**Нижегородской области**

**Государственное бюджетное профессиональное образовательное учреждение**

**«Нижегородский Губернский колледж»**

Допущен к защите:

преподаватель

\_\_\_\_\_\_\_\_\_\_\_ В.А. Авакян,

\_\_\_\_\_\_\_\_\_\_\_ Д.А. Дергачев,

\_\_\_\_\_

«21» марта 2022 г.

**ОТЧЕТ**

**ПО УЧЕБНОЙ ПРАКТИКЕ**

Студент: Пугин А. А.

Специальность, группа: 09.02.07, 44П

Нижний Новгород

2022 г.

Диаграмма базы данных



Рисунок 1. Диаграмма БД

Код форм представлен в Приложении А

Подключение проекта к базе данных SQL



Рисунок 2. Заполнение таблицы «Пользователи»



Рисунок 3. Заполнение таблицы «Услуги»

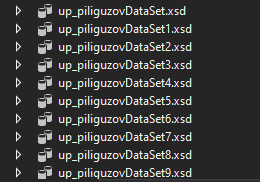


Рисунок 4. Подключенные таблицы

Форма 1 приложения

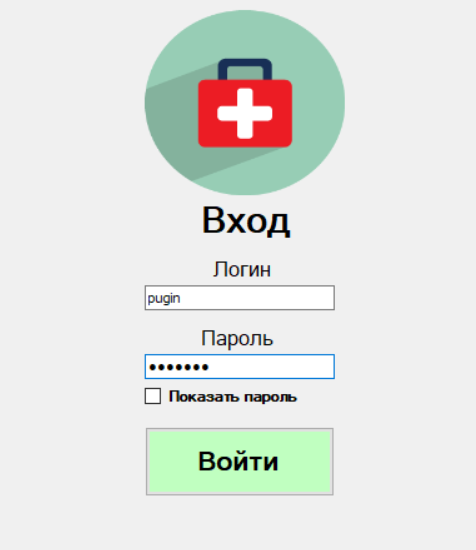


Рисунок 5. Форма авторизации

Форма 2 Главное меню

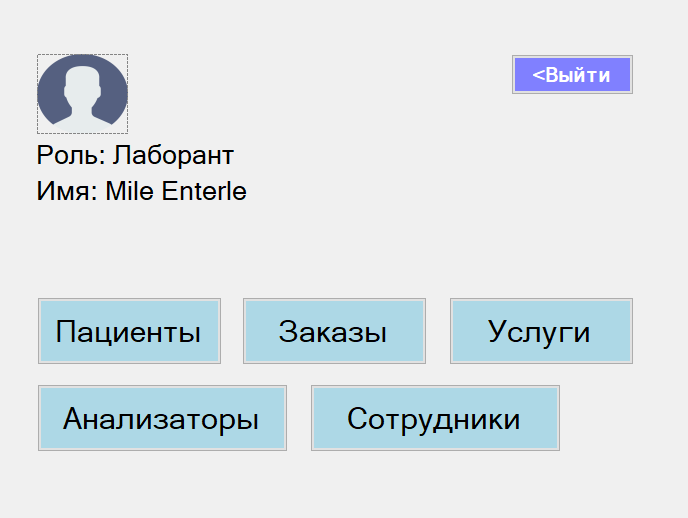


Рисунок 6. Главное меню

Форма 3 Анализатор

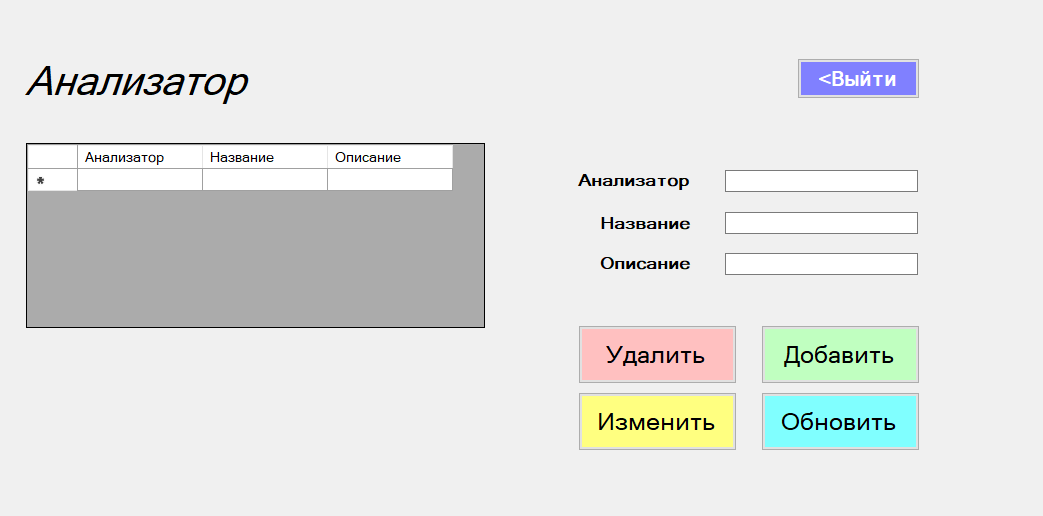


Рисунок 7. Анализатор

Форма 4 Услуги

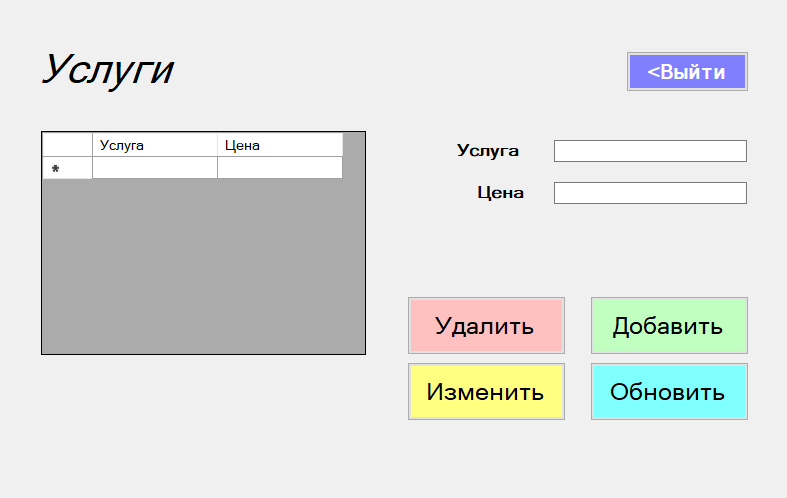


Рисунок 8. Таблица услуг

Форма 5 Заказ

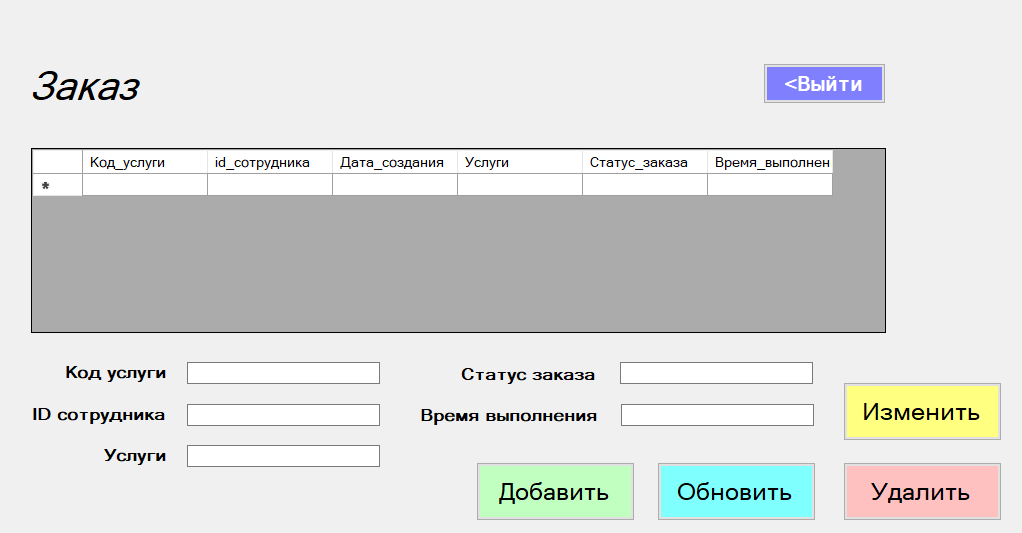


Рисунок 9. Таблица заказов

Форма 6 Сотрудники

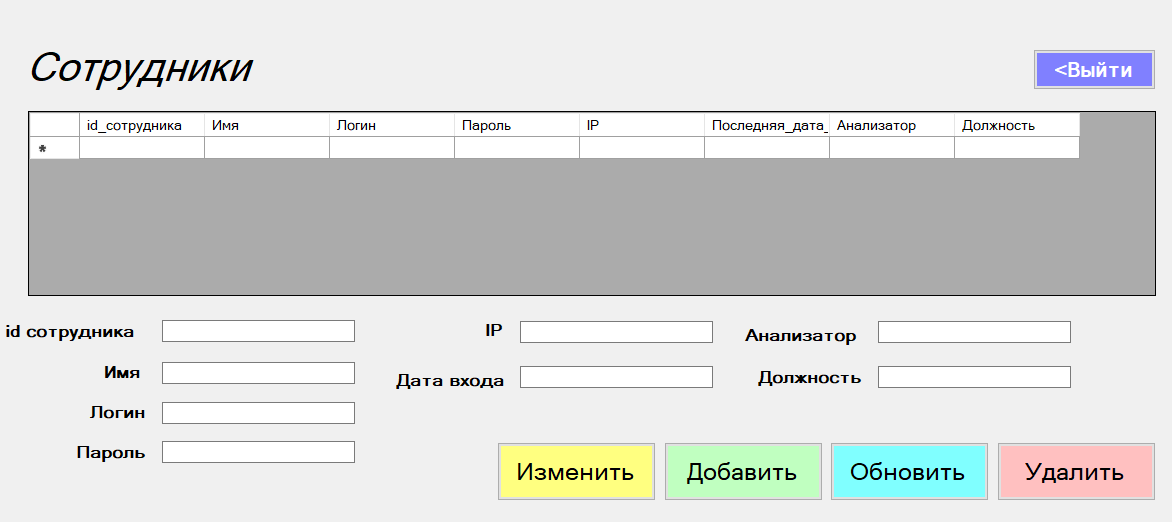


Рисунок 10. Таблица сотрудников

Форма 7 Пациенты

