



**WYŻSZA SZKOŁA
INFORMATYKI i ZARZĄDZANIA**
z siedzibą w Rzeszowie

KOLEGIUM INFORMATYKI STOSOWANEJ

Kierunek: INFORMATYKA

Specjalność: PROGRAMOWANIE

Oleksandr Lobchenko
Nr albumu studenta w68317

APLIKACJA DESKTOPOWA "SZPITAL+"

Prowadzący: mgr inż. Ewa Żesławska

Praca projektowa programowanie obiektowe C#

Rzeszów 2024

Spis treści

Wstęp	4
1 Opis założeń projektu	5
1.1 Cele projektu	5
1.2 Wymagania funkcjonalne	5
1.3 Wymagania niefunkcjonalne	5
2 Opis struktury projektu	6
2.1 Diagram klas	6
2.2 Narzędzie	10
2.3 Baza danych	11
2.4 Minimalne wymagania sprzętowe	12
3 Harmonogram realizacji projektu	13
3.1 Diagram Gantt'a	13
3.2 Repozytorium	13
4 Prezentacja warstwy użytkowej projektu	14
4.1 Logowanie	14
4.2 Główne okno	16
4.3 Dane personalne	18
4.4 Zmienianie hasła	18
4.5 Sprawdzenie nowego hasła	19
4.6 Okna dla recepcjonisty	21
4.7 Okna dla głównego kierownika	24
4.8 Okna dla kierownika	25
5 Podsumowanie	26
Bibliografia	27
Spis rysunków	28

Wstęp

"Aplikacja desktopowa Szpital+" jest aplikacją mającą na celu zmodernizowanie i usprawnienie codziennych operacji w placówkach medycznych. Jest to program umożliwiający do przechowywania i dodawania informacji dotyczących szpitalu. Medycyna zajmuje bardzo ważną część naszego życia. Dlatego uważam, że ułatwienie komunikacji między pracownikami zaoszczędzi czas i ulepszy jakość pracy.

Rozdział 1

Opis założeń projektu

1.1 Cele projektu

Celem głównym projektu jest stworzenie kompleksowej aplikacji desktopowej, mającej na celu usprawnienie procesów zarządzania w środowisku szpitalnym. Aplikacja "Szpital+" ma usprawnić gromadzenie danych medycznych oraz zwiększyć efektywność komunikacji w placówce medycznej. Aplikacja posiada 4 klienta, funkcjonalność których się różni:

- a) Klient Głównego Kierownika
- b) Klient Kierownika Działu
- c) Klient Recepcjonisty
- d) Klient Lekarza

1.2 Wymagania funkcjonalne

- Logowanie do systemu.
- Przegląd informacji o sobie lub innym pracowniku.
- Możliwość dodawać, usuwać lub zmieniać wizyty.
- Możliwość dodawać lub zwalniać pracowników.
- Możliwość wylogowania.
- Dodanie lub usuwanie zapisów w książkach pacjentów.
- Dodanie i usunięcie pacjentów.

1.3 Wymagania niefunkcjonalne

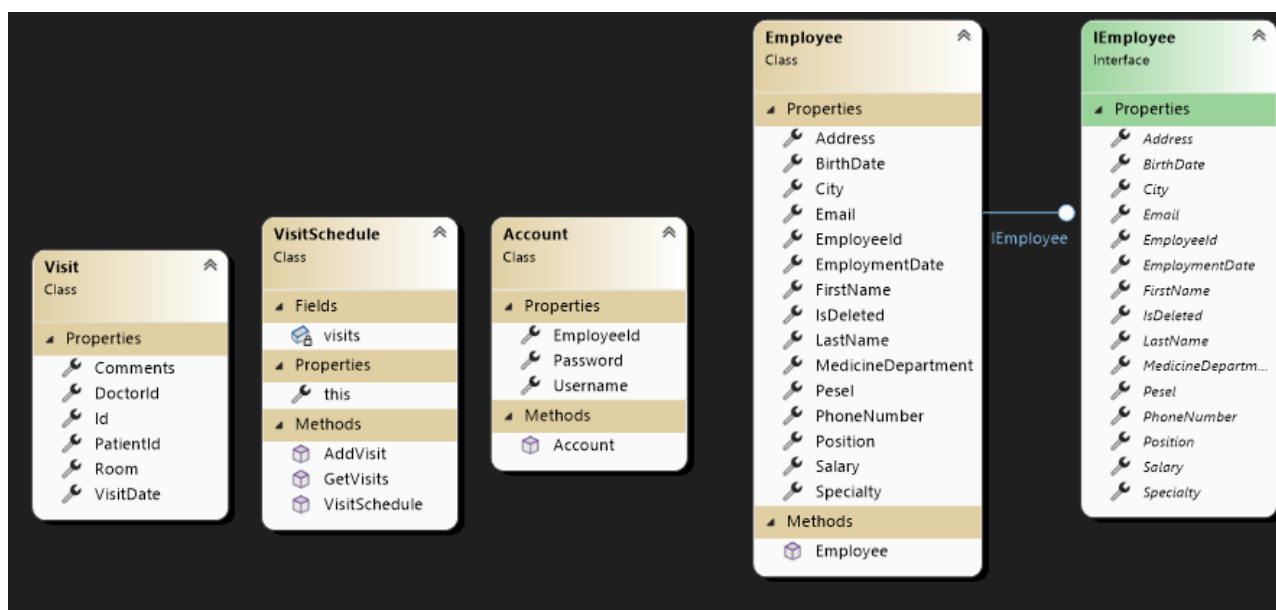
- **Łatwy w użyciu i przejrzysty interfejs użytkownika:** interfejs musi odpowiadać nowoczesnym stylom budowania aplikacji GUI.
- **Walidacja danych:** program musi sprawdzać poprawność wpisanych przez użytkownika danych.
- **Wyświetlenie komunikatów:** przy wpisaniu błędnych danych aplikacja ma powiadomić użytkownika o pojawiającym się błędzie.
- **Prawidłowo działająca baza danych:** baza danych ma pozwalać na operacje CRUD i być logicznie zdefiniowana.

Rozdział 2

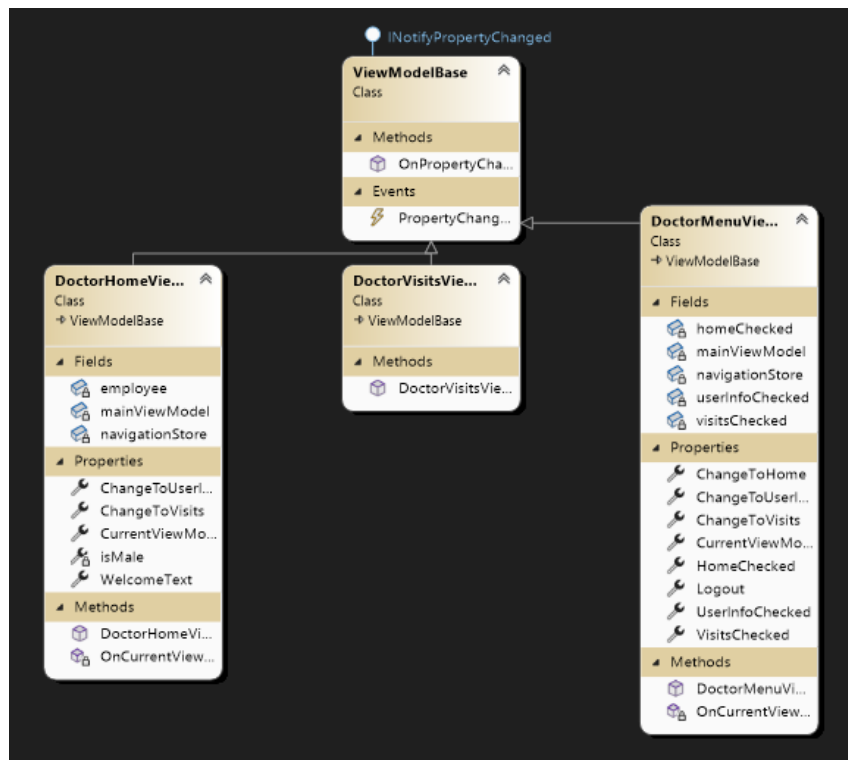
Opis struktury projektu

Projekt składa się z kilku części: Klasy główne (Models), Klasy używane do nawigacji, Komendy, Klasa do przechowywania ViewModels (NavigateStore) i Klasa do wykorzystania bazy danych. Klasy główne używane są do przechowywania danych z bazy (np. Pracownik lub Wizyta). Klasy nawigacyjne używane są do przechowywania danych i operacji nad danymi pobranych od użytkownika w widoku (View) lub z bazy danych (np. UserInfoViewModel przechowuje informacje pobraną z bazy danych i przenosi ją do UserInfoView). Komendy wykorzystane są do nawigacji, przesłania danych do bazy danych lub otrzymania danych.

