Содержание

[Введение 4](#_Toc182256639)

[1 Теоретическая часть 5](#_Toc182256640)

[1.1 Постановка задачи 5](#_Toc182256641)

[1.2 Выбор программ для реализации программного продукта 5](#_Toc182256642)

[2 Практическая часть 8](#_Toc182256643)

[2.1 Структура программного продукта 8](#_Toc182256644)

[2.2 Реализация программного продукта 9](#_Toc182256645)

[Заключение 25](#_Toc182256646)

[Список использованных источников 26](#_Toc182256647)

# **Введение**

Для современного человека очень важно получать достойное обслуживание в сфере здоровья. Для этого огранизации постоянно улучшают свои возможности. Эффективное управление информацией в поликлинике играет решающую роль в обеспечении высокого уровня качества медицинского обслуживания и оптимизации рабочих процессов. Создание базы данных является обязательным шагом для достижения этих целей.

Управление информацией в больнице включает в себя сбор, хранение, обработку и предоставление доступа к различным видам медицинской информации. Это включает в себя данные о пациентах, их истории болезни, назначенные лекарства, проведенные операции, а также о соотрудниках.

Предметом исследования данной курсовой работы является процесс разработки программного обеспечения для редактирования базы данных больница.

Объектом исследования выступает база данных больницы и процессы взаимодействия с ней через разрабатываемое программное обеспечение.

Цель курсового проекта: разработка программных модулей для работы с базой данных «Больница».

Задачи курсового проекта:

1. Определение функционала программного продукта.
2. Выбор программ для реализации программного продукта.
3. Описание структуры программного продукта.
4. Реализация программного продукта.
5. **Теоретическая часть**

**1.1 Постановка задачи**

Разрабатываемый продукт представляет собой настольное приложение «Больница» и предназначен для управления процессами логистики в организации. Он предоставляет возможность просмотра списка медицинского персонала, а также выполнения операций по добавлению, изменению и удалению данных о врачах и других сотрудниках.

Приложение состоит из нескольких модулей:

* Просмотр списка врачей: отображение информации о врачах.
* Удаление данных: возможность удаления записей о врачах, которые не имеют записи на операцию из базы данных.
* Изменение данных: редактирование информации о врачах.
* Добавление данных: ввод информации о новых врачах и сохранение их в системе.

**1.2 Выбор программ для реализации программного продукта**

Для реализации программного обеспечения **«**Больница**»** было выбрано несколько инструментов, обеспечивающих эффективное создание настольного приложения под WPF (Windows Presentation Foundation). Это решение обусловлено тем, что WPF предоставляет мощные средства для разработки современных графических интерфейсов на платформе .NET Framework.

Минимальные технические требования для Visual Studio 2022:

* Процессор: 1,8 ГГц или быстрее (рекомендуется 2 ГГц и выше).
* Оперативная память: 2 ГБ (рекомендуется 8 ГБ).
* Графический процессор: Поддержка DirectX 9 с WDDM 1.0 или выше.
* Операционная система: Windows 7 с пакетом обновления 1 (SP1), Windows 8.1, Windows 10 версии 1703 или новее (64-разрядная или 32-разрядная версия).
* Свободное место на жестком диске: 800 МБ для базовой установки; дополнительное пространство потребуется для установки компонентов и хранения проектов.

Язык программирования и платформа.

Приложение написано на языке программирования C#, используя платформу .NET 6.0. Эта комбинация обеспечивает высокую производительность, кросс-платформенность и богатый набор встроенных библиотек для разработки современного пользовательского интерфейса и бизнес-логики.

Пользовательский интерфейс.

Для создания графического интерфейса используется технология WPF (Windows Presentation Foundation), которая предоставляет мощные средства для построения сложных и динамичных пользовательских интерфейсов. WPF поддерживает стилизацию элементов, анимацию и работу с различными типами данных.

База данных.

Для хранения данных о клиентах и билетах используется Microsoft SQL Server, управляемая через SQL Server Management Studio (SSMS) 18. Это надежное и масштабируемое решение для управления большими объемами данных, обеспечивающее высокую производительность и безопасность.

Минимальные требования для SQL Server Management Studio 18:

* Операционная система: Windows 10, Windows Server 2016 и выше
* Процессор: x86 или x64 процессор с тактовой частотой 1.8 GHz или выше
* Оперативная память: 2 GB RAM (рекомендуется 4 GB или больше)
* Место на жестком диске: 1 GB свободного пространства для установки
* .NET Framework: Требуется установленная версия .NET Framework 4.7.2 или выше

Итоговые минимальные системные требования для запуска приложения:

* Процессор: Intel Core i3 или аналог
* Оперативная память: 4 ГБ
* Жесткий диск: 500 МБ свободного места
* Операционная система: Windows 10 (64-bit)

Применение SQL Server совместно с Visual Studio 2019 и технологией WPF дало возможность разработать современное и эффективное приложение для управления данными сотрудников больницы, соответствующее строгим требованиям безопасности и надежности.

**2 Практическая часть**

**2.1 Структура программного продукта**

С учетом данных, полученных при изучении специфики работы процессов, а также установленных требований к функционалу.

Разработана модульная структура программного продукта, представленная в соответствии с рисунком 1.

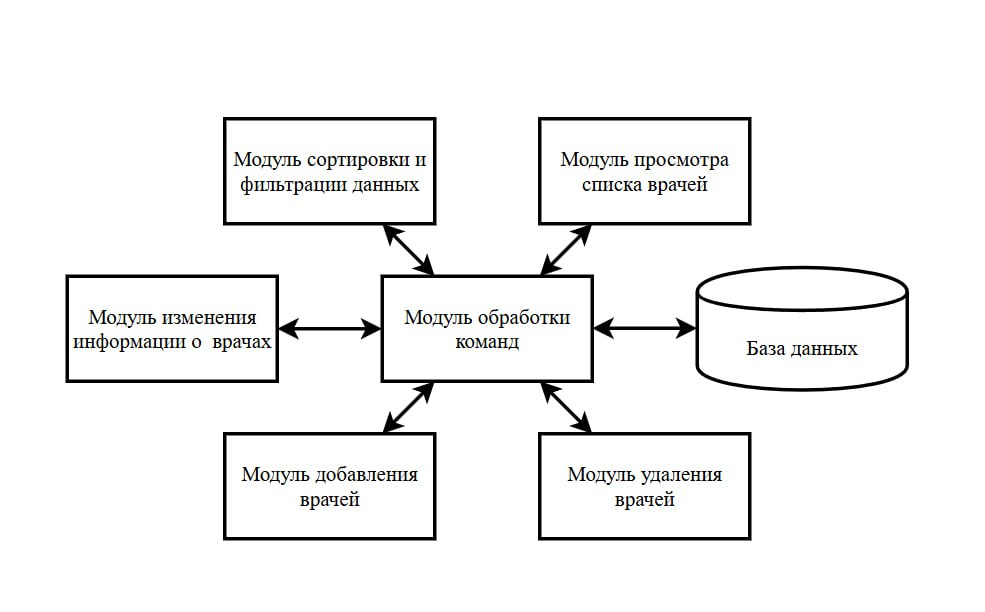


Рисунок 1 – Модульная структура приложения

В структуре разрабатываемого приложения выделены следующие модули:

1. Просмотра списка врачей – модуль, предназначенный для просмотра списка врачей.
2. Модуль обработки команд – модуль, решающий как должна отреагировать команда на отправленную команду пользователем или другим модулем.
3. Модуль редактирования информации – модуль, отвечающий за редактирование информации о врачах.
4. Модуль добавления нового врача – модуль, отвечающий за добавление новых врачей.
5. Модуль удаления врачей – модуль, отвечающий за удаление врачей.
6. Модуль сортировки и фильтрации данных– модуль, отвечающий за возможность фильтровку и сортировки данных.
7. Модуль работы с базой данных – модуль, отвечающий за хранение информации о врачах.

Схема базы данных представлена в соответствии с рисунком 2.

