Государственное бюджетное профессиональное образовательное учреждение

Самарской области

«Тольяттинский социально-экономический колледж»

**КУРСОВАЯ РАБОТА**

**РАЗРАБОТКА И ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОЙ СИСТЕМЫ «УЧЕТ ПАЦИЕНТОВ В ПОЛИКЛИНИКЕ»»**

**ПМ.05 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ИНФОРМАЦИОННЫХ СИСТЕМ**

**МДК 05.02 Разработка кода информационных систем**

**09.02.07 Информационные системы и программирование**

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  | **/** | В.С. Ермаков |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |
| Оценка выполнения и защиты курсовой работы | | |  |
|  | | |  |
| Руководитель |  | **/** | Е.В. Плюснина |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |

Тольятти, 2022

Государственное бюджетное профессиональное образовательное учреждение  
Самарской области  
«Тольяттинский социально-экономический колледж»

|  |  |  |
| --- | --- | --- |
|  |  | Утверждаю:  Заместитель директора по УР  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.С. Киронова  *« » 202 г.* |

**ЗАДАНИЕ**

на курсовую работу

**по ПМ.05 Проектирование и разработка информационных систем модуля, выполняемой в рамках МДК.05.02 Разработка кода информационных систем**

студента группы ИСП-31

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Ермакова Вадима Сергеевича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Фамилия Имя отчество студента*

Тема курсовой работы: «Разработка информационной системы **«**Учёт пациентов в поликлинике»

1. Содержание задания:

1.1 Разработать физическую структуру информационной системы.

1.2 Разработать запросы к информационной системе.

1.3 Разработать приложение.

1. Исходные данные:

Исходные данные для практической реализации автоматизированной информационной системы (АИС) берутся из различных информационных источников (Интернет-ресурсы, печатные издания, периодика и др.).

1. Содержание курсовой работы

Введение

1 Разработка базы данных информационной системы (название)

1.1 Разработка физической структуры БД

1.2 Разработка запросов ИС

2 Разработка интерфейса информационной системы (название)

2.1 Разработка интерфейса

2.2 Разработка руководства пользователя

Заключение

Список использованных источников

Приложения

Приложение А Листинг программы

Дата выдачи задания: «13» января 2022 г.

Дата сдачи работы на отделение: «\_\_»\_\_\_\_\_\_\_\_2022 г.

Руководитель курсового(ой) проекта(работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Е.В. Плюснина

подпись расшифровка подписи

**КАЛЕНДАРНЫЙ ПЛАН**

выполнения курсовой работы

Студентом 3 курса группы ИСП-31

По теме Разработка и проектирование ИС «Учет пациентов в поликлинике»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| №  этапа  работы | Содержание этапов работы | Плановый срок выполнения этапа | Планируемый объем выполнения  этапа, % | Отметка  о  выполнении  этапа |
| 1 | Выбор, обоснование темы и объекта исследования | Январь 2022 | 5% |  |
| 2 | Утверждение темы, согласование плана. Введение, библиография | Январь 2022 | 10% |  |
| 3 | Изучение и анализ информационных материалов по теме | Февраль 2022 | 15% |  |
| 4 | Обоснование актуальности выбранной темы применительно к профессиональной деятельности (введение) | Февраль 2022 | 20% |  |
| 5 | Изложение материала основной части по теме курсовой работы | Февраль 2022 | 20% |  |
| 6 | Подведение итогов проведенного анализа, формулировка выводов УИР применительно к профессиональной деятельности (заключение) | Март 2022 | 20% |  |
| 7 | Оформление работы и сдача на проверку | Март 2022 | 10% |  |
| 8 | Защита работы |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент |  | **/** | В.С. Ермаков |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |
| Руководитель |  | **/** | Е.В. Плюснина |
|  | *подпись* |  | *И.О. Фамилия* |
| \_\_.\_\_.2022 г. |  |  |  |

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 5](#__RefHeading___Toc94035240)

[1 РАЗРАБОТКА БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТ ПАЦИЕНТОВ В ПОЛИКЛИНИКЕ 7](#__RefHeading___Toc94035241)

[1.1 Разработка физической структуры БД 7](#__RefHeading___Toc94035242)

[1.2 Разработка запросов ИС 10](#__RefHeading___Toc94035243)

[2 РАЗРАБОТКА ИНТЕРФЕЙСА ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТ ПАЦИЕНТОВ В ПОЛИКЛИНИКЕ 12](#__RefHeading___Toc94035244)

2.1 Разработка интерфейса ……………………………………………………. 12

2.2 Разработка руководства пользователя ………………………….……….... 19

[ЗАКЛЮЧЕНИЕ 21](#__RefHeading___Toc94035246)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 22](#__RefHeading___Toc94035247)

[ПРИЛОЖЕНИЕ А.](#__RefHeading___Toc94035248) [ЛИСТИНГ ПРОГРАММЫ 24](#__RefHeading___Toc94035249)

# ВВЕДЕНИЕ

# Одной из актуальных проблем любого вида человеческой деятельности является проблема обработки увеличивающегося потока информации.

# Современные медицинские организации производят и накапливают огромные объемы данных. От того, насколько эффективно эта информация используется врачами, руководителями, управляющими органами, зависит качество медицинской помощи, общий уровень жизни населения.

# Говоря о региональной информатизации здравоохранения, крайне важно оценить состояние текущих крупных региональных проектов, ресурсный потенциал, необходимый для их реализации, состояние телекоммуникаций. При этом необходимо иметь в виду, что сегодня акцент постепенно смещается в сторону реализации проектов информатизации медицинской деятельности. То есть приоритеты начали меняться в сторону таких направлений, как автоматизация рабочего места врача, создание единой электронной медицинской карты, распространение интеллектуальных систем поддержки принятия административных решений, применение и широкое распространение новейших медицинских технологий, развитие телемедицины.

# Перспективным способом решения данной проблемы является автоматизация работы с информацией, в частности, создание компьютерных баз данных, позволяющих хранить, систематизировать и обрабатывать данные.

# При изучении предметной области, выяснилось, что информационная система «Учет пациентов в поликлинике» на первый взгляд простая, но для ее реализации необходимо обратится к некоторым разделам, таким как создание базы данных и создание приложения для взаимодействия с этой базой данных.

# Объектом исследования курсовой работы является процесс учета пациентов в стоматологической поликлинике.

# Предметом исследования процесса будет ИС «Учет пациентов в поликлинике».

# Вышеизложенное в целом на теоретико-методологическом уровне определило проблему настоящего исследования: разработка программного модуля «Учет пациентов» в программе MS Visual Studio 2019.

# Целью курсовой работы является разработка и проектирование информационной системы «Учет пациентов в поликлинике».

# Для достижения поставленной цели были определены следующие задачи:

# разработать базы данных информационной системы;

# разработать физическую структуру базы данных;

# разработать запросы ИС;

# разработать интерфейс ИС;

# разработать руководство пользователя.

# Для разработки информационной системы будут применяться методы: анализ деятельности предприятия с выявлением его функции, а также функциональное моделирование системы.

# Практическая значимость заключается в использовании разработанной системы на различных предприятиях по «Учета пациентов».

# Структура работы соответствует логике исследования и включает в себя введение, две главы, заключение, список использованной литературы, приложение.

# 1 РАЗРАБОТКА БАЗЫ ДАННЫХ ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТ ПАЦИЕНТОВ В ПОЛИКЛИНИКЕ

## 1.1 Разработка физической структуры БД

Чтобы создать новую базу данных, откройте SQL Server Management Studio 18. В обозревателе объектов подключитесь к экземпляру компонента Компонент SQL Server Database Engine и разверните его. Щелкните правой кнопкой мыши узел Базы данных и выберите команду создать базу данных. В поле Новая база данных введите имя базы данных. Чтобы создать базу данных, приняв все значения по умолчанию, нажмите кнопку ОК.

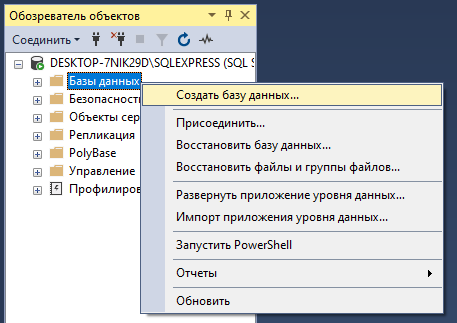


Рисунок 1.1 – Окно обозреватель объектов

В обозревателе объектов открываем контейнер «Базы данных», затем открываем нужную базу данных и щелкаем правой кнопкой мыши по пункту «Таблицы», после «Создать» и выбираем «Таблица…».

