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»

**ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИС**

по дисциплине

**«КОРПОРАТИВНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ»**

Выполнил:

студент гр. ИСТ-19-2б

Рачев Р.И.

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

Функции, которые должны быть реализованы в системе:

* 1. Функция авторизации

Внешний интерфейс для функции авторизации представлен на рисунке 1. В данном окне пользователю необходимо ввести логин и пароль для того, чтобы войти в приложение.

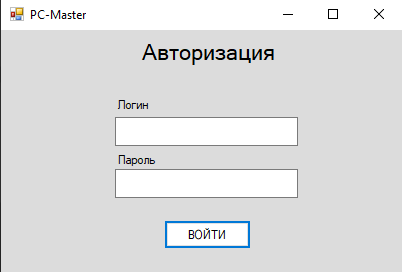


Рисунок 1 – Интерфейс функции авторизации

Программный код для аутентификации с помощью запроса к базе данных:

private bool checkPassword()

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "Select \* from employee WHERE employee\_login='"+loginInput.Text+"' AND employee\_password='"+passwordInput.Text+"'";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

while (reader.Read())

{

name = Convert.ToString(reader.GetValue(1));

role = Convert.ToString(reader.GetValue(4));

}

}

}

Console.WriteLine(name);

conn.Close();

conn.Dispose();

if (name == "") return false;

else return true;

}

* 1. Функция для просмотра записей в базе данных

В системе должна быть функция для просмотра данных в базе по всем товарам, продажам, закупкам, поставщикам и сотрудникам магазина. Интерфейс просмотра справочника представлен на рисунке 2.

Программный код для отображения базы:

private void queryTable(String table\_name)

{

itemId = 0;

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

table\_name = translateTable(table\_name);

string sql = "Select \* from " + table\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = translateColumn(col.Field<String>("ColumnName"));

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

columns.Add(col.Field<String>("ColumnName"));

count++;

}

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.None;

for (int i = 0; i < count; i++)

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

dataGridView1.Columns[0].Width -= 50;

dataGridView1.Columns[1].Width += 50;

dataGridView1.Columns[count - 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime) reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

showControl();

}

* 1. Функция добавления, изменения и удаления записей из базы данных

В системе должна быть функция, обеспечивающая добавление, изменение и удаление записей в каждой таблице. При закупке нового товара или при устройстве нового сотрудника, его данные должны быть добавлены в базу, также и при его увольнении и прочих событиях данные должны быть актуализированы.

Интерфейс ИС «PC-Master» для добавления, удаления и изменения записей в базе данных (рис. 2).

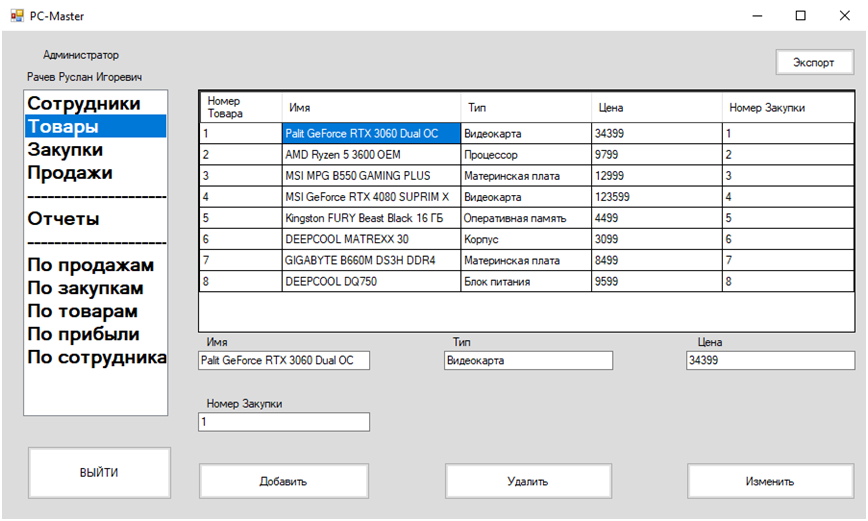


Рисунок 2 – Пример интерфейса для раздела «Товары»

Для добавления записи в базу необходимо ввести данные во все предложенные поля, после чего нажать кнопку «Добавить». После этого отображаемая таблица обновится и в ней появится новая строка.