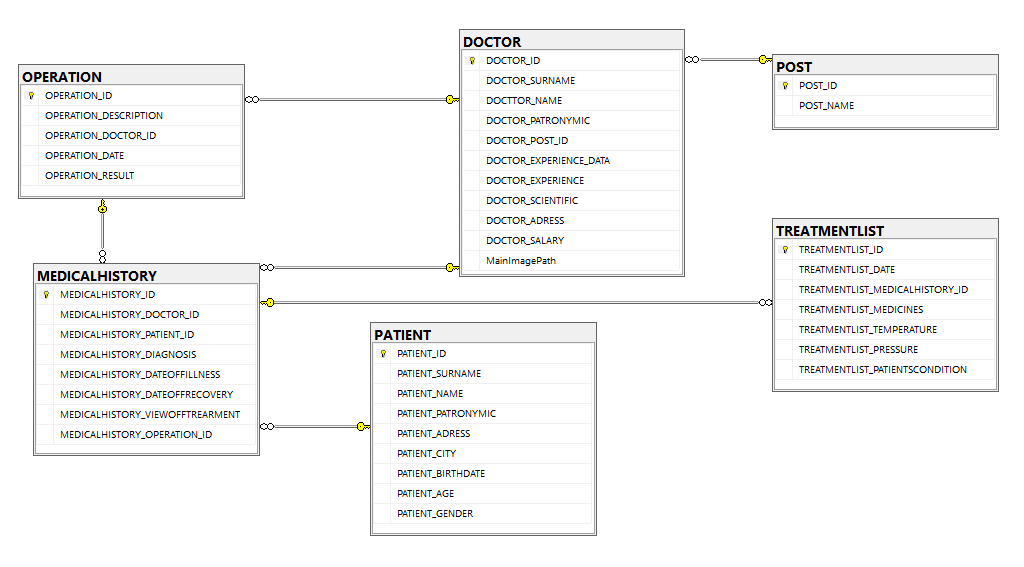


Рисунок 2 – Схема базы данных

Неотъемлемой частью приложения является его база данных, которая состоит из девяти таблиц: «Operation», «Doctor», «Post», «MedicalHistory», «Patient», «TreatmentList».

**2.2 Реализация программного продукта**

После запуска приложения открывается страница «Врачи», которая отображает список сотрудников. На этой странице представлены данные о врачах в соответствии с рисунком 3.

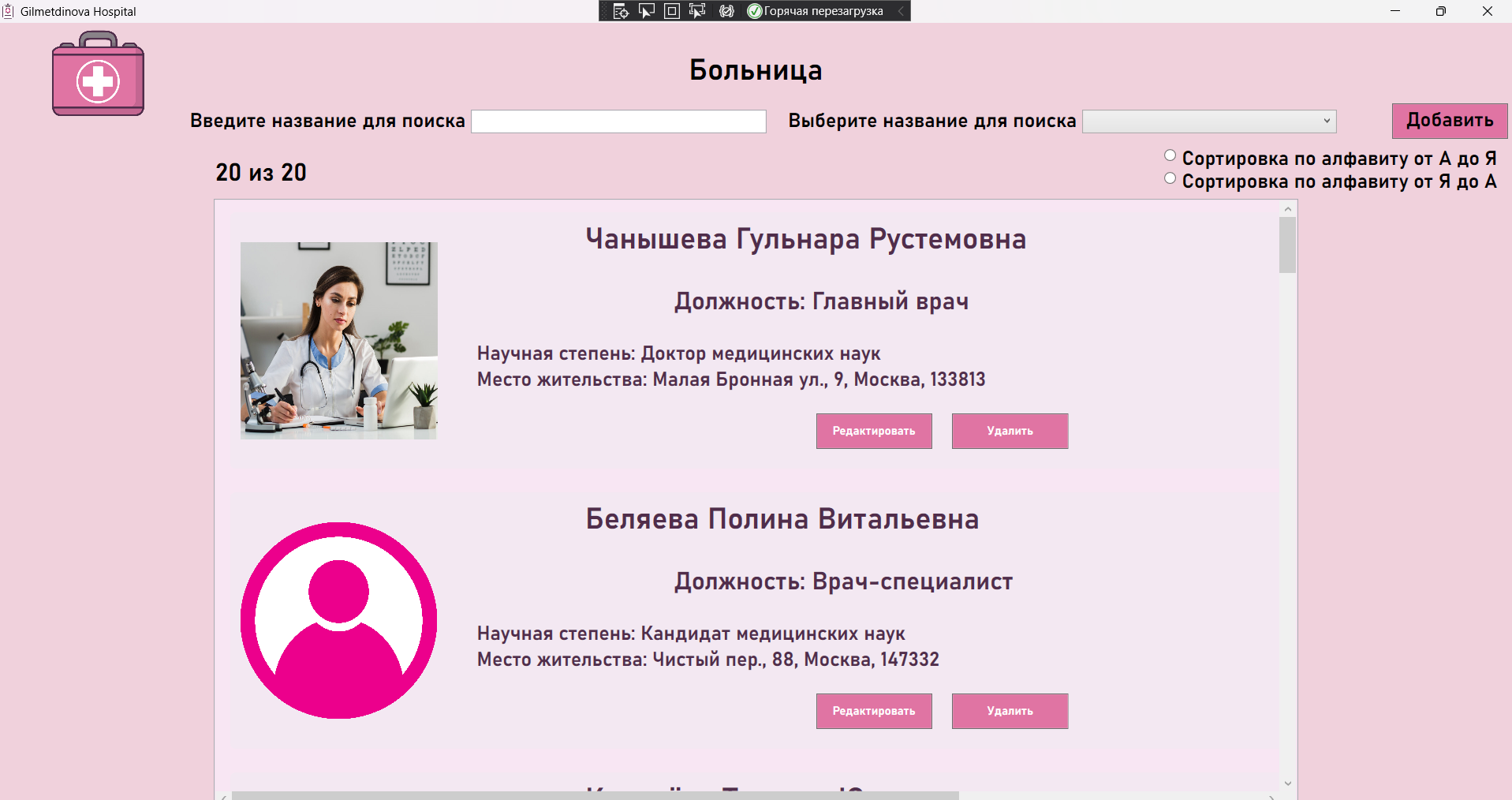


Рисунок 3 – Главное окно

Список сотрудников представляет собой следующую информацию:

* ФИО сотрудника.
* Должность.
* Научную степень.
* Место жительства.

XAML-разметка страницы «MainWindow»:

<Window x:Class="GilmetdinovaHospital.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:GilmetdinovaHospital"

mc:Ignorable="d"

Title="Gilmetdinova Hospital" Height="450" Width="800" Icon="res/icon.ico" FontFamily="Bahnschrift SemiBold">

<Grid Background="#FFF0D1DC">

<Grid.RowDefinitions>

<RowDefinition Height="103.2"/>

<RowDefinition/>

<RowDefinition Height="30"/>

</Grid.RowDefinitions>

<Image Source="res\logo.png" HorizontalAlignment="Left" Margin="52,0,0,1.2" Width="98" RenderTransformOrigin="0.545,0.48"/>

<TextBlock

Text="Больница" FontSize="30" HorizontalAlignment="Center" VerticalAlignment="Center" Height="36" Margin="329,31,328.6,36.2" Width="136">

</TextBlock>

<Button Content="Назад" Name="BntBack" HorizontalAlignment="Right" Click="BntBack\_Click" Margin="0,10,9.6,7.2" Width="142" FontSize="25" Background="#FFDD528F"></Button>

<Frame Name="MainFrame" ContentRendered="MainFrame\_ContentRendered" NavigationUIVisibility="Hidden" Grid.RowSpan="3" Margin="0,78,-0.4,-5"/>

</Grid>

</Window>

Код функционала страницы «MainWindow»:

using System;

using System.Windows;

namespace GilmetdinovaHospital

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

MainFrame.Navigate(new DoctorPage());

Manager.MainFrame = MainFrame;

}

private void BntBack\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.GoBack();

}

private void MainFrame\_ContentRendered(object sender, EventArgs e)

{

if (MainFrame.CanGoBack)

{

BntBack.Visibility = Visibility.Visible;

}

else

{

BntBack.Visibility = Visibility.Hidden;

}}}}

XAML-разметка страницы «Provider »:

<Page x:Class="GilmetdinovaHospital.DoctorPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:GilmetdinovaHospital"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="DoctorPage" FontFamily="Bahnschrift SemiBold">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="auto"></RowDefinition>

<RowDefinition Height="auto"></RowDefinition>

<RowDefinition Height="\*"></RowDefinition>

<RowDefinition Height="auto"></RowDefinition>

</Grid.RowDefinitions>

<StackPanel Orientation="Horizontal" Margin="220,5,0,5" HorizontalAlignment="Stretch" Grid.Row="1" >

<TextBlock FontSize="25" Height="25" Name="ProductCountTextBlock"></TextBlock>

<TextBlock FontSize="25" Height="25" Text=" из "></TextBlock>

<TextBlock FontSize="25" Height="25" Name="ProductMaxCountTextBlock"></TextBlock>

</StackPanel>

<WrapPanel Orientation="Horizontal" HorizontalAlignment="Center">

<StackPanel Margin="10" Orientation="Horizontal">

<TextBlock Text="Введите название для поиска " TextAlignment="Right"

Width="300" FontSize="20"></TextBlock>

<TextBox Name="TBoxSearch" TextChanged="TBoxSearch\_TextChanged"

Width="300" FontSize="15"></TextBox>

</StackPanel>

<StackPanel Orientation="Horizontal" Margin="10">

<TextBlock Text="Выберите название для поиска " TextAlignment="Right"

Width="300" FontSize="20" ></TextBlock>

<ComboBox x:Name="ComboType"