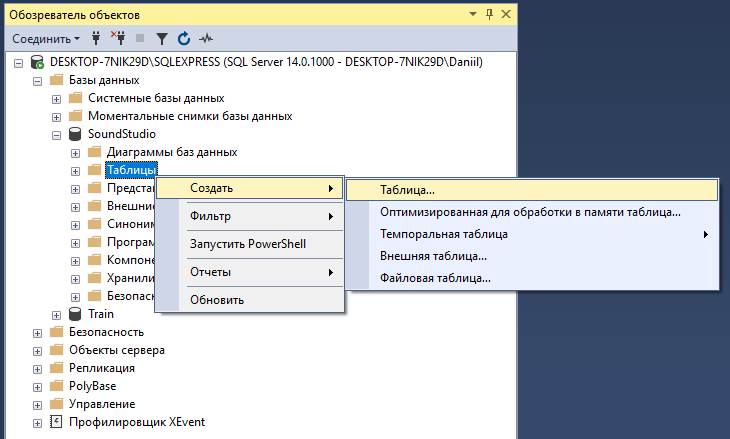


Рисунок 1.2 – Окно создания таблицы

Введите имена столбцов, выберите типы данных и определите для каждого столбца, могут ли в нем присутствовать значения NULL, как показано на следующей иллюстрации:

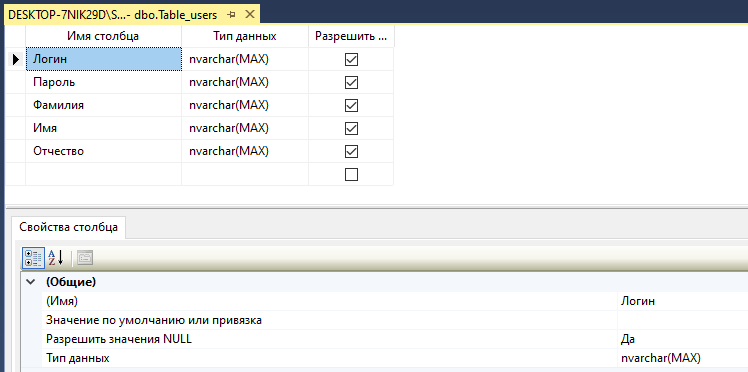


Рисунок 1.3 – Таблица «Пользователи»

Вы также можете задать другие свойства столбца, например, является ли этот столбец столбцом идентификаторов или вычисляемым столбцом. Для этого щелкните столбец на вкладке свойств столбцов. Чтобы указать, что столбец является столбцом первичного ключа, щелкните его правой кнопкой мыши и выберите «Задать первичный ключ». Чтобы создать связи по внешнему ключу, проверочные ограничения или индексы, щелкните правой кнопкой мыши панель конструктора таблиц и выберите в списке объект, как показано на следующей иллюстрации:

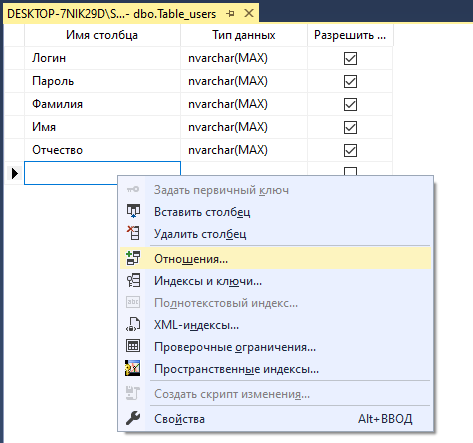


Рисунок 1.4 – Окно создания связей по внешнему ключу

По умолчанию таблица содержится в схеме dbo. Чтобы указать другую схему для таблицы, щелкните правой кнопкой мыши панель конструктора таблиц и выберите «Свойства», как показано на следующей иллюстрации:

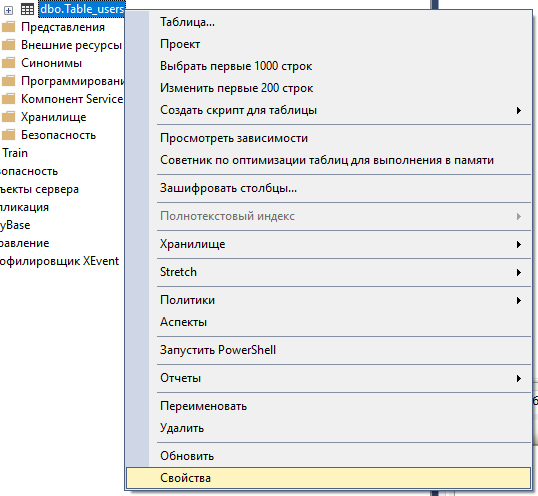


Рисунок 1.5 – Окно выбора свойств таблицы «Пользователи»

Выберите нужную схему из раскрывающегося списка «Схема».

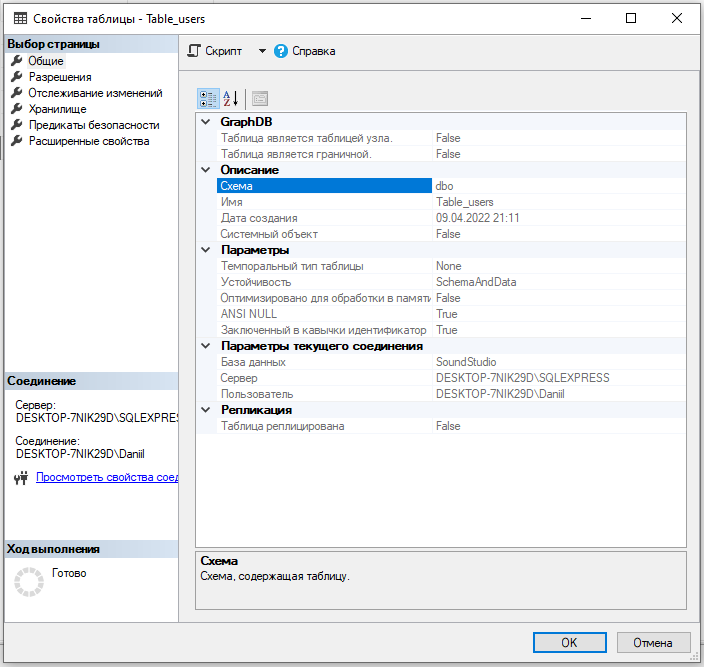


Рисунок 1.6 – Окно свойств таблицы «Пользователи»

В меню Файл выберите команду «Сохранить» и введите название таблицы – «имя\_таблицы».

Чтобы просмотреть новую таблицу, в обозревателе объектов разверните узел Таблицы, а затем нажмите клавишу F5, чтобы обновить список объектов. Новая таблица будет отображена в списке таблиц.

Создание остальных таблиц базы данных для информационной системы происходит аналогично примеру, описанному выше.

## 1.2 Разработка запросов ИС

Чтобы создать представление, в обозревателе объектов открываем контейнер «Базы данных», затем открываем нужную базу данных и щелкаем правой кнопкой мыши по пункту «Представления», после «Создать представление…».

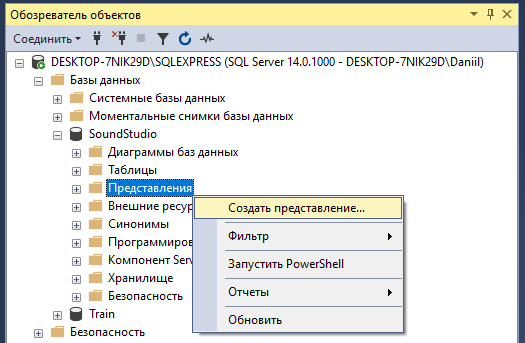


Рисунок 1.7 – Окно создания представления

В диалоговом окне «Добавление таблицы» выберите один или несколько элементов, которые необходимо включить в новое представление, на одной из следующих вкладок: «Таблицы», «Представления», «Функции» и «Синонимы». Щелкните «Добавить», а затем выберите Закрыть.

На Панели диаграмм выберите столбцы или другие элементы для включения в новое представление. На Панели критериев выберите дополнительные условия сортировки или фильтрации для столбцов.

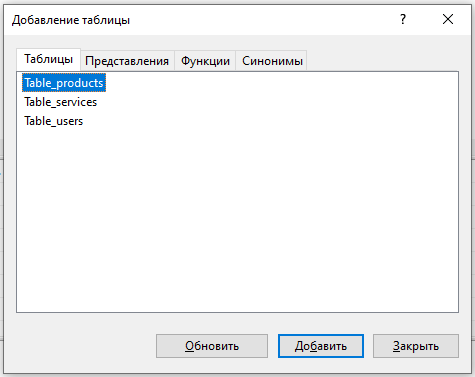


Рисунок 1.8 – Диалоговое окно добавление таблицы

В меню Файл выберите пункт «Сохранить» и введите название представления – «имя\_представления».

Затем создадим запрос для информационной системы путем выбора столбцов или других элементов представления. На панели критериев выберите дополнительные условия сортировки или фильтрации для столбцов.

Общая структура запроса выглядит следующим образом:

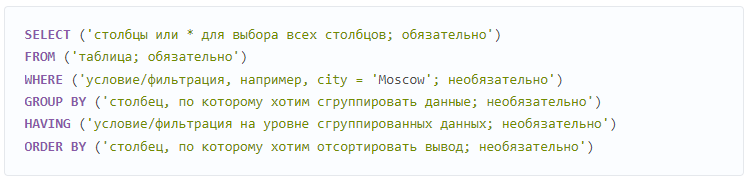


Рисунок 1.9 – Общая структура запроса

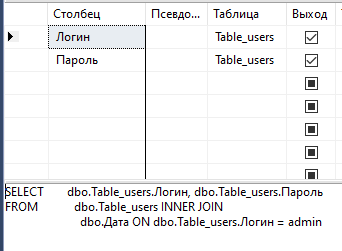


Рисунок 1.10 – Запрос Select для информационной системы

# Создание остальных запросов базы данных для информационной системы происходит аналогично примеру, описанному выше.

# 2 РАЗРАБОТКА ИНТЕРФЕЙСА ИНФОРМАЦИОННОЙ СИСТЕМЫ УЧЕТ ПАЦИЕНТОВ В ПОЛИКЛИНИКЕ

## 2.1 Разработка интерфейса

Первым шагом является создание инфраструктуры приложения. Создайте новый проект приложения WPF в Visual Basic или Visual C# и введите название. Откройте Visual Studio и выберите создать новый проект в меню начало работы. Откроется диалоговое окно «Создание нового проекта». В раскрывающемся списке язык выберите C#. Выберите шаблон приложения WPF (платформа .NET Framework) и нажмите кнопку далее.