Программный код для добавления записи:

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

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

String clmns = "";

for (int i = 1; i < columns.Count; i++)

{

clmns += columns[i] + ", ";

}

clmns = clmns.Substring(0, clmns.Length - 2);

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 0; i < columns.Count - 1; i++)

{

text += "'" + textBoxes[i].Text + "', ";

}

text = text.Substring(0, text.Length - 2);

string sql = "Insert into " + table\_name + " (" + clmns + ") values (" + text + ")";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка добавлена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

conn.Dispose();

queryTable(table\_name);

}

Для удаления нужно щёлкнуть по необходимой строке и нажать на кнопку «Удалить». После этого отображаемая таблица обновится и из нее исчезнет удаленная строка.

Программный код для удаления записи:

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "DELETE FROM " + table\_name + " WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка удалена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

Для изменения строки таблицы, необходимо кликнуть на необходимую строку, данные ее столбцов отобразятся в полях снизу. После этого нужно изменить данные строки на необходимые данные и нажать кнопку «Изменить».

Программный код для изменения записи:

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 1; i < columns.Count; i++)

{

text += columns[i] + "='" + textBoxes[i - 1].Text + "',";

}

text = text.Substring(0, text.Length - 2);

string sql = "UPDATE " + table\_name + " SET " + text +

" WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка изменена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

* 1. Функция генерации отчетов

В системе присутствует 5 видов отчета (рис.3). Каждый из них предназначен для отображения своего набора данных.

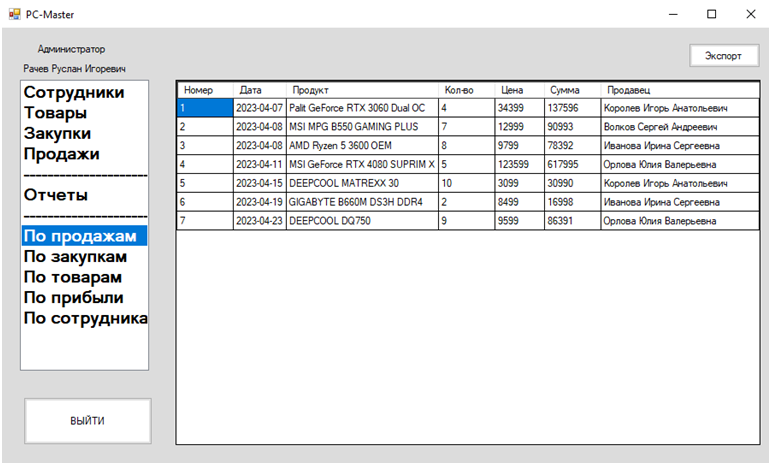


Рисунок 3 – Пример интерфейса для отображения отчета

Отчет по продажам предоставляет данные по всем совершенным продажам.

SQL-код для отчета по продажам:

CREATE VIEW sales\_report

as SELECT

sale.sale\_id as "Номер",

sale.sale\_date as "Дата",

product.product\_name as "Продукт",

sale.sale\_count as "Кол-во",

product.product\_price as "Цена",

(sale.sale\_count\*product.product\_price) as "Сумма",

employee.employee\_name as "Продавец"

FROM sale

JOIN product ON sale.product\_id = product.product\_id

JOIN employee ON sale.employee\_id = employee.employee\_id;

Отчет по закупкам предоставляет данные по всем совершенным закупкам.

