

Politechnika Wrocławska
Wydział Elektroniki

Projektowanie telemedycznych systemów internetowych i mobilnych

SYSTEM UŁATWIAJĄCY REJESTRACJĘ W PRZYCHODNI

Autorzy:

BARTOSZ POWĘSKA, 234720

MATEUSZ LEŚ, 234936

ROBERT GMERSKI, 235757

Wtorek, 11¹⁵ TP

Dr inż. Mariusz Topolski

Semestr zimowy 2019

Spis treści

1	Wstęp	2
2	Technologia realizacji	2
3	Harmonogram realizacji	2
4	Backend	3
5	Strona internetowa	4
6	Aplikacja mobilna	6

1 Wstęp

Nasz projekt będzie miał na celu ułatwienie rejestracji w przychodniach internistycznych, gdzie nie wymagane są żadne informacje. Wymagana jest zaś szybkość działania, gdyż każdy wie, jak wygląda sprawa rejestrowania się w takowych przychodniach. Dzięki naszej aplikacji pacjent będzie w stanie zarejestrować się sam, sprawdzić terminy oraz zmienić/usunąć swoją rezerwację. Otrzyma również informacje, jaki lekarz go będzie badał oraz gdzie. Dzięki takiej aplikacji również nie będzie problemu ze sprawdzeniem rejestracji, gdyż będzie dla pacjenta ona widoczna ciągle.

Z drugiej strony będzie to ułatwieniem w prowadzeniu takiej przychodni. Korzystając z przeglądarki pracownicy przychodni będą mogli dodać terminy pasujące danym lekarzom. Te terminy będą widoczne dla pacjentów korzystających z aplikacji mobilnej.

Dodatkowymi funkcjami będą podstawowe działania bazodanowe na rejestracjach oraz lekarzach. Każdy lekarz będzie miał pokazany swoją dostępność. Ułatwi to pracę również i lekarzom, którzy dzięki uporządkowaniu nie będą musieli spędzać tyle czasu przy dokumentach, wystarczy będzie podać numer rezerwacji a pacjent, godzina i lekarz będą od razu podane.

Sama baza danych by się składała z tabel Lekarze, Pacjenci oraz powiązana z poprzednimi tabelą Rezerwacje. Po minięciu godziny rezerwacji, na którą by się pacjent nie zjawił to zostanie ona usunięta z tabeli, lecz i tak trafi to tabeli z historią rezerwacji.

2 Technologia realizacji

- Xamarin
- JavaScript
- Django
- MySQL

3 Harmonogram realizacji

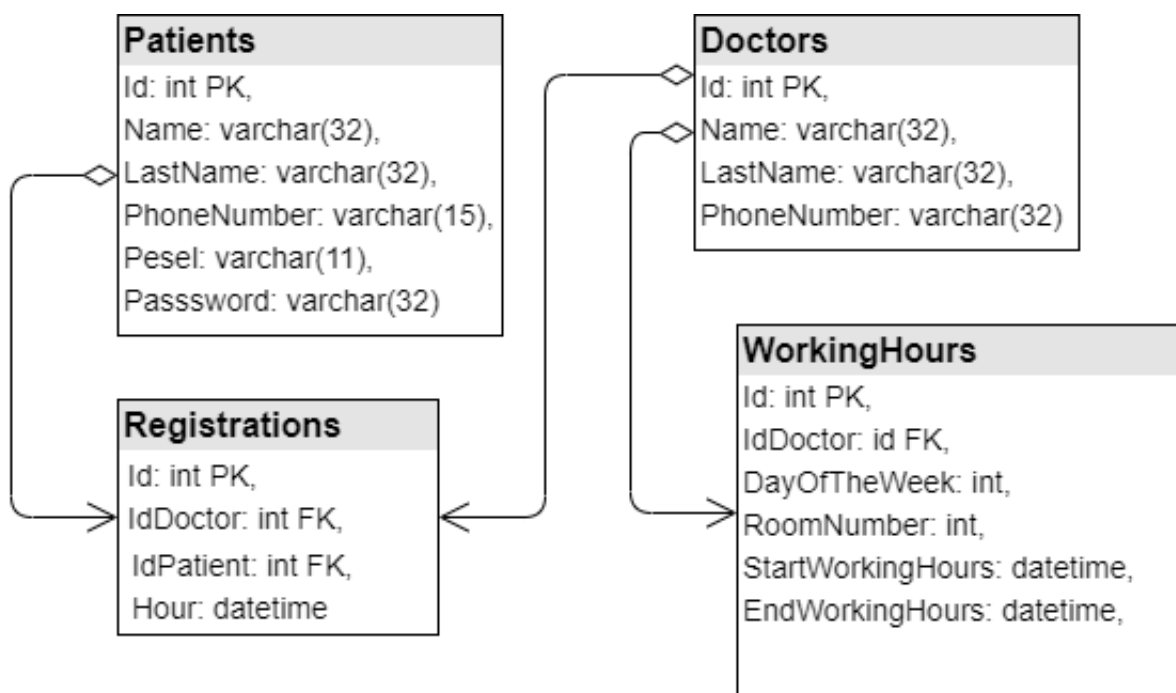
- Wybór tematu projektu (29.10.19)
- Architektura informacji i projekt interfejsu użytkownika (12.11.19)
 - Stworzenie schematu bazy danych oraz prototypu aplikacji (4.11.19)
 - Implementacja bazy danych (11.11.19)
- Projekt oprogramowania (26.11.19)
 - Tworzenie aplikacji webowej (18.11.19)
 - Tworzenie aplikacji mobilnej (24.11.19)
- Raport końcowy i prezentacja (10.12.19)
 - Testy aplikacji (04.12.19)
 - Dokumentacja końcowa (09.12.19)

