ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

(ФГБОУ ВО «ВГТУ»)

Строительно-политехнический колледж

*(наименование структурного подразделения)*

**Отчет**

о прохождении учебной практики профессионального модуля

ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

Название практики: Учебная практика Разработка модулей программного обеспечения для компьютерных систем

Обучающегося Клочковой Анастасии Александровны

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(*Ф.И.О.)*

3 курса ИСП-201о группы

Сроки прохождения практики:

с «13» мая 2023 г. по «30» июня 2023г.

в объеме 36 час.

Место прохождения практики ПАО Сбербанк

*(наименование организации)*

Итоговая оценка \_\_\_\_\_\_\_\_\_\_\_ Дата \_\_\_\_28.04.2023\_\_\_\_\_\_\_\_

Руководитель по практической подготовке

от организации \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(подпись, дата) (И.О.Фамилия, должность)*

Руководитель по практической

подготовке от ВГТУ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_М.А. Балабаева\_\_\_

*(подпись, дата) (И.О.Фамилия, должность)*

Содержание

[Введение 4](#_Toc134178722)

[1.1. Анализ предметной области 6](#_Toc134178723)

[1.2. Разработка технического задания 7](#_Toc134178724)

[1.3. Анализ требований к программному продукту 9](#_Toc134178725)

[1.4. Определение основных технических решений 10](#_Toc134178726)

[1.5. Проектирование программного продукта 11](#_Toc134178727)

[1.6. Разработка программного продукта 13](#_Toc134178728)

[1.7. Тестирование программного продукта 15](#_Toc134178729)

[Заключение 27](#_Toc134178730)

[Список использованных источников 28](#_Toc134178731)

[Приложение А 29](#_Toc134178732)

# Введение

Одним из наиболее перспективных способов эффективного управления образовательным процессом является внедрение информационно-коммуникативных технологий в процесс менеджмента на базе международных стандартов.

Развитие сферы информационных технологий дало толчок к усовершенствованию всех сфер жизнедеятельности общества.

Достаточно актуальным является вопрос автоматизации учебного процесса, в частности, автоматизации процессов управления в высших учебных заведениях (вузах).

Цель автоматизации управления – получить удобные средства мониторинга, анализа и отображения основных процессов приема студентов, административного управления, подготовки и выпуска специалистов университета (института, академии).

Следует отметить, что одной из основных и трудоёмких в работе вуза является деятельность деканата. Работникам деканатов приходится выполнять огромный объем рутинной работы по учету контингента студентов, обеспечению учебного процесса, предоставлению информации в различные подразделения вуза. При этом всю информацию необходимо представлять в различных форматах. При этом необходимо осуществлять прогноз успеваемости студентов с целью эффективного контроля за процессом обучения и своевременного реагирования на возможные негативные результаты (низкие показатели абсолютной и качественной успеваемости).

Таким образом, необходимость внедрения информационной системы, автоматизирующей основные функции образовательного процесса, является актуальной. Прежде чем внедрять автоматизированную информационную систему (АИС) в деканате, как и в любом другом подразделении вуза, необходимо определить основные требования к её работе. Выявленные проблемы предметной области помогут определить направление автоматизации данной сферы деятельности, а разработка модели предметной области обеспечит успешную разработку и внедрение базы данных, как основной части АИС.

Необходимо выполнить задачи, которые исходят из цели работы:

1. Разработать техническое задание в соответствии со своей предметной областью и выполнить анализ функциональных и эксплуатационных требований.
2. Определить основные технические решения.
3. Спроектировать программный продукт, построив необходимые для этого диаграммы.
4. Разработать программный продукт.
5. Произвести отладку программы.
6. Провести оценку качества в соответствии со стандартами и техническим заданием.

Приложение, которое мы разработали, позволяет взаимодействовать с базой данных, а также предоставляет пользователю понятный интерфейс.

* 1. Анализ предметной области

Для успешного и эффективного взаимодействия студентов и преподавателей в ВУЗе необходимо правильно составленное расписание.

Одним из структурных подразделений ВУЗа является деканат.

Деканат – подразделение ВУЗа, осуществляющее контроль и организацию деятельности факультета.

В отделе по составлению расписания для студентов и учителей в распоряжении имеются: сведения о сформированных группах студентов, данные о штате преподавателей, информация об аудиториях и об изучаемых дисциплинах.

Основные задачи, которые необходимо решать в процессе управленческой деятельности деканата:

– разработать концептуальную и физическую модели базы данных хранения информации о студентах;

– обосновать выбор методов и моделей обработки и анализа данных по результатам обучения студентов;

– провести анализ и обосновать выбор существующих программных инструментариев для разработки систем хранения информации и осуществления эффективного документооборота;

– реализовать в виде конечного программного продукта систему «Деканат», которая обеспечит ввод, хранение и анализ информации о студентах и результатах их учебы.

Объектом исследования является процесс ввода, хранения и анализа информации о результатах обучения студентов.

Разрабатываемая система должна быть предназначена для автоматизации обработки, визуализации и анализа данных из БД, к которым относятся процессы, связанные с чтением и обновлением данных, добавлением и удалением данных.

Разрабатываемая система должна быть представлена в виде единого приложения. Разрабатываемая система должна отражать специфику предметной области и представлять собой единое целое.

Требования к надежности. В случае ошибок, вызванных некорректными действиями пользователя при работе, программа должна выдавать соответствующее сообщение и продолжать нормальную работу.

Создаваемое программное обеспечение должно обеспечить дружественный интерфейс пользователя – стандартное расположение компонентов управления (рекомендованное для данной операционной системы), дублирование команд, наличие всплывающих подсказок. При организации интерфейса необходимо использовать цвета в соответствии с правилами технической этики и текущей системной пользовательской палитры. Интерфейс пользователя должен обеспечивать эффективное взаимодействие пользователя с системой с минимальной необходимостью обращения к дополнительным справочным материалам.

* 1. Разработка технического задания

Название приложения: Деканат

Основные функции:

* Добавление, изменение и удаление данных о расписаниях, студентах и преподавателях;
* Поиск данных в таблице расписаний.

Требования к программному обеспечению:

* Приложение должно быть доступно на устройствах с операционной системой Windows;
* Приложение должно иметь понятный и простой интерфейс;
* Приложение должно обеспечивать безопасность и конфиденциальность данных.

Технические требования:

* Поддержка базы данных для хранения информации о расписании занятий;
* Использование библиотеки Entity Framework;
* Передача информации между клиентом и сервером должна быть защищенной.

Системные требования:

* Процессор: 4-ядерный или выше;
* ОЗУ: 4 Гб или выше;
* Свободное место на устройстве: не менее 5 Гб.

Тестирование:

* Обнаруженные ошибки и недоработки должны быть исправлены в кратчайшие сроки.

Поддержка:

* Поддержка приложения и обновления функций должны быть доступны для студентов в течение всего срока обучения.

Для реализации дополнительных функций базы может потребоваться введение дополнительных атрибутов, например, табельного номера и домашнего телефона студента, домашнего адреса и домашнего телефона преподавателя, должности преподавателя, рабочей программы, даты сдачи экзамена (зачета) и т.д.

О преподавателях есть следующая информация:

* Название кафедры;
* Должность;
* Стаж работы;
* ФИО;
* Оклад;
* Специальность;
* Табельный номер;
* Расписание.

Они обеспечивают проведение занятий по некоторым предметам.

Также курируют группу, о которой известно:

* Курс;
* Факультет;
* Направление;
* Количество студентов;
* Название;
* Номера телефонов студентов и куратора.

О студентах будет известна следующая информация:

* ФИО;
* Номер телефона;
* Группа.

Они ходят на пары по разным предметам:

* Название;
* Вид пар (Лекция, семинар или лабораторная);
* День недели;
* Время пары;
* Номер аудитории;
* Группа;
* Преподаватель.
  1. Анализ требований к программному продукту

Функциональные требования пользователя:

* автоматизация работы специалистов отдела расписания;
* возможность просмотра доступной информации для студентов и преподавателей.

Входные данные:

Входными данными при работе с программным продуктом должны быть данные о преподавателях, группах студентов, типах занятий, аудиториях и времени, проводимых занятий. Входной язык приложения - русский.

Выходные данные:

Выходными данными при работе программы являются данные (расписание), выводимые на экран пользователя. Язык выходных данных в приложении - русский.

Требования к интерфейсу:

* Понятность и логичность;
* Функциональность.

Требования к надежности:

* контроль вводимой информации, т.е. возможность отслеживания ошибок, допускаемых пользователем, и последующей реакции программы на них.
  1. Определение основных технических решений

В качестве среды разработки для написания программы был выбран Microsoft Visual Studio. Это интегрированная среда разработки (IDE), которая включает в себя редактор кода, отладчик и тестовую среду. Visual Studio поддерживает широкий спектр языков программирования, в том числе С#.

Windows Forms — это платформа пользовательского интерфейса для создания классических приложений Windows. Она обеспечивает один из самых эффективных способов создания классических приложений с помощью визуального конструктора в Visual Studio. Такие функции, как размещение визуальных элементов управления путем перетаскивания, упрощают создание классических приложений.

С# – это современный объектно-ориентированный язык программирования, который был разработан компанией Microsoft. Он имеет динамическую типизацию, а также возможности для создания GUI (графических пользовательских интерфейсов). Кроме того, в C# реализованы средства безопасности в памяти и управление игроками.

Состав функций программного продукта:

* Функции чтения/добавления/удаления/изменения данных о расписаниях, студентах и преподавателях в базе данных.
* Функция поисковой системы для вкладки «Расписание».
* Функция автозаполнения полей ввода по номеру записи для всех вкладок.
* Функция обновления таблицы базы данных.

Режимы функционирования:

* Режим добавления новых данных. В этом режиме можно добавлять новых преподавателей, студентов, а также информацию о занятиях, которые нужно распределить в расписание.
* Режим просмотра расписания. В этом режиме пользователь может просмотреть готовое расписание.
* Режим изменения расписания. Если возникла необходимость внести изменения в готовое расписание, можно выбрать этот режим и внести нужные корректировки в расписание занятий.
  1. Проектирование программного продукта

Для программы создали 4 диаграммы: Сущность-связь, Диаграмма потоков данных, Функциональная модель бизнес-процесса второго уровня. Построенные диаграммы представлены на рисунках 1.1-1.4.

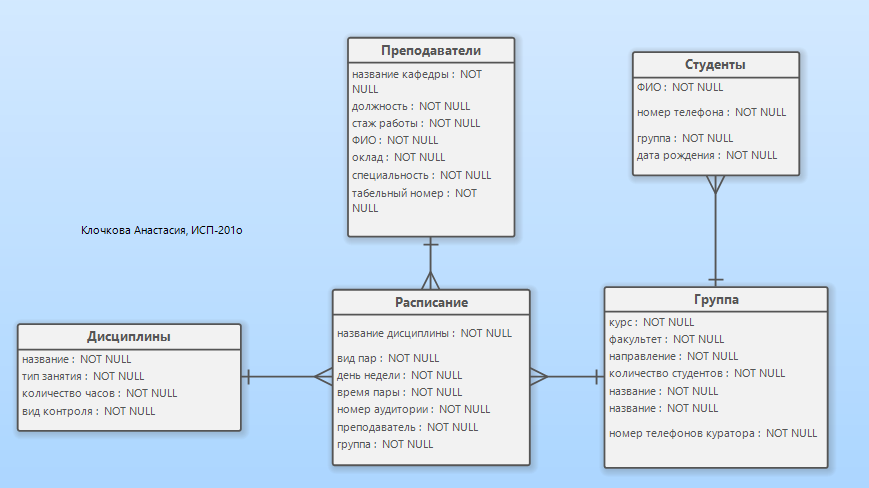


Рисунок 1.1 – Диаграмма сущность-связь

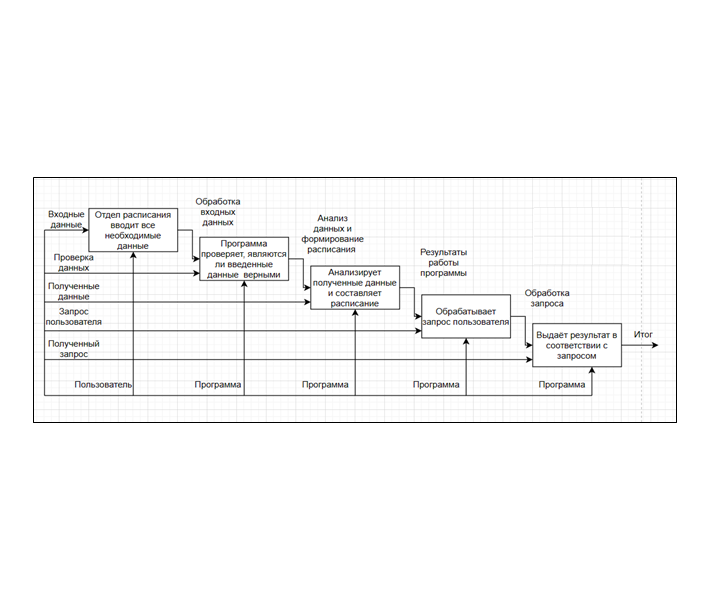


Рисунок 1.2 – Диаграмма потоков данных

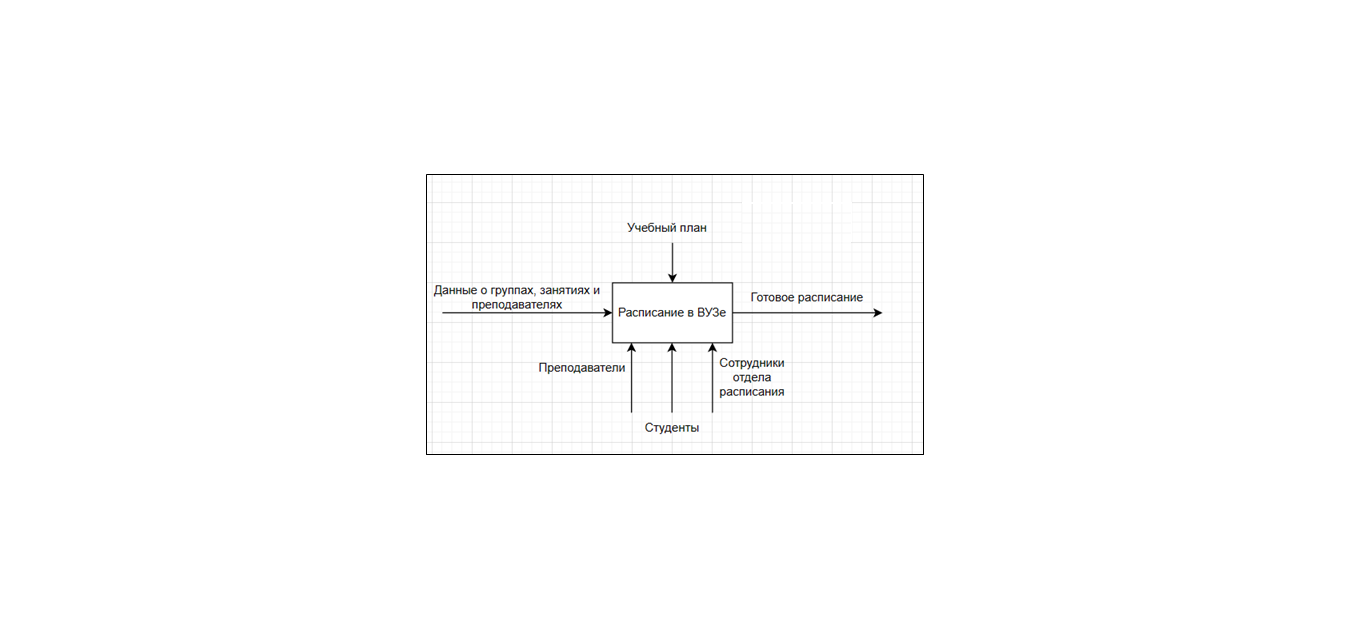


Рисунок 1.3 – Диаграмма IDEF0



Рисунок 1.4 – Функциональная модель бизнес-процесса второго уровня

* 1. Разработка программного продукта

Данная программа разработана для составления расписания в ВУЗе. Она состоит из 3 вкладок:

* Расписание
* Преподаватели
* Студенты

В программе реализовано всего 7 классов (Timetable, Students, Teacher, Groups, Specialist, Faculties, Discipline). Классы используются для связи программы с хранилищем данных. Они позволяют упростить доступ к базе данных и сделать его более безопасным и надежным. Все эти классы содержат в себе поля, которые равны атрибутам сущностей в базе

Листинг кода программы представлен в приложении А.

