МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБЩЕОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им В.Г.ШУХОВА»
(БГТУ им. В.Г.Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

КУРСОВОЙ ПРОЕКТ

по дисциплине: Базы данных
тема: «Реализация системы медицинской организации»

Автор работы_____Потицкий Владислав Сергеевич ВТ-211
(подпись)

Руководитель проекта _____Панченко Максим Владимирович
(подпись)

Оценка _____

Белгород 2024

Содержание

Введение....3
1. Основы разработки системы....4
1.1 Основные понятия....4
1.2 Выбор программного обеспечения....5
1.3 Описание разрабатываемой системы....7
2 Создание системы....10
2.1 Этапы разработки....10
2.2 Описание интерфейса программы....18
2.3 Тестирование готового продукта....20
Заключение....21
Список источников и литературы....22
Приложение А....24

Введение

В современном мире, где качество жизни и здоровье населения являются приоритетными направлениями развития общества, система медицинской организации становится все более актуальной темой для исследований и обсуждений. Реализация данной системы является сложной задачей, требующей комплексного подхода и учета множества факторов.

Целью данного исследования является анализ и оценка существующих систем медицинской организации, а также разработка рекомендаций по их реализации в различных условиях.

На основе анализа многих факторов можно разработать оптимальную модель медицинской организации, которая будет эффективной и доступной для всех слоев населения, а главное простой и минималистичной в понимании.

Таким образом, реализация системы медицинской организации требует глубокого исследования и анализа существующих подходов, а также разработки новых методов и технологий, которые будут способствовать улучшению качества жизни и здоровья населения.

Объектом исследования является процесс разработки системы медицинской организации.

Предмет исследования – системный язык программирования C# и база данных.

Цель – разработать систему медицинской организации.

Для достижения поставленной цели необходимо решить следующие задачи:

- составить требования для разрабатываемой системы;
- разработать структуру системы и описать ее;
- применить на практике язык программирования C#;
- создание системы с использованием базы данных (БД);
- тестирование готового результата.

1. Основы разработки системы

1.1 Основные понятия

База данных – это организованная структура данных, которая хранит и обрабатывает информацию в электронном виде. Базы данных используются для хранения и управления информацией в различных приложениях.

C# – это высокоуровневый объектно-ориентированный язык программирования, разработанный компанией Microsoft. Он используется для создания различных типов приложений, включая веб-приложения, мобильные приложения, игры и т.д.

Объектно-ориентированное программирование (ООП) – это подход к разработке программного обеспечения, который основан на использовании объектов и классов. В C# все данные и методы определяются внутри классов, которые могут быть унаследованы от других классов.

LINQ (Language Integrated Query) – это набор методов и операторов, которые позволяют работать с данными в C#. LINQ используется для выполнения различных операций над данными, такими как фильтрация, сортировка, группировка и т.д.

Visual Studio – это интегрированная среда разработки (IDE), созданная компанией Microsoft для разработки приложений на различных языках программирования, таких как C#, Visual Basic .NET, F# и других. Visual Studio включает в себя множество инструментов для разработки, таких как компилятор, отладчик, редактор кода, инструменты для работы с базами данных и многое другое.

Entity Framework Core – это ORM (Object-Relational Mapper), который позволяет работать с базами данных на уровне объектов и классов.

ASP.NET Core – это платформа для создания веб-приложений на основе .NET Framework.

1.2 Выбор программного обеспечения

При создании системы медицинской организации необходимо выбрать программное обеспечение, для построения клиентских приложений Windows с визуально привлекательными возможностями взаимодействия с пользователем, выбрана графическая подсистема в составе .NET Framework, и будет разработано Приложение WPF (.NET Framework) C# с помощью Visual Studio.

Кроме того, необходимо выбрать ПО для создания базы данных. Для хранения данных о расписании врачей, пациентах, записях на прием и другой информации нужна надежная и производительная база данных. В таблице 1 представлены варианты ПО для создания базы данных.

Таблица 1

Анализ характеристик ПО для БД

База данных	**MySQL**	**PostgreSQL**	SQL Server Management Studio
Цена	Бесплатно, с платными инструментами от Oracle	Бесплатно	Бесплатно
Лицензия	Открытый исходный код	Открытый исходный код	Закрытый исходный код
Производительность	Хорошо подходит для больших объемов данных	Хорошо подходит для работы со сложными запросами	Хорошо подходит для обработки данных в реальном времени
Надежность	Высокая надежность и стабильность	Высокая надежность и стабильность	Высокая надежность, но требует больше ресурсов для работы
Поддержка типов данных	Широкий спектр типов данных	Поддержка большого количества типов данных	Ограниченный набор типов данных

MySQL и PostgreSQL являются популярными решениями для хранения данных, однако для реализации системы медицинской организации была выбрана Microsoft SQL Server. Выбор обусловлен рядом преимуществ, которые предоставляет данная система.

Microsoft SQL Server имеет высокую производительность и масштабируемость, что особенно важно для медицинских организаций, где объем данных может быть большим. Кроме того, SQL Server предлагает более широкий спектр инструментов для работы с данными по сравнению с MySQL или PostgreSQL, что упрощает разработку и поддержку системы.

SQL Server Management Studio (SSMS) – это утилита из Microsoft SQL Server 2005 и более поздних версий для конфигурирования, управления и администрирования всех компонентов Microsoft SQL Server.

Утилита включает скриптовый редактор и графическую программу, которая работает с объектами и настройками сервера.

Главным инструментом SQL Server Management Studio является Object Explorer, который позволяет пользователю просматривать, извлекать объекты сервера, а также полностью ими управлять.

Также стоит отметить, что Microsoft SQL Server является коммерческим продуктом, что может быть важным фактором для некоторых организаций.

1.3 Описание разрабатываемой системы

Подсистема «Управление потоком пациентов»

Функция «Регистрация пациентов»

Следует разработать функцию «Регистрация пациентов» в веб-интерфейсе, и обеспечить работу следующего функционала:

1. Введение информации о пациенте (ФИО, паспортные данные, место работы, страховой полис (номер, срок действия) и т.д.).

2. Сохранение в БД данных о пациенте.

3. Для автоматического распознавания идентификационного кода медицинской карты используется QR-кодов, который в последствии будет реализован с мобильной версией, с отображением информации о найденном пациенте.

4. Возможность подготовки (заполнение) необходимых сопутствующих документов: договор на медицинское обслуживание и согласие на обработку персональных данных в формате .docx (шаблоны предоставлены в ресурсах).

Функция «Госпитализация»

Следует разработать функцию «Госпитализация» в веб-интерфейсе, обеспечив работу следующего функционала:

1. Введение информации о пациенте (ФИО, паспортные данные, место работы, страховой полис (номер, срок действия) и т.д.) для записи на госпитализацию;

2. Введение кода на госпитализацию, полученного у терапевта;

3. Выбор даты госпитализации;

4. Отмена госпитализации терапевтом с отображением причины отказа.

Функция «Направление пациентов на лечебно-диагностические мероприятия»

Следует разработать функцию «Направление пациентов на лечебно-диагностические мероприятия» обеспечив работу следующего функционала:

1. Вход под УЗ доктора;

2. Просмотр доступной информации о пациенте;

3. Работа с мед. картой пациента, записанного на прием: сбор анамнеза, описание симптоматики, указание диагноза, рекомендации по лечению.

Функция «Расписание»

Следует разработать функцию «Расписание» в веб-интерфейсе, обеспечив работу следующего функционала:

1. Вывод из таблицы Врачей данных (фамилия и специализация);

2. Отображение данных о Расписании (день недели, время и т.д.).

Для реализации удобной структуры системы медицинской организации первоначально нужно провести анализ предметной области и после чего составить диаграмму. Она помогает понять структуру системы, определить какие объекты и связи между ними присутствуют (Рис.1). На ней показаны отношения связи.

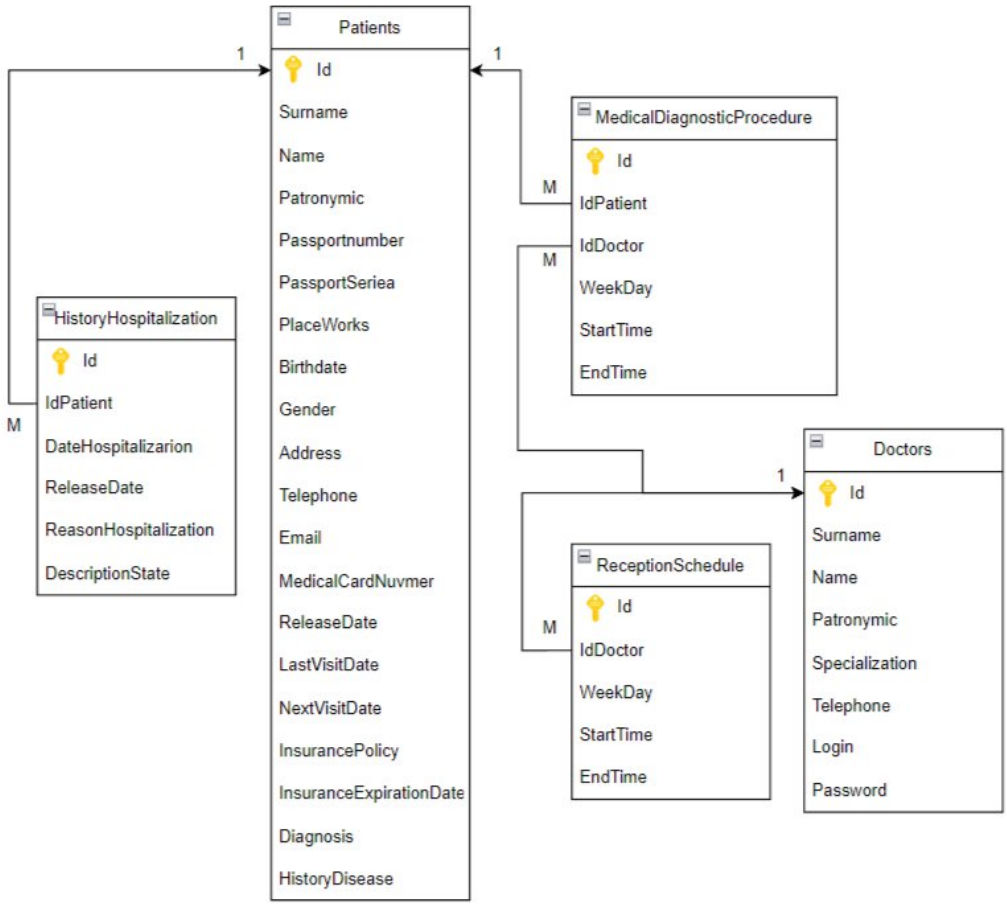

Рис.1. Диаграмма

Отношение «Один ко Многим»:

1. Patients (Один) – HistoryHospitalizations (Много):

1. У пациента может быть множество историй госпитализаций.

