Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный** **исследовательский политехнический университет»**

Факультет: Прикладной математики и механики

Кафедра: Вычислительной математики, механики и биомеханики

Направление: 09.03.02 «Информационные системы и технологии»

Профиль бакалавриата: «Информационные системы и технологии»

**РАЗРАБОТКА ПРОГРАММЫ**

по дисциплине

**«КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ»**

Выполнил:

студент гр.ИСТ-19-2б

Семёнова А.С., Ларина А.А.

Принял:

ст. преподаватель, Банников Р.Ю.

Пермь 2023

# РЕАЛИЗАЦИЯ ПРОГРАММЫ

**Цель работы:** запрограммировать обозначенный в ТЗ функционал с учётом созданной БД и разработанных структур классов.

**Задачи:**

1. Запрограммировать обозначенный в ТЗ функционал с учётом созданной БД и разработанных структур классов;
2. Привести программный код, согласно функциям.
3. **Разработка программы**

Пройдемся по основному функционалу системы:

1. В системе должна быть функция авторизации

Внешний интерфейс для функции авторизации представлен на рисунке 1. Здесь нужно вводить логин и пароль для того, чтобы войти в приложение.

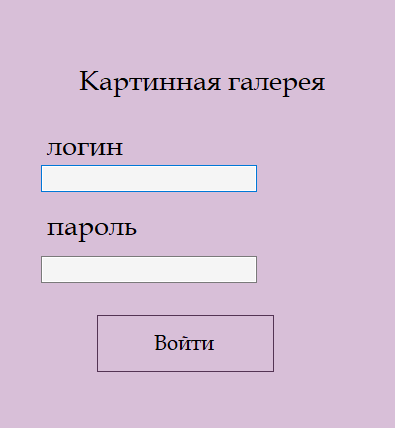


Рисунок 1 – Интерфейс функции авторизации

**Программный код для разделения прав пользователей:**

switch (user\_role)

{

case "1": db.CloseConnection(); this.Visible = false; Admin a = new Admin(); a.Show(); break;

case "2": db.CloseConnection(); this.Visible = false; User u = new User(); u.Show(); break;

}

}

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации о картинах.

Интерфейс ИС «Картинная галерея» для добавления, удаления и изменения картин, содержащий в себе такую информацию, как название картины, дату создания, ФИО автора, название жанра и название выставки (рис. 2).

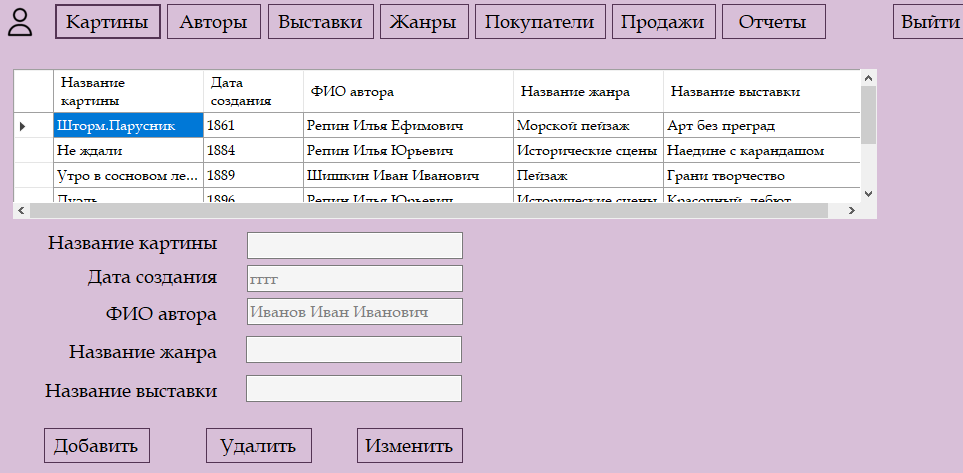


Рисунок 2 – Интерфейс для радела «Картины»

Для добавления картины необходимо ввести все значения. Для удаления нужно щёлкнуть по строке, которую нужно удалить. Для изменения необходимо щёлкнуть по нужному значению из таблицы и данные переносятся в поля. Дана рекомендация о заполнении поля «Дата создания» в формате «гггг».

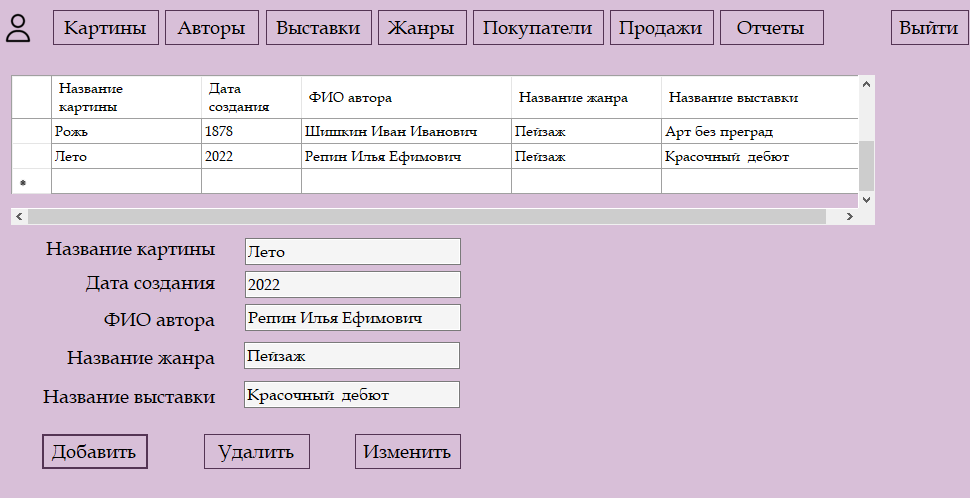


Рисунок 3 – Интерфейс для функции добавления картин

Для того чтобы добавлять картины с наименованиями, а не с идентификаторами нужно написать процедуру добавления картин, а затем программный код.

**SQL-код для добавления картин:**

CREATE PROCEDURE insert\_picture(n\_p VARCHAR(45), y\_c\_p VARCHAR(45), f\_a VARCHAR(45), n\_j VARCHAR(45), n\_e VARCHAR(45))

BEGIN

DECLARE id\_aut int;

DECLARE id\_ja int;

DECLARE id\_ex int;

SELECT author.id\_author into id\_aut FROM author WHERE f\_a = fio\_author;

SELECT janre.id\_janre into id\_ja FROM janre WHERE n\_j = name\_janre;

SELECT exhibitions.id\_exhibitions into id\_ex FROM exhibitions WHERE n\_e = name\_exhibitions;

INSERT INTO picture(name\_picture, year\_create\_picture, id\_author, id\_janre, id\_exhibitions)

VALUES

(n\_p, y\_c\_p, id\_aut, id\_ja, id\_ex);

END

**Программный код для добавления картин:**

db.OpenConnection();

string procedure\_name = "insert\_picture";

MySqlCommand comm\_Add = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Add.CommandType = CommandType.StoredProcedure;

MySqlParameter n\_p = new MySqlParameter

{

ParameterName = "n\_p",

Value = textBox\_name\_picture.Text

};

MySqlParameter y\_c\_p = new MySqlParameter

{

ParameterName = "y\_c\_p",

Value = textBox\_year\_create\_picture.Text

};

MySqlParameter f\_a = new MySqlParameter

{

ParameterName = "f\_a",

Value = textBox\_fio\_author.Text

};

MySqlParameter n\_j = new MySqlParameter

{

ParameterName = "n\_j",

Value = textBox\_name\_janre.Text

};

MySqlParameter n\_e = new MySqlParameter

{

ParameterName = "n\_e",

Value = textBox\_name\_exhibitions.Text

};

comm\_Add.Parameters.Add(n\_p);

comm\_Add.Parameters.Add(y\_c\_p);

comm\_Add.Parameters.Add(f\_a);

comm\_Add.Parameters.Add(n\_j);

comm\_Add.Parameters.Add(n\_e);

comm\_Add.ExecuteNonQuery();

db.CloseConnection();

ShowPicture();

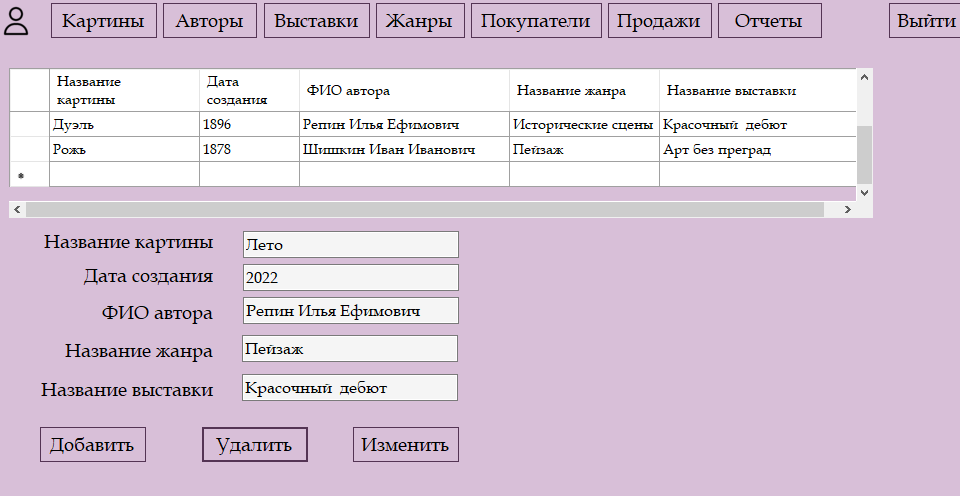
****

Рисунок 4 – Интерфейс для функции удаления картин

Для того чтобы удалять картины с наименованиями, а не с идентификаторами нужно написать процедуру удаления картин, а затем программный код.