Конструктор программы представлен на рисунках 1.5-1.7:

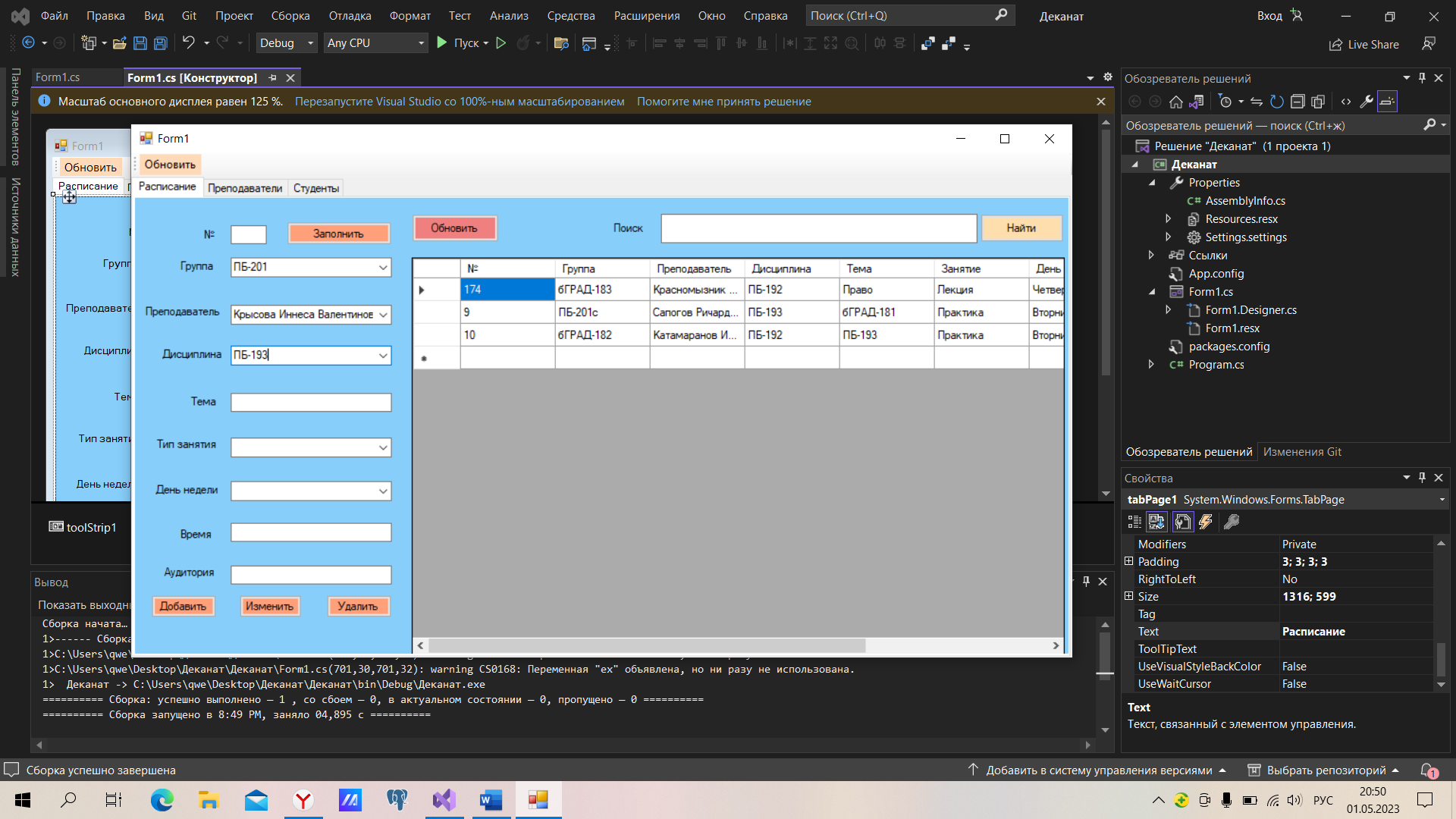


Рисунок 1.5 – Конструктор программы

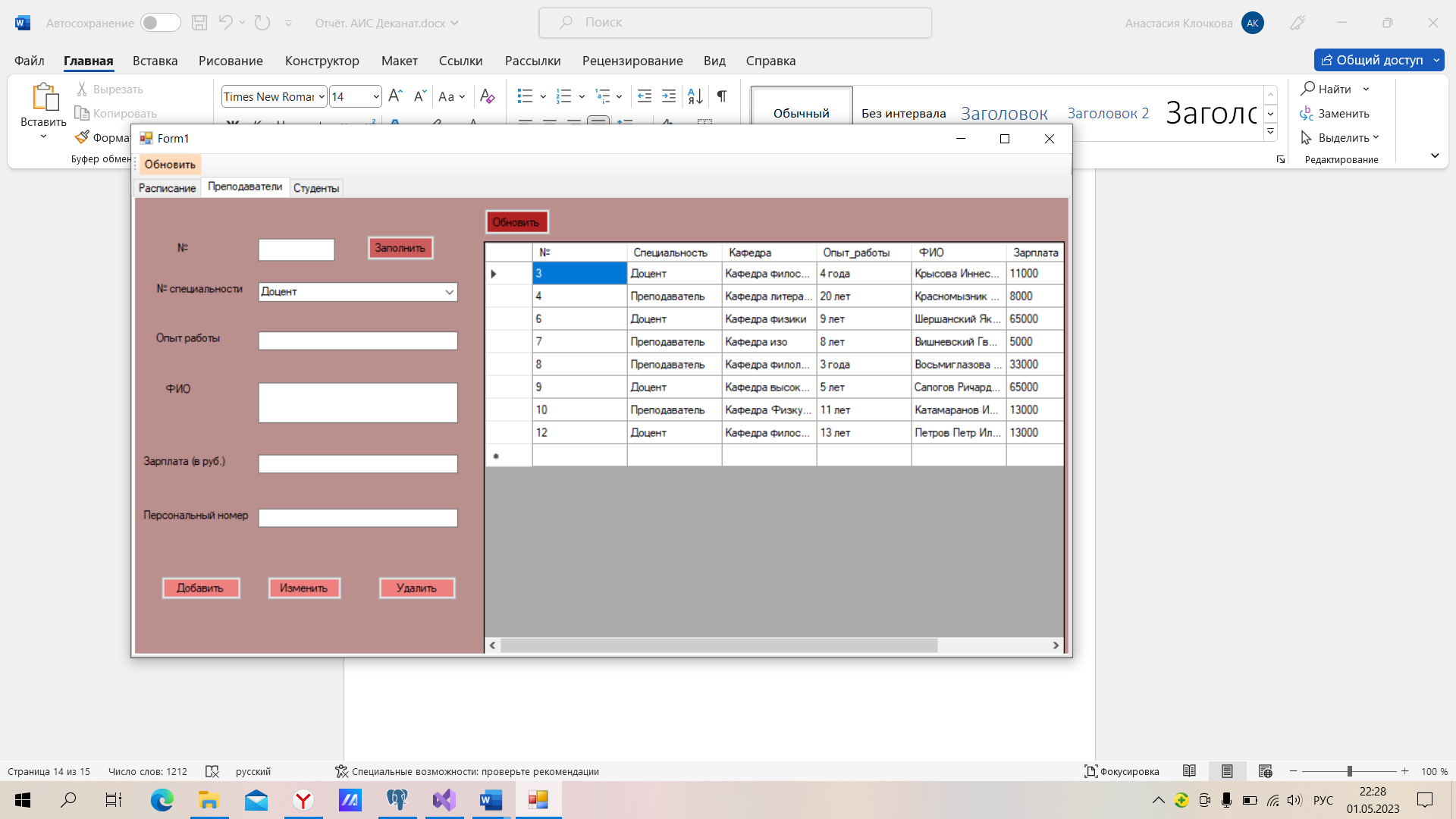


Рисунок 1.6 – Конструктор программы

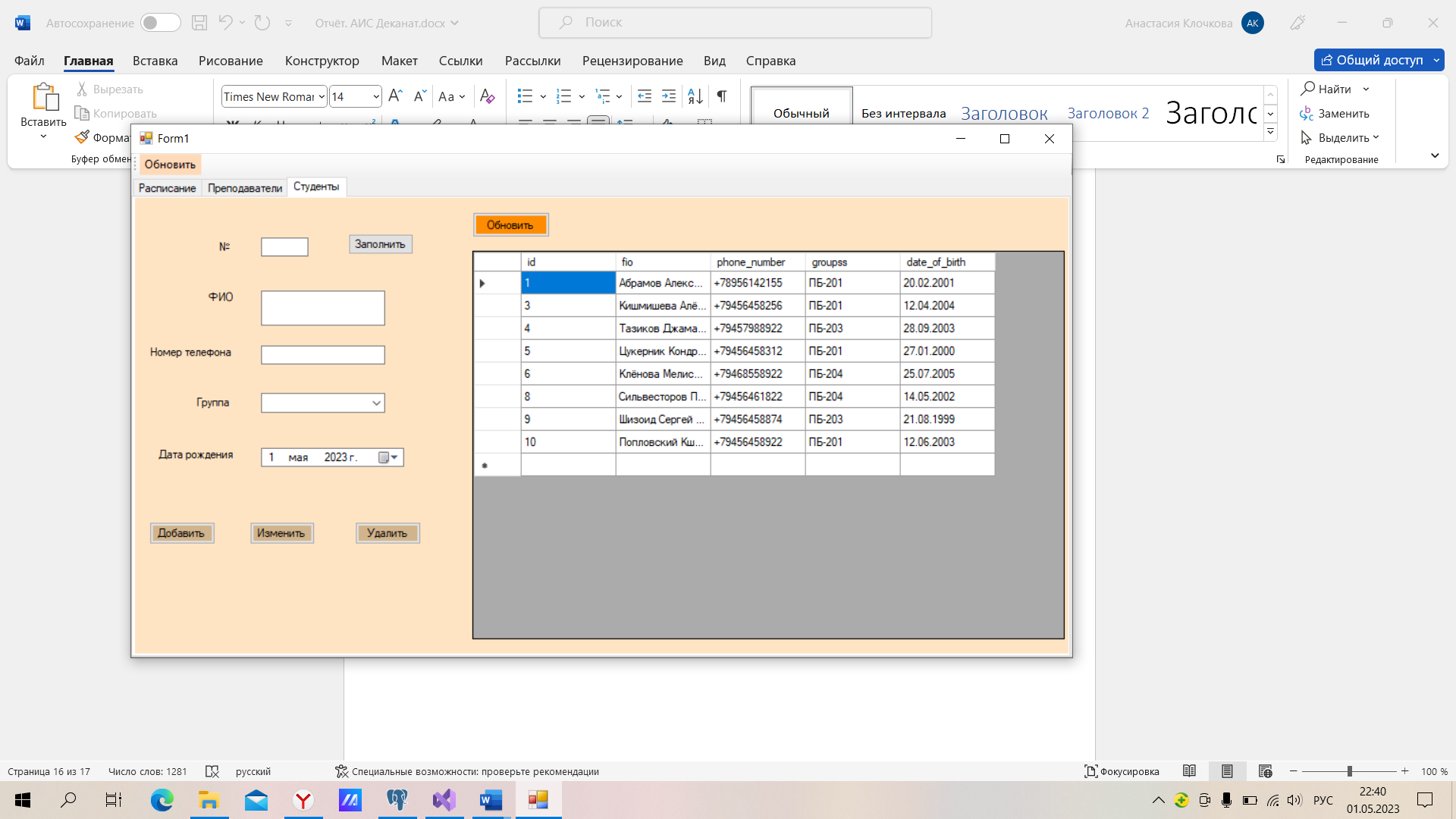


Рисунок 1.7 – Конструктор программы

* 1. Тестирование программного продукта

Мы разработали программный продукт, произвели отладку и тестирование. Результаты тестирования представлены на рисунках 1.12-1.19.

**Для начала работы в приложении:**

1. Запустить программу.
2. Начать вводить данные, как описано ниже (рисунок 1.12).

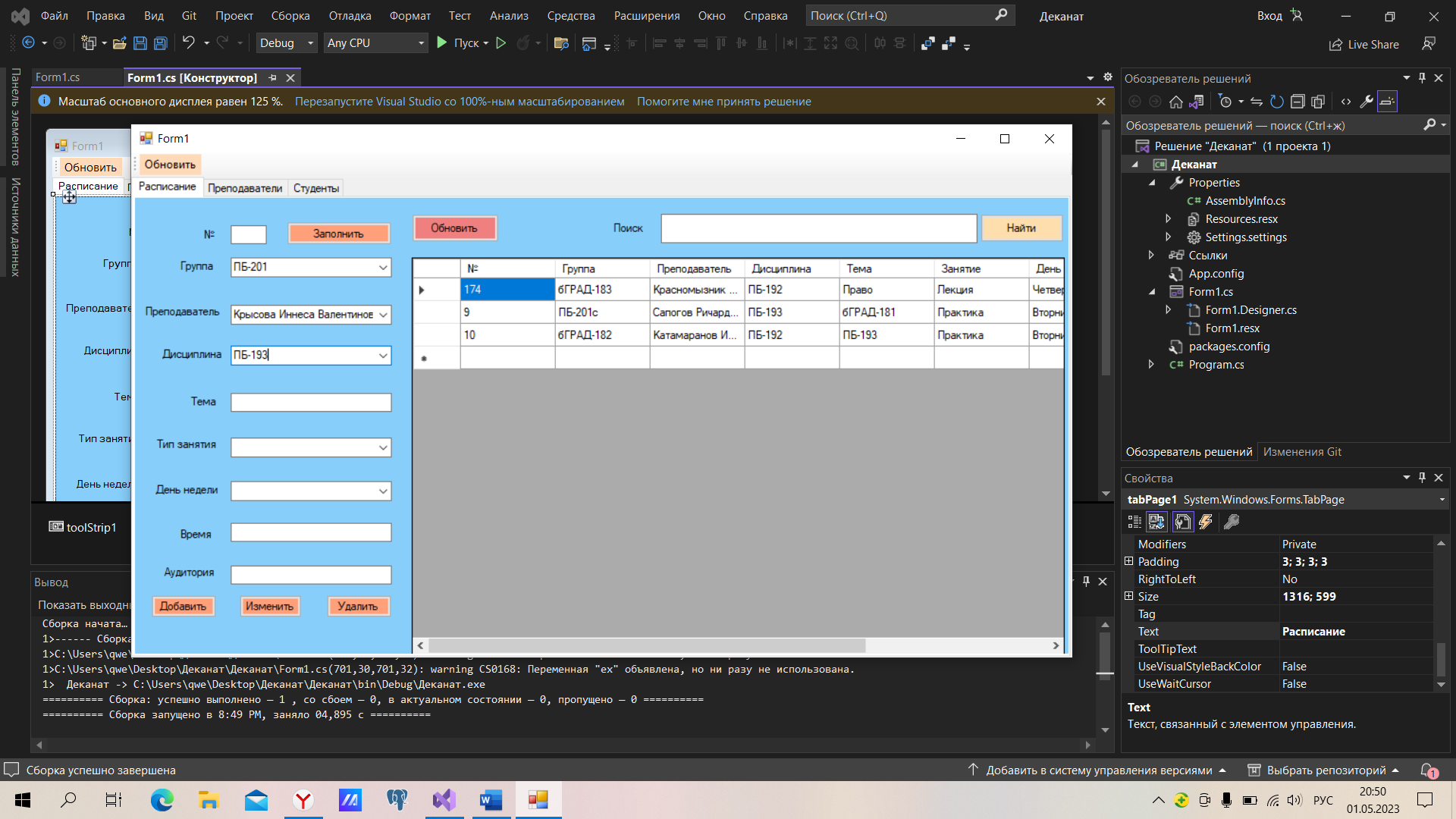


Рисунок 1.12 – АИС «Деканат»

Расписание создаётся по кнопке «Добавить». Вся справочная информация выбирается из выпадающего списка (рисунок 1.13).