FontSize="15" Width="258" SelectionChanged="ComboType\_SelectionChanged">

<TextBlock><Run Text="Все"/></TextBlock>

<TextBlock><Run Text="Главный врач"/></TextBlock>

<TextBlock><Run Text="Младший медицинский персонал"/></TextBlock>

<TextBlock><Run Text="Заведующий отделением"/></TextBlock>

<TextBlock><Run Text="Врач-специалист"/></TextBlock>

<TextBlock><Run Text="Прочий персонал"/></TextBlock>

<TextBlock><Run Text="Технический персонал"/></TextBlock>

<TextBlock><Run Text="Лаборант"/></TextBlock>

<TextBlock><Run Text="Медицинский регистратор"/></TextBlock>

<TextBlock><Run Text="Медсестра общего профиля"/></TextBlock>

<TextBlock><Run Text="Социальный работник"/></TextBlock>

</ComboBox>

</StackPanel>

</WrapPanel>

<StackPanel Grid.Row="1" Grid.Column="0" Orientation="Vertical" Margin="5" HorizontalAlignment="Right">

<RadioButton Content="Сортировка по алфавиту от А до Я" GroupName="Sort"

Width="350" FontSize="20" Name="RButtonUp" Checked="RButtonUp\_Checked"

></RadioButton>

<RadioButton Content="Сортировка по алфавиту от Я до А" GroupName="Sort"

Width="350" FontSize="20" Name="RButtonDown" Checked="RButtonDown\_Checked"

></RadioButton>

</StackPanel>

<Button Content="Добавить" Name="AddButton"

HorizontalAlignment="Right" VerticalAlignment="Bottom" Margin="0,0,5,4.8" Height="36" Width="118" Background="#FFE074A3" FontSize="20" Click="AddButton\_Click" ></Button>

<TextBlock Name="TBCount" FontSize="30" HorizontalAlignment="Right" VerticalAlignment="Bottom"></TextBlock>

<ListView Name="ServiceListView" Width="1100" Grid.Row="2" SelectionChanged="ServiceListView\_SelectionChanged" IsVisibleChanged="ServiceListView\_IsVisibleChanged" Background="#FFF8E6F3">

<ListView.ItemTemplate >

<DataTemplate>

<Border Width="1500" Background="#FFF3E8F2" Margin="10" Padding="10" CornerRadius="5">

<!-- Обертка Border для закругления углов -->

<Grid>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="200"></ColumnDefinition>

<ColumnDefinition Width="\*"></ColumnDefinition>

<ColumnDefinition Width="\*"></ColumnDefinition>

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition Height="40"/>

<RowDefinition Height="30"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

<RowDefinition Height="\*"/>

</Grid.RowDefinitions>

<Image Width="200" Height="200" Grid.Column="0" Stretch="UniformToFill" Grid.RowSpan="5" HorizontalAlignment="Center">

<Image.Source>

<Binding Path="MainImagePath">

<Binding.TargetNullValue>

<ImageSource>

res/zaglushka.png

</ImageSource>

</Binding.TargetNullValue>

</Binding>

</Image.Source>

</Image>

<WrapPanel Grid.Column="1">

<TextBlock Text="{Binding DOCTOR\_SURNAME}" FontWeight="Bold" Margin="150 0 0 0" HorizontalAlignment="Left" FontSize="30" TextWrapping="Wrap" Foreground="#FF502D4B"/>

<TextBlock Text=" "></TextBlock>

<TextBlock Text="{Binding DOCTTOR\_NAME}" FontWeight="Bold" Margin="5 0 0 0" HorizontalAlignment="Left" FontSize="30" TextWrapping="Wrap" Foreground="#FF502D4B"/>

<TextBlock Text=" "></TextBlock>

<TextBlock Text="{Binding DOCTOR\_PATRONYMIC}" FontWeight="Bold" Margin="5 0 0 0" HorizontalAlignment="Left" FontSize="30" TextWrapping="Wrap" Foreground="#FF502D4B"/>

</WrapPanel>

<WrapPanel Grid.Column="1" Grid.Row="1" Margin="240 0 0 0">

<TextBlock Text="Должность: " HorizontalAlignment="Left" FontSize="25" FontWeight="Bold" Foreground="#FF502D4B"></TextBlock>

<TextBlock Text="{Binding POST\_NAME\_STR}" HorizontalAlignment="Left" FontSize="25" Foreground="#FF502D4B"></TextBlock>

</WrapPanel>

<WrapPanel Grid.Column="1" Grid.Row="2" HorizontalAlignment="Left" Margin="40 0 0 0">

<TextBlock Text="Научная степень: " HorizontalAlignment="Left" FontSize="20" FontWeight="Bold" Foreground="#FF502D4B"></TextBlock>

<TextBlock Text="{Binding DOCTOR\_SCIENTIFIC}" Grid.Column="1" Grid.Row="2" HorizontalAlignment="Left" FontSize="20" Foreground="#FF502D4B"></TextBlock>

</WrapPanel>

<WrapPanel Grid.Column="1" Grid.Row="3" Margin="40 0 0 0">

<TextBlock Text="Место жительства: " HorizontalAlignment="Left" FontSize="20" FontWeight="Bold" Foreground="#FF502D4B"></TextBlock>

<TextBlock Text="{Binding DOCTOR\_ADRESS}" Grid.Column="1" Grid.Row="3" HorizontalAlignment="Left" FontSize="20" Foreground="#FF502D4B"></TextBlock>

</WrapPanel>

<StackPanel Grid.Column="1" Grid.Row="4" Orientation="Horizontal">

<TextBlock Text=" "></TextBlock>

</StackPanel>

<StackPanel Orientation="Horizontal" Grid.Row="8" Grid.ColumnSpan="2" HorizontalAlignment="Right" Margin="0,10,0,0">

<Button Content="Редактировать" Name="editDoctor" Click="editDoctor\_Click" Height="36" Width="118" Background="#FFE074A3" Margin="20000,10,0,10" Foreground="White"></Button>

<Button Content="Удалить" Name="deleteDoctor" Click="deleteDoctor\_Click" Height="36" Width="118" Background="#FFE074A3" Margin="20,10,0,10" Foreground="White"></Button>

</StackPanel>

</Grid>

</Border>

<!-- Закрытие Border -->

</DataTemplate>

</ListView.ItemTemplate>

</ListView>

</Grid>

</Page>

Код функционала страницы «ProviderPage»:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace GilmetdinovaHospital

{

/// <summary>

/// Логика взаимодействия для DoctorPage.xaml

/// </summary>

public partial class DoctorPage : Page

{

public DoctorPage()

{

InitializeComponent();

var currentServices = GilmetdinovaHospitalEntities.GetContext().DOCTOR.ToList();

ServiceListView.ItemsSource = currentServices;

int ProductCount = 0;

foreach (DOCTOR product in currentServices)

{

ProductCount++;

}

ProductMaxCountTextBlock.Text = ProductCount.ToString();

}

private void RButtonUp\_Checked(object sender, RoutedEventArgs e)

{

UpdateServices();

}

private void RButtonDown\_Checked(object sender, RoutedEventArgs e)

{

UpdateServices();

}

private void ComboType\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateServices();

}

private void TBoxSearch\_TextChanged(object sender, TextChangedEventArgs e)

{

UpdateServices();

}

private void UpdateServices()

{

var currentServices = GilmetdinovaHospitalEntities.GetContext().DOCTOR.ToList();

if (ComboType.SelectedIndex == 0)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 1 || p.DOCTOR\_POST\_ID == 2 || p.DOCTOR\_POST\_ID == 3 || p.DOCTOR\_POST\_ID == 4 || p.DOCTOR\_POST\_ID == 5 || p.DOCTOR\_POST\_ID == 6

|| p.DOCTOR\_POST\_ID == 7 || p.DOCTOR\_POST\_ID == 8 || p.DOCTOR\_POST\_ID == 9 || p.DOCTOR\_POST\_ID == 10)).ToList();

}

if (ComboType.SelectedIndex == 1)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID ==1)).ToList();

}

if (ComboType.SelectedIndex == 2)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID ==2)).ToList();

}

if (ComboType.SelectedIndex == 3)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 3)).ToList();

}

if (ComboType.SelectedIndex == 4)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 4)).ToList();

}

if (ComboType.SelectedIndex == 5)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 5)).ToList();

}

if (ComboType.SelectedIndex == 6)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 6)).ToList();

}

if (ComboType.SelectedIndex == 7)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 7)).ToList();

}

if (ComboType.SelectedIndex == 8)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 8)).ToList();

}

if (ComboType.SelectedIndex == 9)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 9)).ToList();

}

if (ComboType.SelectedIndex == 10)

{

currentServices = currentServices.Where(p => (p.DOCTOR\_POST\_ID == 10)).ToList();

}

currentServices = currentServices.Where(p => p.DOCTOR\_SURNAME.ToLower().Contains(TBoxSearch.Text.ToLower())).ToList();

if (RButtonDown.IsChecked.Value)

{

currentServices = currentServices.OrderByDescending(p => p.DOCTOR\_SURNAME).ToList();

}

if (RButtonUp.IsChecked.Value)

