﻿using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace frmChucVu

{

public partial class frmPhongBan : Form

{

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

public frmPhongBan()

{

InitializeComponent();

}

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

{

SqlConnection con = new SqlConnection(sCon);

try

{

con.Open();

}

catch (Exception ex)

{

MessageBox.Show("Xảy ra lỗi trong quá trình kết nối DB", "Thông báo");

}

string sQuery = "select \* from PhongBan";

SqlDataAdapter adapter = new SqlDataAdapter(sQuery, con);

DataSet ds = new DataSet();

// Đổ dữ liệu vào DataTable

adapter.Fill(ds, "PhongBan");

// Hiển thị kết quả trên DataGridView

dgvPhongBan.DataSource = ds.Tables["PhongBan"];

con.Close();

}

private void LoadData()

{

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

con.Open();

// Câu lệnh SQL để lấy dữ liệu

string query = @"SELECT MaPhongBan, TenPhongBan, SDT FROM PhongBan";

// Tạo SqlDataAdapter để lấy dữ liệu

SqlDataAdapter da = new SqlDataAdapter(query, con);

// Tạo DataTable để chứa dữ liệu

DataTable dt = new DataTable();

// Đổ dữ liệu từ SqlDataAdapter vào DataTable

da.Fill(dt);

// Gán dữ liệu cho DataGridView

dgvPhongBan.DataSource = dt;

}

catch (Exception ex)

{

MessageBox.Show($"Lỗi khi tải dữ liệu: {ex.Message}", "Lỗi");

}

}

}

private void ResetForm()

{

// Xóa nội dung trong các TextBox

txtMaPB.Clear();

txtTenPB.Clear();

txtSDT.Clear();

// Đặt lại trạng thái mặc định của các trường nhập liệu (nếu cần)

txtMaPB.Focus(); // Đặt con trỏ vào TextBox MaPhongBan

txtMaPB.Enabled = true;

}

private void bt\_Them\_Click(object sender, EventArgs e)

{

string maPhongBan = GenerateMaPhongBan();

string tenPhongBan = txtTenPB.Text.Trim();

string sdt = txtSDT.Text.Trim();

if (!IsValidPhoneNumber(sdt))

{

MessageBox.Show("Số điện thoại không hợp lệ. Vui lòng nhập số có đúng 10 chữ số.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

txtSDT.Focus(); // Đặt con trỏ vào ô nhập SDT

return;

}

// Kiểm tra kết nối SQL

using (SqlConnection con = new SqlConnection(sCon)) // sCon là chuỗi kết nối SQL

{

try

{

con.Open();

// Lệnh SQL thêm mới

string query = @"INSERT INTO PhongBan (MaPhongBan, TenPhongBan, SDT)

VALUES (@MaPhongBan, @TenPhongBan, @SDT)";

// Tạo SqlCommand

SqlCommand command = new SqlCommand(query, con);

// Thêm tham số vào câu lệnh SQL

command.Parameters.AddWithValue("@MaPhongBan", maPhongBan);

command.Parameters.AddWithValue("@TenPhongBan", tenPhongBan);

command.Parameters.AddWithValue("@SDT", sdt);

// Thực thi lệnh

int rowsAffected = command.ExecuteNonQuery();

// Kiểm tra kết quả

if (rowsAffected > 0)

{

MessageBox.Show("Thêm phòng ban thành công!", "Thông báo");

LoadData();

}

else

{

MessageBox.Show("Thêm phòng ban thất bại.", "Thông báo");

}

// Cập nhật lại dữ liệu (nếu cần) và reset form

ResetForm();

}

catch (SqlException ex)

{

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

}

catch (Exception ex)

{

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

}

}

}

private bool IsValidPhoneNumber(string sdt)

{

// Kiểm tra chuỗi có phải là số và có đúng 10 ký tự

return !string.IsNullOrEmpty(sdt) && sdt.Length == 10 && sdt.All(char.IsDigit);

}

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

{

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

// Lấy dữ liệu từ các cột của hàng được chọn

string maPhongBan = row.Cells["MaPhongBan"].Value.ToString();

string tenPhongBan = row.Cells["TenPhongBan"].Value.ToString();

string sdt = row.Cells["SDT"].Value.ToString();

// Gán giá trị cho các TextBox hoặc ComboBox

txtMaPB.Text = maPhongBan;

txtTenPB.Text = tenPhongBan;

txtSDT.Text = sdt;

txtMaPB.Enabled = false;

}

private void bt\_Sua\_Click(object sender, EventArgs e)

{

string maPhongBan = txtMaPB.Text;

string tenPhongBan = txtTenPB.Text;

string sdt = txtSDT.Text;

