

# System zarządzania przychodnią

Maciej Leśniak

24 stycznia 2025

## 1 Projekt koncepcji i założenia

### 1.1 Temat projektu

Tematem projektu jest system zarządzania przychodnią. Aplikacja została zrealizowana jako aplikacja webowa z wykorzystaniem `Node.js` i `Express.js`, a do przechowywania danych wykorzystano bazę danych PostgreSQL.

### 1.2 Cel projektu

Głównym celem projektu jest usprawnienie zarządzania procesami w przychodni, takimi jak obsługa pacjentów, organizacja wizyt oraz zarządzanie pracą personelu medycznego. System ma zapewniać intuicyjną obsługę dla różnych typów użytkowników oraz przejrzyste raportowanie danych.

### 1.3 Typy użytkowników

System obsługuje trzy typy użytkowników:

#### 1. Administratorzy:

- Mogą dodawać i usuwać użytkowników aplikacji.
- Mają podgląd na listę wszystkich użytkowników.
- Każdy użytkownik może zmieniać swoje dane w profilu.

#### 2. Recepcjonistki:

- Mają dostęp do kalendarza wizyt wszystkich lekarzy, gdzie mogą sprawdzić dostępne terminy.
- Mogą dodawać nowych pacjentów oraz umawiać i odwoływać wizyty.
- Mają podgląd na listę wszystkich pacjentów z możliwością filtrowania oraz usuwania ich.

#### 3. Lekarze:

- Mają dostęp do swojego kalendarza wizyt.
- Mogą odbywać wizyty, podczas których wystawiają recepty.

- Mają dostęp do raportów o pacjentach, gdzie widzą historię wizyt i wystawione recepty.
- Mogą generować statystyki swojej pracy (wizyty i recepty) w wybranym okresie (miesiąc, rok).

## 1.4 Funkcje realizowane przez bazę danych

Baza danych PostgreSQL obsługuje następujące funkcjonalności systemu:

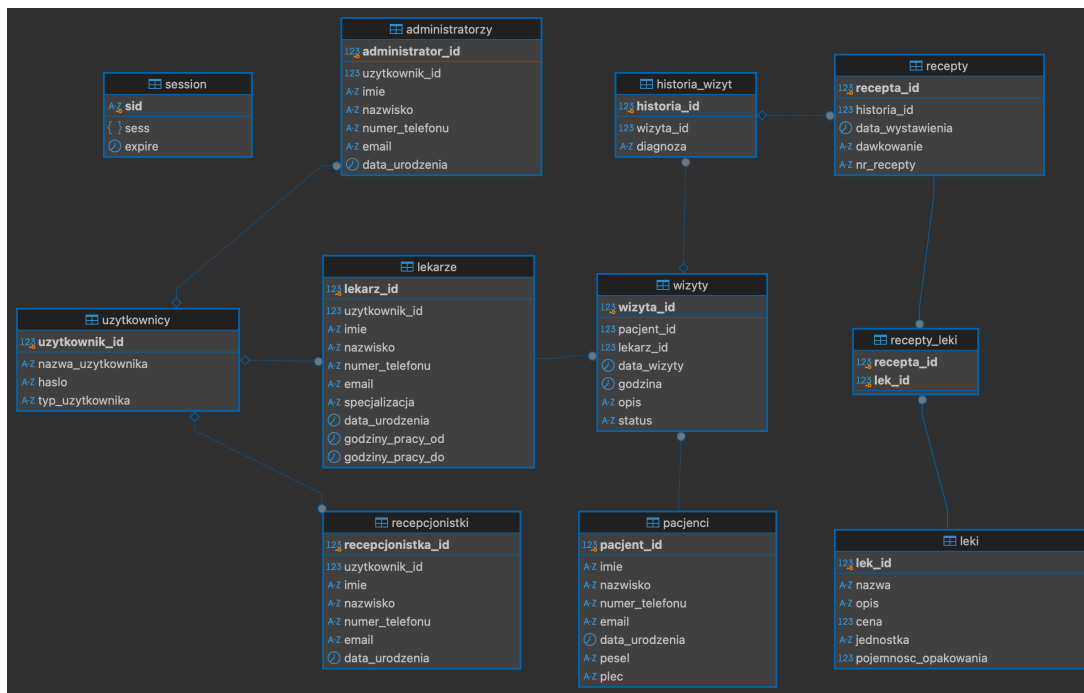
- **Zarządzanie użytkownikami:**
  - Przechowywanie danych użytkowników (administratorzy, recepcjonistki, lekarze) oraz ich ról.
  - Obsługa dodawania, edycji i usuwania użytkowników.
- **Zarządzanie pacjentami:**
  - Przechowywanie danych pacjentów, w tym danych kontaktowych i medycznych.
  - Obsługa wyszukiwania i filtrowania pacjentów.
  - Usuwanie pacjentów.
- **Zarządzanie wizytami:**
  - Planowanie wizyt z uwzględnieniem kalendarza lekarzy.
  - Przechowywanie danych o wizytach, takich jak data, godzina, pacjent, lekarz i opis.
  - Obsługa zmiany statusów wizyt (zaplanowana, odbyta, odwołana).
- **Obsługa recept:**
  - Przechowywanie danych o receptach i powiązanych lekach.
  - Generowanie numerów recept i zapisywanie dawkowania.
- **Raportowanie:**
  - Generowanie statystyk dla lekarzy (wizyty, recepty) w wybranym okresie.
  - Tworzenie raportów historii wizyt dla pacjentów.

System zapewnia integralność danych oraz wspiera walidację poprzez mechanizmy bazy danych, takie jak klucze główne, klucze obce oraz triggerzy do automatycznego sprawdzania poprawności danych.

