Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных систем и технологий

Кафедра «Информационные системы»

Дисциплина «Информационная безопасность»

Лабораторная работа №1

Выполнил:

студент гр. ПИбд-41

Барсуков П. О.

Проверил:  
Мытарев П. В.

Ульяновск, 2025г.

**Индивидуальное задание**

Ограничение на выбираемые пароли: Наличие цифр, знаков препинания и знаков арифметических операций.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Используемый режим шифрования алгоритма DES для шифрования файла | Добавление к ключу случайного значения | Используемый алгоритм хеширования пароля |
| 5 | ECB | Да | MD5 |

Реализованное приложение: <https://github.com/frogg24/InfSec>

**Описание структуры БД**

Таблица Users:

* Id – уникальный индификатор пользователя
* Username – имя пользователя
* Password\_hash – пароль пользователя в виде кэша
* Blocked – флаг блокировки
* Restrictions\_enabled – флаг наличия ограничений
* Min\_length – минимальная длина пароля
* Expiry\_months – срок действия пароля в месяцах

**Описание функций связанных с хэшированием и шифрованием**

Для шифрования файла БД используется алгоритм DES из System.Security.Cryptography в режиме ECB.. Используется добавление случайного значения размером 8 байт, которое генерируется с помощью RandomNumberGenerator.Create(). Ключ получается путем объединения случайного значения и байтов UTF-8 парольной фразы, которые хэшируются и усекаются до первых 8 байтов, которые и становятся ключом.

Для хэширования паролей используется алгоритм MD5 из System.Security.Cryptography.

**Описание мест хранения данных**

users\_encrypted.db – файл, который хранит случайное значение и зашифрованную БД в директории файла проекта

temp\_users.db – файл с расшифрованными данными, который создается во время входа и удаляется во время завершения программы, после перезаписи users\_encrypted.db. Тоже храниться в директории файла проекта.

