﻿using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace Create\_Chamcong

{

public partial class ChamCong : Form

{

string sCon = "Data Source=TTTT\\TTTT;Initial Catalog=NhanSu;Integrated Security=True";

public ChamCong()

{

InitializeComponent();

}

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

{

// Kiểm tra mã công

string sMaCong = textBox1.Text.Trim(); // Lấy mã công từ TextBox

// Kiểm tra nếu mã công trống

if (string.IsNullOrEmpty(sMaCong))

{

MessageBox.Show("Vui lòng nhập mã công để xóa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Xác nhận trước khi xóa

DialogResult confirmResult = MessageBox.Show($"Bạn có chắc muốn xóa công với mã công {sMaCong} không?",

"Xác nhận xóa",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question);

if (confirmResult == DialogResult.No)

{

return;

}

// Câu lệnh SQL để kiểm tra xem mã công có tồn tại không

string checkQuery = "SELECT COUNT(\*) FROM ChamCong WHERE MaCong = @MaCong";

string sQuery = "DELETE FROM ChamCong WHERE MaCong = @MaCong";

// Sử dụng khối 'using' để tự động quản lý tài nguyên kết nối

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

// Mở kết nối

con.Open();

// Kiểm tra xem mã công có tồn tại trong bảng không

using (SqlCommand checkCmd = new SqlCommand(checkQuery, con))

{

checkCmd.Parameters.AddWithValue("@MaCong", sMaCong);

int count = (int)checkCmd.ExecuteScalar();

if (count == 0)

{

MessageBox.Show("Mã công không tồn tại trong hệ thống.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

}

// Thực hiện xóa dữ liệu

using (SqlCommand cmd = new SqlCommand(sQuery, con))

{

cmd.Parameters.AddWithValue("@MaCong", sMaCong);

// Thực thi câu lệnh xóa

int rowsAffected = cmd.ExecuteNonQuery();

if (rowsAffected > 0)

{

MessageBox.Show("Xóa thành công!");

LoadChamCongData(); // Tải lại dữ liệu sau khi xóa

ResetForm(); // Reset lại form sau khi xóa

}

else

{

MessageBox.Show("Không thể xóa công. Vui lòng thử lại.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

catch (Exception ex)

{

// Hiển thị thông báo lỗi chi tiết

MessageBox.Show($"Xảy ra lỗi trong quá trình xóa!\n{ex.Message}", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

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

{

// Kiểm tra các trường dữ liệu không trống

if (string.IsNullOrEmpty(txtMaNV.Text))

{

MessageBox.Show("Mã nhân viên không được để trống.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Lấy dữ liệu từ các trường TextBox và DateTimePicker

string sMaNV = txtMaNV.Text.Trim();

string sNgayCong = NgayCong.Value.ToString("yyyy-MM-dd");

string sGioVao = GioVao.Value.ToString("HH:mm:ss");

string sGioRa = GioRa.Value.ToString("HH:mm:ss");

int iLoaiCong = Loaicong\_NgayCongTac.Checked ? 1 : 0; // Nếu Loại công là ngày công tác

// Kiểm tra nếu mã công đã được lấy từ DataGridView

if (string.IsNullOrEmpty(txtMaNV.Text))

{

MessageBox.Show("Chưa chọn nhân viên nào để sửa.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

try

{

using (SqlConnection con = new SqlConnection(sCon))

{

con.Open();

// Câu lệnh SQL cập nhật dữ liệu

string sQuery = "UPDATE ChamCong SET MaNhanVien = @MaNhanVien, NgayCong = @NgayCong, GioVao = @GioVao, GioRa = @GioRa, LoaiCong = @LoaiCong " +

"WHERE MaCong = @MaCong";

// Tạo lệnh SQL và gán tham số

using (SqlCommand cmd = new SqlCommand(sQuery, con))

{

// Thêm các tham số

cmd.Parameters.AddWithValue("@MaNhanVien", sMaNV);

cmd.Parameters.AddWithValue("@NgayCong", sNgayCong);

cmd.Parameters.AddWithValue("@GioVao", sGioVao);

cmd.Parameters.AddWithValue("@GioRa", sGioRa);

cmd.Parameters.AddWithValue("@LoaiCong", iLoaiCong);

// Cần lấy mã công từ DataGridView

int rowIndex = ThongtinChamCong.SelectedCells[0].RowIndex;

string maCong = ThongtinChamCong.Rows[rowIndex].Cells["MaCong"].Value.ToString();

cmd.Parameters.AddWithValue("@MaCong", maCong);

// Thực thi câu lệnh

int result = cmd.ExecuteNonQuery();

if (result > 0)

{

MessageBox.Show("Cập nhật dữ liệu thành công!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

// Cập nhật lại DataGridView nếu cần thiết

LoadChamCongData(); // Tải lại dữ liệu từ cơ sở dữ liệu vào DataGridView

ResetForm();

}

else

{

MessageBox.Show("Không có dữ liệu nào được cập nhật.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi trong quá trình cập nhật dữ liệu!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void ChamCong\_FormClosing(object sender, FormClosingEventArgs e)