## 2 Projekt diagramów

### 2.1 Diagram ERD

Diagram ERD (Entity-Relationship Diagram) przedstawia logiczną strukturę bazy danych dla systemu zarządzania przychodnią. Pokazuje encje, ich atrybuty oraz relacje między nimi.



Rysunek 1: Diagram ERD przedstawiający strukturę bazy danych systemu zarządzania przychodnią.

## 3 Projekt logiczny

### 3.1 Struktura bazy danych

Poniżej przedstawiono logiczny projekt bazy danych systemu zarządzania przychodnią. W bazie danych zdefiniowano tabele reprezentujące kluczowe elementy systemu, takie jak użytkownicy, pacjenci, wizyty oraz recepty. W relacjach między tabelami uwzględniono klucze obce, zapewniające integralność danych.

### 3.2 Struktura tabel

Listing 1: Struktura tabel bazy danych

```

1  -- Tabela u ytkownicy
2  CREATE TABLE project.uzytkownicy (
3      uzytkownik_id SERIAL PRIMARY KEY,
4      nazwa_uzytkownika VARCHAR(50) NOT NULL UNIQUE,
5      haslo VARCHAR(255) NOT NULL,
6      typ_uzytkownika VARCHAR(20) NOT NULL CHECK (typ_uzytkownika IN ('
7          lekarz', 'recepjonistka', 'administrator'))
8  );
9  -- Tabela administratorzy
10 CREATE TABLE project.administratorzy (
11     administrator_id SERIAL PRIMARY KEY,
12     uzytkownik_id INT REFERENCES project.uzytkownicy(uzytkownik_id) ON
13         DELETE CASCADE,
14     imie VARCHAR(100) NOT NULL,
15     nazwisko VARCHAR(100) NOT NULL,

```

```

15     numer_telefonu VARCHAR(15),
16     email VARCHAR(100),
17     data_urodzenia DATE
18 );
19
20 -- Tabela lekarze
21 CREATE TABLE project.lekarze (
22     lekarz_id SERIAL PRIMARY KEY,
23     uzytkownik_id INT REFERENCES project.uzytkownicy(uzytkownik_id) ON
        DELETE CASCADE,
24     imie VARCHAR(100) NOT NULL,
25     nazwisko VARCHAR(100) NOT NULL,
26     numer_telefonu VARCHAR(15),
27     email VARCHAR(100),
28     specjalizacja VARCHAR(100),
29     data_urodzenia DATE,
30     godziny_pracy_od TIME,
31     godziny_pracy_do TIME
32 );
33
34 -- Tabela recepcjonistki
35 CREATE TABLE project.recepcjonistki (
36     recepcjonistka_id SERIAL PRIMARY KEY,
37     uzytkownik_id INT REFERENCES project.uzytkownicy(uzytkownik_id) ON
        DELETE CASCADE,
38     imie VARCHAR(100) NOT NULL,
39     nazwisko VARCHAR(100) NOT NULL,
40     numer_telefonu VARCHAR(15),
41     email VARCHAR(100),
42     data_urodzenia DATE
43 );
44
45 -- Tabela pacjenci
46 CREATE TABLE project.pacjenci (
47     pacjent_id SERIAL PRIMARY KEY,
48     imie VARCHAR(100) NOT NULL,
49     nazwisko VARCHAR(100) NOT NULL,
50     numer_telefonu VARCHAR(15),
51     email VARCHAR(100),
52     data_urodzenia DATE,
53     pesel CHAR(11) NOT NULL UNIQUE,
54     plec CHAR(1) NOT NULL CHECK (plec IN ('M', 'K'))
55 );
56
57 -- Tabela wizyty
58 CREATE TABLE project.wizyty (
59     wizyta_id SERIAL PRIMARY KEY,
60     pacjent_id INT REFERENCES project.pacjenci(pacjent_id) ON DELETE
        CASCADE,
61     lekarz_id INT REFERENCES project.lekarze(lekarz_id) ON DELETE
        CASCADE,
62     data_wizyty DATE NOT NULL,
63     godzina TIME NOT NULL,
64     opis TEXT,
65     status VARCHAR(20) DEFAULT 'zaplanowana' NOT NULL CHECK (status IN (
        'zaplanowana', 'odbyta', 'odwo ana')),
66     UNIQUE (pacjent_id, lekarz_id, data_wizyty, godzina)
67 );

```

```

68
69 -- Tabela historia wizyt
70 CREATE TABLE project.historia_wizyt (
71     historia_id SERIAL PRIMARY KEY,
72     wizyta_id INT REFERENCES project.wizyty(wizyta_id) ON DELETE CASCADE
73     ,
74     diagnoza TEXT
75 );
76
77 -- Tabela recepty
78 CREATE TABLE project.recepty (
79     recepta_id SERIAL PRIMARY KEY,
80     historia_id INT REFERENCES project.historia_wizyt(historia_id),
81     data_wystawienia DATE,
82     dawkowanie VARCHAR(255),
83     nr_recepty TEXT UNIQUE
84 );
85
86 -- Tabela recepty-leki (relacja wiele do wielu)
87 CREATE TABLE project.recepty_leki (
88     recepta_id INT REFERENCES project.recepty(recepta_id) ON DELETE
89     CASCADE,
90     lek_id INT REFERENCES project.leki(lek_id) ON DELETE CASCADE,
91     PRIMARY KEY (recepta_id, lek_id)
92 );
93
94 -- Tabela leki
95 CREATE TABLE project.leki (
96     lek_id SERIAL PRIMARY KEY,
97     nazwa VARCHAR(255) NOT NULL,
98     opis TEXT,
99     cena NUMERIC(10, 2),
100     jednostka VARCHAR(50),
101     pojemnosc_opakowania INT
102 );
103
104 -- Tabela sesji (przechowywanie sesji uzytkownikow)
105 CREATE TABLE project.session (
106     sid VARCHAR PRIMARY KEY,
107     sess JSON NOT NULL,
108     expire TIMESTAMP(6) NOT NULL
109 );