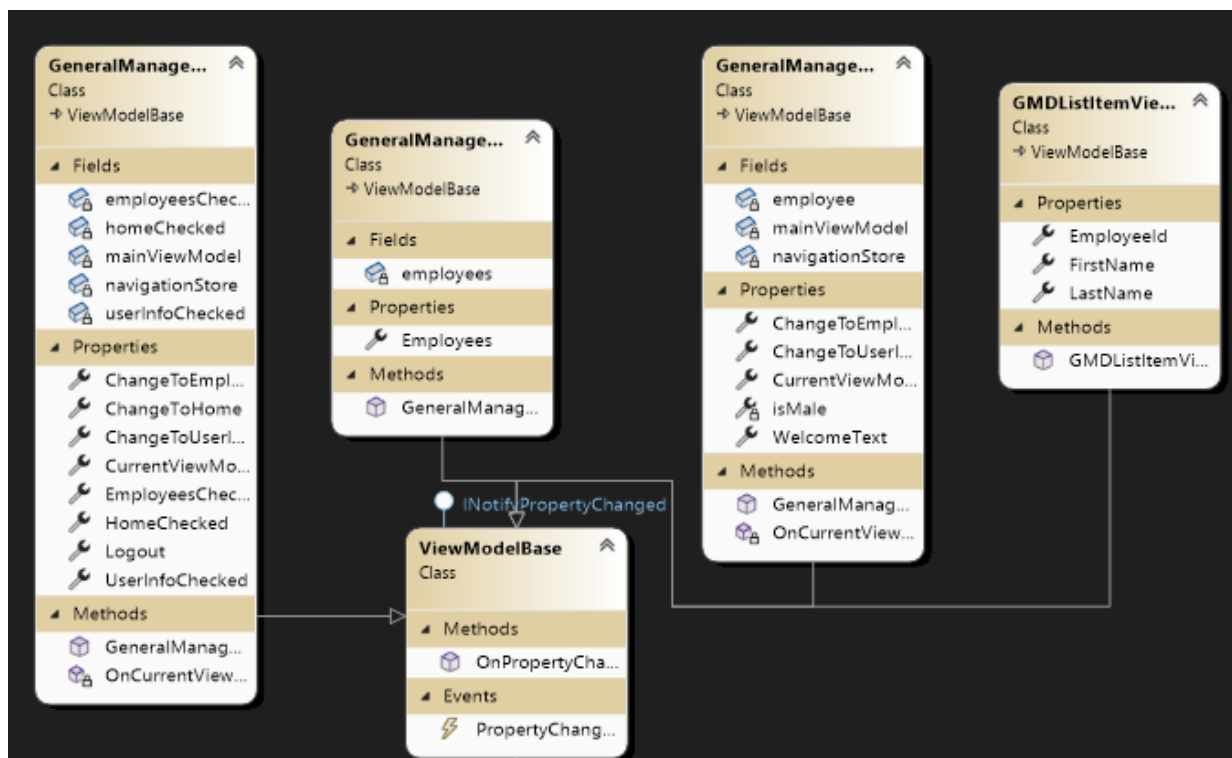
2.1 Diagram klas



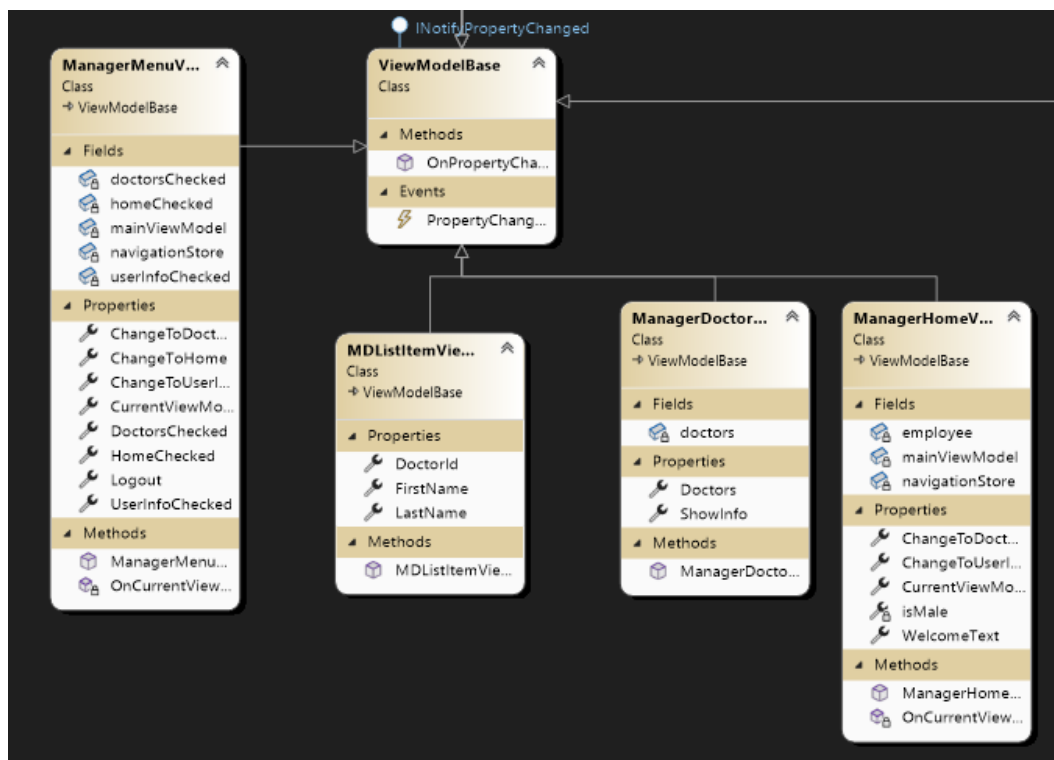
Rysunek 2.1: Klasy główne



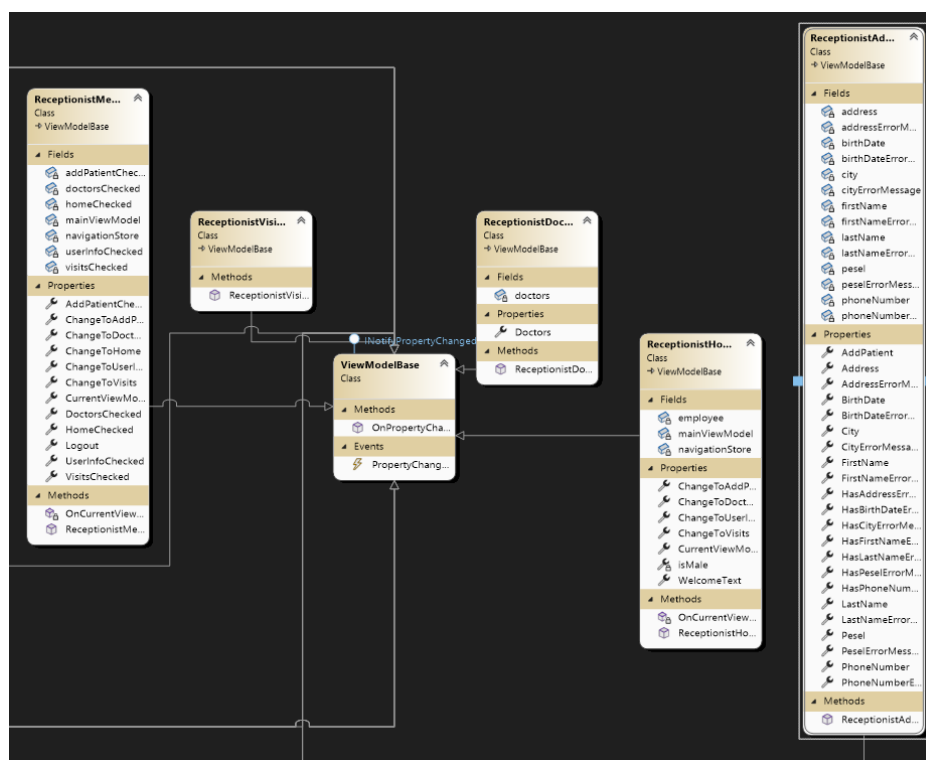
Rysunek 2.2: Klasy nawigacyjne dla doktora



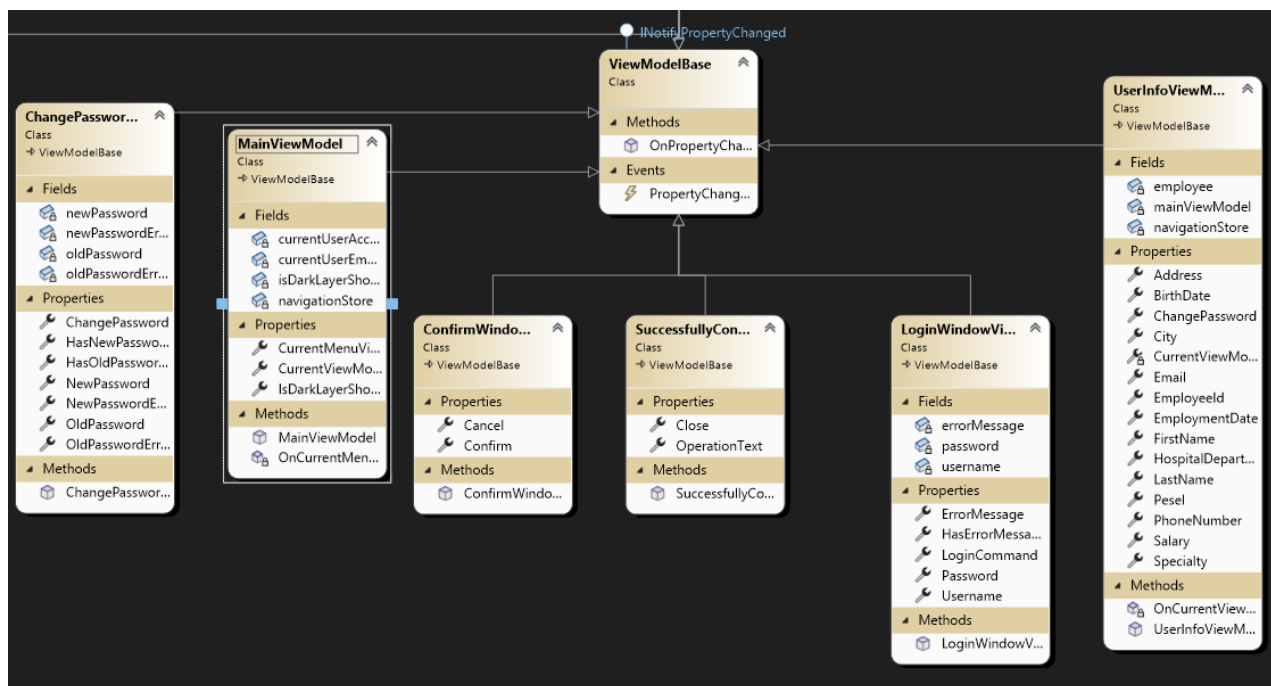
Rysunek 2.3: Klasy nawigacyjne dla głównego kierownika