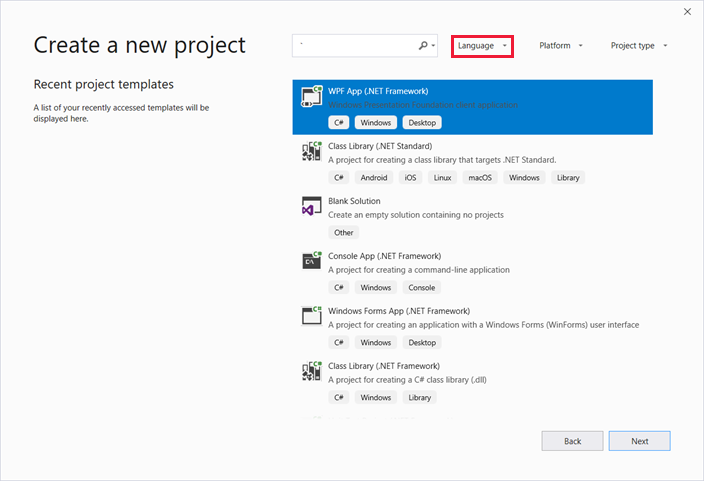


Рисунок 2.1 – Окно создания приложения WPF

Откроется диалоговое окно «Настройка нового проекта». Введите имя проекта и нажмите кнопку создать.

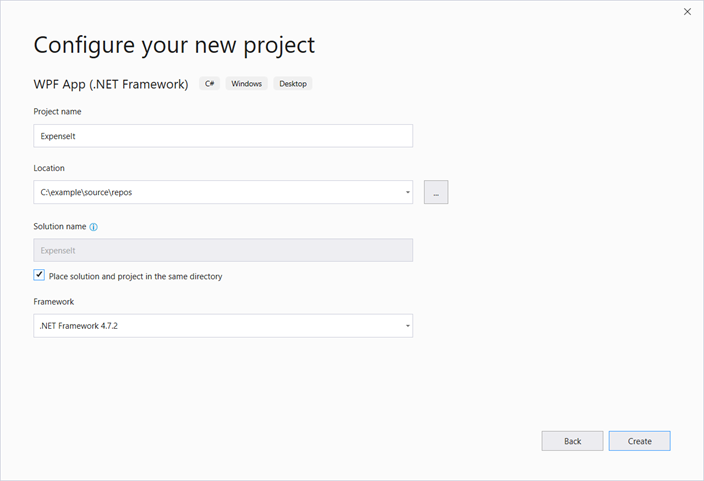


Рисунок 2.2 – Окно создания приложения WPF

Visual Studio создает проект и открывает конструктор для окна приложения по умолчанию с именем MainWindow.xaml.

Чтобы как-то взаимодействовать с пользователем, получать от пользователя ввод с клавиатуры или мыши и использовать введенные данные в программе, нам нужны элементы управления.

Создаем интерфейс окна авторизации, используя элементы управления WPF – Grid, Label, Border для визуального восприятия информации; TextBox, PasswordBox для ввода логина и пароля; Button для просмотра пароля, закрытия окна и для входа в систему.

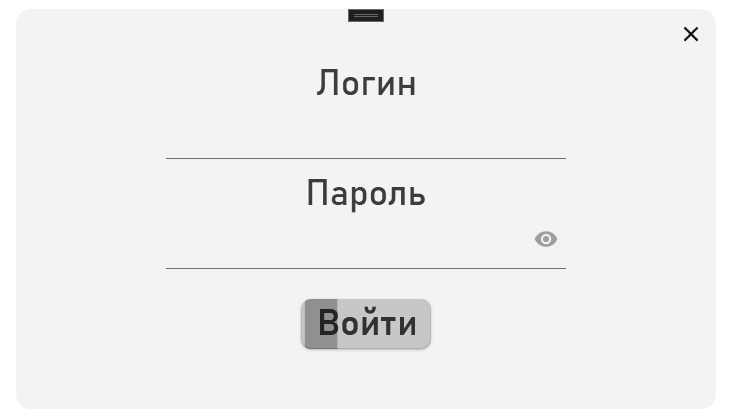


Рисунок 2.3 – Интерфейс окна «Авторизация»

Затем откроем App.xaml (C#). Этот XAML-файл определяет приложение WPF и все ресурсы приложения. Этот файл также используется для указания пользовательского интерфейса, в данном случае окна авторизации, которое автоматически отображается при запуске приложения.

Код XAML для окна авторизации выглядит следующим образом:

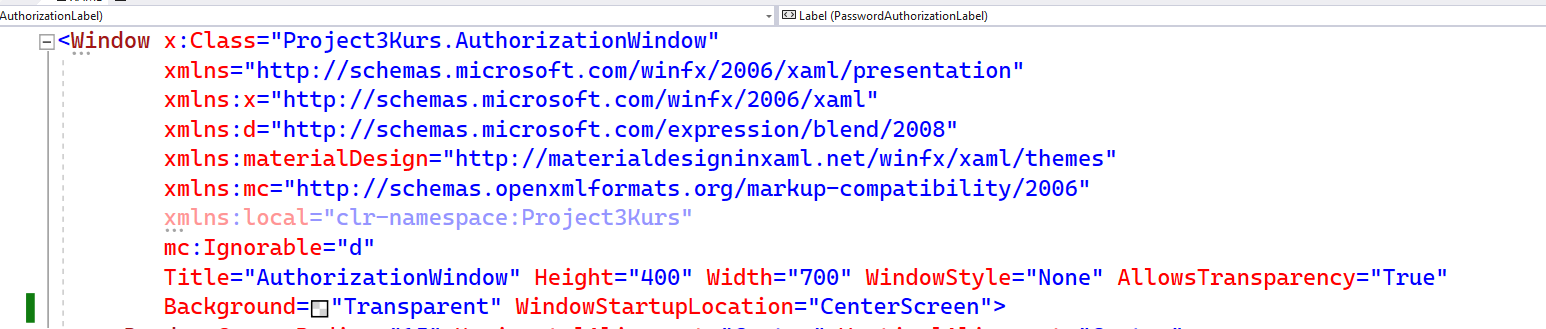


Рисунок 2.4 – XAML-код окна авторизации

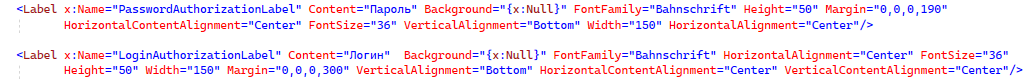


Рисунок 2.5 – XAML-код Label

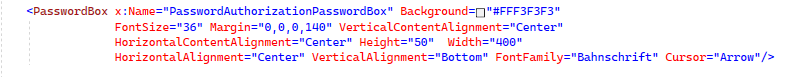


Рисунок 2.6 – XAML-код PasswordBox

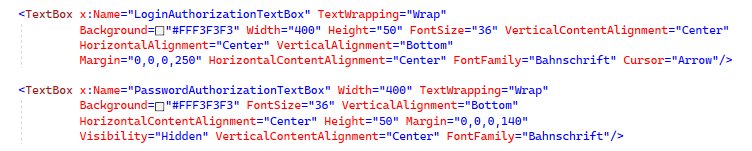


Рисунок 2.7 – XAML-код TextBox



Рисунок 2.8 – XAML-код Button

Следом перейдем к созданию кода на языке C# для окна авторизации. В обозревателе решений нажмем правой кнопкой мыши по проекту, выберем пункт «Добавить» - «Создать элемент».

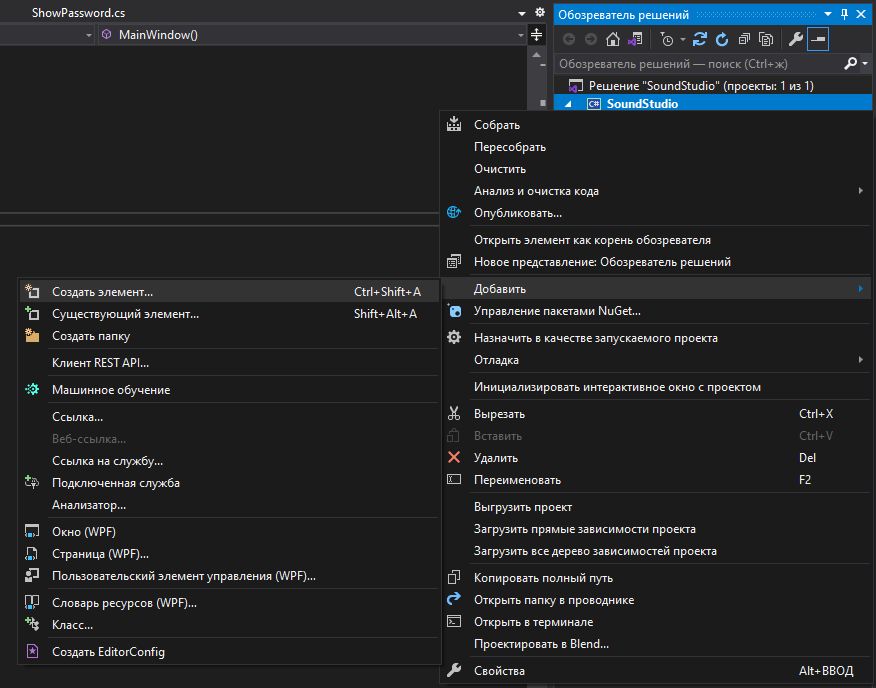


Рисунок 2.9 – Окно обозреватель решений

В открывшемся окне «Добавление нового элемента» выберем элемент «Класс Visual C#» и дадим ему название. Нажмем кнопку «Добавить» и перейдем к нему.