2. В таблице HistoryHospitalizations поле IdPatient ссылается на Id пациента из таблицы Patients.

2. Doctors (Один) – MedicalDiagnosticProcedure (Много):

1. У каждого врача, может быть, множество медицинских процедур.

2. В таблице MedicalDiagnosticProcedure поля IdPatient, IdDoctor указывают на соответствующие Id из таблиц Patients и Doctors.

3. Doctors (Один) – ReceptionSchedule (Много):

1. У врача может быть много записей в расписании приема.

2. В таблице ReceptionSchedule поле IdDoctor ссылается на Id врача из таблицы Doctors.

Отношения «Многие ко Многим» и «Один к Одному»:

В базе данных таких связей нет.

2 Создание системы

2.1 Этапы разработки

Первым шагом при создании приложения является выбор языка программирования и фреймворка. Для этого были выбраны C# и программа Visual Studio 2022. Кроме того, необходимо выбрать базу данных, которая будет использоваться в приложении (Microsoft SQL Management Studio).

После подготовки окружения необходимо определить функциональность приложения и создать базовую структуру проекта.

Файловая структура проекта отражает логику, заложенную в приложение. Например, все формы содержатся в одной директории, пользовательские визуальные компоненты – в другой, классы сущностей – в третьей.

Каждая сущность должна представлена в программе как минимум одним отдельным классом. Классы небольшие и понятные.

Логика представления (работа с пользовательским вводом/выводом, формы, обработка событий) не перемешана с бизнес-логикой (ограничения и требования, сформулированные в заданиях), а также не перемешана с логикой доступа к базе данных (SQL-запросы, запись, получение данных).

Интерфейс приложения играет важную роль и на рисунке 2 можно посмотреть пример главной страницы. Для создания необходимо выбрать дизайн и создать соответствующие страницы. Следует открыть программу и создать Приложение WPF (.NET Framework), дать название проекту и с использованием различных элементов воссоздать макет, таких как: Button, TextBlock, TextBox, Frame и т.д.

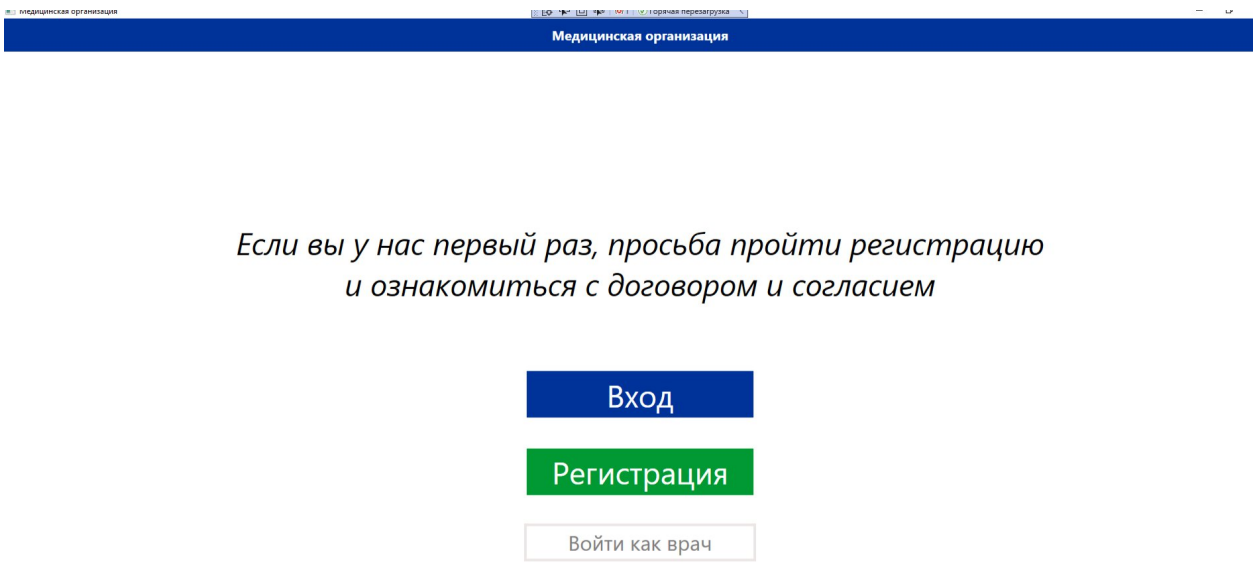

Рис.2. Главная страница

На рисунке 3 можно ознакомиться с входом пациента, аналогичный вход будет и для сотрудника (врача).

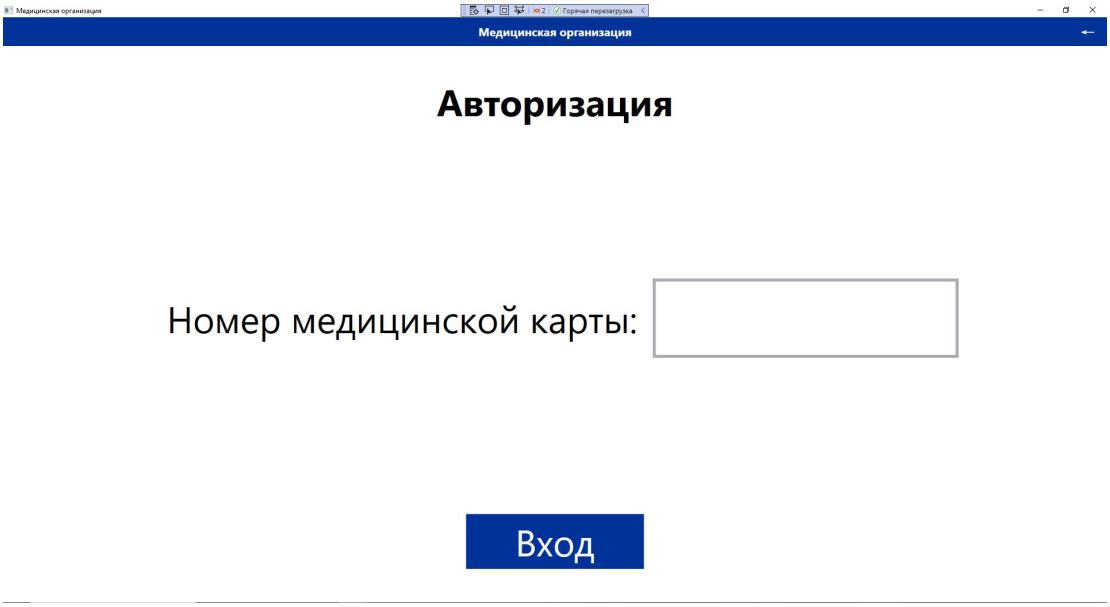

Рис.3. Авторизация пациента

Для того, чтобы вход был подтвержден, необходимо подключить базу данных и затем для создания логики определить основную функциональность приложения и создать соответствующие классы и методы.

На рисунке 4 представлена диаграмма из SQL, со всеми таблицами, полями и связями.

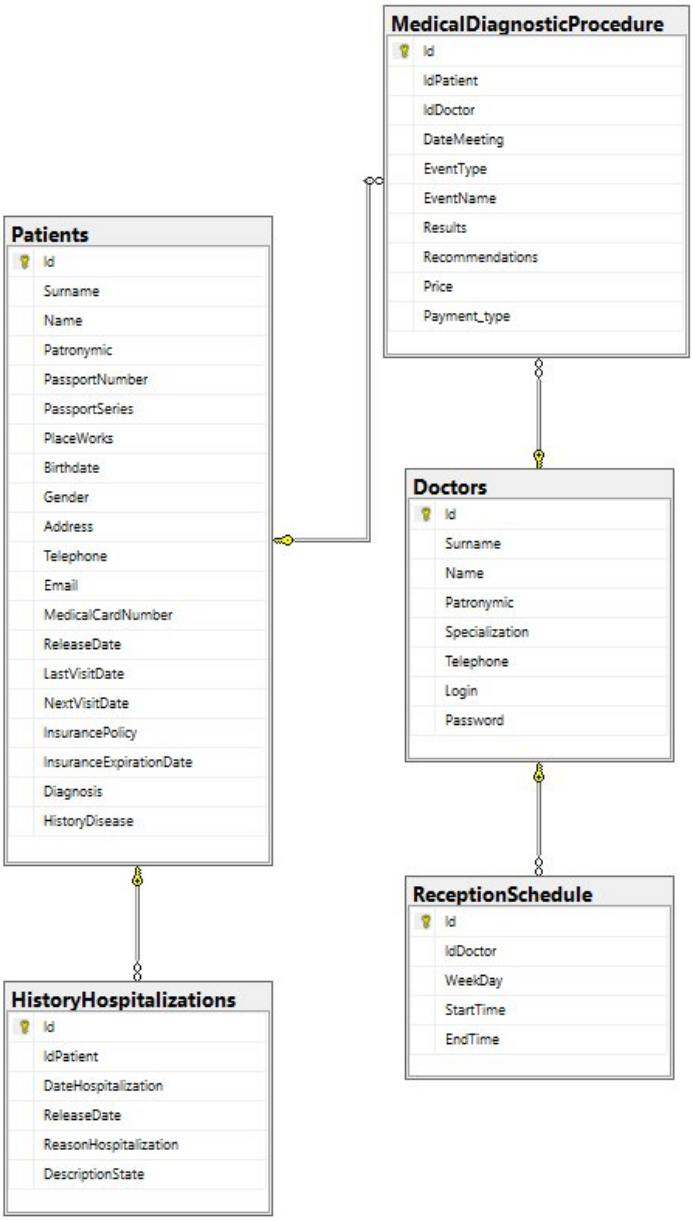

Рис.4. Диаграмма

После следует подключение базы данных (БД) к проекту, в следствии чего программа сможет сверять данные уже существующих пациентов, а также заносить в БД новых пользователей, после их регистрации. После чего необходимо создать 2 класса.