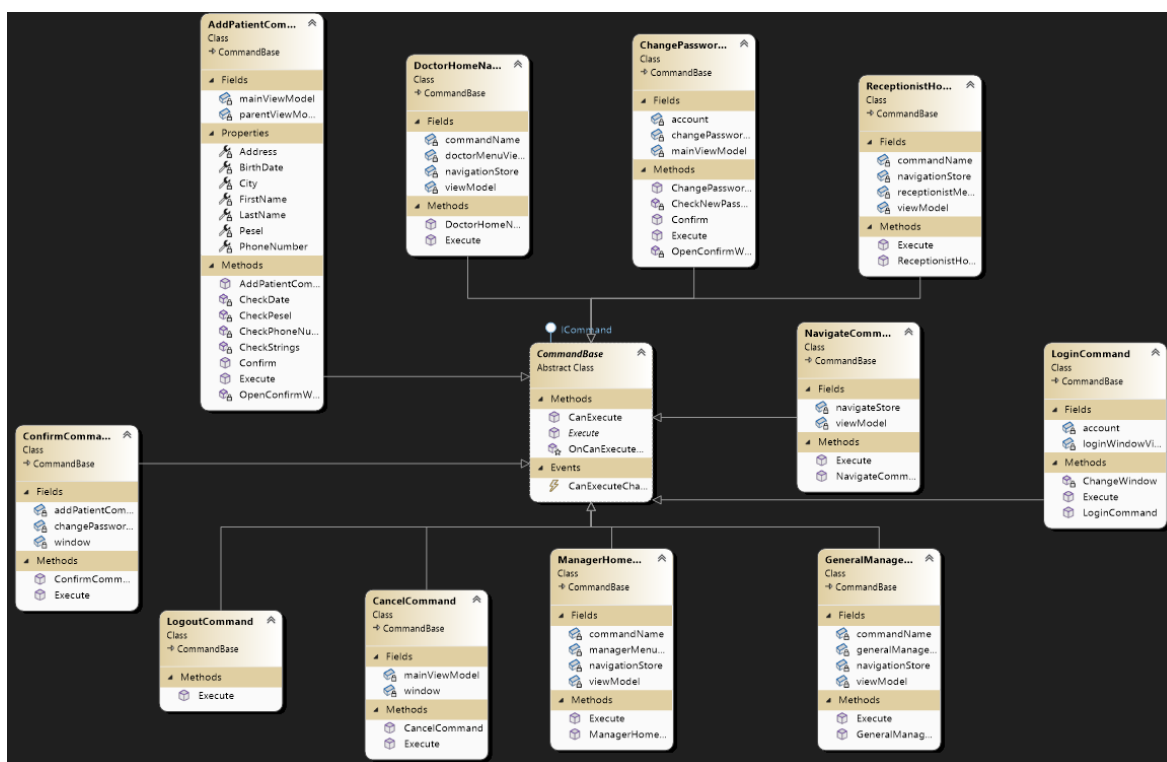
Rysunek 2.4: Klasy nawigacyjne dla kierownika



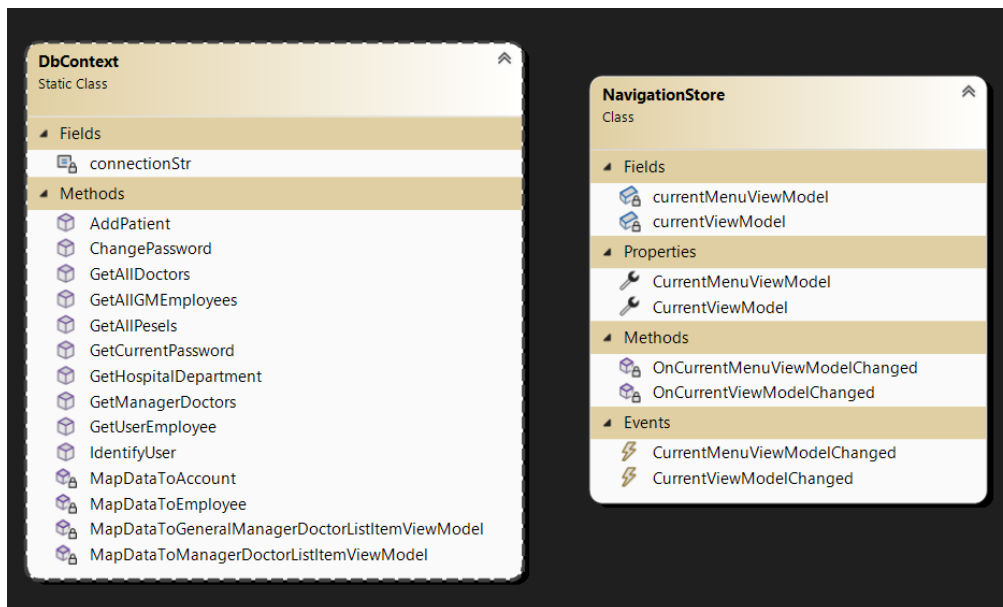
Rysunek 2.5: Klasy nawigacyjne dla recepcjonisty



Rysunek 2.6: Klasy nawigacyjne wspólne dla wszystkich klientów



Rysunek 2.7: Klasy komend



Rysunek 2.8: Klasy dla bazy danych i przechowywania nawigacji

2.2 Narzędzie

Do tworzenia aplikacji "Szpital+" korzystałem z narzędzi [WPF\(Windows Presentation Foundation\)](#). Jest to narzędzie do tworzenia aplikacji desktopowych dla systemów Windows na bazie [.Net Framework](#). Wykorzystuje język opisy interfejsu użytkownika([XAML](#)) oraz język programowania [C#](#) do implementacji funkcjonalności elementów i innych funkcji backend.

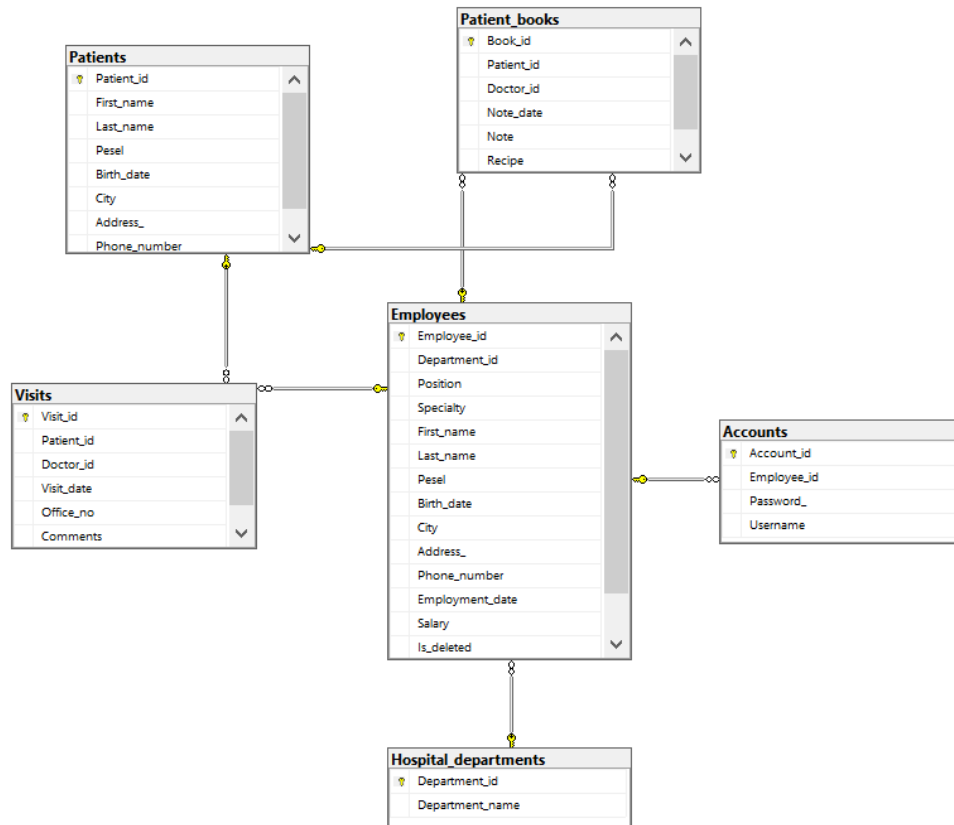
Dodatkowo przestrzegałem [wzoru architektonicznego MVVM](#)(Model-View-ViewModel). Jest to wzór który rozdziela graficzny interfejs użytkownika GUI(View) od implementacji logiki biznesowej oraz logiki backend. Związkiem między tymi warstami jest konwerter wartości (ViewModel), który przyjmuje publiczne właściwości modeli(models) i przekazuje ich do widoków(Views) za pomocą wiązania (Binding).

Do pisania kodu korzystałem z środowiska programistycznego [Microsoft Visual Studio](#).

Także wykorzystałem wtyczki "[FontAwesome.WPF](#)" dla dodania ikon jako tekstu dla nawigacji bocznej i wtyczki "[System.Data.SqlClient](#)" dla połączenia z bazą danych SQL.

2.3 Baza danych

Do stworzenia i zarządzania bazą danych wykorzystywałem [Microsoft SQL Server Management Studio](#). Na lokalnym serwerze stworzyłem bazę danych "Szpital". Także uzupełniłem bazę losowymi wartościami i dodałem kilka ograniczeń. Na przykład ograniczenie na wprowadzenie do nr_gabinetu w wizycie większego od 127.