```

### 3.3 Opis struktury

- **Tabela *uzytkownicy*:** Przechowuje informacje o uzytkownikach systemu, takich jak administratorzy, recepcjonistki i lekarze. Każdy uzytkownik posiada unikalny identyfikator, nazwę uzytkownika, hasło oraz typ uzytkownika (*administrator*, *recepjonistka* lub *lekarz*).
- **Tabela *administratorzy*:** Zawiera szczególowe informacje o administratorach systemu, takie jak imię, nazwisko, numer telefonu, adres e-mail i data urodzenia. Powiązana jest z tabelą *uzytkownicy* relacją jeden-do-jeden.
- **Tabela *recepjonistki*:** Przechowuje dane recepcjonistek, w tym imię, nazwisko,

numer telefonu, adres e-mail oraz datę urodzenia. Każda recepcjonistka jest powiązana z jednym użytkownikiem w tabeli *uzytkownicy*.

- **Tabela *lekarze*:** Zawiera informacje o lekarzach, takie jak imię, nazwisko, numer telefonu, adres e-mail, specjalizacja, data urodzenia oraz godziny pracy. Każdy lekarz jest powiązany z użytkownikiem w tabeli *uzytkownicy*.
- **Tabela *pacjenci*:** Przechowuje dane osobowe pacjentów, takie jak imię, nazwisko, numer telefonu, adres e-mail, data urodzenia, PESEL oraz płeć. PESEL jest unikalny dla każdego pacjenta, a płeć musi być określona jako *M* (mężczyzna) lub *K* (kobieta).
- **Tabela *wizyty*:** Przechowuje informacje o wizytach pacjentów, łącząc je z lekarzami i pacjentami. Zawiera szczegóły takie jak data wizyty, godzina, opis wizyty oraz status (*zaplanowana*, *odbyta*, *odwołana*).
- **Tabela *historia\_wizyt*:** Rejestruje szczegóły dotyczące odbytych wizyt, w tym diagnozy związane z wizytami. Powiązana jest z tabelą *wizyty*.
- **Tabela *recepty*:** Przechowuje informacje o wystawionych receptach, takie jak data wystawienia, dawkowanie oraz unikalny numer recepty. Powiązana jest z tabelą *historia\_wizyt*.
- **Tabela *recepty\_leki*:** Reprezentuje relację wiele-do-wielu między tabelą *recepty* a tabelą *leki*. Każda recepta może zawierać wiele leków, a dany lek może występować na różnych receptach.
- **Tabela *leki*:** Przechowuje dane o dostępnych lekach, takie jak nazwa, opis, cena, jednostka oraz pojemność opakowania.
- **Tabela *session*:** Przechowuje sesje użytkowników systemu, w tym identyfikator sesji (*sid*), dane sesji (*sess*) w formacie JSON oraz czas wygaśnięcia sesji (*expire*).

### 3.4 Słownik danych

Poniżej przedstawiono szczegółowy słownik danych dla każdej tabeli w systemie.

Tabela	Atrybut	Typ danych	Opis
<i>uzytkownicy</i>	<i>uzytkownik_id</i>	SERIAL	Klucz główny użytkownika.
	<i>nazwa_uzytkownika</i>	VARCHAR(50)	Unikalna nazwa użytkownika.
	<i>haslo</i>	VARCHAR(255)	Hasło użytkownika (zaszyfrowane).
	<i>typ_uzytkownika</i>	VARCHAR(20)	Typ użytkownika ( <i>lekarz</i> , <i>recepjonistka</i> , <i>administrator</i> ).
<i>administratorzy</i>	<i>administrator_id</i>	SERIAL	Klucz główny administratora.

Kontynuacja na następnej stronie

Tabela 1 – kontynuacja ze strony poprzedniej

<b>Tabela</b>	<b>Atrybut</b>	<b>Typ danych</b>	<b>Opis</b>
	<i>uzytkownik_id</i>	INT	Klucz obcy odwołujący się do tabeli <i>uzytkownicy</i> .
	<i>imie</i>	VARCHAR(100)	Imię administratora.
	<i>nazwisko</i>	VARCHAR(100)	Nazwisko administratora.
	<i>numer_telefonu</i>	VARCHAR(15)	Numer telefonu administratora.
	<i>email</i>	VARCHAR(100)	Adres e-mail administratora.
	<i>data_urodzenia</i>	DATE	Data urodzenia administratora.
<i>recepcjonistki</i>	<i>recepcjonistka_id</i>	SERIAL	Klucz główny recepcjonistki.
	<i>uzytkownik_id</i>	INT	Klucz obcy odwołujący się do tabeli <i>uzytkownicy</i> .
	<i>imie</i>	VARCHAR(100)	Imię recepcjonistki.
	<i>nazwisko</i>	VARCHAR(100)	Nazwisko recepcjonistki.
	<i>numer_telefonu</i>	VARCHAR(15)	Numer telefonu recepcjonistki.
	<i>email</i>	VARCHAR(100)	Adres e-mail recepcjonistki.
	<i>data_urodzenia</i>	DATE	Data urodzenia recepcjonistki.
	<i>lekarz_id</i>	SERIAL	Klucz główny lekarza.
	<i>uzytkownik_id</i>	INT	Klucz obcy odwołujący się do tabeli <i>uzytkownicy</i> .
	<i>imie</i>	VARCHAR(100)	Imię lekarza.
	<i>nazwisko</i>	VARCHAR(100)	Nazwisko lekarza.
	<i>numer_telefonu</i>	VARCHAR(15)	Numer telefonu lekarza.
	<i>email</i>	VARCHAR(100)	Adres e-mail lekarza.
	<i>specjalizacja</i>	VARCHAR(100)	Specjalizacja lekarza.
	<i>data_urodzenia</i>	DATE	Data urodzenia lekarza.
	<i>godziny_pracy_od</i>	TIME	Godzina rozpoczęcia pracy lekarza.
	<i>godziny_pracy_do</i>	TIME	Godzina zakończenia pracy lekarza.
<i>pacjenci</i>	<i>pacjent_id</i>	SERIAL	Klucz główny pacjenta.
	<i>imie</i>	VARCHAR(100)	Imię pacjenta.