1. AppConnect

```
using Hospital.Model;
namespace Hospital
{
    class AppConnect
    {
        public static HospitalEntities1 model0db;
```

```
    }
}
2. AppFrame
using System.Windows.Controls;
namespace Hospital
{
    class AppFrame
    {
        public static Frame frameMain;
    }
}
```

А также класс Manager, для перехода по кнопкам.

```
Manager
using System.Windows.Controls;
namespace Hospital
{
    internal class Manager
    {
    public static Frame myFrame { get; set; }
    }
}
```

В том случаи, если человек ещё не зарегистрирован, то должен пройти регистрацию. Пример работы страницы регистрации на рисунке 5.

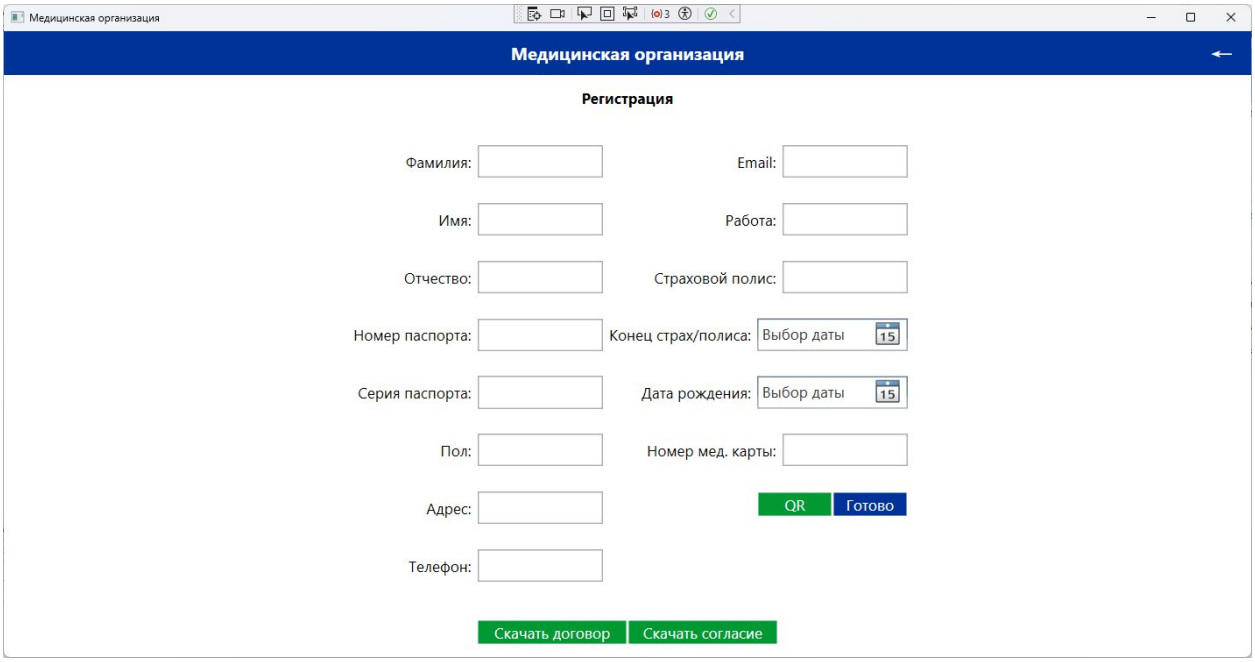

Рис.5. Регистрация

Код регистрации

```
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Controls;
using Microsoft.Win32;
using System.Windows;
```

```
using Hospital.Model;
using System.IO;
using System;

namespace Hospital
{
    public partial class Registration : Page
    {
        private Frame myFrame;

        public Registration()
        {
            InitializeComponent();
            AppConnect.HospitalModel = new HospitalEntities9();
            AppFrame.frameMain = myFrame;
        }

        // Отображает изображение в новом окне
        private void QR(object sender, RoutedEventArgs e)
        {
            Window imageMessageBox = new Window();
            imageMessageBox.Title = "QR Code";
            imageMessageBox.Width = 500;
            imageMessageBox.Height = 500;
            Image image = new Image();
            image.Source = new BitmapImage(new
Uri("pack://application:,,,/Hospital;component/Resources/qr.jpg"));
            imageMessageBox.Content = image;
            imageMessageBox.ShowDialog();
        }

        private void DownloadContract(object sender, RoutedEventArgs e)
        {
            string fileName =
"Форма_договора_предоставления_платных_медицинских_услуг.docx";

            byte[] docBytes =
Properties.Resources.Форма_договора_предоставления_платных_медицинских_услуг;
// Получаем байты документа из ресурсов

            SaveFileDialog saveFileDialog = new SaveFileDialog();
            saveFileDialog.FileName = fileName;
            saveFileDialog.Filter = "Документы Word (*.docx)|*.docx";

            if (saveFileDialog.ShowDialog() == true)
            {
                string filePath = saveFileDialog.FileName;
                File.WriteAllBytes(filePath, docBytes); // Записываем байты в
выбранный файл
                MessageBox.Show("Документ загружен. Путь: " + filePath);
            }
        }

        private void DownloadApproval(object sender, RoutedEventArgs e)
        {
            string fileName = "Согласие_на_обработку_ПД.docx";

            byte[] docBytes = Properties.Resources.Согласие_на_обработку_ПД;

            SaveFileDialog saveFileDialog = new SaveFileDialog();
            saveFileDialog.FileName = fileName;
            saveFileDialog.Filter = "Документы Word (*.docx)|*.docx";
```

```
            if (saveFileDialog.ShowDialog() == true)
            {
                string filePath = saveFileDialog.FileName;
                File.WriteAllBytes(filePath, docBytes);
                MessageBox.Show("Документ загружен. Путь: " + filePath);
            }
        }

        private void Done(object sender, RoutedEventArgs e)
        {
            if (Surname.Text == "" || Name.Text == "" || Patronymic.Text ==
"" || Nomer.Text == "" ||
                Seria.Text == "" || Gender.Text == "" || Address.Text == ""
|| Phone.Text == "" ||
                Mail.Text == "" || Work.Text == "" || Polis.Text == "" ||
Number.Text == "")
            {
                MessageBox.Show("Заполните все поля.");
            }
            else
            {
                Patients people = new Patients();
                people.Surname                 = Surname.Text;
                people.Name                    = Name.Text;
                people.Patronymic              = Patronymic.Text;
                people.PassportSeries          = int.Parse(Seria.Text);
                people.PassportNumber          = int.Parse(Nomer.Text);
                people.Gender                  = Gender.Text;
                people.Address                 = Address.Text;
                people.Telephone               = Phone.Text;
                people.Email                   = Mail.Text;
                people.PlaceWorks              = Work.Text;
                people.InsurancePolicy         = Polis.Text;
                people.MedicalCardNumber       = int.Parse(Number.Text);
                people.Birthdate               = DataPic.SelectedDate.Value;
                people.InsuranceExpirationDate =
DataPolis.SelectedDate.Value;

                VariableClass.Surname    = Surname.Text;
                VariableClass.Name       = Name.Text;
                VariableClass.Patronymic = Patronymic.Text;
                VariableClass.Seria      = Seria.Text;
                VariableClass.Nomer      = Nomer.Text;
                VariableClass.Gender     = Gender.Text;
                VariableClass.Address    = Address.Text;
                VariableClass.Phone      = Phone.Text;
                VariableClass.Mail       = Mail.Text;
                VariableClass.Work       = Work.Text;
                VariableClass.Polis      = Polis.Text;

                if (AppConnect.HospitalModel != null)
                {
                    AppConnect.HospitalModel.Patients.Add(people);
                    AppConnect.HospitalModel.SaveChanges();
                    MessageBox.Show("Вы зарегистрированы!");

                    // Backup
                    string connectionString =
"Server=SPC\\STP;Database=Hospital;Integrated Security=True;";
                    string databaseName = "Hospital";
                    string backupPath = "D:\\";
```

```
                    using (SqlConnection connection = new
SqlConnection(connectionString))
                    {
                        connection.Open();

                        string backupQuery = $"BACKUP DATABASE {databaseName}
TO DISK = '{backupPath + databaseName + "_" + DateTime.Now.ToString("yyyy-MM-
dd_HHmmss") + ".bak"}'";

                        using (SqlCommand command = new
SqlCommand(backupQuery, connection))
                        {
                            command.ExecuteNonQuery();
                            Console.WriteLine("Резервная копия успешно
создана.");
                        }
                    }

                    NavigationService.Navigate(new PatientPersonalAccount());
                }
                else
                {
                    MessageBox.Show("Ошибка в подключении к базе данных.");
                }
            }
        }
    }
}
```

После чего откроется личный кабинет пациента или врача, пример личного кабинета на рисунке 6.

