Dokumentacja techniczna

Serwis Samochodowy Online

Autorzy:

Antoni Dramiński – 140470
Jakub Choma – 140465

1. **Krótki opis działania projektu**

   Serwis Samochodowy Online to aplikacja internetowa umożliwiająca obsługę serwisu samochodowego. Użytkownicy będą mogli dokonywać rezerwacji napraw swoich pojazdów w wybranym przez siebie warsztacie. Ponadto z profilu użytkownika będzie można zarządzać umówionymi naprawami, zobaczyć historię wizyt i napraw konkretnego pojazdu. Z poziomu warsztatu będzie można podejrzeć umówione wizyty oraz zobaczyć historię klienta wraz z umówionym autem.

2. **Specyfikacja wykorzystanych technologii**

   - ASP.NET 8
   - Entity Framework Core
   - LocalDB

3. **Instrukcje pierwszego uruchomienia projektu**

   W Konsoli Menedżera Pakietów (w Visual Studio) wykonaj komendę:
   - Update-Database

4. **Opis struktury projektu**
   Projekt Serwis Samochodowy Online oparty jest na architekturze ASP.NET Core MVC, co oznacza podział na trzy główne warstwy: Model, Widok i Kontroler.

   Controllers/
   - **HomeController.cs –** Obsługuje stronę główną.
   - **UserManagerController.cs –** Zarządza użytkownikami systemu (tylko dla administratorów), umożliwia usuwanie i edytowanie
   - **UserServiceHistoryController.cs –** Wyświetla historię serwisowania pojazdów dla zalogowanego użytkownika, umożliwia usuwanie
     ServiceController.cs – Wyświetla dostępne serwisy samochodowe i umożliwia ich wybór oraz rezerwację usług
   - **ServiceRequestsController.cs –** Obsługuje zgłoszenia serwisowe dla managerów i administratorów, umożliwia zarządzanie ich statusem oraz usuwanie
   - **AppointmentController.cs –** Zarządza rezerwacjami usług serwisowych, umożliwia wybór serwisu i pojazdu
   - **CarManagerController.cs –** Administratorzy i menedżerowie mogą zarządzać pojazdami użytkowników (edycja, usuwanie)
   - **CarController.cs –** Obsługuje dodawanie, edycję i usuwanie pojazdów przez użytkowników

   Models/
   - **AppointmentModel** – Model rezerwacji serwisu samochodowego, zawiera ID użytkownika, pojazdu, usługi, datę wizyty oraz status wykonania
   - **ServiceRequestsViewModel** – Widok zgłoszeń serwisowych, zawiera szczegóły pojazdu, termin wizyty, opis problemu i status naprawy
   - **CarModel** – Model pojazdu, zawiera dane o samochodzie (VIN, marka, model, rocznik, silnik) powiązane z użytkownikiem
   - **AppointmentViewModel** – Widok rezerwacji, łączy użytkownika, samochód i usługę serwisową
   - **ErrorViewModel** – Model obsługi błędów, przechowuje identyfikator błędu
   - **UserManagerModel** – Model zarządzania użytkownikami, przechowuje ID, e-mail, nazwę użytkownika i role
   - **ServiceModel** – Model warsztatu, przechowuje dane o serwisie (nazwa, adres, opis, zdjęcie)
   - **UserServiceHistoryViewModel** – Widok historii usług użytkownika, zawiera szczegóły napraw i terminów
   - **CarManagerModel** – Rozszerzona wersja modelu pojazdu, zarządza informacjami o samochodach

Views/

- **Car –** Widoki związane z zarządzaniem pojazdami użytkownika (lista, dodawanie, edycja, usuwanie).
- **CarManager –** Widoki dla administratorów i menedżerów do zarządzania pojazdami użytkowników.
- **Home –** Widoki dla strony głównej oraz polityki prywatności.
- **Service –** Widoki wyświetlające listę serwisów oraz formularze rezerwacji usług serwisowych.
- **ServiceRequests –** Widoki do zarządzania zgłoszeniami serwisowymi (lista, zmiana statusu, usuwanie).
- **Shared –** Wspólne widoki używane w całej aplikacji (np. layout, nawigacja, stopka).
- **UserManager –** Widoki do zarządzania użytkownikami (lista użytkowników, edycja ról, usuwanie).
- **UserServiceHistory –** Widoki wyświetlające historię serwisowania pojazdów dla użytkownika.