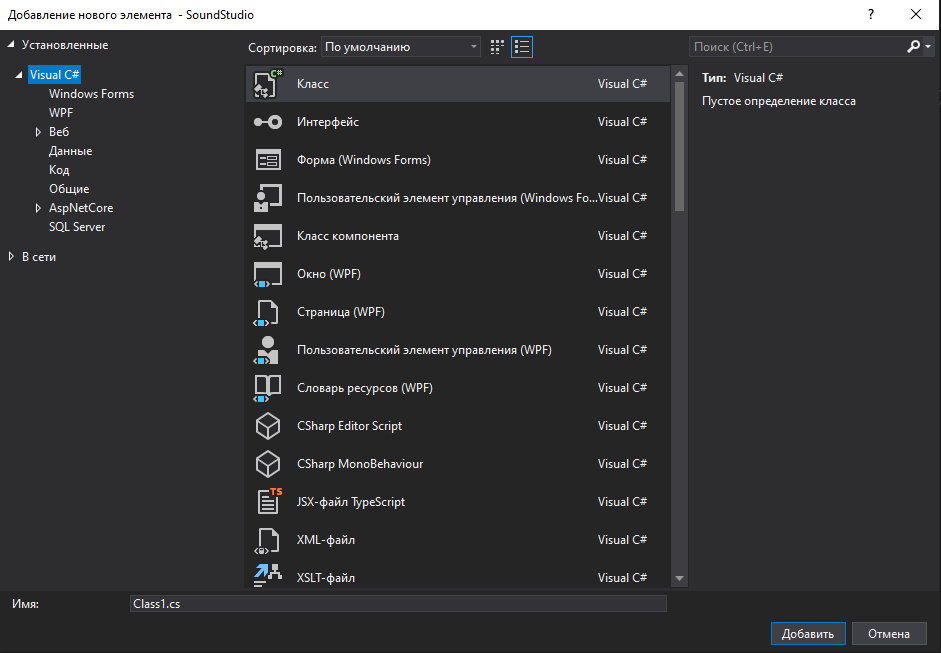


Рисунок 2.10 – Окно добавления нового класса

Создадим класс FirstConnectClass, который будет брать данные о пользователе из sql-базы данных, используя метод Select.

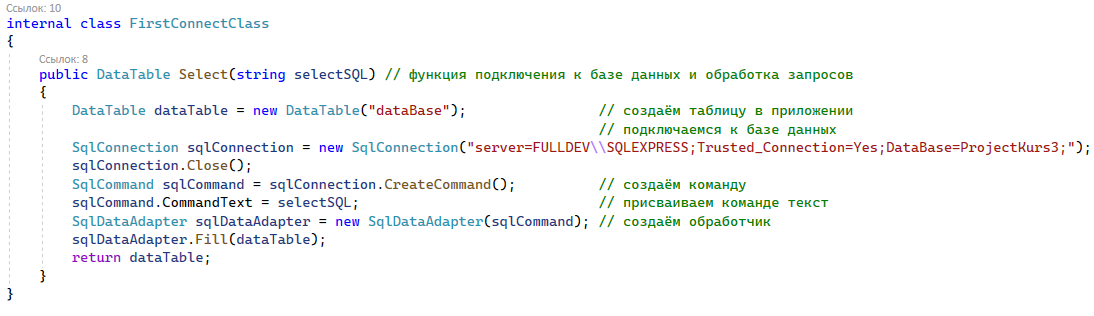


Рисунок 2.11 – Код класса FirstConnectClass

Затем сформируем класс CheckLogPassClass, отвечающий за вход в систему. В этом классе будет обращение к классу FirstConnectClass для получения информации из sql-базы данных о пользователе. Поставим условия входа – при удачном вводе логина и пароля, пользователь попадает на главное окно. При ошибке ввода пользователю сообщаеться об этом.



Рисунок 2.12 – Код класса CheckLogPassClass

Для показа введенного пароля на окне авторизации создадим класс ShowHidePassClass.

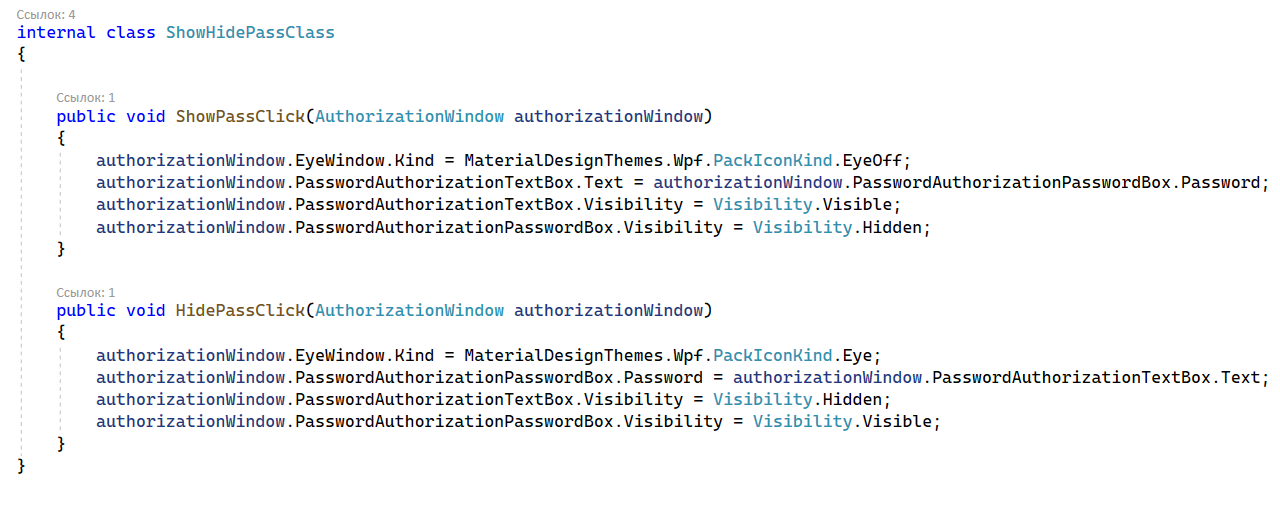


Рисунок 2.13 – Код класса ShowHidePassClass

Конечный код для окна авторизации, в котором будут вызываться все созданные классы, будет выглядеть следующим образом:

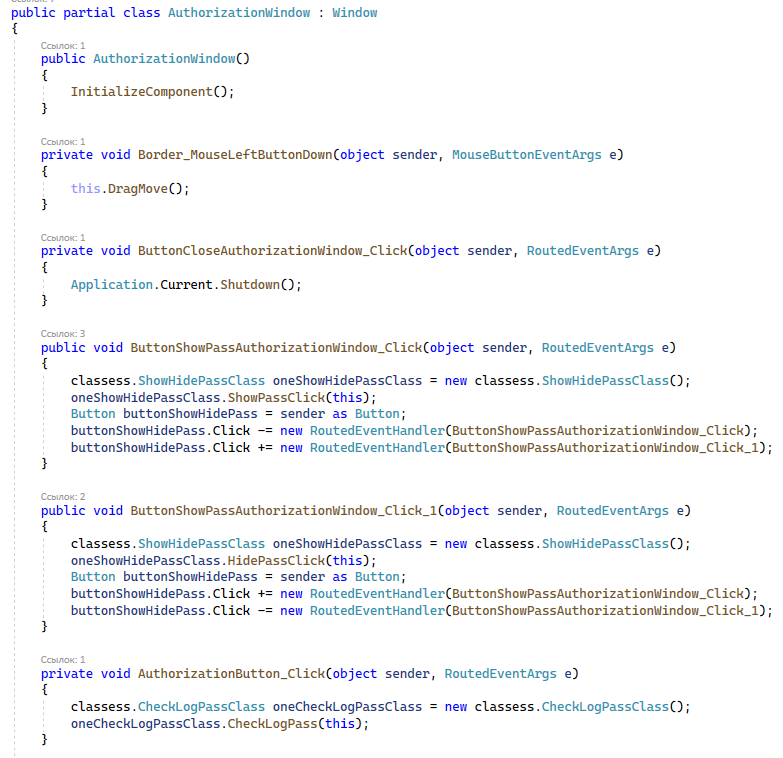


Рисунок 2.14 – Код окна авторизации

## Следом добавим главное окно проекта. Чтобы осуществить открытие нового окна в WPF приложении, для начала нужно создать окно. Для этого правой кнопкой мыши (ПКМ) нажимаем на название проекта в «Обозреватель решений» — пункт «Добавить» — «Окно».