SQL-код для отчета по закупкам:

CREATE VIEW supplies\_report

as SELECT

supply.supply\_id as "Номер",

supply.supply\_date as "Дата",

product.product\_name as "Продукт",

supply.supply\_count as "Кол-во",

REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", "") as "Цена закупки",

supply.supply\_price as "Сумма",

supplier.supplier\_name as "Поставщик",

supplier.supplier\_city as "Город"

FROM supply

JOIN product ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

ORDER BY supply.supply\_date;

Отчет по товарам предоставляет данные по всем товарам на складе.

SQL-код для отчета по товарам:

CREATE VIEW products\_report

as SELECT

product.product\_id as "Номер",

product.product\_name as "Продукт",

product.product\_type as "Тип",

supplier.supplier\_name as "Поставщик",

supply.supply\_date as "Дата поставки",

supply.supply\_count as "Кол-во",

SUM(sale.sale\_count) as "Продано",

(supply.supply\_count-SUM(sale.sale\_count)) as "На складе"

FROM product

JOIN supply ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

JOIN sale ON sale.product\_id = product.product\_id

GROUP BY product.product\_id, supply.supply\_id, supplier.supplier\_id, sale.sale\_id;

Отчет по прибыли предоставляет данные по прибыли, полученной с продаж каждого товара.

SQL-код для отчета по прибыли:

CREATE VIEW profits\_report

as SELECT

product.product\_id as "Номер",

product.product\_name as "Продукт",

product.product\_type as "Тип",

REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", "") as "Закупочная цена",

product.product\_price as "Розничная цена",

SUM(sale.sale\_count) as "Продано",

(SUM(sale.sale\_count)\*product.product\_price) as "Сумма продаж",

(SUM(sale.sale\_count)\*(product.product\_price-REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", ""))) as "Прибыль"

FROM product

JOIN supply ON supply.supply\_id = product.supply\_id

JOIN supplier ON supply.supplier\_id = supplier.supplier\_id

JOIN sale ON sale.product\_id = product.product\_id

GROUP BY product.product\_id, supply.supply\_id, supplier.supplier\_id, sale.sale\_id;

Отчет по сотрудникам предоставляет данные обо всех работниках магазина.

SQL-код для отчета по сотрудникам:

CREATE VIEW employees\_report as

SELECT

abc.employee\_id as "Номер",

abc.employee\_name as "ФИО",

SUM(abc.summ\_count) as "Кол-во продаж",

SUM(abc.summ\_price) as "Сумма продаж",

SUM(abc.summ\_profit) as "Заработал"

from(

SELECT

employee.employee\_id,

employee.employee\_name,

SUM(sale.sale\_count) as summ\_count,

(SUM(sale.sale\_count)\*product.product\_price) as summ\_price,

(SUM(sale.sale\_count)\*(product.product\_price-REPLACE(FORMAT((supply.supply\_price/supply.supply\_count), '######'), ",", ""))) as summ\_profit

FROM employee

JOIN sale ON sale.employee\_id = employee.employee\_id

JOIN product ON sale.product\_id = product.product\_id

JOIN supply ON supply.supply\_id = product.supply\_id

GROUP BY product.product\_id, sale.sale\_id, supply.supply\_id, supply.supply\_price, supply.supply\_count, employee.employee\_id) as abc

GROUP BY abc.employee\_id;

Каждый из разработанных отчетов необходимо отображать в приложении.

Программный код для отображения отчетов:

private void queryReport(String report\_name)

{

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

report\_name = translateReport(report\_name);

string sql = "Select \* from " + report\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = col.Field<String>("ColumnName");

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

count++;

}

for (int i = 0; i < count; i++)

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

if (report\_name == "sales\_report" || report\_name == "employees\_report")

dataGridView1.Columns[count - 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime)reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

}

* 1. Функция выгрузки отчетов в формат Excel

В программе должна быть функция выгрузки отчета в формат xlsx. Для переноса необходимого отчета необходимо нажать кнопку «Экспорт».

