Analiza specyfikacji wymagań

Projekt: Aplikacja do wypożyczania samochodów (C#/XAML)

Data: 2025-06-22

# 1. Wstęp

Dokument prezentuje analizę wymagań dla aplikacji okienkowej do zarządzania wypożyczalnią samochodów. Celem jest dokładne określenie zakresu funkcjonalności i wymagań jakościowych systemu.

# 2. Cele analizy

- Zdefiniowanie funkcjonalności związanych z obsługą samochodów i rezerwacji.  
- Identyfikacja ograniczeń i wymagań niefunkcjonalnych.  
- Ustalenie sposobu przetwarzania i ochrony danych klientów.

# 3. Zakres specyfikacji

- Przeglądanie samochodów: Lista dostępnych pojazdów wraz ze szczegółami (marka, model, VIN).  
- Dodawanie samochodów: Możliwość dodawania nowych pojazdów do bazy.  
- Edycja i usuwanie samochodów: Edytowanie danych pojazdów lub ich usuwanie z systemu.  
- Rezerwacja samochodu: Tworzenie rezerwacji dla klienta, powiązanie samochodu z danymi klienta i terminem wypożyczenia.  
- Edycja daty zwrotu rezerwacji: Możliwość modyfikacji daty końcowej wypożyczenia.  
- Usuwanie rezerwacji: Możliwość anulowania rezerwacji.  
- Wyświetlanie listy rezerwacji: Przeglądanie wszystkich zarejestrowanych wypożyczeń.

# 4. Analiza wymagań funkcjonalnych – model use case’ów

## 4.1 Specyfikacja aktorów

- Użytkownik (pracownik wypożyczalni):  
 - Przegląda samochody  
 - Dodaje/edytuje/usuwa samochody  
 - Tworzy/usuwa rezerwacje  
 - Edytuje datę zwrotu

## 4.2 Lista use case’ów

- Przeglądanie samochodów  
- Dodawanie samochodu  
- Edycja samochodu  
- Usuwanie samochodu  
- Tworzenie rezerwacji  
- Usuwanie rezerwacji  
- Edycja daty zwrotu  
- Przeglądanie rezerwacji

## 4.3 Diagram use case’ów UML

A screenshot of a cell phone

Description automatically generated

## 4.4 Specyfikacja use case’ów

- Przeglądanie samochodów: Użytkownik wyświetla listę wszystkich pojazdów, może filtrować dostępność.  
- Dodawanie samochodu: Użytkownik wprowadza dane pojazdu (marka, model, VIN).  
- Edycja samochodu: Użytkownik zmienia dane istniejącego pojazdu.  
- Usuwanie samochodu: Użytkownik usuwa pojazd z bazy.  
- Tworzenie rezerwacji: Użytkownik wybiera samochód, podaje dane klienta, daty wynajmu.  
- Usuwanie rezerwacji: Użytkownik usuwa wybraną rezerwację.  
- Edycja daty zwrotu: Użytkownik zmienia termin zwrotu wypożyczenia.  
- Przeglądanie rezerwacji: Użytkownik przegląda wszystkie istniejące rezerwacje.

# 5. Analiza wymagań niefunkcjonalnych

- Interfejs: Intuicyjny, graficzny (C#/XAML), prosty w obsłudze.  
- Sprzęt: System Windows, obsługa myszy i klawiatury.  
- Komunikacja: Połączenie z API/bazą danych lokalnie lub przez sieć.  
- Bezpieczeństwo: Brak autoryzacji użytkowników; ochrona danych osobowych klientów w ramach zgodności z RODO.

# 6. Analiza wymagań jakościowych

- Wydajność: Szybka reakcja na operacje użytkownika.  
- Niezawodność: Brak krytycznych awarii, obsługa błędów po stronie aplikacji i API.  
- Użyteczność: Interfejs zrozumiały dla osób niebędących informatykami.  
- Dostępność: System gotowy do pracy w godzinach otwarcia firmy.

# 7. Warunki serwisowania

- Łatwość wdrożenia poprawek (prosta architektura, pliki konfiguracyjne).  
- Możliwość rozbudowy o dodatkowe funkcje (np. integrację z płatnościami).  
- Wsparcie techniczne przez administratora systemu.

# 8. Ograniczenia architektury

- Aplikacja desktopowa C#/XAML (.NET)  
- Brak pracy przez przeglądarkę, brak logowania użytkowników.

# 9. Model bazy danych

## 9.1 Specyfikacja atrybutów

Tabela Car:  
 - Id (int)  
 - Brand (string)  
 - Model (string)  
 - VinNumber (string)  
Tabela Rental:  
 - Id (int)  
 - CarId (int)  
 - CustomerName (string)  
 - RentalStartDate (DateTime)  
 - RentalEndDate (DateTime)

## 9.2 Operacje na danych

- Dodawanie, edycja, usuwanie samochodów i rezerwacji.  
- Filtrowanie dostępnych aut.  
- Walidacja terminów rezerwacji.

## 9.3 Reguły poprawności

- VIN musi być unikalny.  
- Termin wypożyczenia nie może zachodzić na inne rezerwacje tego samego auta.  
- Pola wymagane nie mogą być puste.