5. Wykorzystane modele

- Rezerwacje serwisu (AppointmentModel, AppointmentViewModel)
- Zgłoszenia i historia napraw (ServiceRequestsViewModel, UserServiceHistoryViewModel)
- Pojazdy użytkowników (CarModel, CarManagerModel)
- Zarządzanie użytkownikami (UserManagerModel)
- Warsztaty i usługi (ServiceModel)
- Obsługa błędów (ErrorViewModel)

6. Wykorzystane kontrolery wraz z metodami

HomeController

- Index() – Strona główna
- Privacy() – Polityka prywatności
- Error() – Obsługa błędów

UserManagerController (Admin)

- Index() – Lista użytkowników
- Delete() – Usuwanie użytkownika
- DeleteConfirmed() – Potwierdzenie usunięcia
- AddRole() – Dodanie roli
- RemoveRole() – Usunięcie roli

UserServiceHistoryController (Użytkownik)

- Index() – Historia serwisowa
- Delete() – Usunięcie wizyty
- DeleteConfirmed() – Potwierdzenie usunięcia

ServiceController

- Index() – Lista serwisów

- SelectService() – Wybór serwisu
- Book() – Formularz rezerwacji
- BookPost() – Przetworzenie rezerwacji

ServiceRequestsController (Manager/Admin)
- Index() – Lista zgłoszeń
- UpdateServiceStatus() – Zmiana statusu naprawy
- Delete() – Usunięcie zgłoszenia

AppointmentController (Użytkownik)
- Book() – Rezerwacja usługi

CarManagerController (Admin/Manager)
- Index() – Lista pojazdów
- Edit() – Edycja pojazdu
- Delete() – Usunięcie pojazdu
- DeleteConfirmed() – Potwierdzenie usunięcia

CarController (Użytkownik)
- Index() – Lista pojazdów
- Create() – Dodanie pojazdu
- Edit() – Edycja pojazdu
- Delete() – Usunięcie pojazdu
- DeleteConfirmed() – Potwierdzenie usunięcia

7. **Opis systemu użytkowników**

System obsługuje trzy główne role użytkowników:

- Klient (Member) – Może rezerwować usługi serwisowe, dodawać swoje pojazdy, przeglądać historię napraw.
- Warsztat (Manager) – Zarządza zgłoszeniami napraw, planuje wizyty, aktualizuje statusy napraw.
- Administrator (Admin) – Ma dostęp do panelu zarządzania użytkownikami i kontroluje dane systemowe.

Domyślnie oraz na potrzeby projektu założone są 3 konta:

- admin@serwis.com (Admin - automatycznie zakładany jeśli go nie ma)
- manager@serwis.com (Manager)
- test@test.pl (Member)

Wszystkie z hasłem: *Admin.02@*

Klient może korzystać z podstawowych zasobów jak z umawianie się do naprawy w jednym z trzech serwisów. Dodawanie swoich aut i zarządzanie (edytowanie i usuwanie) nimi oraz zgłaszanie napraw i zarządzanie (edytowanie i usuwanie) nimi do momentu jak zostaną zatwierdzone przez warsztat lub administratora.

Warsztat ma z kolei dodatkowe uprawnienia poza tymi które posiada klient, może edytować dodane pojazdy wszystkich użytkowników. Realizować zgłoszenia i je zatwierdzać.

Administrator posiada wszystkie powyżej wymienione uprawnienia i dodatkowo może usuwać wszystkie auta, usuwać zgłoszenia napraw oraz nadawać uprawnienia do bycia serwisantem.

Poszczególne strony są zabezpieczone przed nieautoryzowanym dostępem oraz działania usuwające są poprzedzone pytaniem weryfikacyjnym.

8. **Charakterystyka najciekawszych funkcjonalności**

- **Rezerwacja wizyt serwisowych (AppointmentModel, AppointmentViewModel)**
  - Klient wybiera pojazd, usługę i termin, a system automatycznie sprawdza dostępność.
  - Statusy rezerwacji (oczekująca, zaakceptowana, w trakcie, zakończona).

- **Historia napraw (UserServiceHistoryViewModel)**
  - Użytkownik przegląda pełną historię napraw swojego pojazdu.
  - Szczegóły wizyt: data, opis, koszt, filtry i sortowanie.

- **System zgłoszeń serwisowych (ServiceRequestsViewModel)**
  - Klient zgłasza usterkę online, warsztat analizuje i umawia naprawę.
  - Statusy zgłoszenia (nowe, weryfikacja, w naprawie, zakończone).

- **Zarządzanie pojazdami (CarModel, CarManagerModel)**
  - Użytkownik dodaje swoje auta, co ułatwia rezerwacje i śledzenie historii.
  - Dane: VIN, marka, model, rocznik, silnik.

- **Panel administracyjny (UserManagerModel, ServiceModel)**
  - Administrator zarządza użytkownikami, warsztatami i monitoruje zgłoszenia.
  - Możliwość edycji, blokowania kont i nadzorowania systemu.