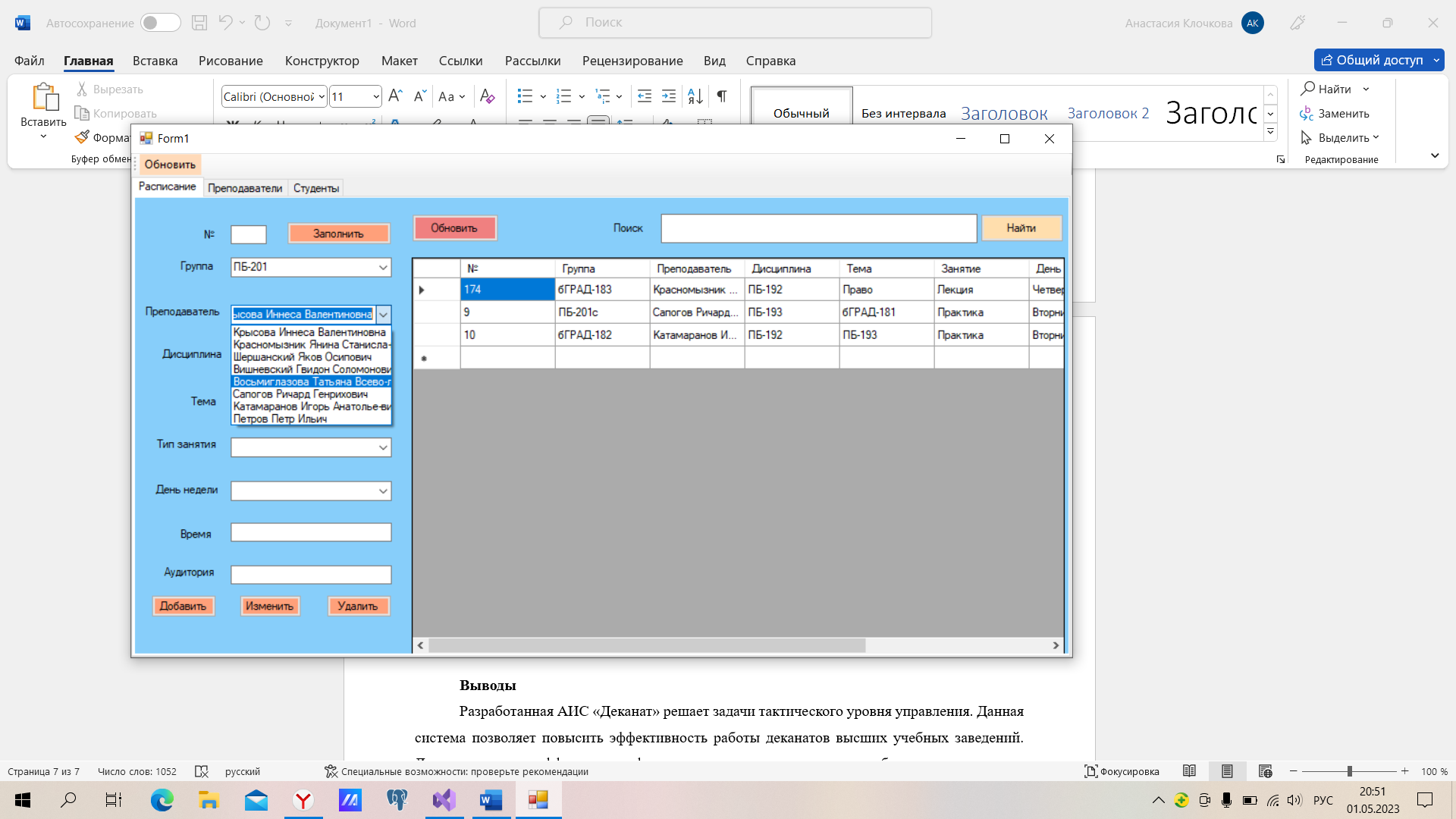


Рисунок 1.13 – АИС «Деканат»

Предусмотрена проверка на ввод всех необходимых данных, как показано на рисунке 1.14.

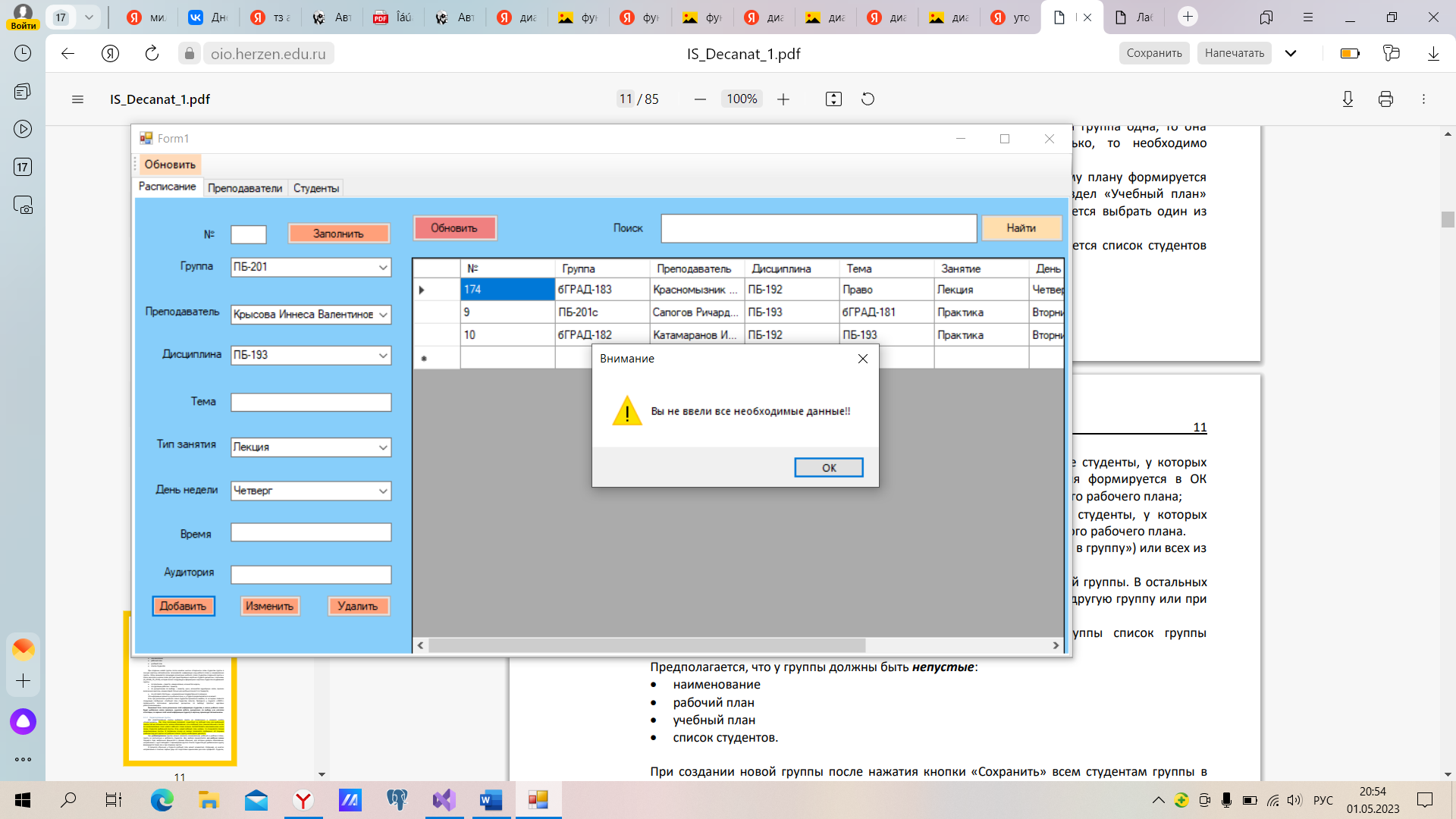


Рисунок 1.14 – АИС «Деканат»

При добавлении записи вылетает окно «Информация добавлена» и автоматически обновляет таблицу (рисунок 1.15).

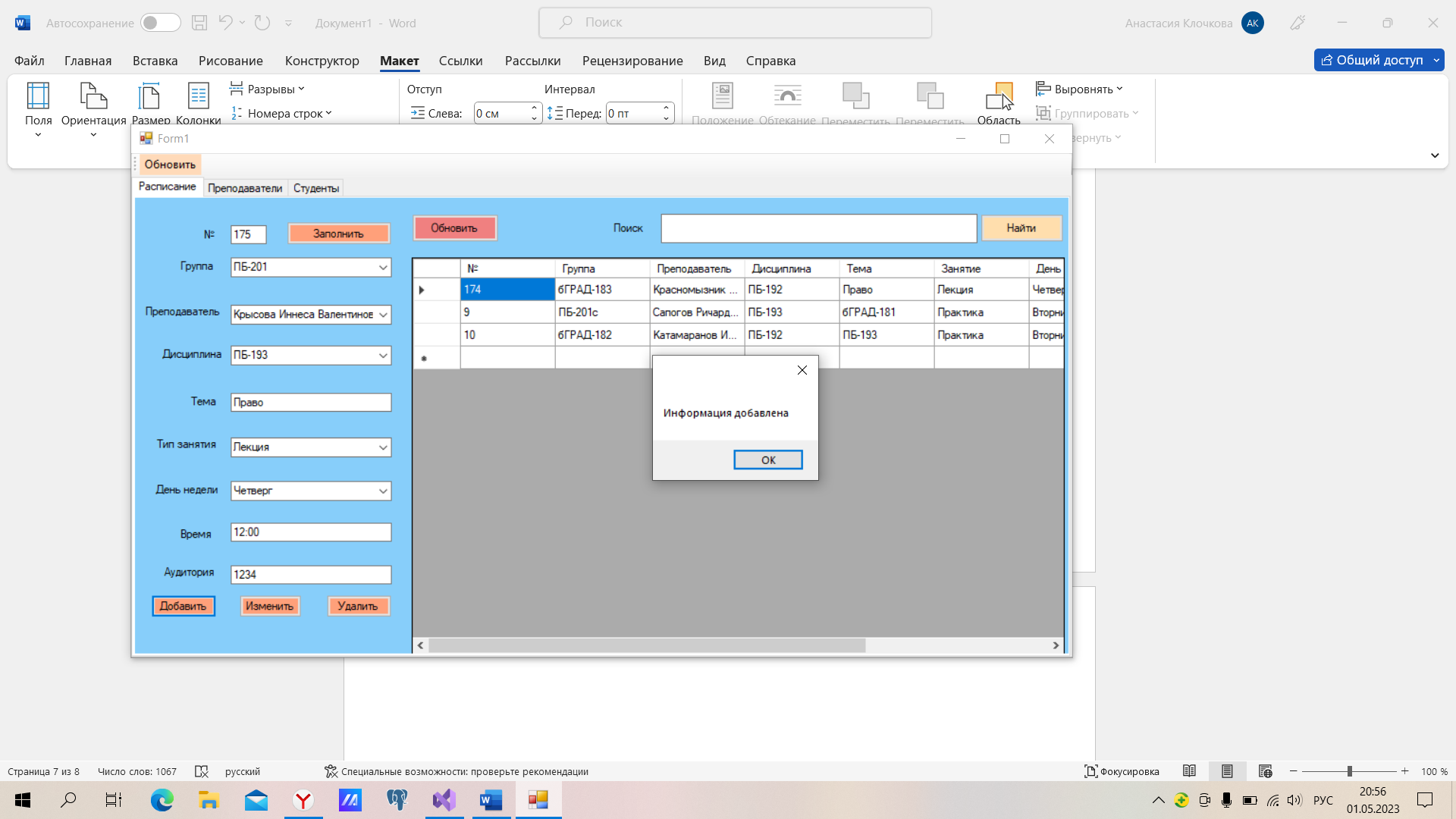


Рисунок 1.15 – АИС «Деканат»

Предусмотрена функция Автозаполнения полей ввода по номеру (рисунок 1.16).

