**SQLite**

Алексеев Иван

**Цель**: Разработать программу для учета расходов с реализацией хранения данных в локальной

**Задачи**:

Добавление новой записи по расходам (это могут быть любые траты, платежи,

покупки), которая характеризуется следующими параметрами:

1. Размер платежа

2. Дата платежа (на выбор: текущая дата или другая)

3. Категория (еда, транспорт, жилье, другое и т.д)

4. Название

5. Комментарий

Удаление выбранной записи

Просмотр всех записей

Поиск и фильтрация записей по выбранным параметрам (по названию, по дате, по

размеру платежа, по категории).

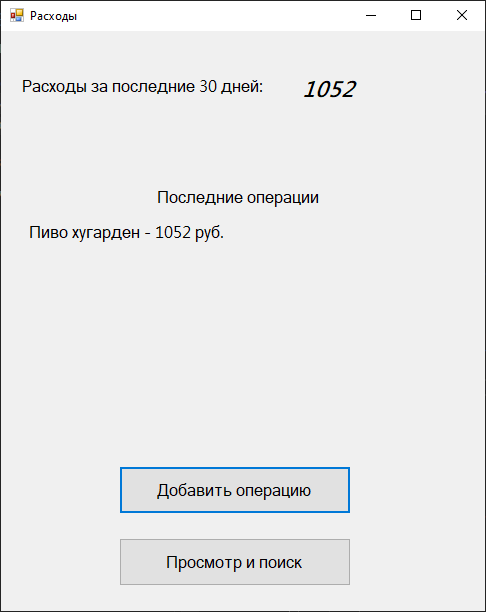


Рисунок 1 - Расходы

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Drawing;

using System.Windows.Forms;

namespace ExpensesApp

{

public partial class Form1 : Form

{

ExpenseTrackerDatabase db;

public Form1()

{

InitializeComponent();

string projectPath = @""C: \Users\mikyl\Desktop\LabSqlite6 - master\Base.db"

db = new ExpenseTrackerDatabase(projectPath);

db.CreateExpenseTable();

CountExpenses();

}

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

{

Form2 добавить = new Form2(db, this);

добавить.Show();

}

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

{

Form3 поиск = new Form3(db);

поиск.Show();

}

public void CountExpenses()

{

flowLayoutPanel1.Controls.Clear();

label2.Text = db.CalculateExpensesLast30Days().ToString();

List<string> lastNExpenses = db.GetLastNExpenses(7);

foreach (string expense in lastNExpenses)

{

Debug.WriteLine(expense);

Label label = new Label

{

AutoSize = true,

ForeColor = Color.Black,

Font = label3.Font,

Text = expense

};

flowLayoutPanel1.Controls.Add(label);

}

}

private void flowLayoutPanel1\_Paint(object sender, PaintEventArgs e)

{

}

}

}

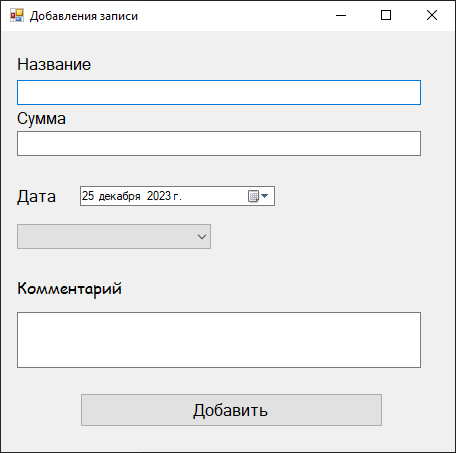


Рисунок 2 - Добавления записи расхода(Form2)

using System;

using System.Linq;

using System.Windows.Forms;

namespace ExpensesApp

{

public partial class Form2 : Form

{

DateTime дата = DateTime.Now;

ExpenseTrackerDatabase database;

Form1 formMain;

public Form2(ExpenseTrackerDatabase db, Form1 form)

{

InitializeComponent();

comboBox1.DropDownStyle = ComboBoxStyle.DropDownList;

this.database = db;

this.formMain = form;

}

private void buttonClick(object sender, EventArgs e)

{

if (!string.IsNullOrEmpty(textBoxNameOp.Text) && !string.IsNullOrEmpty(textBoxCommentOp.Text)

&& !string.IsNullOrEmpty(textBoxSumOp.Text) && !string.IsNullOrEmpty(comboBox1.Text))

{

if (!textBoxSumOp.Text.Any(char.IsLetter) && !textBoxSumOp.Text.Any(c => !char.IsLetterOrDigit(c) && !char.IsWhiteSpace(c)))

{

string имя= textBoxNameOp.Text;

double сумма = Convert.ToDouble(textBoxSumOp.Text);

string категория = comboBox1.Text;

string комментарий = textBoxCommentOp.Text;

database.InsertExpense(имя, сумма, категория, дата, комментарий);