{

currentServices = currentServices.OrderBy(p => p.DOCTOR\_SURNAME).ToList();

}

ServiceListView.ItemsSource = currentServices;

int ProductCount = 0;

foreach (DOCTOR product in currentServices)

{

ProductCount++;

}

ProductCountTextBlock.Text = ProductCount.ToString();

ServiceListView.ItemsSource = currentServices;

}

private void AddButton\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new AddEditPage(null));

UpdateServices();

ServiceListView.Items.Refresh();

}

private void ServiceListView\_SelectionChanged(object sender, SelectionChangedEventArgs e)

{

UpdateServices();

ServiceListView.Items.Refresh();

}

private void ServiceListView\_IsVisibleChanged(object sender, DependencyPropertyChangedEventArgs e)

{

UpdateServices();

}

private void editDoctor\_Click(object sender, RoutedEventArgs e)

{

Manager.MainFrame.Navigate(new AddEditPage((sender as Button).DataContext as DOCTOR));

UpdateServices();

ServiceListView.Items.Refresh();

}

private void deleteDoctor\_Click(object sender, RoutedEventArgs e)

{

var currentDoctor = (sender as Button).DataContext as DOCTOR;

var currentDoctorHistory = GilmetdinovaHospitalEntities.GetContext().OPERATION.ToList();

currentDoctorHistory = currentDoctorHistory.Where(p => p.OPERATION\_DOCTOR\_ID == currentDoctor.DOCTOR\_ID).ToList();

if (currentDoctorHistory.Count != 0)

{

MessageBox.Show("Невозможно выполнить удаление, так как врача запланирована операция!");

}

else

{

if (MessageBox.Show("Вы точно хотите удалить?", "Внимание!", MessageBoxButton.YesNo, MessageBoxImage.Question) == MessageBoxResult.Yes)

{

GilmetdinovaHospitalEntities.GetContext().DOCTOR.Remove(currentDoctor);

GilmetdinovaHospitalEntities.GetContext().SaveChanges();

ServiceListView.ItemsSource = GilmetdinovaHospitalEntities.GetContext().DOCTOR.ToList();

UpdateServices();

}}}}}

XAML-разметка страницы «AddEditPage»:

<<Page x:Class="GilmetdinovaHospital.AddEditPage"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:local="clr-namespace:GilmetdinovaHospital"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="800"

Title="AddEditPage" FontFamily="Bahnschrift SemiBold" FontSize="25" >

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="31\*"/>

<RowDefinition Height="31\*"/>

<RowDefinition Height="31\*"/>

<RowDefinition Height="31\*"/>

<RowDefinition Height="50"></RowDefinition>

<RowDefinition Height="0\*"></RowDefinition>

<RowDefinition Height="31"></RowDefinition>

<RowDefinition Height="31\*"/>

<RowDefinition Height="200.8"></RowDefinition>

<RowDefinition Height="43\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<TextBlock Text="Фамилия" HorizontalAlignment="Center" Grid.Row="0" Margin="109,9,175,16.8" Width="116" Grid.RowSpan="2"></TextBlock>

<TextBlock Text="Имя" HorizontalAlignment="Center" Margin="109,19.6,188,11.2" Grid.RowSpan="2" Width="103" Grid.Row="1"/>

<TextBlock Text="Отчество" HorizontalAlignment="Center" Grid.Row="2" Margin="109,20.2,140,14.6" Grid.RowSpan="2" Width="151"></TextBlock>

<TextBlock Text="Должность" HorizontalAlignment="Center" Grid.Row="3" Margin="109,10.8,126,38.2" Grid.RowSpan="2" Width="165"></TextBlock>

<TextBlock Text="Образование" HorizontalAlignment="Center" Grid.Row="4" Margin="109,24.4,113,18.4" Width="178" Grid.RowSpan="3"/>

<TextBlock Text="Адрес" HorizontalAlignment="Center" Grid.Row="7" Margin="109,0.6,184,148.4" Grid.RowSpan="2" Width="107"></TextBlock>

<TextBox Text="{Binding DOCTTOR\_NAME}" Grid.Row="0" Grid.Column="1" Margin="5,5,5,5.2"></TextBox>

<TextBox Text="{Binding DOCTOR\_SURNAME}" Grid.Row="1" Grid.Column="1" Margin="5,4.8,5,5.4"></TextBox>

<TextBox Text="{Binding DOCTOR\_PATRONYMIC}" Grid.Row="2" Grid.Column="1" Margin="5,4.6"></TextBox>

<TextBox Text="{Binding DOCTOR\_SCIENTIFIC }" Grid.Row="3" Grid.Column="1" Margin="5,5.4,5,4.8"></TextBox>

<ComboBox Grid.Row="3" Grid.Column="1" Margin="5,5.4,5,4.8" Name="ComboType">

<TextBlock><Run Text="Главный врач"/></TextBlock>

<TextBlock><Run Text="Младший медицинский персонал"/></TextBlock>

<TextBlock><Run Text="Заведующий отделением"/></TextBlock>

<TextBlock><Run Text="Врач-специалист"/></TextBlock>

<TextBlock><Run Text="Прочий персонал"/></TextBlock>

<TextBlock><Run Text="Технический персонал"/></TextBlock>

<TextBlock><Run Text="Лаборант"/></TextBlock>

<TextBlock><Run Text="Медицинский регистратор"/></TextBlock>

<TextBlock><Run Text="Медсестра общего профиля"/></TextBlock>

<TextBlock><Run Text="Социальный работник"/></TextBlock>

</ComboBox>

<TextBox Text="{Binding DOCTOR\_SCIENTIFIC}" Grid.Row="4" Grid.Column="1" Margin="5,10.4,5,26.4" Grid.RowSpan="3"></TextBox>

<TextBox Text="{Binding DOCTOR\_ADRESS}" Grid.Row="7" Grid.Column="1" Margin="5,3.6,5,4.6"></TextBox>

<Button Name="UploadPhotoButton" Content="Загрузить фото" Height="36" Background="#FFE074A3" Grid.Column="1" Grid.Row="8" Click="UploadPhotoButton\_Click" HorizontalAlignment="Center" Margin="156,110.4,44,47.4" Width="200"/>

<Image Grid.Column="0" Grid.Row="8" HorizontalAlignment="Center" Name="\_photo" Width="215" Margin="45,15.4,140,13.4">

<Image.Source>

<Binding Path="MainImagePath">

<Binding.TargetNullValue>

<ImageSource>

res/zaglushka.png

</ImageSource>

</Binding.TargetNullValue>

</Binding>

</Image.Source>

</Image>

<Button Name="SaveButton" Content="Сохранить" Grid.Row="9"

Grid.ColumnSpan="2" Click="SaveButton\_Click" Background="#FFE074A3" Margin="339,3.6,334,3.6" RenderTransformOrigin="0.5,0.5">

<Button.RenderTransform>

<TransformGroup>

<ScaleTransform/>

<SkewTransform/>

<RotateTransform Angle="-0.025"/>

<TranslateTransform/>

</TransformGroup>

</Button.RenderTransform>

</Button>

</Grid>

</Page>

Код функционала страницы «AddEditPage»:

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

using Microsoft.Win32;

namespace GilmetdinovaHospital

{

/// <summary>

/// Логика взаимодействия для AddEditPage.xaml

/// </summary>

public partial class AddEditPage : Page

{

public DOCTOR \_currentService = new DOCTOR();

public AddEditPage(DOCTOR SelectedService)

{

InitializeComponent();

if (SelectedService != null)

{

\_currentService = SelectedService;

ComboType.SelectedIndex = \_currentService.DOCTOR\_POST\_ID - 1;

}

if (SelectedService != null)

\_currentService = SelectedService;

//При инициализации установим DataContext страницы - этот созданный объект

DataContext = \_currentService;

}

private void SaveButton\_Click(object sender, RoutedEventArgs e)

{

StringBuilder err = new StringBuilder();

if (string.IsNullOrWhiteSpace(\_currentService.DOCTTOR\_NAME))

err.AppendLine("Укажите имя врача");

if (string.IsNullOrWhiteSpace(\_currentService.DOCTOR\_SURNAME))

err.AppendLine("Укажите фамилию врача");

if (string.IsNullOrWhiteSpace(\_currentService.DOCTOR\_PATRONYMIC))

err.AppendLine("Укажите отчество врача");

//if (ComboType.SelectedItem == null)

// err.AppendLine("Укажите отделение ");

if (ComboType.SelectedItem == null)

err.AppendLine("Укажите отделение ");

else

{

\_currentService.DOCTOR\_POST\_ID = ComboType.SelectedIndex + 1;

}

if (string.IsNullOrWhiteSpace(\_currentService.DOCTOR\_SCIENTIFIC))

err.AppendLine("Укажите образование врача");

if (string.IsNullOrWhiteSpace(\_currentService.DOCTOR\_ADRESS))

err.AppendLine("Укажите адрес врача");

if (err.Length > 0)