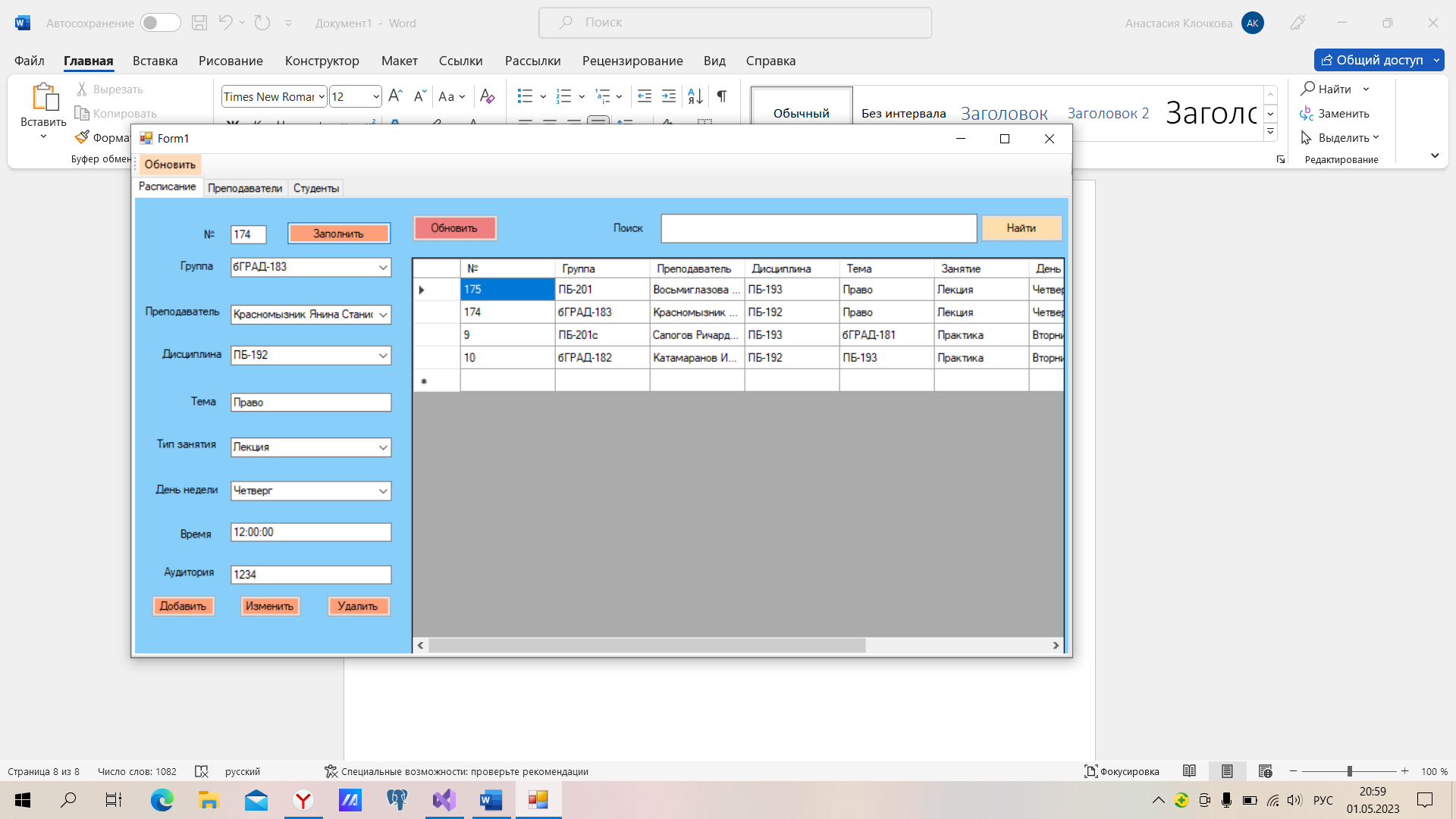


Рисунок 1.16 – АИС «Деканат»

Если не заполнить необходимое поле для ввода, то вылетает окно «Необходимо заполнить данные».

Расписание изменяется по кнопке «Изменить». Вся справочная информация выбирается из выпадающего списка (рисунок 1.17).

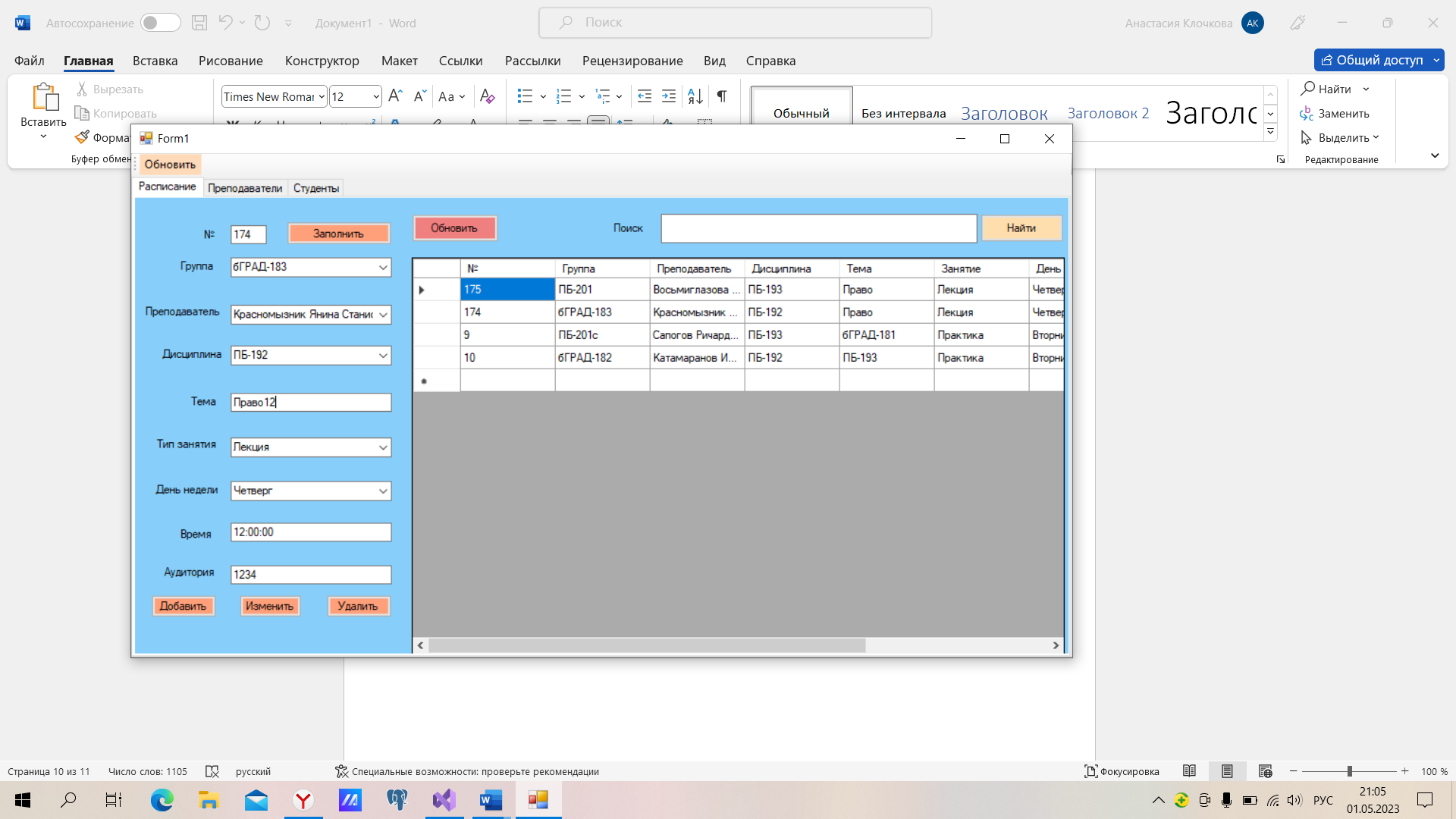


Рисунок 1.17 – АИС «Деканат»

При изменении записи вылетает окно «Информация изменена» и автоматически обновляет таблицу (рисунок 1.18).

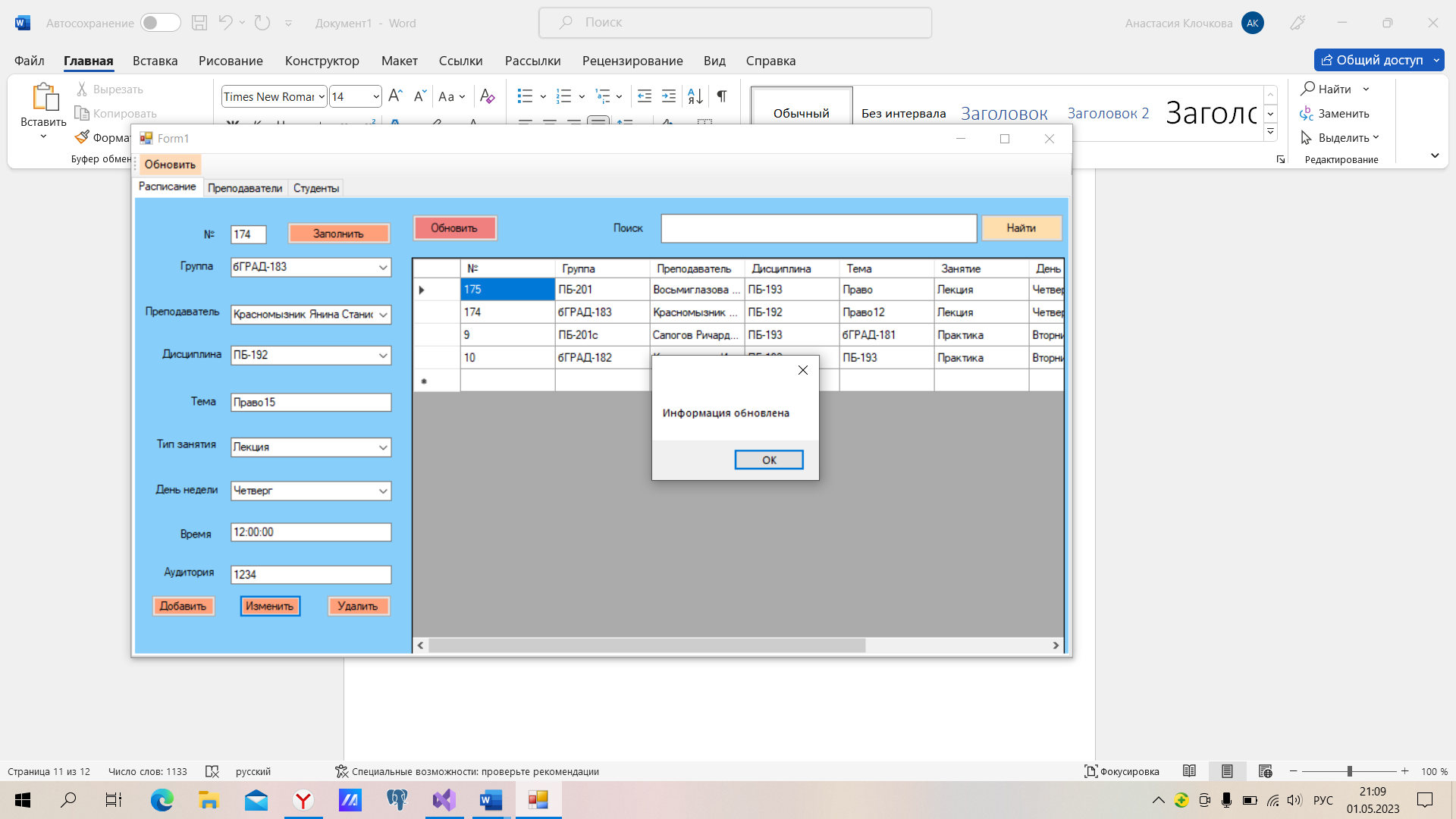


Рисунок 1.18 – АИС «Деканат»

Расписание удаляется по кнопке «Удалить». В поле «№» вводится номер записи, которую необходимо удалить.

При удалении записи вылетает окно «Информация удалена» и автоматически обновляет таблицу (рисунок 1.19).

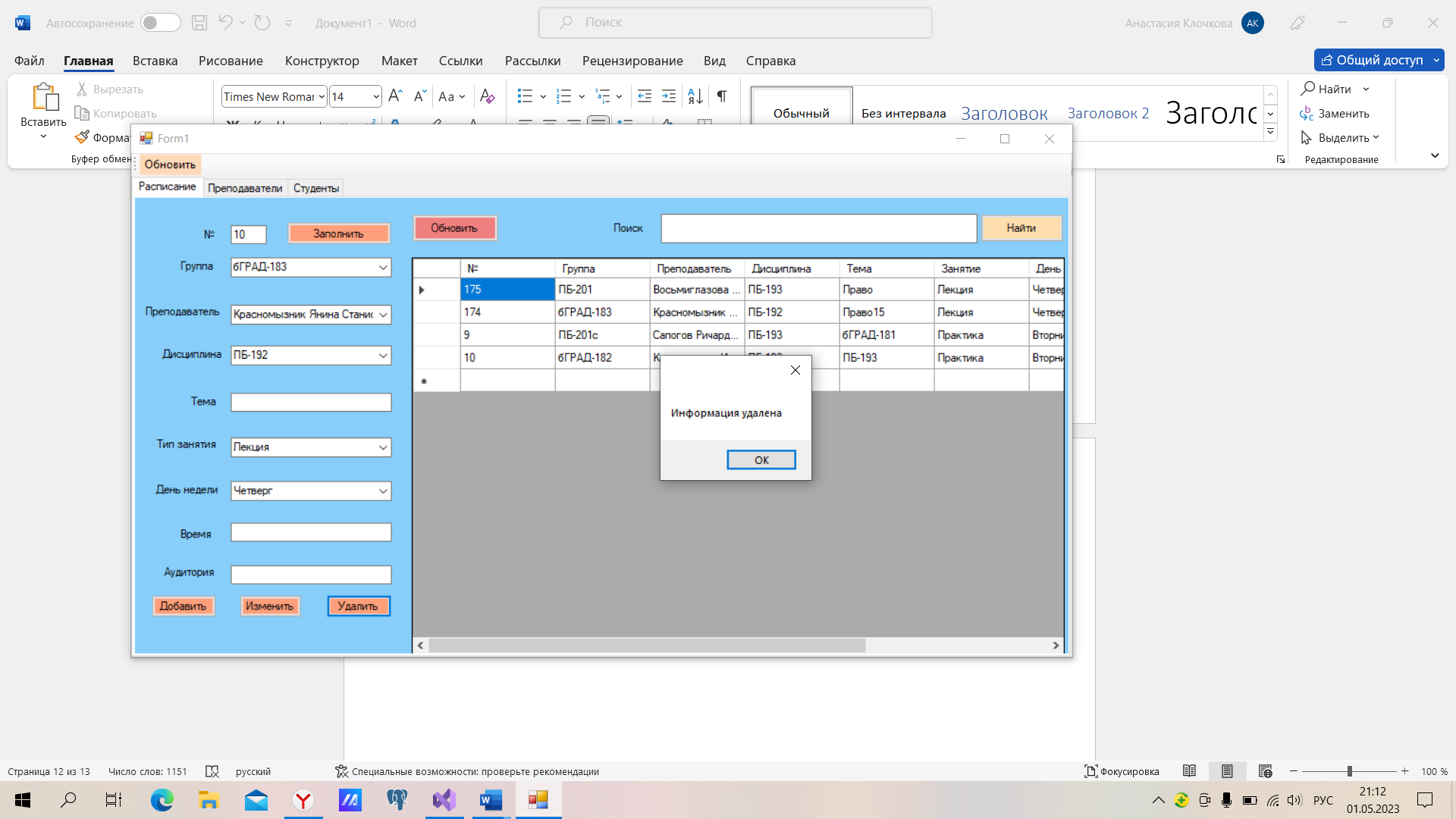


Рисунок 1.19 – АИС «Деканат»

Также предусмотрена проверка на незаполненные поля (рисунок 1.20).