Kontynuacja na następnej stronie

Tabela 1 – kontynuacja ze strony poprzedniej

Tabela	Atrybut	Typ danych	Opis
	<i>nazwisko</i> <i>numer_telefonu</i>  <i>email</i> <i>data_urodzenia</i>  <i>pesel</i>  <i>plec</i>	VARCHAR(100) VARCHAR(15)  VARCHAR(100) DATE  CHAR(11)  CHAR(1)	Nazwisko pacjenta. Numer telefonu pacjenta. Adres e-mail pacjenta. Data urodzenia pacjenta. PESEL pacjenta, unikalny. Płeć pacjenta ( <i>M</i> lub <i>K</i> ).
<i>wizyty</i>	<i>wizyta_id</i> <i>pacjent_id</i>   <i>lekarz_id</i>   <i>data_wizyty</i> <i>godzina</i> <i>opis</i> <i>status</i>	SERIAL INT   INT   DATE TIME TEXT VARCHAR(20)	Klucz główny wizyty. Klucz obcy odwołujący się do tabeli <i>pacjenci</i> .  Klucz obcy odwołujący się do tabeli <i>lekarze</i> . Data wizyty. Godzina wizyty. Opis wizyty. Status wizyty ( <i>zaplanowana</i> , <i>odbyta</i> , <i>odwołana</i> ).
<i>recepty</i>	<i>recepta_id</i> <i>historia_id</i>   <i>data_wystawienia</i>  <i>dawkowanie</i>  <i>nr_recepty</i>	SERIAL INT   DATE  VARCHAR(255)  TEXT	Klucz główny recepty. Klucz obcy odwołujący się do tabeli <i>historia_wizyt</i> . Data wystawienia recepty. Dawkowanie leków na receptę. Unikalny numer recepty.
<i>leki</i>	<i>lek_id</i> <i>nazwa</i> <i>opis</i> <i>cena</i> <i>jednostka</i>  <i>pojemnosc_opakowania</i>	SERIAL VARCHAR(255) TEXT NUMERIC(10,2) VARCHAR(50)  INT	Klucz główny leku. Nazwa leku. Opis leku. Cena leku. Jednostka dawkowania leku. Pojemność opakowania leku.



### 3.5 Analiza zależności funkcyjnych i normalizacja tabel

W ramach projektu wszystkie tabele zostały przeanalizowane pod kątem zależności funkcyjnych i znormalizowane do trzeciej postaci normalnej (3NF). Poniżej przedstawiono analizę zależności funkcyjnych oraz potwierdzenie spełnienia warunków normalizacji dla każdej tabeli.

#### Tabela *uzytkownicy*

- **Atrybuty:** *uzytkownik\_id*, *nazwa\_uzytkownika*, *haslo*, *typ\_uzytkownika*.
- **Zależności funkcyjne:**
  - *uzytkownik\_id* → *nazwa\_uzytkownika*, *haslo*, *typ\_uzytkownika*.
- **Normalizacja:** Tabela jest już w 3NF, ponieważ:
  - Wszystkie atrybuty są zależne od klucza głównego *uzytkownik\_id*.
  - Brak zależności przechodnich.

#### Tabela *lekarze*

- **Atrybuty:** *lekarz\_id*, *uzytkownik\_id*, *imie*, *nazwisko*, *numer\_telefonu*, *email*, *specjalizacja*, *data\_urodzenia*, *godziny\_pracy\_od*, *godziny\_pracy\_do*
- **Zależności:**
  - *lekarz\_id* → *uzytkownik\_id*, *imie*, *nazwisko*, *numer\_telefonu*, *email*, *specjalizacja*, *data\_urodzenia*, *godziny\_pracy\_od*, *godziny\_pracy\_do*
- **Normalizacja:** Tabela jest w 3NF, ponieważ:
  - Wszystkie atrybuty są zależne od klucza głównego *lekarz\_id*.
  - Brak zależności przechodnich.

#### Tabela *pacjenci*

- **Atrybuty:** *pacjent\_id*, *imie*, *nazwisko*, *numer\_telefonu*, *email*, *data\_urodzenia*, *pesel*, *plec*.
- **Zależności funkcyjne:**
  - *pacjent\_id* → *imie*, *nazwisko*, *numer\_telefonu*, *email*, *data\_urodzenia*, *pesel*, *plec*.
  - *pesel* → *imie*, *nazwisko*, *data\_urodzenia*, *plec*.
- **Normalizacja:** Tabela jest w 3NF, ponieważ:
  - Wszystkie atrybuty są zależne od klucza głównego *pacjent\_id*.
  - *pesel* jest unikalnym atrybutem, co eliminuje redundancję.

### Tabela *wizyty*

- **Atrybuty:** *wizyta\_id*, *pacjent\_id*, *lekarz\_id*, *data\_wizyty*, *godzina*, *opis*, *status*.
- **Zależności funkcyjne:**
  - $wizyta\_id \rightarrow pacjent\_id, lekarz\_id, data\_wizyty, godzina, opis, status$ .
  - $pacjent\_id, data\_wizyty \rightarrow lekarz\_id$ .
- **Normalizacja:** Tabela jest w 3NF, ponieważ:
  - Każdy atrybut jest zależny od klucza głównego *wizyta\_id*.
  - Brak zależności przechodnich.