Rysunek 2.9: Diagram bazy danych

Do pobrania danych z bazy do aplikacji wykorzystuję klasy statycznej DbContext w której są metody robiące zapytania na bazę danych i wracające wartości pobrane z bazy.

```

8 references
public static class DbContext
{
    private const string connectionStr = "Data source=DESKTOP-G7VL8PD\\SQLEXPRESS;Initial catalog=Szpital;Integrated Security=True";

    1 reference
    public static Account IdentifyUser(string username, string password)
    {
        Account? result = null;
        using (SqlConnection connection = new SqlConnection(connectionStr))
        {
            connection.Open();
            string selectAccountQuery = $"select * from Accounts where Username = '{username}' and Password_ = '{password}'";

            using (SqlCommand selectLoginCommand = new SqlCommand(selectAccountQuery, connection))
            using (SqlDataReader reader = selectLoginCommand.ExecuteReader())
            {
                while (reader.Read())
                {
                    result = MapDataToAccount(reader);
                }
            }
        }

        if (result is null)
        {
            throw new UserIdentifyException("Niepoprawne hasło lub użytkownik.");
        }

        return result;
    }
}

```

```

1 reference
private static Employee MapDataToEmployee(SqlDataReader reader)
{
    Employee employee = new Employee(
        (int)reader["Employee_id"],
        (int)reader["Department_id"],
        reader["Position"].ToString(),
        reader["Specialty"].ToString(),
        reader["First_name"].ToString(),
        reader["Last_name"].ToString(),
        reader["Pesel"].ToString(),
        reader["Phone_number"].ToString(),
        reader["Email"].ToString(),
        (DateTime)reader["Birth_date"],
        reader["Address_"].ToString(),
        reader["City"].ToString(),
        (DateTime)reader["Employment_date"],
        (decimal)reader["Salary"],
        (bool)reader["Is_deleted"]
    );

    return employee;
}

```

Rysunek 2.10: Klasa do zarządzania bazą danych(Przykładowa metoda)

2.4 Minimalne wymagania sprzętowe

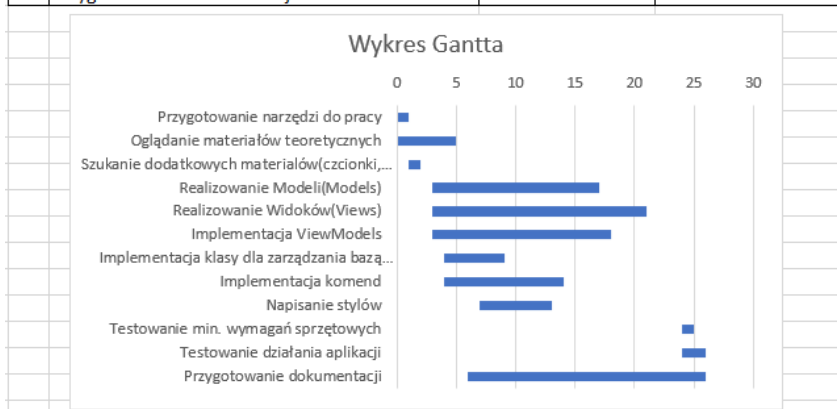
- System operacyjny: Microsoft Windows 10 lub wyżej
- Procesor: x86 lub x64 z szybkością > 800 MHz
- RAM: 512 MB
- Miejsca na dysku: 25 MB
- Zainstalowany .Net 8.0

Rozdział 3

Harmonogram realizacji projektu

3.1 Diagram Gantta

Lp.	Czynność	Czas trwania[dni]	Czas rozpoczęcia[dni]
1	Przygotowanie narzędzi do pracy	1	0
2	Oglądanie materiałów teoretycznych	5	0
3	Szukanie dodatkowych materiałów(czcionki, obrazy)	1	1
4	Realizowanie Modeli(Models)	14	3
5	Realizowanie Widoków(Views)	18	3
6	Implementacja ViewModels	15	3
7	Implementacja klasy dla zarządzania bazą danych	5	4
8	Implementacja komend	10	4
9	Napisanie stylów	6	7
10	Testowanie min. wymagań sprzętowych	1	24
11	Testowanie działania aplikacji	2	24
12	Przygotowanie dokumentacji	20	6



Rysunek 3.1: Wykres Gantta

Zacząłem pracę nad projektem z przygotowania narzędzi do pracy(ustawienia Visual Studio) i oglądania materiałów teoretycznych. Bardzo mi pomogły serie filmików na temat "Architektura MVVM w WPF" na YouTube([link](#)). Następnie próbowałem zrobić swoją aplikację na bazie otrzymanej teorii.

W końcu pozostało mi przetestować minimalne wymagania sprzętowe za pomocą wirtualnej maszyny.

3.2 Repozytorium

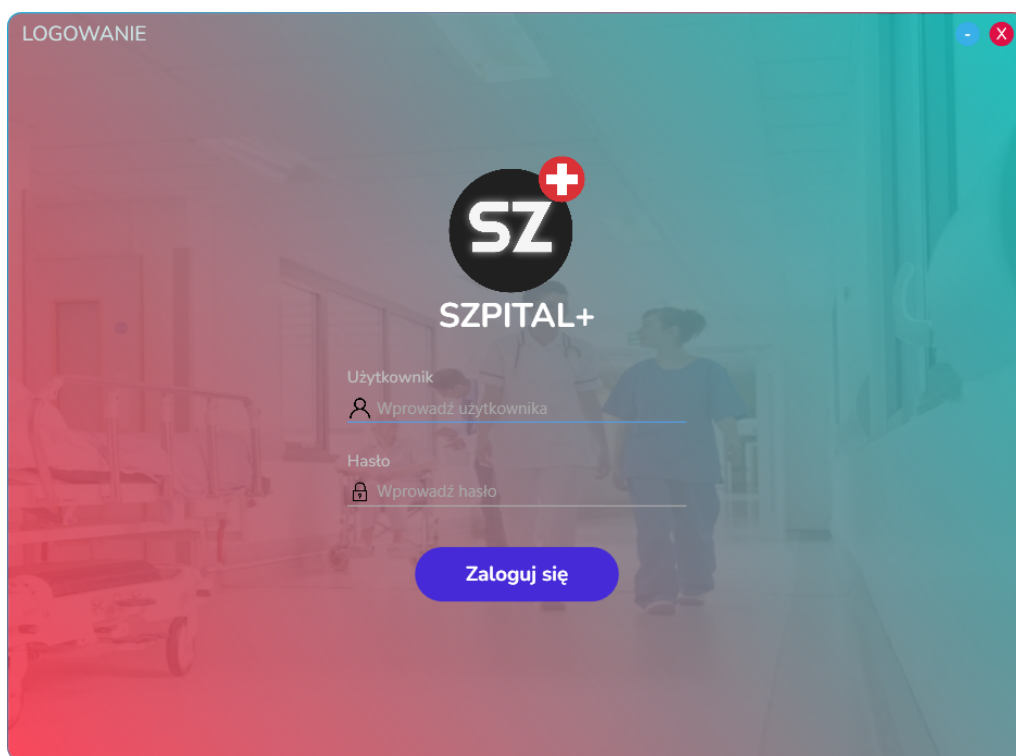
Do kontroli wersji oraz przechowywania wykorzystałem systemu Git oraz serwisu internetowego GitHub. Wszystkie pliki źródłowe projektu oraz dokumentacji są umieszczone w tym linku - [Repozytorium GitHub](#).

Rozdział 4

Prezentacja warstwy użytkowej projektu

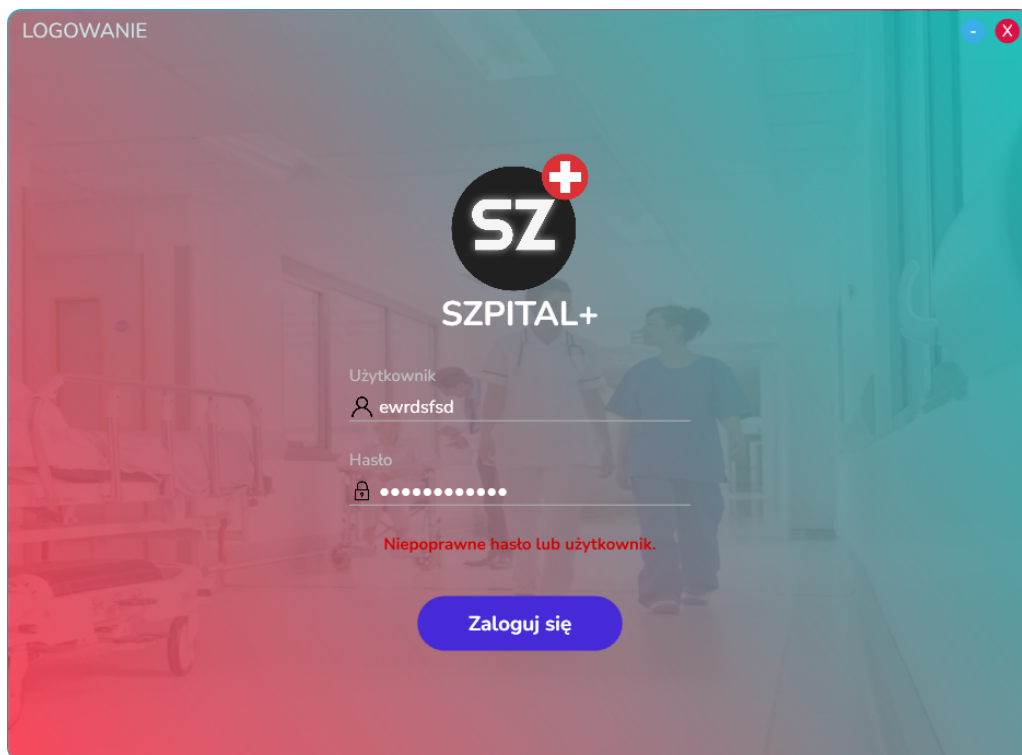
4.1 Logowanie

Uruchamiając aplikację, użytkownikowi wyświetla się okno logowania:



Rysunek 4.1: Okno logowania

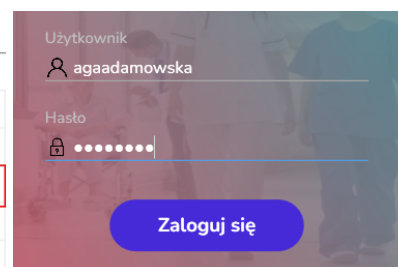
W tym oknie został przerobiony górny standardowy panel aplikacji windows. Rozmiar okna 750x550px. Także tło oraz granica zrobione z gradientu. Okno jest trochę zaokrąglone. Po wprowadzeniu danych i naciśnięciu przycisku "Zaloguj się" klasa statyczna DbContext sprawdza czy użytkownik istnieje i czy hasło jest poprawne za pomocą polecenia SQL. W przypadku gdy użytkownik poda błędne dane DbContext rzuca wyjątek "UserIdentifyException" który jest łapany w LoginCommand, skąd wypisuje się komunikat.