**SQL-код для удаления картин:**

CREATE PROCEDURE delete\_picture(n\_p VARCHAR(45), y\_c\_p VARCHAR(45), a\_i VARCHAR(45), j\_i VARCHAR(45), e\_i VARCHAR(45))

BEGIN

DECLARE id\_aut int;

DECLARE id\_ja int;

DECLARE id\_ex int;

SELECT author.id\_author into id\_aut FROM author WHERE a\_i = fio\_author;

SELECT janre.id\_janre into id\_ja FROM janre WHERE j\_i = name\_janre;

SELECT exhibitions.id\_exhibitions into id\_ex FROM exhibitions WHERE e\_i = name\_exhibitions;

DELETE FROM picture WHERE n\_p = name\_picture AND

y\_c\_p = year\_create\_picture AND

id\_aut = id\_author AND

id\_ja = id\_janre AND

id\_ex = id\_exhibitions;

END

**Программный код для удаления картин:**

db.OpenConnection();

string procedure\_name = "delete\_picture";

MySqlCommand comm\_Del = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Del.CommandType = CommandType.StoredProcedure;

MySqlParameter n\_p = new MySqlParameter

{

ParameterName = "n\_p",

Value = textBox\_name\_picture.Text

};

MySqlParameter y\_c\_p = new MySqlParameter

{

ParameterName = "y\_c\_p",

Value = textBox\_year\_create\_picture.Text

};

MySqlParameter a\_i = new MySqlParameter

{

ParameterName = "a\_i",

Value = textBox\_fio\_author.Text

};

MySqlParameter j\_i = new MySqlParameter

{

ParameterName = "j\_i",

Value = textBox\_name\_janre.Text

};

MySqlParameter e\_i = new MySqlParameter

{

ParameterName = "e\_i",

Value = textBox\_name\_exhibitions.Text

};

comm\_Del.Parameters.Add(n\_p);

comm\_Del.Parameters.Add(y\_c\_p);

comm\_Del.Parameters.Add(a\_i);

comm\_Del.Parameters.Add(j\_i);

comm\_Del.Parameters.Add(e\_i);

comm\_Del.ExecuteNonQuery();

db.CloseConnection();

ShowPicture();

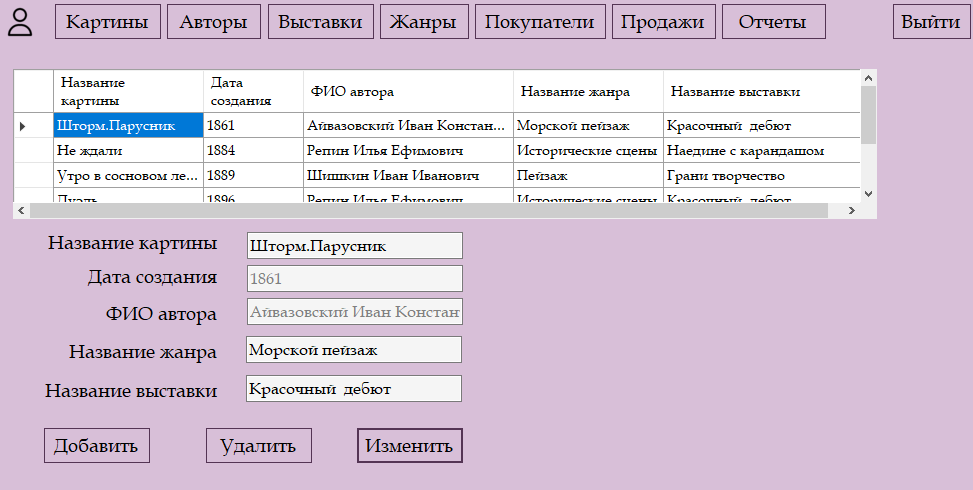


Рисунок 5 – Интерфейс для функции изменения картин

Для того чтобы изменить товар с наименованиями картины, а не с идентификаторами нужно написать процедуру изменения картин, а затем программный код.

**SQL-код для изменения картин:**

CREATE PROCEDURE update\_picture(id INT, new\_n\_p VARCHAR(45), new\_y\_c\_p VARCHAR(45), new\_a\_i VARCHAR(45), new\_j\_i VARCHAR(45), new\_e\_i VARCHAR(45))

BEGIN

DECLARE new\_id\_aut int;

DECLARE new\_id\_ja int;

DECLARE new\_id\_ex int;

SELECT author.id\_author into new\_id\_aut FROM author WHERE new\_a\_i = fio\_author;

SELECT janre.id\_janre into new\_id\_ja FROM janre WHERE new\_j\_i = name\_janre;

SELECT exhibitions.id\_exhibitions into new\_id\_ex FROM exhibitions WHERE new\_e\_i = name\_exhibitions;

UPDATE picture

SET name\_picture = new\_n\_p,

year\_create\_picture = new\_y\_c\_p,

id\_author = new\_id\_aut,

id\_janre = new\_id\_ja,

id\_exhibitions = new\_id\_ex

WHERE id\_picture = id;

END

**Программный код для изменения картин:**

db.OpenConnection();

string procedure\_name = "update\_picture";

MySqlCommand comm\_Upd = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Upd.CommandType = CommandType.StoredProcedure;

string value = dataGridView1.CurrentRow.Cells[0].Value.ToString();

MySqlParameter id\_param = new MySqlParameter

{

ParameterName = "id",

Value = value

};

MySqlParameter new\_n\_p = new MySqlParameter

{

ParameterName = "new\_n\_p",

Value = textBox\_name\_picture.Text

};

MySqlParameter new\_y\_c\_p = new MySqlParameter

{

ParameterName = "new\_y\_c\_p",

Value = textBox\_year\_create\_picture.Text//цена

};

MySqlParameter new\_a\_i = new MySqlParameter

{

ParameterName = "new\_a\_i",

Value = textBox\_fio\_author.Text

};

MySqlParameter new\_j\_i = new MySqlParameter

{

ParameterName = "new\_j\_i",

Value = textBox\_name\_janre.Text//цена

};

MySqlParameter new\_e\_i = new MySqlParameter

{

ParameterName = "new\_e\_i",

Value = textBox\_name\_exhibitions.Text

};

comm\_Upd.Parameters.Add(id\_param);

comm\_Upd.Parameters.Add(new\_n\_p);

comm\_Upd.Parameters.Add(new\_y\_c\_p);

comm\_Upd.Parameters.Add(new\_a\_i);

comm\_Upd.Parameters.Add(new\_j\_i);

comm\_Upd.Parameters.Add(new\_e\_i);

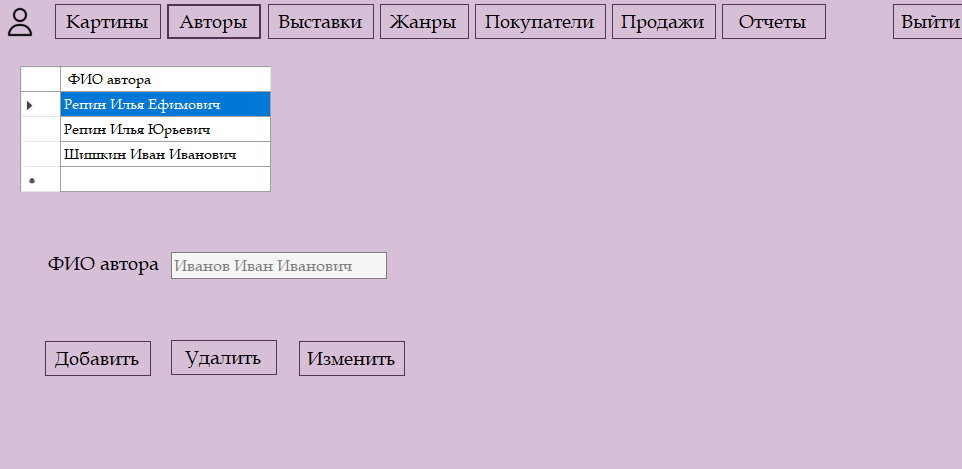
comm\_Upd.ExecuteNonQuery();

db.CloseConnection();

ShowPicture();

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации об авторах.

Интерфейс ИС «Картинная галерея» для добавления, удаления и изменения авторов (рис. 6).



Для добавления автора необходимо ввести все значения. Для удаления нужно щёлкнуть по строке, которую нужно удалить. Для изменения необходимо щёлкнуть по нужному значению из таблицы и данные переносятся в поля.