### Tabela *recepty*

- **Atrybuty:** *recepta\_id*, *historia\_id*, *data\_wystawienia*, *dawkowanie*, *nr\_recepty*.
- **Zależności funkcyjne:**
  - $recepta\_id \rightarrow historia\_id, data\_wystawienia, dawkowanie, nr\_recepty$ .
- **Normalizacja:** Tabela jest w 3NF, ponieważ:
  - Wszystkie atrybuty są zależne od klucza głównego *recepta\_id*.
  - Brak zależności przechodnich.

## 3.6 Podsumowanie normalizacji

Wszystkie tabele zostały przeanalizowane pod kątem zależności funkcyjnych. Wyniki analizy pokazują, że każda tabela spełnia wymagania trzeciej postaci normalnej (3NF). Dzięki temu struktura bazy danych eliminuje redundancję, zapewnia integralność danych oraz umożliwia łatwą ich obsługę.

## 4 Projekt funkcjonalny

Poniżej przedstawiono przykłady korzystania z aplikacji, zilustrowane za pomocą zrzutów ekranu. Każdy krok został opisany w celu łatwiejszego zrozumienia funkcjonalności systemu.

## 4.1 Dodanie użytkownika

Wybierz typ użytkownika:

☐ Administrator ☒ Lekarz ☐ Recepcjonistka

Adres email

Nazwa użytkownika

Hasło

Data urodzenia

Imię

Nazwisko

Numer telefonu

Specjalizacja (tylko dla lekarza)

Godzina rozpoczęcia pracy

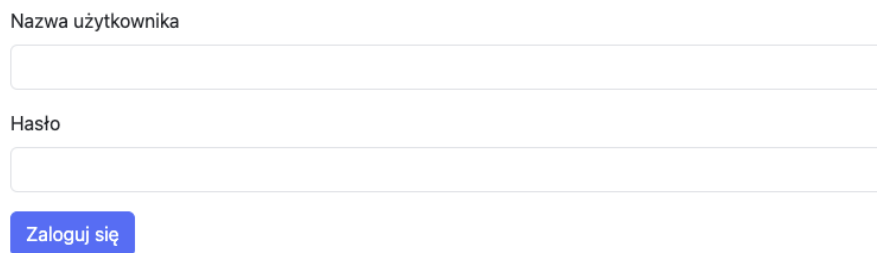
Godzina zakończenia pracy

Stwórz użytkownika

Rysunek 2: Dodanie nowego użytkownika do systemu.

Panel tworzenia użytkownika jest dostępny wyłącznie dla administratorów i pozwala na dodanie nowego użytkownika do systemu. Administrator może wybrać jeden z trzech typów użytkowników: administrator, recepcjonistka lub lekarz. W przypadku wyboru lekarza dostępna jest dodatkowa opcja wprowadzenia specjalizacji. Formularz wymaga również podania takich danych jak adres e-mail, nazwa użytkownika, hasło, data urodzenia, imię, nazwisko, numer telefonu oraz godziny pracy (dla lekarzy). Po wypełnieniu wszystkich pól należy nacisnąć przycisk "Stwórz użytkownika", aby dodać nową osobę do systemu.

## 4.2 Logowanie



Nazwa użytkownika

Hasło

Zaloguj się

Rysunek 3: Ekran logowania do systemu. Użytkownik wprowadza nazwę użytkownika i hasło, aby uzyskać dostęp.

## 4.3 Lista zarejestrowanych użytkowników

**Panel Administratora**

<input type="checkbox"/> Wybierz wszystkie	Imię	Nazwisko	Email	Nazwa użytkownika	Typ użytkownika
<input type="checkbox"/>	Maciej	Lesniak	lesniakmaciek9040@gmail.com	admin1	administrator
<input type="checkbox"/>	Maciej	Lesniak	lesniakmaciek9040@gmail.com	lekarz1	lekarz
<input type="checkbox"/>	Szymon	Paździoch	simon@gmail.com	dr_pazdzioch	lekarz
<input type="checkbox"/>	Janek	Kowalski	jan.kowalski@gmail.com	dr_kowalski	lekarz
<input type="checkbox"/>	Paweł	Jakubowski	elpablito@mex.com	dr_jakubowski	lekarz
<input type="checkbox"/>	Maciej	Lesniak	lesniakmaciek9040@gmail.com	rec1	recepcjonistka

Usuń zaznaczone

Rysunek 4: Przykład widoku listy użytkowników.

Panel administracyjny przedstawia listę wszystkich zarejestrowanych użytkowników systemu, takich jak administratorzy, recepcjonistki oraz lekarze. Funkcjonalność panelu dostępna jest wyłącznie dla administratorów. Lista zawiera szczegóły każdego użytkownika, w tym imię, nazwisko, adres e-mail, nazwę użytkownika oraz typ konta. Administrator ma możliwość zaznaczania użytkowników i usuwania przypadkowo dodanych kont, takich jak recepcjonistki lub inni administratorzy, z wyjątkiem swojego własnego konta. Operacja usuwania wymaga uprzedniego zaznaczenia wybranych użytkowników i potwierdzenia działania.

## 4.4 Rejestracja pacjenta

### Rejestracja Pacjenta

Imię

Nazwisko

PESEL

Data urodzenia



Płeć

☐ Mężczyzna ☐ Kobieta

Numer telefonu

Adres e-mail

Zarejestruj pacjenta

Rysunek 5: Formularz dodania nowego pacjenta do systemu.