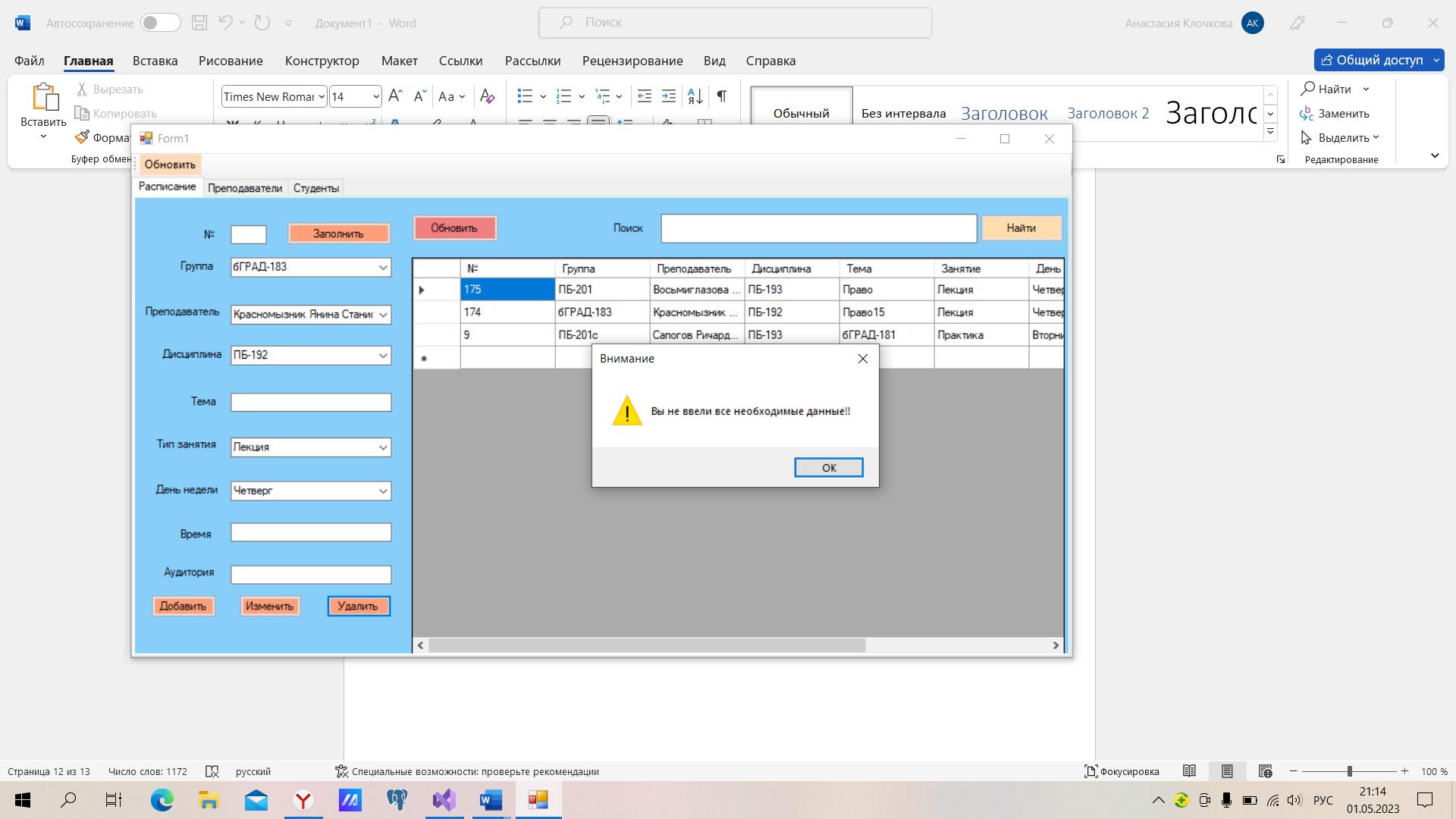


Рисунок 1.20 – АИС «Деканат»

Предусмотрен фильтр данных через поисковую структуру (рисунок 1.21).

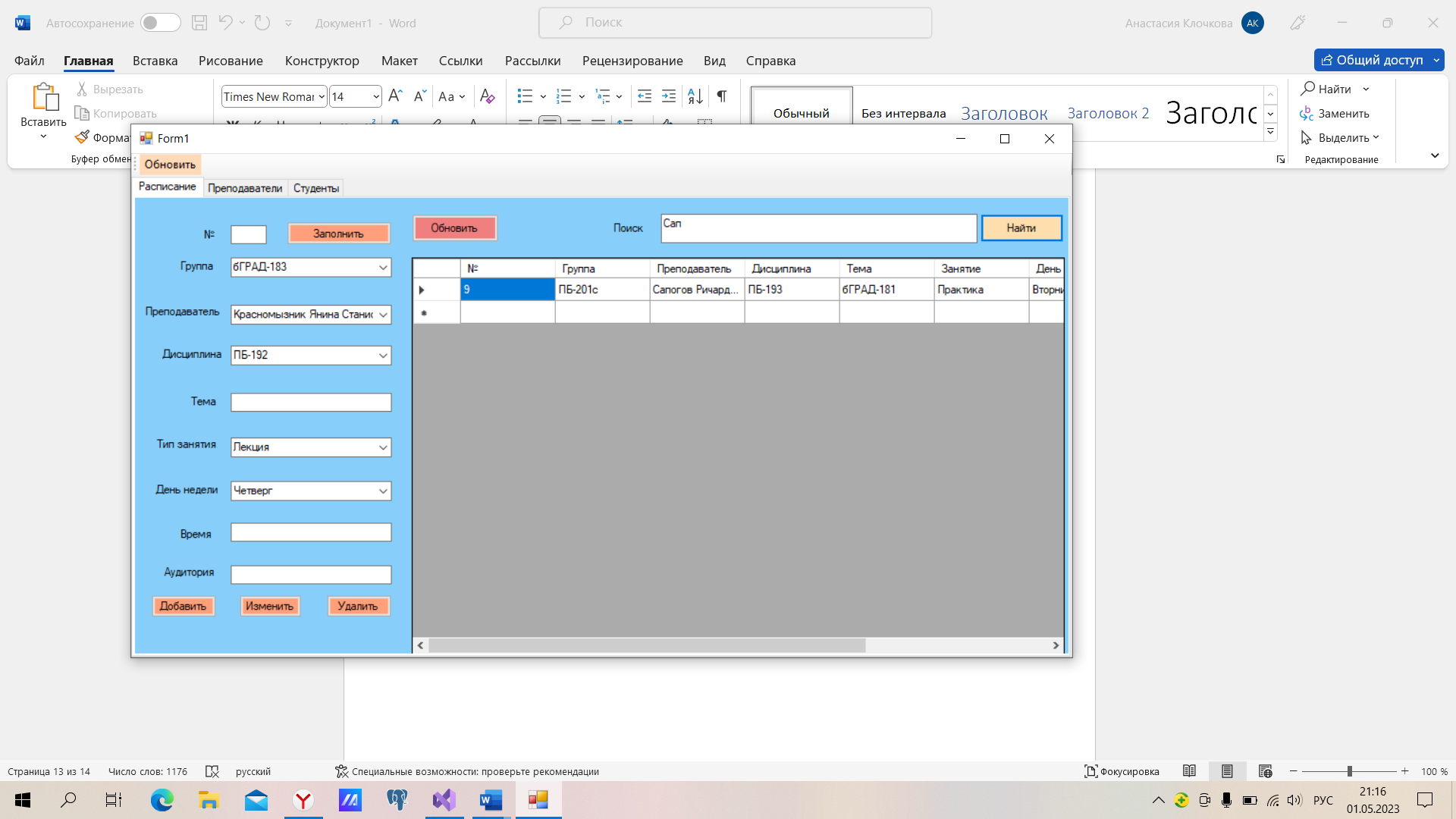


Рисунок 1.21 – АИС «Деканат»

Кнопка «Обновить» не только обновляет данные, но и сбрасывает фильтр (рисунок 1.22).

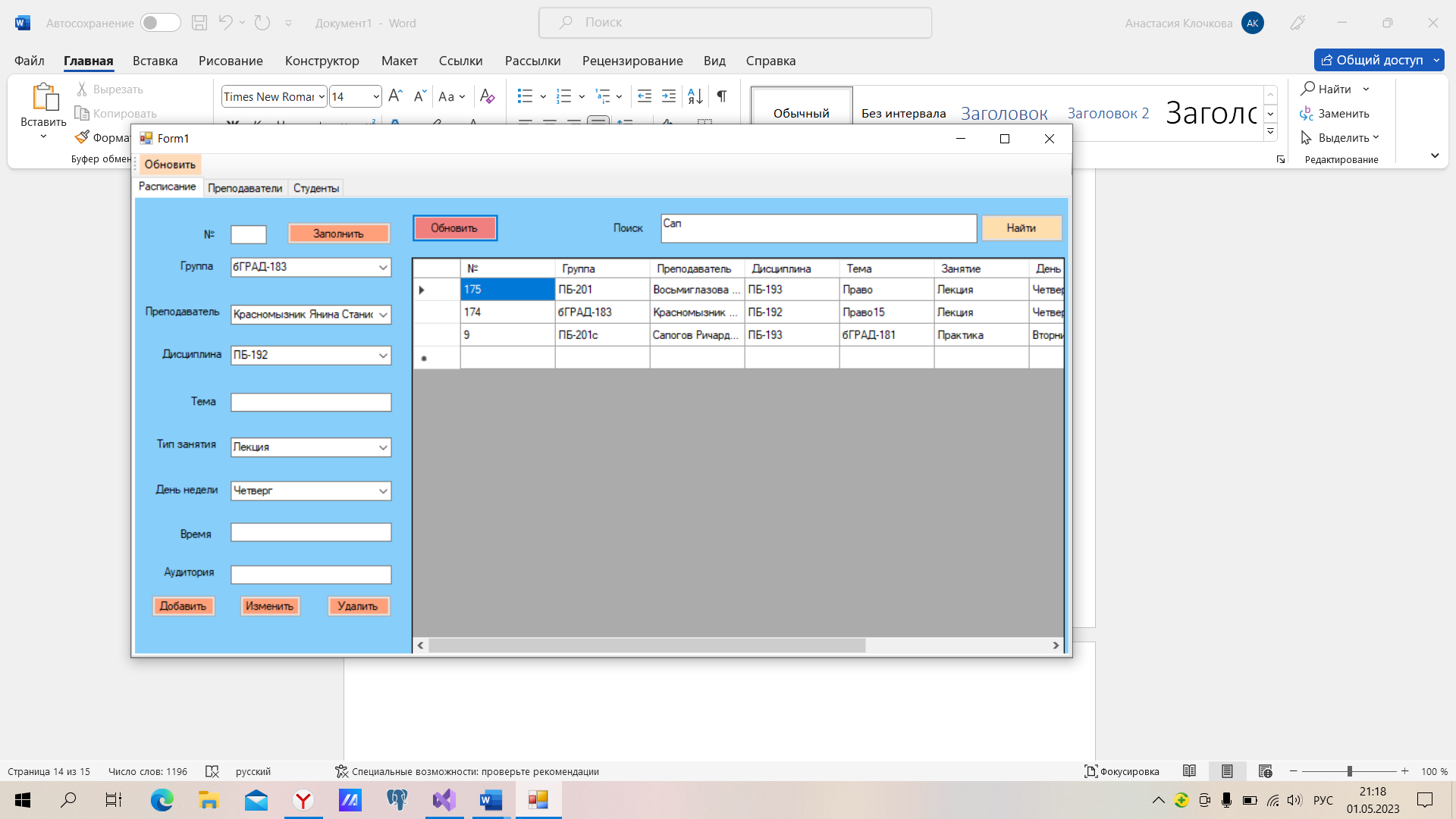


Рисунок 1.22 – АИС «Деканат»

На вкладке «Преподаватель» для редактирования данных также есть кнопки «Добавить», «Изменить», «Удалить» (рисунок 1.23).

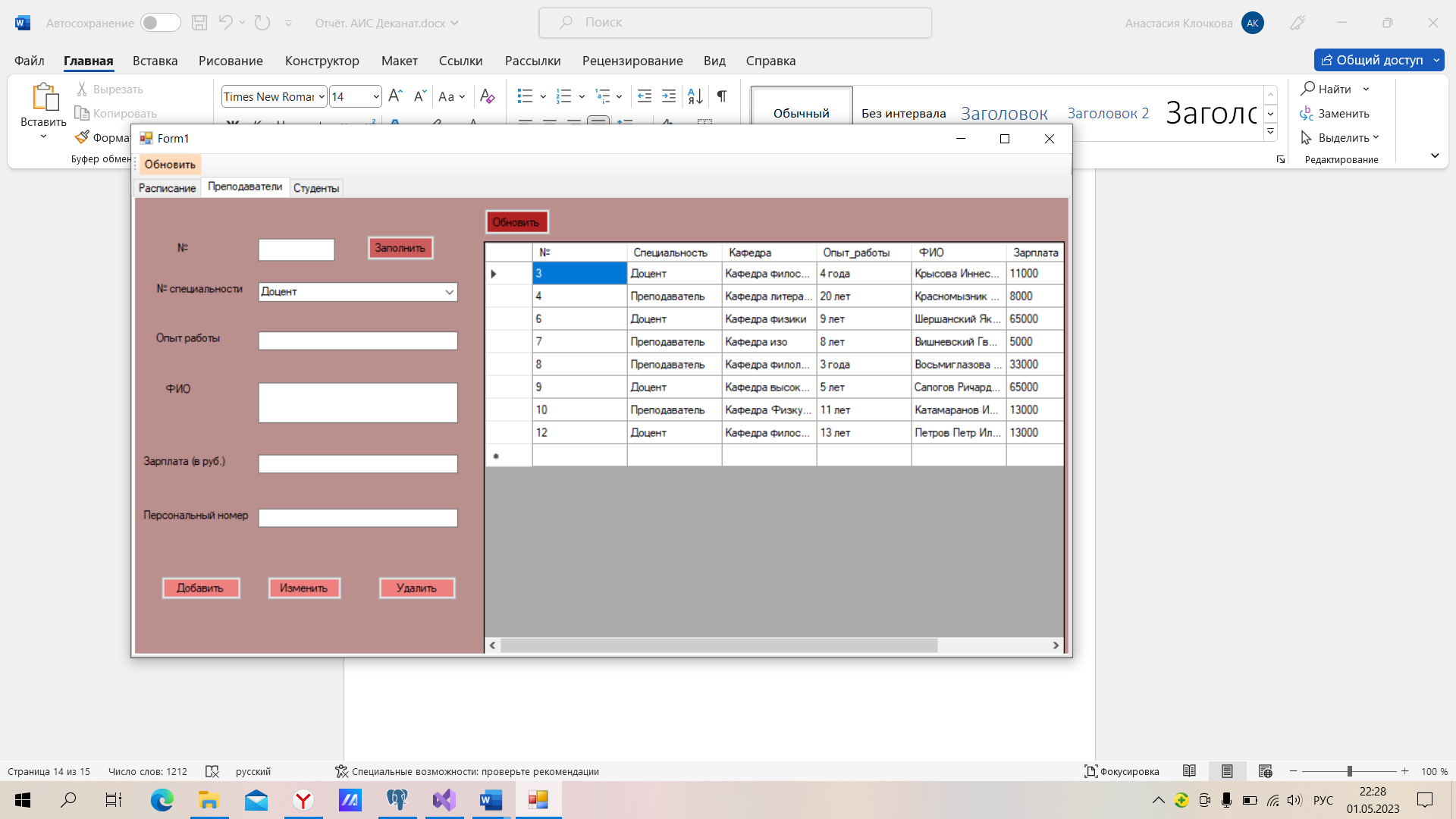


Рисунок 1.23 – АИС «Деканат»

Проверка полей на пустоту при добавлении записи (рисунок 1.24).

