Министерство цифрового развития, связи и массовых коммуникаций

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра «Информатика»

Лабораторная работа №1

по дисциплине

«Проектирование клиентских приложений»

Выполнил:

студент гр.БЭИ2102

Борисов А. М.

Вариант № 4

Принял:

Доцент к.т.н

Халабия Р. Ф.

Москва 2024

**1. Постановка задачи**

Написать программу согласно варианту, которая добавляет, удаляет и изменяет записи данных, хранящиеся в файле формата XML. Если файла нет, то необходимо его создавать, если все записи в файле удалены, то файл должен оставаться пустым. Также мы должны выводить информацию из файла на экран в табличном виде. Для редактирования данных необходимо небольшое простое меню.

**2. Разработка схемы алгоритмов**

Разработанная схема программы представлена на рисунках 1-4.

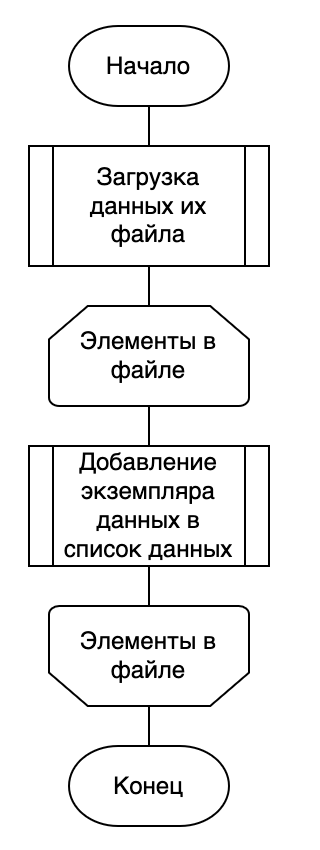