MessageBox.Show("Успешно");

formMain.CountExpenses();

this.Close();

} else

{

MessageBox.Show("Введите корректную сумму");

}

}

else

{

MessageBox.Show("Заполните все данные");

}

}

private void dateTimePicker1\_ValueChanged(object sender, EventArgs e)

{

дата = dateTimePicker1.Value;

}

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

{

}

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

{

}

}

}

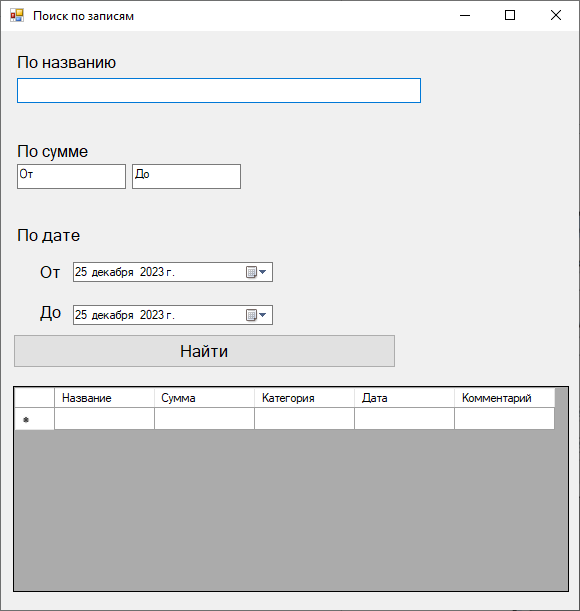


Рисунок 3 - Поиск расходов

using System;

using System.Data;

using System.Linq;

using System.Windows.Forms;

namespace ExpensesApp

{

public partial class Form3 : Form

{

ExpenseTrackerDatabase expenseTrackerDatabase;

public Form3(ExpenseTrackerDatabase db)

{

InitializeComponent();

Data();

this.expenseTrackerDatabase = db;

}

private void Data()

{

//Создаем datatable с названиями столбцов

DataTable dataTable = new DataTable("Расходы");

dataTable.Columns.Add("Название", typeof(string));

dataTable.Columns.Add("Сумма", typeof(string));

dataTable.Columns.Add("Категория", typeof(string));

dataTable.Columns.Add("Дата", typeof(string));

dataTable.Columns.Add("Комментарий", typeof(string));

dataGridView1.DataSource = dataTable;

}

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

{

string имя = filterName.Text;

DateTime начальная\_дата = filterDateStart.Value;

DateTime конечная\_дата= filterDateEnd.Value;

double макс = 0;

double мин = 0;

if (!string.IsNullOrWhiteSpace(filterMax.Text) && !filterMax.Text.Any(char.IsLetter) &&

!filterMax.Text.Any(c => !char.IsLetterOrDigit(c) && !char.IsWhiteSpace(c)))

{

макс = Convert.ToDouble(filterMax.Text);

}

if (!string.IsNullOrWhiteSpace(filterMin.Text) && !filterMin.Text.Any(char.IsLetter) &&

!filterMin.Text.Any(c => !char.IsLetterOrDigit(c) && !char.IsWhiteSpace(c)))

{

мин = Convert.ToDouble(filterMin.Text);

}

dataGridView1.DataSource = expenseTrackerDatabase.GetDataTableWithFilters(имя, мин, макс, начальная\_дата,

конечная\_дата);

}

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

{

}

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

{

}

}

}

Разработка класса с БД (ExpensesTrackerDatabase) для хранения данных расходов. Таблица должна содержать в себе название расхода, сумму, категорию, дату и комментарий.

Запрос на создание БД .  
  
using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SQLite;

using System.Diagnostics;

public class ExpenseTrackerDatabase

{

private string конект;

public ExpenseTrackerDatabase(string dbFilePath)

{

конект = $"Data Source={dbFilePath};Version=3;";

}

public void CreateExpenseTable()

{

using (SQLiteConnection connection = new SQLiteConnection(конект))

{

connection.Open();

// Запрос на создание таблицы расходов

string createTableQuery = @"

CREATE TABLE IF NOT EXISTS Expenses (

Id INTEGER PRIMARY KEY AUTOINCREMENT,

Name TEXT NOT NULL,

Amount REAL NOT NULL,

Category TEXT NOT NULL,

Date TEXT NOT NULL,

Comment TEXT

);";

using (SQLiteCommand command = new SQLiteCommand(createTableQuery, connection))

{

command.ExecuteNonQuery();

}

}

}

public void InsertExpense(string name, double amount, string category, DateTime date, string comment)

{

using (SQLiteConnection connection = new SQLiteConnection(конект))

{

connection.Open();

string insertQuery = @"

INSERT INTO Expenses (Name, Amount, Category, Date, Comment)

