Specyfikacja techniczna

1. Użyte technologie

Frontend – React

Backend – Rest w ASP.NET Core

1. Planowane usługi/biblioteki dla specyficznych wymagań projektu

Diagramy – UML

Dokumenty - OneDrive

Bazy danych – SSMS, Azure

Autoryzacja i uwierzytelnianie - JWT (JSON Web Tokens)

1. Obsługiwane urządzenia/software użytkowników końcowych

Przeglądarka

Android, iOS

1. Wymagane zasoby

Chmura: Azure SQL Database i Azure App Service

Runtimes i środowiska: .NET Core runtime, Node.js i npm

Narzędzia deweloperskie: Visual Studio i Vuisual Studio Code

CI/CD: Azure DevOps Services

1. Ilość użytkowników

Dla jednej placówki/dziennie

lekarze, pielęgniarki, pomoc medyczna, specjaliści, pacjenci: 50 – 150 użytkowników

1. Planowany ruch

Przyjmując średnio 10 interakcji z systemem na pacjenta: 500 – 1500 interakcji dziennie/placówkę

Specyfikacja funkcjonalna

1. Analiza potrzeb biznesowych:
   * Problem:

Trudności w zarządzaniu rezerwacjami wizyt, trudności w dostępie do danych pacjenta przez personel medyczny, brak zdalnego dostępu dla pacjentów

* + Wymagania:

- Zautomatyzowany system rezerwacji wizyt

- Bezpieczny dostęp do danych pacjenta

- Możliwość zdalnego dostępu dla pacjentów

- Wsparcie dla urządzeń mobilnych

- System raportowania i analizy danych

1. Synteza powstałej analizy:
   * Rejestracja i logowanie:

- Dla personelu medycznego, administracyjnego i pacjentów.

- Wsparcie dla wielu poziomów uprawnień

* + System rezerwacji wizyt

- Kalendarz z widokiem dostępności

- Automatyczne przypomnienia dla pacjentów

- Możliwość rezerwacji przez pacjentów online

* + Elektroniczna dokumentacja medyczna pacjenta:

- Historia chorób, wizyt i recept

- Możliwość dodawania notatek przez lekarzy

- Integracja z systemami laboratoryjnymi i diagnostycznymi

* + Zdalny dostęp dla pacjentów:

- Przeglądanie własnej historii chorób, wizyt, recept

- Rezerwacja i anulowanie wizyt

1. Technologie potrzebne do realizacji wymagań biznesowych:
   * Backend: ASP.NET Core, Entity Framework.
   * Frontend: React
   * Baza danych: Microsoft Azure SQL Database
   * Usługi chmurowe: Azure Cloud Services
   * Mobilne: React Native

Przypadki użycia

1. Rejestracja pacjenta
   * Aktor: Pacjent
   * Opis: Pacjent wypełnia ankietę na urządzeniu mobilnym (tablet), podaje wszystkie wymagane dane, potwierdza, że wszystkie dane są prawdziwe, daje podpis i zatwierdza ankietę
   * Forma realizacji: System powinien udostępnić ankietę, pole na wprowadzenie podpisu, po zatwierdzeniu przesłać dane do bazy i wyświetlić potwierdzenie rejestracji pacjenta
2. Zarządzanie harmonogramem przez lekarza
   * Aktor: Lekarz
   * Opis: Lekarz loguje się do systemu, przegląda swoje zaplanowane wizyty, dodaje dostępny termin oraz może anulować już zaplanowane wizyty
   * Forma realizacji: System powinien umożliwić lekarzowi wgląd w jego harmonogram, dodawanie nowych terminów oraz anulowanie wizyt
3. Zarządzanie harmonogramem prze recepcjonistę
   * Aktor: Recepcjonista
   * Opis: Recepcjonista loguje się do systemu, przegląda zaplanowanych wizyt do wszystkich lekarzy i specjalistów w danej placówce, dodaje nową wizytę, modyfikuje lub anuluje już zaplanowaną wizytę
   * Forma realizacji: System powinien umożliwić recepcjoniście wgląd w zaplanowane wizyty, dostępne terminy oraz dodawanie, modyfikowanie i anulowanie wizyt.
4. Wgląd w dane pacjenta przez lekarza
   * Aktor: Lekarz
   * Opis. Lekarz loguje się do systemu, przegląda zaplanowane wizyty, po wybraniu wizyty ma wgląd w dane pacjenta, historię chorób i recepty
   * Forma realizacji: System powinien umożliwić lekarzowi wgląd w dane pacjenta, historię chorób i recept.
5. Wgląd w dane pacjenta przez pacjenta
   * Aktor: Pacjent
   * Opis: Pacjent loguje się na aplikacji mobilnej, przegląda zaplanowane wizyty, historię chorób i recepty
   * Forma realizacji: System powinien umożliwić pacjentowi zalogowanie się na aplikacji mobilnej na swoje konto, wgląd w zaplanowane wizyty, historię chorób i recepty.

Specyfikacja niefunkcjonalna

1. Wydajność
   * Obsługa przynajmniej 10 000 użytkowników jednocześnie
   * Czas odpowiedzi systemu nie przekracza 2 sekund dla większości operacji
   * Szybkość ładowania strony głównej nie przekracza 3 sekund
2. Dostępność
   * Czas pracy systemu (uptime) wynosi co najmniej 99,9%
   * Możliwość korzystania z systemu 24/7
3. Skalowalność:
   * Możliwość dodawania nowych serwerów lub zasobów w razie wzrostu liczby użytkowników
4. Bezpieczeństwo
   * Szyfrowane połączenie HTTPS
   * Zabezpieczenie przed atakami typu SQL Injection, XSS oraz CSRF
   * Wielopoziomowa weryfikacja tożsamości dla użytkowników
   * Regularne kopie zapasowe danych
5. Niezawodność:
   * System jest w stanie odzyskać się po awarii w ciągu 1 godziny
   * Automatyczne tworzenie kopii zapasowych co najmniej raz dziennie
6. Użyteczność:
   * Intuicyjny interfejs użytkownika
   * Responsywność designu dla urządzeń mobilnych
   * Dokumentacja dla użytkowników i personelu
7. Kompatybilność
   * Działanie na najpopularniejszych przeglądarkach internetowych (Chrome, Firefox, Edge)
   * Wsparcie dla urządzeń mobilnych z systemami Android i iOS

Wycena projektu

1. Czas specyfikacji:
   * Analizy: 20 godzin
   * Research: 16 godzin
   * Przygotowanie oferty: 10 godzin
2. Wireframes/mockups: 20 godzin
3. Produkcja:
   * Programowanie: 400 godzin
   * Zarządzanie projektem: 30 godzin
   * Testy: 30 godzin
   * Wdrożenie: 20 godzin
4. Koszt produkcji:
   * Narzędzia (licencje programistyczne, narzędzia do projektowania):

2 000zł

* + Usługi SaaS (hosting, systemy śledzenia błędów): 1 500zł/rok

1. Czas na komunikację: 20 godzin
2. Oszacowanie kosztów:
   * Analizy, Research, Oferta, Mockups: (20+16+10+20) \* 70zł = 4 620zł
   * Programowanie: 400 \* 70zł = 28 000zł
   * Zarządzanie projektem: 30 \* 70zł = 2 100zł
   * Testy: 30 \* 70zł = 2 100zł
   * Wdrożenie: 20 \* 70zł = 1 400zł
   * Komunikacja: 20 \* 70zł = 1 400zł

Całkowity koszt roboczy: 39 620zł

Łączny oszacowany koszt: 43 120zł