// Kiểm tra các dữ liệu đầu vào

if (string.IsNullOrEmpty(maPhongBan) || string.IsNullOrEmpty(tenPhongBan) || string.IsNullOrEmpty(sdt))

{

MessageBox.Show("Vui lòng điền đầy đủ thông tin.", "Thông báo");

return;

}

if (!IsValidPhoneNumber(sdt))

{

MessageBox.Show("Số điện thoại không hợp lệ. Vui lòng nhập số có đúng 10 chữ số.", "Cảnh báo", MessageBoxButtons.OK, MessageBoxIcon.Warning);

txtSDT.Focus(); // Đặt con trỏ vào ô nhập SDT

return;

}

// Kết nối tới cơ sở dữ liệu

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

con.Open();

// Câu lệnh SQL để cập nhật thông tin phòng ban

string query = @"UPDATE PhongBan

SET TenPhongBan = @TenPhongBan,

SDT = @SDT

WHERE MaPhongBan = @MaPhongBan";

// Tạo SqlCommand với câu lệnh query

SqlCommand command = new SqlCommand(query, con);

// Thêm tham số vào câu lệnh SQL

command.Parameters.AddWithValue("@MaPhongBan", maPhongBan);

command.Parameters.AddWithValue("@TenPhongBan", tenPhongBan);

command.Parameters.AddWithValue("@SDT", sdt);

// Thực thi câu lệnh SQL

int rowsAffected = command.ExecuteNonQuery();

// Kiểm tra xem có bản ghi nào được cập nhật không

if (rowsAffected > 0)

{

MessageBox.Show("Cập nhật thông tin phòng ban thành công!", "Thông báo");

LoadData(); // Tải lại dữ liệu vào DataGridView

}

else

{

MessageBox.Show("Không tìm thấy phòng ban với mã này để cập nhật.", "Thông báo");

}

// Reset form

ResetForm();

}

catch (SqlException ex)

{

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

}

catch (Exception ex)

{

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

}

}

}

private void bt\_Xoa\_Click(object sender, EventArgs e)

{

string maPhongBan = txtMaPB.Text.Trim();

// Kiểm tra xem mã phòng ban có trống hay không

if (string.IsNullOrEmpty(maPhongBan))

{

MessageBox.Show("Vui lòng chọn phòng ban cần xóa!", "Thông báo");

return;

}

// Hiển thị hộp thoại xác nhận trước khi xóa phòng ban

DialogResult result = MessageBox.Show($"Bạn có chắc chắn muốn xóa phòng ban có mã '{maPhongBan}' không?",

"Xác nhận xóa phòng ban",

MessageBoxButtons.YesNo,

MessageBoxIcon.Warning);

if (result == DialogResult.Yes)

{

// Tạo một hộp thoại yêu cầu người dùng chọn phòng ban mới

string queryPhongBanMoi = "SELECT MaPhongBan, TenPhongBan FROM PhongBan WHERE MaPhongBan != @MaPhongBan";

// Mở kết nối với cơ sở dữ liệu và lấy dữ liệu phòng ban

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand(queryPhongBanMoi, con);

cmd.Parameters.AddWithValue("@MaPhongBan", maPhongBan);

SqlDataReader reader = cmd.ExecuteReader();

// Tạo một ComboBox tạm để người dùng chọn phòng ban mới

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

while (reader.Read())

{

phongBanList.Add(reader["MaPhongBan"].ToString() + " - " + reader["TenPhongBan"].ToString());

}

reader.Close();

// Kiểm tra nếu có phòng ban khác để chọn

if (phongBanList.Count == 0)

{

MessageBox.Show("Không có phòng ban nào để chọn", "Thông báo");

return;

}

// Hiển thị ComboBox cho người dùng chọn phòng ban mới

string selectedPhongBan = ShowComboBoxDialog(phongBanList);

// Kiểm tra người dùng có chọn phòng ban mới không

if (string.IsNullOrEmpty(selectedPhongBan))

{

MessageBox.Show("Vui lòng chọn phòng ban mới để chuyển các nhân viên.", "Thông báo");

return;

}

// Cập nhật tất cả nhân viên có phòng ban này sang phòng ban mới

string updateQuery = @"

UPDATE NhanVien

SET MaPhongBan = @MaPhongBanMoi

WHERE MaPhongBan = @MaPhongBan";

SqlCommand updateCommand = new SqlCommand(updateQuery, con);

updateCommand.Parameters.AddWithValue("@MaPhongBanMoi", selectedPhongBan.Split('-')[0].Trim()); // Mã phòng ban mới

updateCommand.Parameters.AddWithValue("@MaPhongBan", maPhongBan);

updateCommand.ExecuteNonQuery();