**Скриншоты экранов**

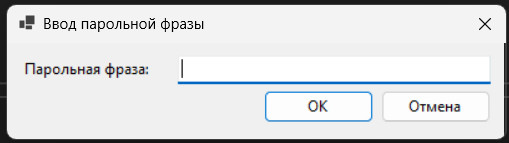


Рисунок 1 - Форма для ввода парольной фразы

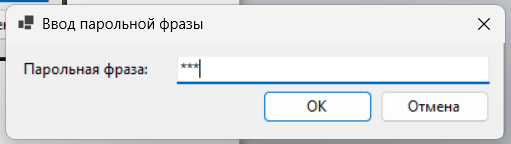


Рисунок 2 - Ввод на форму ввода парольной фразы

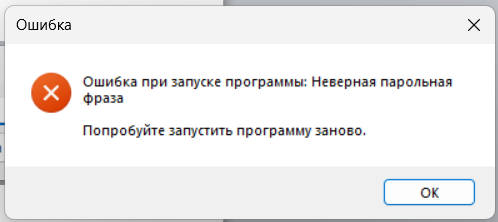


Рисунок 3 - Сообщение о неверном вводе парольной фразы

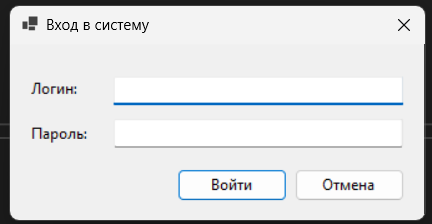


Рисунок 4 - Форма входа

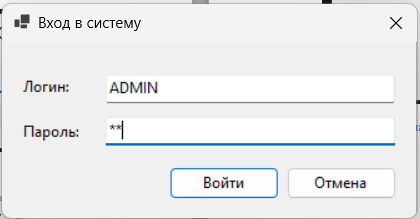


Рисунок 5 - Ввод на форму входа

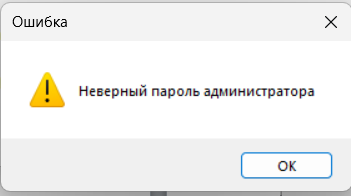


Рисунок 6 - Сообщение о неверном пароле

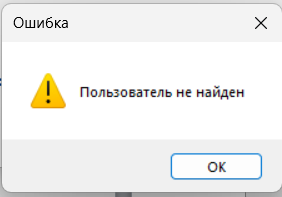


Рисунок 7 - Сообщение о неверном имени пользователя

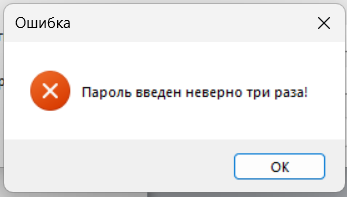


Рисунок 8 - Сообщение о трехкратной ошибке ввода пароля

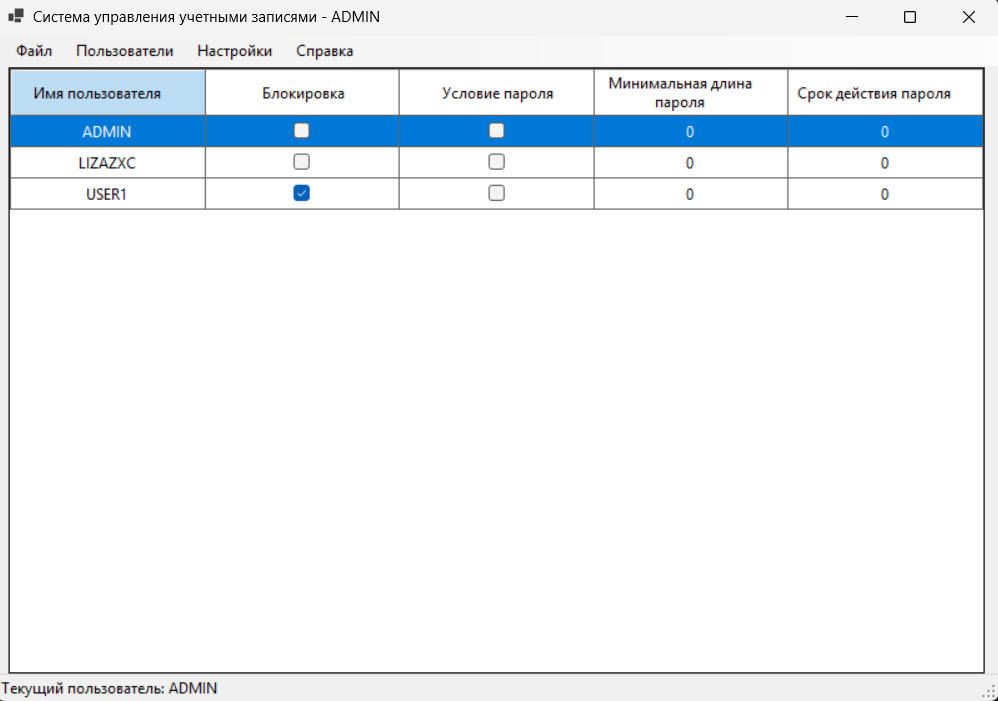


Рисунок 9 - Главная форма от лица администратора