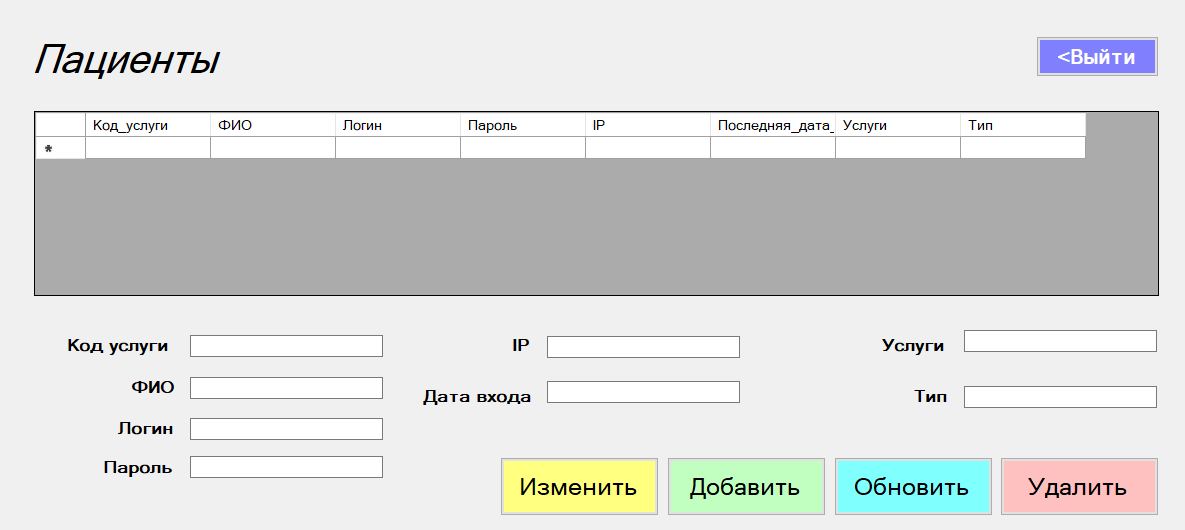


Рисунок 11. Таблица пациентов

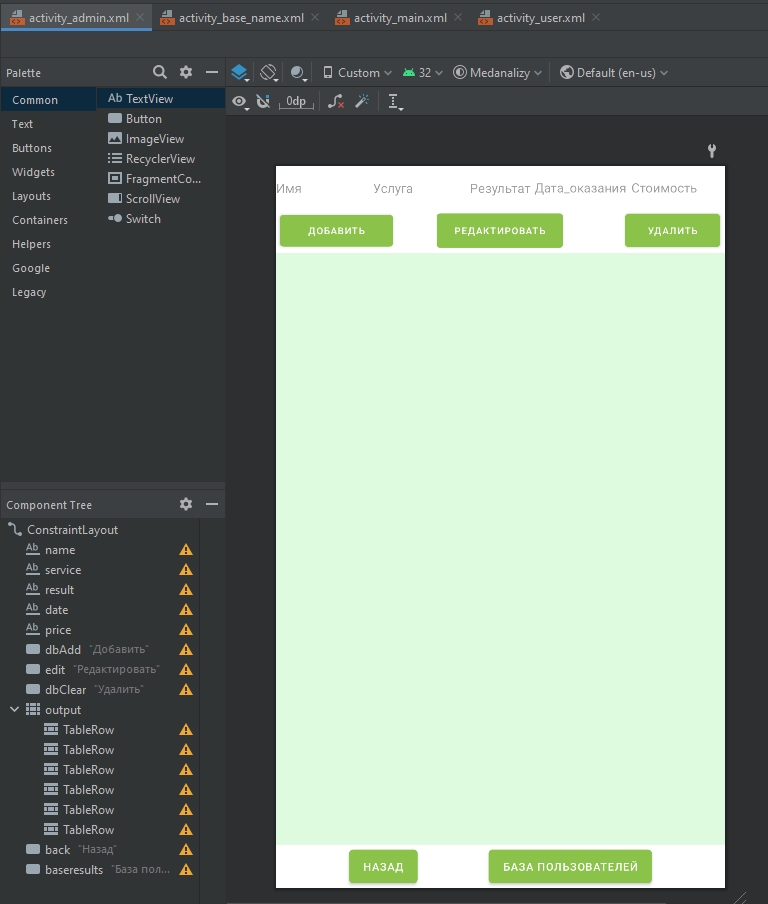


Рисунок 12. Activity admin

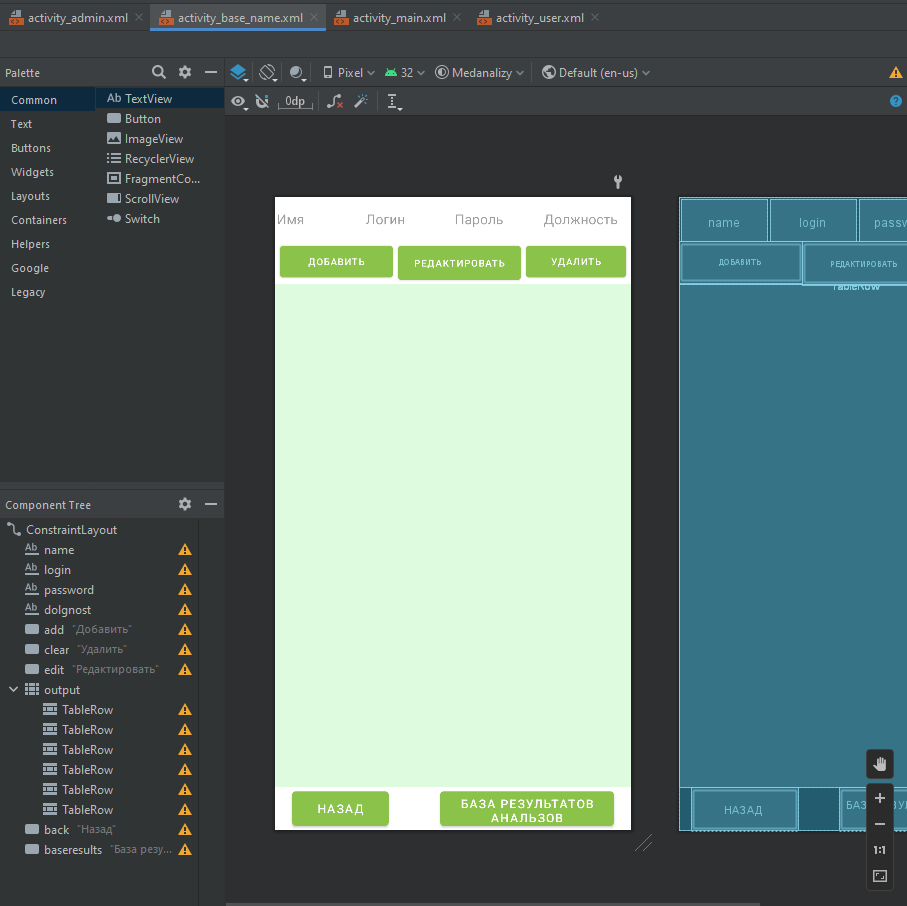


Рисунок 13. Activity base name

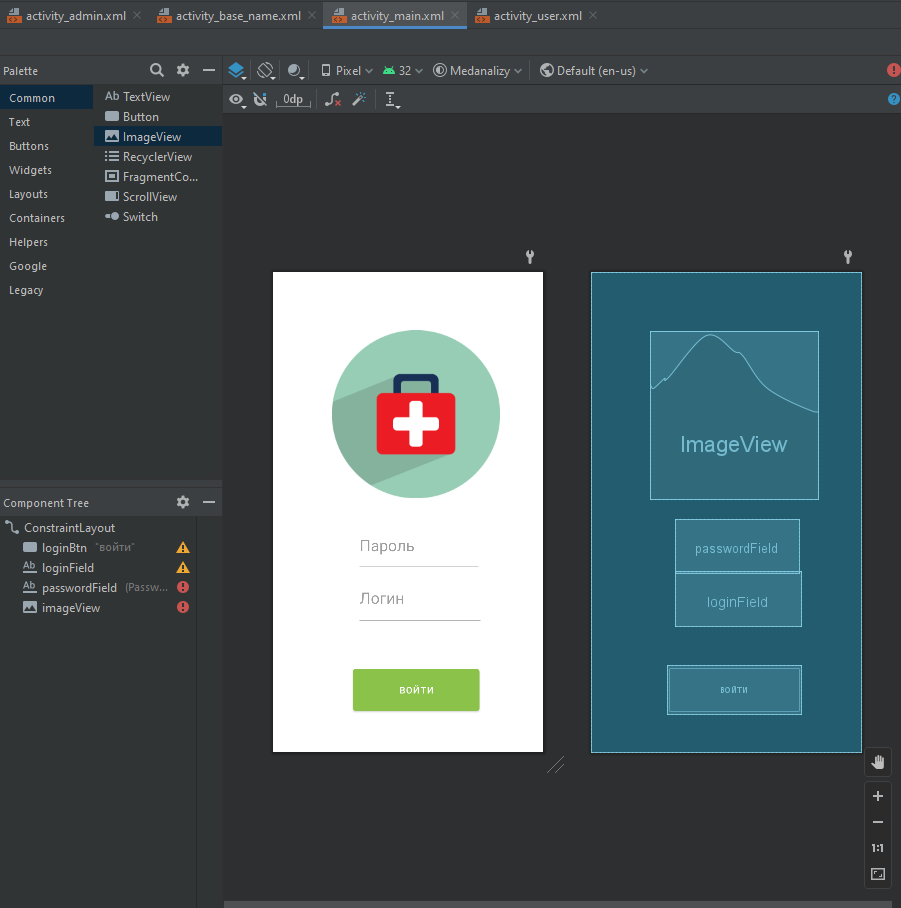


Рисунок 14. Activity main

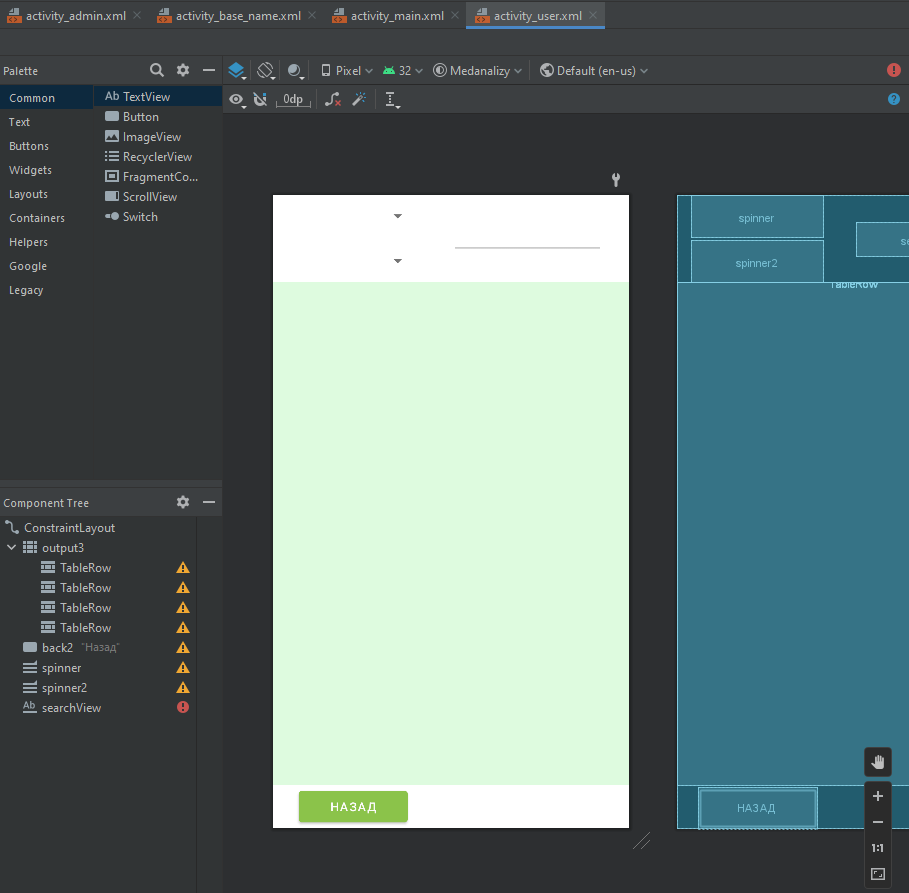


Рисунок 15. Activity user

ПРИЛОЖЕНИЕ А

namespace UP\_interface

{

public partial class Form3 : Form

{

private int r;

string ConnStr = @"Data Source=sql;Initial Catalog = 'up\_pugin\_med'; Integrated Security = True";

public object MyClass { get; private set; }

public Form3()