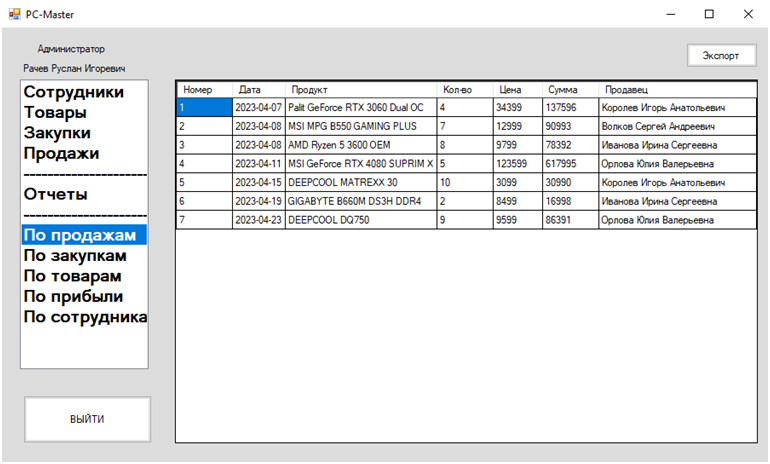


Рисунок 4 – Интерфейс для выгрузки отчета

Программный код для выгрузки отчетов:

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

{

Excel.Application ExcelApp = new Excel.Application();

Excel.Workbook ExcelWorkBook;

Excel.Worksheet ExcelWorkSheet;

ExcelWorkBook = ExcelApp.Workbooks.Add(System.Reflection.Missing.Value);

ExcelWorkSheet = (Excel.Worksheet)ExcelWorkBook.Worksheets.get\_Item(1);

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

ExcelApp.Cells[1, j + 1] = dataGridView1.Columns[j].HeaderText.ToString();

ExcelApp.Cells[i + 2, j + 1] = dataGridView1.Rows[i].Cells[j].Value;

}

}

ExcelApp.Visible = true;

ExcelApp.UserControl = true;

ExcelApp.Rows.AutoFit();

ExcelApp.Columns.AutoFit();

}

После выгрузки открывается Excel и отчет автоматически копируется в новый документ (рис.5). После этого, сделав все необходимые изменения, пользователь может сохранить отчет в формате xlsx.

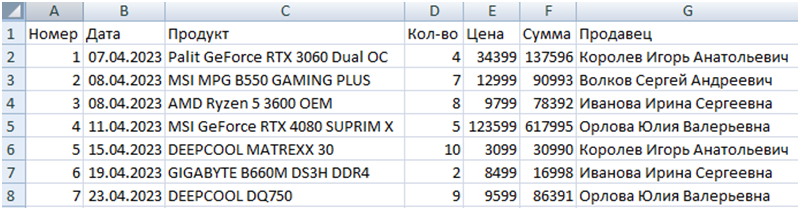


Рисунок 5 – Просмотр отчета в Excel

* 1. Реакция системы на ошибочный ввод.

При неправильном вводе выдается сообщение при аутентификации пользователя (рис.6), при не правильном типе данных выводится надпись (рис.7).

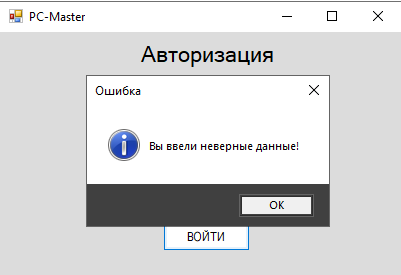


Рисунок 6 – Вывод ошибки при авторизации

Программный код реакции на ошибку при авторизации:

if (loginInput.Text != "" && passwordInput.Text != "")

{

bool userExists = checkPassword();

if (userExists)

{

loginInput.Text = "";

passwordInput.Text = "";

this.Hide();