{

MessageBox.Show(err.ToString());

return;

}

if (\_currentService.DOCTOR\_ID == 0)

GilmetdinovaHospitalEntities.GetContext().DOCTOR.Add(\_currentService);

try

{

GilmetdinovaHospitalEntities.GetContext().SaveChanges();

MessageBox.Show("Информация сохранена");

Manager.MainFrame.GoBack();

}

catch (Exception ex)

{

MessageBox.Show(ex.ToString());

}}

private void UploadPhotoButton\_Click(object sender, RoutedEventArgs e)

{

OpenFileDialog myOpenFileDialog = new OpenFileDialog();

if (myOpenFileDialog.ShowDialog() == true)

{

FileInfo fileInfo = new FileInfo(myOpenFileDialog.FileName);

if (fileInfo.Length < 2 \* 1024 \* 1024)

{

\_currentService.MainImagePath = myOpenFileDialog.FileName;

\_photo.Source = new BitmapImage(new Uri(myOpenFileDialog.FileName));

}

else

{

MessageBox.Show("Размер файла превышает 2 мегабайта. Выберите другой файл.");}}}}}

Работа модулей редактирования и добавления поставщиков представлена в соответствии с рисунками 4,5:

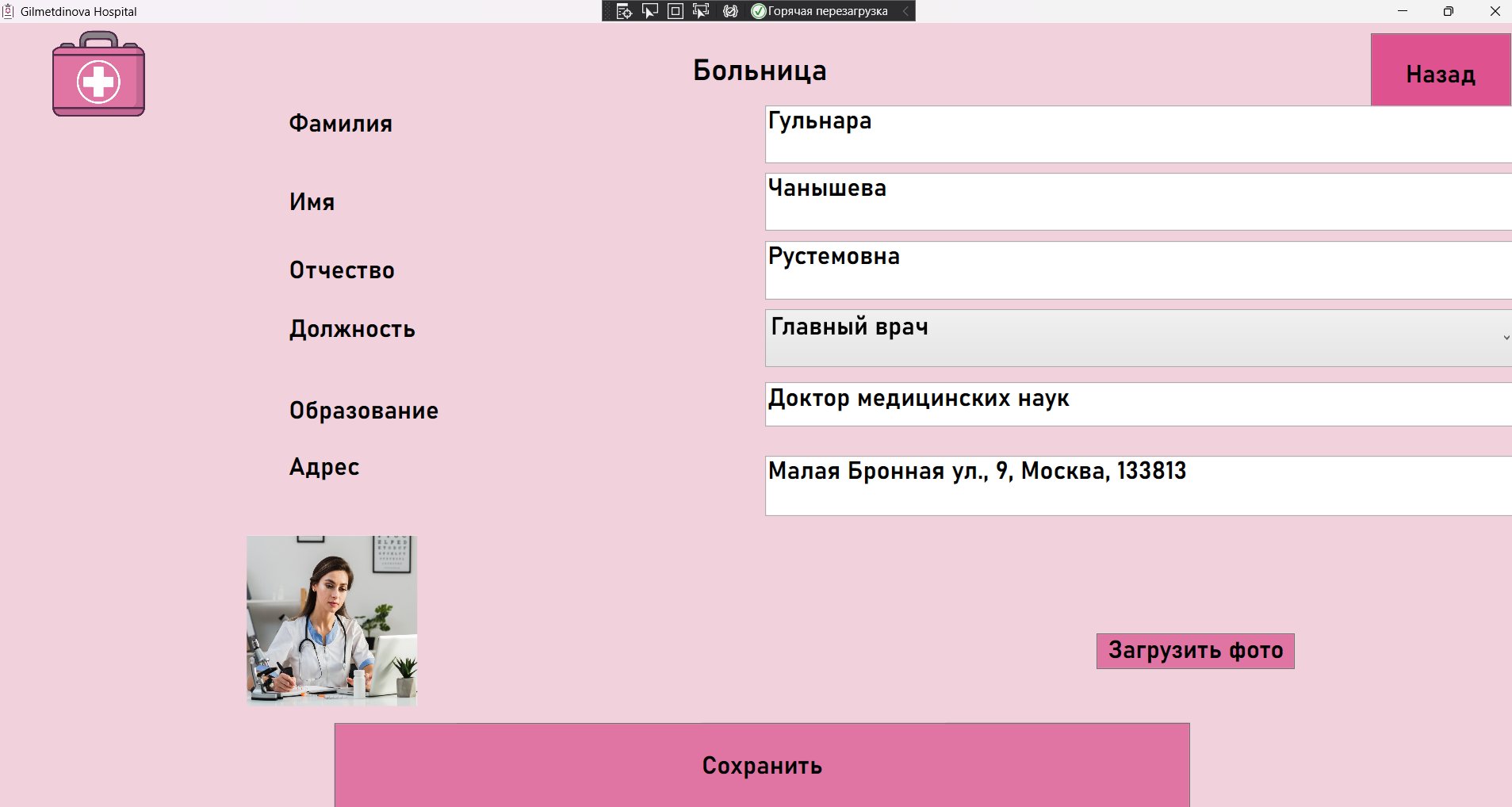


Рисунок 4 – Окно редактирования

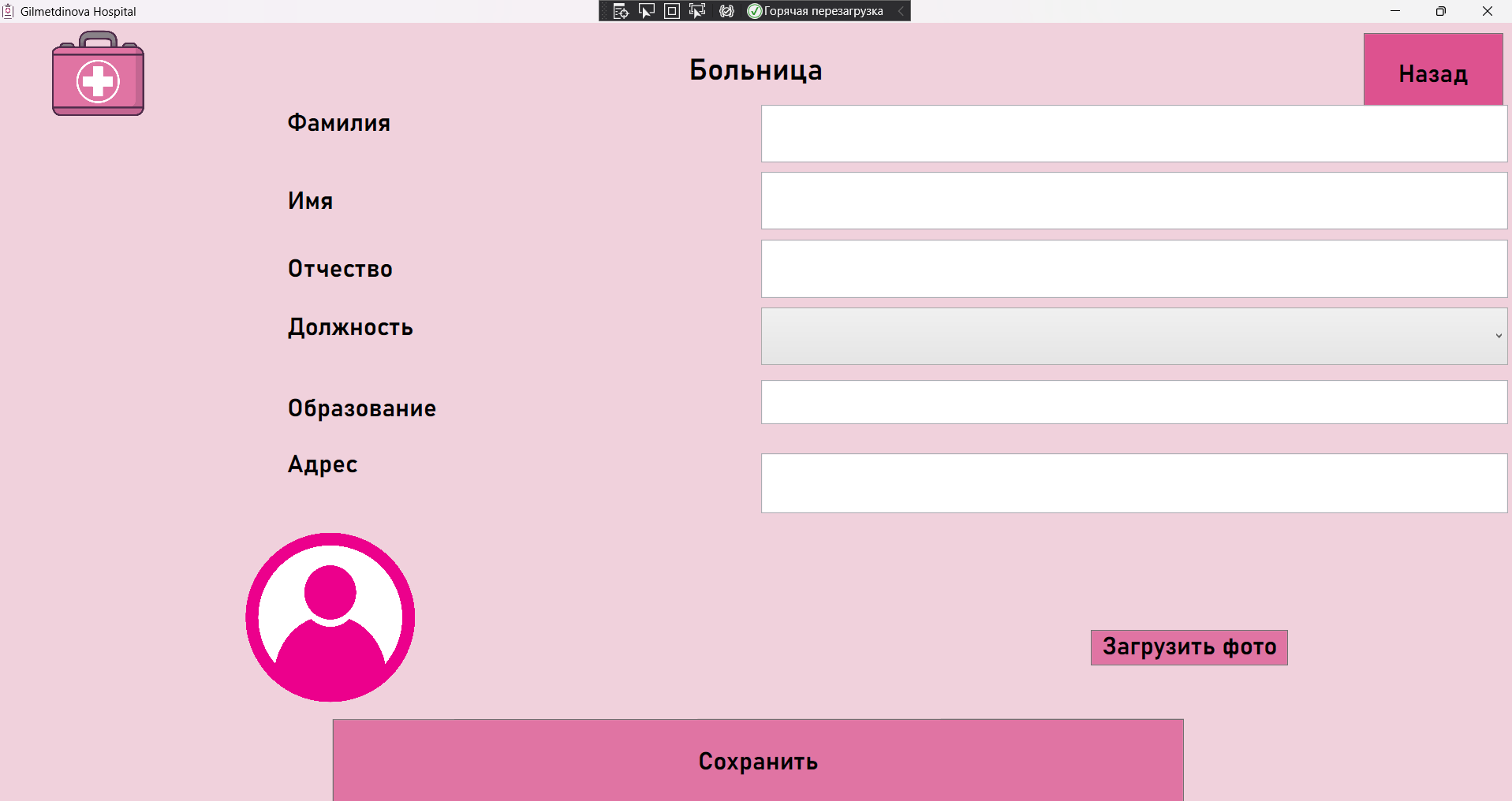


Рисунок 5 – Окно добавления

На странице присутствует поисковая строка для поиска поставщиков, представленная в соответствии с рисунком 6.

