3: Budowanie i uruchamianie projektu

W Visual Studio kliknij przycisk "Start" lub naciśnij F5.

Opis architektury

Aplikacja opiera się na paradygmacie obiektowym, a jej główne elementy to:

Klasy

1. User (klasa bazowa):

- Właściwości: Id, FirstName, LastName, Username, Password.
- Dziedziczona przez klasę Driver.

2. Driver (klasa pochodna):

- Dodatkowe właściwości: LicensePlate, CarType.

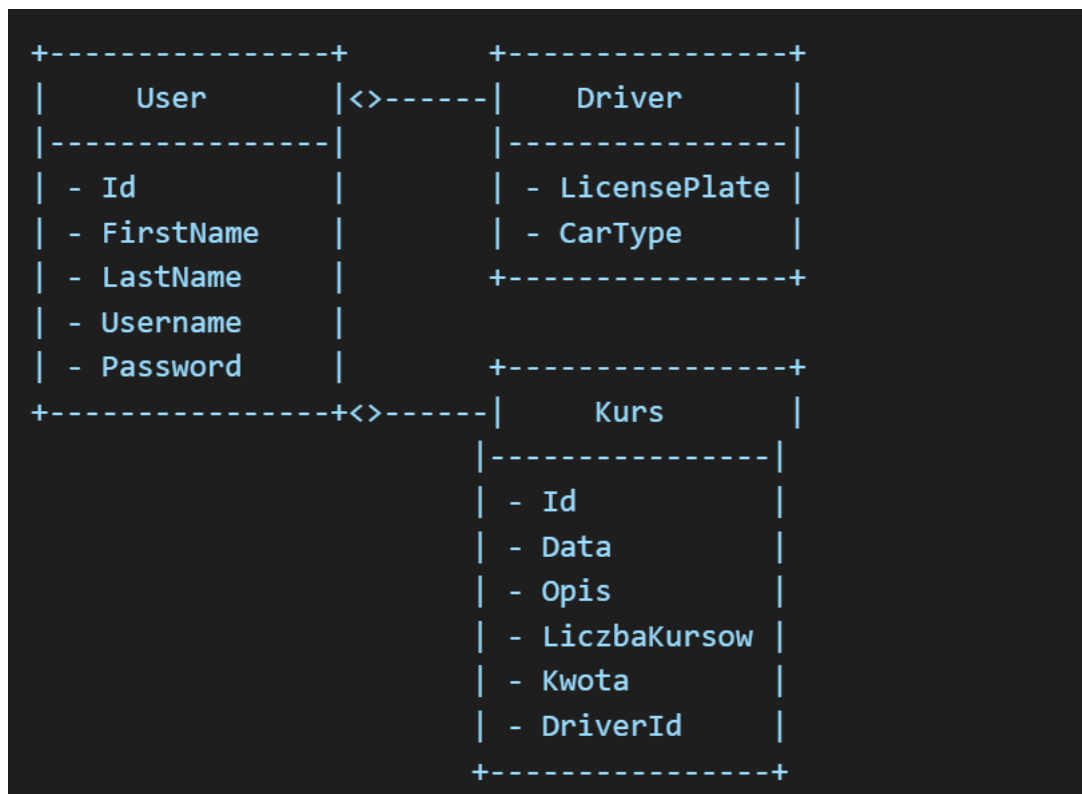
3. Kurs:

- Właściwości: Id, Data, Opis, LiczbaKursow, Kwota, DriverId (klucz obcy).

4. DatabaseHelper:

- Odpowiada za operacje CRUD na bazie danych SQLite (dodawanie użytkowników, kierowców i kursów).

Diagram klas UML



Architektura

- **Encapsulacja:** Dane użytkowników, kierowców i kursów są przechowywane w odpowiednich klasach.
- **Dziedziczenie:** Klasa Driver dziedziczy po klasie User, rozszerzając jej funkcjonalność o specyficzne dane kierowcy.
- **Abstrakcja:** Użytkownik końcowy korzysta z interfejsów graficznych, które abstrahują szczegóły dotyczące bazy danych.
- **Polimorfizm:** Możliwość rozbudowy klasy User o dodatkowe typy użytkowników.

Licencja

Projekt jest dostępny na licencji MIT. Szczegóły znajdziesz w pliku LICENSE.

Autorzy

- Mateusz Sendek
- Denis Skowronek
- Kamil Szótysek