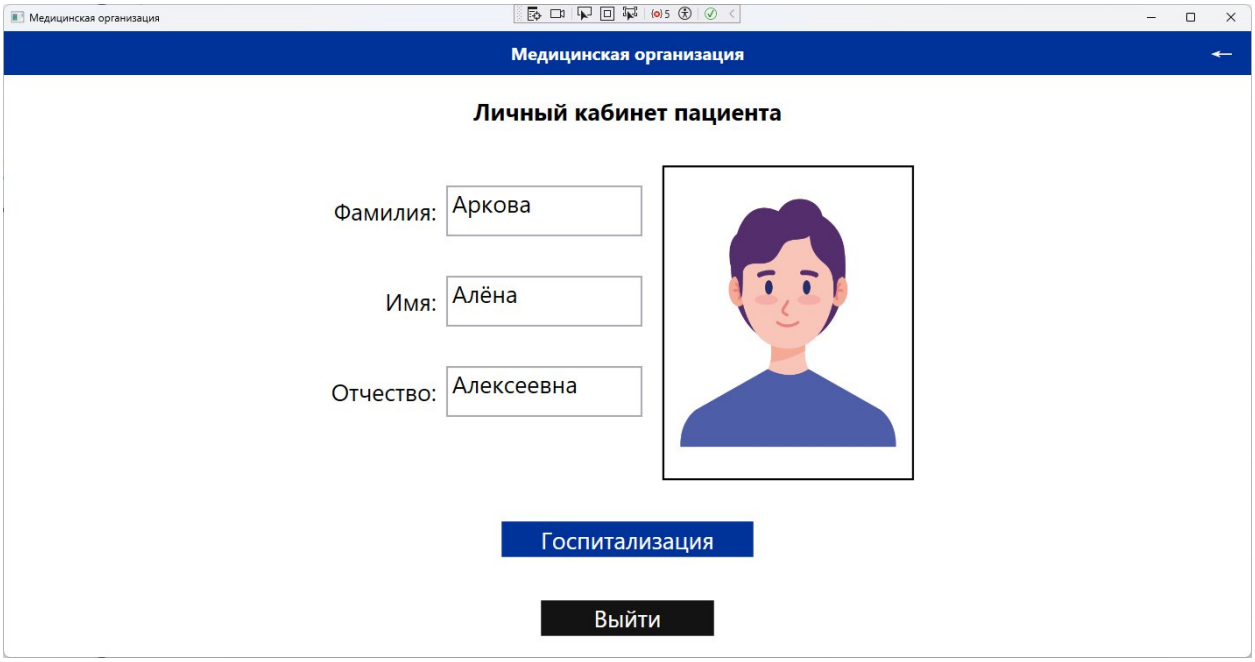

Рис.6. Личный кабинет

Далее, в зависимости от личного кабинет будут доступны разные функции, такие как: Госпитализация, Расписание, Лечебно-диагностические мероприятия.

Также важным процессом является расписание, с которым может ознакомиться как пациент, так и врач (Рис.7). Для её реализации необходимо наличие таблиц ReceptionSchedule и Doctors, из которых будут браться данными благодаря методу:

```
      private void LoadData()
              {
                  using (var context = new HospitalEntities9())
                  {
                      var schedule = context.ReceptionSchedule.ToList(); // Получаем все записи из таблицы Расписания
                      var doctors = context.Doctors.ToList(); // Получаем все записи из таблицы Врачи
                      DGridSchedule.ItemsSource = schedule.Join(doctors, rp => rp.IdDoctor, vr => vr.Id, (rp, vr) => new { ReceptionSchedule = rp, Doctors = vr }); // Устанавливаем источник данных для DataGrid
                  }
              }
```

Фамилия врача	Специализация	День недели	Время начала	Время окончания
Арбузова	Кардиолог	Понедельник	08:00:00	16:00:00
Бурлаков	Администратор	Понедельник	08:00:00	16:00:00
Варежкин	Хирург	Понедельник	08:00:00	16:00:00
Горшкова	Окулист	Понедельник	12:00:00	18:00:00
Дроздова	Кардиолог	Понедельник	12:00:00	18:00:00
Ершова	Терапевт	Понедельник	08:00:00	16:00:00
Жилков	Хирург	Понедельник	12:00:00	18:00:00
Зенина	Окулист	Понедельник	08:00:00	16:00:00
Ильина	Кардиолог	Понедельник	08:00:00	16:00:00
Краснова	Терапевт	Понедельник	08:00:00	16:00:00
Лисова	Хирург	Понедельник	12:00:00	18:00:00
Морозова	Окулист	Вторник	08:00:00	16:00:00
Назаров	Администратор	Вторник	12:00:00	18:00:00
Арбузова	Кардиолог	Вторник	08:00:00	16:00:00
Бурлаков	Администратор	Вторник	08:00:00	16:00:00
Варежкин	Хирург	Вторник	08:00:00	16:00:00
Горшкова	Окулист	Вторник	12:00:00	18:00:00
Дроздова	Кардиолог	Вторник	12:00:00	18:00:00
Ершова	Терапевт	Вторник	12:00:00	18:00:00
Жилков	Хирург	Вторник	08:00:00	16:00:00
Зенина	Окулист	Вторник	08:00:00	16:00:00
Ильина	Кардиолог	Вторник	08:00:00	16:00:00
Краснова	Терапевт	Вторник	12:00:00	18:00:00
Лисова	Хирург	Вторник	08:00:00	16:00:00
Морозова	Окулист	Вторник	12:00:00	18:00:00
Назаров	Администратор	Среда	08:00:00	16:00:00
Арбузова	Кардиолог	Среда	08:00:00	16:00:00
Бурлаков	Администратор	Среда	08:00:00	16:00:00
Варежкин	Хирург	Среда	08:00:00	16:00:00
Горшкова	Окулист	Среда	12:00:00	18:00:00
Дроздова	Кардиолог	Среда	12:00:00	18:00:00
Ершова	Терапевт	Среда	12:00:00	18:00:00
Жилков	Хирург	Среда	08:00:00	16:00:00
Зенина	Окулист	Среда	08:00:00	16:00:00
Ильина	Кардиолог	Среда	08:00:00	16:00:00
Краснова	Терапевт	Среда	12:00:00	18:00:00
Лисова	Хирург	Среда	08:00:00	16:00:00
Морозова	Окулист	Среда	12:00:00	18:00:00
Назаров	Администратор	Среда	08:00:00	16:00:00
Арбузова	Кардиолог	Четверг	08:00:00	16:00:00
Бурлаков	Администратор	Четверг	08:00:00	16:00:00
Варежкин	Хирург	Четверг	08:00:00	16:00:00
Горшкова	Окулист	Четверг	12:00:00	18:00:00
Дроздова	Кардиолог	Четверг	12:00:00	18:00:00
Ершова	Терапевт	Четверг	12:00:00	18:00:00
Жилков	Хирург	Четверг	08:00:00	16:00:00
Зенина	Окулист	Четверг	08:00:00	16:00:00

Рис.7. Расписание

Помимо метода сбора данных из БД, есть возможность сохранения в разных вариациях в проводник файлов с данными.

2.2 Описание интерфейса программы

Программа имеет простой и понятный интерфейс, состоящий из нескольких основных элементов. В верхней части окна располагается панель название и кнопка для возвращения назад.

Основную часть окна занимает рабочая область, в которой отображаются данные, с которыми работает пользователь.

Код визуальной части окна:

```
<Window x:Class="Hospital.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"
        mc:Ignorable="d"
        Title="Медицинская организация" MinHeight="450" MinWidth="800">
    <Grid>
        <Grid.RowDefinitions>
            <RowDefinition Height="50"/>
            <RowDefinition Height="188*"/>
        </Grid.RowDefinitions>
        <TextBlock Text="Медицинская организация"
HorizontalAlignment="Center" VerticalAlignment="Center"
Foreground="White" FontWeight="Bold" FontSize="20"/>
        <Button x:Name="TXTbutton" Content="←" Background="{x:Null}"
Width="30" Height="35" HorizontalAlignment="Right" Click="back"
FontSize="30" FontFamily="SimSun" BorderBrush="{x:Null}"
Foreground="White" Margin="20 0"/>

        <Grid Background="#003399" Panel.ZIndex="-2"
Grid.ColumnSpan="2"/>
        <Frame Grid.Row="1" ContentRendered="myFrame_ContentRendered"
NavigationUIVisibility="Hidden" Name="myFrame"/>
    </Grid>

</Window>
```

Рабочая область разделена на несколько страниц, каждая из которых соответствует определенному типу данных. Например, на странице «Госпитализация» отображаются данные пользователя с возможностью записи, а на странице «Расписание» весь недельный график работы врачей и другие (Рис.8).