Rysunek 4.2: Wprowadzenie błędnych danych do logowania

Jeśli użytkownik istnieje i hasło jest poprawne —logujemy się do aplikacji.

	Employee_id	Position	Password_	Username
1	1	Główny kierownik	K0cqXxOG	wannagorna
2	2	Kierownik	3M88KBi9	beakuwalska
3	3	Doktor	egupxRTf	agaadamowska
4	4	Doktor	HGvTveW	hannowak
5	5	Doktor	08wWU7AW	izabitel



```

public class LoginCommand : CommandBase
{
    private readonly LoginWindowViewModel loginWindowViewModel;
    private Account? account;

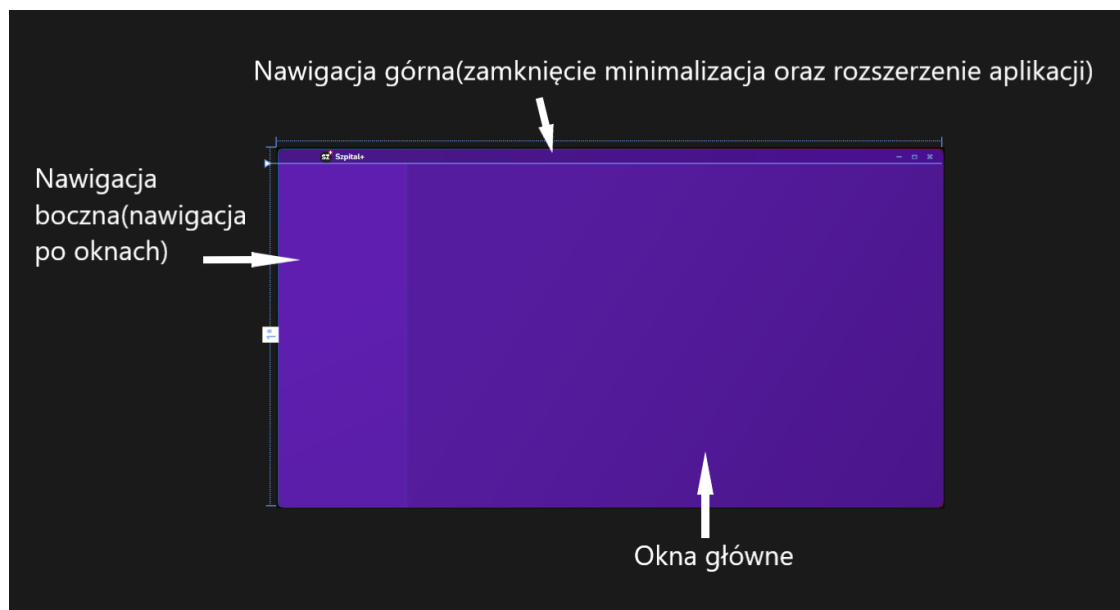
    1 reference
    public override void Execute(object? parameter)
    {
        loginWindowViewModel.ErrorMessage = string.Empty;
        try
        {
            account = DbContext.IdentifyUser(loginWindowViewModel.Username, loginWindowViewModel.Password);
            ChangeWindow(); ≤ 98ms elapsed
        }
        catch (UserIdentifyException e)
        {
            loginWindowViewModel.ErrorMessage = e.Message;
        }
        catch (Exception e)
        {
            loginWindowViewModel.ErrorMessage = e.Message;
        }
    }
}

```

Rysunek 4.3: Logowanie (DbContext nie rzucił wyjątku)

4.2 Główne okno

MainWindow(Okno główne) składa się z trzech części:



Rysunek 4.4: Główne okno

Aplikacja pobiera dane z bazy i ze względu na to, na jakim stanowisku pracuje użytkownik, wykorzystuje jeden z szablonów dotyczący jego posady żeby stworzyć nowy wygląd nawigacji bocznej.

```
1 reference
public MainViewModel(Account account)
{
    currentUserAccount = account;
    currentUserEmployee = DbContext.GetUserEmployee(currentUserAccount);

    navigationStore = new NavigationStore();

    switch (currentUserEmployee.Position)
    {
        case "Główny kierownik":
            navigationStore.CurrentMenuViewModel = new GeneralManagerMenuViewModel(navigationStore, this, currentUserAccount, currentUserEmployee);
            break;
        case "Kierownik":
            navigationStore.CurrentMenuViewModel = new ManagerMenuViewModel(navigationStore, this, currentUserAccount, currentUserEmployee);
            break;
        case "Doktor":
            navigationStore.CurrentMenuViewModel = new DoctorMenuViewModel(navigationStore, this, currentUserAccount, currentUserEmployee);
            break;
        case "Recepcjonista":
            navigationStore.CurrentMenuViewModel = new ReceptionistMenuViewModel(navigationStore, this, currentUserAccount, currentUserEmployee);
            break;
    }

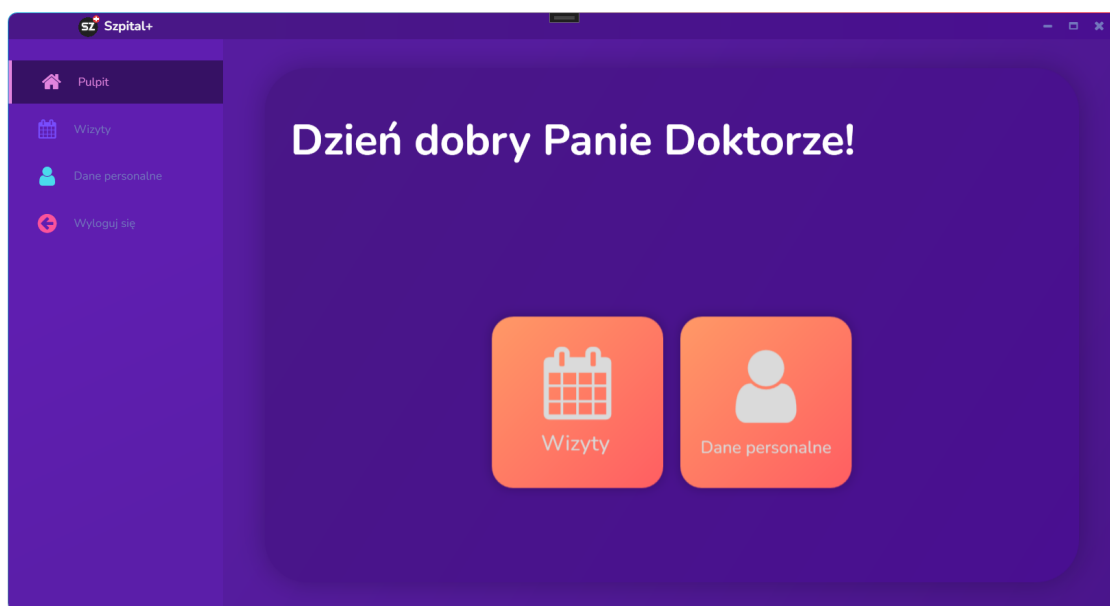
    navigationStore.CurrentMenuViewModelChanged += OnCurrentMenuViewModelChanged;
}
```

Rysunek 4.5: Pobieranie stanowiska użytkownika i przypisanie nowego wyglądu

Na przykład do aplikacji został zalogowany Recepcjonista. Wtedy wygląd jego aplikacji będzie następny (także aplikacja sprawdza jaki zwrot wykorzystać na pulpicie ze względu na płeć osoby):



Rysunek 4.6: Wygląd aplikacji recepcjonisty(Katarzyna Pabiniak)



Rysunek 4.7: Wygląd aplikacji doktora(Filip Jabcoń)

Każdy klient na panelu nawigacyjnym bocznym ma przyciski: pulpit(okno z nawigacją), dane personalne(dane o użytkowniku), wyloguj się(wylogowanie z systemu i wracanie do okna logowania).

4.3 Dane personalne

Id pracownika	59
Imię	Katarzyna
Nazwisko	Pabiniak
Dział szpitalu	Recepcjonisty
Specjalność	Recepcjonista
Pesel	97063068368
Data urodzenia	30.06.1997
Adres zamieszkania	Powstańców Śląskich 106

Pesel	97063068368
Data urodzenia	30.06.1997
Adres zamieszkania	Powstańców Śląskich 106
Miejscowość	Rzeszów
Numer telefonu	+48 662 855 223
Data zatrudnienia	14.05.2020
Wynagrodzenie (miesięcznie)	12 500,00
Email	katpabiniak@pracownik.szp.pl

Zmień hasło

Rysunek 4.8: Wygląd danych personalnych

4.4 Zmianianie hasła

Zmianianie hasła dostępne na dołu okna Dane personalne za pomocą przycisku.

Zmień hasło

Stare Hasło:

Nowe Hasło:

Zmień hasło

Hasło musi spełniać warunki:

- Długość hasła ma być nie mniej 8 i nie więcej 30 znaków.
- Hasło musi zawierać minimum jedną wielką i małą literę.
- Hasło musi zawierać minimum jedną cyfrę.

Rysunek 4.9: Wygląd okna "Zmień hasło"

Jeśli stare hasło nie będzie takim samym jak w bazie danych wystąpi błąd:



The screenshot shows a password change interface with a red background and rounded corners. It has two input fields: 'Stare Hasło:' (Old Password) and 'Nowe Hasło:' (New Password). The 'Stare Hasło' field contains a masked password '.....'. Below it, a red error message reads 'Niepoprawne stare hasło.' (Incorrect old password). The 'Nowe Hasło' field is empty. At the bottom, there is a blue button labeled 'Zmień hasło' (Change password).