Рисунок 1 – Алгоритм подгрузки данных

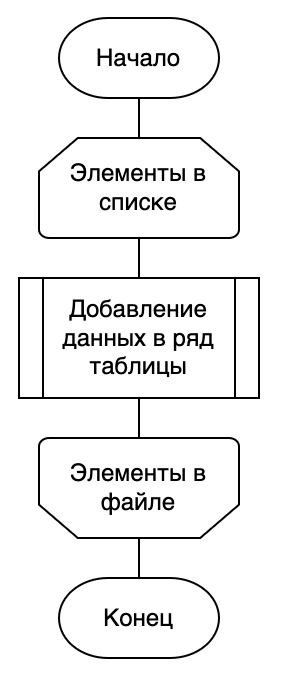


Рисунок 2 – Алгоритм инициализации данных

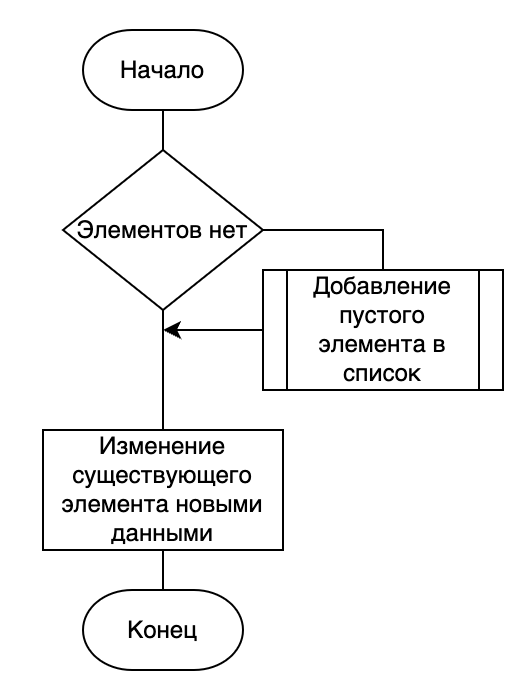


Рисунок 3 – Алгоритм изменения данных

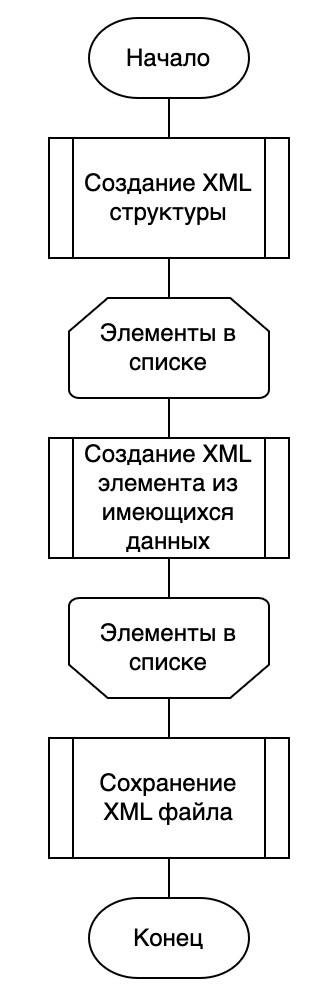


Рисунок 4 – Алгоритм сохранения данных

**3. Разработка программы**

**3.1 Разработка графического интерфейса пользователя**

Разработанный интерфейс программы, представлен на рисунке 5.

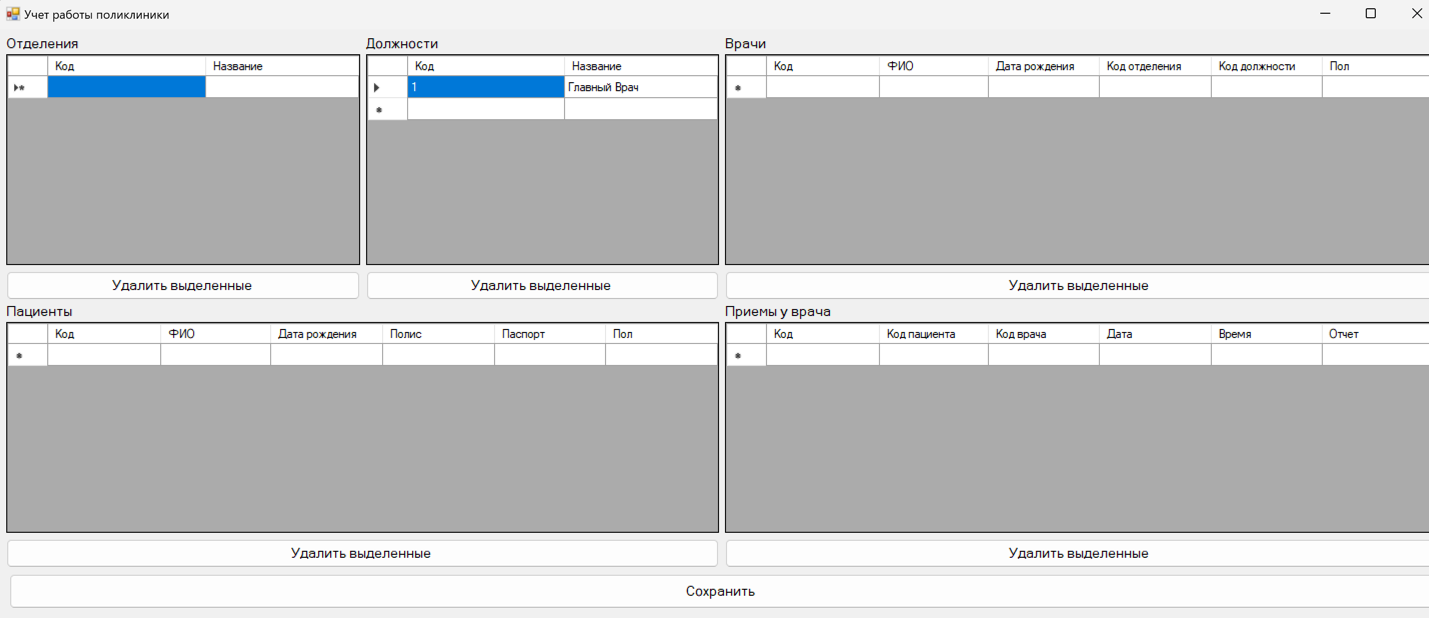


Рисунок 5 – Интерфейс программы

**3.2 Программный код**

Код класса работы с главным файлом:

using System.IO;

using System.Xml;

namespace HLP\_RKP\_LR1.Models

{

internal class XMLEditor

{

private static string file = "document.xml";

public static XmlDocument doc = new XmlDocument();

public static void CreateBaseDocStructure()

