Obraz zawierający tekst, Czcionka, Grafika, zrzut ekranu

Opis wygenerowany automatycznie

**Kolegium Nauk Przyrodniczych**

**Kierunek:**Informatyka

**Rok akademicki:**2023/2024 (4 semestr)

**Przedmiot:**Bazy danych

**Prowadzący:**dr inż. Piotr Grochowalski

**Temat projektu:**System zarządzania placówką medyczną

**Wykonanie:**Krzysztof Motas  
125145

Patryk Jarosiewicz  
123066

1. **Opis projektu**

Projekt bazy danych został stworzony z myślą o zarządzaniu informacjami medycznymi w placówce medycznej, zapewniając wsparcie w prowadzeniu dokumentacji dotyczącej personelu, lekarstw, metod leczenia oraz pacjentów.

Dzięki naszej aplikacji personel medyczny może zarządzać danymi pacjentów, śledzić historię chorób i leczenia, oraz szybko dostępować do informacji o lekarzach, lekach i  gabinetach.

1. **Funkcjonalności projektu**

Baza danych umożliwia dokładne śledzenie historii wizyt każdego pacjenta u lekarza, w tym datę wizyty, powód wizyty oraz diagnozę lekarza. Ta funkcjonalność pozwala na  skuteczne monitorowanie stanu zdrowia pacjentów oraz świadczenie im właściwej opieki medycznej.

Ważnym elementem projektu jest również moduł dotyczący leków i recept. Baza danych przechowuje informacje o dostępnych lekach, wraz z ich ceną. Ponadto, system umożliwia przypisywanie leków do konkretnych recept przepisanych pacjentom podczas wizyt, wraz z informacjami dotyczącymi dawkowania oraz odpłatności leku.

Niektóre z dostępnych funkcji:

* rejestrowanie pacjentów,
* przyjmowanie wizyt,
* tworzenie dokumentacji medycznej,
* przepisywanie leków,
* analiza danych, np. popularność przepisywanych leków, lekarze z największą liczbą wizyt, średni czas trwania wizyty, statystyki dotyczące wieku pacjentów, popularność specjalizacji lekarskich, średnia liczba wizyt w zależności od wieku pacjentów, rozkład płci pacjentów w różnych grupach wiekowych itp.,
* historia pacjentów.

Dzięki temu systemowi personel medyczny może śledzić pacjentów i zapewnić im  odpowiednią opiekę medyczną.

1. **Zaprojektowane tabele oraz powiązania**

Na podstawie wyżej wymienionych założeń i opisu funkcjonalności projektu, stworzono następujące tabele:

* Tabela patients (Pacjenci):
* Identyfikator: id (integer, klucz główny)
* Imię: name (varchar)
* Nazwisko: last\_name (varchar)
* Płeć: gender (varchar)
* Adres: address (varchar)
* Numer telefonu: phone\_number (varchar)
* Data urodzenia: date\_of\_birth (date)
* Tabela doctors (Lekarze):
* Identyfikator: id (integer, klucz główny)
* Imię: name (varchar)
* Nazwisko: last\_name (varchar)
* Specjalizacja: specialization (varchar)
* Numer telefonu: phone\_number (varchar)
* Tabela visits (Wizyty):
* Identyfikator: id (integer, klucz główny)
* Identyfikator pacjenta: patient\_id (integer, klucz obcy od patients.id)
* Identyfikator lekarza: doctor\_id (integer, klucz obcy od doctors.id)
* Powód: reason (varchar)
* Data rozpoczęcia: start\_date (datetime)
* Data zakończenia: end\_date (datetime)
* Tabela documentations (Dokumentacje):
* Identyfikator: id (integer, klucz główny)
* Identyfikator wizyty: visit\_id (integer, klucz obcy od visits.id)
* Data: date (datetime)
* Diagnoza: diagnosis (varchar)
* Metoda leczenia: treatment\_method (varchar)
* Tabela prescriptions (Recepty):
* Identyfikator: id (integer, klucz główny)
* Identyfikator wizyty: visit\_id (integer, klucz obcy od visits.id)
* Data ważności: expiration\_date (datetime)
* Kod: code (varchar)
* Tabela medicines (Leki):
* Identyfikator: id (integer, klucz główny)
* Nazwa: name (varchar)
* Cena: price (integer)
* Tabela prescription\_medicines (Leki na receptach):
* Identyfikator recepty: prescription\_id (integer, klucz obcy od prescriptions.id) Identyfikator leku: medicine\_id (integer, klucz obcy od medicines.id) Dawkowanie: dosage (varchar)
* Opłata: payment (integer)

Pomiędzy tabelami zostały stworzone następujące relacje:

* patients.id (Tabela patients) < visits.patient\_id (Tabela visits)

Jest to relacja jeden-do-wielu (<). Jeden pacjent może mieć wiele wizyt.