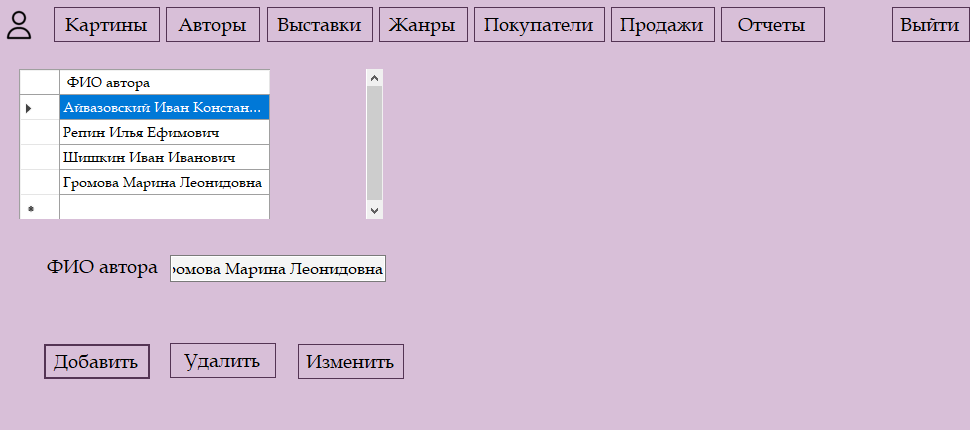


Рисунок 7 – Интерфейс для функции добавления автора

**Программный код для добавления автора:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "INSERT INTO author (fio\_author) VALUES ('" + textBox\_fio\_author2.Text + "');";

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowAuthor();

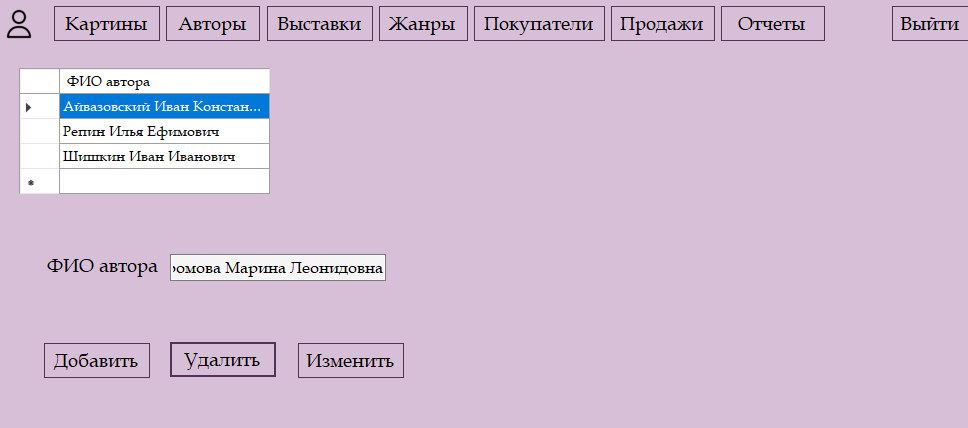


Рисунок 8 – Интерфейс для функции удаления автора

**Программный код для удаления автора:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "DELETE FROM author WHERE id\_author = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowAuthor();

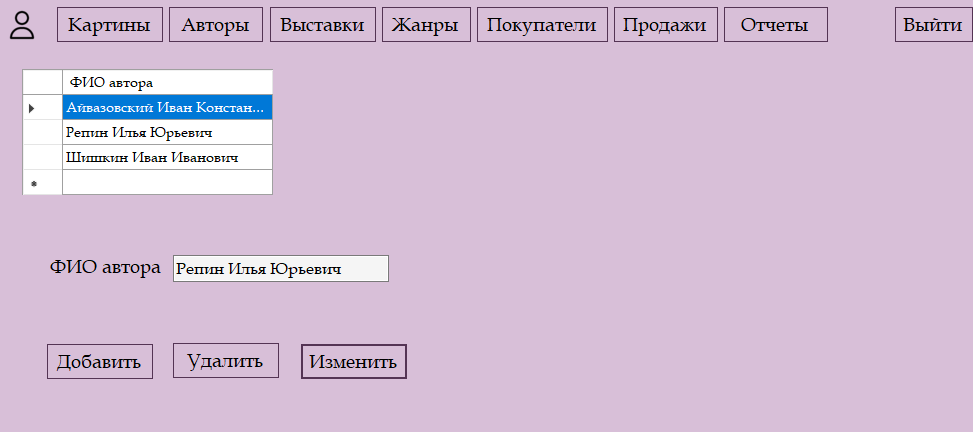


Рисунок 9 – Интерфейс для функции изменения автора

**Программный код для изменения автора:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "UPDATE author SET fio\_author = '" + textBox\_fio\_author2.Text + "' WHERE id\_author = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowAuthor();

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации о выставках.

В разделе «Выставки» в таблице представлены названия выставок, даты начала и окончания выставок, количество проданных билетов, цена за один билет и название музея.

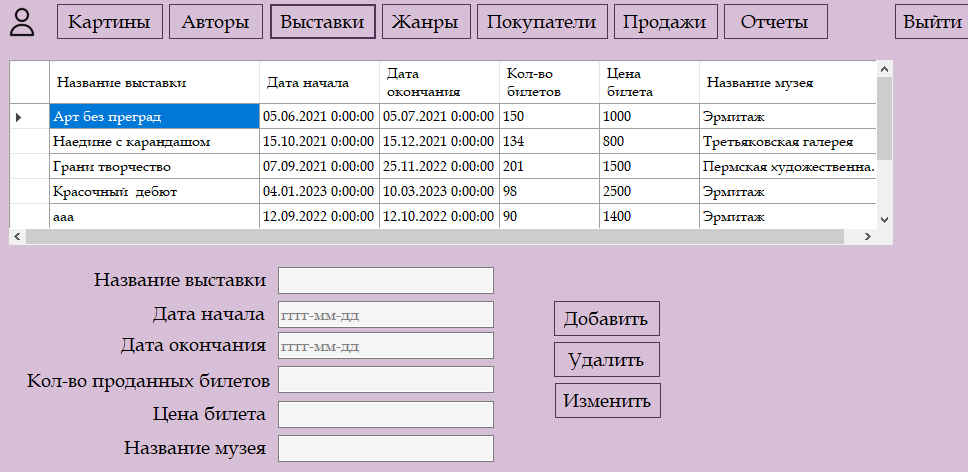


Рисунок 10 – Интерфейс для радела «Выставки»

Для добавления автора необходимо ввести все значения. Для удаления нужно щёлкнуть по строке, которую нужно удалить. Для изменения необходимо щёлкнуть по нужному значению из таблицы и данные переносятся в поля.

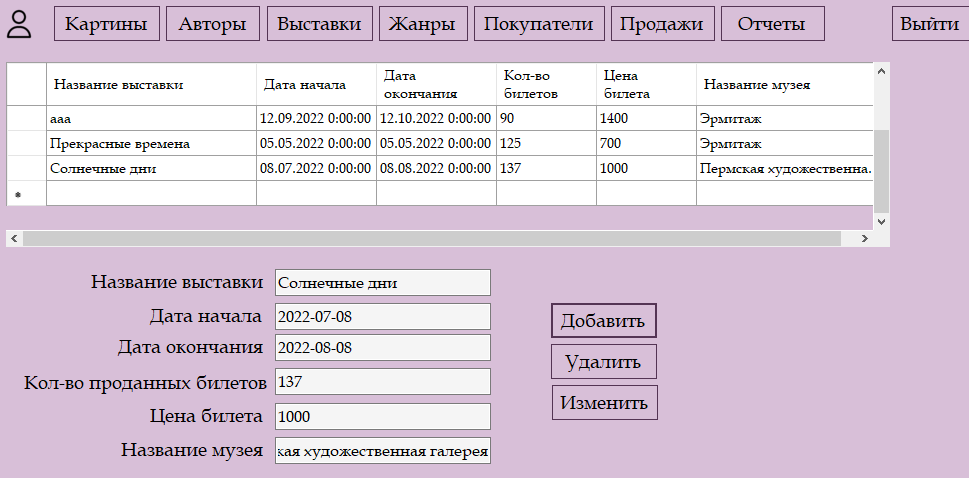


Рисунок 11 – Интерфейс для функции добавления выставки

**Программный код для добавления выставок:**

db.OpenConnection();

string procedure\_name = "insert\_exhibitions";

MySqlCommand comm\_Add = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Add.CommandType = CommandType.StoredProcedure;

MySqlParameter n\_e = new MySqlParameter

{

ParameterName = "n\_e",

Value = textBox\_name\_exhibitions2.Text

};

MySqlParameter d\_s = new MySqlParameter

{

ParameterName = "d\_s",

Value = textBox\_date\_start.Text

};

MySqlParameter d\_e = new MySqlParameter

{

ParameterName = "d\_e",

Value = textBox\_date\_end.Text

};

MySqlParameter s\_p = new MySqlParameter

{

ParameterName = "s\_p",

Value = textBox\_sold\_places.Text

};

MySqlParameter p\_e = new MySqlParameter

{

ParameterName = "p\_e",

Value = Convert.ToInt32(textBox\_price\_exhibitions.Text)//цена билета

};

MySqlParameter n\_m = new MySqlParameter

{

ParameterName = "n\_m",

Value = textBox\_name\_museum2.Text

};

comm\_Add.Parameters.Add(n\_e);

comm\_Add.Parameters.Add(d\_s);

comm\_Add.Parameters.Add(d\_e);

comm\_Add.Parameters.Add(s\_p);

comm\_Add.Parameters.Add(p\_e);

comm\_Add.Parameters.Add(n\_m);

comm\_Add.ExecuteNonQuery();

db.CloseConnection();

ShowExhibitions();

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации о жанрах.

В разделе «Жанры» в таблице представлены названия жанров.

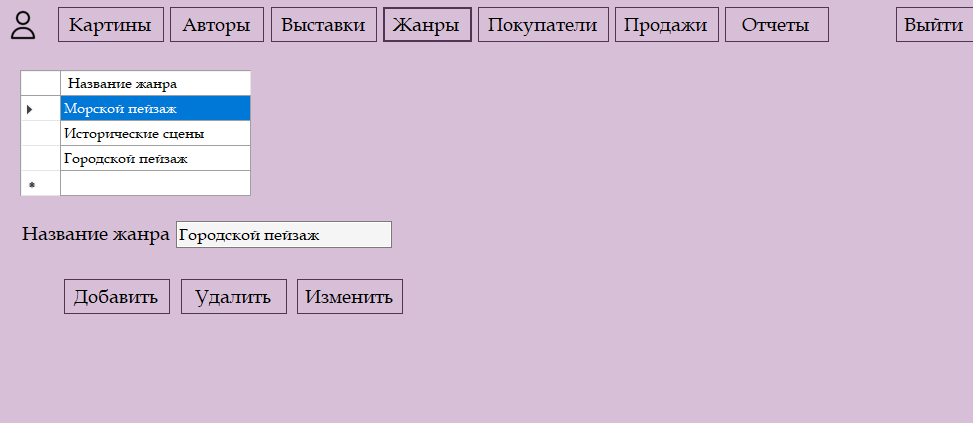


Рисунок 12 – Интерфейс для радела «Жанры»

Для добавления жанра необходимо ввести все значения. Для удаления нужно щёлкнуть по строке, которую нужно удалить. Для изменения необходимо щёлкнуть по нужному значению из таблицы и данные переносятся в поля.