{

InitializeComponent();

}

private void FillАнализатор()

{

SqlDataAdapter da = new SqlDataAdapter();

DataSet ds = new DataSet();

da.Fill(ds, "[Анализатор]");

dataGridView1.DataSource = ds.Tables["[Анализатор]"].DefaultView;

}

private void label4\_Click(object sender, EventArgs e)

{

}

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

{

string Анализатор = textBox1.Text.ToString();

string Название = textBox2.Text.ToString();

string Описание = textBox3.Text.ToString();

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med' ;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "INSERT INTO Анализатор VALUES ('" + Анализатор + "','" + Название + "', '" + Описание + "')";

SqlCommand dbCommand = new SqlCommand(query, dbConnection);

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные добавлены!", "Внимание!");

dbConnection.Close();

}

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

{

//Проверяем количество выбранных строк

if (dataGridView1.SelectedRows.Count != 1)

{

MessageBox.Show("Выберите одну строку для удаления!", "Сообщение!");

return;

}

// Запомним выбранную строку

int index = dataGridView1.SelectedRows[0].Index;

//Проверим данные в таблице

if (dataGridView1.Rows[index].Cells[0].Value == null)

{

MessageBox.Show("Не все данные введены!", "Сообщение!");

return;

}

//Считаем данные

string id = dataGridView1.Rows[index].Cells[0].Value.ToString();

SqlConnection con = new SqlConnection(ConnStr);

// Выполняем запрос к базе данных

con.Open();//открываем соединение

string SqlText = "DELETE FROM [Анализатор] WHERE Анализатор=" + id;//строка запроса

SqlCommand dbCommand = new SqlCommand(SqlText, con);//команда

//Выполняем запрос

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные удалены !", "Сообщение!");

// Удаляем данные из таблицы в форме

dataGridView1.Rows.RemoveAt(index);

//Закрываем соединение с БД

con.Close();

}

private void Form3\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet.Анализатор". При необходимости она может быть перемещена или удалена.

this.анализаторTableAdapter1.Fill(this.up\_piliguzovDataSet.Анализатор);

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Анализатор]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Анализатор]");

dataGridView1.DataSource = ds.Tables["[Анализатор]"].DefaultView;

dbConnection.Close();

}

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

{

Form2 f = new Form2();

f.Show();

}

public void MyExecuteNonQuery(string SqlText)

{

SqlConnection cn;

SqlCommand cmd;

cn = new SqlConnection(ConnStr);

cn.Open();

cmd = cn.CreateCommand();

cmd.CommandText = SqlText;

cmd.ExecuteNonQuery();

cn.Close();

}

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

{

{

if (r == 0)

{

int index, n;

string Анализатор, Название, Описание;

n = dataGridView1.Rows.Count;

if (n == 1) return;

index = dataGridView1.CurrentRow.Index;

Анализатор = dataGridView1[0, index].Value.ToString();

Название = dataGridView1[1, index].Value.ToString();

Описание = dataGridView1[2, index].Value.ToString();

textBox1.Text = Анализатор;

textBox2.Text = Название;

textBox3.Text = Описание;

r = 1;

}

else if (r == 1)

{

string SqlText = "update [Анализатор] set Анализатор = " + textBox1.Text + ", Название = \'" + textBox2.Text + "\', Описание = " + textBox3.Text + " where Анализатор = " + textBox1.Text;

MyExecuteNonQuery(SqlText);

FillАнализатор();

textBox1.Text = "";

textBox2.Text = "";

textBox3.Text = "";

r = 0;

}

}

//else if (MyClass.Должность == "lab")

//MessageBox.Show("У вас недостаточно прав для редактирования таблицы.", "Внимание!");

}

}

}

namespace UP\_interface

{

public partial class Form4 : Form

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

public Form4()

{

InitializeComponent();

}

private void Form4\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet1.Оказанная\_услуга". При необходимости она может быть перемещена или удалена.

this.оказанная\_услугаTableAdapter2.Fill(this. up\_pugin\_medDataSet1.Оказанная\_услуга);

}

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

{

}

private void label2\_Click(object sender, EventArgs e)

{

}

private void label4\_Click(object sender, EventArgs e)

{

}

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

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

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

{

//Проверяем количество выбранных строк

if (dataGridView1.SelectedRows.Count != 1)

{

MessageBox.Show("Выберите одну строку для удаления!", "Сообщение!");

return;

}

// Запомним выбранную строку

int index = dataGridView1.SelectedRows[0].Index;

//Проверим данные в таблице

if (dataGridView1.Rows[index].Cells[0].Value == null)

{

MessageBox.Show("Не все данные введены!", "Сообщение!");

return;

}

//Считаем данные

string id = dataGridView1.Rows[index].Cells[0].Value.ToString();

SqlConnection con = new SqlConnection(ConnStr);

// Выполняем запрос к базе данных

con.Open();//открываем соединение

string SqlText = "DELETE FROM [Оказанная\_услуга] WHERE Услуга=" + id;//строка запроса

SqlCommand dbCommand = new SqlCommand(SqlText, con);//команда

//Выполняем запрос

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные удалены !", "Сообщение!");

// Удаляем данные из таблицы в форме

dataGridView1.Rows.RemoveAt(index);

//Закрываем соединение с БД

con.Close();

}

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

{

string Услуга = textBox1.Text.ToString();

string Цена = textBox2.Text.ToString();

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "INSERT INTO Оказанная\_услуга VALUES ('" + Услуга + "','" + Цена + "')";

SqlCommand dbCommand = new SqlCommand(query, dbConnection);

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные добавлены!", "Внимание!");

dbConnection.Close();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Оказанная\_услуга]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Оказанная\_услуга]");

dataGridView1.DataSource = ds.Tables["[Оказанная\_услуга]"].DefaultView;

dbConnection.Close();

}

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

{

Form2 f = new Form2();

f.Show();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Оказанная\_услуга]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Оказанная\_услуга]");

dataGridView1.DataSource = ds.Tables["[Оказанная\_услуга]"].DefaultView;

dbConnection.Close();

}

}

}

public partial class Form5 : Form

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

public Form5()

{

InitializeComponent();

}

private void Form5\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet5.Заказ". При необходимости она может быть перемещена или удалена.

this.заказTableAdapter4.Fill(this.up\_piliguzovDataSet5.Заказ);

}

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

{

Form2 f = new Form2();

f.Show();

}

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

{

string ID\_сотрудника = textBox2.Text.ToString();

string Услуги = textBox3.Text.ToString();

string Статус\_заказа = textBox5.Text.ToString();

string Время\_выполнения = textBox4.Text.ToString();

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "INSERT INTO [Заказ] VALUES ('" + "','" + ID\_сотрудника + "', '" + Услуги + "', '" + Статус\_заказа + "', '" + Время\_выполнения + "')";

SqlCommand dbCommand = new SqlCommand(query, dbConnection);

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные добавлены!", "Внимание!");

dbConnection.Close();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Заказ]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Заказ]");

dataGridView1.DataSource = ds.Tables["[Заказ]"].DefaultView;

dbConnection.Close();