Main mainForm = new Main(name, role, this);

mainForm.Show();

}

else

{

MessageBox.Show("Вы ввели неверные данные!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

}

else

{

MessageBox.Show("Введите логин и пароль!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

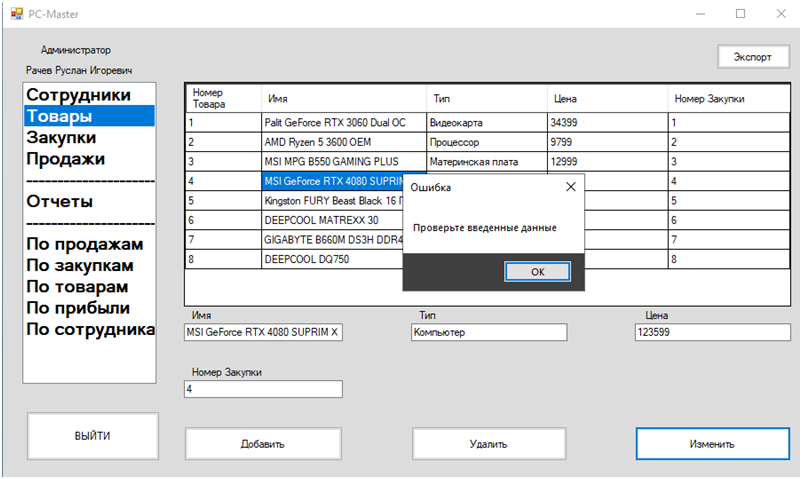


Рисунок 7 – Вывод ошибки в типе данных

Программный код реакции на ошибку в типе данных:

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка добавлена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

1. Выводы

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

Приложение 1

Код класса Autorization:

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.Common;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace PC\_Master

{

public partial class Authorization : Form

{

String name = "";

String role = "";

public Authorization()

{

InitializeComponent();

}

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

{

if (loginInput.Text != "" && passwordInput.Text != "")

{

bool userExists = checkPassword();

if (userExists)

{

loginInput.Text = "";

passwordInput.Text = "";

this.Hide();

Main mainForm = new Main(name, role, this);

mainForm.Show();

}

else

{

MessageBox.Show("Вы ввели неверные данные!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

}

else

{

MessageBox.Show("Введите логин и пароль!",

"Ошибка",

MessageBoxButtons.OK,

MessageBoxIcon.Information,

MessageBoxDefaultButton.Button1,

MessageBoxOptions.DefaultDesktopOnly);

}

}

private bool checkPassword()

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "Select \* from employee WHERE employee\_login='"+loginInput.Text+"' AND employee\_password='"+passwordInput.Text+"'";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

while (reader.Read())

{

name = Convert.ToString(reader.GetValue(1));

role = Convert.ToString(reader.GetValue(4));

}

}

}

Console.WriteLine(name);

conn.Close();

conn.Dispose();

if (name == "") return false;

else return true;

}

}

}

Код класса Main:

using MySql.Data.MySqlClient;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.Common;

using System.Windows.Forms;

using Excel = Microsoft.Office.Interop.Excel;

namespace PC\_Master

{

public partial class Main : Form

{

List<string> columns = new List<string>();

Form authForm;

int itemId = 0;

int rows = 0;

public Main(String name, String role, Form Auth)

{

InitializeComponent();

label1.Text = role;

label2.Text = name;

authForm = Auth;

int tableCount = tableChoice.Items.Count;

if (role != "Администратор")

{

for (int i = 4; i < tableCount; i++)

{

tableChoice.Items.Remove(tableChoice.Items[tableChoice.Items.Count-1]);

}

tableChoice.Items.Remove(tableChoice.Items[0]);

}

}

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

{

if (tableChoice.SelectedIndex < 4)

{

dataGridView1.Height = 253;

queryTable(tableChoice.SelectedItem.ToString());