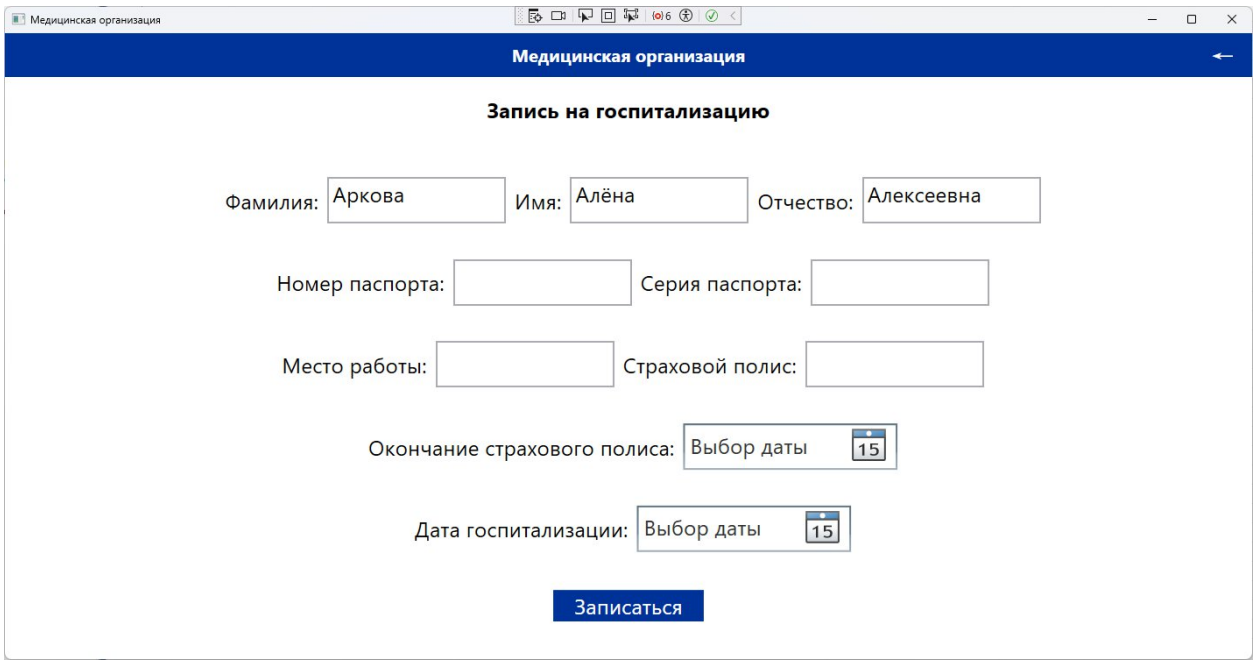

Рис.8. Госпитализация

Цветовая схема

В качестве основного фона используется белый цвет; в качестве дополнительного: #009933.

Для акцентирования внимания пользователя на целевое действие интерфейса используйте цвет #003399. Подробнее в таблице 2.

Таблица 2

Цветовая схема

Основной фон	Дополнительный	Акцентирование внимания
#FFFFFF	#009933	#003399

2.3 Тестирование готового продукта

По завершению работы создания структуры, следует её протестировать, для этого подойдёт отчёт о тестировании модулей функций системы, представленный в таблице 1

Таблица 1

Отчёт о тестировании модулей функций

№	Шаги	Ожидаемый результат	Результат тестирования
1	Запустить систему	На экран выходит начальный интерфейс	Вывод окна с первой страницей для входа.
2	Функция авторизации	При наличии номера медицинской карты открывается личный кабинет пользователя. При наличии логина и пароля открывается личный кабинет врача.	Открылся личный кабинет пациента. Открылся личный кабинет врача.
3	Функция регистрации	После заполнения всех полей выходит сообщение об успешной регистрации и переход в личный кабинет. При успешной регистрации данные сохранены в БД. При наличии не заполненного поля выводится ошибка.	Открылся личный кабинет пациента. Данные сохранились в БД. Вывелась ошибка.
4	Функция госпитализации	После заполнения всех полей выходит сообщение об успешной госпитализации. При успешной госпитализации данные сохранены в БД. При наличии не заполненного поля выводится ошибка.	Сообщение об успешной госпитализации. Данные сохранились в БД. Вывелась ошибка.
5	Функция кнопки назад	После нажатия на кнопку назад возврат на страницу назад.	Возврат на страницу назад
6	Функция отображения расписания	Расписание отобразилось.	Расписание отобразилось.
7	Функция отображения мероприятий	Лечебно-диагностические мероприятия отобразились.	Лечебно-диагностические мероприятия отобразились..

Результаты тестирования положительны и не требуют изменений.

Заключение

В ходе курсовой работы были применены навыки использования языка программирования C# и работа с БД.

Были изучены существующие решения и определены основные требования к системе. Была разработана архитектура системы, включающая в себя модули управления доступом, пользовательского интерфейса. Разработаны алгоритмы, сохранения данных в БД.

В заключении можно сказать, что разработанная система медицинской организации является эффективным инструментом для оптимизации работы медицинских учреждений. С помощью этой системы можно автоматизировать многие процессы, такие как запись на прием, управление расписанием врачей, ведение базы пациентов и т.д.

Однако, для полного раскрытия потенциала системы необходимо дальнейшее развитие и внедрение новых функций. В частности, необходимо разработать мобильное приложение, которое позволит пациентам получать информацию о своем здоровье и записываться на прием в любое время.

Таким образом, разработанная система медицинской организации представляет собой важный шаг в развитии медицины и здравоохранения. Ее использование позволяет улучшить качество обслуживания пациентов и повысить эффективность работы медицинских учреждений.

Разработанный пользовательский интерфейс был протестирован и признан удобным и понятным для пользователей. В результате выполнения курсовой работы была создана система медицинской организации, отвечающая всем требованиям и обеспечивающая высокий уровень безопасности и контроля доступа.

После того, как структура проекта была готова, она должна была быть протестирована. Итоговый вариант прошёл тесты успешно.

Таким образом, все поставленные задачи были выполнены, цель достигнута.

Список источников и литературы

1. Комлев Н.Ю. Объектно-ориентированное программирование. Хорошая книга для Хороших Людей. Н. Ю. Комлев – М.: СОЛОН-Пресс, 2014. – 298 с.

2. Эванс Э. Проектирование предметной области. Методология DDD. Эрик Эванс – Питер, 2010. – 448 с.

3. Кариев Ч. А. Технология Microsoft ADO.NET Entity Framework 4. Эндрю Д. Катулис – Питер, 2011. – 400 с.

4. Марк Прайс Современная кроссплатформенная разработка, 2023.

5. Фримен А. Программирование на C# 8 и .NET Core 3.0 для профессионалов. Адам Фримен – Питер, 2020. – 896 с.

6. Хитсон Д. SQL. Язык структурированных запросов. Джеймс Р. Хитсон – Издательский дом "Вильямс", 2018. – 544 с.

7. Хант Э. Программирование на C#. Эндрю Хант - Питер, 2016. - 848 с.

8. Маккалоу Дж., Рансом Дж. Entity Framework: Технология доступа ко всему спектру данных. Джулия Маккалоу, Ричард Рансом - Вильямс, 2015. - 432 с.

9. Стейл Л. Программирование в среде Windows с использованием Visual Studio. Ларс Стейл - Вильямс, 2018. - 416 с.

10. Харрисон М. Использование баз данных СУБД SQL Server. Мэтт Харрисон - Символ-Плюс, 2017. - 336 с.

11. Робертсон М., Киркхэм Т. Microsoft SQL Server 2017. Дэвид Ф. Робертсон, Брайан Киркхэм - ДМК Пресс, 2018. - 368 с.

12. Вон Вокс Р. Программирование Windows Presentation Foundation 4.5 на C#. Роджер Вон Вокс - Питер, 2014. - 736 с.

13. Метафреймс. Большая книга Visual Studio. Метафреймс - Ридерз Дайджест, 2019. - 384 с.

14. Вольчков В.Б. SQL Server 2019: Полное руководство. Владимир Б. Вольчков - БХВ-Петербург, 2021. - 1024 с.

15. Тепляков В.А. Entity Framework Core. Практическо-ориентированное руководство. Владимир А. Тепляков - Питер, 2020. - 496 с.

16. Фанер С. SQL Server 2019 для профессионалов. Стивен Фанер - ДМК Пресс, 2021. - 432 с.

17. Хантер А. C# 8.0 и .NET Core: Руководство для профессионалов. Эндрю Хантер - Питер, 2019. - 1104 с.

18. Кудряшов А.В. Базы данных. Проектирование и реализация с использованием Microsoft SQL Server. Андрей В. Кудряшов - Питер, 2013. - 544 с.

Приложение А

Программный код страницы Authorization.xaml:

```
<Page x:Class="Hospital.Authorization"
      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"
      mc:Ignorable="d"
      d:DesignHeight="400" d:DesignWidth="800"
      Title="Authorization">
    <Grid>
        <Viewbox Stretch="Uniform">
            <StackPanel>
                <TextBlock Text="Авторизация"
HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"
FontSize="{Binding ActualHeight, ElementName=mainWindow}" Margin="0 10"/>

                <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 40">
                    <Label Content="Номер медицинской карты:"/>
                    <TextBox x:Name="Txt_Card" MinWidth="100"
MaxWidth="180" Background="{x:Null}"/>
                </StackPanel>

                <Button Content="Вход" HorizontalAlignment="Center"
VerticalAlignment="Center" Width="60" Margin="0 10" BorderBrush="{x:Null}"
Background="#003399" Foreground="White" Click="LoginButton"/>
            </StackPanel>
        </Viewbox>
    </Grid>

</Page>
```

Программный код страницы Authorization.xaml.cs:

```
using System.Windows.Controls;
using System.Windows;
using Hospital.Model;
using System.Linq;
using System;

namespace Hospital
{
    public partial class Authorization : Page
    {
        private Frame myFrame;
        public Authorization()
        {
            InitializeComponent();
            AppConnect.HospitalModel = new HospitalEntities9(); //
подключение модели базы данных к странице авторизации
            AppFrame.frameMain = myFrame;
        }

        private void LoginButton(object sender, RoutedEventArgs e)
        {
            try
            {
```

```
                    int cardNumber;
                    if (int.TryParse(Txt_Card.Text, out cardNumber))
                    {
                        var userObj =
AppConnect.HospitalModel.Patients.FirstOrDefault(x => x.MedicalCardNumber ==
cardNumber);
                        if (userObj == null)
                        {
                            MessageBox.Show("Такого пациента не
существует.");
                        }
                        else
                        {
                            VariableClass.Surname = userObj.Surname;
                            VariableClass.Name = userObj.Name;
                            VariableClass.Patronymic = userObj.Patronymic;
                            Manager.myFrame.Navigate(new
PatientPersonalAccount());
                        }
                    }
                    else
                    {
                        MessageBox.Show("Некорректный номер медицинской
карты.");
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Ошибка! " + ex.Message + " Проверьте
введенные данные и повторите попытку.");
                }
            }
        }
    }
```