}

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

{

//Проверяем количество выбранных строк

if (dataGridView1.SelectedRows.Count != 1)

{

MessageBox.Show("Выберите одну строку для удаления!", "Сообщение!");

return;

}

// Запомним выбранную строку

int index = dataGridView1.SelectedRows[0].Index;

//Проверим данные в таблице

if (dataGridView1.Rows[index].Cells[0].Value == null)

{

MessageBox.Show("Не все данные введены!", "Сообщение!");

return;

}

//Считаем данные

string id = dataGridView1.Rows[index].Cells[0].Value.ToString();

SqlConnection con = new SqlConnection(ConnStr);

// Выполняем запрос к базе данных

con.Open();//открываем соединение

string SqlText = "DELETE FROM [Заказ] WHERE id\_сотрудника=" + id;//строка запроса

SqlCommand dbCommand = new SqlCommand(SqlText, con);//команда

//Выполняем запрос

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные удалены !", "Сообщение!");

// Удаляем данные из таблицы в форме

dataGridView1.Rows.RemoveAt(index);

//Закрываем соединение с БД

con.Close();

}

}

}

public partial class Form7 : Form

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

public Form7()

{

InitializeComponent();

}

private void Form7\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet9.Сотрудники". При необходимости она может быть перемещена или удалена.

this.сотрудникиTableAdapter2.Fill(this.up\_piliguzovDataSet9.Сотрудники);

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet6.Сотрудники". При необходимости она может быть перемещена или удалена.

this.сотрудникиTableAdapter1.Fill(this.up\_piliguzovDataSet6.Сотрудники);

}

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

{

string id\_сотрудника = textBox1.Text.ToString();

string Имя = textBox2.Text.ToString();

string Логин = textBox3.Text.ToString();

string Пароль = textBox5.Text.ToString();

string IP = textBox6.Text.ToString();

string Последняя\_дата\_входа = textBox4.Text.ToString();

string Анализатор = textBox8.Text.ToString();

string Должность = textBox7.Text.ToString();

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "INSERT INTO Сотрудники VALUES ('" + id\_сотрудника + "','" + Имя + "', '" + Логин + "','" + Пароль + "','" + IP + "','" + Последняя\_дата\_входа + "','" + Анализатор + "','" + Должность + "')";

SqlCommand dbCommand = new SqlCommand(query, dbConnection);

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные добавлены!", "Внимание!");

dbConnection.Close();

}

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

{

//Проверяем количество выбранных строк

if (dataGridView1.SelectedRows.Count != 1)

{

MessageBox.Show("Выберите одну строку для удаления!", "Сообщение!");

return;

}

// Запомним выбранную строку

int index = dataGridView1.SelectedRows[0].Index;

//Проверим данные в таблице

if (dataGridView1.Rows[index].Cells[0].Value == null)

{

MessageBox.Show("Не все данные введены!", "Сообщение!");

return;

}

//Считаем данные

string id = dataGridView1.Rows[index].Cells[0].Value.ToString();

SqlConnection con = new SqlConnection(ConnStr);

// Выполняем запрос к базе данных

con.Open();//открываем соединение

string SqlText = "DELETE FROM [Сотрудники] WHERE id\_сотрудника=" + id;//строка запроса

SqlCommand dbCommand = new SqlCommand(SqlText, con);//команда

//Выполняем запрос

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные удалены !", "Сообщение!");

// Удаляем данные из таблицы в форме

dataGridView1.Rows.RemoveAt(index);

//Закрываем соединение с БД

con.Close();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Сотрудники]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Сотрудники]");

dataGridView1.DataSource = ds.Tables["[Сотрудники]"].DefaultView;

dbConnection.Close();

}

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

{

Form2 f = new Form2();

f.Show();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med';Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Сотрудники]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Сотрудники]");

dataGridView1.DataSource = ds.Tables["[Сотрудники]"].DefaultView;

dbConnection.Close();

}

}

}

namespace UP\_interface

{

public partial class Form8 : Form

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

public Form8()

{

InitializeComponent();

}

private void Form8\_Load(object sender, EventArgs e)

{

// TODO: данная строка кода позволяет загрузить данные в таблицу "up\_piliguzovDataSet7.Пользователи". При необходимости она может быть перемещена или удалена.

this.пользователиTableAdapter2.Fill(this.up\_piliguzovDataSet7.Пользователи);

}

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

{

}

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

{

string Код\_услуги = textBox1.Text.ToString();

string ФИО = textBox2.Text.ToString();

string Логин = textBox3.Text.ToString();

string Пароль = textBox5.Text.ToString();

string IP = textBox6.Text.ToString();

string Последняя\_дата\_входа = textBox4.Text.ToString();

string Услуги = textBox8.Text.ToString();

string Тип = textBox7.Text.ToString();

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "INSERT INTO Пользователи VALUES ('" + Код\_услуги + "','" + ФИО + "', '" + Логин + "','" + Пароль + "','" + IP + "','" + Последняя\_дата\_входа + "','" + Услуги + "','" + Тип + "')";

SqlCommand dbCommand = new SqlCommand(query, dbConnection);

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные добавлены!", "Внимание!");

dbConnection.Close();

}

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

{

Form2 f = new Form2();

f.Show();

}

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

{

string ConnStr = @"Data Source=sql;Initial Catalog= up\_pugin\_med;Integrated Security=True";

SqlConnection dbConnection = new SqlConnection(ConnStr);

dbConnection.Open();

string query = "SELECT \* FROM [Пользователи]";

SqlDataAdapter da = new SqlDataAdapter(query, ConnStr);

DataSet ds = new DataSet();

da.Fill(ds, "[Пользователи]");

dataGridView1.DataSource = ds.Tables["[Пользователи]"].DefaultView;

dbConnection.Close();

}

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

{

//Проверяем количество выбранных строк

if (dataGridView1.SelectedRows.Count != 1)

{

MessageBox.Show("Выберите одну строку для удаления!", "Сообщение!");

return;

}

// Запомним выбранную строку

int index = dataGridView1.SelectedRows[0].Index;

//Проверим данные в таблице

if (dataGridView1.Rows[index].Cells[0].Value == null)

{

MessageBox.Show("Не все данные введены!", "Сообщение!");

return;

}

//Считаем данные

string id = dataGridView1.Rows[index].Cells[0].Value.ToString();

SqlConnection con = new SqlConnection(ConnStr);

// Выполняем запрос к базе данных

con.Open();//открываем соединение

string SqlText = "DELETE FROM [Пользователи] WHERE Код\_услуги=" + id;//строка запроса

SqlCommand dbCommand = new SqlCommand(SqlText, con);//команда

//Выполняем запрос

if (dbCommand.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка выполнения запроса!", "Ошибка!");

else

MessageBox.Show("Данные удалены !", "Сообщение!");

// Удаляем данные из таблицы в форме

dataGridView1.Rows.RemoveAt(index);

//Закрываем соединение с БД

con.Close();

}

}

}

ПРИЛОЖЕНИЕ Б

public class adminActivity extends AppCompatActivity implements View.OnClickListener{  
  
 Button Add, Clear,edit, BackBtn, BasenameBtn;  
 EditText etName, etService, etResult, etDate, etPrice;  
  
 DBHelper dbHelper;  
 SQLiteDatabase database;  
 ContentValues contentValues;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_admin);  
  
 Add = (Button) findViewById(R.id.dbAdd);  
 Add.setOnClickListener(this);  
  
 edit = (Button) findViewById(R.id.edit);  
 edit.setOnClickListener(this);  
  
