**Dokumentacja techniczna systemu rezerwacji wizyt**

**Autorzy:**

- Wojciech Brocki (138282)

- Bartosz Nowak (138350)

**Tytuł projektu:**

System rezerwacji wizyt

**1. Przegląd funkcjonalności projektu:**

System rezerwacji wizyt ma na celu usprawnienie procesu rezerwacji terminów, takich jak konsultacje lekarskie. Aplikacja oferuje:

Kluczowe funkcje:

- Widok kalendarza: Użytkownicy mogą przeglądać dostępne oraz zajęte terminy wizyt.

- Rezerwacja wizyt: Umożliwia rezerwację dostępnych terminów.

- Wybór lekarza: Użytkownicy mogą wybrać konkretnego lekarza podczas dokonywania rezerwacji.

- Zarządzanie rezerwacjami: Umożliwia modyfikację lub anulację istniejących rezerwacji.

- Intuicyjny interfejs:

- Dla klientów: Prosty i przejrzysty interfejs do przeglądania oraz rezerwacji terminów.

- Dla lekarzy: Dedykowany interfejs do zarządzania informacjami o pacjentach i wizytach.

- Integracja z kartoteką pacjenta: Automatyczne łączenie istniejących kartotek pacjentów z rezerwacjami, co ułatwia pracę lekarzom.

- Zarządzanie informacjami: Pozwala lekarzom na dodawanie notatek lub szczegółów do kartotek pacjentów podczas lub po wizycie.

**2. Stos technologiczny:**

Język programowania i framework:

- C# z .NET 8: Wykorzystuje najnowszą wersję platformy .NET do budowy skalowalnych i wydajnych aplikacji.

Baza danych:

- Microsoft SQL Server (MS SQL): Niezawodna i efektywna baza danych relacyjnych do zarządzania danymi użytkowników, wizyt i pacjentów.

**3.Instrukcje pierwszego uruchomienia:**

Użycie instrukcji Update-Database w konsoli menedżera pakietów NuGet

**4.Struktura projektu**

Controllers:

-AppointmentController.cs

-DoctorsController.cs

-PatientsController.cs

-HomeController.cs

Models:

-Appointment.cs

-Doctor.cs

-Patient.cs

-ErrorViewModel.cs

Views:

-Appointments

-Create.cshtml

-Delete.cshtml

-Details.cshtml

-Edit.cshtml

-Index.cshtml

-Doctors

-Create.cshtml

-Delete.cshtml

-Details.cshtml

-Edit.cshtml

-Index.cshtml

-Home

-Create.cshtml

-Delete.cshtml

-Details.cshtml

-Edit.cshtml

-Index.cshtml

-Patients

-Create.cshtml

-Delete.cshtml

-Details.cshtml

-Edit.cshtml

-Index.cshtml

-Shared

-\_viewImports.cshtml

-\_viewStart.cshtml

appsettings.json

Data:

-Migrations

-ApplicationDbContext.cs

Program.cs

**5.Modele:**

5.1 Appointment.cs - Reprezentuje pojedynczą wizytę, w tym datę wizyty, jej stan rezerwacji oraz powiązania z pacjentem i lekarzem.

| Pole | Typ danych | Opis | Ograniczenia/Walidacje |
| --- | --- | --- | --- |
| Id | int | Identyfikator wizyty | Klucz główny, autoinkrementacja |
| Date | DateTime | Data wizyty i godzina | Wymagane, |
| IsBooked | bool | Status wizyty (zarezerwowana czy nie) | Domyślnie false |
| PatientId | int | Identyfikator pacjenta. | Wymagane, klucz obcy do tabeli Patient. |
| DoctorId | int | Identyfikator lekarza | Wymagane, klucz obcy do tabeli Doctor |
| Patient | Patient | Obiekt reprezentujący pacjenta przypisanego do wizyty (relacja). | - |
| Doctor | Doctor | Obiekt reprezentujący lekarza przypisanego do wizyty(relacja) | - |

5.2 Patient.cs - Reprezentuje pacjenta w systemie, w tym dane osobowe oraz powiązanie z kontem użytkownika (IdentityUser).

| Pole | Typ danych | Opis | Ograniczenie/Walidacje |
| --- | --- | --- | --- |
| Id | int | Unikalny identyfikator pacjenta | Klucz główny |
| Name | string | Imię pacjenta | Maksymalnie 30 znaków, tylko litery, wymagane |
| Surname | string | Nazwisko pacjenta | Maksymalnie 30 znaków, tylko litery, wymagane |
| UserId | string? | Identyfikator powiązanego użytkownika w tabeli użytkowników(IdntityUser) | Opcjonalne |
| User | IdentityUser | Obiekt reprezentujący użytkownika przypisanego do pacjenta (relacja) | Opcjonalne |

5.3 Doctor.cs - Reprezentuje lekarza w systemie, w tym jego dane osobowe.

| Pole | Typ danych | Opis | Ograniczenie/Walidacje |
| --- | --- | --- | --- |
| Id | int | Identyfikator lekarza | Klucz główny, autoinkrementacja |
| Name | string | Imię lekarza | Maksymalnie 30 znaków, tylko litery, wymagane |
| Surname | string | Nazwisko lekarza | Maksymalnie 30 znaków, tylko litery, wymagane |

**6.Kontrolery**

6.1 AppointmentsController - Zarządzanie wizytami (Appointment) w systemie. Obsługuje operacje CRUD oraz relacje z modelami Doctor i Patient.