Программный код страницы AuthorizationDoctor.xaml:

```
    <Page x:Class="Hospital.AuthorizationDoctor"
          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"
          mc:Ignorable="d"
          d:DesignHeight="400" d:DesignWidth="800"
          Title="AuthorizationDoctor">

        <Grid>
            <Viewbox Stretch="Uniform">
                <StackPanel>
                    <TextBlock Text="Авторизация"
HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"
FontSize="{Binding ActualHeight, ElementName=mainWindow}" Margin="0 10"/>

                    <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                        <Label Content="Логин:"/>
                        <TextBox x:Name="Txt_Login" MinWidth="100"
MaxWidth="180" Background="{x:Null}"/>
                    </StackPanel>

                    <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
```

```
                        <Label Content="Пароль:"/>
                        <PasswordBox x:Name="Txt_Password" MinWidth="100"
MaxWidth="180" Background="{x:Null}"/>
                    </StackPanel>

                    <Button Content="Вход" HorizontalAlignment="Center"
VerticalAlignment="Center" Width="90" Margin="0 10" BorderBrush="{x:Null}"
Background="#003399" Foreground="White" Click="DoctorAuthorizationButton"/>
                </StackPanel>
            </Viewbox>
        </Grid>

    </Page>
```

Программный код класса VariableClass:

```
    using System;
    using System.Collections.Generic;
    using System.Linq;
    using System.Text;
    using System.Threading.Tasks;

    namespace Hospital
    {
        internal class VariableClass
    {
        public static string Surname, Name, Patronymic, Seria, Nomer,
Gender, Address, Phone, Mail, Work, Polis, Number;

    }
    }
```

Программный код страницы AuthorizationDoctor.xaml.cs:

```
    using System;
    using System.Linq;
    using System.Windows;
    using System.Windows.Controls;
    using Hospital.Model;

    namespace Hospital
    {
        public partial class AuthorizationDoctor : Page
        {
            private Frame myFrame;

            public AuthorizationDoctor()
            {
                InitializeComponent();
                AppConnect.HospitalModel = new HospitalEntities9();
                AppFrame.frameMain = myFrame;
            }

            private void DoctorAuthorizationButton(object sender,
RoutedEventArgs e)
            {
                try
                {
                    var user0bj =
AppConnect.HospitalModel.Doctors.FirstOrDefault(x => x.Login ==
Txt_Login.Text && x.Password == Txt_Password.Password); // Создание
динамической переменной user0bj, внутри которой будут храниться данные,
получаемые из заполняемых полей
```

```
                    if (user0bj == null)
                    {
                        MessageBox.Show("Такого пользователя нет.");
                    }
                    else
                    {
                        VariableClass.Surname = user0bj.Surname;
                        VariableClass.Name = user0bj.Name;
                        VariableClass.Patronymic = user0bj.Patronymic;
                        Manager.myFrame.Navigate(new
DoctorPersonalAccount());
                    }
                }
                catch (Exception Ex)
                {
                    MessageBox.Show("ОШИБКА! " + Ex.Message.ToString() + "
Сделайте всё заново и проверьте поля!");
                }
            }
        }
    }
```

Программный код страницы DoctorPersonalAccount.xaml:

```
<Page x:Class="Hospital.DoctorPersonalAccount"
          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"
          mc:Ignorable="d"
          d:DesignHeight="400" d:DesignWidth="800"
          Title="DoctorPersonalAccount" Background="White">

        <Grid Background="White">
            <Viewbox Stretch="Uniform">
                <StackPanel HorizontalAlignment="Center" >
                    <TextBlock Text="Личный кабинет врача"
HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"
FontSize="{Binding ActualHeight, ElementName=mainWindow}" Margin="0 10"/>
                    <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                        <StackPanel>
                            <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                <Label Content="Фамилия:"/>
                                <TextBox MinWidth="100" MaxWidth="180"
x:Name="Surname"/>
                            </StackPanel>

                            <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                <Label Content="Имя:"/>
                                <TextBox MinWidth="100" MaxWidth="180"
Width="125" x:Name="Name"/>
                            </StackPanel>

                            <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                <Label Content="Отчество:"/>
                                <TextBox MinWidth="100" MaxWidth="180"
x:Name="Patronymic"/>
                            </StackPanel>
```

```
                            <Button Content="Направление на лечебные
мероприятия" HorizontalAlignment="Center" Width="240" Margin="0 10"
BorderBrush="{x:Null}" Background="#003399" Foreground="White"
Click="HospitalizationButton"/>
                            <Button Content="Расписание приемов"
HorizontalAlignment="Center" Width="190" Margin="0 10" BorderBrush="{x:Null}"
Background="#003399" Foreground="White" Click="ScheduleButton"/>
                            <Button Content="Выйти"
HorizontalAlignment="Center" Width="82" Margin="0 10" BorderBrush="{x:Null}"
Background="#131313" Foreground="White" Click="ExitButton"/>
                        </StackPanel>
                    </StackPanel>
                </StackPanel>
            </Viewbox>
        </Grid>
    </Page>
```

Программный код страницы DoctorPersonalAccount.xaml.cs:

```
using System.Windows.Controls;
using System.Windows;

namespace Hospital
{
    public partial class DoctorPersonalAccount : Page
    {
        public DoctorPersonalAccount()
        {
            InitializeComponent();
            Surname.Text = VariableClass.Surname;
            Name.Text = VariableClass.Name;
            Patronymic.Text = VariableClass.Patronymic;
        }

        private void HospitalizationButton(object sender,
RoutedEventArgs e)
        {
            Manager.myFrame.Navigate(new MedicalDiagnosticProcedure());
        }

        private void ScheduleButton(object sender, RoutedEventArgs e)
        {
            Manager.myFrame.Navigate(new ReceptionSchedule());
        }

        public void ExitButton(object sender, RoutedEventArgs e)
        {
            Manager.myFrame.Navigate(new Main());
        }
    }
}
```

Программный код страницы PatientPersonalAccount.xaml:

```
<Page x:Class="Hospital.PatientPersonalAccount"
      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"
```

```
            mc:Ignorable="d"
            d:DesignHeight="400" d:DesignWidth="800"
            Title="PatientPersonalAccount">

          <Grid Background="White">
              <Viewbox Stretch="Uniform">
                  <StackPanel HorizontalAlignment="Center">
                      <TextBlock Text="Личный кабинет пациента"
HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"
FontSize="{Binding ActualHeight, ElementName=mainWindow}" Margin="0 10"/>
                      <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                          <StackPanel>
                              <StackPanel Orientation="Horizontal"
HorizontalAlignment="Right" Margin="0 10">
                                  <Label Content="Фамилия:"/>
                                  <TextBox MinWidth="100" MaxWidth="180"
x:Name="Surname"/>
                              </StackPanel>

                              <StackPanel Orientation="Horizontal"
HorizontalAlignment="Right" Margin="0 10">
                                  <Label Content="Имя:"/>
                                  <TextBox MinWidth="100" MaxWidth="180"
x:Name="Name"/>
                              </StackPanel>

                              <StackPanel Orientation="Horizontal"
HorizontalAlignment="Right" Margin="0 10">
                                  <Label Content="Отчество:"/>
                                  <TextBox MinWidth="100" MaxWidth="180"
x:Name="Patronymic"/>
                              </StackPanel>

                              <StackPanel Orientation="Horizontal"
HorizontalAlignment="Right" Margin="0 10"/>
                          </StackPanel>
                          <Border BorderBrush="Black" BorderThickness="1"
Width="128" Height="160" Margin="10 0">
                              <Image x:Name="Image_Patient"
Source="Resources/boy.png"/>
                          </Border>
```

```
                    </StackPanel>
                    <Button Content="Расписание приемов"
HorizontalAlignment="Center" VerticalAlignment="Center" Width="130" Margin="0
10" BorderBrush="{x:Null}" Background="#003399" Foreground="White"
Click="ScheduleButton"/>
                    <Button Content="Госпитализация"
HorizontalAlignment="Center" VerticalAlignment="Center" Width="110" Margin="0
10" BorderBrush="{x:Null}" Background="#003399" Foreground="White"
Click="HospitalizationButton"/>
                    <Button Content="Выйти" HorizontalAlignment="Center"
VerticalAlignment="Center" Width="90" Margin="0 10" BorderBrush="{x:Null}"
Background="#131313" Foreground="White" Click="ExitButton"/>
                </StackPanel>
            </Viewbox>
        </Grid>
    </Page>
```

Программный код страницы PatientPersonalAccount.xaml.cs:

```
using System.Windows.Controls;
using System.Windows;

namespace Hospital
{
    public partial class PatientPersonalAccount : Page
    {
        public PatientPersonalAccount()
        {
            InitializeComponent();
            Surname.Text = VariableClass.Surname;
            Name.Text = VariableClass.Name;
            Patronymic.Text = VariableClass.Patronymic;
        }

        private void ScheduleButton(object sender, RoutedEventArgs e)
        {
            Manager.myFrame.Navigate(new ReceptionSchedule());
        }

        public void HospitalizationButton(object sender, RoutedEventArgs
e)
        {
            Manager.myFrame.Navigate(new Hospitalization());
```

```
            }


            public void ExitButton(object sender, RoutedEventArgs e)
            {
                Manager.myFrame.Navigate(new Main());
            }
        }
    }
```