 Clear = (Button) findViewById(R.id.dbClear);  
 Clear.setOnClickListener(this);  
  
 BackBtn = (Button) findViewById(R.id.back);  
 BackBtn.setOnClickListener(this);  
  
 BasenameBtn = (Button) findViewById(R.id.baseresults);  
 BasenameBtn.setOnClickListener(this);  
  
 etName = (EditText) findViewById(R.id.name);  
 etService = (EditText) findViewById(R.id.service);  
 etResult = (EditText) findViewById(R.id.result);  
 etDate = (EditText) findViewById(R.id.date);  
 etPrice = (EditText) findViewById(R.id.price);  
  
 dbHelper = new DBHelper(this);  
 database = dbHelper.getWritableDatabase();  
  
 UpdateTable();  
 }  
  
 public void UpdateTable() {  
 Cursor cursor = database.query(DBHelper.TABLE\_RESULTS, null, null, null, null, null, null);  
  
 if (cursor.moveToFirst()) {  
 int idIndex = cursor.getColumnIndex(DBHelper.KEY\_ID\_RESULT);  
 int nameIndex = cursor.getColumnIndex(DBHelper.KEY\_NAME2);  
 int serviceIndex = cursor.getColumnIndex(DBHelper.KEY\_SERVICE);  
 int resultIndex = cursor.getColumnIndex(DBHelper.KEY\_RESULT);  
 int dateIndex = cursor.getColumnIndex(DBHelper.KEY\_DATE);  
 int priceIndex = cursor.getColumnIndex(DBHelper.KEY\_PRICE);  
 TableLayout dbOutput = findViewById(R.id.output);  
 dbOutput.removeAllViews();  
 do {  
 TableRow dbOutputRow = new TableRow(this);  
 dbOutputRow.setLayoutParams(new LinearLayout.LayoutParams(LinearLayout.LayoutParams.MATCH\_PARENT, LinearLayout.LayoutParams.WRAP\_CONTENT));  
  
 TableRow.LayoutParams params = new TableRow.LayoutParams(TableRow.LayoutParams.WRAP\_CONTENT, TableRow.LayoutParams.WRAP\_CONTENT);  
  
 TextView outputName = new TextView(this);  
 params.weight = 5.0f;  
 outputName.setLayoutParams(params);  
 outputName.setText(cursor.getString(nameIndex));  
 dbOutputRow.addView(outputName);  
  
 TextView outputService = new TextView(this);  
 params.weight = 5.0f;  
 outputService.setLayoutParams(params);  
 outputService.setText(cursor.getString(serviceIndex));  
 dbOutputRow.addView(outputService);  
  
 TextView outputResult = new TextView(this);  
 params.weight = 2.0f;  
 outputResult.setLayoutParams(params);  
 outputResult.setText(cursor.getString(resultIndex));  
 dbOutputRow.addView(outputResult);  
  
 TextView outputDate = new TextView(this);  
 params.weight = 4.0f;  
 outputDate.setLayoutParams(params);  
 outputDate.setText(cursor.getString(dateIndex));  
 dbOutputRow.addView(outputDate);  
  
 TextView outputPrice = new TextView(this);  
 params.weight = 3.0f;  
 outputPrice.setLayoutParams(params);  
 outputPrice.setText(cursor.getString(priceIndex));  
 dbOutputRow.addView(outputPrice);  
  
 Button btnDelete = new Button(this);  
 btnDelete.setOnClickListener(this);  
 params.weight = 1.0f;  
 btnDelete.setLayoutParams(params);  
 btnDelete.setText("Удалить");  
 btnDelete.setId(cursor.getInt(idIndex));  
 dbOutputRow.addView(btnDelete);  
  
 dbOutput.addView(dbOutputRow);  
  
 } while (cursor.moveToNext());  
 }  
 cursor.close();  
 }  
  
 @Override  
 public void onClick(View v) {  
 switch (v.getId()) {  
  
 case R.id.back:  
 Intent intent = new Intent(this, MainActivity.class);  
 startActivity(intent);  
 break;  
  
 case R.id.baseresults:  
 Intent intent2 = new Intent(this, BaseNameActivity.class);  
 startActivity(intent2);  
 break;  
  
 case R.id.dbAdd:  
 String name = etName.getText().toString();  
 String service = etService.getText().toString();  
 String result = etResult.getText().toString();  
 String date = etDate.getText().toString();  
 String price = etPrice.getText().toString();  
  
 contentValues = new ContentValues();  
  
 contentValues.put(DBHelper.KEY\_NAME2, name);  
 contentValues.put(DBHelper.KEY\_SERVICE, service);  
 contentValues.put(DBHelper.KEY\_RESULT, result);  
 contentValues.put(DBHelper.KEY\_DATE, date);  
 contentValues.put(DBHelper.KEY\_PRICE, price);  
  
 database.insert(DBHelper.TABLE\_RESULTS, null, contentValues);  
 UpdateTable();  
 break;  
  
  
 // редатировать записи  
 case R.id.edit:  
 String name3 = etName.getText().toString();  
 String Service3 = etService.getText().toString();  
 String etResult3 = etResult.getText().toString();  
 String etDate3 = etDate.getText().toString();  
 String etPrice3 = etPrice.getText().toString();  
  
 contentValues = new ContentValues();  
  
 contentValues.put(DBHelper.KEY\_SERVICE, Service3);  
 contentValues.put(DBHelper.KEY\_RESULT, etResult3);  
 contentValues.put(DBHelper.KEY\_DATE, etDate3);  
 contentValues.put(DBHelper.KEY\_PRICE, etPrice3);  
  
 database.update( DBHelper.TABLE\_RESULTS, contentValues, "name = ?",  
 new String[] { name3 });  
 UpdateTable();  
 break;  
  
 case R.id.dbClear:  
 database.delete(DBHelper.TABLE\_RESULTS, null, null);  
 break;  
//конпка удалить в таблице  
 default:  
 View outputDBRow = (View) v.getParent();  
 ViewGroup outputDB = (ViewGroup) outputDBRow.getParent();  
  
 outputDB.removeView(outputDBRow);  
 outputDB.invalidate();  
  
 //удаление из бд записей  
 database.delete(DBHelper.TABLE\_RESULTS, DBHelper.KEY\_ID\_RESULT + " = ?", new String[]{String.valueOf((v.getId()))});  
  
 contentValues = new ContentValues();  
 Cursor cursorUpdater = database.query(DBHelper.TABLE\_RESULTS, null, null, null, null, null, null);  
 if (cursorUpdater.moveToFirst()) {  
 int idIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_ID\_RESULT);  
 int nameIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_NAME2);  
 int serviceIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_SERVICE);  
 int resultIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_RESULT);  
 int dateIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_DATE);  
 int priceIndex = cursorUpdater.getColumnIndex(DBHelper.KEY\_PRICE);  
 int realID = 1;  
 do {  
 if (cursorUpdater.getInt(idIndex) > realID) {  
 contentValues.put(DBHelper.KEY\_ID\_RESULT, realID);  
 contentValues.put(DBHelper.KEY\_NAME2, cursorUpdater.getString(nameIndex));  
 contentValues.put(DBHelper.KEY\_SERVICE, cursorUpdater.getString(serviceIndex));  
 contentValues.put(DBHelper.KEY\_RESULT, cursorUpdater.getString(resultIndex));  
 contentValues.put(DBHelper.KEY\_DATE, cursorUpdater.getString(dateIndex));  
 contentValues.put(DBHelper.KEY\_PRICE, cursorUpdater.getString(priceIndex));  
 database.replace(DBHelper.TABLE\_RESULTS, null, contentValues);  
 }  
 realID++;  
 } while (cursorUpdater.moveToNext());  
 if (cursorUpdater.moveToLast() && v.getId() != realID) {  
 database.delete(DBHelper.TABLE\_RESULTS, DBHelper.KEY\_ID\_RESULT + " = ?", new String[]{cursorUpdater.getString(idIndex)});  
 }  
 UpdateTable();  
 }  
 break;  
 }  
 }  
}

public class BaseNameActivity extends AppCompatActivity implements View.OnClickListener{  
  
