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 №3

по дисциплине

«Проектирование клиентских приложений»

Вариант № 4

Выполнил:

студент гр.БЭИ2102

Борисов А. М.

Проверил:

Доцент к.т.н

Халабия Р. Ф.

Москва 2024

**1. Постановка задачи**

Доработать программу из ЛР№2, добавив функционал добавления, изменения и удаления столбцов таблиц.

**2. Разработка схемы алгоритмов**

Разработанная схема программы представлена на рисунках 1-6.

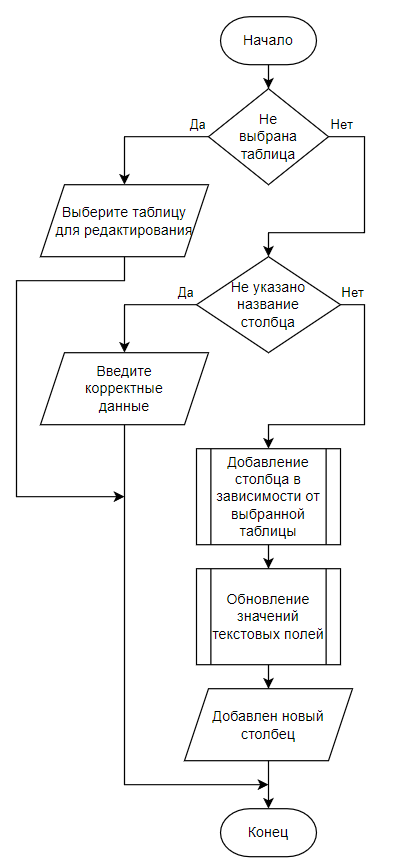


Рисунок 1 – Алгоритм нажатия на кнопку добавления столбца

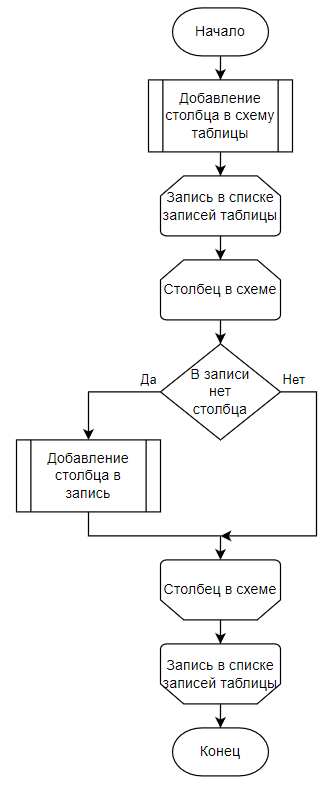