Recepcjonistka ma dostęp do panelu rejestracji pacjentów, który umożliwia dodanie nowej osoby do systemu. Formularz wymaga wypełnienia takich danych, jak: imię, nazwisko, numer PESEL, data urodzenia, płeć, numer telefonu oraz adres e-mail. Wszystkie pola są weryfikowane w podstawowy sposób, aby zapewnić poprawność wprowadzanych danych. Dodatkowo numer PESEL jest weryfikowany za pomocą dedykowanej funkcji w bazie danych, która sprawdza jego poprawność zgodnie z oficjalnym algorytmem dostępnym na stronie rządowej. Po wypełnieniu formularza i kliknięciu przycisku „Zarejestruj pacjenta” dane zostają zapisane w systemie.

## 4.5 Lista zarejestrowanych pacjentów

**Lista Pacjentów**

Wyszukaj...

Płeć: ☒ Wszystkie ☐ Mężczyźni ☐ Kobiety

Sortowanie: ☒ Brak sortowania ☐ Imię ☐ Nazwisko ☐ Data Urodzenia

Imię	Nazwisko	Numer telefonu	Email	Data Urodzenia	PESEL	Płeć	Usun
Kornel	Zacharczuk	798863859	adajozwik@glusko.com	08.08.1988	80880887818	M	Usun
Inga	Koj	+48320874561	siewieradagmara@fpuh.net	17.12.1950	50121734125	K	Usun
Tobiasz	Maciuszek	505520779	grabarasebastian@yahoo.com	22.08.1967	67082212345	M	Usun
Anita	Engel	+48509561963	jozefwochnik@stowarzyszenie.com	27.05.1965	65052765432	K	Usun
Damian	Zacharczuk	518763957	damian.zach@gmail.com	14.02.1981	81021463418	M	Usun
Ewa	Koj	+48220547856	ewa.koj@example.com	19.04.1973	73041998214	K	Usun
Grzegorz	Maciuszek	602354987	grzegorz.maciuszek@yahoo.com	11.11.1985	85111137429	M	Usun
Katarzyna	Engel	502384652	katarzyna.engel@gmail.com	22.07.1993	93072278915	K	Usun
Radosław	Warych	607438562	radek.warych@example.com	30.06.1990	90063098213	M	Usun
Ewelina	Koj	508765349	ewelina.koj@example.com	24.12.1988	88122456234	K	Usun

1 2 3

Rysunek 6: Przykład widoku listy pacjentów.

Widok listy zarejestrowanych pacjentów jest dostępny dla recepcjonistki i pozwala na wygodne zarządzanie danymi pacjentów. Lista prezentuje takie informacje jak: imię, nazwisko, numer telefonu, adres e-mail, data urodzenia, PESEL oraz płeć pacjenta.

Dostępne funkcjonalności:

- **Filtrowanie po płci:** Użytkownik może wyświetlić pacjentów o wybranej płci (mężczyźni, kobiety) lub wszystkich pacjentów.
- **Wyszukiwanie:** Pole wyszukiwania umożliwia szybkie odnalezienie konkretnego pacjenta na podstawie dowolnych danych.
- **Sortowanie:** Lista pacjentów może być posortowana według imienia, nazwiska lub daty urodzenia, co ułatwia organizację i przeglądanie danych.
- **Usuwanie:** Usunięcie pacjenta skutkuje usunięciem jego obecności ze wszystkich tabel w tym jego wizyt czy recept. Funkcjonalność realizowana za pomocą triggera po usunięciu z tabeli pacjenci.

Interfejs wspiera również paginację, co pozwala na przeglądanie większej liczby pacjentów w przejrzysty sposób. Dzięki tym funkcjom recepcjonistka może szybko zarządzać danymi pacjentów oraz odnajdywać potrzebne informacje.

## 4.6 Umawianie wizyty



Rysunek 7: Formularz dodawania nowej wizyty w systemie.

Formularz umawiania wizyty pozwala na szybkie i wygodne zaplanowanie spotkania między pacjentem a lekarzem. Proces odbywa się w kilku krokach:

- **Wybór lekarza i pacjenta:** Użytkownik wybiera lekarza oraz pacjenta z rozwijanych list, które pobierają dane z bazy danych.
- **Opis wizyty:** Opcjonalnie można wprowadzić krótki opis wizyty, który zostanie zapisany w systemie.
- **Wybór terminu:** Po wskazaniu daty wizyty system automatycznie wyświetla listę dostępnych, niezajętych godzin. Informacje te są dynamicznie pobierane z bazy danych w czasie rzeczywistym, aby zapewnić aktualność danych.
- **Potwierdzenie wizyty:** Po wypełnieniu wszystkich pól użytkownik klika przycisk „Umów Wizytę”, aby zapisać dane w systemie.

Dzięki takiej funkcjonalności system zapewnia precyzyjne planowanie wizyt, eliminując możliwość kolizji terminów.

## 4.7 Rozpoczęcie wizyty

**Rozpocznij Wizytę**

Wybierz datę  
01/15/2025

Wybierz pacjenta  
Wybierz pacjenta...

Diagnoza

☒ Wystaw receptę

Wybierz Lek  
Wybierz lek...

Dawkowanie

Rozpocznij Wizytę

Rysunek 8: Formularz umożliwiający postawienie diagnozy oraz wypisanie recepty.

