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.
- 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.

4. Obsługa bazy danych SQLite

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

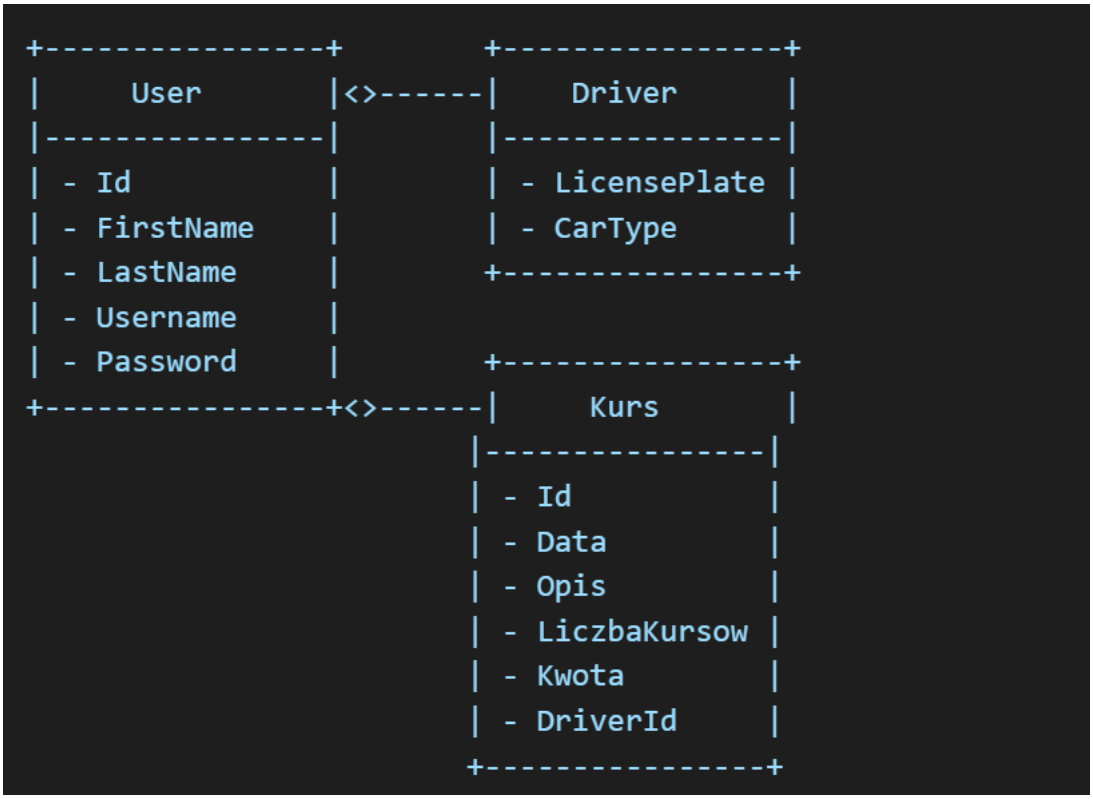
Opis architektury

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

- **Dziedziczenie:** Klasa Driver dziedziczy po klasie User, rozszerzając jej funkcjonalność o specyficzne dane kierowcy.

```
public class Driver : Uzytkownik
{
    public string LicensePlate { get; set; }
    public string CarType { get; set; }

    public Driver(string firstName, string lastName, string username, string licensePlate, string carType)
        : base(firstName, lastName, username)
    {
        LicensePlate = licensePlate;
        CarType = carType;
    }

    // Nadpisanie metody DisplayInfo
    public override void DisplayInfo()
    {
        base.DisplayInfo(); // Wywołanie metody z klasy bazowej
        Console.WriteLine($"Kierowca pojazdu: {CarType}, Numer rejestracyjny: {LicensePlate}");
    }
}
```

- **Abstrakcja:** Klasa DriverForm używa jedynie metod AddDriver które są zawarte w DatabaseHelper.

```
if (string.IsNullOrWhiteSpace(licensePlate) || string.IsNullOrWhiteSpace(carType))

    MessageBox.Show("Pola muszą być wypełnione!", "Błąd", MessageBoxButtons.OK, MessageBoxIcon.Error);
    return;


ar dbHelper = new DatabaseHelper();
bHelper.AddDriver(_uzytkownik.FirstName, _uzytkownik.LastName, _uzytkownik.Username, _uzytkownik.Password, licensePlate, carType);
```

Plany na dalszy rozwój aplikacji

- Modyfikacja listy kursów
    - Dodanie możliwości edycji kursów: użytkownik będzie mógł aktualizować liczbę kursów, opisy oraz kwoty.
    - Wprowadzenie funkcji usuwania kursów, umożliwiającej zarządzanie listą kursów.
- Podsumowanie miesiąca
    - Implementacja funkcji podsumowania miesiąca, wyświetlającej całkowite zarobki za wybrany miesiąc.
    - Dodanie szczegółowego podsumowania, obejmującego sumę kursów i kwot dla poszczególnych dni oraz łączną kwotę zarobioną w miesiącu.
    - Wprowadzenie filtrowania danych, umożliwiającego przeglądanie wyników dla dowolnego miesiąca i roku.
- Eksport danych do pliku PDF
    - Implementacja eksportu danych do pliku PDF, generującego raport zawierający: daty, liczby kursów, opisy, zarobki za każdy dzień oraz całkowite zarobki w miesiącu.
    - Integracja biblioteki QuestPDF w celu generowania raportów PDF.

Licencja

Autorzy

- Mateusz Sendek
- Denis Skowronek
- Kamil Szołtysek