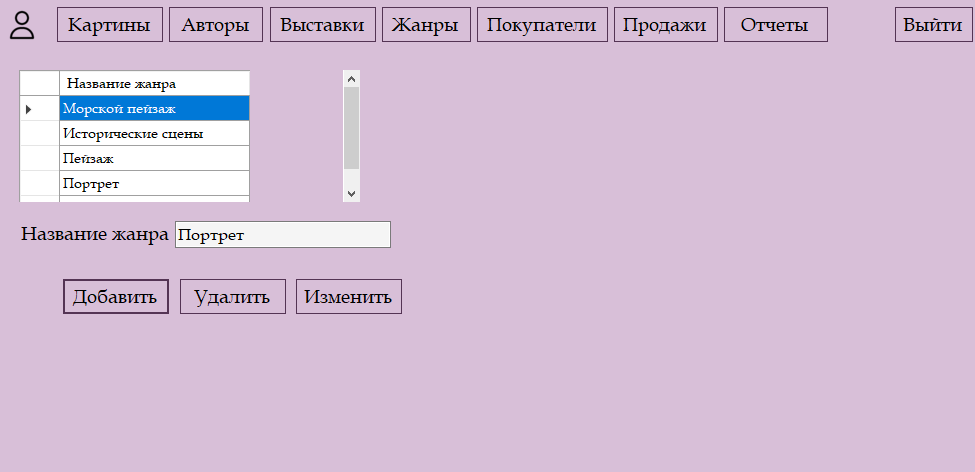


Рисунок 13 – Интерфейс для функции добавления жанра

**Программный код для добавления жанра:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "INSERT INTO janre (name\_janre) VALUES ('" + textBox\_name\_janre2.Text + "');";

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowJanre();

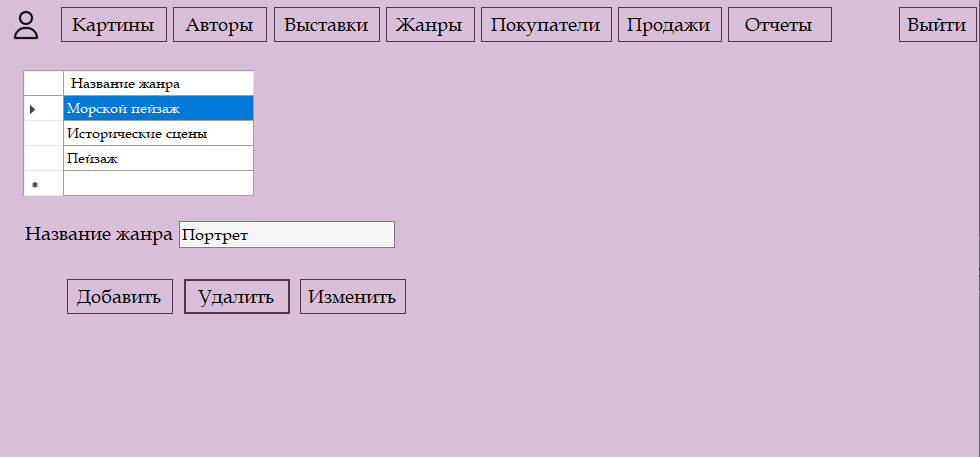
****

Рисунок 14 – Интерфейс для функции удаления жанра

**Программный код для удаления жанра:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "DELETE FROM janre WHERE id\_janre = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowJanre();

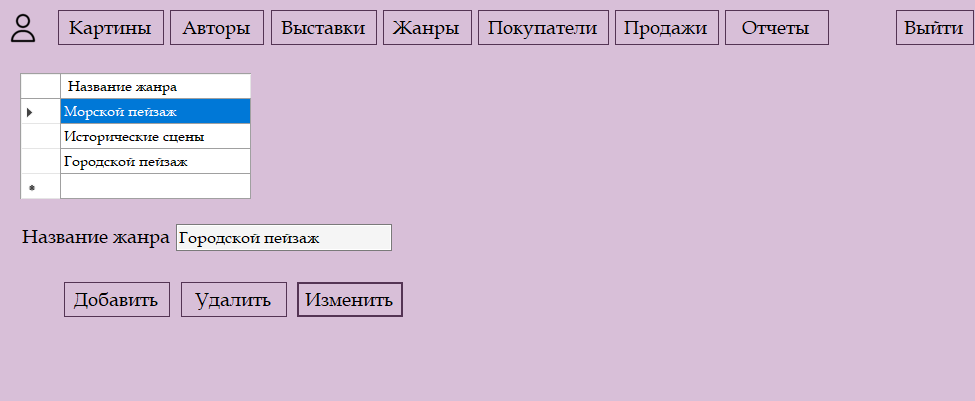
****

Рисунок 15 – Интерфейс для функции изменения жанра

**Программный код для изменения жанра:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "UPDATE janre SET name\_janre = '" + textBox\_name\_janre2.Text + "' WHERE id\_janre = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowJanre();

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации о покупателях.

В разделе «Покупатели» в таблице представлены ФИО покупателей и их контактные данные.

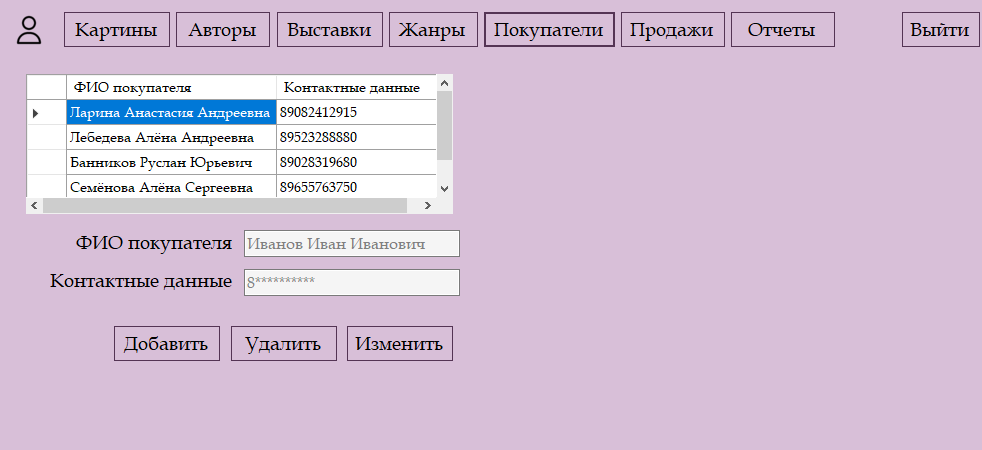


Рисунок 16 – Интерфейс для радела «Покупатели»

Для добавления покупателя необходимо ввести все значения. Для удаления нужно щёлкнуть по строке, которую нужно удалить. Для изменения необходимо щёлкнуть по нужному значению из таблицы и данные переносятся в поля.

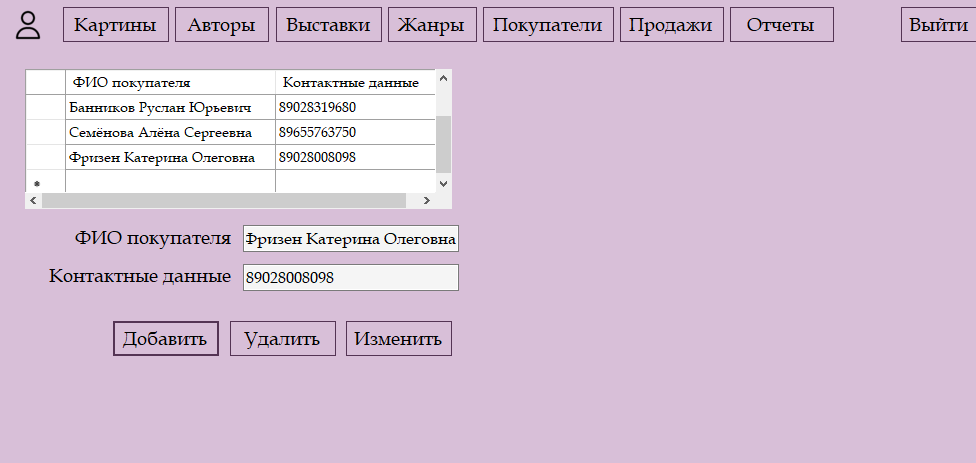


Рисунок 17 – Интерфейс для функции добавления покупателя

**Программный код для добавления покупателя:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "INSERT INTO bought\_mainer (fio\_bought\_mainer, contact\_bought\_mainer) VALUES ('" + textBox\_fio\_bought\_mainer.Text + "', '" + textBox\_contact\_bought\_mainer.Text + "');";

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowBoughtMainer();

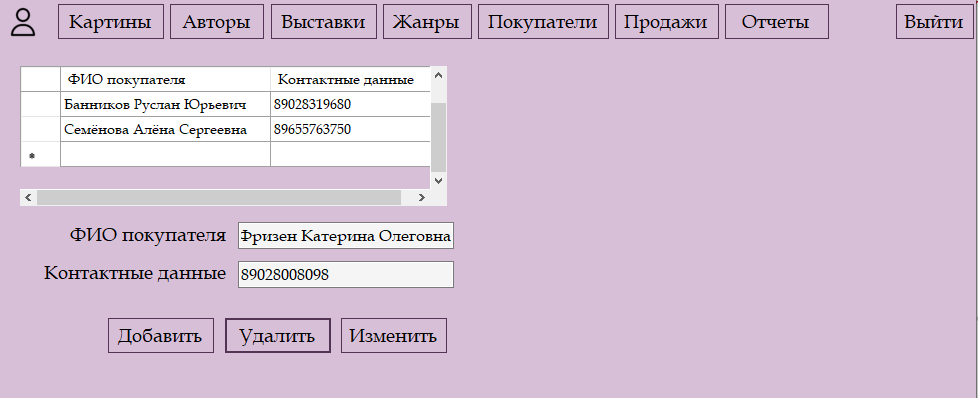


Рисунок 18 – Интерфейс для функции удаления покупателя

**Программный код для удаления покупателя:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "DELETE FROM bought\_mainer WHERE id\_bought\_mainer = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowBoughtMainer();