Formularz rozpoczęcia wizyty jest dostępny dla lekarza i umożliwia realizację zaplanowanego spotkania z pacjentem. Główne funkcjonalności formularza obejmują:

- **Wybór daty i pacjenta:** Lekarz wybiera datę wizyty oraz pacjenta przypisanego do tego terminu. Lista pacjentów jest dynamicznie pobierana z bazy danych, co zapewnia aktualność informacji.
- **Diagnoza:** Pole tekstowe pozwala lekarzowi wprowadzić diagnozę pacjenta, która zostanie zapisana w historii wizyty.
- **Wypisanie recepty:** Lekarz może opcjonalnie wystawić receptę, zaznaczając odpowiednie pole. W przypadku recepty dostępne są:
  - **Wybór leków:** Lista leków jest pobierana z bazy danych, co pozwala na szybkie i precyzyjne przypisanie odpowiednich medykamentów.
  - **Dawkowanie:** Lekarz wprowadza szczegóły dotyczące dawkowania wybranych leków.
- **Potwierdzenie wizyty:** Po wypełnieniu formularza i kliknięciu przycisku „Rozpocznij Wizytę” dane zostają zapisane w systemie, w tym diagnoza oraz szczegóły ewentualnej recepty.

Dzięki temu formularzowi lekarz ma możliwość kompleksowego zarządzania wizytą oraz obsługi pacjenta w sposób efektywny i zorganizowany.



## 4.8 Przeglądanie historii wizyt

### Historia Wizyt Pacjentów

Wybierz Pacjenta

Krzysztof Zacharczuk (PESEL: 77110384712) x

#### Szczegóły Wizyt

##### Wizyty

Wizyta z dnia: 12.02.2025

**Opis wizyty:**  
aaaaaaa

**Lekarz prowadzący:**  
Janek Kowalski

**Diagnoza:**  
aaaaaaa

**Recepta:**  
Numer recepty: REC20250212-8666

**Data wystawienia:** 12.02.2025

**Leki:**

- Nystatyna (Lek przeciwgrzybiczy) - 60 szt.

Wizyta z dnia: 22.01.2025

**Opis wizyty:**  
aaaaa

**Lekarz prowadzący:**  
Janek Kowalski

**Diagnoza:**  
aaaaa

**Recepta:**  
Numer recepty: REC20250122-4561

**Data wystawienia:** 22.01.2025

**Leki:**

- Metoprolol (Lek na nadciśnienie) - 20 szt.

Rysunek 9: Przykład widoku historii wizyt pacjenta.

Historia wizyt pacjenta pozwala lekarzowi na przeglądanie szczegółowych informacji o przeszłych wizytach wybranego pacjenta. Funkcjonalność ta zawiera następujące dane:

- **Data wizyty:** Każda wizyta jest przypisana do konkretnej daty.
- **Opis wizyty:** Informacje wprowadzone przez lekarza w trakcie wizyty.
- **Lekarz prowadzący:** Nazwisko lekarza, który prowadził wizytę.
- **Diagnoza:** Szczegóły dotyczące diagnozy postawionej w trakcie wizyty.
- **Recepta:**
  - **Numer recepty:** Unikalny numer wygenerowany za pomocą funkcji w bazie danych.
  - **Data wystawienia:** Data, kiedy recepta została wydana.
  - **Przepisane leki:** Lista leków wraz z ich nazwami, przeznaczeniem oraz ilością.

Historia wizyt jest dynamicznie generowana na podstawie danych przechowywanych w bazie, co pozwala na szybki dostęp do istotnych informacji medycznych. Dzięki tej funkcji lekarz może w łatwy sposób przeanalizować historię leczenia pacjenta i podejmować bardziej świadome decyzje dotyczące dalszego leczenia.

## 4.9 Kalendarz lekarza

**Kalendarz wizyt**

Wybierz lekarza:  
Janek Kowalski

← Poprzedni tydzień      Tydzień: 13.01.2025 - 17.01.2025      Następny tydzień →

Godzina	Poniedziałek 13.01.2025	Wtorek 14.01.2025	Środa 15.01.2025	Czwartek 16.01.2025	Piątek 17.01.2025
07:00					
07:30					
08:00		81021463418			
08:30		74042556789			
09:00		76061865243			
09:30					
10:00					
10:30					
11:00					
11:30					
12:00					
12:30					
13:00					

Rysunek 10: Kalendarz wizyt dla wybranego lekarza.

Kalendarz wizyt umożliwia zarządzanie harmonogramem pracy lekarzy. Widok kalendarza różni się w zależności od rodzaju użytkownika:

- **Recepcjonistka:**

- Może wybrać dowolnego lekarza z rozwijanej listy.
- Widzi szczegółowy harmonogram wizyt wybranego lekarza w układzie tygodniowym.
- Ma możliwość zarządzania wizytami, w tym ich odwoływania.

- **Lekarz:**

- Widzi jedynie swój własny kalendarz wizyt.
- Ma możliwość bezpośredniego przejścia z kalendarza do formularza rozpoczęcia wizyty.
- Harmonogram wyświetla godziny zaplanowanych wizyt, co pozwala lekarzowi na łatwe planowanie pracy.

Kalendarz umożliwia nawigację między tygodniami za pomocą przycisków „Poprzedni tydzień” i „Następny tydzień”. Dzięki tej funkcji recepcjonistka i lekarz mogą w łatwy sposób zarządzać swoimi obowiązkami i harmonogramem pracy.

## 4.10 Raport pracy lekarza

### Raport Pracy Lekarza

Wybierz Rok:

2025

#### Statystyki Roczne

Całkowita liczba wizyt: 7

Średnia liczba wizyt na miesiąc: 2.33

Ilość przepisanych leków: 9

Łączny koszt leków: 318.00 PLN

#### Statystyki Miesięczne

Wybierz Miesiąc:

styczeń

Całkowita liczba wizyt w miesiącu: 5

Ilość przepisanych leków: 6

Łączny koszt leków w miesiącu: 154.00 PLN

Rysunek 11: Raport pracy lekarza, który zawiera statystyki miesięczne jak i roczne.