Rysunek 4.10: Błąd przy wpisaniu starego hasła

4.5 Sprawdzenie nowego hasła

Sprawdzenie nowego hasła odbywa się za pomocą [wyrażenia regularnego](#)(Klasa `Regex`).

```
1 reference
private void CheckNewPassword(string? newPassword)
{
    if (newPassword is null)
    {
        throw new InvalidNewPasswordException("Nowe hasło nie zostało wpisane.");
    }
    if (newPassword.Length < 8 || newPassword.Length > 30)
    {
        throw new InvalidNewPasswordException("Niepoprawna długość nowego hasła.");
    }
    if (!new Regex(".*[A-Z].*").IsMatch(newPassword))
    {
        throw new InvalidNewPasswordException("Hasło nie zawiera jednej wielkiej litery.");
    }
    if (!new Regex(".*[a-z].*").IsMatch(newPassword))
    {
        throw new InvalidNewPasswordException("Hasło nie zawiera min. jednej małej litery.");
    }
    if (!new Regex(".*\\d.*").IsMatch(newPassword))
    {
        throw new InvalidNewPasswordException("Hasło nie zawiera min. jednej cyfry.");
    }
}
```

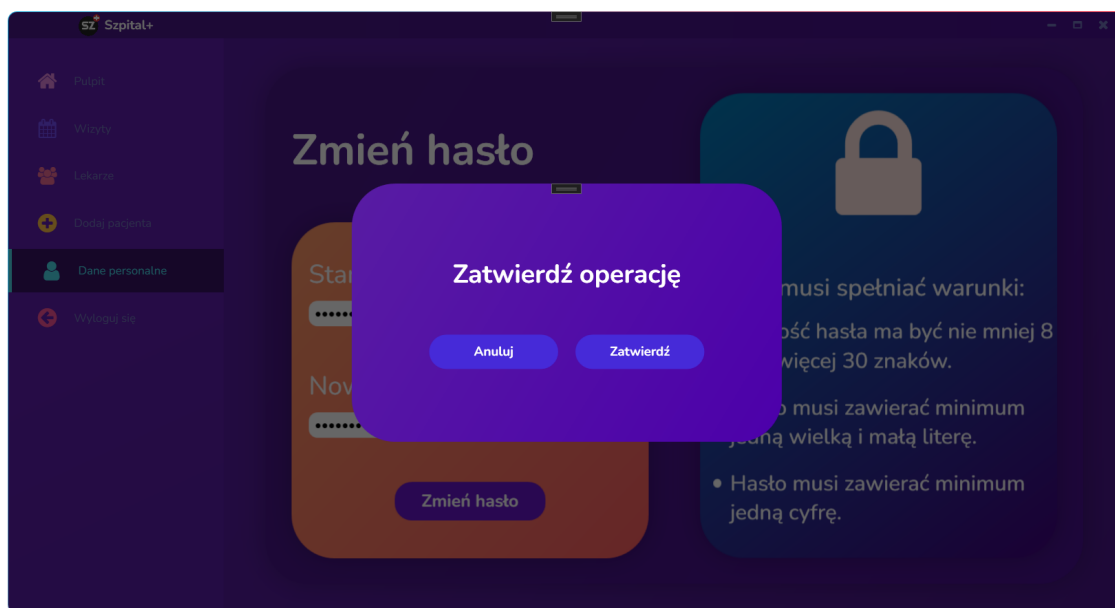
Rysunek 4.11: Metoda do sprawdzenia nowego hasła



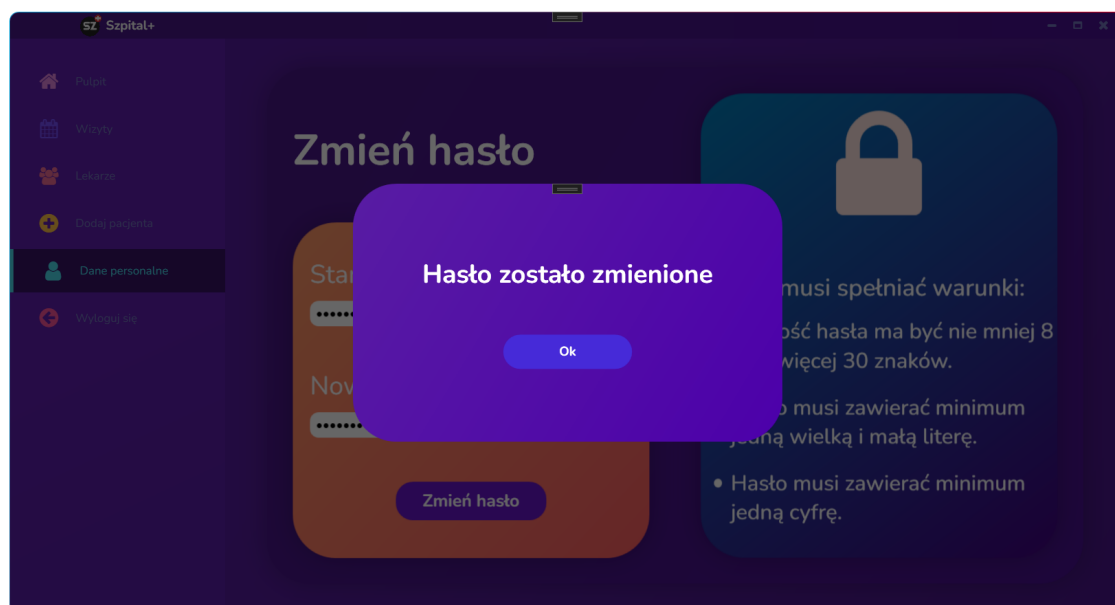
The screenshot shows the same password change interface as before. The 'Stare Hasło' field is filled with '.....'. The 'Nowe Hasło' field also contains a masked password '.....'. Below it, a red error message reads 'Hasło nie zawiera jednej wielkiej litery.' (Password does not contain one uppercase letter). The blue 'Zmień hasło' button is still at the bottom.

Rysunek 4.12: Wystąpienie błędu przy nowym hasł

Jeśli nowe hasło odpowiada wszystkim warunkom, pojawia się okno do zatwierdzenia operacji. Przy tym interakcja z głównym oknem jest niemożliwa dopóki użytkownik anuluje lub zatwierdzi operację.



Rysunek 4.13: Okno do zatwierdzenia operacji



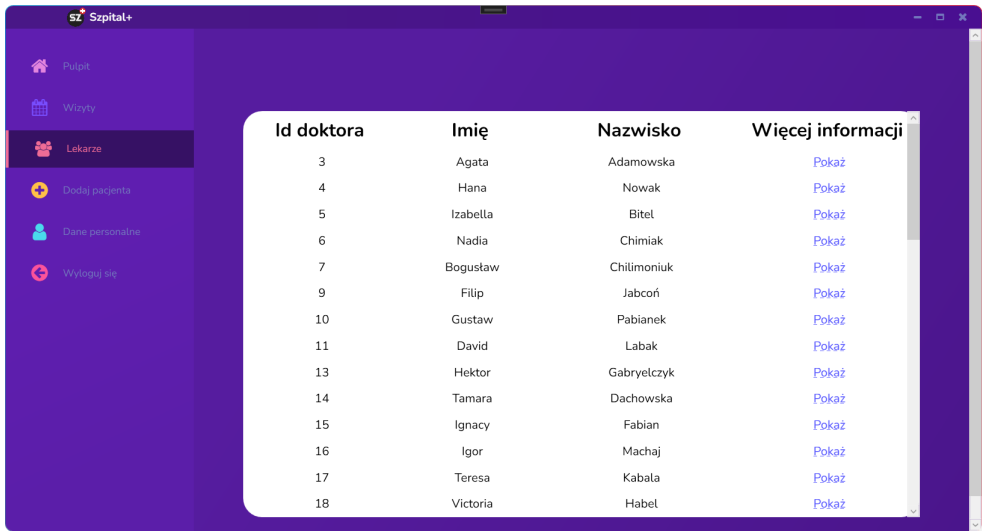
Rysunek 4.14: Hasło zostało zmienione

59	59	Jg0sSDI8	katpabiniak	59	59	123ASDzxc	katpabiniak
----	----	----------	-------------	----	----	-----------	-------------

Rysunek 4.15: Stare i zmienione hasło

4.6 Okna dla recepcjonisty

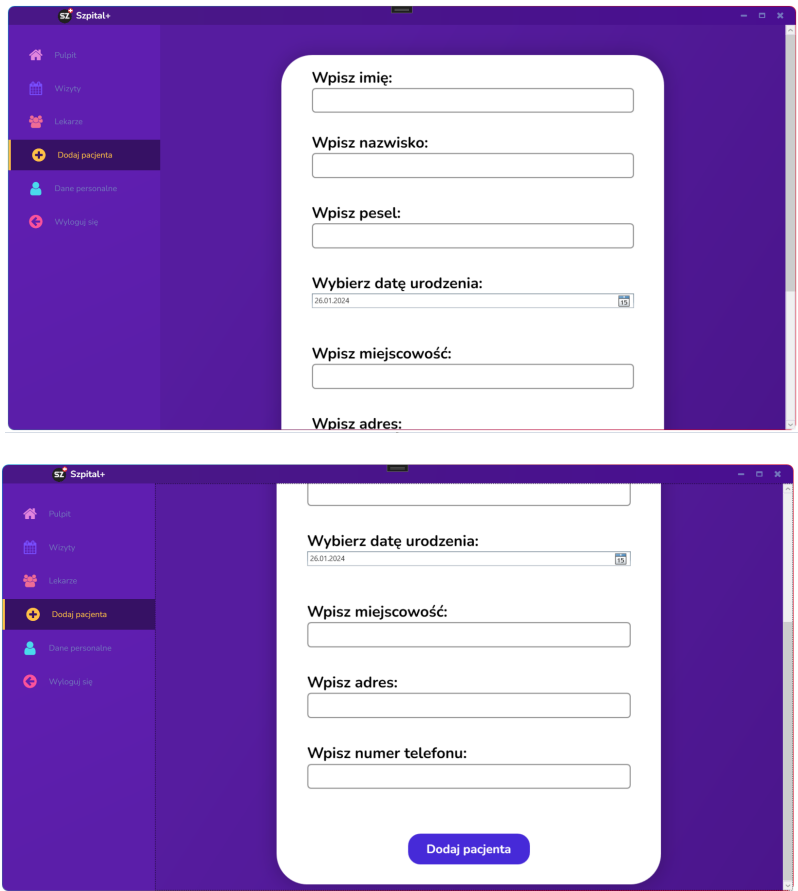
Lekarze



Id doktora	Imię	Nazwisko	Więcej informacji
3	Agata	Adamowska	Pokaż
4	Hana	Nowak	Pokaż
5	Izabella	Bitel	Pokaż
6	Nadia	Chimiak	Pokaż
7	Bogustaw	Chilimoniuk	Pokaż
9	Filip	Jabcoń	Pokaż
10	Gustaw	Pabianek	Pokaż
11	David	Labak	Pokaż
13	Hektor	Gabryelczyk	Pokaż
14	Tamara	Dachowska	Pokaż
15	Ignacy	Fabian	Pokaż
16	Igor	Machaj	Pokaż
17	Teresa	Kabala	Pokaż
18	Victoria	Habel	Pokaż