Рисунок 2 – Алгоритм добавления столбца

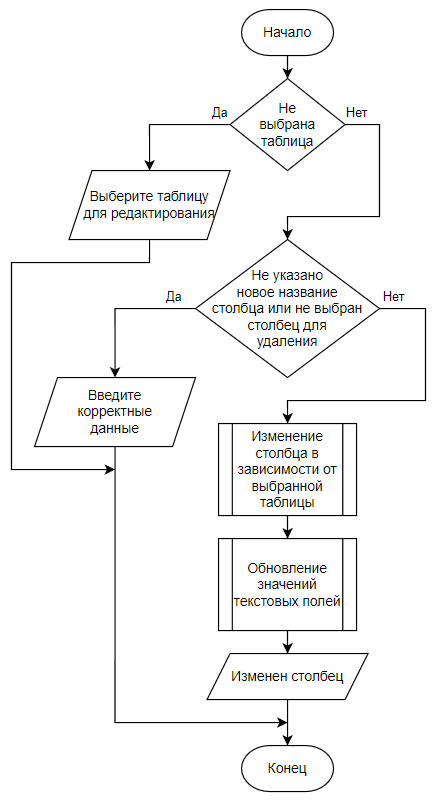


Рисунок 3 – Алгоритм нажатия на кнопку изменения столбца

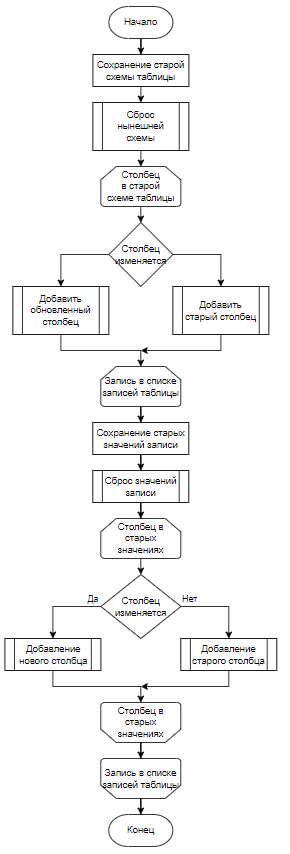


Рисунок 4 – Алгоритм изменения столбца

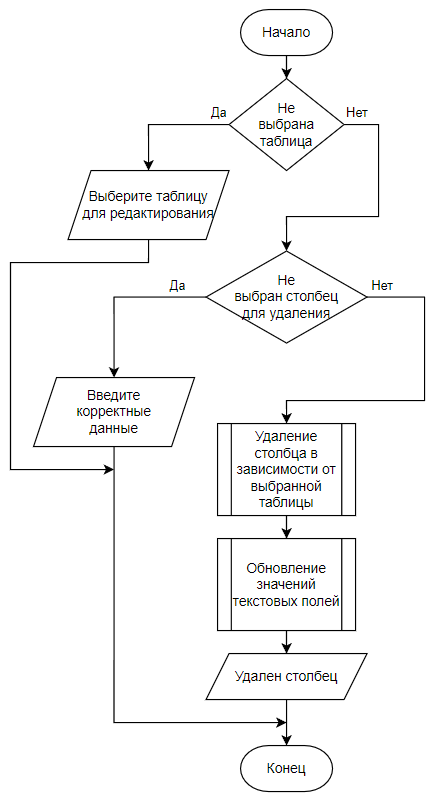


Рисунок 5 – Алгоритм нажатия на кнопку удаления столбца

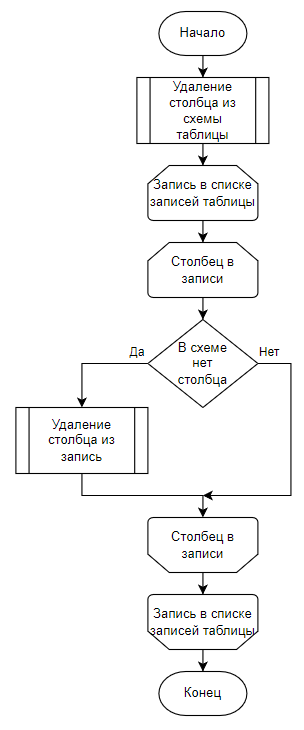


Рисунок 6 – Алгоритм удаления столбца

**3. Разработка программы**

**3.1 Разработка графического интерфейса пользователя**

Разработанный интерфейс программы, представлен на рисунках 7-8.

