Analiza specyfikacji wymagań

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

Data: 2025-06-22

Table of Contents

[1. Wstęp 2](#_Toc201829508)

[2. Cele analizy 2](#_Toc201829509)

[3. Zakres specyfikacji 2](#_Toc201829510)

[4. Analiza wymagań funkcjonalnych – model use case’ów 2](#_Toc201829511)

[4.1 Specyfikacja aktorów 2](#_Toc201829512)

[4.2 Lista use case’ów 2](#_Toc201829513)

[4.3 Diagram use case’ów UML 3](#_Toc201829514)

[4.4 Specyfikacja use case’ów 3](#_Toc201829515)

[4.5 Diagramy aktywności UML dla use case’ów 4](#_Toc201829516)

[5. Analiza wymagań niefunkcjonalnych 8](#_Toc201829517)

[6. Analiza wymagań jakościowych 8](#_Toc201829518)

[7. Warunki serwisowania 8](#_Toc201829519)

[8. Ograniczenia architektury 8](#_Toc201829520)

[9. Model bazy danych 8](#_Toc201829521)

[9.1 Specyfikacja atrybutów 8](#_Toc201829522)

[9.2 Operacje na danych 9](#_Toc201829523)

[9.3 Reguły poprawności 9](#_Toc201829524)

# 

# Wstęp

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

# Cele analizy

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

# 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.

# Analiza wymagań funkcjonalnych – model use case’ów

## Specyfikacja aktorów

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

## Lista use case’ów

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

## Diagram use case’ów UML

A screenshot of a cell phone

Description automatically generated

## 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.

## Diagramy aktywności UML dla use case’ów

* + 1. **Przeglądanie samochodów**

A black and white diagram

Description automatically generated

* + 1. **Dodawanie samochodu**

A diagram of a diagram

Description automatically generated

* + 1. **Usuwanie samochodu**

A diagram of a company

Description automatically generated

* + 1. **Modyfikacja samochodu**

A diagram with text on it

Description automatically generated

# 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.

# 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.

# Warunki serwisowania

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

# Ograniczenia architektury

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

# Model bazy danych

## Specyfikacja atrybutów

**Tabela Car – informacje o samochodach**

* Id (int, PK) – unikalny identyfikator samochodu
* Brand (varchar (50)) – marka samochodu
* Model (varchar (50)) – model samochodu
* VinNumber (varchar (17), UNIQUE) – numer VIN, musi być unikalny
* Year (int) – rok produkcji
* Mileage (int) – przebieg w kilometrach

**Tabela Rental – informacje o wypożyczeniach**

* Id (int, PK) – unikalny identyfikator rezerwacji
* CarId (int, FK) – identyfikator wypożyczonego samochodu (klucz obcy do Car.Id)
* CustomerName (varchar(100)) – imię i nazwisko klienta
* RentalStartDate (datetime) – data i godzina rozpoczęcia rezerwacji
* RentalEndDate (datetime) – data i godzina zakończenia rezerwacji

## Operacje na danych

 Dodawanie, edycja i usuwanie samochodów i rezerwacji

 Filtrowanie i wyszukiwanie dostępnych samochodów (np. po marce, modelu, dostępności)

 Sprawdzanie dostępności samochodu w zadanym terminie

 Walidacja terminów rezerwacji: data rozpoczęcia musi być wcześniejsza niż data zakończenia

 Zapobieganie konfliktom – brak możliwości rezerwacji samochodu w okresie pokrywającym się z inną rezerwacją

 Obsługa statusów rezerwacji (np. zakończona, aktywna, anulowana)

## Reguły poprawności

**Poprawność danych**

* Numer VIN (VinNumber) musi być unikalny dla każdego samochodu
* Pola wymagane (np. Brand, Model, VinNumber, CustomerName, daty rezerwacji) nie mogą być puste
* Data rozpoczęcia rezerwacji musi być wcześniejsza niż data zakończenia
* Rezerwacje tego samego auta nie mogą się nakładać na siebie w czasie

**Zgodność typów danych**

* Wartości liczbowe (int) dla identyfikatorów, przebiegu, roku
* Wartości tekstowe (varchar) dla nazw, modeli i numerów VIN
* Daty (datetime) dla terminów rezerwacji

**Zabezpieczenia danych**

* Dostęp do operacji administracyjnych (np. usuwanie danych) tylko dla uprawnionych użytkowników
* Weryfikacja danych wejściowych i zapobieganie wprowadzeniu błędnych lub niezgodnych informacji
* Możliwość uwierzytelniania i autoryzacji użytkowników korzystających z systemu