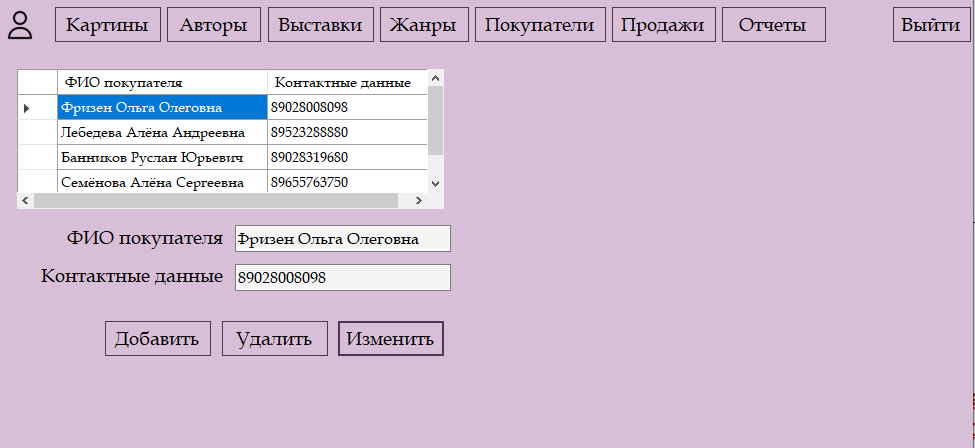


Рисунок 19 – Интерфейс для функции изменения покупателя

**Программный код для изменения покупателя:**

MySqlDataAdapter msda = new MySqlDataAdapter();

string sql = "UPDATE bought\_mainer SET fio\_bought\_mainer = '" + textBox\_fio\_bought\_mainer.Text + "', contact\_bought\_mainer = '" + textBox\_contact\_bought\_mainer.Text + "' WHERE id\_bought\_mainer = " + value;

DataTable dt = new DataTable();

db.OpenConnection();

MySqlCommand command = new MySqlCommand(sql, db.GetConnection());

msda.SelectCommand = command;

msda.Fill(dt);

db.CloseConnection();

ShowBoughtMainer();

1. В системе функция также присутствует функция с правильной рекомендацией заполнения полей(рис.20). Например, строку ФИО нужно заполнять в соответствии с правилом «Фамилия-Имя- Отчество», а контактный номер должен начинаться с 8.

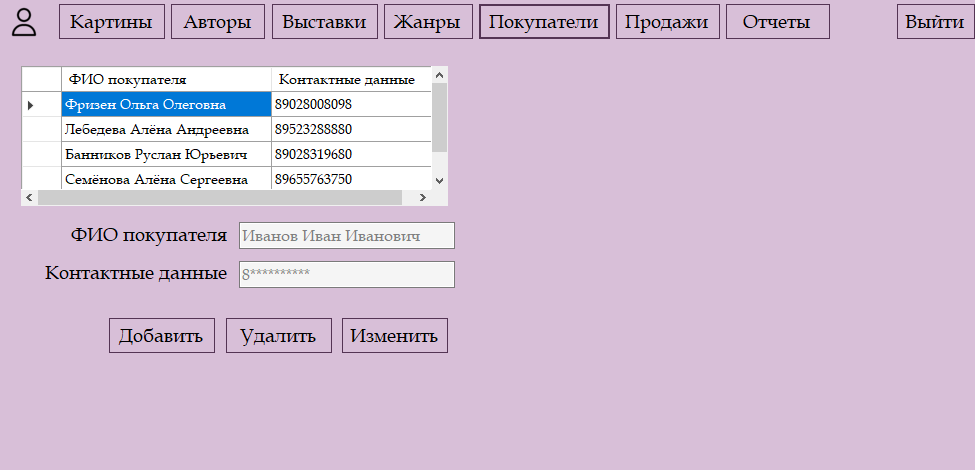


Рисунок 20 – Интерфейс с рекомендацией для правильного заполнения полей

1. В системе должна быть функция, обеспечивающая добавление, изменение и удаление информации о продажах.

В разделе «Продажи» в таблице представлены ФИО покупателя, ФИО пользователя, название картины и цена сделки.

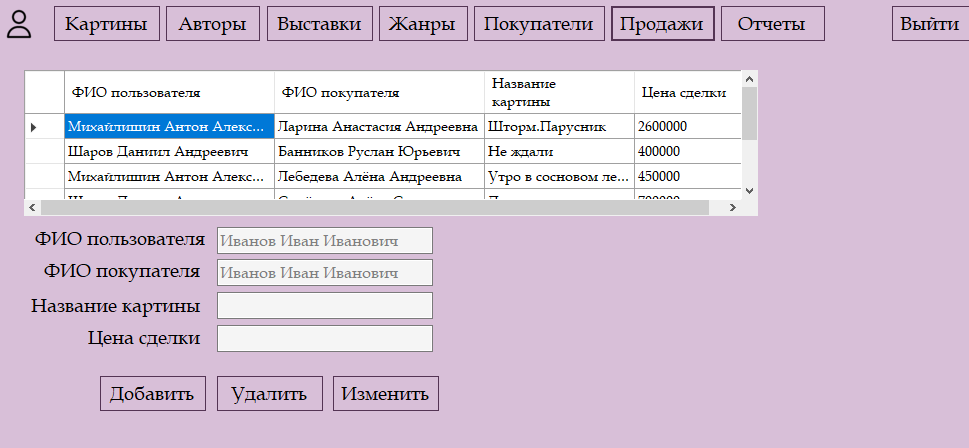


Рисунок 21 – Интерфейс для радела «Продажи»

Для добавления покупателя необходимо ввести все значения.

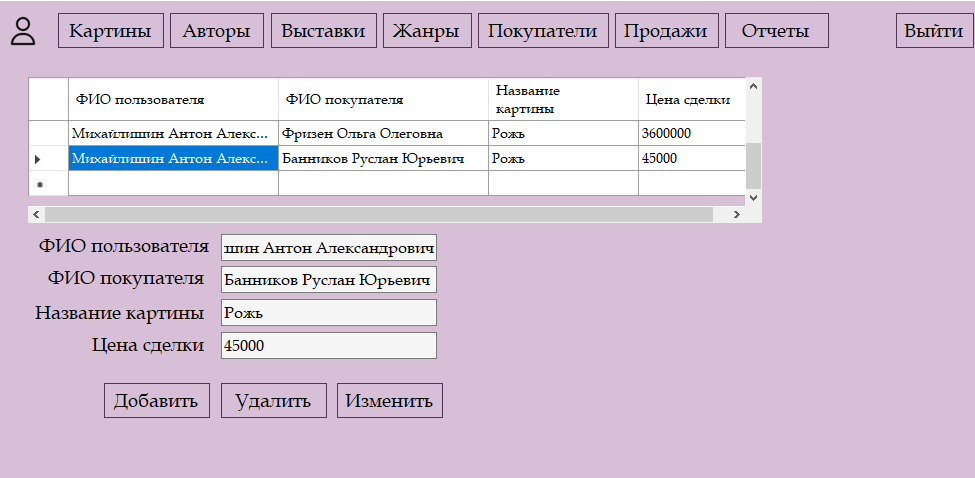


Рисунок 22 – Интерфейс для функции добавления продажи

**Программный код для добавления продажи:**

db.OpenConnection();

string procedure\_name = "insert\_rashod";

MySqlCommand comm\_Add = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Add.CommandType = CommandType.StoredProcedure;

MySqlParameter f\_u = new MySqlParameter

{

ParameterName = "f\_u",

Value = textBox\_fio\_user.Text

};

MySqlParameter f\_b\_m = new MySqlParameter

{

ParameterName = "f\_b\_m",

Value = textBox\_\_fio\_bought\_mainer2.Text

};

MySqlParameter n\_p = new MySqlParameter

{

ParameterName = "n\_p",

Value = textBox\_name\_picture2.Text

};

MySqlParameter c\_r = new MySqlParameter

{

ParameterName = "c\_r",

Value = Convert.ToInt32(textBox\_cost\_rashod.Text)//цена продажи картины

};

comm\_Add.Parameters.Add(f\_u);

comm\_Add.Parameters.Add(f\_b\_m);

comm\_Add.Parameters.Add(n\_p);

comm\_Add.Parameters.Add(c\_r);

comm\_Add.ExecuteNonQuery();

db.CloseConnection();

ShowRashod();

**Программный код для удаления продажи:**

db.OpenConnection();

string procedure\_name = "delete\_rashod";

MySqlCommand comm\_Del = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Del.CommandType = CommandType.StoredProcedure;

MySqlParameter f\_u = new MySqlParameter

{

ParameterName = "f\_u",

Value = textBox\_fio\_user.Text

};

MySqlParameter f\_b\_m = new MySqlParameter

{

ParameterName = "f\_b\_m",

Value = textBox\_\_fio\_bought\_mainer2.Text

};

MySqlParameter n\_p = new MySqlParameter

{

ParameterName = "n\_p",

Value = textBox\_name\_picture2.Text

};