{

doc = new XmlDocument();

XmlDeclaration xmlDeclaration = doc.CreateXmlDeclaration("1.0", "UTF-8", null);

XmlElement root = doc.DocumentElement;

doc.InsertBefore(xmlDeclaration, root);

XmlElement element = doc.CreateElement(string.Empty, "body", string.Empty);

doc.AppendChild(element);

}

public static XmlElement LoadSection(string sectionName)

{

if (!File.Exists(file))

{

CreateBaseDocStructure();

doc.Save(file);

}

doc.Load(file);

XmlElement xRoot = doc.DocumentElement;

foreach (XmlElement el in xRoot)

{

if (el.Name == sectionName)

{

return el;

}

}

return doc.CreateElement(string.Empty, sectionName, string.Empty);

}

public static void Save()

{

CreateBaseDocStructure();

XmlElement appointments = Appointment.Save();

XmlElement departments = Department.Save();

XmlElement doctors = Doctor.Save();

XmlElement patients = Patient.Save();

XmlElement positions = Position.Save();

doc.DocumentElement.AppendChild(appointments);

doc.DocumentElement.AppendChild(departments);

doc.DocumentElement.AppendChild(doctors);

doc.DocumentElement.AppendChild(patients);

doc.DocumentElement.AppendChild(positions);

doc.Save(file);

}

}

}

Код одного из подобных друг другу классов, который отвечает за работу с данными в пределах одной таблицы:

using System;

using System.Collections.Generic;

using System.Windows.Forms;

using System.Xml;

namespace HLP\_RKP\_LR1.Models

{

internal class Position

{

private static List<Position> items;

private static readonly string fileName = "positions.xml";

private static readonly string xmlItemName = "position";

public string ID { get; set; }

public string Name { get; set; }

public Position(string id, string name)

{

ID = id;

Name = name;

}

public static void Init(DataGridView dgv)

{

int rowsCounter = 0;

foreach (Position item in items)

{

dgv.Rows.Add();

dgv[0, rowsCounter].Value = item.ID;

dgv[1, rowsCounter].Value = item.Name;

rowsCounter++;

}

}

public static void Load()

{

XmlElement xRoot = XMLEditor.LoadSection(xmlItemName + "s");

items = new List<Position>();

foreach (XmlElement element in xRoot)

{

string id = "";

string name = "";

foreach (XmlElement childElement in element.ChildNodes)

{

if (childElement.Name == "id")

{

id = Convert.ToString(childElement.InnerText);

}

if (childElement.Name == "name")

{

name = Convert.ToString(childElement.InnerText);

}

}

items.Add(new Position(id, name));

}

}

public static void Edit(DataGridView dgv, DataGridViewCellEventArgs e)

{

if (items.Count - 1 < e.RowIndex)

{

items.Add(new Position("", ""));

}

if (e.ColumnIndex == 0)

{

items[e.RowIndex].ID = dgv[e.ColumnIndex, e.RowIndex].Value.ToString();

}

if (e.ColumnIndex == 1)

{

items[e.RowIndex].Name = dgv[e.ColumnIndex, e.RowIndex].Value.ToString();

}

}

public static XmlElement Save()

{

XmlElement positions = XMLEditor.doc.CreateElement(string.Empty, xmlItemName + "s", string.Empty);

foreach (Position item in items)

{

XmlElement positionElement = XMLEditor.doc.CreateElement(string.Empty, xmlItemName, string.Empty);

XmlElement idElement = XMLEditor.doc.CreateElement(string.Empty, "id", string.Empty);

idElement.InnerText = item.ID;

XmlElement nameElement = XMLEditor.doc.CreateElement(string.Empty, "name", string.Empty);

nameElement.InnerText = item.Name;

positionElement.AppendChild(idElement);

positionElement.AppendChild(nameElement);

positions.AppendChild(positionElement);

}

return positions;

}

public static void DeleteRows(DataGridView dgv)

{

foreach (DataGridViewRow row in dgv.SelectedRows)

{

if (items.Count - 1 >= row.Index)

{

items.Remove(items[row.Index]);

dgv.Rows.RemoveAt(row.Index);

}

}

}

}

}

**4. Результат работы программы**

Результаты работы программы представлен на рисунках 6-9.