| Akcja | Metoda HTTP | Ścieżka | Opis |
| --- | --- | --- | --- |
| Index | GET | /Appointments | Pobiera listę wszystkich wizyt i ich szczegóły (relacje z lekarzami i pacjentami). |
| Details | GET | /Appointments/Details/{id} | Pobiera szczegóły konkretnej wizyty na podstawie identyfikatora. |
| Create | GET | /Appointments/Create | Renderuje formularz tworzenia nowej wizyty. |
| Create | POST | /Appointments/Create | Tworzy nową wizytę na podstawie przesłanych danych formularza. |
| Edit | GET | /Appointments/Edit/{id} | Renderuje formularz edycji istniejącej wizyty. |
| Edit | POST | /Appointments/Edit/{id} | Aktualizuje dane istniejącej wizyty. |
| Delete | GET | /Appointments/Delete/{id} | Wyświetla szczegóły wizyty przed potwierdzeniem usunięcia. |
| DeleteConfirmed | POST | /Appointments/Delete/{id} | Usuwa wizytę na podstawie identyfikatora. |

6.2 DoctorsController - Zarządzanie informacjami o lekarzach (Doctor) w systemie. Obsługuje podstawowe operacje CRUD.

| Akcja | Metoda HTTP | Ścieżka | Opis |
| --- | --- | --- | --- |
| Index | GET | /Doctors | Pobiera listę wszystkich lekarzy w systemie. |
| Details | GET | /Doctors/Details/{id} | Pobiera szczegóły konkretnego lekarza. |
| Create | GET | /Doctors/Create | Renderuje formularz tworzenia nowego lekarza. |
| Create | POST | /Doctors/Create | Tworzy nowego lekarza na podstawie przesłanych danych formularza. |
| Edit | GET | /Doctors/Edit/{id} | Renderuje formularz edycji danych lekarza. |
| Edit | POST | /Doctors/Edit/{id} | Aktualizuje dane istniejącego lekarza. |
| Delete | GET | /Doctors/Delete/{id} | Wyświetla szczegóły lekarza przed potwierdzeniem usunięcia. |
| DeleteConfirmed | POST | /Doctors/Delete/{id} | Usuwa lekarza na podstawie identyfikatora. |

6.3 PatinetsController - Zarządzanie informacjami o pacjentach (Patient) w systemie. Operacje CRUD są dostępne dla zalogowanych użytkowników. Dodatkowo kontroler obsługuje relację z IdentityUser.

| Akcja | Metoda HTTP | Ścieżka | Opis |
| --- | --- | --- | --- |
| Index | GET | /Patients | Pobiera listę pacjentów przypisanych do zalogowanego użytkownika. |
| Details | GET | /Patients/Details/{id} | Pobiera szczegóły konkretnego pacjenta. |
| Create | GET | /Patients/Create | Renderuje formularz tworzenia nowego pacjenta. |
| Create | POST | /Patients/Create | Tworzy nowego pacjenta i przypisuje go do zalogowanego użytkownika. |
| Edit | GET | /Patients/Edit/{id} | Renderuje formularz edycji danych pacjenta. |
| Edit | POST | /Patients/Edit/{id} | Aktualizuje dane istniejącego pacjenta. |
| Delete | GET | /Patients/Delete/{id} | Wyświetla szczegóły pacjenta przed potwierdzeniem usunięcia. |
| DeleteConfirmed | POST | /Patients/Delete/{id} | Usuwa pacjenta na podstawie identyfikatora. |

**7.Opis systemu użytkowników**

Użytkownicy zalogowani dzielą się na lekarzy oraz na pacjentów, lekarze mają wgląd do wszystkich umówionych wizyt oraz do danych umówionego pacjenta, pacjenci mają wgląd do własnego kalendarza oraz wszystkich umówionych przez nich wizyt, oraz do dostępnych wizyt, niezalogowani użytkownicy mogą wyłącznie umówić się na wolne terminy nie mają mogą również sprawdzić wszystkich swoich terminów.

**8.Najciekawsze funkcjonalności**

Personalizowane widoki i dostęp do danych pacjenta

W aplikacji System Rezerwacji zrealizowano funkcjonalność logowania użytkowników oraz zapewnienia pełnej prywatności danych pacjentów. Funkcjonalność ta pozwala na personalizację doświadczenia użytkownika poprzez przypisanie pacjentów do ich indywidualnych kont. Dzięki temu pacjent może przeglądać i zarządzać tylko swoimi danymi, a nie danymi innych użytkowników.

Kluczowe cechy:

Logowanie użytkowników – Zrealizowano pełny proces logowania użytkowników za pomocą wbudowanego mechanizmu autoryzacji ASP.NET Core. Każdy użytkownik (pacjent) musi się zalogować, aby uzyskać dostęp do swoich danych, co zapewnia bezpieczeństwo i ochronę prywatności.

Ograniczony dostęp do danych – Po zalogowaniu, pacjent ma dostęp tylko do swoich wpisów. Mechanizm filtracji w kontrolerze PatientsController zapewnia, że dane wyświetlane na stronie Patients są przypisane wyłącznie do zalogowanego użytkownika, dzięki czemu pacjent nie ma dostępu do danych innych pacjentów.

Indywidualne widoki pacjentów – W widokach, takich jak lista pacjentów, edycja czy usuwanie danych, system wyświetla tylko te informacje, które są związane z kontem użytkownika, a także uniemożliwia nieautoryzowany dostęp do funkcji takich jak edycja czy usuwanie danych pacjentów, którzy nie są właścicielami danego konta.