MySqlParameter c\_r = new MySqlParameter

{

ParameterName = "c\_r",

Value = Convert.ToInt32(textBox\_cost\_rashod.Text)//цена продажи картины

};

comm\_Del.Parameters.Add(f\_u);

comm\_Del.Parameters.Add(f\_b\_m);

comm\_Del.Parameters.Add(n\_p);

comm\_Del.Parameters.Add(c\_r);

comm\_Del.ExecuteNonQuery();

db.CloseConnection();

ShowRashod();

**Программный код для изменения продажи:**

db.OpenConnection();

string procedure\_name = "update\_rashod";

MySqlCommand comm\_Upd = new MySqlCommand(procedure\_name, db.GetConnection());

comm\_Upd.CommandType = CommandType.StoredProcedure;

string value = dataGridView6.CurrentRow.Cells[0].Value.ToString();

MySqlParameter id\_param = new MySqlParameter

{

ParameterName = "id",

Value = value

};

MySqlParameter new\_f\_u = new MySqlParameter

{

ParameterName = "new\_f\_u",

Value = textBox\_fio\_user.Text

};

MySqlParameter new\_f\_b\_m = new MySqlParameter

{

ParameterName = "new\_f\_b\_m",

Value = textBox\_\_fio\_bought\_mainer2.Text

};

MySqlParameter new\_n\_p = new MySqlParameter

{

ParameterName = "new\_n\_p",

Value = textBox\_name\_picture2.Text

};

MySqlParameter new\_c\_r = new MySqlParameter

{

ParameterName = "new\_c\_r",

Value = Convert.ToInt32(textBox\_cost\_rashod.Text)//цена продажи картины

};

comm\_Upd.Parameters.Add(id\_param);

comm\_Upd.Parameters.Add(new\_f\_u);

comm\_Upd.Parameters.Add(new\_f\_b\_m);

comm\_Upd.Parameters.Add(new\_n\_p);

comm\_Upd.Parameters.Add(new\_c\_r);

comm\_Upd.ExecuteNonQuery();

db.CloseConnection();

ShowRashod();

1. Функция генерации отчетов – не менее пяти.

В системе присутствует 5 видов отчета (рис.23). Отчеты формируются на основе веденных данных.

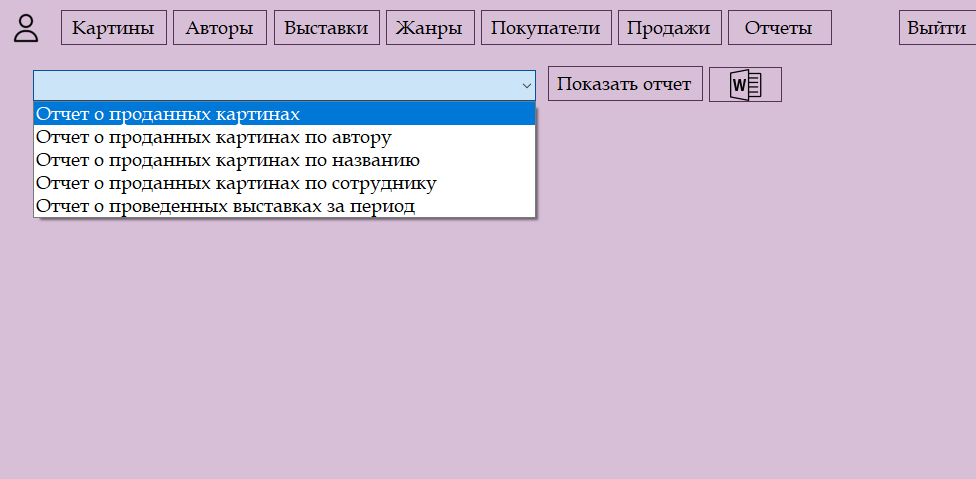


Рисунок 23 – Интерфейс для раздела «Отчеты»

**Программный код генерации отчетов:**

db.OpenConnection();

if (comboBox\_report.Text == "Отчет о проданных картинах")

{

dataGridView7.Rows.Clear();

dataGridView7.ColumnCount = 4;

dataGridView7.Columns[0].HeaderText = "ФИО пользователя";

dataGridView7.Columns[1].HeaderText = "ФИО покупателя";

dataGridView7.Columns[2].HeaderText = "Название картины";

dataGridView7.Columns[3].HeaderText = "Цена сделки";

string sql1 = "SELECT (SELECT fio\_user FROM user WHERE user.id\_user = rashod.id\_user), (SELECT fio\_bought\_mainer FROM bought\_mainer WHERE bought\_mainer.id\_bought\_mainer = rashod.id\_bought\_mainer), (SELECT name\_picture FROM picture WHERE picture.id\_picture = rashod.id\_picture), cost\_rashod FROM pic.rashod;";

MySqlCommand command1 = new MySqlCommand(sql1, db.GetConnection());

MySqlDataReader reader1 = command1.ExecuteReader();

List<string[]> data1 = new List<string[]>();

while (reader1.Read())

{

data1.Add(new string[4]);

data1[data1.Count - 1][0] = reader1[0].ToString();

data1[data1.Count - 1][1] = reader1[1].ToString();

data1[data1.Count - 1][2] = reader1[2].ToString();

data1[data1.Count - 1][3] = reader1[3].ToString();

}

reader1.Close();

db.CloseConnection();

foreach (string[] s in data1)

dataGridView7.Rows.Add(s);

}

else if (comboBox\_report.Text == "Отчет о проданных картинах по автору")

{

dataGridView7.Rows.Clear();

dataGridView7.ColumnCount = 3;

dataGridView7.Columns[0].HeaderText = "ФИО покупателя";

dataGridView7.Columns[1].HeaderText = "Название картины";

dataGridView7.Columns[2].HeaderText = "Цена сделки";

string sql1 = "SELECT (SELECT fio\_bought\_mainer FROM bought\_mainer WHERE bought\_mainer.id\_bought\_mainer = rashod.id\_bought\_mainer), (SELECT name\_picture FROM picture WHERE picture.id\_picture = rashod.id\_picture), cost\_rashod FROM pic.rashod INNER JOIN picture ON picture.id\_picture = rashod.id\_picture INNER JOIN author ON author.id\_author = picture.id\_author WHERE fio\_author='" + textBox\_fio\_author3.Text + "';";

MySqlCommand command1 = new MySqlCommand(sql1, db.GetConnection());

MySqlDataReader reader1 = command1.ExecuteReader();

List<string[]> data1 = new List<string[]>();

while (reader1.Read())

{

data1.Add(new string[3]);

data1[data1.Count - 1][0] = reader1[0].ToString();

data1[data1.Count - 1][1] = reader1[1].ToString();

data1[data1.Count - 1][2] = reader1[2].ToString();

}

reader1.Close();

db.CloseConnection();

foreach (string[] s in data1)

dataGridView7.Rows.Add(s);

}

else if (comboBox\_report.Text == "Отчет о проданных картинах по названию")

{

dataGridView7.Rows.Clear();

dataGridView7.ColumnCount = 2;

dataGridView7.Columns[0].HeaderText = "ФИО покупателя";

dataGridView7.Columns[1].HeaderText = "Цена сделки";

string sql1 = "SELECT (SELECT fio\_bought\_mainer FROM bought\_mainer WHERE bought\_mainer.id\_bought\_mainer = rashod.id\_bought\_mainer), cost\_rashod FROM pic.rashod INNER JOIN picture ON picture.id\_picture = rashod.id\_picture WHERE name\_picture = '" + textBox\_name\_picture4.Text + "';";

MySqlCommand command1 = new MySqlCommand(sql1, db.GetConnection());

MySqlDataReader reader1 = command1.ExecuteReader();

List<string[]> data1 = new List<string[]>();

while (reader1.Read())

{

data1.Add(new string[2]);

data1[data1.Count - 1][0] = reader1[0].ToString();

data1[data1.Count - 1][1] = reader1[1].ToString();

}

reader1.Close();

db.CloseConnection();

foreach (string[] s in data1)

dataGridView7.Rows.Add(s);

}

else if (comboBox\_report.Text == "Отчет о проданных картинах по сотруднику")

{

dataGridView7.Rows.Clear();

dataGridView7.ColumnCount = 2;

dataGridView7.Columns[0].HeaderText = "Название картины";

dataGridView7.Columns[1].HeaderText = "Цена сделки";

string sql1 = "SELECT (SELECT name\_picture FROM picture WHERE picture.id\_picture = rashod.id\_picture), cost\_rashod FROM pic.rashod INNER JOIN user ON user.id\_user = rashod.id\_user WHERE fio\_user = '" + textBox\_fio\_user2.Text + "';";

MySqlCommand command1 = new MySqlCommand(sql1, db.GetConnection());

MySqlDataReader reader1 = command1.ExecuteReader();

List<string[]> data1 = new List<string[]>();

while (reader1.Read())

{

data1.Add(new string[2]);

data1[data1.Count - 1][0] = reader1[0].ToString();

data1[data1.Count - 1][1] = reader1[1].ToString();

}

reader1.Close();

db.CloseConnection();

foreach (string[] s in data1)

dataGridView7.Rows.Add(s);

}

//(comboBox\_report.Text == "Отчет о проведенных выставках за период")

else

{

dataGridView7.Rows.Clear();

dataGridView7.ColumnCount = 5;

dataGridView7.Columns[0].HeaderText = "Название выставки";

dataGridView7.Columns[1].HeaderText = "Кол-во проданных билетов";

dataGridView7.Columns[2].HeaderText = "Цена билета";

dataGridView7.Columns[3].HeaderText = "Сумма";

dataGridView7.Columns[4].HeaderText = "Название музея";

string sql1 = "SELECT name\_exhibitions, sold\_places, price\_exhibitions, (sold\_places\*price\_exhibitions),(SELECT name\_museum FROM museum WHERE museum.id\_museum = exhibitions.id\_museum) FROM exhibitions WHERE date\_start = '" + textBox\_date\_ot.Text.ToString() + "' AND date\_end = '" + textBox\_date\_do.Text.ToString() + "';";

MySqlCommand command1 = new MySqlCommand(sql1, db.GetConnection());

MySqlDataReader reader1 = command1.ExecuteReader();

List<string[]> data1 = new List<string[]>();

while (reader1.Read())

{

data1.Add(new string[5]);

data1[data1.Count - 1][0] = reader1[0].ToString();

data1[data1.Count - 1][1] = reader1[1].ToString();

data1[data1.Count - 1][2] = reader1[2].ToString();

data1[data1.Count - 1][3] = reader1[3].ToString();

data1[data1.Count - 1][4] = reader1[4].ToString();

}

reader1.Close();

db.CloseConnection();

foreach (string[] s in data1)

dataGridView7.Rows.Add(s);

}

}

Отчет о проданных картинах представлен на рисунке 24.