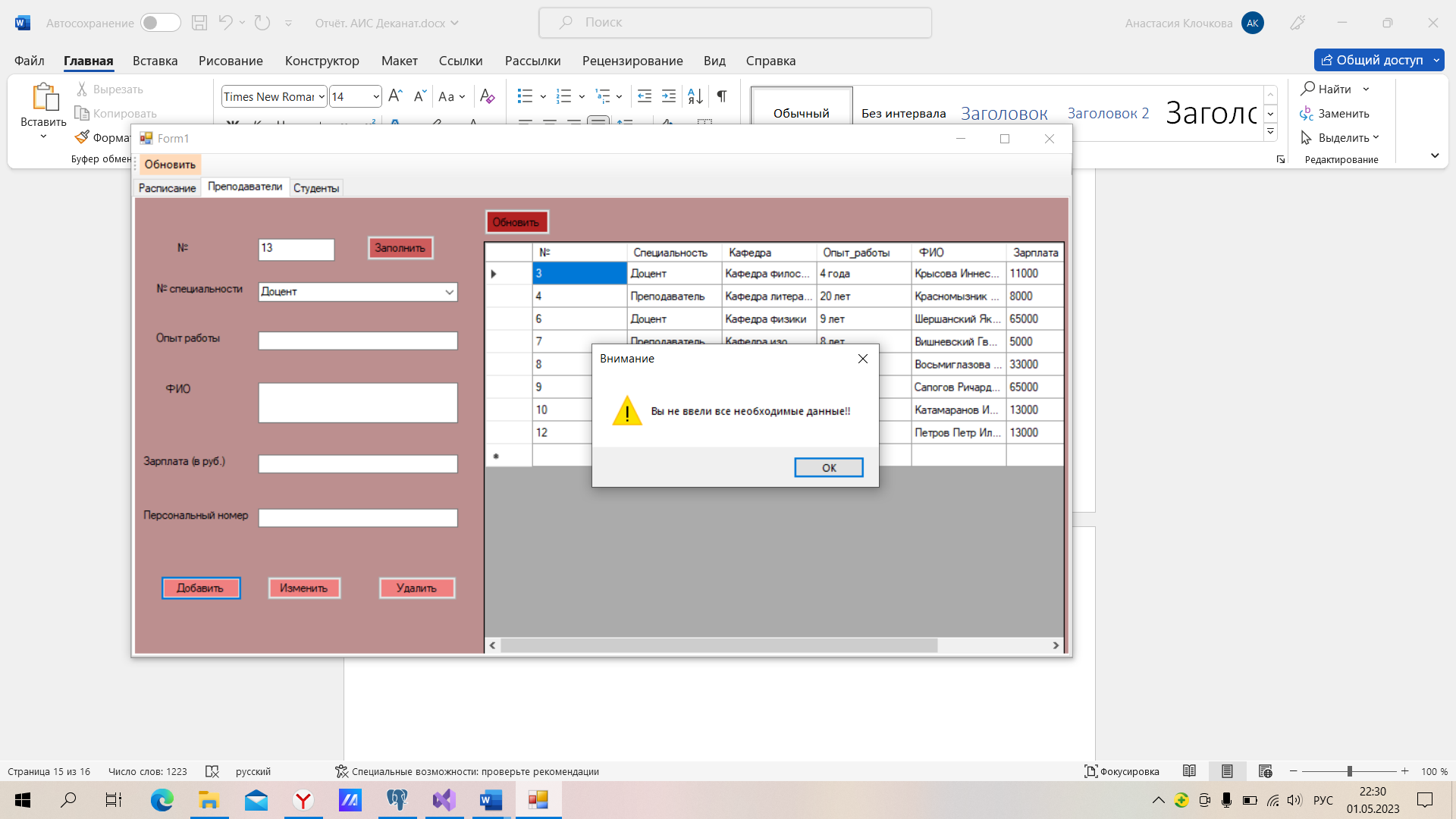


Рисунок 1.24 – АИС «Деканат»

При добавлении записи вылетает окно и данные автоматически обновляется в таблице (рисунок 1.25).

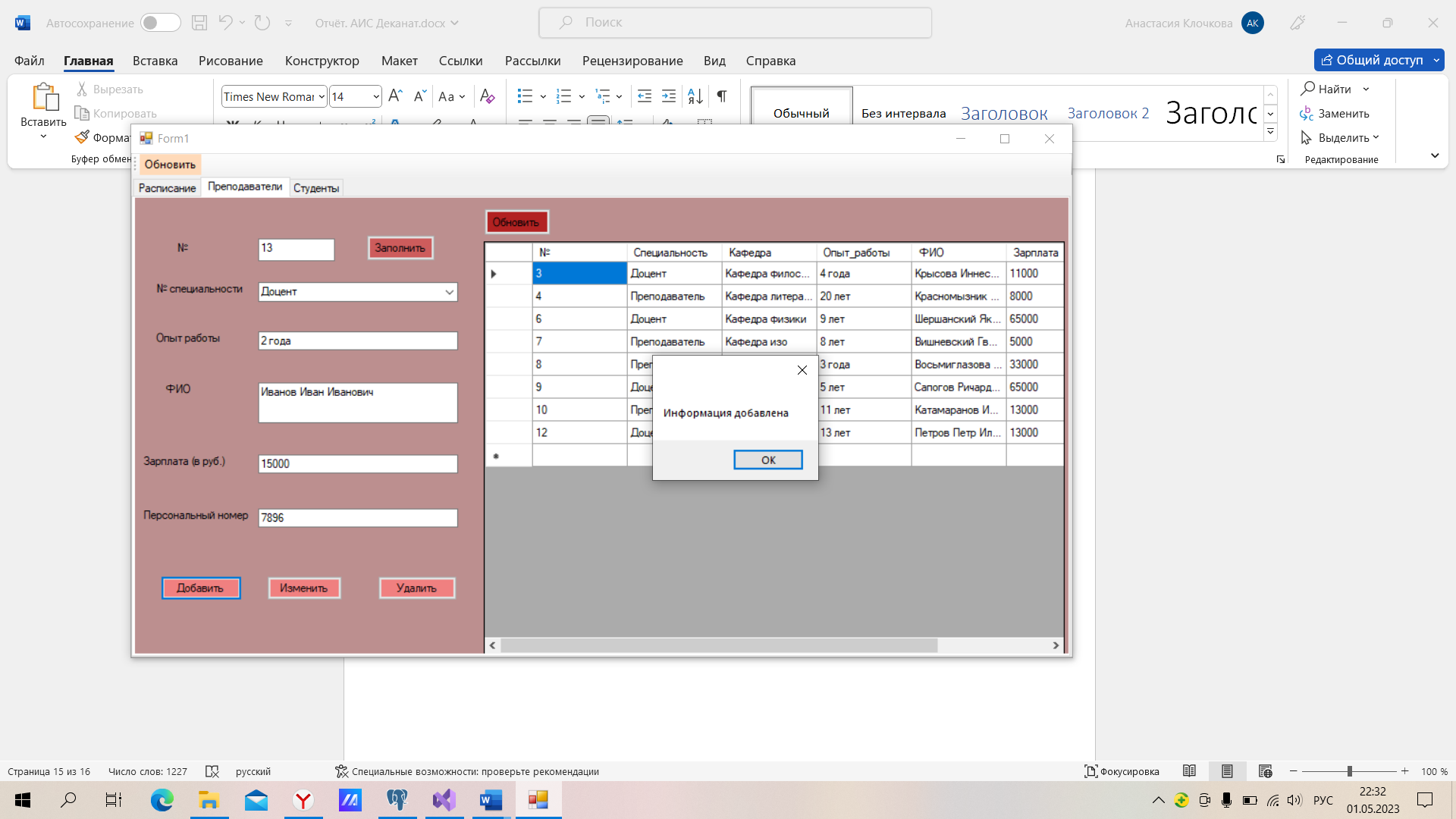


Рисунок 1.25 – АИС «Деканат»

Кнопки «Изменить» и «Удалить» работают аналогично.

Также предусмотрена функция автозаполнения всех полей по номеру (рисунок 1.26).

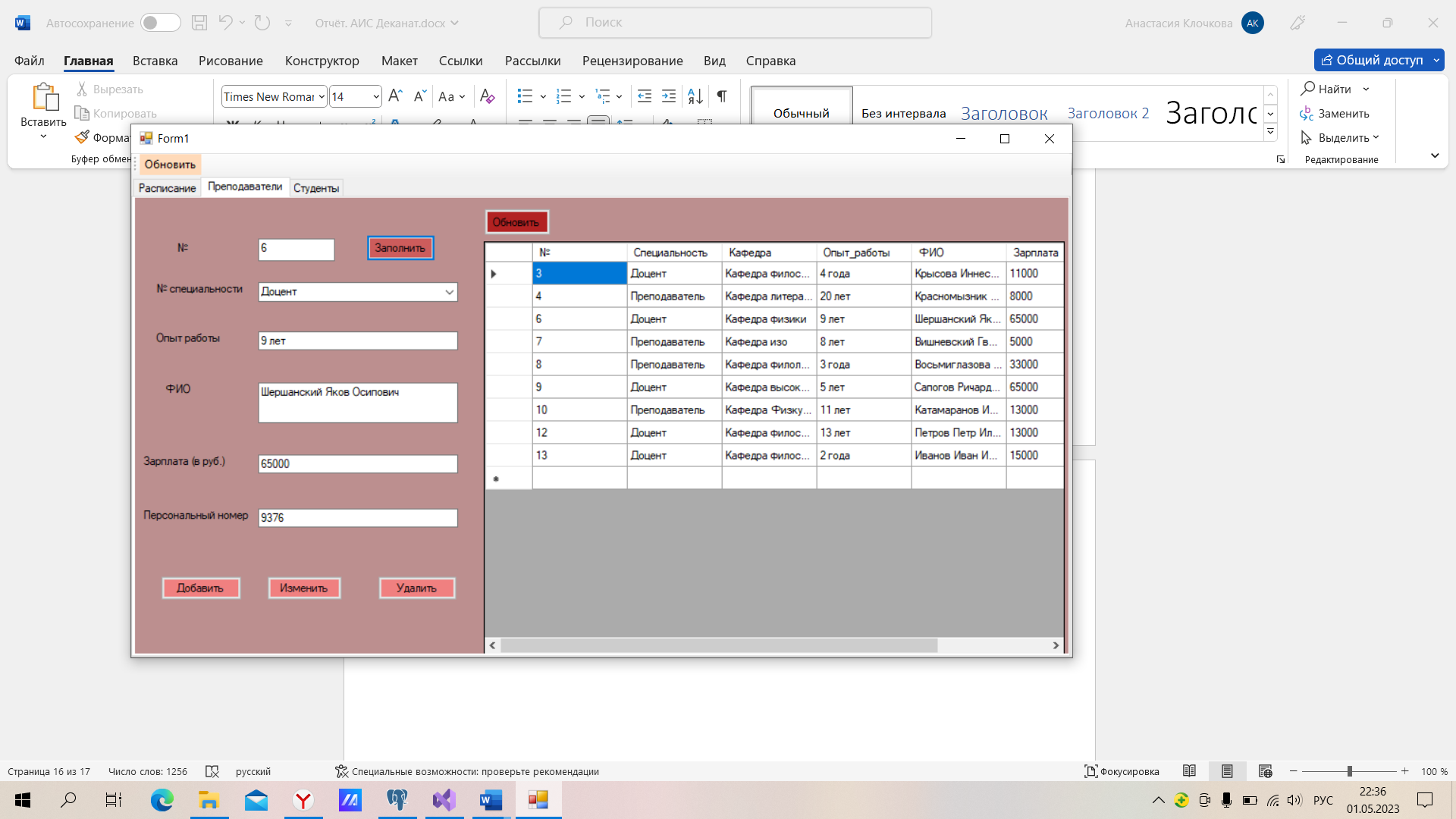


Рисунок 1.26 – АИС «Деканат»

Также есть кнопка обновления данных.

Вкладка «Студенты» спроектирована схожим образом с вкладкой «Преподаватели».

Для редактирования данных используются кнопки «Добавить», «Изменить», «Удалить» (рисунок 1.27).

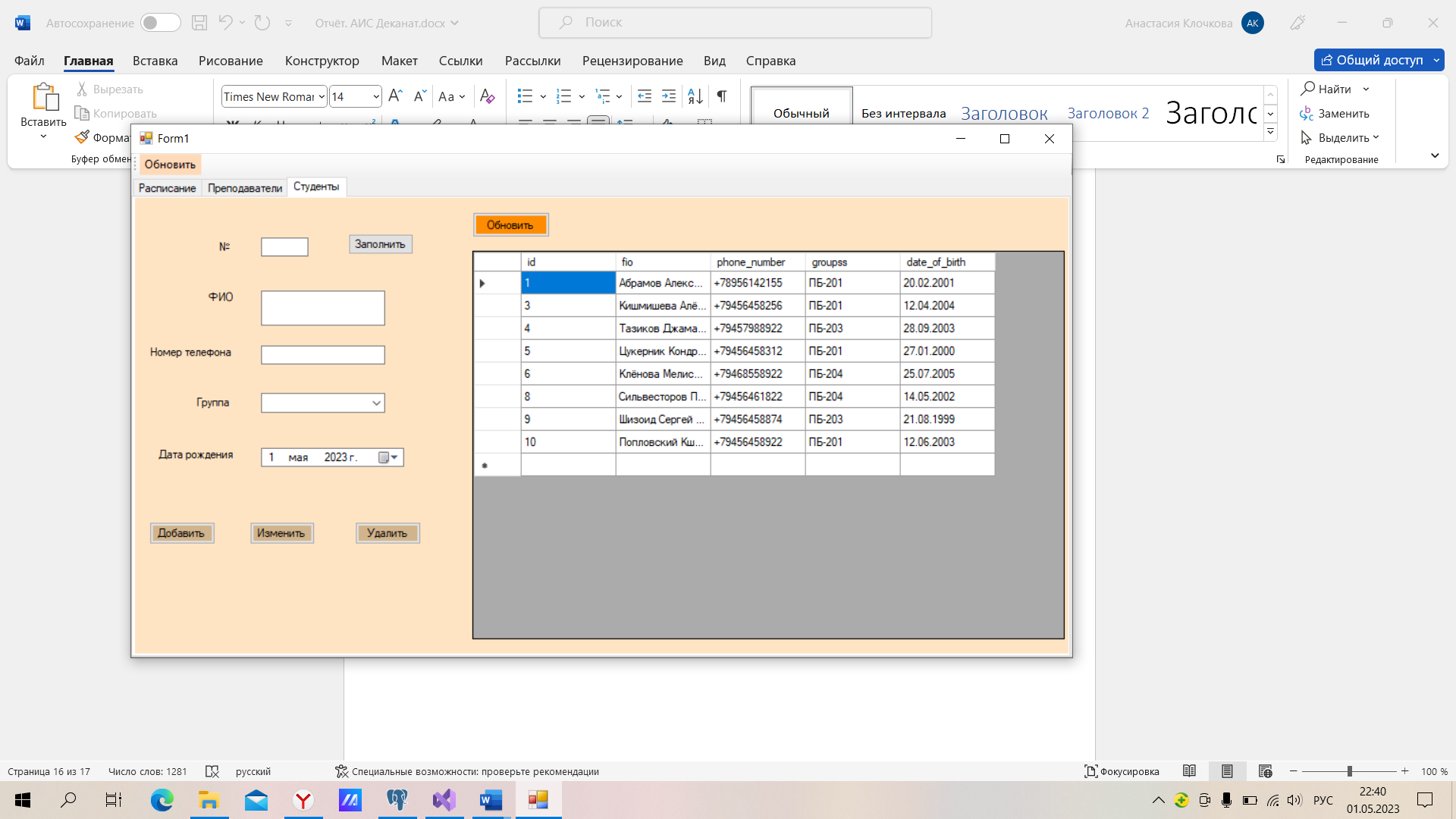


Рисунок 1.27 – АИС «Деканат»

При добавлении, изменении, удалении записи вылетает окно и данные автоматически обновляется в таблице (рисунок 1.28).



Рисунок 1.28 – АИС «Деканат»

При наличии пустых полей вылетает окно (рисунок 1.29).