// Xóa phòng ban

string deleteQuery = "DELETE FROM PhongBan WHERE MaPhongBan = @MaPhongBan";

SqlCommand deleteCommand = new SqlCommand(deleteQuery, con);

deleteCommand.Parameters.AddWithValue("@MaPhongBan", maPhongBan);

deleteCommand.ExecuteNonQuery();

MessageBox.Show("Xóa phòng ban thành công!", "Thông báo");

LoadData();

}

catch (SqlException ex)

{

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

}

catch (Exception ex)

{

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

}

}

}

else

{

MessageBox.Show("Đã hủy thao tác xóa phòng ban.", "Thông báo");

}

ResetForm();

}

private string ShowComboBoxDialog(List<string> phongBanList)

{

Form form = new Form();

form.Text = "Chọn phòng ban thay thế cho nhân viên"; // Tiêu đề form

// Tạo form tạm để hiển thị ComboBox

ComboBox comboBox = new ComboBox();

comboBox.DataSource = phongBanList;

comboBox.DropDownStyle = ComboBoxStyle.DropDownList;

comboBox.Width = 500;

Button buttonOk = new Button() { Text = "Ok", DialogResult = DialogResult.OK, Dock = DockStyle.Bottom };

Button buttonCancel = new Button() { Text = "Cancel", DialogResult = DialogResult.Cancel, Dock = DockStyle.Bottom };

form.Controls.Add(comboBox);

form.Controls.Add(buttonOk);

form.Controls.Add(buttonCancel);

form.StartPosition = FormStartPosition.CenterScreen;

// Hiển thị form tạm để chọn phòng ban

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

{

return comboBox.SelectedItem.ToString();

}

return string.Empty;

}

private void bt\_Xem\_Click(object sender, EventArgs e)

{

string maPhongBan = txtMaPB.Text.Trim();

string tenPhongBan = txtTenPB.Text.Trim();

string sdt = txtSDT.Text.Trim();

using (SqlConnection con = new SqlConnection(sCon))

{

try

{

con.Open();

// Tạo câu lệnh SQL với điều kiện LIKE cho tất cả các trường

string query = @"

SELECT MaPhongBan, TenPhongBan, SDT

FROM PhongBan

WHERE (@MaPhongBan IS NULL OR MaPhongBan LIKE @MaPhongBan)

AND (@TenPhongBan IS NULL OR TenPhongBan LIKE @TenPhongBan)

AND (@SDT IS NULL OR SDT LIKE @SDT)";

// Tạo SqlCommand với câu lệnh SQL

SqlCommand command = new SqlCommand(query, con);

// Thêm các tham số vào câu lệnh SQL

command.Parameters.AddWithValue("@MaPhongBan", string.IsNullOrEmpty(maPhongBan) ? (object)DBNull.Value : "%" + maPhongBan + "%");

command.Parameters.AddWithValue("@TenPhongBan", string.IsNullOrEmpty(tenPhongBan) ? (object)DBNull.Value : "%" + tenPhongBan + "%");

command.Parameters.AddWithValue("@SDT", string.IsNullOrEmpty(sdt) ? (object)DBNull.Value : "%" + sdt + "%");

// Tạo SqlDataAdapter để lấy dữ liệu

SqlDataAdapter adapter = new SqlDataAdapter(command);

// Tạo DataTable để chứa dữ liệu

DataTable dt = new DataTable();

// Đổ dữ liệu vào DataTable

adapter.Fill(dt);

// Kiểm tra nếu có kết quả

if (dt.Rows.Count > 0)

{

// Gán dữ liệu vào DataGridView

dgvPhongBan.DataSource = dt;

}

else

{

MessageBox.Show("Không tìm thấy phòng ban nào khớp với từ khóa tìm kiếm!", "Thông báo");

}

}

catch (SqlException ex)

{

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

}

catch (Exception ex)

{

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

}

}

}

private string GenerateMaPhongBan()

{

SqlConnection con = new SqlConnection(sCon);

string newMaPB = "";

try

{

con.Open();

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

string query = "SELECT TOP 1 MaPhongBan FROM PhongBan ORDER BY MaPhongBan DESC";

SqlCommand cmd = new SqlCommand(query, con);

// Dùng ExecuteScalar để chỉ lấy 1 giá trị

var result = cmd.ExecuteScalar();

if (result != null)

{

string lastMaPB = result.ToString();

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

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

int newNumber = lastNumber + 1;

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

newMaPB = "PB" + newNumber.ToString().PadLeft(4, '0');

}

else

{

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

newMaPB = "PB0001";

}

}

catch (Exception ex)

{

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

}

finally

{

if (con.State == ConnectionState.Open)

con.Close();

}

return newMaPB;

}

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

{

ResetForm();

LoadData();

}

}

}