{

MessageBox.Show("Hẹn gặp lại lần sau!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

private string GenerateMaChamCong()

{

SqlConnection con = new SqlConnection(sCon);

string newMaChamCong = "";

try

{

con.Open();

// Lấy mã chấm công cuối cùng từ bảng, thay "MaChamCong" bằng "MaCong"

string query = "SELECT TOP 1 MaCong FROM ChamCong ORDER BY MaCong DESC";

SqlCommand cmd = new SqlCommand(query, con);

// Dùng ExecuteScalar để chỉ lấy 1 giá trị

var result = cmd.ExecuteScalar();

if (result != null)

{

string lastMaChamCong = result.ToString();

// Lấy số từ mã chấm công cuối cùng và tăng lên 1

int lastNumber = int.Parse(lastMaChamCong.Substring(2)); // Lấy phần số sau "MC"

int newNumber = lastNumber + 1;

// Đảm bảo rằng mã mới có độ dài là 9 chữ số

newMaChamCong = "MC" + newNumber.ToString().PadLeft(9, '0');

}

else

{

// Nếu không có mã nào, bắt đầu từ "MC000000001"

newMaChamCong = "MC000000001";

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi khi tạo mã chấm công!");

}

finally

{

if (con.State == ConnectionState.Open)

con.Close();

}

return newMaChamCong;

}

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

{

string sMaNV = txtMaNV.Text.Trim();

string sNgayCong = NgayCong.Value.ToString("yyyy-MM-dd");

string sGioVao = GioVao.Value.ToString("HH:mm:ss");

string sGioRa = GioRa.Value.ToString("HH:mm:ss");

int iLoaiCong = Loaicong\_NgayCongTac.Checked ? 1 : 0;

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

con.Open();

// Đăng ký sự kiện InfoMessage để nhận thông báo từ SQL Server

con.InfoMessage += (s, ev) =>

{

foreach (SqlError err in ev.Errors)

{

// Hiển thị thông báo thông qua MessageBox

MessageBox.Show(err.Message, "Thông báo từ SQL Server");

}

};

// Tạo SqlCommand để gọi thủ tục thêm chấm công

SqlCommand command = new SqlCommand("dbo.pr\_Insert\_ChamCong", con);

command.CommandType = CommandType.StoredProcedure;

// Thêm các tham số vào thủ tục

command.Parameters.AddWithValue("@MaNhanVien", sMaNV);

command.Parameters.AddWithValue("@NgayCong", sNgayCong);

command.Parameters.AddWithValue("@GioVao", sGioVao);

command.Parameters.AddWithValue("@GioRa", sGioRa);

command.Parameters.AddWithValue("@LoaiCong", iLoaiCong);

// Thêm tham số đầu ra (ret)

SqlParameter retParam = new SqlParameter("@retval", SqlDbType.Bit)

{

Direction = ParameterDirection.Output

};

command.Parameters.Add(retParam);

// Thực thi thủ tục

command.ExecuteNonQuery();

// Lấy giá trị trả về từ tham số OUTPUT

bool isSuccess = Convert.ToBoolean(retParam.Value);

// Kiểm tra kết quả thực thi

if (isSuccess)

{

ResetForm();

LoadChamCongData(); // Hàm cập nhật lại dữ liệu hiển thị

}

}

catch (SqlException ex)

{

// Hiển thị thông báo lỗi từ SQL

MessageBox.Show($"Có lỗi xảy ra: {ex.Message}", "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

catch (Exception ex)

{

// Xử lý lỗi

MessageBox.Show("Có lỗi xảy ra: " + ex.Message, "Lỗi", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

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

{

DataGridViewRow row = ThongtinChamCong.Rows[e.RowIndex];

textBox1.Text = ThongtinChamCong.Rows[e.RowIndex].Cells["MaCong"].Value.ToString();

txtMaNV.Text = ThongtinChamCong.Rows[e.RowIndex].Cells["MaNhanVien"].Value.ToString();

NgayCong.Value = Convert.ToDateTime(ThongtinChamCong.Rows[e.RowIndex].Cells["NgayCong"].Value);

GioVao.Value = DateTime.Parse(row.Cells["GioVao"].Value?.ToString());

GioRa.Value = DateTime.Parse(row.Cells["GioRa"].Value?.ToString());

int iLoaiCong = Convert.ToInt16(ThongtinChamCong.Rows[e.RowIndex].Cells["LoaiCong"].Value);

if (iLoaiCong == 1)

{

Loaicong\_NgayCongTac.Checked = true;

}

else

{

Loaicong\_Ngaythuong.Checked = true;

}

textBox1.Enabled = false;

}

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

{

// Sử dụng khối 'using' để tự động quản lý tài nguyên kết nối

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

// Kiểm tra kết nối trước khi mở

if (con.State != ConnectionState.Open)

{

con.Open();

}

txtMaCong.Enabled = false; // Vô hiệu hóa mã công

// Lấy mã nhân viên từ TextBox và loại bỏ khoảng trắng đầu và cuối

string sMaNV = txtMaNV.Text.Trim();