Rysunek 4.16: Okno recepcjonisty(Lekarze)

Dodaj pacjenta



Wpisz imię:

Wpisz nazwisko:

Wpisz pesel:

Wybierz datę urodzenia:

Wpisz miejscowość:

Wpisz adres:

Wybierz datę urodzenia:

Wpisz miejscowość:

Wpisz adres:

Wpisz numer telefonu:

Dodaj pacjenta

Rysunek 4.17: Okno recepcjonisty(Dodaj pacjenta)

```

1 reference
private void CheckStrings()
{
    if (FirstName is null)
    {
        throw new PatientStringNullException(nameof(FirstName));
    }
    if (LastName is null)
    {
        throw new PatientStringNullException(nameof(LastName));
    }
    if (Pesel is null)
    {
        throw new PatientStringNullException(nameof(Pesel));
    }
    if (City is null)
    {
        throw new PatientStringNullException(nameof(City));
    }
    if (Address is null)
    {
        throw new PatientStringNullException(nameof(Address));
    }
    if (PhoneNumber is null)
    {
        throw new PatientStringNullException(nameof(PhoneNumber));
    }
}

```

Rysunek 4.18: Sprawdzenie na puste pola

Rysunek 4.19: Wystąpienie błędu przy pustym błędzie

```

1 reference
private void CheckPhoneNumber()
{
    if (!new Regex("[+]?48\\s?\\d\\d\\d\\d\\s?\\d\\d\\d\\d\\s?\\d\\d\\d\\d").IsMatch(PhoneNumber))
    {
        throw new InvalidPhoneNumberException("Numer telefonu nie odpowiada wzorowi. (+48 xxx xxx xxx)");
    }
}

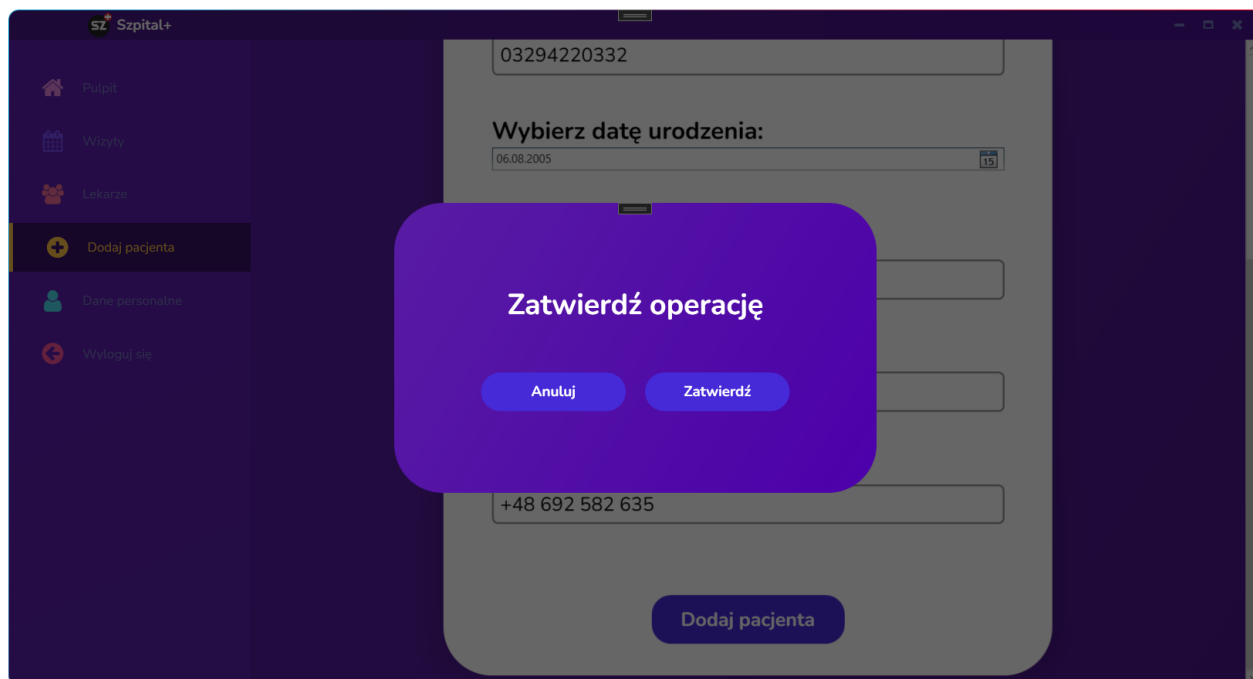
```

Rysunek 4.20: Sprawdzenie formatu telefonu

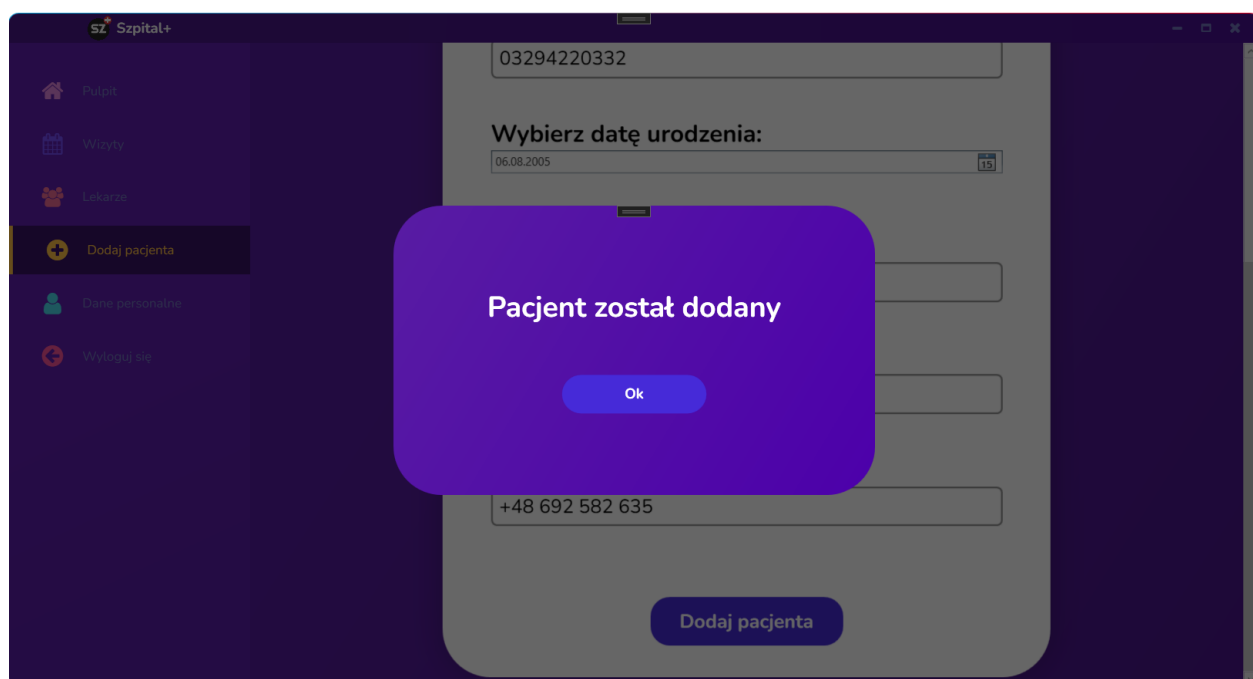
Wpisz numer telefonu:

Numer telefonu nie odpowiada wzorowi. (+48 xxx xxx xxx)