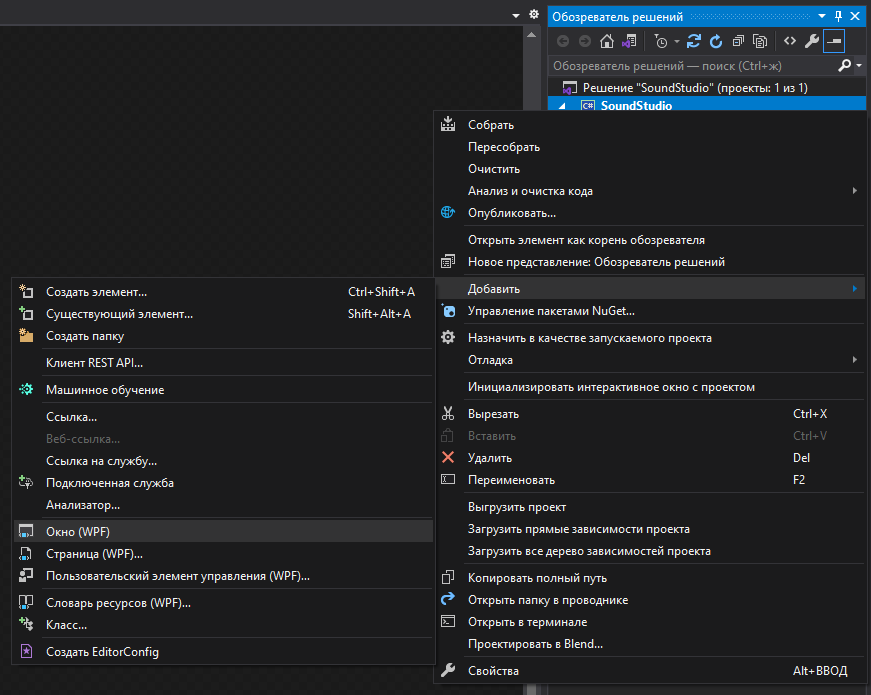


Рисунок 2.15 – Добавление главного окна проекта

После нужно указать название и нажать кнопку «Далее».

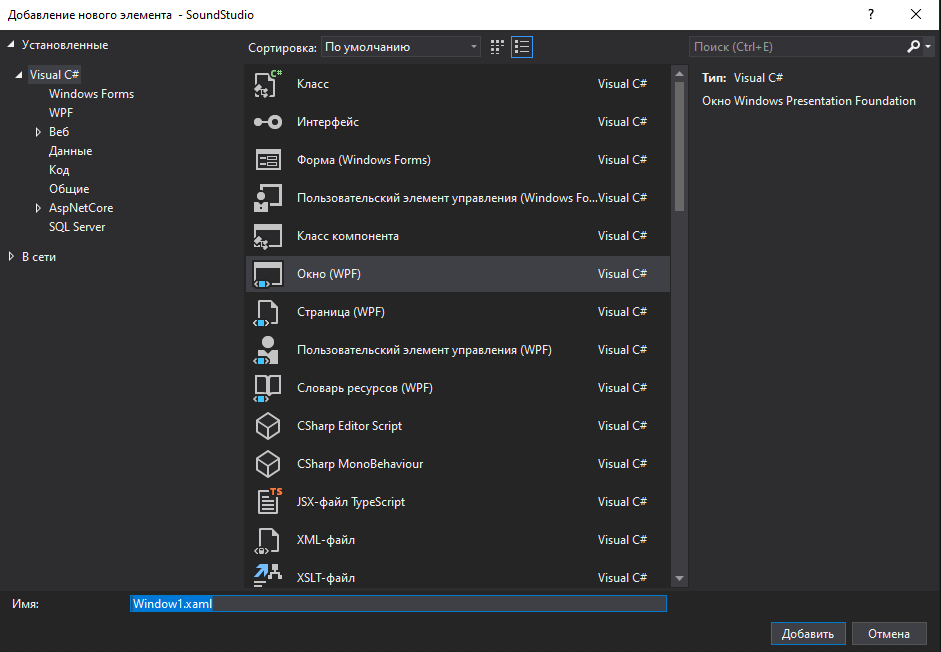


Рисунок 2.16 – Добавление имени главного окна проекта

Интерфейс главного окна и программный код разрабатываются аналогично описанному выше примеру.

## 2.2 Разработка руководства пользователя

Информационная система «Учет пациентов в поликлинике» представляет собой программный комплекс для ознакомления заказчиков и потенциальных клиентов с деятельностью и юридической информацией об организации ООО «3DentaL».

Данное руководство предназначено для пользователей приложения, установленного на устройствах, работающих под управлением операционной системы Windows.

Для установки приложения компьютер конечного пользователя должен удовлетворять требованиям.

Минимальная конфигурация сервера:

* + - тип процессора – Pentium IV и выше или совместимый с ним;
    - объем оперативного запоминающего устройства 4Гб и более;
    - жесткий диск 80 Гб;
    - модем, для выхода в Интернет;
    - монитор, клавиатура, мышь.

Требования, предъявляемые к конфигурации клиентских станций:

* + - процессор, с тактовой частотой не менее 1200 MHz,
    - 2 GB оперативной памяти;
    - монитор – SVGA;
    - клавиатура - 101/102 клавиши;
    - манипулятор типа «мышь».

Для реализации данной системы требуется установка следующего ПО: Microsoft Visual Studio 2021 и СУБД Microsoft SQL Server 2018 Express Edition.

Минимальная конфигурация:

* + - процессор x86 (Intel, AMD) с частотой как минимум 1,8 ГГц;
    - 2 ГБ ОЗУ; рекомендуется 4 ГБ ОЗУ;
    - доступное место на жестком диске объемом от 2 до 10 ГБ.

Для авторизации введите логин и пароль в открывшемся окне при входе в приложение, затем нажмите кнопку «ВОЙТИ». Для просмотра введенного вами пароля нажмите на изображение «Глаза» в поле ввода пароля.

Открытие главного окна обозначит успешный вход в приложение.

Диалоговое окно «Неправильные данные» обозначит ошибку ввода логина или пароля. Нажмите «Ок» и попробуйте ввести данные заново.

Для закрытия окна авторизации нажмите кнопку «Х» в правом верхнем углу экрана.

На главном окне находится основной функционал программы.

В верхнем левом углу расположена информация о пользователе.

В верхнем меню, расположенном на главном окне, представлены разделы:

* + - Расписание;
    - Пациенты;
    - Врачи;
    - Услуги.

Для возврата из главного окна на окно авторизации нажмите кнопку, расположенную в левом верхнем углу.

Для закрытия главного окна нажмите кнопку «X» в правом верхнем углу экрана.

# ЗАКЛЮЧЕНИЕ

В первой части курсовой работы была произведена разработка базы данных информационной системы «Учет пациентов в поликлинике». Была разработана физическая структура базы данных для информационной системы. Также, был рассмотрен процесс разработки запросов ИС «Учет пациентов в поликлинике». В дальнейшем создана база данных системы в программной среде MS SQL Server Management Studio 18.

Во второй части курсовой работы на основе рассмотренных процессов отдела «Учет пациентов в поликлинике» был разработан и подробно описан интерфейс информационной системы в программе MS Visual Studio 19, а также было разработано руководство пользователя.

Затем был разработан программный модуль автоматизированной информационной системы «Учет пациентов в поликлинике» с собственным интерфейсом и запросами при помощи программ Microsoft Visual Studio 2019 и MS SQL Server Management Studio 18.

Подробно описан и проиллюстрирован функционал и структура самого модуля, а также проведено тестирование готового программного продукта. В последнюю очередь была составлена сопутствующая документация.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 34.601 – 90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
2. ГОСТ 34.602 - 2020. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.
3. ГОСТ 19.201 - 78 ЕСПД. Техническое задание. Требования к содержанию и оформлению.
4. ГОСТ 19.202 - 78 ЕСПД. Спецификация. Требования к содержанию и оформлению.
5. ГОСТ Р ИСО/МЭК 12207. Процессы жизненного цикла программных средств.
6. Золотов С.Ю. Проектирование информационных систем: Учебно-методическое пособие. Методические рекомендации для выполнения курсового проекта, лабораторных работ и практических занятий по дисциплине «Проектирование информационных систем» - Томск: ТУСУР, 2013. - 34 с.
7. Шнайдер, Роберт Microsoft SQL Server 6.5. Проектирование высокопроизводительных баз данных; М.: Лори, 2010. - 361 c
8. Петкович, Душан Microsoft SQL Server 2012. Руководство для начинающих / Душан Петкович. - М.: БХВ-Петербург, 2012. - 460 c.
9. Тейлор, Аллен SQL для чайников / Аллен Тейлор. - М.: Вильямс, 2014. - 416 c.
10. Браст, Э.Дж. Разработка приложений на основе Microsoft SQL Server 2008 / Э.Дж. Браст. - М.: Русская Редакция, 2010. - 751 c.
11. Хетагуров, Я. А. Проектирование автоматизированных систем обработки информации и управления (АСОИУ). Учебник / Я.А. Хетагуров. - М.: Бином. Лаборатория знаний, 2015. - 240 c.
12. Кристофер, Д. Маннинг Введение в информационный поиск / Кристофер Д. Маннинг, ПрабхакарРагхаван ,ХайнрихШютце. - М.: Вильямс, 2014. - 528 c.
13. Бишоп Дж. С# в кратком изложении; Бином. Лаборатория знаний - М., 2015. - 234 c.
14. ГриффитсИэн Программирование на C# 5.0; Эксмо - М., 2014. - 580 c.
15. Гуриков С. Р. Введение в программирование на языке Visual C#; ИЛ - Москва, 2013. - 448 c.
16. Шилдт Герберт C# 4.0. Полное руководство; Вильямс - М., 2015. - 291 c.
17. Эндрю Троелсен Язык программирования C# 5.0 и платформа .NET 4.5; Диалектика / Вильямс - М., 2015. - 126 c.