}

if (tableChoice.SelectedIndex > 6)

{

hideControl();

dataGridView1.Height = 427;

queryReport(tableChoice.SelectedItem.ToString());

}

}

private void queryTable(String table\_name)

{

itemId = 0;

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

table\_name = translateTable(table\_name);

string sql = "Select \* from " + table\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = translateColumn(col.Field<String>("ColumnName"));

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

columns.Add(col.Field<String>("ColumnName"));

count++;

}

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.None;

for (int i = 0; i < count; i++)

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

dataGridView1.Columns[0].Width -= 50;

dataGridView1.Columns[1].Width += 50;

dataGridView1.Columns[count - 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime) reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

showControl();

}

private void queryReport(String report\_name)

{

dataGridView1.Columns.Clear();

dataGridView1.Rows.Clear();

columns.Clear();

clearTextBoxes();

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

report\_name = translateReport(report\_name);

string sql = "Select \* from " + report\_name;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

columns = new List<string>();

using (DbDataReader reader = cmd.ExecuteReader())

{

if (reader.HasRows)

{

int count = 0;

DataTable schema = null;

schema = reader.GetSchemaTable();

foreach (DataRow col in schema.Rows)

{

string сname = col.Field<String>("ColumnName");

dataGridView1.Columns.Add(col.Field<String>("ColumnName").ToUpper(), сname);

count++;

}

for (int i = 0; i < count; i++)

dataGridView1.Columns[i].Width = (int)(dataGridView1.Width \* 1 / count);

if (report\_name == "sales\_report" || report\_name == "employees\_report")

dataGridView1.Columns[count - 1].AutoSizeMode = DataGridViewAutoSizeColumnMode.Fill;

dataGridView1.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.AllCells;

int row = 0;

while (reader.Read())

{

dataGridView1.Rows.Add();

for (int column = 0; column < count; column++)

{

string value = "";

if (reader.GetValue(column).GetType() == typeof(DateTime))

{

DateTime time = (DateTime)reader.GetValue(column);

value = time.ToString("yyyy-MM-dd");

}

else value = Convert.ToString(reader.GetValue(column));

dataGridView1.Rows[row].Cells[column].Value = value;

}

row++;

}

rows = row;

}

}

conn.Close();

conn.Dispose();

}

private string translateTable(String name)

{

if (name == "Сотрудники") name = "employee";

if (name == "Товары") name = "product";

if (name == "Продажи") name = "sale";

if (name == "Закупки") name = "supply";

if (name == "Поставщики") name = "supplier";

return name;

}

private string translateColumn(String name)

{

if (name == "employee\_id") name = "Номер Сотрудника";

if (name == "product\_id") name = "Номер Товара";

if (name == "sale\_id") name = "Номер Продажи";

if (name == "supply\_id") name = "Номер Закупки";

if (name == "supplier\_id") name = "Номер Поставщика";

if (name.EndsWith("\_name")) name = "Имя";

if (name.EndsWith("\_login")) name = "Логин";

if (name.EndsWith("\_password")) name = "Пароль";

if (name.EndsWith("\_role")) name = "Роль";

if (name.EndsWith("\_date")) name = "Дата";

if (name.EndsWith("\_price")) name = "Цена";

if (name.EndsWith("\_count")) name = "Количество";

if (name.EndsWith("\_type")) name = "Тип";

return name;

}

private string translateReport(String name)

{

if (name == "По продажам") name = "sales\_report";

if (name == "По закупкам") name = "supplies\_report";

if (name == "По товарам") name = "products\_report";

if (name == "По прибыли") name = "profits\_report";

if (name == "По сотрудникам") name = "employees\_report";

return name;

}

private void showControl()

{

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

Label[] labels = new Label[6] { column1, column2, column3, column4, column5, column6 };

for (int i = 0; i < columns.Count-1; i++)

