Ростовская Елизавета, ПР23/2

Лабораторная работа №2

Тема: «Разработка модульной структуры проекта

(диаграммы модулей)»

Цель: научиться разрабатывать Windows Forms приложения в Visual Studio и разрабатывать в них несколько функциональных модулей.

Ход работы

1. Создайте базу данных «sklad».
2. В базе данных «sklad» создайте таблицу со следующей структурой:

CREATE TABLE products (

id INT PRIMARY KEY,

name NVARCHAR(100),

stillage INT;

cell INT;

quantity INT

);

1. Добавьте данные в таблицу:

INSERT INTO products (id, name, stillage, cell, quantity) VALUES (1, 'Хлеб', 2, 3, 20);

INSERT INTO products (id, name, stillage, cell, quantity) VALUES (2, 'Вода', 1, 2, 15);

INSERT INTO products (id, name, stillage, cell, quantity) VALUES (3, 'Сладости', 2, 1, 100);

INSERT INTO products (id, name, stillage, cell, quantity) VALUES (3, 'Газировка', 20, 100, 15);

1. Создайте новый проект SkladApp в Visual Studio. Переименуйте форму Form1.cs в Sklad.cs.
2. Добавьте все необходимые элементы на форму.
3. В первом модуле реализовать возможность изменения названия продуктов. А также возможность изменить место хранения продуктов и их количество. Реализовать удаление выбранных продуктов. А также сохранение существующих продуктов в файл.
4. Во втором модуле реализовать поиск продуктов по названию.
5. В третьем модуле реализовать поиск продуктов по координатам (стеллажу и ячейке).
6. Нарисовать диаграмму модулей реализованного приложения.