# ПРИЛОЖЕНИЕ А

# ЛИСТИНГ ПРОГРАММЫ

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;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class AuthorizationWindow : Window

{

public AuthorizationWindow()

{

InitializeComponent();

}

private void Border\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

this.DragMove();

}

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

{

Application.Current.Shutdown();

}

public void ButtonShowPassAuthorizationWindow\_Click(object sender, RoutedEventArgs e)

{

classess.ShowHidePassClass oneShowHidePassClass = new classess.ShowHidePassClass();

oneShowHidePassClass.ShowPassClick(this);

Button buttonShowHidePass = sender as Button;

buttonShowHidePass.Click -= new RoutedEventHandler(ButtonShowPassAuthorizationWindow\_Click);

buttonShowHidePass.Click += new RoutedEventHandler(ButtonShowPassAuthorizationWindow\_Click\_1);

}

public void ButtonShowPassAuthorizationWindow\_Click\_1(object sender, RoutedEventArgs e)

{

classess.ShowHidePassClass oneShowHidePassClass = new classess.ShowHidePassClass();

oneShowHidePassClass.HidePassClick(this);

Button buttonShowHidePass = sender as Button;

buttonShowHidePass.Click += new RoutedEventHandler(ButtonShowPassAuthorizationWindow\_Click);

buttonShowHidePass.Click -= new RoutedEventHandler(ButtonShowPassAuthorizationWindow\_Click\_1);

}

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

{

classess.CheckLogPassClass oneCheckLogPassClass = new classess.CheckLogPassClass();

oneCheckLogPassClass.CheckLogPass(this);

}

}

}

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.Shapes;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

namespace Project3Kurs

{

/// <summary>

/// Логика взаимодействия для PrimaryWindow.xaml

/// </summary>

public partial class PrimaryWindow : Window

{

public PrimaryWindow()

{

InitializeComponent();

}

private void Border\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)

{

this.DragMove();

}

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

{

Application.Current.Shutdown();

}

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

{

AuthorizationWindow oneAuthorizationWindow = new AuthorizationWindow();

oneAuthorizationWindow.Show();

this.Close();

}

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

{

classess.CompetitionTablePatients oneCompletionTablePatient = new classess.CompetitionTablePatients();

oneCompletionTablePatient.MethodCompetitionTablePatient(this);

}

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

{

classess.CompetitionTableDoctors oneCompetitionTableDoctors = new classess.CompetitionTableDoctors();

oneCompetitionTableDoctors.MethodCompetitionTableDoctors(this);

}

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

{

classess.CompetitionTableServices oneCompetitionTableServices = new classess.CompetitionTableServices();

oneCompetitionTableServices.MethodCompetitionTableServices(this);

}

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

{

classess.CompetitionTableShedule oneCompetitionTableShedule = new classess.CompetitionTableShedule();

oneCompetitionTableShedule.MethodCompetitionTableShedule(this);

}

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

{

dynamic itemSelectList = ListPrimaryShedule.SelectedItem;

if (itemSelectList != null)

{

classess.DeleteTableSchedule oneDeleteTableSchedule = new classess.DeleteTableSchedule();

oneDeleteTableSchedule.MethodDeleteTableSchedule(this);

}

}

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

{

}

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

{

classess.AddTableSchedule oneAddTableSchedule = new classess.AddTableSchedule();

oneAddTableSchedule.MethodAddTableSchedule(this);

}

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

{

classess.AddTablePatients oneAddTablePatients = new classess.AddTablePatients();

oneAddTablePatients.MethodAddTablePatients(this);

}

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

{

dynamic itemSelectList = ListPrimaryPatients.SelectedItem;

if (itemSelectList != null)

{

classess.DeleteTablePatients oneDeleteTablePatients = new classess.DeleteTablePatients();

oneDeleteTablePatients.MethodDeleteTablePatients(this);

}

}

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

{

}

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

{

classess.AddTableDoctors oneAddTableDoctors = new classess.AddTableDoctors();

oneAddTableDoctors.MethodAddTableDoctors(this);

}

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

{

dynamic itemSelectList = ListPrimaryDoctors.SelectedItem;

if (itemSelectList != null)

{

classess.DeleteTableDoctors oneDeleteTableDoctors = new classess.DeleteTableDoctors();

oneDeleteTableDoctors.MethodDeleteTableDoctors(this);

}

}

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

{

classess.AddTableServices oneAddTableServices = new classess.AddTableServices();

oneAddTableServices.MethodAddTableServices(this);

}

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

{

dynamic itemSelectList = ListPrimaryServices.SelectedItem;

if (itemSelectList != null)

{

classess.DeleteTableServices oneDeleteTableServices = new classess.DeleteTableServices();

oneDeleteTableServices.MethodDeleteTableServices(this);

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class AddTableDoctors

{

public void MethodAddTableDoctors(PrimaryWindow onePrimaryWindow)

{

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open();

SqlCommand sqlCommand = sqlConnection.CreateCommand();

sqlCommand.CommandText = "Insert into doctors (DoctorFIO,DoctorPosition,DoctorCabinet)values(@DoctorFIO,@DoctorPosition,@DoctorCabinet)";

sqlCommand.Parameters.AddWithValue("@DoctorFIO", onePrimaryWindow.DoctorFIODoctors.Text);

sqlCommand.Parameters.AddWithValue("@DoctorPosition", onePrimaryWindow.DoctorPositionDoctors.Text);

sqlCommand.Parameters.AddWithValue("@DoctorCabinet", onePrimaryWindow.DoctorCabinetDoctors.Text);

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand);

sqlDataAdapter.Fill(dataTable);

classess.CompetitionTableDoctors oneCompetitionTableDoctors = new classess.CompetitionTableDoctors();

oneCompetitionTableDoctors.MethodCompetitionTableDoctors(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class AddTablePatients

{

public void MethodAddTablePatients(PrimaryWindow onePrimaryWindow)

{

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open();

SqlCommand sqlCommand = sqlConnection.CreateCommand();

sqlCommand.CommandText = "Insert into patients (PatientFIO,PatientPol,PatientDateOfBirth,PatientAge,PatientTelephoneNumber)values(@PatientFIO,@PatientPol,@PatientDateOfBirth,@PatientAge,@PatientTelephoneNumber)";

sqlCommand.Parameters.AddWithValue("@PatientFIO", onePrimaryWindow.PatientFIOPatients.Text);

sqlCommand.Parameters.AddWithValue("@PatientPol", onePrimaryWindow.PatientPolPatients.Text);

sqlCommand.Parameters.AddWithValue("@PatientDateOfBirth", onePrimaryWindow.PatientDateOfBirthPatients.Text);

sqlCommand.Parameters.AddWithValue("@PatientAge", onePrimaryWindow.PatientAgePatients.Text);

sqlCommand.Parameters.AddWithValue("@PatientTelephoneNumber", onePrimaryWindow.PatientTelephoneNumberPatients.Text);

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand);

sqlDataAdapter.Fill(dataTable);

classess.CompetitionTablePatients oneCompletionTablePatient = new classess.CompetitionTablePatients();

oneCompletionTablePatient.MethodCompetitionTablePatient(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Windows;

namespace Project3Kurs.classess

{

internal class AddTableSchedule

{

public void MethodAddTableSchedule(PrimaryWindow onePrimaryWindow)

{

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open();

SqlCommand sqlCommand = sqlConnection.CreateCommand();

sqlCommand.CommandText = "Insert into schedule (DateVisit,DoctorCabinet,PatientFio,DoctorFio,DoctorPosition,Service)values(@DateVisit,@DoctorCabinet,@PatientFio,@DoctorFio,@DoctorPosition,@Service)";

sqlCommand.Parameters.AddWithValue("@DateVisit", onePrimaryWindow.DateVisitShedule.Text);

sqlCommand.Parameters.AddWithValue("@DoctorCabinet", onePrimaryWindow.DoctorCabinetSchedule.Text);

sqlCommand.Parameters.AddWithValue("@PatientFio", onePrimaryWindow.PatientFioSchedule.Text);

sqlCommand.Parameters.AddWithValue("@DoctorFio", onePrimaryWindow.DoctorFioSchedule.Text);

sqlCommand.Parameters.AddWithValue("@DoctorPosition", onePrimaryWindow.DoctorPositionSchedule.Text);

sqlCommand.Parameters.AddWithValue("@Service", onePrimaryWindow.ServiceSchedule.Text);