{

textBoxes[i].Visible = true;

labels[i].Visible = true;

labels[i].Text = translateColumn(columns[i+1]);

}

btnDelete.Visible = true;

btnAdd.Visible = true;

btnChange.Visible = true;

}

private void hideControl()

{

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

Label[] labels = new Label[6] { column1, column2, column3, column4, column5, column6 };

for (int i = 0; i < 6; i++)

{

textBoxes[i].Visible = false;

textBoxes[i].Text = "";

labels[i].Visible = false;

labels[i].Text = "";

}

btnDelete.Visible = false;

btnAdd.Visible = false;

btnChange.Visible = false;

}

private void clearTextBoxes()

{

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

Label[] labels = new Label[6] { column1, column2, column3, column4, column5, column6 };

for (int i = 0; i < 6; i++)

{

textBoxes[i].Visible = false;

textBoxes[i].Text = "";

labels[i].Text = "";

}

}

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

{

Int32.TryParse(dataGridView1.Rows[e.RowIndex].Cells[0].Value.ToString(), out itemId);

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 0; i < columns.Count-1; i++)

textBoxes[i].Text = dataGridView1.Rows[e.RowIndex].Cells[i+1].Value.ToString();

}

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

{

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

String clmns = "";

for (int i = 1; i < columns.Count; i++)

{

clmns += columns[i] + ", ";

}

clmns = clmns.Substring(0, clmns.Length - 2);

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 0; i < columns.Count - 1; i++)

{

text += "'" + textBoxes[i].Text + "', ";

}

text = text.Substring(0, text.Length - 2);

string sql = "Insert into " + table\_name + " (" + clmns + ") values (" + text + ")";

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка добавлена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

conn.Dispose();

queryTable(table\_name);

}

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

String text = "";

TextBox[] textBoxes = new TextBox[6] { textBox1, textBox2, textBox3, textBox4, textBox5, textBox6 };

for (int i = 1; i < columns.Count; i++)

{

text += columns[i] + "='" + textBoxes[i - 1].Text + "', ";

}

text = text.Substring(0, text.Length - 2);

string sql = "UPDATE " + table\_name + " SET " + text +

" WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка изменена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

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

{

if (itemId != 0)

{

String table\_name = translateTable(tableChoice.SelectedItem.ToString());

MySqlConnection conn = DBConnection.GetDBConnection();

conn.Open();

string sql = "DELETE FROM " + table\_name + " WHERE " + columns[0] + "=" + itemId;

MySqlCommand cmd = new MySqlCommand();

cmd.Connection = conn;

cmd.CommandText = sql;

try

{

cmd.ExecuteNonQuery();

MessageBox.Show("Строка удалена!", "Message");

}

catch (Exception)

{

MessageBox.Show("Проверьте введенные данные", "Ошибка");

}

conn.Close();

queryTable(table\_name);

} else MessageBox.Show("Ни одна строка не выбрана!", "Ошибка");

}

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

{

Excel.Application ExcelApp = new Excel.Application();

Excel.Workbook ExcelWorkBook;

Excel.Worksheet ExcelWorkSheet;

ExcelWorkBook = ExcelApp.Workbooks.Add(System.Reflection.Missing.Value);

ExcelWorkSheet = (Excel.Worksheet)ExcelWorkBook.Worksheets.get\_Item(1);

for (int i = 0; i < dataGridView1.Rows.Count; i++)

{

for (int j = 0; j < dataGridView1.ColumnCount; j++)

{

ExcelApp.Cells[1, j + 1] = dataGridView1.Columns[j].HeaderText.ToString();

ExcelApp.Cells[i + 1, j + 1] = dataGridView1.Rows[i].Cells[j].Value;

}

}

ExcelApp.Visible = true;

ExcelApp.UserControl = true;

ExcelApp.Rows.AutoFit();

ExcelApp.Columns.AutoFit();

}

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

{

this.Close();

authForm.Show();

}

}

}