VALUES (@Name, @Amount, @Category, @Date, @Comment);";

using (SQLiteCommand command = new SQLiteCommand(insertQuery, connection))

{

command.Parameters.AddWithValue("@Name", name);

command.Parameters.AddWithValue("@Amount", amount);

command.Parameters.AddWithValue("@Category", category);

command.Parameters.AddWithValue("@Date", date.ToString("yyyy-MM-dd HH:mm:ss"));

command.Parameters.AddWithValue("@Comment", comment);

command.ExecuteNonQuery();

}

}

}

public DataTable GetDataTableWithFilters(string nameFilter, double? minAmountFilter, double? maxAmountFilter,

DateTime? startDateFilter, DateTime? endDateFilter)

{

DataTable dataTable = new DataTable();

using (SQLiteConnection connection = new SQLiteConnection(конект))

{

connection.Open();

string selectAllQuery = "SELECT \* FROM Expenses WHERE";

if (!string.IsNullOrWhiteSpace(nameFilter))

{

selectAllQuery += " Name = @nameFilter AND";

}

if (minAmountFilter.HasValue && minAmountFilter != 0)

{

selectAllQuery += " Amount >= @minAmountFilter AND";

}

if (maxAmountFilter.HasValue && maxAmountFilter != 0)

{

selectAllQuery += " Amount <= @maxAmountFilter AND";

}

selectAllQuery += " Date BETWEEN @startDateFilter AND @endDateFilter";

selectAllQuery = selectAllQuery.TrimEnd(' ', 'A', 'N', 'D');

dataTable.Columns.Add("Название", typeof(string));

dataTable.Columns.Add("Сумма", typeof(string));

dataTable.Columns.Add("Категория", typeof(string));

dataTable.Columns.Add("Дата", typeof(string));

dataTable.Columns.Add("Комментарий", typeof(string));

Debug.WriteLine(selectAllQuery);

using (SQLiteCommand command = new SQLiteCommand(selectAllQuery, connection))

{

if (!string.IsNullOrWhiteSpace(nameFilter))

{

command.Parameters.AddWithValue("@nameFilter", nameFilter);

}

if (minAmountFilter.HasValue)

{

command.Parameters.AddWithValue("@minAmountFilter", minAmountFilter.Value);

}

if(maxAmountFilter.HasValue && maxAmountFilter != 0)

{

command.Parameters.AddWithValue("@maxAmountFilter", maxAmountFilter.Value);

}

command.Parameters.AddWithValue("@startDateFilter", startDateFilter.Value.ToString(("yyyy-MM-dd HH:mm:ss")));

command.Parameters.AddWithValue("@endDateFilter", endDateFilter.Value.ToString(("yyyy-MM-dd HH:mm:ss")));

using (SQLiteDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

dataTable.Rows.Add(

reader["Name"].ToString(),

reader["Amount"].ToString(),

reader["Category"].ToString(),

reader["Date"].ToString(),

reader["Comment"].ToString()

);

}

}

}

}

return dataTable;

}

public double CalculateExpensesLast30Days()

{

double totalExpenses = 0;

using (SQLiteConnection connection = new SQLiteConnection(конект))

{

connection.Open();

string query = "SELECT SUM(Amount) FROM Expenses WHERE Date >= @StartDate AND Date <= @EndDate";

DateTime endDate = DateTime.Now;

DateTime startDate = endDate.AddDays(-30);

Debug.WriteLine(startDate.ToString("yyyy-MM-dd HH:mm:ss"));

Debug.WriteLine(endDate.ToString("yyyy-MM-dd HH:mm:ss"));

using (SQLiteCommand command = new SQLiteCommand(query, connection))

{

command.Parameters.AddWithValue("@StartDate", startDate.ToString("yyyy-MM-dd HH:mm:ss"));

command.Parameters.AddWithValue("@EndDate", endDate.ToString("yyyy-MM-dd HH:mm:ss"));

object result = command.ExecuteScalar();

if (result != null && result != DBNull.Value)

{

Debug.WriteLine(result.ToString());

// конвертируем

totalExpenses = Convert.ToDouble(result);

}

}

connection.Close();

}

return totalExpenses;

}

public List<string> GetLastNExpenses(int n)

{

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

using (SQLiteConnection connection = new SQLiteConnection(конект))

{

connection.Open();

string query = "SELECT Name, Amount FROM Expenses ORDER BY Id DESC LIMIT @N;";

using (SQLiteCommand command = new SQLiteCommand(query, connection))

{

command.Parameters.AddWithValue("@N", n);

using (SQLiteDataReader reader = command.ExecuteReader())

{

while (reader.Read())

{

string expenseString = $"{reader["Name"]} - {reader["Amount"]} руб.";

expensesList.Add(expenseString);

}

}

}

connection.Close();

}

return expensesList;

}

}