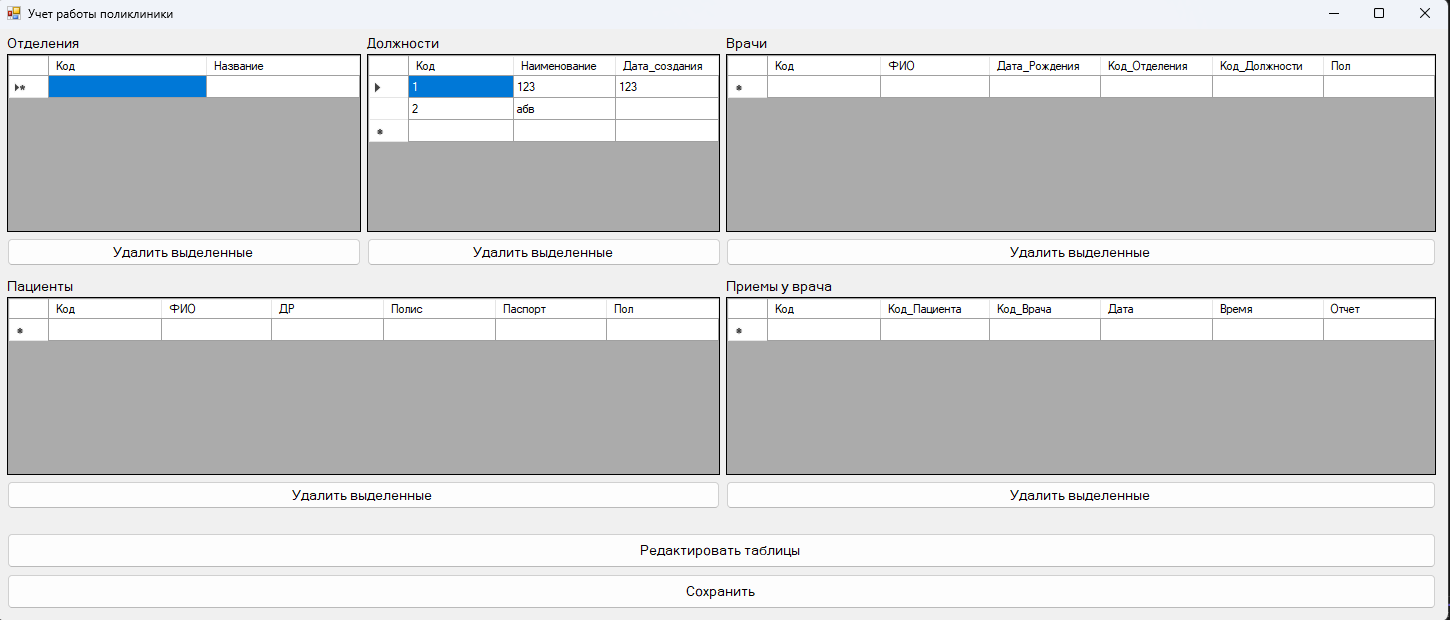


Рисунок 7 – Интерфейс программы

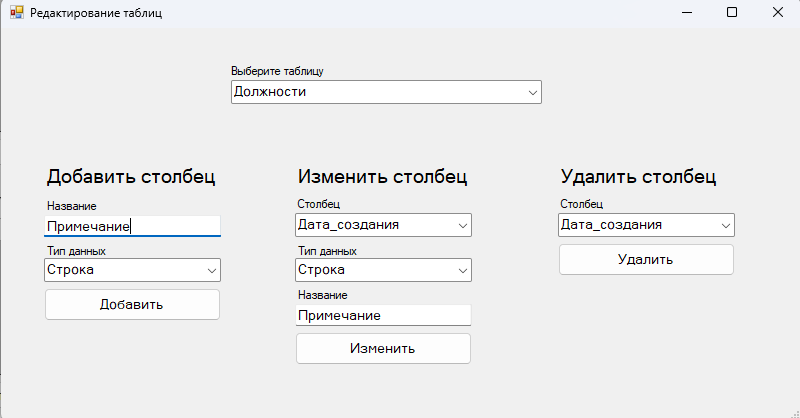


Рисунок 8 – Интерфейс формы редактирования

**3.2 Программный код**

Код методов по работе со столбцами:

public static void UpdateTableOnAdd(string name, ItemTypes type, List<TableItem> items, Dictionary<string, ItemTypes> schema)

{

schema.Add(name, type);

foreach (TableItem item in items)

{

foreach (KeyValuePair<string, ItemTypes> pair in schema)

{

if (!item.Values.ContainsKey(pair.Key))

{

item.Values.Add(pair.Key, new TypedItem("", pair.Value));

break;

}

}

}

}

public static void UpdateTableOnDelete(string name, List<TableItem> items, Dictionary<string, ItemTypes> schema)

{

schema.Remove(name);

foreach (TableItem item in items)

{

foreach (KeyValuePair<string, TypedItem> pair in item.Values)

{

if (!schema.ContainsKey(pair.Key))

{

item.Values.Remove(pair.Key);

break;

}

}

}

}

public static void UpdateTableOnEdit(string oldColName, string newColName, ItemTypes newType, List<TableItem> items, Dictionary<string, ItemTypes> schema)

{

Dictionary<string, ItemTypes> oldSchema = schema.ToDictionary(entry => entry.Key, entry => entry.Value);

schema.Clear();

foreach (KeyValuePair<string, ItemTypes> pair in oldSchema)

{

if (pair.Key == oldColName)

{

schema.Add(newColName, newType);

}

else

{

schema.Add(pair.Key, pair.Value);

}

}

foreach (TableItem item in items)

{

Dictionary<string, TypedItem> oldValues = item.Values.ToDictionary(entry => entry.Key, entry => entry.Value);

item.Values.Clear();

foreach (KeyValuePair<string, TypedItem> pair in oldValues)

{

if (pair.Key == oldColName)

{

TypedItem newItem = pair.Value;

ItemTypes schemaType = schema[newColName];

newItem.Type = schemaType;

newItem.Validate(false);

item.Values.Add(newColName, newItem);

}

else

{

item.Values.Add(pair.Key, pair.Value);

}

}

}

}

Код методов-обработчиков нажатия на кнопки работы со столбцами:

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

{

string tableName = tableNameCBox.Text;

if (tableName == string.Empty)

{

MessageBox.Show("Выберите таблицу для редактирования", "Ошибка редактирования", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string formattedName = Formatters.FormatColumnName(addColTBox.Text);

if (formattedName == string.Empty)