 Button Add, Clear,edit, BackBtn, BaseresultsBtn;  
 EditText etName, etLogin, etPassword, etDolgnost;  
  
 DBHelper dbHelper;  
 SQLiteDatabase database;  
 ContentValues contentValues;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_base\_name*);  
  
 Add = (Button) findViewById(R.id.*add*);  
 Add.setOnClickListener(this);  
  
 Clear = (Button) findViewById(R.id.*clear*);  
 Clear.setOnClickListener(this);  
  
 edit = (Button) findViewById(R.id.*edit*);  
 edit.setOnClickListener(this);  
  
 BackBtn = (Button) findViewById(R.id.*back*);  
 BackBtn.setOnClickListener(this);  
  
 BaseresultsBtn = (Button) findViewById(R.id.*baseresults*);  
 BaseresultsBtn.setOnClickListener(this);  
  
 etName = (EditText) findViewById(R.id.*name*);  
 etLogin = (EditText) findViewById(R.id.*login*);  
 etPassword = (EditText) findViewById(R.id.*password*);  
 etDolgnost = (EditText) findViewById(R.id.*dolgnost*);  
  
 dbHelper = new DBHelper(this);  
 database = dbHelper.getWritableDatabase();  
  
 UpdateTable();  
 }  
  
 public void UpdateTable() {  
 Cursor cursor = database.query(DBHelper.*TABLE\_USERS*, null, null, null, null, null, null);  
  
 if (cursor.moveToFirst()) {  
 int idIndex = cursor.getColumnIndex(DBHelper.*KEY\_ID\_USER*);  
 int nameIndex = cursor.getColumnIndex(DBHelper.*KEY\_NAME*);  
 int loginIndex = cursor.getColumnIndex(DBHelper.*KEY\_LOGIN*);  
 int passwordIndex = cursor.getColumnIndex(DBHelper.*KEY\_PASSWORD*);  
 int dolgnostIndex = cursor.getColumnIndex(DBHelper.*KEY\_DOLGNOST*);  
 TableLayout dbOutput = findViewById(R.id.*output*);  
 dbOutput.removeAllViews();  
 do {  
 TableRow dbOutputRow = new TableRow(this);  
 dbOutputRow.setLayoutParams(new LinearLayout.LayoutParams(LinearLayout.LayoutParams.*MATCH\_PARENT*, LinearLayout.LayoutParams.*WRAP\_CONTENT*));  
  
 TableRow.LayoutParams params = new TableRow.LayoutParams(TableRow.LayoutParams.*WRAP\_CONTENT*, TableRow.LayoutParams.*WRAP\_CONTENT*);  
  
 TextView outputName = new TextView(this);  
 params.weight = 5.0f;  
 outputName.setLayoutParams(params);  
 outputName.setText(cursor.getString(nameIndex));  
 dbOutputRow.addView(outputName);  
  
 TextView outputLogin = new TextView(this);  
 params.weight = 5.0f;  
 outputLogin.setLayoutParams(params);  
 outputLogin.setText(cursor.getString(loginIndex));  
 dbOutputRow.addView(outputLogin);  
  
 TextView outputPassword = new TextView(this);  
 params.weight = 5.0f;  
 outputPassword.setLayoutParams(params);  
 outputPassword.setText(cursor.getString(passwordIndex));  
 dbOutputRow.addView(outputPassword);  
  
 TextView outputDolgnost = new TextView(this);  
 params.weight = 4.0f;  
 outputDolgnost.setLayoutParams(params);  
 outputDolgnost.setText(cursor.getString(dolgnostIndex));  
 dbOutputRow.addView(outputDolgnost);  
  
 Button btnDelete = new Button(this);  
 btnDelete.setOnClickListener(this);  
 params.weight = 1.0f;  
 btnDelete.setLayoutParams(params);  
 btnDelete.setText("Удалить");  
 btnDelete.setId(cursor.getInt(idIndex));  
 dbOutputRow.addView(btnDelete);  
  
 dbOutput.addView(dbOutputRow);  
  
 } while (cursor.moveToNext());  
 }  
 cursor.close();  
 }  
  
 @Override  
 public void onClick(View v) {  
 switch (v.getId()) {  
  
 case R.id.*back*:  
 Intent intent = new Intent(this, MainActivity.class);  
 startActivity(intent);  
 break;  
  
 case R.id.*baseresults*:  
 Intent intent2 = new Intent(this, adminActivity.class);  
 startActivity(intent2);  
 break;  
  
 case R.id.*add*:  
 String name = etName.getText().toString();  
 String login = etLogin.getText().toString();  
 String password = etPassword.getText().toString();  
 String dolgnost = etDolgnost.getText().toString();  
  
 contentValues = new ContentValues();  
  
 contentValues.put(DBHelper.*KEY\_NAME*, name);  
 contentValues.put(DBHelper.*KEY\_LOGIN*, login);  
 contentValues.put(DBHelper.*KEY\_PASSWORD*, password);  
 contentValues.put(DBHelper.*KEY\_DOLGNOST*, dolgnost);  
  
 database.insert(DBHelper.*TABLE\_USERS*, null, contentValues);  
 UpdateTable();  
 break;  
  
 // редатировать записи  
 case R.id.*edit*:  
 String name3 = etName.getText().toString();  
 String login3 = etLogin.getText().toString();  
 String password3 = etPassword.getText().toString();  
 String dolgnost3 = etDolgnost.getText().toString();  
  
 contentValues = new ContentValues();  
  
 contentValues.put(DBHelper.*KEY\_LOGIN*, login3);  
 contentValues.put(DBHelper.*KEY\_PASSWORD*, password3);  
 contentValues.put(DBHelper.*KEY\_DOLGNOST*, dolgnost3);  
  
 database.update( DBHelper.*TABLE\_USERS*, contentValues, "name = ?",  
 new String[] { name3 });  
 UpdateTable();  
 break;  
  
 case R.id.*clear*:  
 database.delete(DBHelper.*TABLE\_USERS*, null, null);  
 break;  
//конпка удалить в таблице  
 default:  
 View outputDBRow = (View) v.getParent();  
 ViewGroup outputDB = (ViewGroup) outputDBRow.getParent();  
  
 outputDB.removeView(outputDBRow);  
 outputDB.invalidate();  
//удаление из бд записей  
 database.delete(DBHelper.*TABLE\_USERS*, DBHelper.*KEY\_ID\_USER* + " = ?", new String[]{String.*valueOf*((v.getId()))});  
  