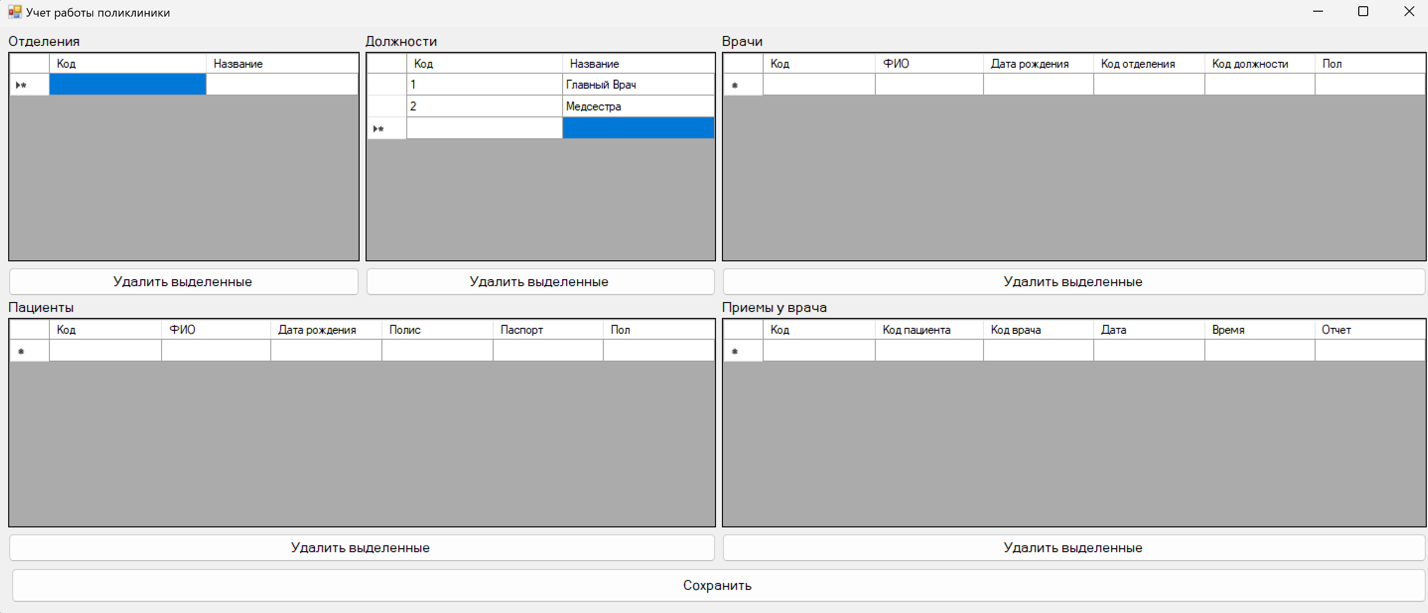


Рисунок 6 – Результат добавления записи

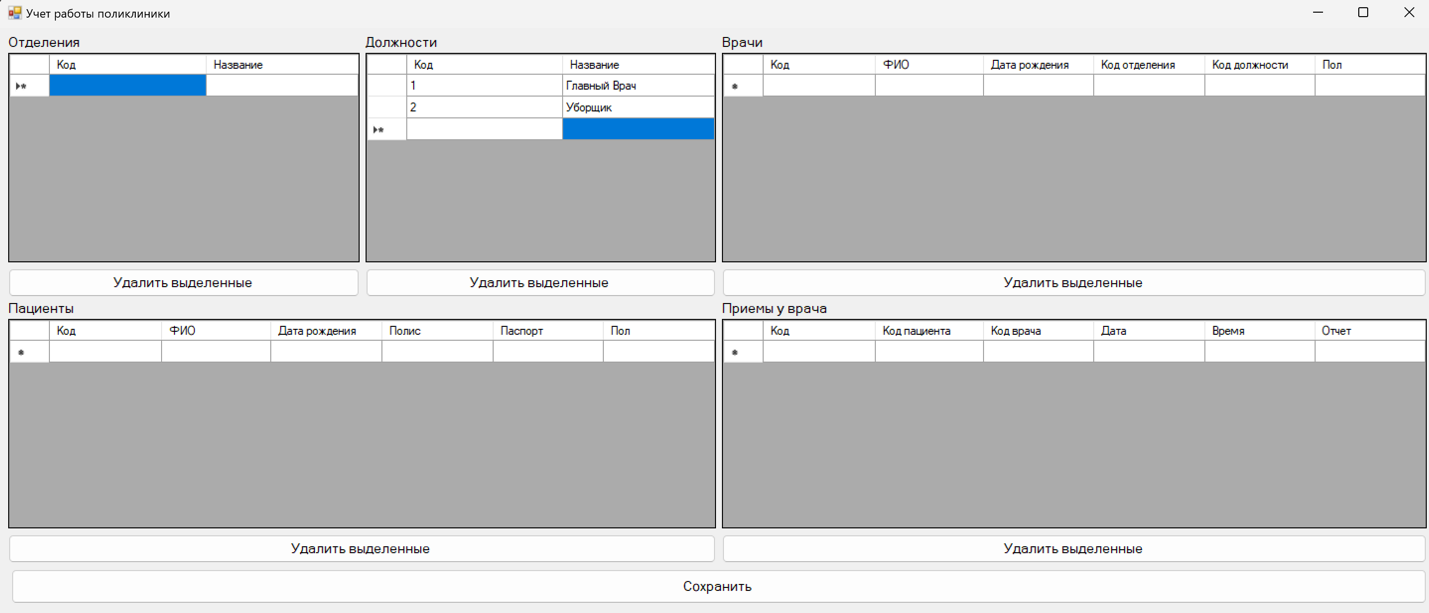