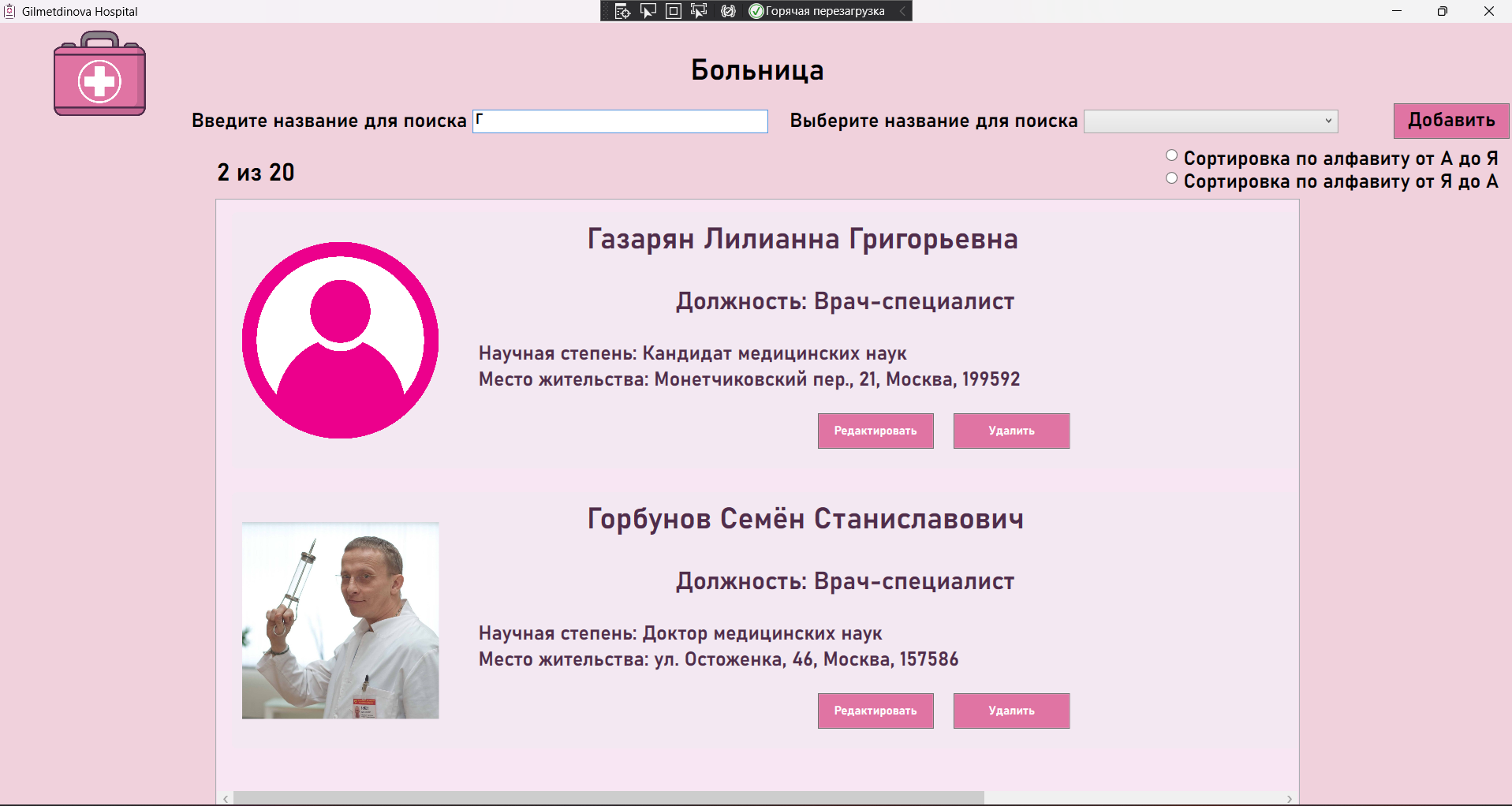


Рисунок 6 –Результат поиска

Также есть сортировка списка фамилий сотрудников в алфавитном порядке и в обратном алфавитном порядке, как показано в соответствии с рисунком 7.

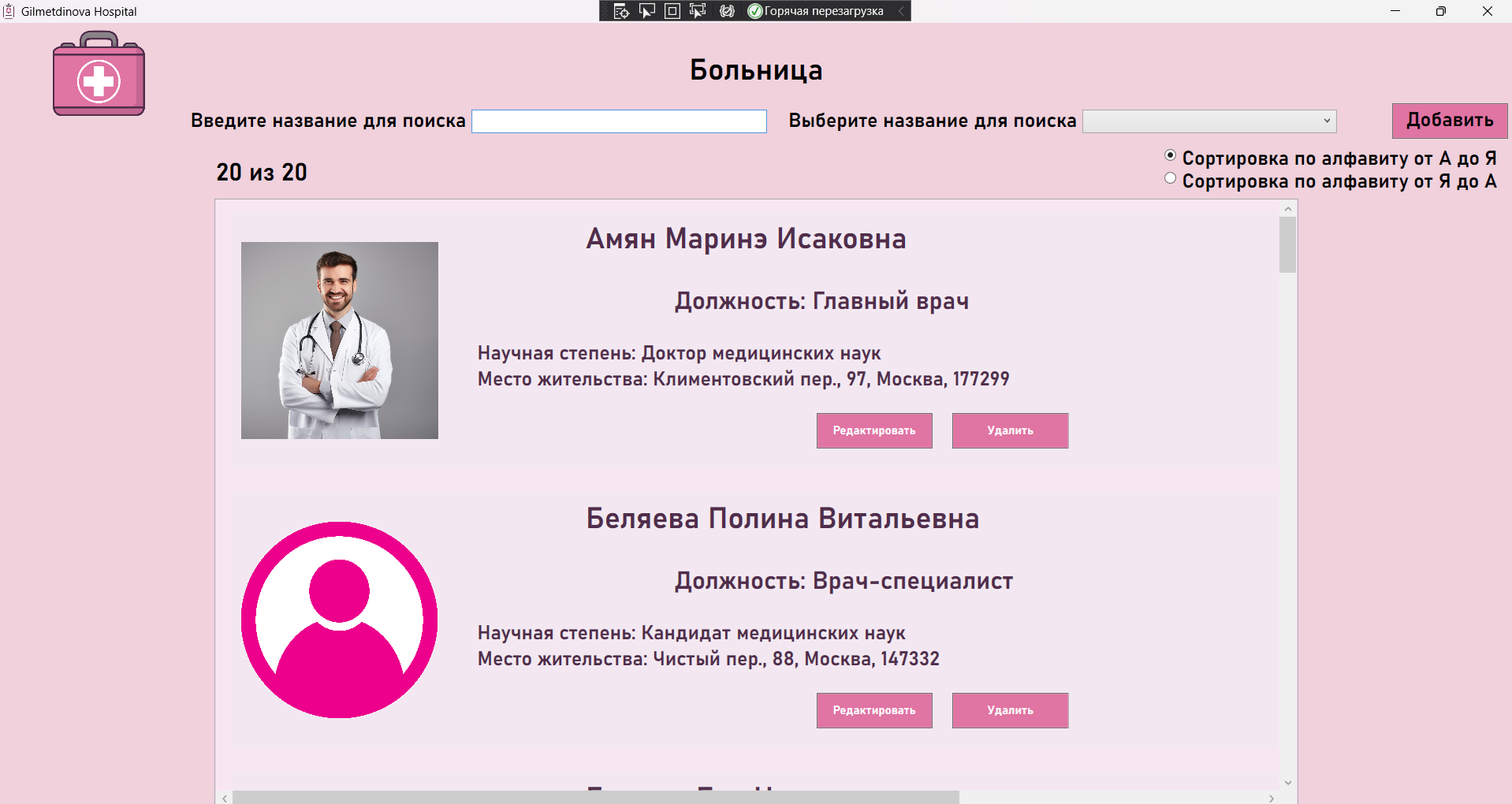


Рисунок 7 – Результат сортировки

Есть фильтрация списка поставщиков по виду должности, как показано в соответствии с рисунком 8.

