Documentary on the CRUD Program

This program is a simple Windows Forms application written in C# that performs basic CRUD (Create, Read, Update, Delete) operations on a SQL Server database named `crud`. The application allows users to manage records through a graphical user interface (GUI).

Key Components and Their Functions

**1. Namespace and Class Definition**

csharp

using System.Data;

using System.Data.SqlClient;

using System.Xml.Linq;

namespace CRUD

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

GridView1.Visible = false;

}

}

}

- The program is encapsulated within the `CRUD` namespace.

- `Form1` is the main form class that inherits from `Form`.

**2. Form Load Event**

csharp

private void Form1\_Load(object sender, EventArgs e)

{

}

- This method is executed when the form is loaded. Currently, it does nothing.

3. Label Click Events

csharp

private void label1\_Click(object sender, EventArgs e)

{

}

private void label4\_Click(object sender, EventArgs e)

{

}

- These methods handle click events for labels `label1` and `label4`. They are currently placeholders and do nothing.

**4. Toggle Visibility Button (button1)**

csharp

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

{

foreach (Control control in this.Controls)

{

if (control is DataGridView)

{

control.Visible = false;

}

else

{

control.Visible = true;

}

}

}

- This button toggles the visibility of the `DataGridView` control and shows the table for the database.

**5. Delete Record Button (button4)**

csharp

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

{

SqlConnection con = new SqlConnection("Data Source=DESKTOP-UVGR9G8\\SQLEXPRESS;Initial Catalog=crud;Integrated Security=True;TrustServerCertificate=True");

con.Open();

string deleteQuery = "DELETE FROM crud WHERE id=@id";

SqlCommand cmd = new SqlCommand(deleteQuery, con);

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

int count = cmd.ExecuteNonQuery();

if (count > 0)

{

MessageBox.Show("deleted Successfully", "info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

- This button deletes a record from the database where the ID matches the value from `numericUpDown1`.

**6. DataGridView Cell Content Click Event**

csharp

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

{

}

- This method is a placeholder for handling clicks within the `DataGridView` cells.

7. Insert Record Button (button2)

csharp

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

{

bool isAnyempty = false;

foreach (Control control in this.Controls)

{

if (control is TextBox)

{

if (control.Text.Length == 0)

{

isAnyempty = true;

break;

}

}

}

if (isAnyempty)

{

MessageBox.Show("Fill the required information ", " info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

SqlConnection con = new SqlConnection("Data Source=DESKTOP-UVGR9G8\\SQLEXPRESS;Initial Catalog=crud;Integrated Security=True;TrustServerCertificate=True");

con.Open();

string insertQuery = "INSERT INTO crud (email, address, username, password) VALUES (@email, @address, @username, @password)";

SqlCommand cmd = new SqlCommand(insertQuery, con);

cmd.Parameters.AddWithValue("@email", txtEmail.Text);

cmd.Parameters.AddWithValue("@address", txtAddress.Text);

cmd.Parameters.AddWithValue("@username", txtUserName.Text);

cmd.Parameters.AddWithValue("@password", txtPassword.Text);

int count = cmd.ExecuteNonQuery();

con.Close();

if (count > 0)

{

MessageBox.Show("Created succesfully", "info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("Error in Creating", "info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

}

- This button inserts a new record into the database with values from `txtEmail`, `txtAddress`, `txtUserName`, and `txtPassword` textboxes.

**8. Show Data Button (button5)**

csharp

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

{

foreach (Control control in this.Controls)

{

if (control is TextBox)

{

control.Visible = false;

}

else if (control is NumericUpDown)

{

control.Visible = false;

}

else if (control is Label)

{

control.Visible = false;

}

else

{

control.Visible = true;

}

}

SqlConnection con = new SqlConnection("Data Source=DESKTOP-UVGR9G8\\SQLEXPRESS;Initial Catalog=crud;Integrated Security=True;TrustServerCertificate=True");

string readQuery = "SELECT \* FROM crud";

SqlDataAdapter sda = new SqlDataAdapter(readQuery, con);

SqlCommandBuilder cmd = new SqlCommandBuilder(sda);

DataTable dt = new DataTable();

sda.Fill(dt);

GridView1.DataSource = dt;

}

- This button hides all `TextBox`, `NumericUpDown`, and `Label` controls and shows the `DataGridView` with data from the database.

**9. Update Record Button (button3)**

csharp

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

{

using (SqlConnection con = new SqlConnection("Data Source=DESKTOP-UVGR9G8\\SQLEXPRESS;Initial Catalog=crud;Integrated Security=True;TrustServerCertificate=True"))

{

con.Open();

// Start building the update query dynamically

string updateQuery = "UPDATE crud SET ";

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

SqlCommand cmd = new SqlCommand();

// Check each input field and add only the modified ones to the query

if (!string.IsNullOrWhiteSpace(txtEmail.Text))

{

updateFields.Add("email = @email");

cmd.Parameters.AddWithValue("@email", txtEmail.Text);

}

if (!string.IsNullOrWhiteSpace(txtAddress.Text))

{

updateFields.Add("address = @address");

cmd.Parameters.AddWithValue("@address", txtAddress.Text);

}

if (!string.IsNullOrWhiteSpace(txtUserName.Text))

{

updateFields.Add("username = @username");

cmd.Parameters.AddWithValue("@username", txtUserName.Text);

}

if (!string.IsNullOrWhiteSpace(txtPassword.Text))

{

updateFields.Add("password = @password");

cmd.Parameters.AddWithValue("@password", txtPassword.Text);

}

// Ensure at least one field is being updated

if (updateFields.Count == 0)

{

MessageBox.Show("No fields to update. Please provide at least one value.", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Combine the query parts

updateQuery += string.Join(", ", updateFields) + " WHERE id = @id";

cmd.CommandText = updateQuery;

cmd.Connection = con;

// Add the ID parameter

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

// Execute the query

int count = cmd.ExecuteNonQuery();

con.Close();

// Display the result

if (count > 0)

{

MessageBox.Show("Update Successful!", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

MessageBox.Show("No record updated. Please check the ID.", "Info", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

}

- This button updates a record in the database with values provided in the textboxes. It dynamically builds the SQL `UPDATE` statement based on non-empty inputs, ensuring only modified fields are updated.