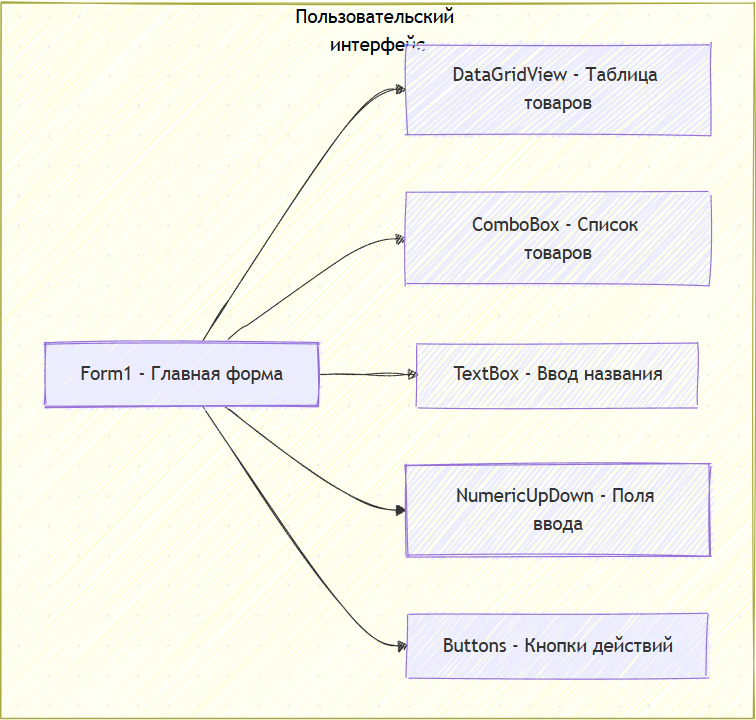


Рис. 1. Диаграмма модулей

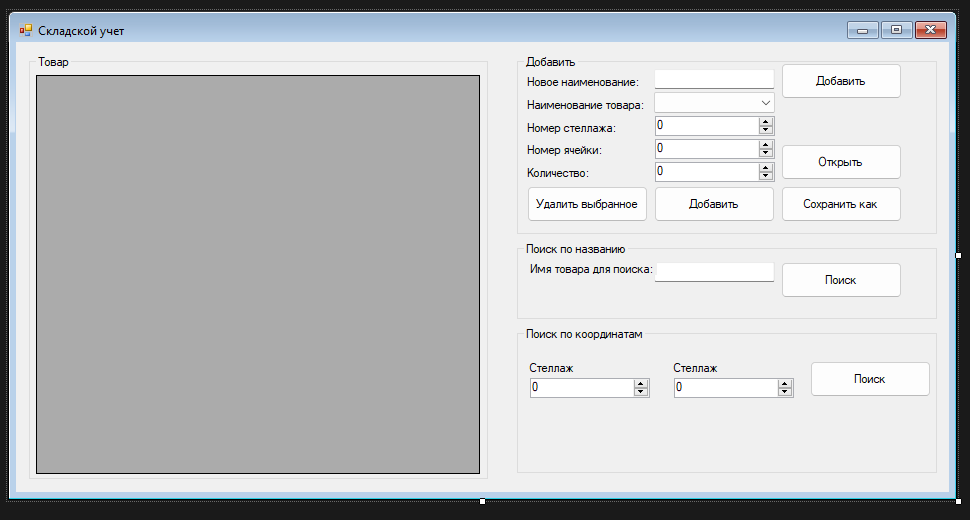


Рис. 2. Форма приложения

Код приложения:

using System; // Пространство имён для базовых типов и функций

using System.Data; // Для работы с данными, DataTable и т.п.

using System.Data.SqlClient; // Для взаимодействия с SQL Server

using System.IO; // Для работы с файлами и потоками

using System.Windows.Forms; // Для создания графического интерфейса Windows Forms

namespace SkladApp // Объявление пространства имён для организации кода

{

public partial class Form1 : Form // Определение формы приложения, наследование от базового класса Form

{

public Form1() // Конструктор формы

{

InitializeComponent(); // Инициализация компонентов формы (автоматическая часть)

LoadEquipment(); // Загрузка данных о оборудовании (товарах) при запуске формы

LoadProductNames(); // Загрузка списка названий товаров для ComboBox

}

private const string ConnectionString = "Server=DESKTOP-RA97ENK;Database=Sklad;Integrated Security=True;"; // Строка подключения к базе данных

// Метод для загрузки всех товаров из базы и отображения в DataGridView

private void LoadEquipment()

{

using (SqlConnection conn = new SqlConnection(ConnectionString)) // Создаем подключение к базе

{

try

{

conn.Open(); // Открываем соединение

string query = @"SELECT \* from products"; // SQL-запрос для получения всех товаров

SqlDataAdapter adapter = new SqlDataAdapter(query, conn); // Создаем адаптер для заполнения DataTable

DataTable dt = new DataTable(); // Создаем таблицу для хранения данных

adapter.Fill(dt); // Заполняем таблицу данными из базы

dataGridView1.DataSource = dt; // Устанавливаем таблицу как источник данных для DataGridView

dataGridView1.Columns["id"].Visible = false; // Скрываем колонку id

}

catch (Exception ex) // Обработка ошибок

{

MessageBox.Show($"Ошибка загрузки данных: {ex.Message}"); // Вывод сообщения об ошибке

}

}

}

// Метод для загрузки названий товаров в ComboBox

private void LoadProductNames()

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открываем соединение

string query = @"SELECT name from products"; // SQL-запрос для получения всех названий

SqlCommand cmd = new SqlCommand(query, conn); // Создаем команду

SqlDataReader reader = cmd.ExecuteReader(); // Выполняем чтение данных

comboBox1.Items.Clear(); // Очищаем текущие элементы ComboBox

while (reader.Read()) // Пока есть строки

{

comboBox1.Items.Add(reader["name"].ToString()); // Добавляем название товара в список

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка загрузки наименований: {ex.Message}"); // Обработка ошибок

}

}

}

// Обработчик кнопки удаления выбранного товара

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

{

if (dataGridView1.SelectedRows.Count > 0) // Проверка, что выбрана строка

{

int id = Convert.ToInt32(dataGridView1.SelectedRows[0].Cells["id"].Value); // Получение id выбранной строки

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открытие подключения

string query = "DELETE FROM products WHERE id = @id"; // SQL-запрос для удаления

SqlCommand cmd = new SqlCommand(query, conn); // Создание команды

cmd.Parameters.AddWithValue("@id", id); // Передача параметра id

cmd.ExecuteNonQuery(); // Выполнение запроса

MessageBox.Show("Товар успешно удален"); // Подтверждение удаления

LoadEquipment(); // Обновление данных на форме

LoadProductNames(); // Обновление списка названий

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка удаления: {ex.Message}"); // Обработка ошибок

}

}

}

else

{

MessageBox.Show("Выберите товар для удаления"); // Предупреждение, если ничего не выбрано