Рисунок 7 – Результат редактирования данных

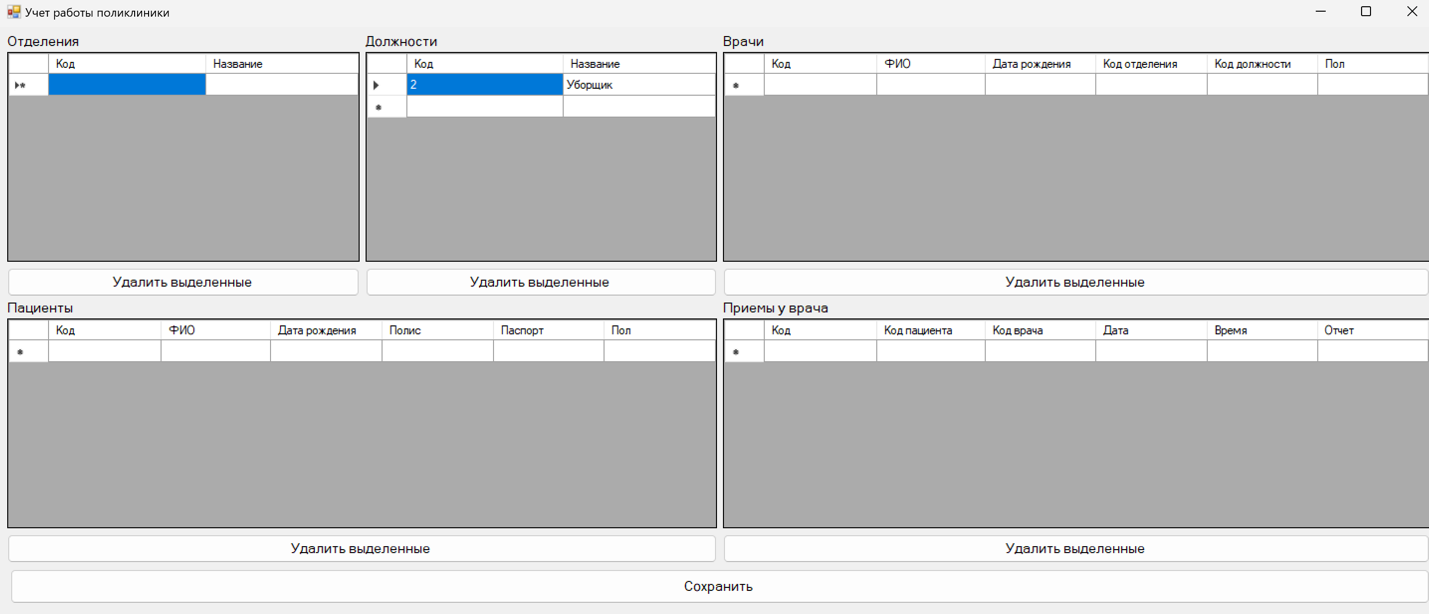