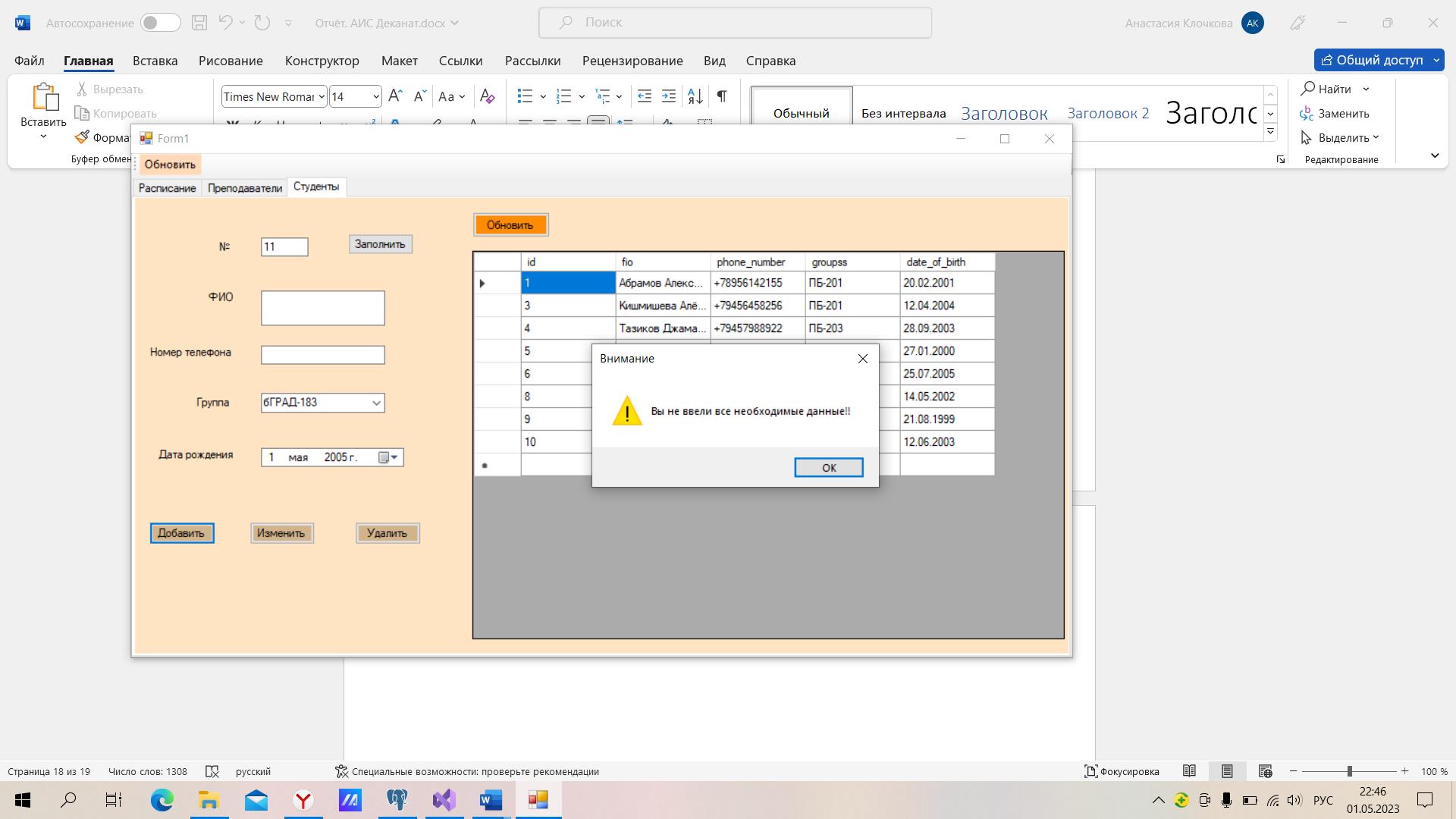


Рисунок 1.29 – АИС «Деканат»

Имеется функция автозаполнения по номеру (кнопка «Заполнить») (рисунок 1.30).

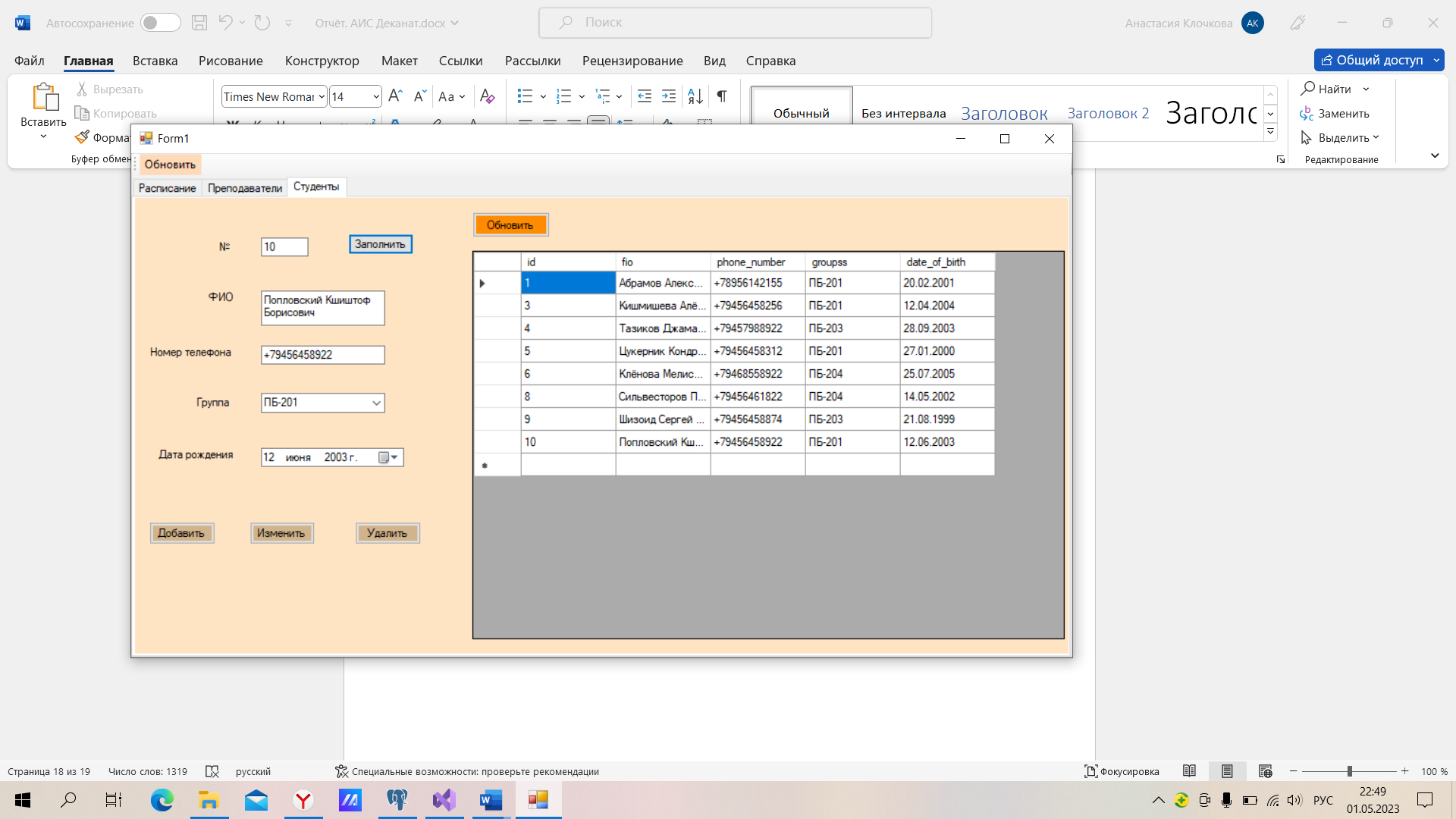


Рисунок 1.30 – АИС «Деканат»

Также есть кнопка для обновления данных.

# Заключение

В данной учебной практике было разработано приложение, которое взаимодействует с базой данных.

Изначально мы написали техническое задание, провели анализ функциональных требований и выделили основные технические решения. В приложении Draw io была разработана диаграмма сущность-связь и диаграмма потоков данных. Далее, разработали программный продукт, провели отладку и тестирование. Затем мы произвели оценку качества программного продукта.

Была установлена СУБД – PostgreSQL. На основе реляционной модели данных, полученной ранее, была создана база данных.

Благодаря созданию программы, заметно снизится количество ошибок, совершаемых сотрудниками отдела расписания в процессе работы и повысится производительность. Пользователи же в свою очередь получат простой и удобный интерфейс, а также возможность быстро получить необходимую информацию.

Разработанная АИС «Деканат» решает задачи тактического уровня управления. Данная система позволяет повысить эффективность работы деканатов высших учебных заведений. Для успешного и эффективного функционирования вуза в целом необходимо комплексное внедрение информационно-коммуникационных технологий, охватывающее все без исключения функциональные подсистемы учебного заведения (такие как деканаты, библиотека, приёмная комиссия, бухгалтерия, кафедры и др.)

# Список использованных источников

1. Стасышин, В.М.Базы данных: технологии доступа: учебное пособие для среднего профессионального образования/ В.М.Стасышин, Т.Л.Стасышина. — 2-е изд., испр. и доп.— Москва: Издательство Юрайт, 2020. — 164 с. — Режим доступа: https://urait.ru/bcode/455863 -ЭБС Юрайт.
2. Туманов В.Е. Основы проектирования реляционных баз данных [Электронный ресурс]/ Туманов В.Е.— Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 502 c.— Режим доступа: http://www.iprbookshop.ru/52221.html.— ЭБС «IPRbooks»
3. Култыгин О.П. Администрирование баз данных. СУБД MS SQL Server [Электронный ресурс]: учебное пособие/ Култыгин О.П. - Электрон.текстовые данные. - М.: Московский финансово-промышленный университет «Синергия», 2012. - 232 c. - Режим доступа:http://www.iprbookshop.ru/17009
4. <https://www.sql.ru/-> SQL Сайт, посвященный SQL, программированию, базам данных, разработке информационных систем

Приложение А

using Npgsql;

using Npgsql.Internal;

using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.OleDb;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Xml.Linq;

using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Деканат

{

public partial class Form1 : Form

{

NpgsqlConnection conn = new NpgsqlConnection("Server=localhost; User ID=postgres; Password=postgres; Port=5432; Database=Dekanat");

public Form1()

{

InitializeComponent();

NpgsqlConnection conn = new NpgsqlConnection("Server=localhost; User ID=postgres; Password=postgres; Port=5432; Database=Dekanat");

conn.Open();

NpgsqlCommand comm = new NpgsqlCommand();

comm.Connection = conn;

comm.CommandType = CommandType.Text;

comm.CommandText = "select \* from myTimetable4";

NpgsqlDataReader dr = comm.ExecuteReader();

if(dr.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dr);

dataGridView1.DataSource = dt;

/\*dataGridView1.Columns["Id"].ReadOnly = true;\*/

}

comm.Dispose();

conn.Close();

/\*conn.Open();

NpgsqlCommand com = new NpgsqlCommand();

com.Connection = conn;

com.CommandType = CommandType.Text;

com.CommandText = "select \* from Groups";

NpgsqlDataReader dy = com.ExecuteReader();

if (dy.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dy);

dataGridView2.DataSource = dt;

dataGridView2.Columns["Id"].ReadOnly = true;

}

com.Dispose();

conn.Close();\*/

/\*

\* fsdgfsjdgfsjgdfjss

\*/

conn.Open();

NpgsqlCommand cos = new NpgsqlCommand();

cos.Connection = conn;

cos.CommandType = CommandType.Text;

cos.CommandText = "select \* from myTeacher1";

NpgsqlDataReader de = cos.ExecuteReader();

if (de.HasRows)

{

DataTable dt = new DataTable();

dt.Load(de);

dataGridView3.DataSource = dt;

/\*dataGridView3.Columns["Id"].ReadOnly = true;\*/

}

cos.Dispose();

conn.Close();

conn.Open();

NpgsqlCommand coe = new NpgsqlCommand();

coe.Connection = conn;

coe.CommandType = CommandType.Text;

coe.CommandText = "select \* from Students";

NpgsqlDataReader dw = coe.ExecuteReader();

if (dw.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dw);

dataGridView4.DataSource = dt;

dataGridView4.Columns["Id"].ReadOnly = true;

}

coe.Dispose();

conn.Close();

/\*

conn.Open();

NpgsqlCommand cow = new NpgsqlCommand();

cow.Connection = conn;

cow.CommandType = CommandType.Text;

cow.CommandText = "select \* from Faculties";

NpgsqlDataReader dq = cow.ExecuteReader();

if (dq.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dq);

dataGridView5.DataSource = dt;

dataGridView5.Columns["Id"].ReadOnly = true;

}

cow.Dispose();

conn.Close();

conn.Open();

NpgsqlCommand coq = new NpgsqlCommand();

coq.Connection = conn;

coq.CommandType = CommandType.Text;

coq.CommandText = "select \* from Specialist";

NpgsqlDataReader da = coq.ExecuteReader();

if (da.HasRows)

{

DataTable dt = new DataTable();

dt.Load(da);

dataGridView6.DataSource = dt;

dataGridView6.Columns["Id"].ReadOnly = true;

}

coq.Dispose();

conn.Close();

conn.Open();

NpgsqlCommand coa = new NpgsqlCommand();

coa.Connection = conn;

coa.CommandType = CommandType.Text;

coa.CommandText = "select \* from Discipline";

NpgsqlDataReader ds = coa.ExecuteReader();

if (ds.HasRows)

{

DataTable dt = new DataTable();

dt.Load(ds);

dataGridView7.DataSource = dt;

dataGridView7.Columns["Id"].ReadOnly = true;

}

coa.Dispose();

conn.Close();\*/

/\*conn.Open();

String sCommand = "select \* from Groups";

using (NpgsqlCommand cmd = new NpgsqlCommand(sCommand, conn))

{

cmd.CommandType = CommandType.Text;

DataTable table = new DataTable();

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter(cmd);

adapter.Fill(table);

comboBox1.DisplayMember = "ID";

comboBox1.ValueMember = "Name";

comboBox1.DataSource = table;

}

conn.Close();\*/

conn.Open();

String sCommand1 = "select \* from Teacher";

using (NpgsqlCommand cmd = new NpgsqlCommand(sCommand1, conn))

{

cmd.CommandType = CommandType.Text;

DataTable table = new DataTable();

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter(cmd);

adapter.Fill(table);

comboBox4.DisplayMember = "ID";

comboBox4.ValueMember = "FIO";

comboBox4.DataSource = table;

}

conn.Close();

conn.Open();

String sCommand2 = "select \* from Discipline";

using (NpgsqlCommand cmd = new NpgsqlCommand(sCommand2, conn))

{

cmd.CommandType = CommandType.Text;

DataTable table = new DataTable();

NpgsqlDataAdapter adapter = new NpgsqlDataAdapter(cmd);

adapter.Fill(table);

comboBox5.DisplayMember = "ID";

comboBox5.ValueMember = "name";

comboBox5.DataSource = table;

}

conn.Close();

comboBox2.Items.Add("Практика");

comboBox2.Items.Add("Лекция");

comboBox3.Items.Add("Понедельник");

comboBox3.Items.Add("Вторник");

comboBox3.Items.Add("Среда");

comboBox3.Items.Add("Четверг");

comboBox3.Items.Add("Пятница");

comboBox3.Items.Add("Суббота");

comboBox3.Items.Add("Воскресенье");

comboBox7.Items.Add("ПБ-201");

comboBox7.Items.Add("ПБ-202с");

comboBox7.Items.Add("бГРАД-183");

comboBox7.Items.Add("ПБ-205");

comboBox7.Items.Add("ПБ-201с");

comboBox7.Items.Add("бГРАД-182");