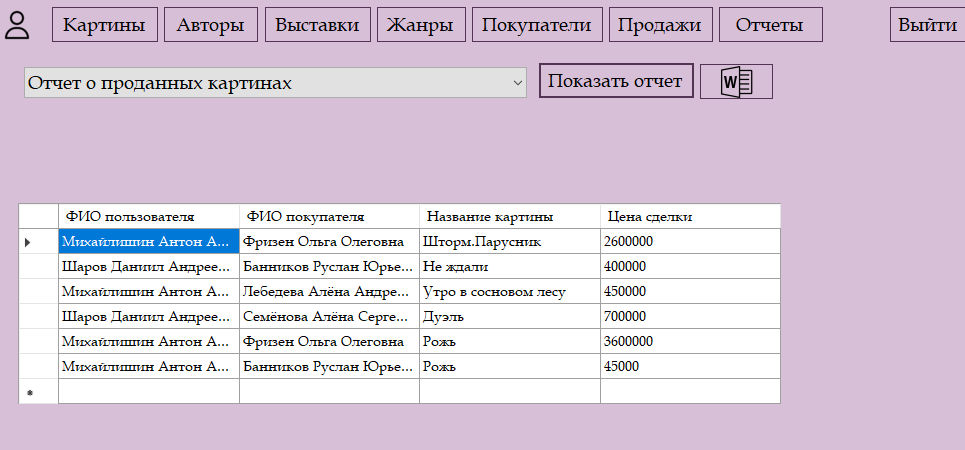


Рисунок 24 – Отчет по проданным картинам

Отчет о проданных картинах по автору на рисунке 25. Для этого отчета обязательно должно быть выбрано ФИО автора.

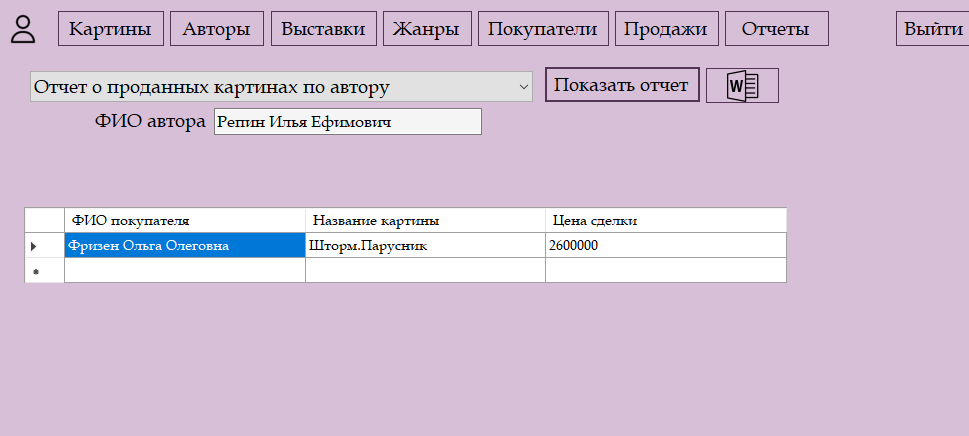


Рисунок 25 – Отчет о проданных картинах по автору

Отчет о проданных картинах по названию на рисунке 26. Для этого отчета обязательно должно быть выбрано название картины.

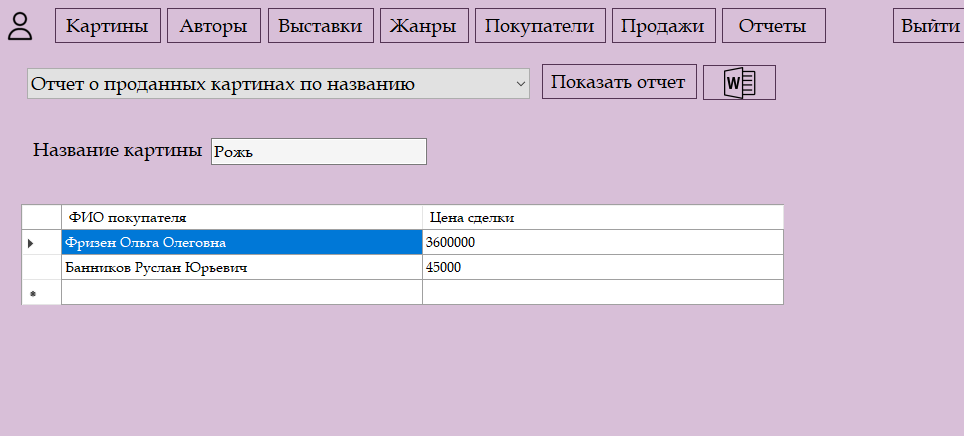


Рисунок 26 – Отчет о проданных картинах по названию

Отчет о проданных картинах по сотруднику на рисунке 27. Для этого отчета обязательно должно быть выбрано ФИО пользователя.

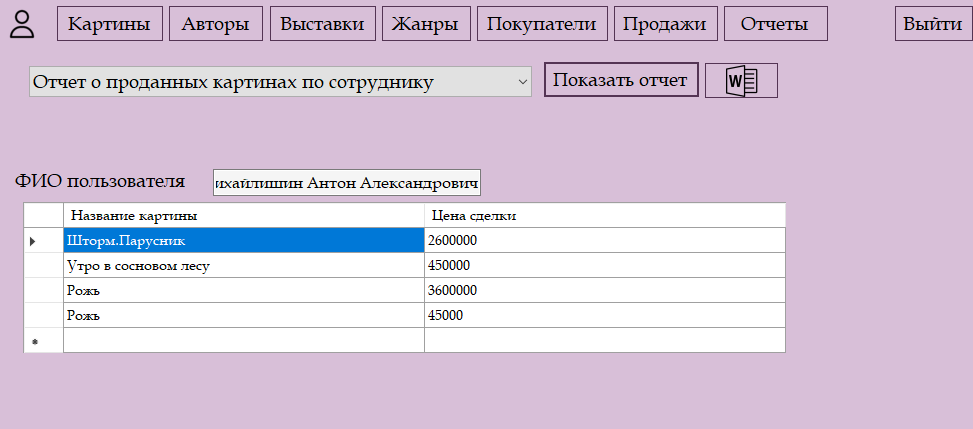


Рисунок 27 – Отчет о проданных картинах по сотруднику

Отчет о проведенных выставках за период на рисунке 28. Для этого отчета обязательно должны быть выбраны дата начала и дата окончания выставки.

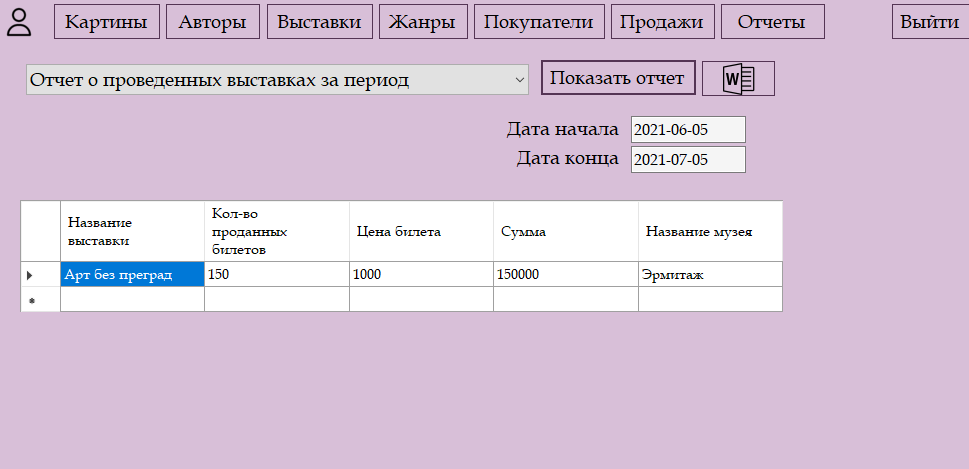


Рисунок 28 – Отчет о проведенных выставках за период

1. Формирование отчетов в формате файла doc.

Все отчеты формируются в формате doc. Для этого нужно выбрать название отчета, ФИО автора (если отчет о проданных картинах по автору). Откроется путь, сформируется имя отчета: «Название отчета» + ФИО автора (рис.29-31).