Рисунок 8 – Результат удаления записи

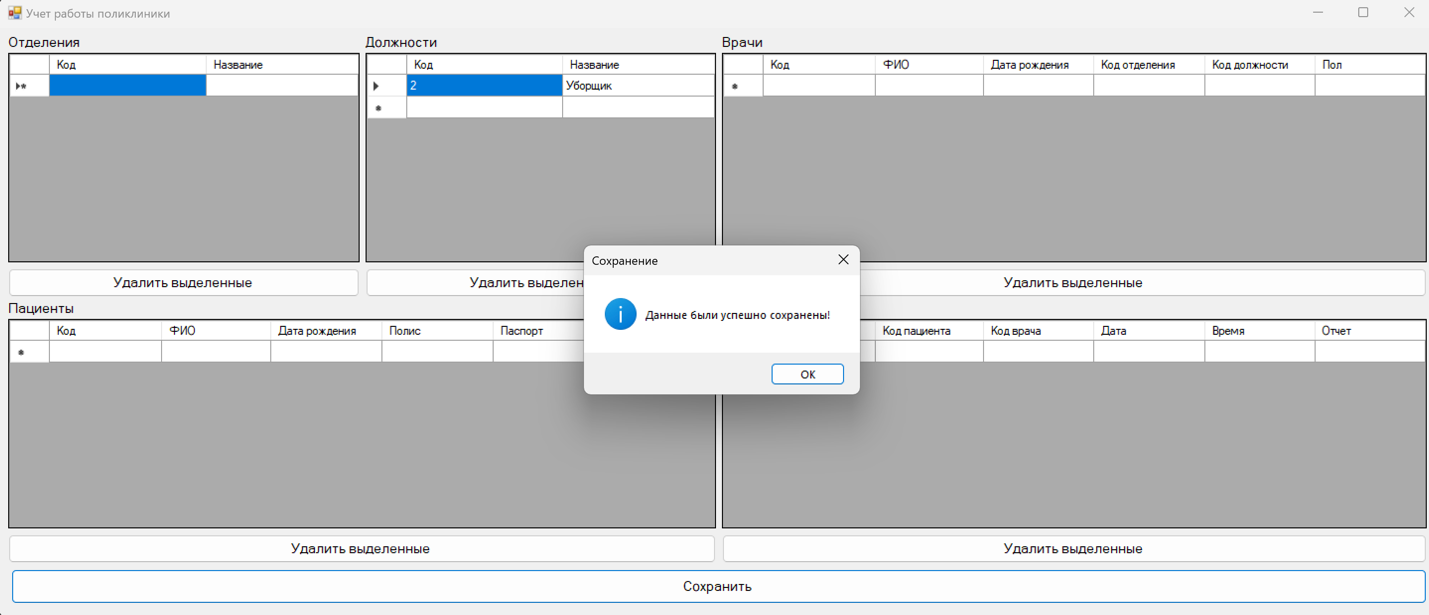


Рисунок 9 – Результат сохранения файла