Программный код страницы MedicalDiagnosticProcedure.xaml.cs:

```
    using System.Windows.Controls;
    using Hospital.Model;
    using System.Linq;

    namespace Hospital
    {
        public partial class MedicalDiagnosticProcedure : Page
        {
            public MedicalDiagnosticProcedure()
            {
                InitializeComponent();
                LoadData();
            }

            private void LoadData()
            {
                using (var context = new HospitalEntities9())
                {
                    var treatments =
context.MedicalDiagnosticProcedure.ToList();
                    var patients = context.Patients.ToList();
                    DGridPatients.ItemsSource = treatments.Join(patients, t
=> t.Id, p => p.Id, (t, p) => new {MedicalDiagnosticProcedure = t, Patients =
p }).ToList();
                }
            }
        }
    }
```

Программный код страницы Hospitalization.xaml:

```
    <Page x:Class="Hospital.Hospitalization"
          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:Hospital"
          mc:Ignorable="d"
          d:DesignHeight="400" d:DesignWidth="800"
          Title="Hospitalization" Background="White">

        <Grid>
            <Grid>
                <Viewbox Stretch="Uniform">
                    <StackPanel HorizontalAlignment="Center">
```

```
                        <TextBlock Text="Запись на госпитализацию"
HorizontalAlignment="Center" VerticalAlignment="Center" FontWeight="Bold"
FontSize="{Binding ActualHeight, ElementName=mainWindow}" Margin="0 10"/>
                        <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                            <StackPanel>
                                <StackPanel Orientation="Horizontal"
HorizontalAlignment="Left" Margin="0 10">
                                    <Label Content="Фамилия:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Surname"/>
                                    <Label Content="Имя:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Name"/>
                                    <Label Content="Отчество:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Patronymic"/>
                                </StackPanel>

                                <StackPanel Orientation="Horizontal"
HorizontalAlignment="center" Margin="0 10">
                                    <Label Content="Номер паспорта:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Nomer"/>
                                    <Label Content="Серия паспорта:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Seria"/>
                                </StackPanel>

                                <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                    <Label Content="Место работы:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Work"/>
                                    <Label Content="Страховой полис:"/>
                                    <TextBox MinWidth="100" MaxWidth="180"
x:Name="Polis"/>
                                </StackPanel>
                                <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                    <Label Content="Окончание страхового
полиса:"/>
                                    <DatePicker Width="120"/>
                                </StackPanel>
                                <StackPanel Orientation="Horizontal"
HorizontalAlignment="Center" Margin="0 10">
                                    <Label Content="Дата госпитализации:"/>
                                    <DatePicker Width="120"
x:Name="Data_Hospitalization"/>
                                </StackPanel>

                                <Button Content="Записаться"
HorizontalAlignment="Center" Width="86" Margin="0 10" BorderBrush="{x:Null}"
Background="#003399" Foreground="White" Click="RecordButton"/>
                            </StackPanel>

                        </StackPanel>
                    </StackPanel>
                </Viewbox>
            </Grid>
        </Grid>
    </Page>
```

Программный код страницы Hospitalization.xaml.cs:

```
using System.Windows.Navigation;
using System.Windows.Controls;
using System.Windows;
using Hospital.Model;
using System.Linq;

namespace Hospital
{
    public partial class Hospitalization : Page
    {
        private Frame myFrame;

        public Hospitalization()
        {
            InitializeComponent();
            AppConnect.HospitalModel = new HospitalEntities9();
            AppFrame.frameMain = myFrame;

            Patronymic.Text = VariableClass.Patronymic;
            Surname.Text = VariableClass.Surname;
            Seria.Text = VariableClass.Seria;
            Nomer.Text = VariableClass.Nomer;
            Polis.Text = VariableClass.Polis;
            Name.Text = VariableClass.Name;
            Work.Text = VariableClass.Work;
        }

        // Метод поиска Id пациента по фамилии
        private int GetPatientIdBySurname(string surname)
        {
            var patient =
AppConnect.HospitalModel.Patients.FirstOrDefault(p => p.Surname == surname);
            return patient != null ? patient.Id : -1;
        }

        private void RecordButton(object sender, RoutedEventArgs e)
        {
            if (Surname.Text == "" || Name.Text == "" || Patronymic.Text
== "" || Nomer.Text == "" || Seria.Text == "" || Work.Text == "" ||
Polis.Text == "")
            {
                MessageBox.Show("Заполните все поля.");
            }
            else
            {
                int patientId = GetPatientIdBySurname(Surname.Text);
                if (patientId != -1)
                {
                    HistoryHospitalizations history = new
HistoryHospitalizations();
                    history.IdPatient = patientId;
                    history.DateHospitalization =
Data_Hospitalization.SelectedDate.Value;

                    if (AppConnect.HospitalModel != null)
                    {

AppConnect.HospitalModel.HistoryHospitalizations.Add(history);
                        AppConnect.HospitalModel.SaveChanges();
                        MessageBox.Show("Дата госпитализации
сохранена!");
```

```
                            NavigationService.Navigate(new
PatientPersonalAccount());
                        }
                        else
                        {
                            MessageBox.Show("Ошибка в подключении к базе
данных.");
                        }
                    }
                }
            }
        }
```

Программный код страницы Main.xaml:

```
    <Page x:Class="Hospital.Main"
          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"
          mc:Ignorable="d"
          d:DesignHeight="400" d:DesignWidth="800"
          Title="Main">

        <Grid>
            <Viewbox Stretch="Uniform">
                <StackPanel>
                    <TextBlock Text="Если вы у нас первый раз, просьба
пройти регистрацию&#xA;и ознакомиться с договором и согласием"
HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="{Binding
ActualHeight, ElementName=mainWindow}" Margin="90 20" TextAlignment="Center"
FontStyle="Italic"/>
                    <Button Content="Вход" HorizontalAlignment="Center"
Width="90" Margin="0 5" BorderBrush="{x:Null}" Background="#003399"
Foreground="White" Click="LoginButton" />
                    <Button Content="Регистрация"
HorizontalAlignment="Center" Width="90" Margin="0 5" BorderBrush="{x:Null}"
Background="#009933" Foreground="White" Click="RegisterButton"/>
                    <Button Content="Войти как врач"
HorizontalAlignment="Center" Width="90" Margin="0 5" BorderBrush="#FFECE6E6"
Click="LoginDoctorButton" FontSize="8" Background="{x:Null}"
Foreground="#FF807E7E"/>
                </StackPanel>
            </Viewbox>
        </Grid>

    </Page>
```

Программный код страницы Main.xaml.cs:

```
    using System.Windows.Controls;
    using System.Windows;

    namespace Hospital
    {
        public partial class Main : Page
        {
            public Main()
            {
                InitializeComponent();
            }
```

```
            public void LoginButton(object sender, RoutedEventArgs e)
            {
                Manager.myFrame.Navigate(new Authorization());
            }

            public void RegisterButton(object sender, RoutedEventArgs e)
            {
                Manager.myFrame.Navigate(new Registration());
            }

            public void LoginDoctorButton(object sender, RoutedEventArgs e)
            {
                Manager.myFrame.Navigate(new AuthorizationDoctor());
            }
        }
    }
```

Программный код страницы MedicalDiagnosticProcedure.xaml:

```
    <Page x:Class="Hospital.MedicalDiagnosticProcedure"
          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"
          mc:Ignorable="d"
          d:DesignHeight="400" d:DesignWidth="800"
          Title="MedicalDiagnosticProcedure">

        <Grid Background="White">
            <StackPanel Orientation="Vertical">
                <TextBlock Text="История лечебно-диагностических
мероприятий" HorizontalAlignment="Center"  Height="30"
VerticalAlignment="Top" FontWeight="Bold" FontSize="24"/>
                <DataGrid x:Name="DGridPatients" AutoGenerateColumns="False"
IsReadOnly="True" HorizontalAlignment="Stretch">
                    <DataGrid.Columns>
                        <DataGridTextColumn Binding="{Binding
Patients.Surname}" FontSize="20" Header="Фамилия"
Width="140"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
Patients.MedicalCardNumber}" FontSize="20" Header="Медицинская карта"
Width="130"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
MedicalDiagnosticProcedure.EventType}" FontSize="20" Header="Диагностические
мероприятия" Width="*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
MedicalDiagnosticProcedure.Results}" FontSize="20" Header="Результаты"
Width="*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
MedicalDiagnosticProcedure.Recommendations}" FontSize="20"
Header="Рекомендации" Width="*"></DataGridTextColumn>

                        <DataGridTemplateColumn Width="200">
                            <DataGridTemplateColumn.CellTemplate>
                                <DataTemplate>
                                    <Button
Content="Редактировать"></Button>
                                </DataTemplate>
                            </DataGridTemplateColumn.CellTemplate>
                        </DataGridTemplateColumn>
                    </DataGrid.Columns>
```

```
                </DataGrid>
            </StackPanel>
        </Grid>
    </Page>
```

Программный код страницы MedicalDiagnosticProcedure.xaml.cs:

```
    using System.Windows.Controls;
    using Hospital.Model;
    using System.Linq;

    namespace Hospital
    {
        public partial class MedicalDiagnosticProcedure : Page
        {
            public MedicalDiagnosticProcedure()
            {
                InitializeComponent();
                LoadData();
            }

            private void LoadData()
            {
                using (var context = new HospitalEntities9())
                {
                    var treatments =
context.MedicalDiagnosticProcedure.ToList();
                    var patients = context.Patients.ToList();
                    DGridPatients.ItemsSource = treatments.Join(patients, t
=> t.Id, p => p.Id, (t, p) => new {MedicalDiagnosticProcedure = t, Patients =
p }).ToList();
                }
            }
        }
    }
```