Raport pracy lekarza pozwala na przeglądanie szczegółowych statystyk dotyczących liczby wizyt i przepisanych leków w określonym przedziale czasu. Lekarz ma dostęp do następujących informacji:

- **Statystyki roczne:**

- *Całkowita liczba wizyt:* Łączna liczba pacjentów obsłużonych w wybranym roku.
- *Średnia liczba wizyt na miesiąc:* Obliczona na podstawie całkowitej liczby wizyt w roku.
- *Ilość przepisanych leków:* Łączna liczba leków przypisanych pacjentom w ciągu roku.
- *Łączny koszt leków:* Suma kosztów wszystkich leków przepisanych w danym roku.

- **Statystyki miesięczne:**

- *Całkowita liczba wizyt w miesiącu:* Liczba wizyt przeprowadzonych w wybranym miesiącu.
- *Ilość przepisanych leków:* Liczba leków przepisanych pacjentom w wybranym miesiącu.
- *Łączny koszt leków w miesiącu:* Suma kosztów leków przypisanych w określonym miesiącu.

Lekarz może wybierać rok i miesiąc za pomocą rozwijanych list, co pozwala na szybki dostęp do interesujących go statystyk. Dzięki tej funkcji lekarz ma możliwość analizowania swojej pracy w sposób dokładny i przejrzysty.

## 5 Dokumentacja

### 5.1 Wprowadzanie danych

Dane zostały wprowadzone ręcznie, z wyjątkiem tabeli leków, która została zaimportowana z pliku zewnętrznego.

### 5.2 Instrukcja obsługi aplikacji

Poniżej przedstawiono podstawowe kroki obsługi aplikacji:

#### 1. Logowanie do systemu:

- Użytkownik wprowadza nazwę użytkownika oraz hasło w formularzu logowania.
- Po zalogowaniu zostaje przekierowany do odpowiedniego panelu w zależności od swojej roli (lekarz, recepcjonistka, administrator).

#### 2. Dodawanie użytkowników (administrator):

- Administrator ma możliwość dodawania nowych użytkowników (administratorów, recepcjonistek, lekarzy).
- W formularzu tworzenia użytkownika należy wypełnić wymagane dane, takie jak imię, nazwisko, adres e-mail i typ użytkownika.

#### 3. Umawianie wizyt (recepcjonistka):

- Recepcjonistka wybiera pacjenta i lekarza, a następnie ustala datę i godzinę wizyty.
- System wyświetla tylko dostępne godziny w harmonogramie lekarza.

#### 4. Przeglądanie historii wizyt (lekarz):

- Lekarz ma dostęp do historii wizyt pacjentów, gdzie znajdują się szczegóły diagnoz, przepisanych leków i recept.
- W celu rozpoczęcia wizyty można przejść bezpośrednio z kalendarza lekarza do odpowiedniego formularza.

#### 5. Raport pracy lekarza:

- Lekarz może wygenerować statystyki roczne i miesięczne, aby przeanalizować swoją pracę.
- Raport zawiera dane o liczbie wizyt, przepisanych lekach oraz łącznych kosztach leków.

### 5.3 Technologie użyte w projekcie

W projekcie zastosowano nowoczesne technologie backendowe i frontendowe, które umożliwiają dynamiczne generowanie treści oraz efektywne zarządzanie danymi. Poniżej opisano kluczowe technologie użyte w aplikacji:

### 5.3.1 Node.js

Node.js to środowisko uruchomieniowe JavaScript, które umożliwia tworzenie szybkich i skalowalnych aplikacji serwerowych. W projekcie Node.js został wykorzystany jako baza do obsługi backendu aplikacji, dzięki czemu:

- Możliwe było stworzenie wydajnych API do zarządzania danymi.
- Obsługa asynchroniczna pozwoliła na jednoczesne przetwarzanie wielu zapytań użytkowników.
- Rozszerzono aplikację o liczne moduły z wykorzystaniem menedżera pakietów `npm`.

### 5.3.2 Express.js

Express.js to minimalistyczny framework dla Node.js, który upraszcza tworzenie serwerów HTTP oraz zarządzanie trasami (routing). W projekcie Express.js został wykorzystany do:

- Tworzenia tras dla różnych funkcjonalności aplikacji (np. logowanie, rejestracja pacjentów, umawianie wizyt).
- Obsługi formularzy oraz przetwarzania danych przesyłanych metodami HTTP (POST, GET).
- Integracji z szablonami EJS w celu dynamicznego generowania stron HTML.

### 5.3.3 EJS (Embedded JavaScript)

EJS to silnik szablonów dla Node.js, który umożliwia dynamiczne generowanie stron HTML z osadzonymi danymi. W projekcie EJS został wykorzystany do:

- Tworzenia przejrzystych i łatwo modyfikowalnych szablonów widoków (np. logowanie, rejestracja, raporty).
- Dynamicznego przekazywania danych z backendu do frontendu.
- Reużywalności komponentów dzięki mechanizmowi dziedziczenia układu (`layouts`).

## 5.4 Literatura

1. PostgreSQL Global Development Group, *PostgreSQL Documentation*, dostępne online: <https://www.postgresql.org/docs/>.
2. Hans-Jürgen Schönig, *Mastering PostgreSQL 15*, Packt Publishing, 2023.
3. Ethan Brown, *Web Development with Node and Express: Leveraging the JavaScript Stack*, O'Reilly Media, 2019.
4. Express.js Documentation, dostępne online: <https://expressjs.com/>.
5. EJS Documentation, dostępne online: <https://ejs.co/>.
6. Jacob Thornton, Mark Otto, *Bootstrap Documentation*, dostępne online: <https://getbootstrap.com/>.

## 5.5 Link

<http://pascal.fis.agh.edu.pl:4013>