Rysunek 4.21: Błąd przy polu numeru telefonu



Rysunek 4.22: Zatwierdzenie operacji



Rysunek 4.23: Pacjent został dodany

select * from Patients

105 %

Results Messages

Patient_id	First_name	Last_name	Pesel	Birth_date	City	Address_	Phone_number
66	Kinga	Zalewska	83031234567	1983-03-12	Rzeszów	Sosnowa 35/122	+48 384 019 012
67	Rafal	Czarnecki	96042087654	1996-04-20	Rzeszów	Malinowa 36	+48 789 012 345
68	Katarzyna	Kowalik	81050598765	1981-05-05	Boguchwała	Leśna 37	+48 234 567 890
69	Kamil	Mazurek	94070123456	1994-07-01	Krasne	Wiosenna 38	+48 901 234 567
70	Dominika	Kaczorowska	78081834567	1978-08-18	Rzeszów	Polna 39	+48 345 678 901
71	Paweł	Dębski	90092587654	1990-09-25	Krasne	Brzozowa 40/13	+48 678 901 234
72	Alicja	Lipińska	95010798765	1995-01-07	Boguchwała	Górska 41	+48 432 109 876
73	Lukasz	Piotrowski	77040623456	1977-04-06	Tyczyn	Malinowa 42	+48 789 371 634
74	Natalia	Kowalczyk	93051234567	1993-05-12	Krasne	Jesienna 43/23	+48 456 987 012
75	Krzysztof	Sikora	85062187654	1985-06-21	Rzeszów	Dębowa 44/123	+48 257 623 322
76	Martyna	Witkowska	91072798765	1991-07-27	Rzeszów	Lipowa 45	+48 216 937 654
77	Damian	Nowak	96083023456	1996-08-30	Boguchwała	Słoneczna 46/99	+48 789 012 345
78	Dominika	Duda	80020134567	1980-02-01	Rzeszów	Miodowa 47	+48 675 456 219
79	Bartosz	Zieliński	94030487654	1994-03-04	Tyczyn	Wiosenna 48	+48 901 234 567
80	Katarzyna	Cieślak	88050998765	1988-05-09	Rzeszów	Leśna 43/1	+48 678 901 234
81	Tomasz	Szymański	92061623456	1992-06-16	Krasne	Jesienna 50	+48 939 238 047

select * from Patients

105 %

Results Messages

Patient_id	First_name	Last_name	Pesel	Birth_date	City	Address_	Phone_number
67	Rafal	Czarnecki	96042087654	1996-04-20	Rzeszów	Malinowa 36	+48 789 012 345
68	Katarzyna	Kowalik	81050598765	1981-05-05	Boguchwała	Leśna 37	+48 234 567 890
69	Kamil	Mazurek	94070123456	1994-07-01	Krasne	Wiosenna 38	+48 901 234 567
70	Dominika	Kaczorowska	78081834567	1978-08-18	Rzeszów	Polna 39	+48 345 678 901
71	Paweł	Dębski	90092587654	1990-09-25	Krasne	Brzozowa 40/13	+48 678 901 234
72	Alicja	Lipińska	95010798765	1995-01-07	Boguchwała	Górska 41	+48 432 109 876
73	Lukasz	Piotrowski	77040623456	1977-04-06	Tyczyn	Malinowa 42	+48 789 371 634
74	Natalia	Kowalczyk	93051234567	1993-05-12	Krasne	Jesienna 43/23	+48 456 987 012
75	Krzysztof	Sikora	85062187654	1985-06-21	Rzeszów	Dębowa 44/123	+48 257 623 322
76	Martyna	Witkowska	91072798765	1991-07-27	Rzeszów	Lipowa 45	+48 216 937 654
77	Damian	Nowak	96083023456	1996-08-30	Boguchwała	Słoneczna 46/99	+48 789 012 345
78	Dominika	Duda	80020134567	1980-02-01	Rzeszów	Miodowa 47	+48 675 456 219
79	Bartosz	Zieliński	94030487654	1994-03-04	Tyczyn	Wiosenna 48	+48 901 234 567
80	Katarzyna	Cieślak	88050998765	1988-05-09	Rzeszów	Leśna 43/1	+48 678 901 234
81	Tomasz	Szymański	92061623456	1992-06-16	Krasne	Jesienna 50	+48 939 238 047
82	Oleksandr	Lochenko	03294220332	2005-08-06	Rzeszów	Mikołajczyka 12/93	+48 692 582 635

Rysunek 4.24: Tabela pacjentów przed i po dodaniu rekordu

4.7 Okna dla głównego kierownika

Pracownicy

sz Szpital+

Pulpit

Pracownicy

Dane personalne

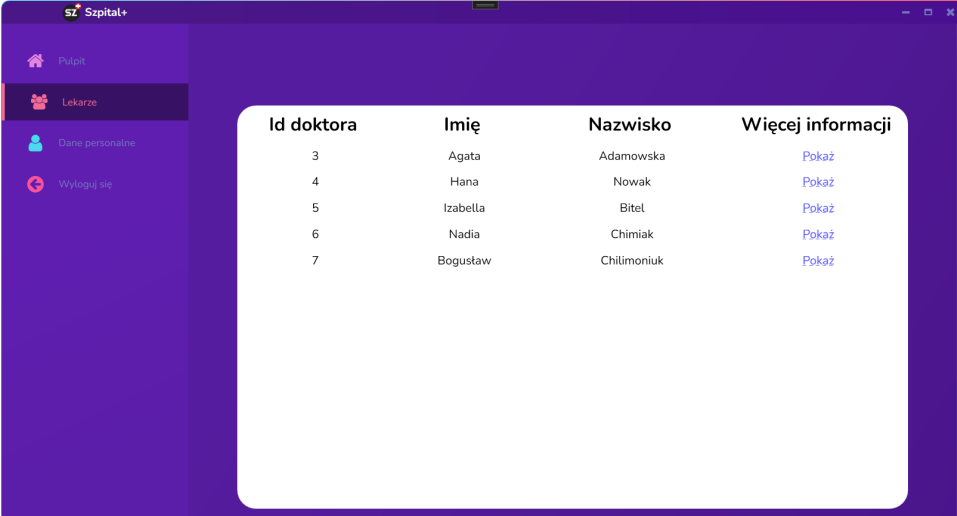
Wyloguj się

Id pracownika	Imię	Nazwisko	Więcej informacji
2	Beata	Kuwałska	Pokaż
3	Agata	Adamowska	Pokaż
4	Hana	Nowak	Pokaż
5	Izabella	Bitel	Pokaż
6	Nadia	Chimiak	Pokaż
7	Bogustaw	Chilimoniuk	Pokaż
8	Daria	Pabich	Pokaż
9	Filip	Jabcoń	Pokaż
10	Gustaw	Pabianek	Pokaż
11	David	Labak	Pokaż
12	Konrad	Ulczyk	Pokaż
13	Hektor	Gabryelczyk	Pokaż

Rysunek 4.25: Okno głównego kierownika(Pracownicy)

4.8 Okna dla kierownika

Lekarze



Id doktora	Imię	Nazwisko	Więcej informacji
3	Agata	Adamowska	Pokaż
4	Hana	Nowak	Pokaż
5	Izabella	Bitel	Pokaż
6	Nadia	Chimiak	Pokaż
7	Bogustaw	Chilimoniuk	Pokaż

Rysunek 4.26: Okno kierownika(Lekarze)

Rozdział 5

Podsumowanie

Podczas robienia projektu nauczyłem się planowania harmonogramu projektu(3), implementacji wzoru MVVM(2.2), pisania interfejsu użytkownika(2.2), pracy w środowisku Visual Studio(2.2) i programowania w języku C#(2.2).

Dalsza praca będzie polegała na zrealizowaniu wszystkich funkcjonalności których nie zdążyłem zaimplementować, poprawienie i dopasowanie widoków, dodanie animacji oraz ulepszenie wydajności aplikacji.

Bibliografia

- [1] https://youtu.be/fZxZswmC_BY?si=gn6SI5tqut9n-vOM
- [2] https://youtu.be/oSeYvMEH7jc?si=aWGmXG_Enb7S_z3j
- [3] <https://youtu.be/PzP8mw7JUzI?si=twPT-bTfhYE-FMau>
- [4] <https://stackoverflow.com/questions/833943/watermark-hint-placeholder-text-in-textbox>
- [5] <https://leanactionplan.pl/wykres-gantt/>
- [6] https://pl.wikipedia.org/wiki/Windows_Presentation_Foundation
- [7] https://pl.wikipedia.org/wiki/.NET_Framework
- [8] https://pl.wikipedia.org/wiki/Extensible_Application_Markup_Language
- [9] <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93viewmodel>
- [10] <https://stackoverflow.com/questions/58272836/display-datetime-in-textblock-with-dashed-underlined-textdecorations>
- [11] <https://stackoverflow.com/questions/30231252/disable-main-window-wpf>
- [12] <https://stackoverflow.com/questions/26258450/how-do-i-add-a-bullet-point-in-front-of-a-text-binding-in-wpf>
- [13] <https://learn.microsoft.com/en-us/archive/msdn-magazine/2009/february/patterns-wpf-apps-with-the-model-view-viewmodel-design-pattern>
- [14] Rob Miles, *C# Programming Yellow Book*, 2019.

Spis rysunków

2.1	Klasy główne	6
2.2	Klasy nawigacyjne dla doktora	7
2.3	Klasy nawigacyjne dla głównego kierownika	7
2.4	Klasy nawigacyjne dla kierownika	8
2.5	Klasy nawigacyjne dla recepcjonisty	8
2.6	Klasy nawigacyjne wspólne dla wszystkich klientów . .	9
2.7	Klasy komend	9
2.8	Klasy dla bazy danych i przechowywania nawigacji . . .	10
2.9	Diagram bazy danych	11
2.10	Klasa do zarządzania bazą danych(Przykładowa metoda)	12
3.1	Wykres Gantta	13
4.1	Okno logowania	14
4.2	Wprowadzenie błędnych danych do logowania	15
4.3	Logowanie (DbContext nie rzucił wyjątku)	15
4.4	Główne okno	16
4.5	Pobieranie stanowiska użytkownika i przypisanie nowego wyglądu	16
4.6	Wygląd aplikacji recepcjonisty(Katarzyna Pabiniak) . . .	17
4.7	Wygląd aplikacji doktora(Filip Jabcoń)	17
4.8	Wygląd danych personalnych	18
4.9	Wygląd okna "Zmień hasło"	18
4.10	Błąd przy wpisaniu starego hasła	19
4.11	Metoda do sprawdzenia nowego hasła	19
4.12	Wystąpienie błędu przy nowym hasłem	19
4.13	Okno do zatwierdzenia operacji	20
4.14	Hasło zostało zmienione	20
4.15	Stare i zmienione hasło	20

4.16	Okno recepcjonisty(Lekarze)	21
4.17	Okno recepcjonisty(Dodaj pacjenta)	21
4.18	Sprawdzenie na puste pola	22
4.19	Wystąpienie błędu przy pustym błędzie	22
4.20	Sprawdzenie formatu telefonu	22
4.21	Błąd przy polu numeru telefonu	22
4.22	Zatwierdzenie operacji	23
4.23	Pacjent został dodany	23
4.24	Tabela pacjentów przed i po dodaniu rekordu	24
4.25	Okno głównego kierownika(Pracownicy)	24
4.26	Okno kierownika(Lekarze)	25