Программный код страницы ReceptionSchedule.xaml:

```
    <Page x:Class="Hospital.ReceptionSchedule"
          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"
          mc:Ignorable="d"
          d:DesignHeight="450" d:DesignWidth="800"
          Title="ReceptionSchedule">

        <Grid Background="White">
            <StackPanel Orientation="Vertical">
                <StackPanel HorizontalAlignment="Center"
Orientation="Horizontal">
                    <TextBlock Text="Расписание приёмов"
HorizontalAlignment="Center"  Height="30" VerticalAlignment="Top"
FontWeight="Bold" FontSize="24"/>
                    <Button Content="Экспортировать в xlsx"
HorizontalAlignment="Right" Width="130" Height="30" Margin="100 5 10 5"
BorderBrush="{x:Null}" Background="#009933" Foreground="White"
Click="ExportXlsxButton"/>
                    <Button Content="Экспортировать в csv"
HorizontalAlignment="Right" Width="130" Height="30" Margin="0 5 10 5"
BorderBrush="{x:Null}" Background="#009933" Foreground="White"
Click="ExportCsvButton"/>
```

```
                    <Button Content="Экспортировать в doc"
HorizontalAlignment="Right" Width="130" Height="30" Margin="0 5 10 5"
BorderBrush="{x:Null}" Background="#009933" Foreground="White"
Click="ExportDocButton"/>
                </StackPanel>
                <DataGrid x:Name="DGridSchedule" AutoGenerateColumns="False"
IsReadOnly="True" HorizontalAlignment="Stretch">
                    <DataGrid.Columns>
                        <DataGridTextColumn Binding="{Binding
Doctors.Surname}" Header="Фамилия врача" Width="130*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
Doctors.Specialization}" Header="Специализация"
Width="150*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
ReceptionSchedule.WeekDay}" Header="День недели"
Width="120*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
ReceptionSchedule.StartTime}" Header="Время начала"
Width="120*"></DataGridTextColumn>
                        <DataGridTextColumn Binding="{Binding
ReceptionSchedule.EndTime}" Header="Время окончания"
Width="130*"></DataGridTextColumn>
                    </DataGrid.Columns>
                </DataGrid>
            </StackPanel>
        </Grid>
    </Page>
```

Программный код страницы ReceptionSchedule.xaml.cs:

```
using DocumentFormat.OpenXml.Wordprocessing;
using DocumentFormat.OpenXml.Packaging;
using System.Windows.Controls;
using DocumentFormat.OpenXml;
using System.Data.SqlClient;
using ClosedXML.Excel;
using System.Windows;
using Hospital.Model;
using System.Linq;
using System.Data;
using System.Xml;
using System;

namespace Hospital
{
    public partial class ReceptionSchedule : Page
    {
        private System.Windows.Controls.Frame myFrame;

        private void LoadData()
        {
            using (var context = new HospitalEntities9())
            {
                var schedule = context.ReceptionSchedule.ToList(); //
Получаем все записи из таблицы Расписания
                var doctors = context.Doctors.ToList(); // Получаем все
записи из таблицы Врачи
                DGridSchedule.ItemsSource = schedule.Join(doctors, rp =>
rp.IdDoctor, vr => vr.Id, (rp, vr) => new { ReceptionSchedule = rp, Doctors =
vr }); // Устанавливаем источник данных для DataGrid
            }
        }
```

```
            public ReceptionSchedule()
            {
                InitializeComponent();
                AppConnect.HospitalModel = new HospitalEntities9();
                AppFrame.frameMain = myFrame;
                LoadData();
            }

            private void ExportXlsxButton(object sender, RoutedEventArgs e)
            {
                string connectionString =
"Server=SPC\\STP;Database=Hospital;Integrated Security=True;";
                string query = "SELECT Patients.Name, Patients.Surname,
HistoryHospitalizations.DateHospitalization,
HistoryHospitalizations.ReleaseDate,
HistoryHospitalizations.ReasonHospitalization,
HistoryHospitalizations.DescriptionState\r\nFROM Patients\r\nJOIN
HistoryHospitalizations ON Patients.Id = HistoryHospitalizations.IdPatient;";
                string filePath = "D:\\ReceptionSchedule.xlsx";

                using (SqlConnection connection = new
SqlConnection(connectionString))
                {
                    SqlCommand command = new SqlCommand(query, connection);
                    SqlDataAdapter dataAdapter = new
SqlDataAdapter(command);
                    DataTable dataTable = new DataTable();

                    dataAdapter.Fill(dataTable);
                    using (XLWorkbook workbook = new XLWorkbook())
                    {
                        var worksheet = workbook.Worksheets.Add("Sheet1");

                        for (int i = 0; i < dataTable.Columns.Count; i++)
                            worksheet.Cell(1, i + 1).Value =
dataTable.Columns[i].ColumnName;

                        for (int i = 0; i < dataTable.Rows.Count; i++)
                            for (int j = 0; j < dataTable.Columns.Count;
j++)
                                worksheet.Cell(i + 2, j + 1).Value =
dataTable.Rows[i][j].ToString();

                        workbook.SaveAs(filePath);
                        MessageBox.Show("Данные успешно сохранены в файл: "
+ filePath);
                    }
                }
            }

            private void ExportCsvButton(object sender,
System.Windows.RoutedEventArgs e)
            {
                string connectionString =
"Server=SPC\\STP;Database=Hospital;Integrated Security=True;";
                string query = "SELECT Patients.Name, Patients.Surname,
HistoryHospitalizations.DateHospitalization,
HistoryHospitalizations.ReleaseDate,
HistoryHospitalizations.ReasonHospitalization,
HistoryHospitalizations.DescriptionState\r\nFROM Patients\r\nJOIN
HistoryHospitalizations ON Patients.Id = HistoryHospitalizations.IdPatient;";
                string filePath = "D:\\ReceptionSchedule.csv";
```

```
                try
                {
                    using (XmlWriter writer = XmlWriter.Create(filePath))
                    {
                        writer.WriteStartDocument();
                        writer.WriteStartElement("Data");


                        using (SqlConnection connection = new
SqlConnection(connectionString))
                        {
                            connection.Open();
                            using (SqlCommand command = new
SqlCommand(query, connection))
                            {
                                using (SqlDataReader reader =
command.ExecuteReader())
                                {
                                    while (reader.Read())
                                    {
                                        writer.WriteStartElement("Row");
                                        for (int i = 0; i <
reader.FieldCount; i++)

writer.WriteElementString(reader.GetName(i), reader[i].ToString());
                                        writer.WriteEndElement();
                                    }
                                }
                            }
                        }
                        writer.WriteEndElement();
                        writer.WriteEndDocument();
                    }
                    MessageBox.Show("Данные успешно сохранены в файл: " +
filePath);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Произошла ошибка: " + ex.Message);
                }
            }

            private void ExportDocButton(object sender, RoutedEventArgs e)
            {
                string connectionString =
"Server=SPC\\STP;Database=Hospital;Integrated Security=True;";
                string query = "SELECT Patients.Name, Patients.Surname,
HistoryHospitalizations.DateHospitalization,
HistoryHospitalizations.ReleaseDate,
HistoryHospitalizations.ReasonHospitalization,
HistoryHospitalizations.DescriptionState\r\nFROM Patients\r\nJOIN
HistoryHospitalizations ON Patients.Id = HistoryHospitalizations.IdPatient;";
                string filePath = "D:\\ReceptionSchedule.docx";

                try
                {
                    using (WordprocessingDocument wordDocument =
WordprocessingDocument.Create(filePath, WordprocessingDocumentType.Document))
                    {
                        MainDocumentPart mainPart =
wordDocument.AddMainDocumentPart();
                        mainPart.Document = new Document();
                        Body body = new Body();
```

```
                        mainPart.Document.Append(body);
                        using (SqlConnection connection = new
SqlConnection(connectionString))
                        {
                            connection.Open();
                            SqlCommand command = new SqlCommand(query,
connection);
                            SqlDataAdapter dataAdapter = new
SqlDataAdapter(command);
                            DataTable dataTable = new DataTable();
                            dataAdapter.Fill(dataTable);

                            foreach (DataRow row in dataTable.Rows)
                            {
                                Paragraph para = new Paragraph();
                                para.Append(new Run(new Text(
                                    $"Name: {row["Name"]}," +
                                    $"Surname: {row["Surname"]}," +
                                    $"DateHospitalization:
{row["DateHospitalization"]}," +
                                    $"ReleaseDate: {row["ReleaseDate"]}," +
                                    $"ReasonHospitalization:
{row["ReasonHospitalization"]}," +
                                    $"DescriptionState:
{row["DescriptionState"]},\n")));
                                body.Append(para);
                            }
                        }
                        MessageBox.Show("Данные успешно сохранены в файл: "
+ filePath);
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show($"Произошла ошибка при сохранении данных
в документ Word: {ex.Message}");
                }
            }
        }
    }
```

Программный код окна MainWindow.xaml.cs:

```
using System.Windows;
using System;

namespace Hospital
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            myFrame.Navigate(new Main());
            Manager.myFrame = myFrame;
        }

        public void back(object sender, RoutedEventArgs e)
        {
            Manager.myFrame.GoBack();
        }
```

```
        // Отображение/отключение кнопки возвращения
        private void myFrame_ContentRendered(object sender, EventArgs e)
        {
            if (myFrame.CanGoBack)
                TXTbutton.Visibility = Visibility.Visible;
            else
                TXTbutton.Visibility = Visibility.Hidden;
        }
    }
}
```