4 Backend

Aplikacja backendowa została napisana w języku C# w środowisku .net core rest api. Do komunikacji z bazą danych został użyty entity framework. API komunikuje się z środowiskiem zewnętrznym za pomocą formatu JSON. Aplikacja składa się z modeli reprezentujących elementy odzwierciedlające pracę przychodni takie jak: pacjenci, lekarze, godziny pracy, rejestracje, oraz kontrolerów zarządzających żadaniami HTTP. Dodatkowo są też klasy reprezentujące wygląd zserializowanych danych wysyłanych oraz przyjmowanych. Baza danych została wygenerowana przy pomocy entity framework na podstawie stworzonych w C# modeli. Klasa ApplicationDbContext zawiera szczegółowe informacje o tabelach i relacjach między encjami. Kontrolery podzielone są na podstawie tego do jakiej tabeli są kierowane żądania. Stąd mamy DoctorsController, PatientsController, RegistrationsController oraz WorkingHoursController. Kontrolery zawierają zestaw funkcji, do każdej funkcji przyporządkowany jest odpowiedni adres url i rodzaj żądania (get lub post). Funkcje te deserializują otrzymywane dane, realizują odpowiednie przekształcenia, wysyłają zapytanie do bazy danych za pomocą entity framework i wysyłają odpowiedź w formacie JSON do klienta.

Aplikacja realizuje takie funkcje jak dodawanie, usuwanie i przegląd lekarzy. Dodawanie pacjenta oraz wyszukiwanie pacjentów w bazie danych przez różne atrybuty. Dodawanie i usuwanie godzin pracy lekarzy oraz wyszukanie tych godzin ze względu na id lekarza.

Baza danych jak i aplikacja są hostowane na platformie Microsoft Azure. Platforma umożliwia prostą administrację i przez pół roku można ją testować za darmo.



