Вот простой пример кода для WPF-приложения, который демонстрирует шифрование (хеширование) пароля с использованием соли и сохранение данных в базе данных через bcrypt. В этом примере используется SQLite для базы данных.

1. Установите необходимые NuGet-пакеты:

Microsoft.Data.Sqlite (для работы с SQLite)

BCrypt.Net-Next (для хеширования паролей)

2. Код WPF-приложения:

MainWindow.xaml (основное окно с формой регистрации)

<Window x:Class="BcryptExample.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Регистрация" Height="250" Width="300">

<StackPanel Margin="10">

<TextBlock Text="Логин:" />

<TextBox x:Name="txtUsername" Margin="0,0,0,10" />

<TextBlock Text="Пароль:" />

<PasswordBox x:Name="txtPassword" Margin="0,0,0,10" />

<Button Content="Зарегистрировать" Click="BtnRegister\_Click" />

</StackPanel>

</Window>

MainWindow.xaml.cs (логика регистрации)

using System;

using System.Data;

using Microsoft.Data.Sqlite;

using BCrypt.Net;

using System.Windows;

namespace BcryptExample

{

public partial class MainWindow : Window

{

private const string ConnectionString = "Data Source=users.db";

public MainWindow()

{

InitializeComponent();

InitializeDatabase();

}

private void InitializeDatabase()

{

using (var connection = new SqliteConnection(ConnectionString))

{

connection.Open();

var command = connection.CreateCommand();

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Users (

Id INTEGER PRIMARY KEY AUTOINCREMENT,

Username TEXT NOT NULL UNIQUE,

PasswordHash TEXT NOT NULL

)";

command.ExecuteNonQuery();

}

}

private void BtnRegister\_Click(object sender, RoutedEventArgs e)

{

string username = txtUsername.Text;

string password = txtPassword.Password;

if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))

{

MessageBox.Show("Логин и пароль не могут быть пустыми!");

return;

}

try

{

// Генерируем хеш пароля с автоматической солью (bcrypt сам добавляет соль)

string passwordHash = BCrypt.HashPassword(password);

// Сохраняем пользователя в БД

using (var connection = new SqliteConnection(ConnectionString))

{

connection.Open();

var command = connection.CreateCommand();

command.CommandText = "INSERT INTO Users (Username, PasswordHash) VALUES (@username, @passwordHash)";

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

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

command.ExecuteNonQuery();

}

MessageBox.Show("Пользователь успешно зарегистрирован!");

}

catch (Exception ex)

{

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

}

}

}

}

LoginWindow.xaml (окно входа, если нужно)

<Window x:Class="BcryptExample.LoginWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="Вход" Height="200" Width="300">

<StackPanel Margin="10">

<TextBlock Text="Логин:" />

<TextBox x:Name="txtUsername" Margin="0,0,0,10" />

<TextBlock Text="Пароль:" />

<PasswordBox x:Name="txtPassword" Margin="0,0,0,10" />

<Button Content="Войти" Click="BtnLogin\_Click" />

</StackPanel>

</Window>

LoginWindow.xaml.cs (логика входа)

using System.Data;

using Microsoft.Data.Sqlite;

using BCrypt.Net;

using System.Windows;

namespace BcryptExample

{

public partial class LoginWindow : Window

{

private const string ConnectionString = "Data Source=users.db";

public LoginWindow()

{

InitializeComponent();

}

private void BtnLogin\_Click(object sender, RoutedEventArgs e)

{

string username = txtUsername.Text;

string password = txtPassword.Password;

try

{

using (var connection = new SqliteConnection(ConnectionString))

{

connection.Open();

var command = connection.CreateCommand();

command.CommandText = "SELECT PasswordHash FROM Users WHERE Username = @username";

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

using (var reader = command.ExecuteReader())

{

if (reader.Read())

{

string storedHash = reader.GetString("PasswordHash");

// Проверяем пароль с хешем из БД

bool isPasswordValid = BCrypt.Verify(password, storedHash);

if (isPasswordValid)

{

MessageBox.Show("Успешный вход!");

// Открываем главное окно или выполняем другие действия

}

else

{

MessageBox.Show("Неверный пароль!");

}

}

else

{

MessageBox.Show("Пользователь не найден!");

}

}

}

}

catch (Exception ex)

{

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

}

}

}

}

3. Как это работает:

Регистрация:

Пароль хешируется с помощью BCrypt.HashPassword(), который автоматически генерирует соль и включает её в хеш.

Хеш сохраняется в базу данных SQLite.

Вход:

При проверке пароля используется BCrypt.Verify(), который извлекает соль из сохранённого хеша и проверяет соответствие.

4. Особенности bcrypt:

Соль генерируется автоматически и хранится вместе с хешем.

Устойчив к brute-force благодаря медленному хешированию.

Поддержка разных версий (алгоритм адаптируется под новые атаки).

Теперь у вас есть готовый пример безопасного хранения паролей в WPF-приложении! 🚀