# **Dokumentacja systemu wypożyczalni**

## **Autorzy**

* Mateusz Pawlak
* Michał Pieczyński

## **1. Instrukcja uruchomienia**

### **Backend (ASP.NET Core)**

1. **Zainstaluj wymagane oprogramowanie:**
   * .NET SDK (wersja zgodna z projektem)
   * SQL Server lub SQLite (w zależności od konfiguracji bazy danych)
2. **Skonfiguruj bazę danych:**
   * Sprawdź plik appsettings.json, czy baza danych jest poprawnie skonfigurowana.
   * Jeśli korzystasz z Entity Framework, uruchom migracje:  
      dotnet ef database update
3. **Uruchom backend:**
   * Przejdź do katalogu głównego backendu i uruchom polecenie:  
      dotnet run
   * Serwer będzie działał domyślnie na http://localhost:5000 lub https://localhost:5001.

### **Frontend (React.js)**

1. **Zainstaluj Node.js i npm (jeśli nie są zainstalowane).**
2. **Zainstaluj zależności:** cd client

npm install

1. **Uruchom frontend:** npm start
   * Aplikacja frontendowa uruchomi się pod adresem http://localhost:3000.

## **2. Struktura projektu**

### **Architektura MVC**

Projekt backendu jest oparty na wzorcu **Model-View-Controller (MVC)**, który dzieli aplikację na:

* **Model** – reprezentuje dane i logikę biznesową (np. Car.cs, User.cs).
* **View** – w przypadku API nie występuje bezpośrednio (dane zwracane są w formacie JSON).
* **Controller** – przetwarza żądania i zwraca odpowiednie dane (np. CarController.cs, AuthController.cs).

### **Backend**

* Program.cs – Główna klasa uruchamiająca aplikację.
* AppDbContext.cs – Konfiguracja bazy danych i kontekst Entity Framework.
* Car.cs, Client.cs, Review.cs, User.cs – Modele encji w bazie danych.
* CarController.cs, ClientsController.cs, ReviewController.cs, AuthController.cs – Kontrolery obsługujące żądania API.
* CarService.cs, ClientService.cs, ReviewService.cs – Logika biznesowa i operacje na bazie danych.
* Migrations/ – Pliki migracji bazy danych.
* wwwroot/ – Zasoby statyczne.

### **Frontend**

* App.js – Główna struktura aplikacji React.
* LoginForm.js, RegisterForm.js – Komponenty do logowania i rejestracji użytkowników.
* AddCarForm.js, AddReviewForm.js – Formularze do dodawania samochodów i recenzji.
* package.json – Konfiguracja zależności React.
* public/ – Pliki statyczne (np. ikony, favicon).

## **3. Opis systemu użytkowników**

### **Role użytkowników:**

* **Administrator** – Może zarządzać użytkownikami, samochodami, recenzjami.
* **Zarejestrowany użytkownik** – Może wypożyczać samochody, dodawać recenzje.
* **Niezalogowany użytkownik** – Może przeglądać oferty, ale nie może dokonywać rezerwacji.

### **Komponenty systemu użytkowników:**

* **Rejestracja i logowanie** (AuthController.cs)
  + Użytkownik może się zarejestrować, logować i otrzymywać token JWT.
* **Zarządzanie samochodami** (CarController.cs)
  + Administrator może dodawać, edytować i usuwać samochody.
  + Użytkownicy mogą przeglądać samochody.
* **Zarządzanie recenzjami** (ReviewController.cs)
  + Użytkownicy mogą dodawać recenzje tylko do wypożyczonych samochodów.

System posiada podstawowe mechanizmy autoryzacji oparte na tokenach JWT.