* doctors.id (Tabela doctors) < visits.doctor\_id (Tabela visits)

Jest to relacja jeden-do-wielu (<). Jeden lekarz może przeprowadzić wiele wizyt.

* visits.id (Tabela visits) - documentations.visit\_id (Tabela documentations)

Jest to relacja jeden-do-jednego (-). Jedna wizyta może mieć tylko jedną dokumentację medyczną.

* prescriptions.visit\_id (Tabela prescriptions) - visits.id (Tabela visits)

Jest to relacja jeden-do-jednego (-). Jedna wizyta może mieć tylko jedną receptę.

* prescription\_medicines.prescription\_id (Tabela prescription\_medicines) > prescriptions.id (Tabela prescriptions)

Relacja wiele-do-jednego (>). Wiele leków może być przypisanych do jednej recepty.

* prescription\_medicines.medicine\_id (Tabela prescription\_medicines) - medicines.id (Tabela medicines)

Relacja jeden-do-jednego (-). Oznacza to, że każde przypisanie leku na recepcie może być powiązane tylko z jednym lekiem, który jest zdefiniowany w tabeli leków.

Do wizualnego przedstawienia tabel oraz powiązań pomiędzy nimi, stworzono poniższy diagram ERD.

Obraz zawierający tekst, zrzut ekranu, oprogramowanie, numer

Opis wygenerowany automatycznie

Zrzut ekranu 1 - Diagram ERD

1. **Wybór technologii**

Projekt został zrealizowany przy użyciu narzędzi i technologii. Wykorzystane technologie obejmują różne języki programowania, bazę danych, biblioteki oraz framework.

* 1. **Bazy danych**
* **Oracle Database:**
* Wersja: 18c
* Link do dokumentacji: <https://docs.oracle.com/en/database/>
  1. **Języki programowania**
* **PHP:**
* Wersja: PHP 8.2.12
* Link do dokumentacji: <https://www.php.net/docs.php>
* **JavaScript:**
* Wersja: ECMAScript 2023
* Link do dokumentacji: <https://developer.mozilla.org/enUS/docs/Web/JavaScript>
  1. **Języki znaczników**
* **HTML:**
* Wersja: HTML 5
* Link do dokumentacji: <https://developer.mozilla.org/en-US/docs/Web/HTML>
  1. **Języki arkuszy stylów**
* **CSS:**
* Wersja: CSS 5
* Link do dokumentacji: <https://developer.mozilla.org/en-US/docs/Web/CSS>
  1. **Narzędzia**
* **Docker:**
* Wersja: 26.1.3
* Link: <https://docs.docker.com/>
* **Composer:**
* Wersja: 2.7.6
* Link do dokumentacji: <https://getcomposer.org/doc/>
  1. **Biblioteki i frameworki**
* **Laravel:**
* Wersja: 11.0.9
* Link do dokumentacji: <https://laravel.com/docs/11.x/readme>
* **Bootstrap:**
* Wersja: 5.3.3
* Link do dokumentacji: <https://getbootstrap.com/>
  1. **Środowiska programistyczne**
* **Visual Studio Code:**
* Wersja: 1.89.1
* Link: <https://code.visualstudio.com/>

1. **Funkcjonalności**

W celu zaimplementowania głównych funkcjonalności aplikacji, stworzono wiele procedur i funkcji przy użyciu języka PL/SQL. Dzięki temu możliwe było zrealizowanie skomplikowanych operacji na bazie danych.