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand);

sqlDataAdapter.Fill(dataTable);

classess.CompetitionTableShedule oneCompetitionTableShedule = new classess.CompetitionTableShedule();

oneCompetitionTableShedule.MethodCompetitionTableShedule(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class AddTableServices

{

public void MethodAddTableServices(PrimaryWindow onePrimaryWindow)

{

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open();

SqlCommand sqlCommand = sqlConnection.CreateCommand();

sqlCommand.CommandText = "Insert into services (JobTitle,ServiceTitle)values(@JobTitle,@ServiceTitle)";

sqlCommand.Parameters.AddWithValue("@JobTitle", onePrimaryWindow.JobTitleServices.Text);

sqlCommand.Parameters.AddWithValue("@ServiceTitle", onePrimaryWindow.ServiceTitleServices.Text);

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand);

sqlDataAdapter.Fill(dataTable);

classess.CompetitionTableServices oneCompetitionTableServices = new classess.CompetitionTableServices();

oneCompetitionTableServices.MethodCompetitionTableServices(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Windows;

namespace Project3Kurs.classess

{

internal class CheckLogPassClass

{

public void CheckLogPass(AuthorizationWindow authorizationWindow)

{

if (authorizationWindow.PasswordAuthorizationTextBox.Text.Length > 0)

{

authorizationWindow.PasswordAuthorizationPasswordBox.Password = authorizationWindow.PasswordAuthorizationTextBox.Text;

}

FirstConnectClass firstConnect = new classess.FirstConnectClass();

DataTable dt\_StroyCompany = firstConnect.Select("SELECT \* FROM [dbo].[users] WHERE [login] = '" + authorizationWindow.LoginAuthorizationTextBox.Text + "' AND [password] = '" + authorizationWindow.PasswordAuthorizationPasswordBox.Password + "'");

if (authorizationWindow.LoginAuthorizationTextBox.Text.Length <= 0)

{

MessageBox.Show("Вы не ввели логин!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

}

else if (authorizationWindow.PasswordAuthorizationPasswordBox.Password.Length <= 0)

{

MessageBox.Show("Вы не ввели пароль!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

authorizationWindow.LoginAuthorizationTextBox.Clear();

}

else if (dt\_StroyCompany.Rows.Count > 0)

{

PrimaryWindow onePrimaryWindow = new PrimaryWindow();

onePrimaryWindow.Owner = authorizationWindow;

onePrimaryWindow.LabelPrimaryNameAcc.Content = dt\_StroyCompany.Rows[0][3].ToString();

onePrimaryWindow.Show();

authorizationWindow.Hide();

classess.CompetitionTableShedule oneCompetitionTableShedule = new classess.CompetitionTableShedule();

oneCompetitionTableShedule.MethodCompetitionTableShedule(onePrimaryWindow);

}

else

{

MessageBox.Show("Неверные данные!", "Ошибка", MessageBoxButton.OK, MessageBoxImage.Error);

authorizationWindow.PasswordAuthorizationTextBox.Clear();

authorizationWindow.LoginAuthorizationTextBox.Clear();

authorizationWindow.PasswordAuthorizationPasswordBox.Clear();

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

namespace Project3Kurs.classess

{

class CompetitionTableDoctors

{

internal class Rents

{

public string id\_Doctor { get; set; }

public string DoctorFIO { get; set; }

public string DoctorPosition { get; set; }

public string DoctorCabinet { get; set; }

}

public void MethodCompetitionTableDoctors(PrimaryWindow onePrimaryWindow)

{

onePrimaryWindow.ListPrimaryPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeletePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFIOPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientAgePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientDateOfBirthPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientPolPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientTelephoneNumberPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryServices.Visibility = Visibility.Hidden;

onePrimaryWindow.JobTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DateVisitShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryDoctors.Items.Clear();

onePrimaryWindow.DoctorFIODoctors.Text = null;

onePrimaryWindow.DoctorPositionDoctors.Text = null;

onePrimaryWindow.DoctorCabinetDoctors.Text = null;

onePrimaryWindow.ListPrimaryDoctors.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorFIODoctors.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorPositionDoctors.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorCabinetDoctors.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonAddDoctors.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonDeleteDoctors.Visibility = Visibility.Visible;

classess.FirstConnectClass firstConnect = new classess.FirstConnectClass();

DataTable dt\_StroyCompany = firstConnect.Select("SELECT \* FROM [dbo].[doctors]"); // данные из БД

for (int i = 0; i < dt\_StroyCompany.Rows.Count; i++) // перебираем данные

{

Rents Lol = new Rents() // создаём экземпляр класса

{

id\_Doctor = dt\_StroyCompany.Rows[i][0].ToString(),

DoctorFIO = dt\_StroyCompany.Rows[i][1].ToString(), // указываем изображение из таблицы

DoctorPosition = dt\_StroyCompany.Rows[i][2].ToString(), // указываем название товара

DoctorCabinet = dt\_StroyCompany.Rows[i][3].ToString(), // указываем описание

};

onePrimaryWindow.ListPrimaryDoctors.Items.Add(Lol); // выводим строку в список

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

namespace Project3Kurs.classess

{

internal class CompetitionTablePatients

{

internal class Rents

{

public string id\_Patient { get; set; }

public string PatientFIO { get; set; }

public string PatientPol { get; set; }

public string PatientDateOfBirth { get; set; }

public string PatientAge { get; set; }

public string PatientTelephoneNumber { get; set; }

}

public void MethodCompetitionTablePatient(PrimaryWindow onePrimaryWindow)

{

onePrimaryWindow.ButtonAddSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DateVisitShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFIODoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryServices.Visibility = Visibility.Hidden;

onePrimaryWindow.JobTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryPatients.Items.Clear();

onePrimaryWindow.PatientFIOPatients.Text = null;

onePrimaryWindow.PatientAgePatients.Text = null;

onePrimaryWindow.PatientDateOfBirthPatients.Text = null;

onePrimaryWindow.PatientPolPatients.Text = null;

onePrimaryWindow.PatientTelephoneNumberPatients.Text = null;

onePrimaryWindow.ListPrimaryPatients.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonAddPatients.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonDeletePatients.Visibility = Visibility.Visible;

onePrimaryWindow.PatientFIOPatients.Visibility = Visibility.Visible;

onePrimaryWindow.PatientAgePatients.Visibility = Visibility.Visible;

onePrimaryWindow.PatientDateOfBirthPatients.Visibility = Visibility.Visible;

onePrimaryWindow.PatientPolPatients.Visibility = Visibility.Visible;

onePrimaryWindow.PatientTelephoneNumberPatients.Visibility = Visibility.Visible;

classess.FirstConnectClass firstConnect = new classess.FirstConnectClass();

DataTable dt\_StroyCompany = firstConnect.Select("SELECT \* FROM [dbo].[patients]"); // данные из БД

for (int i = 0; i < dt\_StroyCompany.Rows.Count; i++) // перебираем данные

{

Rents Lol = new Rents() // создаём экземпляр класса

{

id\_Patient = dt\_StroyCompany.Rows[i][0].ToString(),

PatientFIO = dt\_StroyCompany.Rows[i][1].ToString(), // указываем изображение из таблицы

PatientPol = dt\_StroyCompany.Rows[i][2].ToString(), // указываем название товара

PatientDateOfBirth = dt\_StroyCompany.Rows[i][3].ToString(), // указываем описание

PatientAge = dt\_StroyCompany.Rows[i][4].ToString(), // указываем наличие

PatientTelephoneNumber = dt\_StroyCompany.Rows[i][5].ToString() // указываем наличие

};

Lol.PatientDateOfBirth = Lol.PatientDateOfBirth.TrimEnd(':', '0');

onePrimaryWindow.ListPrimaryPatients.Items.Add(Lol); // выводим строку в список

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

namespace Project3Kurs.classess

{

internal class CompetitionTableServices

{

internal class Rents

{

public string id\_Services { get; set; }

public string JobTitle { get; set; }

public string ServicesTitle { get; set; }

}

public void MethodCompetitionTableServices(PrimaryWindow onePrimaryWindow)

{

onePrimaryWindow.ListPrimaryDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFIODoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DateVisitShedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFioSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceSchedule.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeletePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFIOPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientAgePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientDateOfBirthPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientPolPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientTelephoneNumberPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryServices.Items.Clear();

onePrimaryWindow.JobTitleServices.Text = null;

onePrimaryWindow.ServiceTitleServices.Text = null;

onePrimaryWindow.ListPrimaryServices.Visibility = Visibility.Visible;

onePrimaryWindow.JobTitleServices.Visibility = Visibility.Visible;

onePrimaryWindow.ServiceTitleServices.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonAddServices.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonDeleteServices.Visibility = Visibility.Visible;

classess.FirstConnectClass firstConnect = new classess.FirstConnectClass();

DataTable dt\_StroyCompany = firstConnect.Select("SELECT \* FROM [dbo].[services]"); // данные из БД

for (int i = 0; i < dt\_StroyCompany.Rows.Count; i++) // перебираем данные

{

Rents Lol = new Rents() // создаём экземпляр класса

{

id\_Services = dt\_StroyCompany.Rows[i][0].ToString(),

JobTitle = dt\_StroyCompany.Rows[i][1].ToString(), // указываем изображение из таблицы

ServicesTitle = dt\_StroyCompany.Rows[i][2].ToString(), // указываем название товара