Rysunek 1: Diagram ERD bazy danych

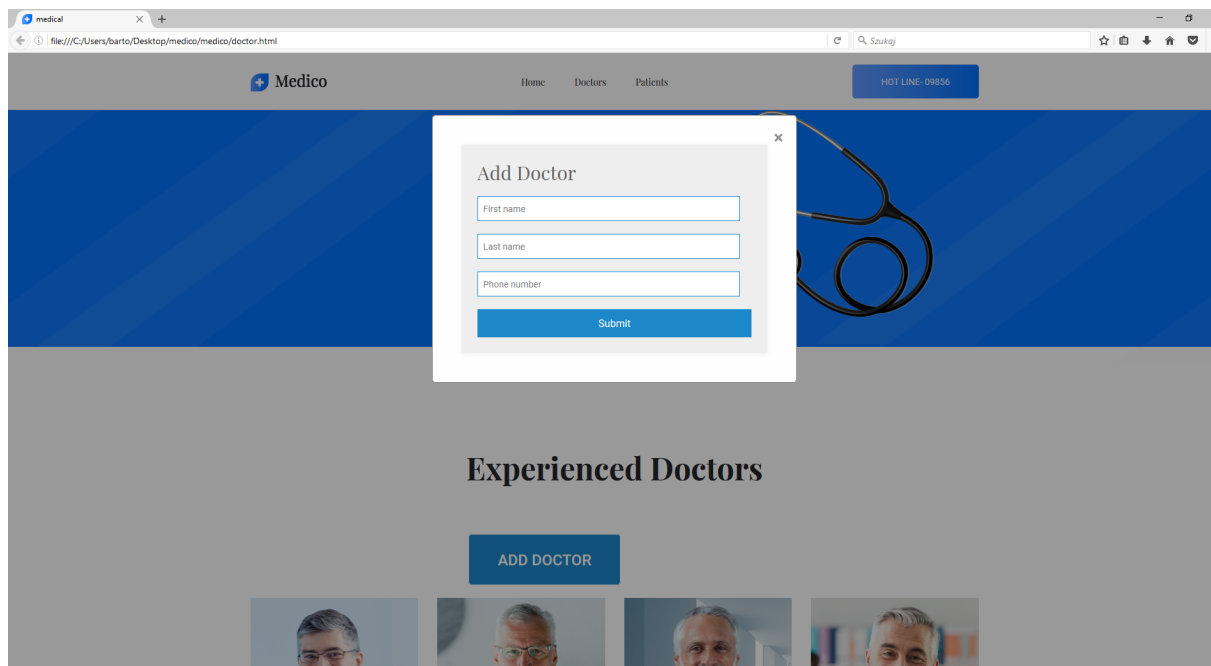
5 Strona internetowa

Strona będąca częścią naszego projektu została napisana przy użyciu htmla, cssa, javascriptu i elementów pythona dzięki użyciu paczek django i jinja2

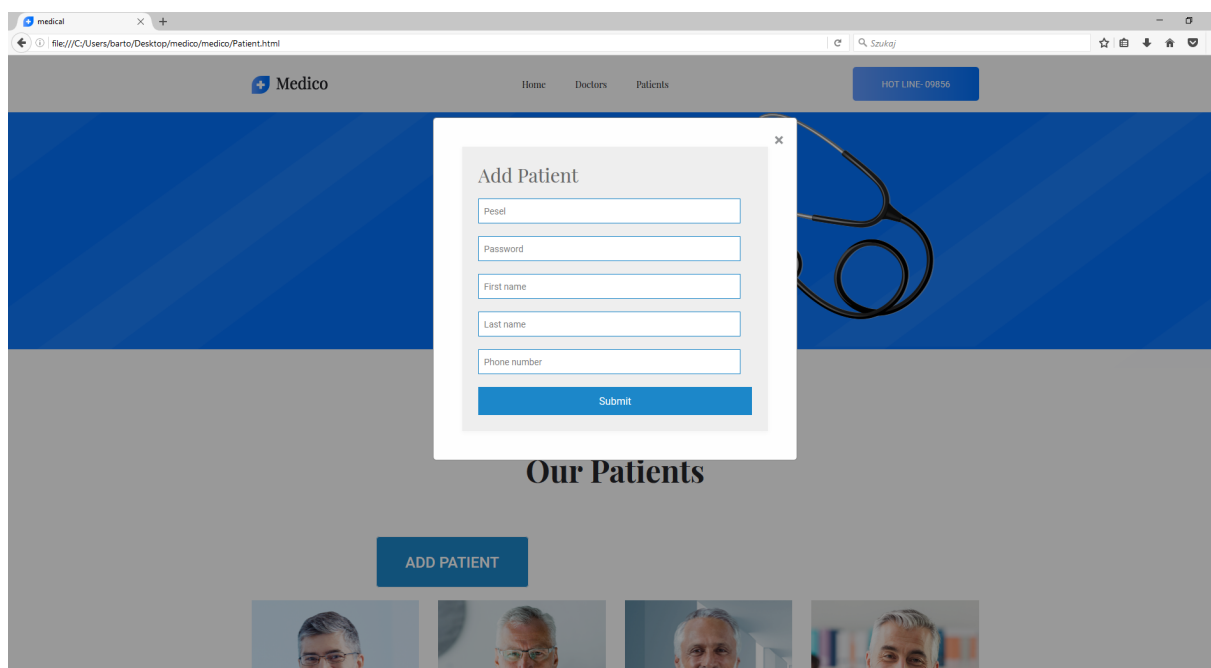
Kolejne strony tworzone są standardowo przez połączenie statycznych elementów htmla wspomaganych javascriptem i cssem, oraz elementów pythona ułatwiających komunikację z api jak i wyświetlanie danych.