}

public class Group

{

public string name { get; set; }

public int ID { get; set; }

public override string ToString()

{

return string.Format("name", name);

}

}

public class Teacher

{

public string fio { get; set; }

public int ID { get; set; }

public override string ToString()

{

return string.Format("FIO", fio);

}

}

public class Discipline

{

public string name { get; set; }

public int ID { get; set; }

public override string ToString()

{

return string.Format("FIO", name);

}

}

public class Specialist

{

public string position { get; set; }

public int ID { get; set; }

public override string ToString()

{

return string.Format("Position", position);

}

}

private void Form1\_Load(object sender, EventArgs e)

{

//Расписание

conn.Open();

NpgsqlCommand Command2 = new NpgsqlCommand();

Command2.Connection = conn;

Command2.CommandText = $"Select ID, name from Groups";

List<Group> group = new List<Group>();

{

NpgsqlDataReader dr = Command2.ExecuteReader();

while (dr.Read())

{

Group gr = new Group

{

ID = int.Parse(dr["ID"].ToString()),

name = dr["name"].ToString()

};

group.Add(gr);

}

}

comboBox1.DisplayMember = "name";

comboBox1.ValueMember = "ID";

comboBox1.DataSource = group;

conn.Close();

conn.Open();

NpgsqlCommand Command3 = new NpgsqlCommand();

Command3.Connection = conn;

Command3.CommandText = $"Select ID, fio from Teacher";

List<Teacher> teacher = new List<Teacher>();

{

NpgsqlDataReader de = Command3.ExecuteReader();

while (de.Read())

{

Teacher th = new Teacher

{

ID = int.Parse(de["ID"].ToString()),

fio = de["FIO"].ToString()

};

teacher.Add(th);

}

}

comboBox4.DisplayMember = "fio";

comboBox4.ValueMember = "ID";

comboBox4.DataSource = teacher;

conn.Close();

conn.Open();

NpgsqlCommand Command4 = new NpgsqlCommand();

Command4.Connection = conn;

Command4.CommandText = $"Select ID, name from Discipline";

List<Discipline> discipline = new List<Discipline>();

{

NpgsqlDataReader dp = Command4.ExecuteReader();

while (dp.Read())

{

Discipline di = new Discipline

{

ID = int.Parse(dp["ID"].ToString()),

name = dp["name"].ToString()

};

discipline.Add(di);

}

}

comboBox5.DisplayMember = "name";

comboBox5.ValueMember = "ID";

comboBox5.DataSource = discipline;

conn.Close();

//Преподаватели

conn.Open();

NpgsqlCommand Command5 = new NpgsqlCommand();

Command5.Connection = conn;

Command5.CommandText = $"Select ID, Position from Specialist";

List<Specialist> position = new List<Specialist>();

{

NpgsqlDataReader po = Command5.ExecuteReader();

while (po.Read())

{

Specialist dq = new Specialist

{

ID = int.Parse(po["ID"].ToString()),

position = po["Position"].ToString()

};

position.Add(dq);

}

}

comboBox6.DisplayMember = "position";

comboBox6.ValueMember = "ID";

comboBox6.DataSource = position;

conn.Close();

}

private void toolTip1\_Popup(object sender, PopupEventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (textBox9.Text.Trim() == "" || textBox4.Text.Trim() == "" || textBox7.Text.Trim() == "" || textBox8.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

try

{

conn.Open();

var com = new NpgsqlCommand(

$"Insert Into Timetable (ID, ID\_Groups, ID\_Teacher, ID\_Discipline, NAME, Kind\_Of\_Couples, Day\_Of\_Week, Steam\_Time, Audience\_Number) Values({int.Parse(textBox9.Text)}, {comboBox1.SelectedValue}, {comboBox4.SelectedValue}, {comboBox5.SelectedValue}, N'{textBox4.Text}', N'{comboBox2.Text}', N'{comboBox3.Text}', '{DateTime.Parse(textBox7.Text).ToString("yyyy-MM-dd HH:mm:ss")}', {int.Parse(textBox8.Text)})", conn);

com.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация добавлена");

textBox9.Clear();

textBox4.Clear();

textBox7.Clear();

textBox8.Clear();

button3\_Click(null, null);

}

catch (Exception ex) { }

/\*$"Insert Into Timetable (ID, ID\_Groups, ID\_Teacher, ID\_Discipline, NAME, Kind\_Of\_Couples, Day\_Of\_Week, Steam\_Time, Audience\_Number) Values({int.Parse(textBox9.Text)}, {int.Parse(comboBox1.Text)}, {int.Parse(textBox2.Text)}, {int.Parse(textBox3.Text)}, N'{textBox4.Text}', N'{comboBox2.Text}', N'{comboBox3.Text}', '{DateTime.Parse(textBox7.Text).ToString("yyyy-MM-dd HH:mm:ss")}', {int.Parse(textBox8.Text)})", conn);

com.ExecuteNonQuery();\*/

/\* (Группа, Преподаватель, Дисциплина, Тема, Занятие, День, Время, Аудитория) Values(N'{comboBox1.Text}', N'{textBox2.Text}', N'{textBox3.Text}', N'{textBox4.Text}', N'{comboBox2.Text}', N'{comboBox3.Text}', '{DateTime.Parse(textBox7.Text).ToString("yyyy-MM-dd HH:mm:ss")}', { int.Parse(textBox8.Text)})", conn);

com.ExecuteNonQuery();\*/

}

private void LoadData()

{

}

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void comboBox3\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void comboBox1\_SelectedValueChanged(object sender, EventArgs e)

{

/\*comboBox1.SelectedValue.ToString();\*/

}

private void button1\_Click\_1(object sender, EventArgs e)

{

try

{

conn.Open();

int ID = int.Parse(textBox9.Text);

var Groups\_ID = comboBox1.SelectedValue;

var Teacher\_ID = comboBox4.SelectedValue;

var Discipline\_ID = comboBox5.SelectedValue;

string name = textBox4.Text;

string kind = comboBox2.Text;

string day = comboBox3.Text;

string time = DateTime.Parse(textBox7.Text).ToString("yyyy-MM-dd HH:mm:ss");

int aud = int.Parse(textBox8.Text);

string Query = "UPDATE timetable SET ID\_Groups = '" + Groups\_ID + "', ID\_Teacher = '" + Teacher\_ID + "', ID\_Discipline = '" + Discipline\_ID + "', NAME = '" + name + "', Kind\_Of\_Couples = '" + kind + "', Day\_Of\_Week = '" + day + "', Steam\_Time = '" + time + "', Audience\_Number = '" + aud + "' WHERE ID = '" + ID + "'";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация обновлена");

textBox9.Clear();

textBox4.Clear();

textBox7.Clear();

textBox8.Clear();

}

catch { }

}

private void button2\_Click(object sender, EventArgs e)

{

if (textBox9.Text.Trim() == "" )

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

conn.Open();

int ID = int.Parse(textBox9.Text);

string Query = "DELETE from Timetable where ID= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация удалена");

textBox9.Clear();

button3\_Click(null, null);

}

private void toolStripButton1\_Click(object sender, EventArgs e)

{

}

private void button3\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

string Query = "select \* from myTimetable4";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

NpgsqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dr);

dataGridView1.DataSource = dt;

}

dataGridView1.Refresh();

conn.Close();

}

catch { }

}

private void button4\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

var searchData = textBox5.Text;

string Query = "Select \* from myTimetable4 where Группа like '%" + searchData + "%' or Преподаватель like '%" + searchData + "%' or Дисциплина like '%" + searchData + "%' or Тема like '%" + searchData + "%' or Занятие like '%" + searchData + "%' or День like '%" + searchData + "%' or Аудитория like '%" + searchData + "%'";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

NpgsqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dr);

dataGridView1.DataSource = dt;

}

dataGridView1.Refresh();

conn.Close();

}

catch { }

}

private void textBox5\_TextChanged(object sender, EventArgs e)

{

}

private void button5\_Click(object sender, EventArgs e)

{

if (textBox9.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

try

{

conn.Open();

int ID = int.Parse(textBox9.Text);

string Query = "select \* from myTimetable4 where №= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

var reader = cmd.ExecuteReader();

if (reader.Read())

{

comboBox1.Text = reader["Группа"].ToString();

comboBox4.Text = reader["Преподаватель"].ToString();

comboBox5.Text = reader["Дисциплина"].ToString();

textBox4.Text = reader["Тема"].ToString();

comboBox2.Text = reader["Занятие"].ToString();

comboBox3.Text = reader["День"].ToString();

textBox7.Text = reader["Время"].ToString();

textBox8.Text = reader["Аудитория"].ToString();

}

conn.Close();

}

catch { }

}

private void button6\_Click(object sender, EventArgs e)

{

conn.Open();

string name = comboBox1.Text;

string Query = "select id from Groups where name = '%" + name + "%'";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

var reader = cmd.ExecuteReader();

if (reader.Read())

{

comboBox1.Text = reader["ID"].ToString();

}

}

private void comboBox4\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void comboBox5\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void label16\_Click(object sender, EventArgs e)

{

}

//Преподаватели

private void button6\_Click\_1(object sender, EventArgs e)

{

try

{

if (textBox1.Text.Trim() == "" || textBox2.Text.Trim() == "" || textBox3.Text.Trim() == "" || textBox6.Text.Trim() == "" || textBox10.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

conn.Open();

var com = new NpgsqlCommand(

$"Insert Into Teacher (ID, ID\_Specialist, work\_experience, fio, salary, personal\_number) Values({int.Parse(textBox1.Text)}, {comboBox6.SelectedValue}, N'{textBox2.Text}', N'{textBox3.Text}', '{textBox6.Text}', {textBox10.Text})", conn);

com.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация добавлена");

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox6.Clear();

textBox10.Clear();

button9\_Click(null, null);

}

catch (Exception ex) { }

}

private void button9\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

string Query = "select \* from myTeacher1";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

NpgsqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dr);

dataGridView3.DataSource = dt;

}

dataGridView3.Refresh();

conn.Close();

}

catch { }

}

private void comboBox6\_SelectedIndexChanged(object sender, EventArgs e)

{

}

private void button7\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

int ID = int.Parse(textBox1.Text);

var id\_specialist = comboBox6.SelectedValue;

string work = textBox2.Text;

string fio = textBox3.Text;

string salary = textBox6.Text;

string pers = textBox10.Text;

string Query = "UPDATE teacher SET ID\_Specialist = '" + id\_specialist + "', work\_experience = '" + work + "', fio = '" + fio + "', salary = '" + salary + "', personal\_number = '" + pers + "' WHERE ID = '" + ID + "'";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация обновлена");

textBox1.Clear();

textBox2.Clear();

textBox3.Clear();

textBox6.Clear();

textBox10.Clear();

}

catch { }

}

private void button8\_Click(object sender, EventArgs e)

{

if (textBox1.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

try

{

conn.Open();

int ID = int.Parse(textBox1.Text);

string Query = "DELETE from Teacher where ID= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация удалена");

textBox1.Clear();

button9\_Click(null, null);

}

catch { }

}

// Студенты

private void button11\_Click(object sender, EventArgs e)

{

try

{

if (textBox11.Text.Trim() == "" || textBox12.Text.Trim() == "" || textBox13.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

conn.Open();

var com = new NpgsqlCommand(

$"Insert Into Students (ID, fio, phone\_number, groupss, date\_of\_birth) Values({int.Parse(textBox11.Text)}, N'{textBox12.Text}', N'{textBox13.Text}', N'{comboBox7.Text}', '{DateTime.Parse(dateTimePicker1.Text).ToString("yyyy-MM-dd HH:mm:ss")}')", conn);

com.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация добавлена");

textBox11.Clear();

textBox12.Clear();

textBox13.Clear();

textBox6.Clear();

button10\_Click(null, null);

}

catch { }

}

private void button10\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

string Query = "select \* from Students";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

NpgsqlDataReader dr = cmd.ExecuteReader();

if (dr.HasRows)

{

DataTable dt = new DataTable();

dt.Load(dr);

dataGridView4.DataSource = dt;

}

dataGridView4.Refresh();

conn.Close();

}

catch { }

}

private void button12\_Click(object sender, EventArgs e)

{

try

{

conn.Open();

int ID = int.Parse(textBox11.Text);

string fio = textBox12.Text;

var group = comboBox7.Text;

string numb = textBox13.Text;

string date = DateTime.Parse(dateTimePicker1.Text).ToString("yyyy-MM-dd HH:mm:ss");

string Query = "UPDATE Students SET fio = '" + fio + "', phone\_number = '" + numb + "', groupss = '" + group + "', date\_of\_birth = '" + date + "' WHERE ID = '" + ID + "'";

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация обновлена");

textBox11.Clear();

textBox12.Clear();

textBox13.Clear();

}

catch { }

}

private void button13\_Click(object sender, EventArgs e)

{

if (textBox11.Text.Trim() == "" )

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

try

{

conn.Open();

int ID = int.Parse(textBox11.Text);

string Query = "DELETE from students where ID= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

cmd.ExecuteNonQuery();

conn.Close();

MessageBox.Show("Информация удалена");

textBox1.Clear();

button10\_Click(null, null);

}

catch { }

}

//Преподаватели (Автозаполнение)

private void button14\_Click(object sender, EventArgs e)

{

if (textBox1.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

try

{

conn.Open();

int ID = int.Parse(textBox1.Text);

string Query = "select \* from myTeacher1 where №= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

var reader = cmd.ExecuteReader();

if (reader.Read())

{

comboBox6.Text = reader["Специальность"].ToString();

textBox2.Text = reader["Опыт\_работы"].ToString();

textBox3.Text = reader["ФИО"].ToString();

textBox6.Text = reader["Зарплата"].ToString();

textBox10.Text = reader["Персональный\_номер"].ToString();

}

conn.Close();

}

catch { }

}

private void textBox9\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar)) return;

else

e.Handled = true;

}

private void textBox8\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar)) return;

else

e.Handled = true;

}

private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar)) return;

else

e.Handled = true;

}

private void textBox10\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar)) return;

else

e.Handled = true;

}

private void textBox11\_KeyPress(object sender, KeyPressEventArgs e)

{

if (Char.IsDigit(e.KeyChar)) return;

else

e.Handled = true;

}

//Студенты (Автозаполнение)

private void button15\_Click(object sender, EventArgs e)

{

if (textBox11.Text.Trim() == "")

{

MessageBox.Show("Вы не ввели все необходимые данные!!", "Внимание", MessageBoxButtons.OK, MessageBoxIcon.Exclamation, MessageBoxDefaultButton.Button1);

return;

}

else

{

}

try

{

conn.Open();

int ID = int.Parse(textBox11.Text);

string Query = "select \* from Students where ID= " + ID;

NpgsqlCommand cmd = new NpgsqlCommand(Query, conn);

var reader = cmd.ExecuteReader();

if (reader.Read())

{

comboBox7.Text = reader["groupss"].ToString();

textBox12.Text = reader["fio"].ToString();

textBox13.Text = reader["phone\_number"].ToString();

dateTimePicker1.Text = reader["date\_of\_birth"].ToString();

}

conn.Close();

}

catch { }

}

}

}