}

}

// Обработчик кнопки добавления нового товара

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

{

if (!string.IsNullOrEmpty(textBox1.Text)) // Проверка, что название товара введено

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открытие соединения

// Вставка нового товара с параметрами

string query = @"INSERT INTO products (name, stillage, cell, colvo)

VALUES (@name, @stillage, @cell, @colvo)";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@name", textBox1.Text); // Название из текстового поля

cmd.Parameters.AddWithValue("@stillage", numericUpDown1.Value); // Стеллаж

cmd.Parameters.AddWithValue("@cell", numericUpDown2.Value); // Ячейка

cmd.Parameters.AddWithValue("@colvo", numericUpDown3.Value); // Количество

cmd.ExecuteNonQuery(); // Выполнение вставки

MessageBox.Show("Товар успешно добавлен");

LoadEquipment(); // Обновление данных

LoadProductNames();

textBox1.Clear(); // Очистка поля ввода

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка добавления: {ex.Message}");

}

}

}

else

{

MessageBox.Show("Введите наименование товара"); // Предупреждение, если название не введено

}

}

// Обработчик кнопки сохранения данных в CSV

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

{

SaveFileDialog saveFileDialog = new SaveFileDialog(); // Диалог сохранения файла

saveFileDialog.Filter = "CSV файлы (\*.csv)|\*.csv|Все файлы (\*.\*)|\*.\*"; // Фильтр по типу файла

saveFileDialog.Title = "Сохранить данные о товарах"; // Заголовок диалога

if (saveFileDialog.ShowDialog() == DialogResult.OK) // Если пользователь выбрал файл

{

try

{

using (StreamWriter writer = new StreamWriter(saveFileDialog.FileName))

{

writer.WriteLine("Наименование;Стеллаж;Ячейка;Количество"); // Заголовки колонок

// Перебор строк таблицы DataGridView

foreach (DataGridViewRow row in dataGridView1.Rows)

{

if (!row.IsNewRow) // Исключая новую (пустую) строку

{

// Запись строки в CSV формате

writer.WriteLine($"{row.Cells["name"].Value};{row.Cells["stillage"].Value};{row.Cells["cell"].Value};{row.Cells["colvo"].Value}");

}

}

}

MessageBox.Show("Данные успешно сохранены в CSV");

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка сохранения: {ex.Message}");

}

}

}

// Обработчик кнопки обновления выбранного товара

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

{

if (comboBox1.SelectedItem != null) // Проверка, что выбран товар

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открываем соединение

// Обновление данных товара по имени

string query = @"UPDATE products

SET name = @name, stillage = @stillage, cell = @cell, colvo = @colvo

WHERE name = @name\_2";

SqlCommand cmd = new SqlCommand(query, conn);

// Передача новых значений

cmd.Parameters.AddWithValue("@name", textBox1.Text);

cmd.Parameters.AddWithValue("@name\_2", comboBox1.SelectedItem.ToString()); // имя для условия

cmd.Parameters.AddWithValue("@stillage", numericUpDown1.Value);

cmd.Parameters.AddWithValue("@cell", numericUpDown2.Value);

cmd.Parameters.AddWithValue("@colvo", numericUpDown3.Value);

cmd.ExecuteNonQuery(); // Выполнение обновления

MessageBox.Show("Данные товара успешно обновлены");

LoadEquipment(); // Обновление отображаемых данных

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка обновления: {ex.Message}");

}

}

}

else

{

MessageBox.Show("Выберите товар для изменения");

}

}

// Обработчик кнопки загрузки данных из CSV файла

private void button5\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog = new OpenFileDialog(); // Диалог открытия файла

openFileDialog.Filter = "CSV файлы (\*.csv)|\*.csv|Все файлы (\*.\*)|\*.\*"; // Фильтр

openFileDialog.Title = "Загрузить данные о товарах"; // Заголовок диалога

if (openFileDialog.ShowDialog() == DialogResult.OK) // Если выбран файл

{

try

{

ClearDatabase(); // Очистка таблицы перед загрузкой

using (StreamReader reader = new StreamReader(openFileDialog.FileName))

{

string line = reader.ReadLine(); // Читаем заголовки

while ((line = reader.ReadLine()) != null) // Пока есть строки

{

string[] parts = line.Split(';'); // Разделение по семиколону

if (parts.Length == 4) // Проверка правильности данных

{

// Добавление товара в базу из файла

AddProductFromFile(parts[0], int.Parse(parts[1]), int.Parse(parts[2]), int.Parse(parts[3]));