Strona pozwala na przegląd wszystkich pacjentów i lekarzy, dodawanie nowych, jak również przeglądanie i edycję godzin pracy lekarzy, oraz przeglądanie wizyt.

Z bazą danych aplikacja łączy się przez REST Api. Po wysłaniu odpowiedniego komunikatu do Api otrzymujemy dane w formacie JSON. Odpowiedź jest zależna od url. Po otrzymaniu odpowiedzi parsujemy dane do odpowiedniego obiektu. Następnie możemy przejść do wyświetlenia odpowiednich parametrów obiektu w odpowiednim miejscu na naszej stronie internetowej.



Rysunek 2: Dodawanie lekarzy



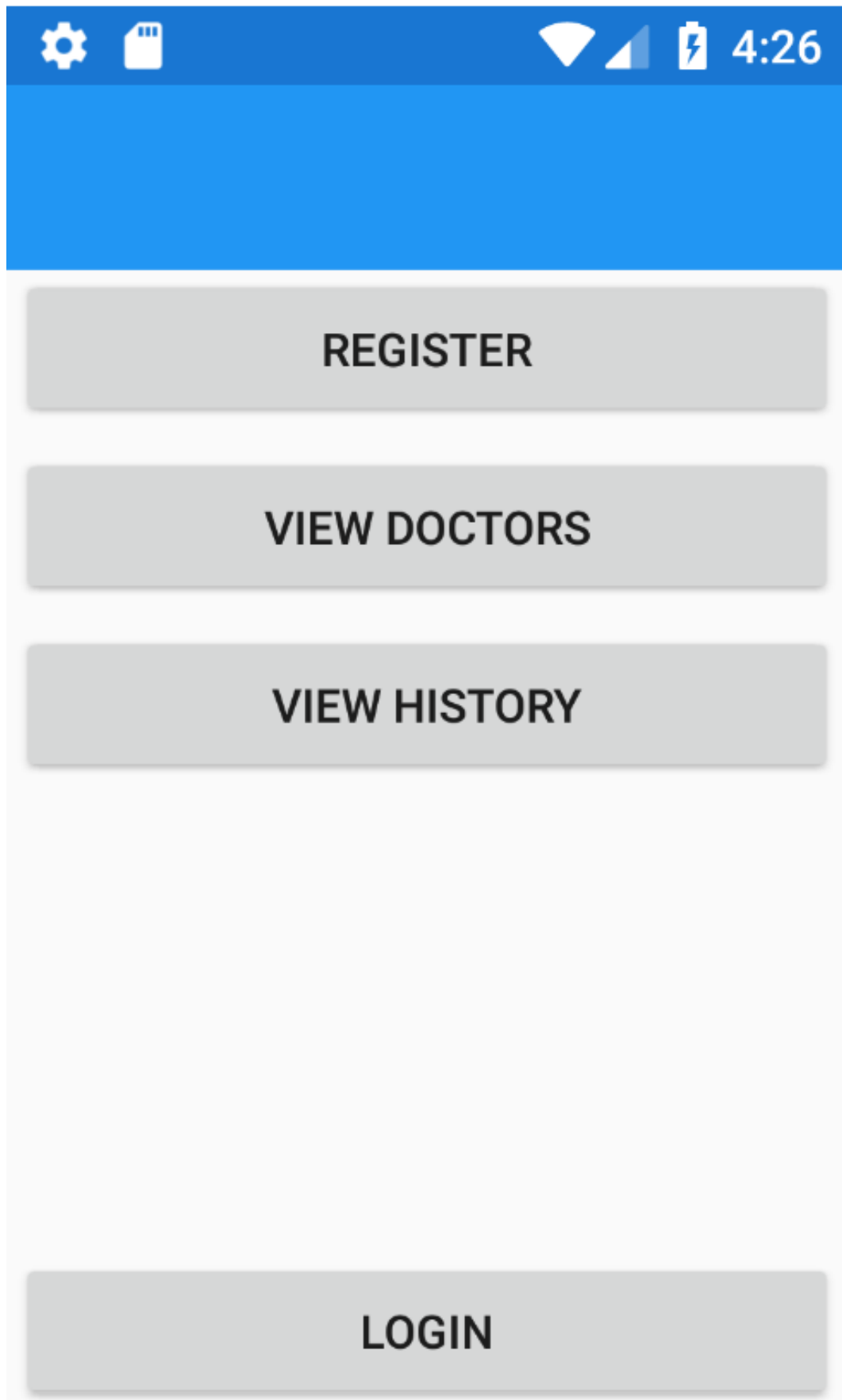
Rysunek 3: Dodawanie pacjentów

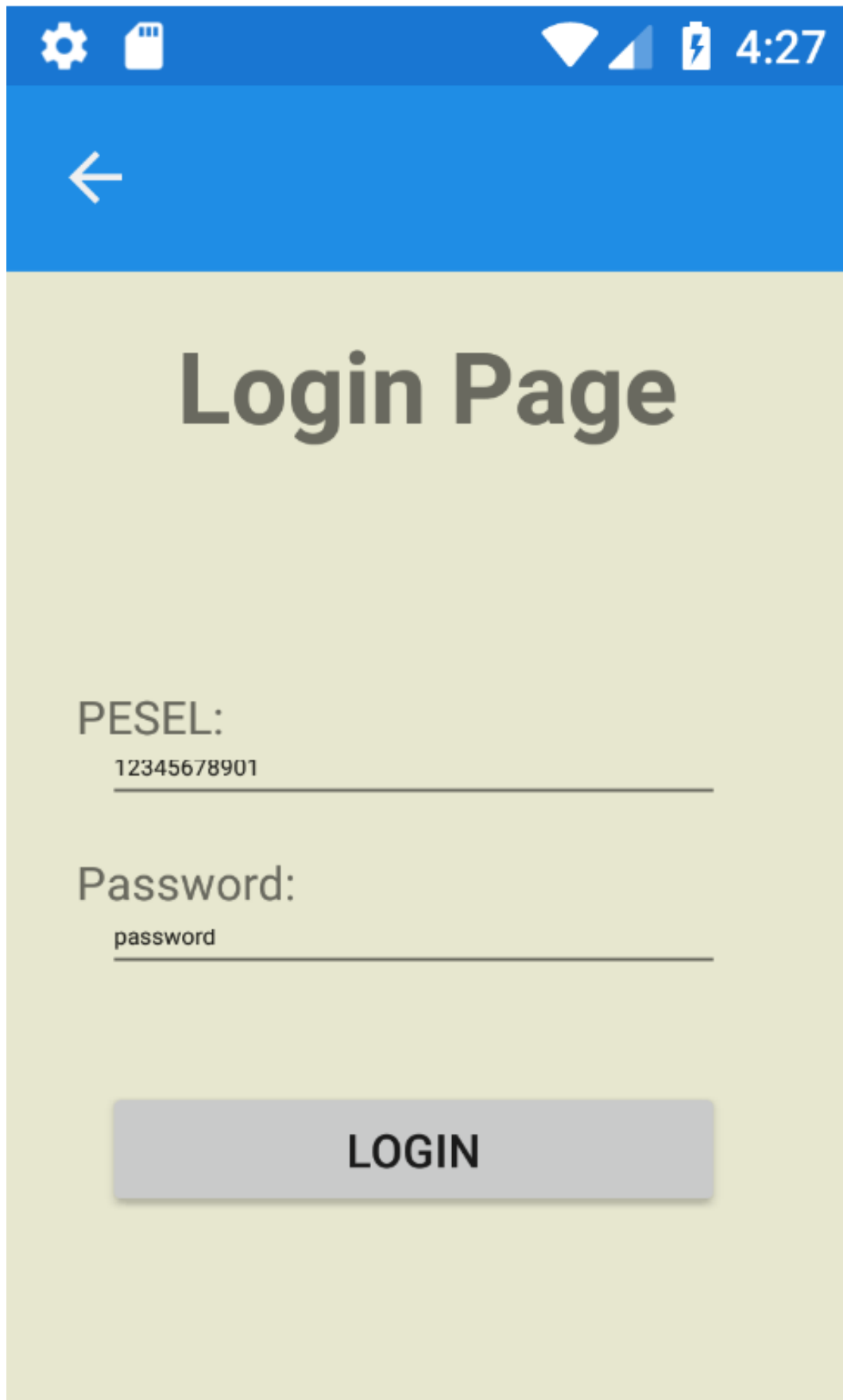
6 Aplikacja mobilna

Aplikacja będąca częścią projektu została napisana w języku C# i środowisku Xamarin. Dodatkowo do aplikacji zostały dołączone odpowiednie paczki NUGET: - Microsoft.AspNet.WebApi.Client (w celu zapewnienia połączenia z API), - Newtonsoft.Json (zapewniająca proste przenoszenie danych w formacie JSON na obiekt klasy C#).