{

MessageBox.Show("Введите корректные данные", "Ошибка добавления", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

ItemTypes type = TypedItem.HumanToTypeDict[addColCBox.Text];

switch (tableName)

{

case departmentsTableName:

TablesActions.AddColumn(formattedName, type, DGVDepartments, Department.items, Department.schema);

break;

case positionsTableName:

TablesActions.AddColumn(formattedName, type, DGVPositions, Position.items, Position.schema);

break;

case doctorsTableName:

TablesActions.AddColumn(formattedName, type, DGVDoctors, Doctor.items, Doctor.schema);

break;

case patientsTableName:

TablesActions.AddColumn(formattedName, type, DGVPatients, Patient.items, Patient.schema);

break;

case appointmentsTableName:

TablesActions.AddColumn(formattedName, type, DGVAppointments, Appointment.items, Appointment.schema);

break;

}

UpdateCBoxesValues();

MessageBox.Show("Добавлен новый столбец", "Добавление столбца", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

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

{

string tableName = tableNameCBox.Text;

if (tableName == string.Empty)

{

MessageBox.Show("Выберите таблицу для редактирования", "Ошибка редактирования", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string colName = deleteColCBox.Text;

if (colName == string.Empty)

{

MessageBox.Show("Выберите столбец", "Ошибка удаления", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

switch (tableName)

{

case departmentsTableName:

TablesActions.DeleteColumn(colName, DGVDepartments, Department.items, Department.schema);

break;

case positionsTableName:

TablesActions.DeleteColumn(colName, DGVPositions, Position.items, Position.schema);

break;

case doctorsTableName:

TablesActions.DeleteColumn(colName, DGVDoctors, Doctor.items, Doctor.schema);

break;

case patientsTableName:

TablesActions.DeleteColumn(colName, DGVPatients, Patient.items, Patient.schema);

break;

case appointmentsTableName:

TablesActions.DeleteColumn(colName, DGVAppointments, Appointment.items, Appointment.schema);

break;

}

UpdateCBoxesValues();

MessageBox.Show("Удален столбец", "Удаление столбца", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

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

{

string tableName = tableNameCBox.Text;

if (tableName == string.Empty)

{

MessageBox.Show("Выберите таблицу для редактирования", "Ошибка редактирования", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

string oldColName = editColCBox.Text;

string newColName = Formatters.FormatColumnName(editColTBox.Text);

if (oldColName == string.Empty || newColName == string.Empty)

{

MessageBox.Show("Введите корректные данные", "Ошибка изменения", MessageBoxButtons.OK, MessageBoxIcon.Error);

return;

}

ItemTypes newType = TypedItem.HumanToTypeDict[editColTypeCBox.Text];

switch (tableName)

{

case departmentsTableName:

TablesActions.EditColumn(oldColName, newColName, newType, DGVDepartments, Department.items, Department.schema);

break;

case positionsTableName:

TablesActions.EditColumn(oldColName, newColName, newType, DGVPositions, Position.items, Position.schema);

break;

case doctorsTableName:

TablesActions.EditColumn(oldColName, newColName, newType, DGVDoctors, Doctor.items, Doctor.schema);

break;

case patientsTableName:

TablesActions.EditColumn(oldColName, newColName, newType, DGVPatients, Patient.items, Patient.schema);

break;

case appointmentsTableName:

TablesActions.EditColumn(oldColName, newColName, newType, DGVAppointments, Appointment.items, Appointment.schema);

break;

}

UpdateCBoxesValues();

MessageBox.Show("Изменен столбец", "Изменение столбца", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

**4. Результат работы программы**

Результаты работы программы представлен на рисунках 9-11.