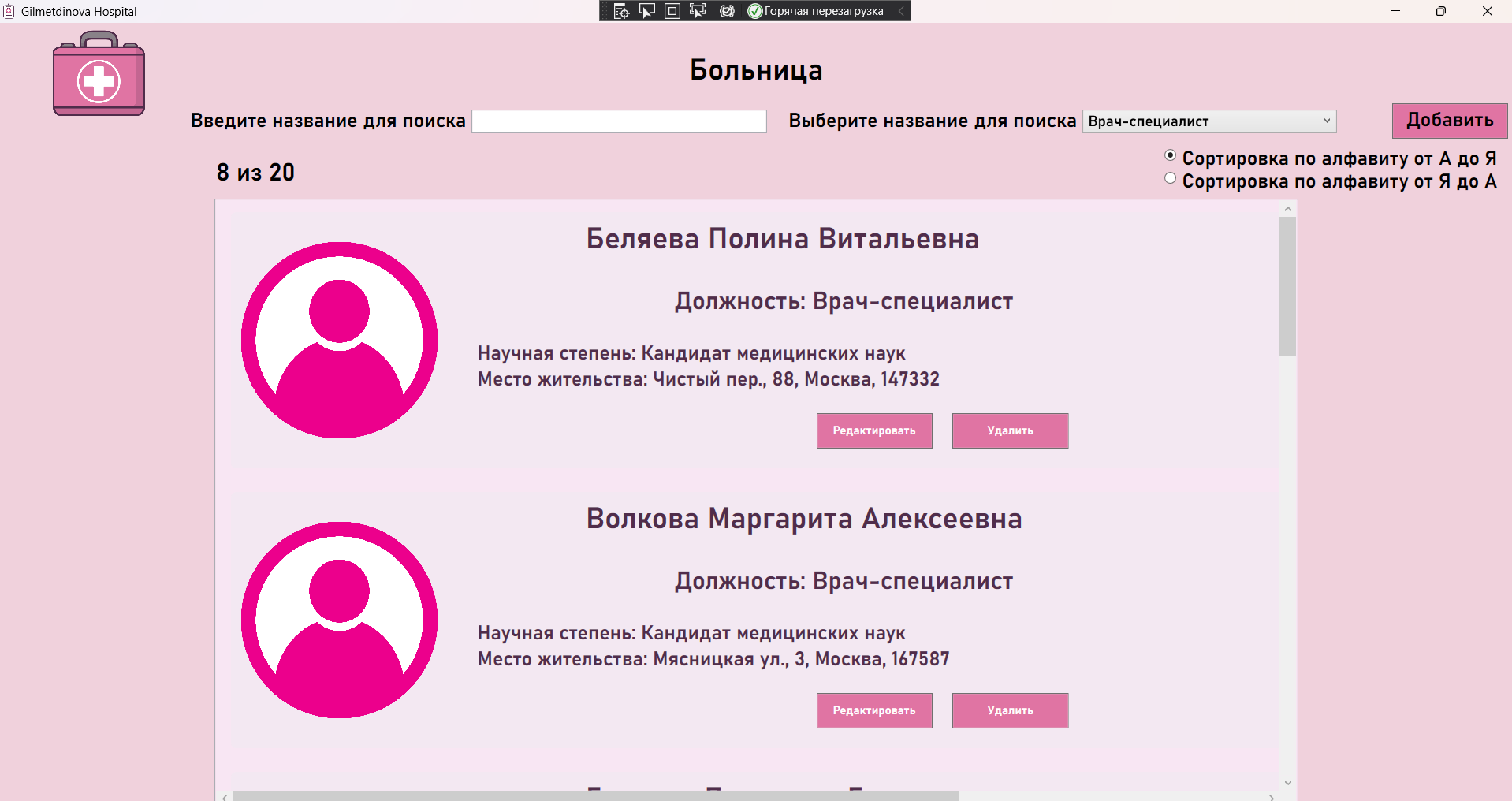


Рисунок 8 – Результат фильтрации

На странице есть возможность добавить, изменить или удалить клиента, как показано в соответствии с рисунками 9– 12.