 contentValues = new ContentValues();  
 Cursor cursorUpdater = database.query(DBHelper.*TABLE\_USERS*, null, null, null, null, null, null);  
 if (cursorUpdater.moveToFirst()) {  
 int idIndex = cursorUpdater.getColumnIndex(DBHelper.*KEY\_ID\_USER*);  
 int nameIndex = cursorUpdater.getColumnIndex(DBHelper.*KEY\_NAME*);  
 int loginIndex = cursorUpdater.getColumnIndex(DBHelper.*KEY\_LOGIN*);  
 int passwordIndex = cursorUpdater.getColumnIndex(DBHelper.*KEY\_PASSWORD*);  
 int dolgnostIndex = cursorUpdater.getColumnIndex(DBHelper.*KEY\_DOLGNOST*);  
 int realID = 1;  
 do {  
 if (cursorUpdater.getInt(idIndex) > realID) {  
 contentValues.put(DBHelper.*KEY\_ID\_RESULT*, realID);  
 contentValues.put(DBHelper.*KEY\_NAME2*, cursorUpdater.getString(nameIndex));  
 contentValues.put(DBHelper.*KEY\_SERVICE*, cursorUpdater.getString(loginIndex));  
 contentValues.put(DBHelper.*KEY\_RESULT*, cursorUpdater.getString(passwordIndex));  
 contentValues.put(DBHelper.*KEY\_DATE*, cursorUpdater.getString(dolgnostIndex));  
 database.replace(DBHelper.*TABLE\_RESULTS*, null, contentValues);  
 }  
 realID++;  
 } while (cursorUpdater.moveToNext());  
 if (cursorUpdater.moveToLast() && v.getId() != realID) {  
 database.delete(DBHelper.*TABLE\_USERS*, DBHelper.*KEY\_ID\_USER* + " = ?", new String[]{cursorUpdater.getString(idIndex)});  
 }  
 UpdateTable();  
 }  
 break;  
 }  
 }

public class MainActivity extends AppCompatActivity implements View.OnClickListener{  
  
 Button loginBtn;  
 EditText loginField, passwordField;  
  
 DBHelper dbHelper;  
 SQLiteDatabase database;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 loginField = (EditText) findViewById(R.id.*loginField*);  
 passwordField = (EditText) findViewById(R.id.*passwordField*);  
  
 loginBtn = (Button) findViewById(R.id.*loginBtn*);  
 loginBtn.setOnClickListener(this);  
  
 dbHelper = new DBHelper(this);  
 database = dbHelper.getWritableDatabase();  
 }  
  
 @Override  
 public void onClick(View v) {  
 switch (v.getId()) {  
 case R.id.*loginBtn*:  
 //startActivity(new Intent(this, AdminActivity.class));  
 Cursor logCursor = database.query(DBHelper.*TABLE\_USERS*, null, null, null, null, null, null);  
 boolean logged = false;  
 if (logCursor.moveToFirst()) {  
 int userIndex = logCursor.getColumnIndex(DBHelper.*KEY\_LOGIN*);  
 int nameIndex = logCursor.getColumnIndex(DBHelper.*KEY\_NAME*);  
 int passwordIndex = logCursor.getColumnIndex(DBHelper.*KEY\_PASSWORD*);  
 int dolgnost = logCursor.getColumnIndex(DBHelper.*KEY\_DOLGNOST*);  
 do {  
 if((loginField.getText().toString().equals(logCursor.getString(userIndex))) && (passwordField.getText().toString().equals(logCursor.getString(passwordIndex))) && (logCursor.getString(dolgnost).equals("admin"))){  
 startActivity(new Intent(this, adminActivity.class));  
 logged = true;  
 }  
 else if ((loginField.getText().toString().equals(logCursor.getString(userIndex))) && (passwordField.getText().toString().equals(logCursor.getString(passwordIndex)))) {  
 Intent intent = new Intent(this, userActivity.class);  
 intent.putExtra("name", logCursor.getString(nameIndex));  
 startActivity(intent);  
 logged = true;  
 }  
 } while (logCursor.moveToNext());  
 }  
 logCursor.close();  
 if (!logged)  
 Toast.*makeText*(this, "Введённый логин и пароль не найден.", Toast.*LENGTH\_LONG*).show();  
 break;  
 }  
 }  
}

public class userActivity extends AppCompatActivity implements View.OnClickListener {  
 Button BackBtn;  
 EditText etName, etService, etResult, etDate, etPrice;  
  
 DBHelper dbHelper;  
 SQLiteDatabase database;  
  
 String name;  
  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_user*);  
  
  
 BackBtn = (Button) findViewById(R.id.*back2*);  
 BackBtn.setOnClickListener(this);  
  
 etName = (EditText) findViewById(R.id.*name*);  
 etService = (EditText) findViewById(R.id.*login*);  
 etResult = (EditText) findViewById(R.id.*password*);  
 etDate = (EditText) findViewById(R.id.*dolgnost*);  
 etPrice = (EditText) findViewById(R.id.*price*);  
  
 dbHelper = new DBHelper(this);  
 database = dbHelper.getWritableDatabase();  
  
 Intent intent = getIntent();  
 name = intent.getStringExtra("name");  
  
 UpdateTable();  
 }  
  
 public void UpdateTable() {  
 Cursor cursor = database.query(DBHelper.*TABLE\_RESULTS*, null, DBHelper.*KEY\_NAME2* + " = ?", new String[]{name}, null, null, null);  
  
 if (cursor.moveToFirst()) {  
 int idIndex = cursor.getColumnIndex(DBHelper.*KEY\_ID\_RESULT*);  
 int nameIndex = cursor.getColumnIndex(DBHelper.*KEY\_NAME2*);  
 int serviceIndex = cursor.getColumnIndex(DBHelper.*KEY\_SERVICE*);  
 int resultIndex = cursor.getColumnIndex(DBHelper.*KEY\_RESULT*);  
 int dateIndex = cursor.getColumnIndex(DBHelper.*KEY\_DATE*);  
 int priceIndex = cursor.getColumnIndex(DBHelper.*KEY\_PRICE*);  
 TableLayout dbOutput = findViewById(R.id.*output3*);  
 dbOutput.removeAllViews();  
 do {  
 TableRow dbOutputRow = new TableRow(this);  
 dbOutputRow.setLayoutParams(new LinearLayout.LayoutParams(LinearLayout.LayoutParams.*MATCH\_PARENT*, LinearLayout.LayoutParams.*WRAP\_CONTENT*));  
  
 TableRow.LayoutParams params = new TableRow.LayoutParams(TableRow.LayoutParams.*WRAP\_CONTENT*, TableRow.LayoutParams.*WRAP\_CONTENT*);  
  
 TextView outputName = new TextView(this);  
 params.weight = 5.0f;  
 outputName.setLayoutParams(params);  
 outputName.setText(cursor.getString(nameIndex));  
 dbOutputRow.addView(outputName);  
  
 TextView outputService = new TextView(this);  
 params.weight = 5.0f;  
 outputService.setLayoutParams(params);  
 outputService.setText(cursor.getString(serviceIndex));  
 dbOutputRow.addView(outputService);  
  
 TextView outputResult = new TextView(this);  
 params.weight = 2.0f;  
 outputResult.setLayoutParams(params);  
 outputResult.setText(cursor.getString(resultIndex));  
 dbOutputRow.addView(outputResult);  
  
 TextView outputDate = new TextView(this);  
 params.weight = 4.0f;  
 outputDate.setLayoutParams(params);  
 outputDate.setText(cursor.getString(dateIndex));  
 dbOutputRow.addView(outputDate);  
  
 TextView outputPrice = new TextView(this);  
 params.weight = 3.0f;  
 outputPrice.setLayoutParams(params);  
 outputPrice.setText(cursor.getString(priceIndex));  
 dbOutputRow.addView(outputPrice);  
  
 dbOutput.addView(dbOutputRow);  
  
 } while (cursor.moveToNext());  
 }  
 cursor.close();  
 }  
  
 @Override  
 public void onClick(View v) {  
 switch (v.getId()) {  
 case R.id.*back2*:  
 Intent intent = new Intent(this, MainActivity.class);  
 startActivity(intent);  
 break;  
 }  
 }  
}