// Kiểm tra mã nhân viên không được để trống hoặc chỉ chứa khoảng trắng

if (string.IsNullOrWhiteSpace(sMaNV))

{

MessageBox.Show("Vui lòng nhập mã nhân viên!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Chuẩn bị câu lệnh SQL với kiểu dữ liệu cụ thể cho tham số

string sQuery = @"SELECT MaCong, MaNhanVien, NgayCong, GioVao, GioRa, LoaiCong

FROM ChamCong

WHERE MaNhanVien = @MaNhanVien";

// Tạo SqlCommand và gán tham số với kiểu dữ liệu chính xác

using (SqlCommand cmd = new SqlCommand(sQuery, con))

{

// Giả sử MaNhanVien là kiểu VARCHAR trong cơ sở dữ liệu

cmd.Parameters.Add("@MaNhanVien", SqlDbType.VarChar, 50).Value = sMaNV;

// Sử dụng SqlDataAdapter để lấy dữ liệu

using (SqlDataAdapter da = new SqlDataAdapter(cmd))

{

DataTable dt = new DataTable();

da.Fill(dt);

// Hiển thị dữ liệu trong DataGridView

if (dt.Rows.Count > 0)

{

ThongtinChamCong.DataSource = dt;

}

else

{

MessageBox.Show("Không tìm thấy dữ liệu cho mã nhân viên này.", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Information);

ThongtinChamCong.DataSource = null; // Xóa dữ liệu cũ trong bảng

}

}

}

}

catch (Exception ex)

{

MessageBox.Show($"Lỗi: {ex.Message}\n{ex.StackTrace}", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void LoadChamCongData()

{

try

{

using (SqlConnection con = new SqlConnection(sCon))

{

con.Open();

string sQuery = "SELECT MaCong, MaNhanVien, NgayCong, GioVao, GioRa, LoaiCong FROM ChamCong";

SqlDataAdapter da = new SqlDataAdapter(sQuery, con);

DataTable dt = new DataTable();

da.Fill(dt);

ThongtinChamCong.DataSource = dt; // Gán dữ liệu vào DataGridView

}

}

catch (Exception ex)

{

MessageBox.Show("Lỗi trong quá trình tải dữ liệu!", "Thông báo", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

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

{

LoadChamCongData(); // Gọi phương thức tải dữ liệu khi form được mở

}

private void ResetForm()

{

// Xóa nội dung trong TextBox

textBox1.Text = string.Empty;

txtMaNV.Text = string.Empty;

GioVao.Text = string.Empty;

GioRa.Text = string.Empty;

// Đặt DateTimePicker về ngày hiện tại

NgayCong.Value = DateTime.Now;

// Đặt RadioButton về giá trị mặc định (ví dụ: Công ngày)

Loaicong\_Ngaythuong.Checked = true;

Loaicong\_NgayCongTac.Checked = false;

textBox1.Focus();

textBox1.Enabled = false;

}

private void LoadChamCongData(object sender, EventArgs e)

{

}

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

{

int thang = NgayCong.Value.Month;

int nam = NgayCong.Value.Year;

// Chuỗi kết nối SQL

string connectionString = sCon; // Thay `sCon` bằng chuỗi kết nối của bạn.

// Kết nối tới cơ sở dữ liệu và gọi thủ tục

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

// Tạo SqlCommand để gọi thủ tục

using (SqlCommand cmd = new SqlCommand("ThongKeThongTinNhanVien", con))

{

cmd.CommandType = CommandType.StoredProcedure;

// Thêm các tham số đầu vào

cmd.Parameters.AddWithValue("@Thang", thang);

cmd.Parameters.AddWithValue("@Nam", nam);

// Tạo SqlDataAdapter để nhận dữ liệu trả về

SqlDataAdapter da = new SqlDataAdapter(cmd);

// Lưu dữ liệu trả về vào DataTable

DataTable dtThongKe = new DataTable();

da.Fill(dtThongKe);

// Hiển thị kết quả trong DataGridView (hoặc các thành phần khác)

ThongtinChamCong.DataSource = dtThongKe;

}

MessageBox.Show("Thống kê hoàn thành!", "Thông báo");

ResetForm();

}

catch (SqlException ex)

{

MessageBox.Show($"Lỗi SQL: {ex.Message}", "Lỗi");

}

catch (Exception ex)

{

MessageBox.Show($"Lỗi: {ex.Message}", "Lỗi");

}

}

}

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

{

// Xóa nội dung trong TextBox

textBox1.Text = string.Empty;

txtMaNV.Text = string.Empty;

GioVao.Text = string.Empty;

GioRa.Text = string.Empty;

// Đặt DateTimePicker về ngày hiện tại

NgayCong.Value = DateTime.Now;

// Đặt RadioButton về giá trị mặc định (ví dụ: Công ngày)

Loaicong\_Ngaythuong.Checked = true;

Loaicong\_NgayCongTac.Checked = false;

textBox1.Focus();

LoadChamCongData();

}

}

}