|  |  |  |
| --- | --- | --- |
| Nazwa | Typ | Opis |
| SEARCH\_EXPENSIVE\_MEDICINES | FUNCTION | Szuka drogich leków |
| CALCULATE\_AVERAGE\_MEDICINE\_PRICE | FUNCTION | Oblicza średnią cenę leku |
| GENERATE\_DOCTOR\_PATIENT\_COUNT\_REPORT | FUNCTION | Generuje raport ilości pacjentów |
| GENERATE\_TOP\_DIAGNOSIS\_REPORT | FUNCTION | Generuje raport najczęstszych diagnoz |
| GENERATE\_VISIT\_COUNT\_BY\_SPECIALIZATION\_REPORT | FUNCTION | Generuje raport wizyt wg specjalizacji |
| GET\_ALL\_DOCTORS | FUNCTION | Pobiera wszystkich lekarzy |
| GET\_ALL\_PATIENTS | FUNCTION | Pobiera wszystkich pacjentów |
| GET\_ALL\_VISITS | FUNCTION | Pobiera wszystkie wizyty |
| GET\_PRESCRIPTION\_MEDICINES | FUNCTION | Pobiera leki z recepty |
| SEARCH\_DOCTORS\_BY\_SPECIALIZATION | FUNCTION | Szuka lekarzy wg specjalizacji |
| SEARCH\_TOP\_PRESCRIBED\_MEDICINES\_BY\_DOCTOR | FUNCTION | Szuka najczęściej przepisywanych leków |
| SEARCH\_VISITS\_BY\_PATIENT\_LAST\_NAME | FUNCTION | Szuka wizyt wg nazwiska pacjenta |
| LOGIN\_PATIENT | FUNCTION | „Loguje” pacjenta |
| LOGIN\_DOCTOR | FUNCTION | „Loguje” lekarza |
| GET\_ALL\_MEDICINES | FUNCTION | Pobiera wszystkie leki |
| GET\_PATIENT\_ID | FUNCTION | Pobiera ID pacjenta |
| GET\_DOCTOR\_VISITS | FUNCTION | Pobiera wizyty lekarza |
| GET\_PATIENT\_PRESCRIPTIONS | FUNCTION | Pobiera recepty pacjenta |
| CALCULATE\_AVERAGE\_VISIT\_TIME | FUNCTION | Oblicza średni czas wizyty |
| CALCULATE\_AVERAGE\_VISITS\_BY\_AGE | FUNCTION | Oblicza średnią wizyt wg wieku |
| GET\_PATIENT\_VISITS | FUNCTION | Pobiera wizyty pacjenta |
| ADD\_DOCUMENTATION | PROCEDURE | Dodaje dokumentację |
| ADD\_MEDICINE | PROCEDURE | Dodaje lek |
| ADD\_PATIENT | PROCEDURE | Dodaje pacjenta |
| ADD\_PRESCRIPTION | PROCEDURE | Dodaje receptę |
| ADD\_PRESCRIPTION\_MEDICINE | PROCEDURE | Dodaje lek do recepty |
| ADD\_VISIT | PROCEDURE | Dodaje wizytę |
| DELETE\_DOCUMENTATION | PROCEDURE | Usuwa dokumentację |
| DELETE\_PRESCRIPTION | PROCEDURE | Usuwa receptę |
| DELETE\_PRESCRIPTION\_MEDICINE | PROCEDURE | Usuwa lek z recepty |
| UPDATE\_DOCUMENTATION | PROCEDURE | Aktualizuje dokumentację |
| UPDATE\_PATIENT | PROCEDURE | Aktualizuje dane pacjenta |
| UPDATE\_PRESCRIPTION | PROCEDURE | Aktualizuje receptę |
| UPDATE\_PRESCRIPTION\_MEDICINE | PROCEDURE | Aktualizuje lek w recepcie |
| DELETE\_VISIT | PROCEDURE | Usuwa wizytę |
| DELETE\_PRESCRIPTION\_MEDICINE\_BY\_PRESCRIPTION\_ID | PROCEDURE | Usuwa lek z recepty wg ID |
| DELETE\_VISIT\_AND\_ASSOCIATED\_DATA | PROCEDURE | Usuwa wizytę i powiązane dane |
| UPDATE\_VISIT | PROCEDURE | Aktualizuje wizytę |

* 1. **Opis przykładowych funkcji i procedur z zakresu funkcjonalności typu CRUD**
* **ADD\_VISIT**

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Zrzut ekranu 2 - Implementacja ADD\_VISIT

Procedura służy do dodawania nowych wizyt do systemu medycznego. Przyjmuje ona parametry dotyczące identyfikatora pacjenta, identyfikatora lekarza, powodu wizyty oraz daty rozpoczęcia i zakończenia wizyty. Następnie procedura dodaje te informacje do tabeli "VISITS" w bazie danych i potwierdza transakcję. W przypadku wystąpienia błędu, procedura wykonuje rollback transakcji i przekazuje wyjątek do obsługi wyżej.

Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 3 - Wykorzystanie procedury w kodzie PHP

Obraz zawierający tekst, zrzut ekranu, numer, Czcionka

Opis wygenerowany automatycznie

Zrzut ekranu 4 - Formularz tworzenia wizyty

Obraz zawierający tekst, Jaskrawoniebieski, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 5 - Potwierdzenie dodania wizyty

* **GET\_PATIENT\_PRESCRIPTIONS**

Obraz zawierający tekst, zrzut ekranu, Czcionka, numer

Opis wygenerowany automatycznie

Zrzut ekranu 6 - Implementacja funkcji

Funkcja zwraca kursor typu SYS\_REFCURSOR zawierający informacje o receptach pacjenta o określonym identyfikatorze. Wewnętrznie, funkcja otwiera kursor i  wykonuje zapytanie SQL, które pobiera wszystkie informacje o receptach pacjenta, łącząc je z informacjami o wizytach (takimi jak data rozpoczęcia wizyty) oraz danymi lekarza, który przepisał receptę. Następnie kursor jest zwracany jako wynik funkcji.

Obraz zawierający tekst, Czcionka, linia, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 7 - Wykorzystanie funkcji w kodzie PHP

Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

Zrzut ekranu 8 - Implementacja funkcji w PHP obsługującej funkcję zwracającą kursor

Obraz zawierający tekst, Czcionka, linia, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 9 - Lista wszystkich recept dla danego pacjenta

Obraz zawierający tekst, zrzut ekranu, oprogramowanie

Opis wygenerowany automatycznie

Zrzut ekranu 10 - Szczegóły danej recepty

* **UPDATE\_PATIENT**

Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

Zrzut ekranu 11 - Implementacja procedury

Procedura do aktualizacji danych pacjenta w systemie medycznym. Przyjmuje ona parametry, takie jak identyfikator pacjenta, imię, nazwisko, płeć, adres, numer telefonu oraz datę urodzenia. Następnie procedura wykonuje zapytanie UPDATE, aktualizując dane pacjenta w tabeli "PATIENTS" zgodnie z przekazanymi parametrami. Po udanej aktualizacji, zmiany są zatwierdzane przez COMMIT. W  przypadku wystąpienia błędu, procedura wykonuje rollback transakcji i  przekazuje wyjątek do obsługi wyżej.

Obraz zawierający tekst, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 12 - Wykorzystanie procedury w kodzie PHP

Obraz zawierający tekst, zrzut ekranu, numer, Czcionka

Opis wygenerowany automatycznie

Zrzut ekranu 13 - Ustawienia konta pacjenta

Obraz zawierający tekst, Jaskrawoniebieski, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 14 - Powiadomienie o pomyślnym zaktualizowaniu ustawień

* **DELETE\_VISIT\_AND\_ASSOCIATED\_DATA**

Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

Obraz zawierający tekst, zrzut ekranu, Czcionka, dokument

Opis wygenerowany automatycznie

Zrzut ekranu 15 - Implementacja procedury

Procedura służy do usuwania danych związanych z wizytą w systemie medycznym na podstawie przekazanego identyfikatora wizyty.

Najpierw procedura próbuje pobrać identyfikator dokumentacji medycznej i  recepty przypisanej do wizyty. Jeśli takie dane istnieją, są one usuwane zgodnie z  kolejnością: najpierw leki przypisane do recepty, następnie sama recepta oraz dokumentacja medyczna. Po usunięciu wszystkich powiązanych danych, procedura usuwa informacje o samej wizycie.

Po pomyślnym zakończeniu operacji, zmiany są zatwierdzane przez COMMIT. W  przypadku wystąpienia błędu, procedura wykonuje rollback transakcji i  wyświetla informacje o błędzie.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Zrzut ekranu 16 - Wykorzystanie procedury w kodzie PHP

Obraz zawierający tekst, zrzut ekranu, multimedia, gadżet

Opis wygenerowany automatycznie

Zrzut ekranu 17 - Potwierdzenie usunięcia wizyty

Obraz zawierający tekst, Jaskrawoniebieski, Czcionka, zrzut ekranu

Opis wygenerowany automatycznie

Zrzut ekranu 18 – Powiadomienie

* 1. **Opis funkcji do analizy danych**
* **SEARCH\_EXPENSIVE\_MEDICINES**
* **CALCULATE\_AVERAGE\_MEDICINE\_PRICE**
* **GENERATE\_DOCTOR\_PATIENT\_COUNT\_REPORT**
* **GENERATE\_TOP\_DIAGNOSIS\_REPORT**
* **GENERATE\_VISIT\_COUNT\_BY\_SPECIALIZATION\_REPORT**
* **SEARCH\_TOP\_PRESCRIBED\_MEDICINES\_BY\_DOCTOR**
* **CALCULATE\_AVERAGE\_VISIT\_TIME**
* **CALCULATE\_AVERAGE\_VISITS\_BY\_AGE**