W Xamarinie tworząc aplikację tworzymy widoki (strony). Każda strona to każdy możliwy widok w aplikacji. Strony są napisane w języku XAML. Każda strona posiada swoją klasę, która obsługuje interakcje z użytkownikiem. Tak więc każda strona składa się z dwóch plików. Jeśli jej nazwa to MainPage, to posiadamy pliki MainPage.xaml oraz MainPage.xaml.cs. Najpierw tworzony jest interfejs z pliku xaml, potem (opcjonalnie) funkcja OnAppearing() w pliku .cs. Następnie trzeba zająć się wydarzeniami generowanymi przez elementy widoku, typu Button (przycisk) czy Editor (pole tekstowe edytowalne). Żeby je utworzyć potrzebujemy dodać własność do danego elementu w pliku .xaml. Wygeneruje nam to funkcję w odpowiadającym tej stronie pliku .cs. Niestety, owe elementy są całkiem surowe. Potrzeba doświadczenia, by móc samodzielnie stworzyć elementy, których potrzebujemy mając podstawowe, udostępnione przez środowisko.

Aplikacja pozwala na przegląd wizyt pacjenta, przegląd wszystkich lekarzy oraz ogólną rejestrację. Jest jeszcze jednorazowe logowanie, gdyż często osoby starsze mogą mieć problem z wpisywaniem danych. Rejestracja ogólna pokazuje nam najbliższe możliwe terminy, jest to najlepsza