};

onePrimaryWindow.ListPrimaryServices.Items.Add(Lol); // выводим строку в список

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

namespace Project3Kurs.classess

{

internal class CompetitionTableShedule

{

internal class Rents

{

public string id\_Visit { get; set; }

public string DateVisit { get; set; }

public string DoctorCabinet { get; set; }

public string PatientFio { get; set; }

public string DoctorFio { get; set; }

public string DoctorPosition { get; set; }

public string Services { get; set; }

}

public void MethodCompetitionTableShedule(PrimaryWindow onePrimaryWindow)

{

onePrimaryWindow.ListPrimaryDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorFIODoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorPositionDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.DoctorCabinetDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteDoctors.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryServices.Visibility = Visibility.Hidden;

onePrimaryWindow.JobTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ServiceTitleServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeleteServices.Visibility = Visibility.Hidden;

onePrimaryWindow.ListPrimaryPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonAddPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.ButtonDeletePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientFIOPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientAgePatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientDateOfBirthPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientPolPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.PatientTelephoneNumberPatients.Visibility = Visibility.Hidden;

onePrimaryWindow.DateVisitShedule.Text = null;

onePrimaryWindow.ListPrimaryShedule.Items.Clear();

onePrimaryWindow.DoctorCabinetSchedule.Items.Clear();

onePrimaryWindow.PatientFioSchedule.Items.Clear();

onePrimaryWindow.DoctorFioSchedule.Items.Clear();

onePrimaryWindow.DoctorPositionSchedule.Items.Clear();

onePrimaryWindow.ServiceSchedule.Items.Clear();

onePrimaryWindow.DateVisitShedule.Visibility = Visibility.Visible;

onePrimaryWindow.ListPrimaryShedule.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorCabinetSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.PatientFioSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorFioSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.DoctorPositionSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.ServiceSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonAddSchedule.Visibility = Visibility.Visible;

onePrimaryWindow.ButtonDeleteSchedule.Visibility = Visibility.Visible;

classess.FirstConnectClass firstConnect = new classess.FirstConnectClass();

DataTable dt\_schedule = firstConnect.Select("SELECT \* FROM [dbo].[schedule]"); // данные из БД

for (int i = 0; i < dt\_schedule.Rows.Count; i++) // перебираем данные

{

Rents Lol = new Rents() // создаём экземпляр класса

{

id\_Visit = dt\_schedule.Rows[i][0].ToString(),

DateVisit = dt\_schedule.Rows[i][1].ToString(), // указываем изображение из таблицы

DoctorCabinet = dt\_schedule.Rows[i][2].ToString(), // указываем название товара

PatientFio = dt\_schedule.Rows[i][3].ToString(), // указываем описание

DoctorFio = dt\_schedule.Rows[i][4].ToString(), // указываем наличие

DoctorPosition = dt\_schedule.Rows[i][5].ToString(),

Services = dt\_schedule.Rows[i][5].ToString()// указываем наличие

};

Lol.DateVisit = Lol.DateVisit.TrimEnd(':', '0');

onePrimaryWindow.ListPrimaryShedule.Items.Add(Lol); // выводим строку в список

}

DataTable dt\_doctors = firstConnect.Select("SELECT \* FROM [dbo].[doctors]"); // данные из БД

for (int i = 0; i < dt\_doctors.Rows.Count; i++) // перебираем данные

{

onePrimaryWindow.DoctorFioSchedule.Items.Add(dt\_doctors.Rows[i][1].ToString());

onePrimaryWindow.DoctorPositionSchedule.Items.Add(dt\_doctors.Rows[i][2].ToString());

onePrimaryWindow.DoctorCabinetSchedule.Items.Add(dt\_doctors.Rows[i][3].ToString());

}

DataTable dt\_patients = firstConnect.Select("SELECT \* FROM [dbo].[patients]"); // данные из БД

for (int i = 0; i < dt\_patients.Rows.Count; i++) // перебираем данные

{

onePrimaryWindow.PatientFioSchedule.Items.Add(dt\_patients.Rows[i][1].ToString());

}

DataTable dt\_services = firstConnect.Select("SELECT \* FROM [dbo].[services]"); // данные из БД

for (int i = 0; i < dt\_services.Rows.Count; i++) // перебираем данные

{

onePrimaryWindow.ServiceSchedule.Items.Add(dt\_services.Rows[i][2].ToString());

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class DeleteTableDoctors

{

public void MethodDeleteTableDoctors(PrimaryWindow onePrimaryWindow)

{

dynamic itemSelectList = onePrimaryWindow.ListPrimaryDoctors.SelectedItem;

var id = itemSelectList.id\_Doctor;

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = "DELETE FROM doctors WHERE [id\_Doctor] = @id";

sqlCommand.Parameters.AddWithValue("@id", id); // присваиваем команде текст

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); ;

onePrimaryWindow.ListPrimaryDoctors.Items.Clear();

classess.CompetitionTableDoctors oneCompetitionTableDoctors = new classess.CompetitionTableDoctors();

oneCompetitionTableDoctors.MethodCompetitionTableDoctors(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

using System.Data;

namespace Project3Kurs.classess

{

internal class DeleteTablePatients

{

public void MethodDeleteTablePatients(PrimaryWindow onePrimaryWindow)

{

dynamic itemSelectList = onePrimaryWindow.ListPrimaryPatients.SelectedItem;

var id = itemSelectList.id\_Patient;

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = "DELETE FROM patients WHERE [id\_Patient] = @id";

sqlCommand.Parameters.AddWithValue("@id", id); // присваиваем команде текст

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable);

onePrimaryWindow.ListPrimaryPatients.Items.Clear();

classess.CompetitionTablePatients oneCompletionTablePatient = new classess.CompetitionTablePatients();

oneCompletionTablePatient.MethodCompetitionTablePatient(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Data.OleDb;

using System.Windows;

using System.Data;

namespace Project3Kurs.classess

{

class DeleteTableSchedule

{

public void MethodDeleteTableSchedule(PrimaryWindow onePrimaryWindow)

{

dynamic itemSelectList = onePrimaryWindow.ListPrimaryShedule.SelectedItem;

var id = itemSelectList.id\_Visit;

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = "DELETE FROM schedule WHERE [id\_Visit] = @id";

sqlCommand.Parameters.AddWithValue("@id", id); // присваиваем команде текст

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable);;

onePrimaryWindow.ListPrimaryShedule.Items.Clear();

classess.CompetitionTableShedule oneCompetitionTableShedule = new classess.CompetitionTableShedule();

oneCompetitionTableShedule.MethodCompetitionTableShedule(onePrimaryWindow);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class DeleteTableServices

{

public void MethodDeleteTableServices(PrimaryWindow onePrimaryWindow)

{

dynamic itemSelectList = onePrimaryWindow.ListPrimaryServices.SelectedItem;

var id = itemSelectList.id\_Services;

DataTable dataTable = new DataTable("dataBase");

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Open(); // открываем базу данных

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = "DELETE FROM services WHERE [id\_Service] = @id";

sqlCommand.Parameters.AddWithValue("@id", id); // присваиваем команде текст

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable); ;

onePrimaryWindow.ListPrimaryServices.Items.Clear();

classess.CompetitionTableServices oneCompetitionTableServices = new classess.CompetitionTableServices();

oneCompetitionTableServices.MethodCompetitionTableServices(onePrimaryWindow);

}

}

}

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;

using System.Data;

using System.Data.SqlClient;

namespace Project3Kurs.classess

{

internal class FirstConnectClass

{

public DataTable Select(string selectSQL) // функция подключения к базе данных и обработка запросов

{

DataTable dataTable = new DataTable("dataBase"); // создаём таблицу в приложении

// подключаемся к базе данных

SqlConnection sqlConnection = new SqlConnection("server=FULLDEV\\SQLEXPRESS;Trusted\_Connection=Yes;DataBase=ProjectKurs3;");

sqlConnection.Close();

SqlCommand sqlCommand = sqlConnection.CreateCommand(); // создаём команду

sqlCommand.CommandText = selectSQL; // присваиваем команде текст

SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sqlCommand); // создаём обработчик

sqlDataAdapter.Fill(dataTable);

return dataTable;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

namespace Project3Kurs.classess

{

internal class ShowHidePassClass

{

public void ShowPassClick(AuthorizationWindow authorizationWindow)

{

authorizationWindow.EyeWindow.Kind = MaterialDesignThemes.Wpf.PackIconKind.EyeOff;

authorizationWindow.PasswordAuthorizationTextBox.Text = authorizationWindow.PasswordAuthorizationPasswordBox.Password;

authorizationWindow.PasswordAuthorizationTextBox.Visibility = Visibility.Visible;

authorizationWindow.PasswordAuthorizationPasswordBox.Visibility = Visibility.Hidden;

}

public void HidePassClick(AuthorizationWindow authorizationWindow)

{

authorizationWindow.EyeWindow.Kind = MaterialDesignThemes.Wpf.PackIconKind.Eye;

authorizationWindow.PasswordAuthorizationPasswordBox.Password = authorizationWindow.PasswordAuthorizationTextBox.Text;

authorizationWindow.PasswordAuthorizationTextBox.Visibility = Visibility.Hidden;

authorizationWindow.PasswordAuthorizationPasswordBox.Visibility = Visibility.Visible;

}

}

}