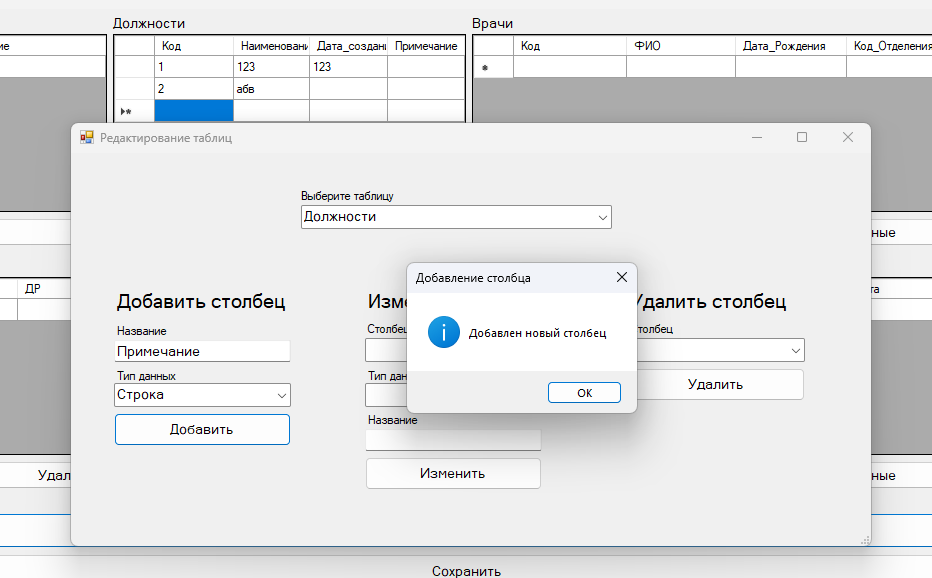


Рисунок 9 – Результат добавления столбца

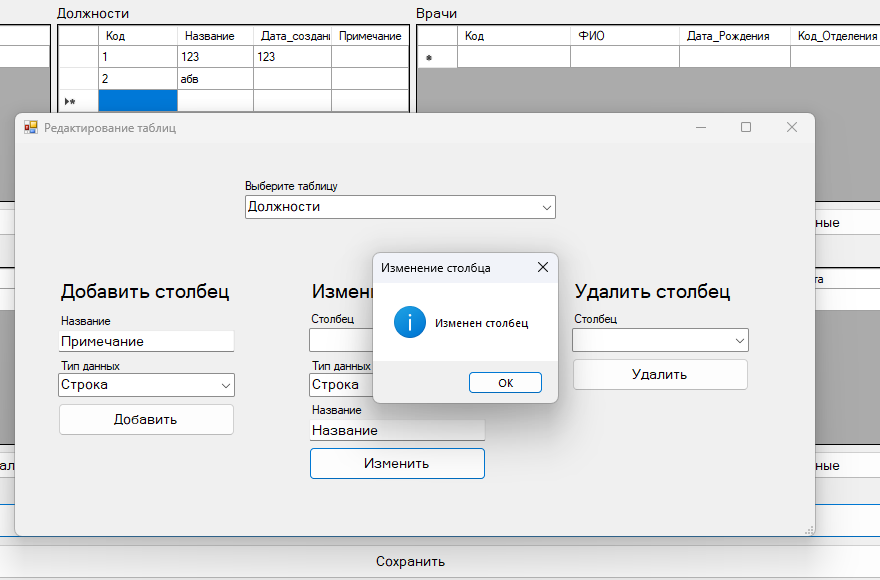


Рисунок 10 – Результат изменения столбца

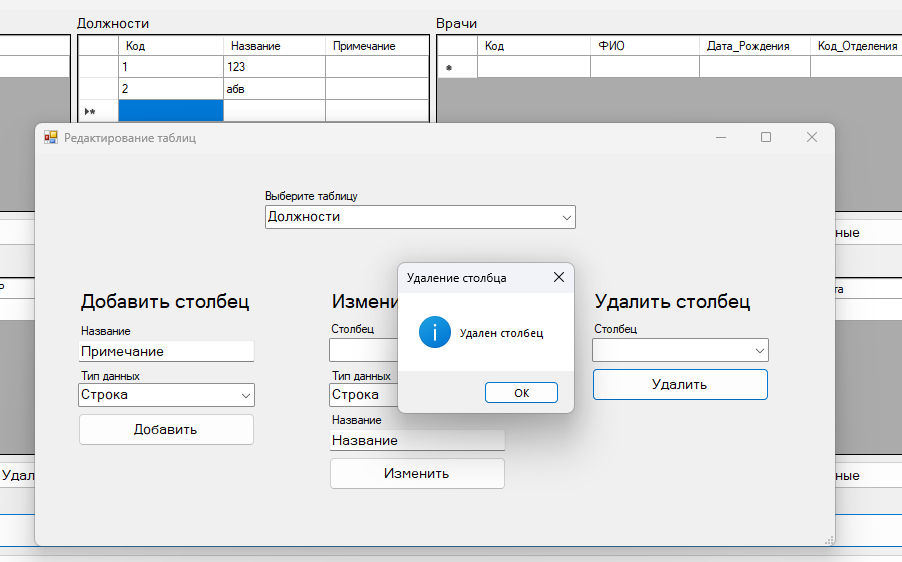


Рисунок 11 – Результат удаления столбца