opcja, gdy potrzebna jest szybka lub krótka wizyta u lekarza i nie ma znaczenia, do jakiego lekarza trafimy. Opcja druga jest opcją początkowo pokazującą dane lekarzy dostępnych. Wybierając lekarza dostajemy opcję rejestracji u niego w najbliższych terminach od dzisiaj do końca tygodnia. Z bazą danych aplikacja łączy się przez REST Api. Po wysłaniu odpowiedniego komunikatu do Api otrzymujemy dane w formacie JSON. Odpowiedź jest zależna od url. Po otrzymaniu odpowiedzi zmieniamy ją funkcją `ReadAsStringAsync<T>` na obiekt klasy `T`. Jest możliwość również wyboru, jakie dane chcemy otrzymać. Wybieramy je w definicji klasy `T`. Ważne jest to, aby pola klasy miały taką samą nazwę jak nagłówki otrzymane w odpowiedziach. Przydatną do tego stroną generującą takie klasy jest strona internetowa <http://json2csharp.com>.





Android Emulator - Android_Accelerated_x86_Oreo:5554

Settings, SD Card, Wi-Fi, Signal, Battery, 4:27

←

Login Page

PESEL:

12345678901

Password:

password

LOGIN



Doctor's ID	Name	Last Name
1	Patryk	Wieczorek
2	Andrzej	Nowak
3	Krystyna	Kowalska
4	Karolina	Wieczorek