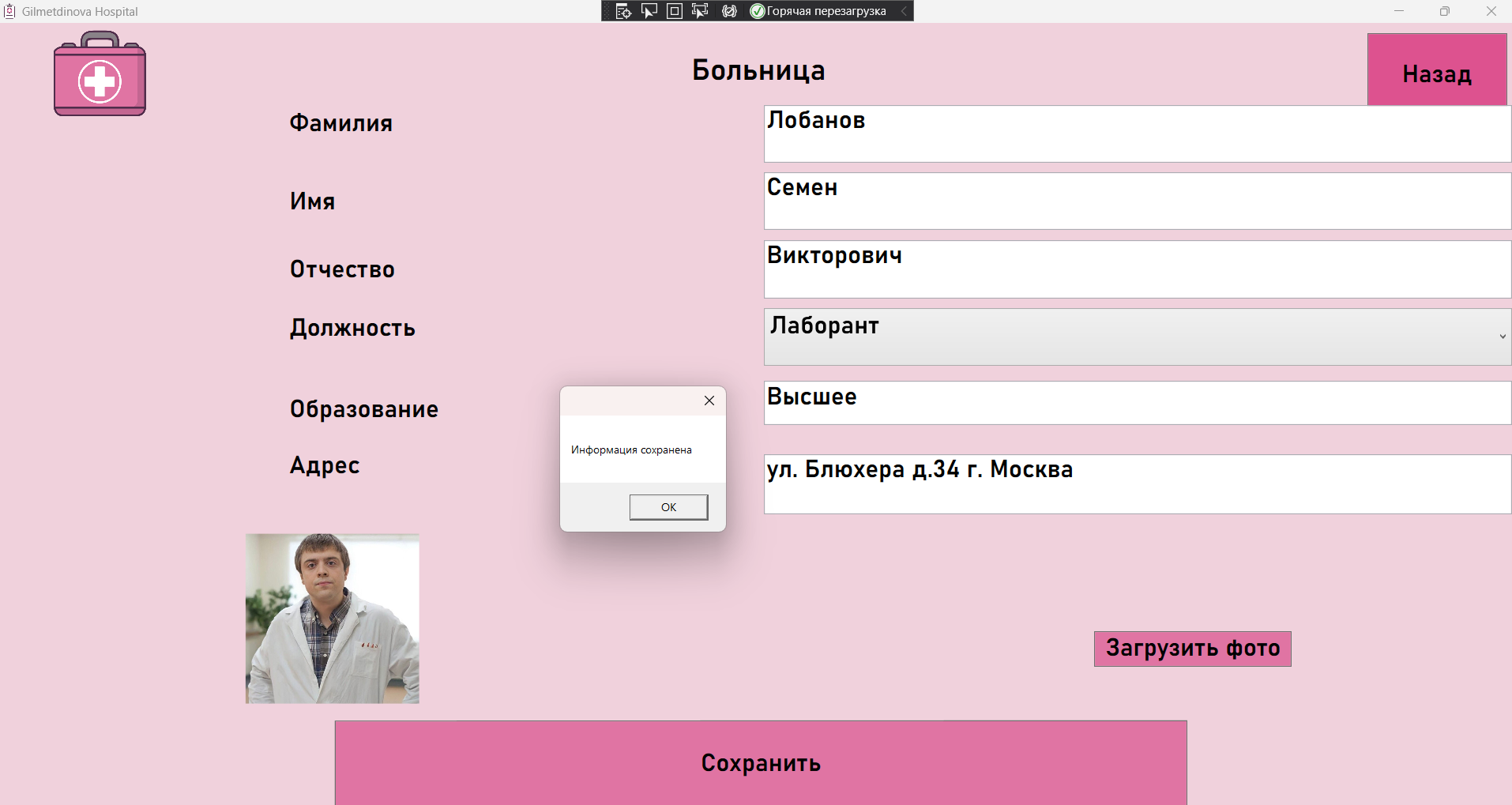


Рисунок 9 – Результат добавления новой записи

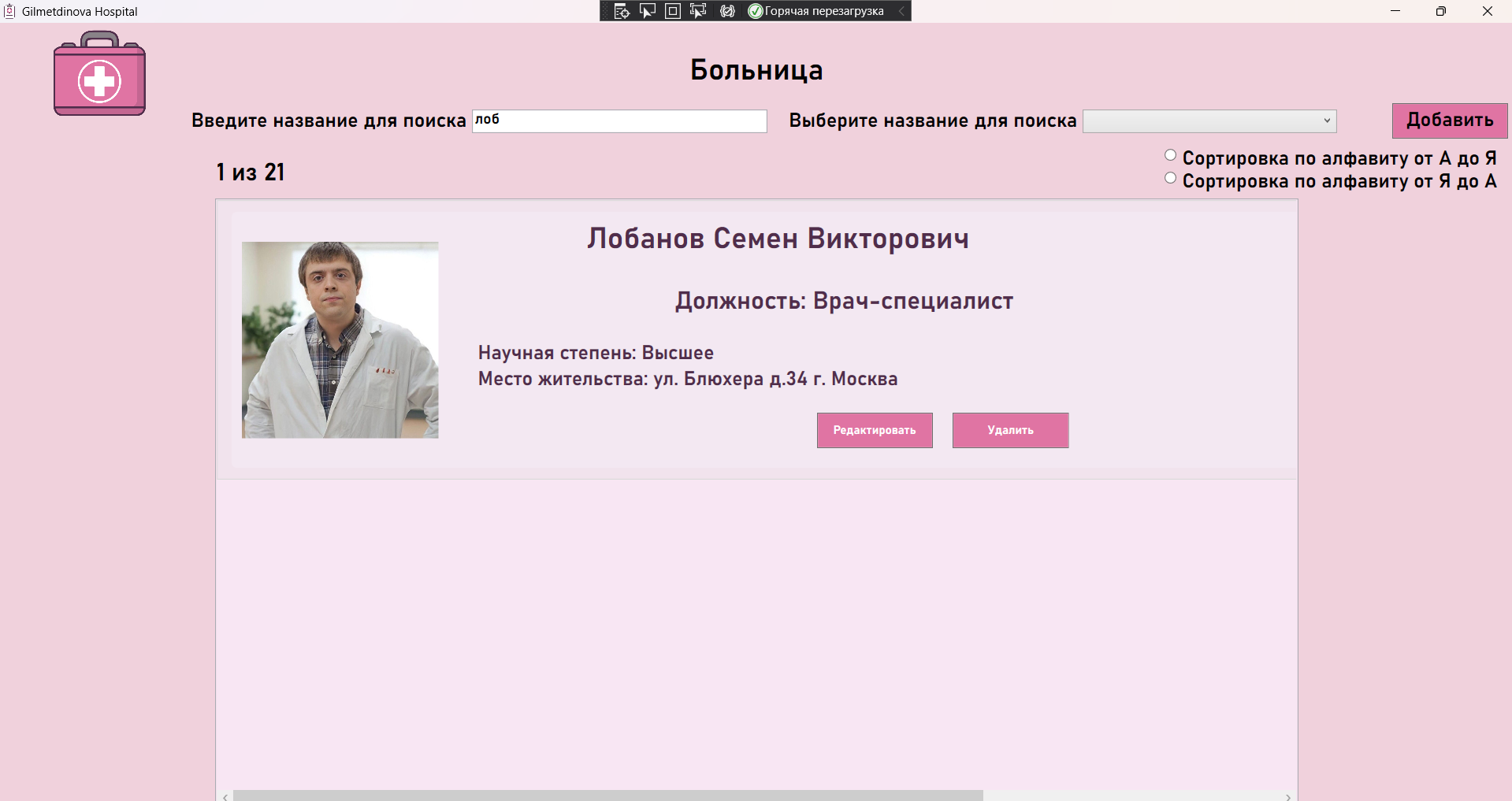


Рисунок 10 – Результат изменения данных записи

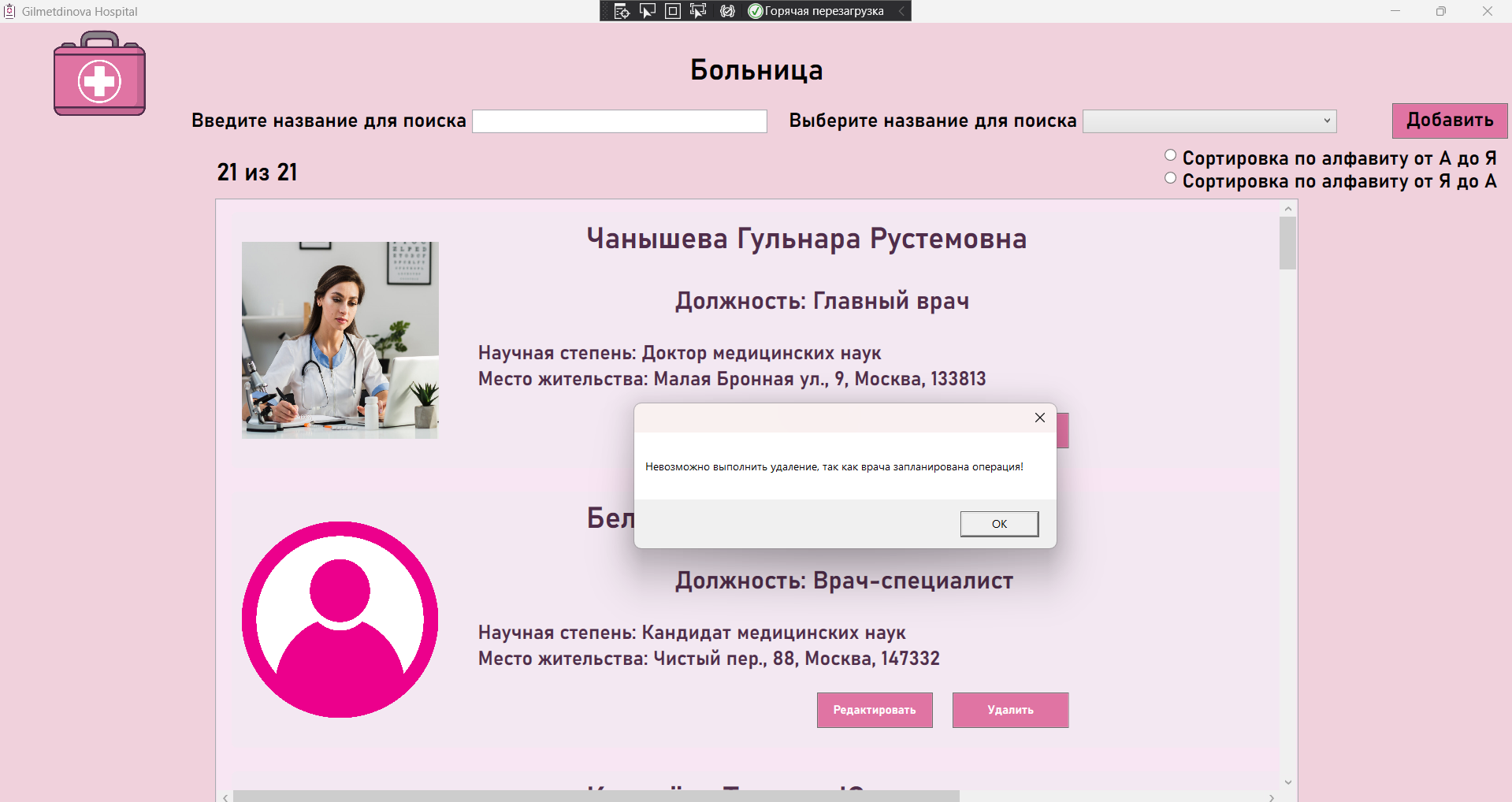


Рисунок 11 – Попытка удаления записи



Рисунок 12 – Результат удаления записи

Разработка программного обеспечения – важная часть программной инженерии. Для успешного проекта необходимы тщательное планирование, современные методологии, эффективная командная работа, гибкость и постоянный контроль на всех этапах создания продукта.

# **Заключение**

В ходе выполнения курсового проекта удалось успешно достичь всех поставленных целей благодаря прохождению всех этапов разработки программного обеспечения.

На первом этапе был определён необходимый функционал приложения, включающий ключевые требования к его работе и интерфейсу для пользователей. Это позволило чётко сформулировать задачи и ожидания от итогового продукта.

Затем был осуществлён выбор технологий и инструментов для реализации проекта. Были рассмотрены различные программные средства, учитывая их функциональные возможности, совместимость и удобство использования. В итоге выбраны наиболее подходящие решения, обеспечивающие высокую производительность и надёжность разработки.

Далее была разработана подробная структура программного обеспечения. Определена архитектура системы, выделены модули и компоненты, а также описаны их взаимодействия. Это способствовало упорядоченной организации кода и упростило дальнейшую разработку и поддержку продукта.

Четвёртым этапом стала непосредственная реализация программного решения. Код был написан в соответствии с разработанной структурой, проведены тесты, и исправлены выявленные ошибки. Итоговый продукт полностью соответствует установленным требованиям и готов к использованию.

Цели проекта достигнуты, задачи успешно выполнены.

# **Список использованных источников**

1. Робинсон, Адам. "Технологии проектирования и разработки пользовательских интерфейсов". – СПб.: БХВ-Петербург, 2021. – 320 с.
2. Зубкова, Т. М. Технология разработки программного обеспечения : учебное пособие для спо / Т. М. Зубкова. — 2-е изд., стер. — Санкт-Петербург : Лань, 2022. — 252 с. — ISBN 978-5-8114-9556-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/200462 — Режим доступа: для авториз. пользователей.
3. Белугина, С. В. Разработка программных модулей программного обеспечения для компьютерных систем. Прикладное программирование / С. В. Белугина. — 4-е изд., стер. — Санкт-Петербург : Лань, 2023. — 312 с. — ISBN 978-5-507-46061-8. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/296975 — Режим доступа: для авториз. пользователей.
4. Троелсен Э., Джеппсон Ф. "C# 10 и платформа .NET 6. Разработка на современном уровне". – СПб.: Питер, 2022. – 864 с.
5. MacDonald, Matthew. Pro WPF in C#: Windows Presentation Foundation in .NET 7. Apress, 2023.
6. Скотт Майерс. "Эффективное программирование: 42 рекомендации по улучшению структуры и качества кода C++". – М.: Вильямс, 2020. – 336 с.

# **Приложение А**