}

}

}

MessageBox.Show("Данные успешно загружены из CSV");

LoadEquipment(); // Обновление отображения

LoadProductNames();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка загрузки данных: {ex.Message}");

}

}

}

// Метод для очистки таблицы базы данных

private void ClearDatabase()

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open();

string query = "DELETE FROM products"; // Удаление всех записей из таблицы

SqlCommand cmd = new SqlCommand(query, conn);

cmd.ExecuteNonQuery(); // Выполнение

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка очистки базы данных: {ex.Message}");

}

}

}

// Метод для добавления товара из файла в базу

private void AddProductFromFile(string name, int stillage, int cell, int count)

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open();

string query = @"INSERT INTO products (name, stillage, cell, colvo)

VALUES (@name, @stillage, @cell, @colvo)"; // Вставка нового товара

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@name", name);

cmd.Parameters.AddWithValue("@stillage", stillage);

cmd.Parameters.AddWithValue("@cell", cell);

cmd.Parameters.AddWithValue("@colvo", count);

cmd.ExecuteNonQuery(); // Выполнение вставки

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка добавления товара: {ex.Message}");

}

}

}

// Обработчик кнопки поиска товаров по названию

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

{

if (!string.IsNullOrEmpty(textBox2.Text)) // Проверка, что есть строка поиска

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открытие соединения

// Поиск по названию с использованием LIKE

string query = @"SELECT \* FROM products WHERE name LIKE @name";

SqlDataAdapter adapter = new SqlDataAdapter(query, conn);

adapter.SelectCommand.Parameters.AddWithValue("@name", "%" + textBox2.Text + "%"); // Вставка шаблона поиска

DataTable dt = new DataTable(); // Создаем таблицу для результатов

adapter.Fill(dt); // Заполняем таблицу

dataGridView1.DataSource = dt; // Отображение результатов

dataGridView1.Columns["id"].Visible = false; // Скрываем id

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка поиска: {ex.Message}");

}

}

}

else

{

MessageBox.Show("Введите название для поиска"); // Предупреждение

}

}

// Обработчик поиска по координатам (стеллаж и ячейка)

private void button7\_Click(object sender, EventArgs e)

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открытие соединения

// Поиск товаров по координатам

string query = @"SELECT \* FROM products

WHERE stillage = @stillage AND cell = @cell";

SqlDataAdapter adapter = new SqlDataAdapter(query, conn);

adapter.SelectCommand.Parameters.AddWithValue("@stillage", numericUpDown4.Value); // Координата стеллажа

adapter.SelectCommand.Parameters.AddWithValue("@cell", numericUpDown5.Value); // Координата ячейки

DataTable dt = new DataTable(); // Таблица для результатов

adapter.Fill(dt); // Заполнение

dataGridView1.DataSource = dt; // Отображение

dataGridView1.Columns["id"].Visible = false; // Скрытие id

if (dt.Rows.Count == 0) // Если ничего не найдено

{

MessageBox.Show("Товары с указанными координатами не найдены");

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка поиска: {ex.Message}");

}

}

}

// Обработчик изменения выбранного элемента в ComboBox

private void comboBox1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBox1.SelectedItem != null) // Если выбран элемент

{

using (SqlConnection conn = new SqlConnection(ConnectionString))

{

try

{

conn.Open(); // Открытие соединения

// Запрос для получения данных выбранного товара

string query = @"SELECT stillage, cell, colvo FROM products WHERE name = @name";

SqlCommand cmd = new SqlCommand(query, conn);

cmd.Parameters.AddWithValue("@name", comboBox1.SelectedItem.ToString()); // Передача выбранного имени

using (SqlDataReader reader = cmd.ExecuteReader()) // Чтение данных

{

if (reader.Read()) // Если есть результат

{

// Установка значений numericUpDown в соответствии с данными

numericUpDown1.Value = Convert.ToDecimal(reader["stillage"]);

numericUpDown2.Value = Convert.ToDecimal(reader["cell"]);

numericUpDown3.Value = Convert.ToDecimal(reader["colvo"]);

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка загрузки данных товара: {ex.Message}");

}

}

}

}

}

}