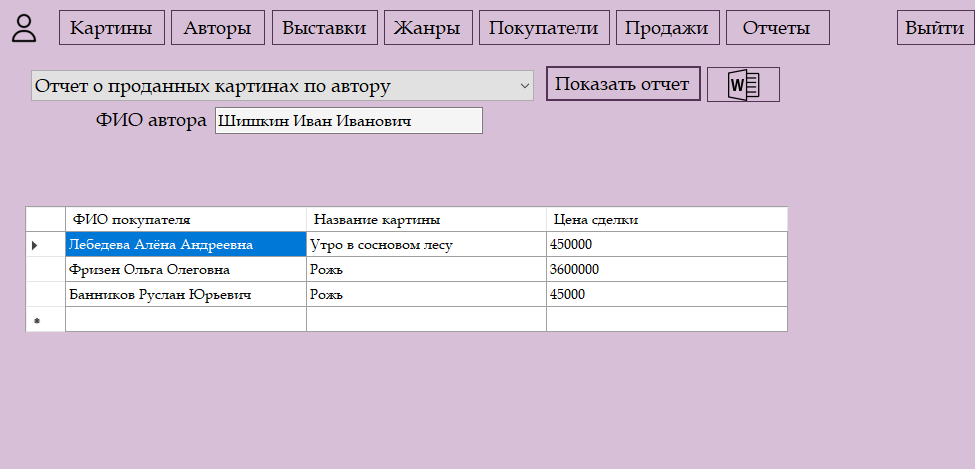


Рисунок 29 – Отчет о проданных картинах по автору

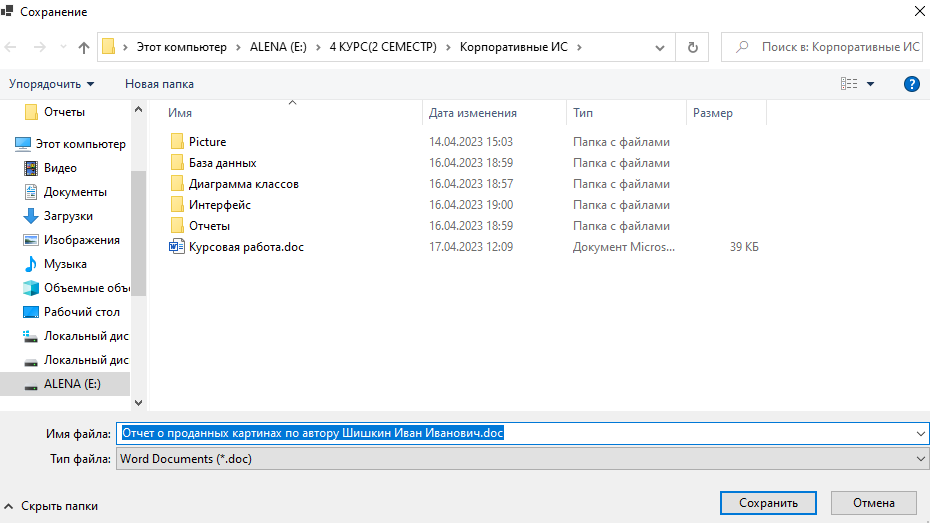


Рисунок 30 – Сохранение отчета

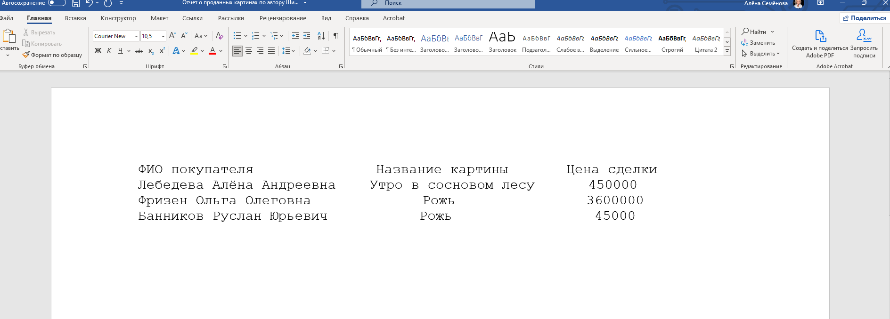


Рисунок 30 – Просмотр отчета в формате doc

**Программный код формирования отчета в doc:**

SaveFileDialog sfd = new SaveFileDialog();

sfd.Filter = "Word Documents (\*.doc)|\*.doc";

sfd.FileName = file\_name;

if (sfd.ShowDialog() == DialogResult.OK)

{

Export\_Data\_To\_Word(dataGridView7, sfd.FileName);

}

}

public async void Export\_Data\_To\_Word(DataGridView DGV, string filename)

{

string stOutput = "";

string sHeaders = "";

for (int j = 0; j < DGV.Columns.Count; j++)

sHeaders = sHeaders.ToString() + Convert.ToString(DGV.Columns[j].HeaderText) + "\t";

stOutput += sHeaders + "\r\n";

for (int i = 0; i < DGV.RowCount - 1; i++)

{

string stLine = "";

for (int j = 0; j < DGV.Rows[i].Cells.Count; j++)

stLine = stLine.ToString() + Convert.ToString(DGV.Rows[i].Cells[j].Value) + "\t";

stOutput += stLine + "\r\n";

}

using (FileStream fs = new FileStream(filename, FileMode.Create))

{

byte[] text = Encoding.Default.GetBytes(stOutput);

await fs.WriteAsync(text, 0, text.Length);

}

1. Реакция системы на ошибочный ввод.

При неправильном вводе выдается сообщение при аутентификации пользователя (рис.31).

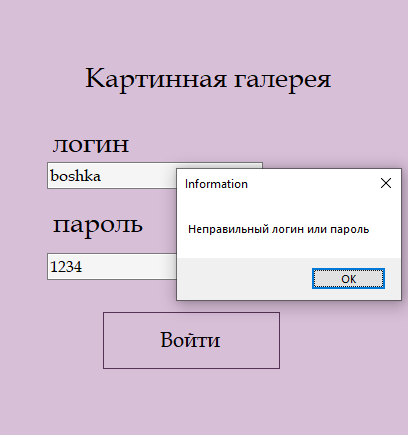


Рисунок 31 – Интерфейс реакция на ошибку в аутентификации

Также при неправильном вводе данных появляется вывод отдельного окна с надписью о том, какой формат должен быть в поле (рис.32).

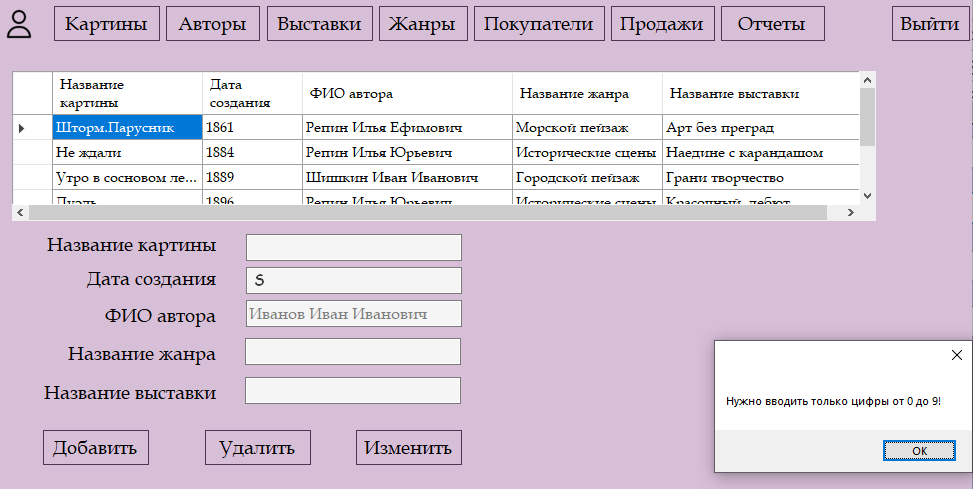


Рисунок 32 – Интерфейс реакция на ошибку в типе данных

**Программный код реакции на ошибочный ввод:**

char number = e.KeyChar;

if (Char.IsDigit(number) || (number == (char)Keys.Back))

{

return;

}

else

{

e.Handled = true;

MessageBox.Show("Нужно вводить только цифры от 0 до 9!");

textBox\_price\_exhibitions.Focus();

}

}

При неправильном вводе выдается сообщение о том, что превышено количество символов в поле. (рис.33).

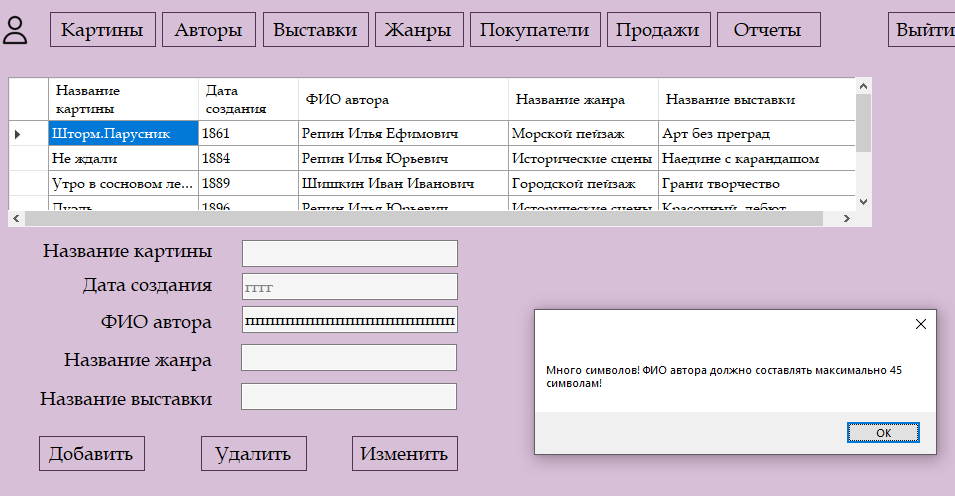


Рисунок 33 – Интерфейс реакция на ошибку количества символов

**Программный код реакции на ошибочный ввод:**

if (textBox\_fio\_user.Text.Length > 45)

{

MessageBox.Show("Много символов! ФИО пользователя должно составлять максимально 45 символам!");

textBox\_fio\_user.Clear();

textBox\_fio\_user.Focus();

}

}

1. **Выводы**

В данной лабораторной работе была разработана программа информационной системы Картинная галерея. Реализован основной функционал: аутентификация, вид под разными пользователями (в зависимости от роли), редактировании базы данных, формирование отчетов, реакция системы на ошибочный ввод.