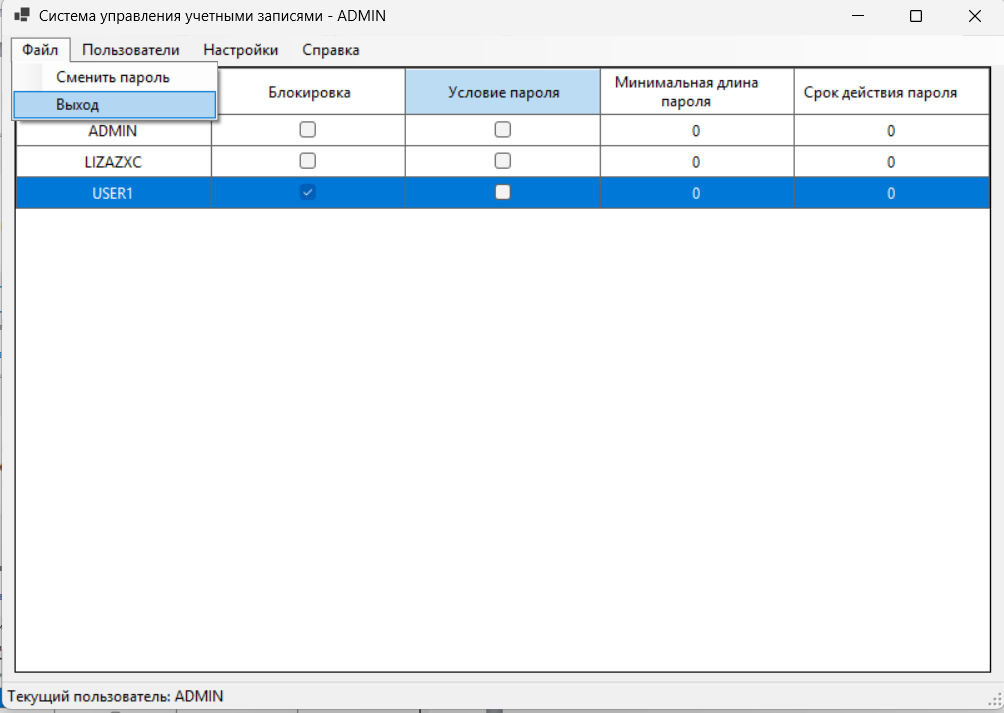


Рисунок 10 - Выход из программы

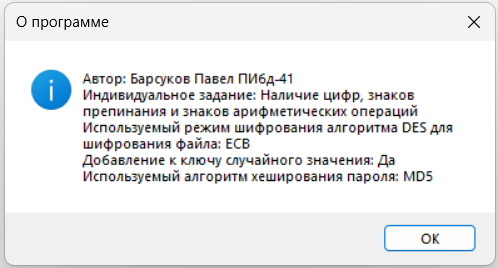


Рисунок 11 - Форма справочной информации

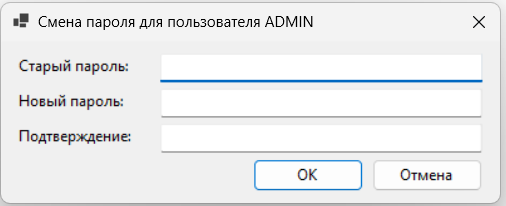


Рисунок 12 - Форма смены пароля

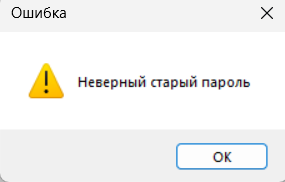


Рисунок 13 - Одна из ошибок смены пароля

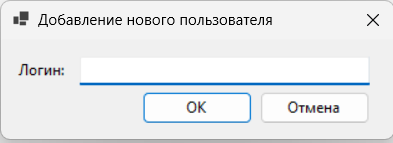


Рисунок 14 - Форма добавления пользователя

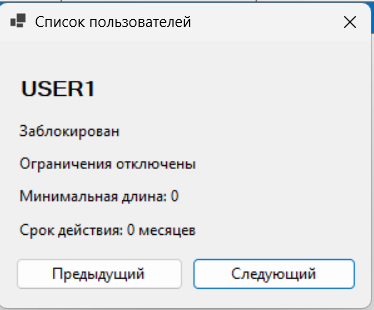


Рисунок 15 - Форма просмотра пользователей по одному

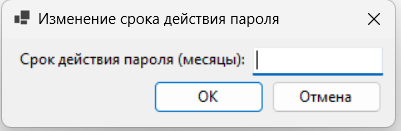


Рисунок 16 - Форма смены срока действия пароля

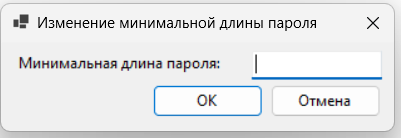


Рисунок 17 - Форма смены минимальной длины пароля

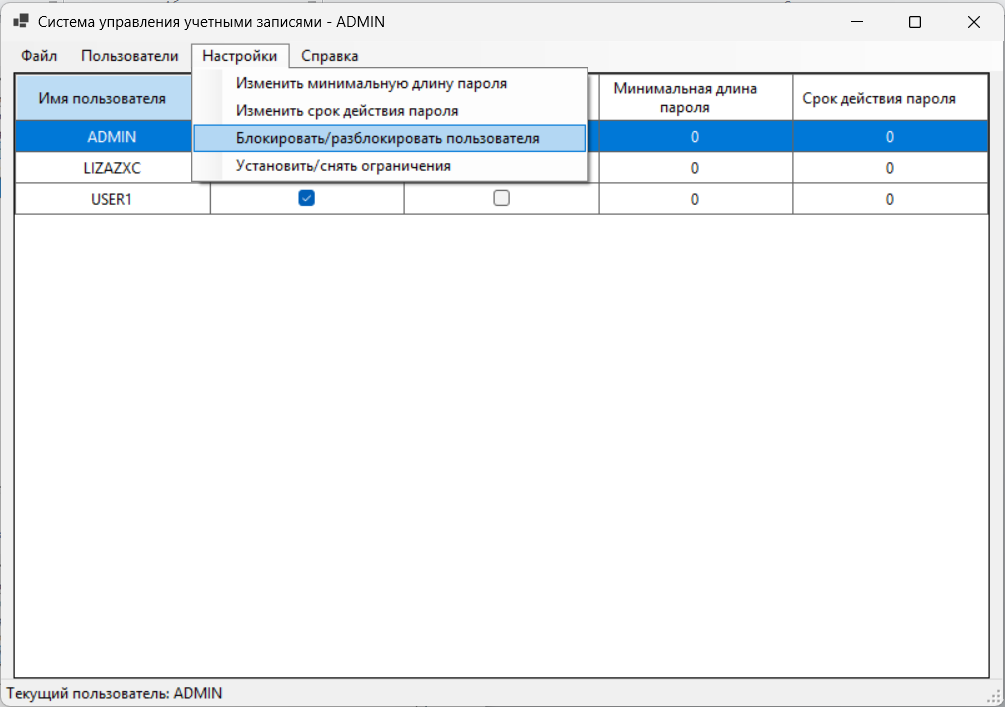


Рисунок 18 - Процесс блокировки/разблокировки пользователя

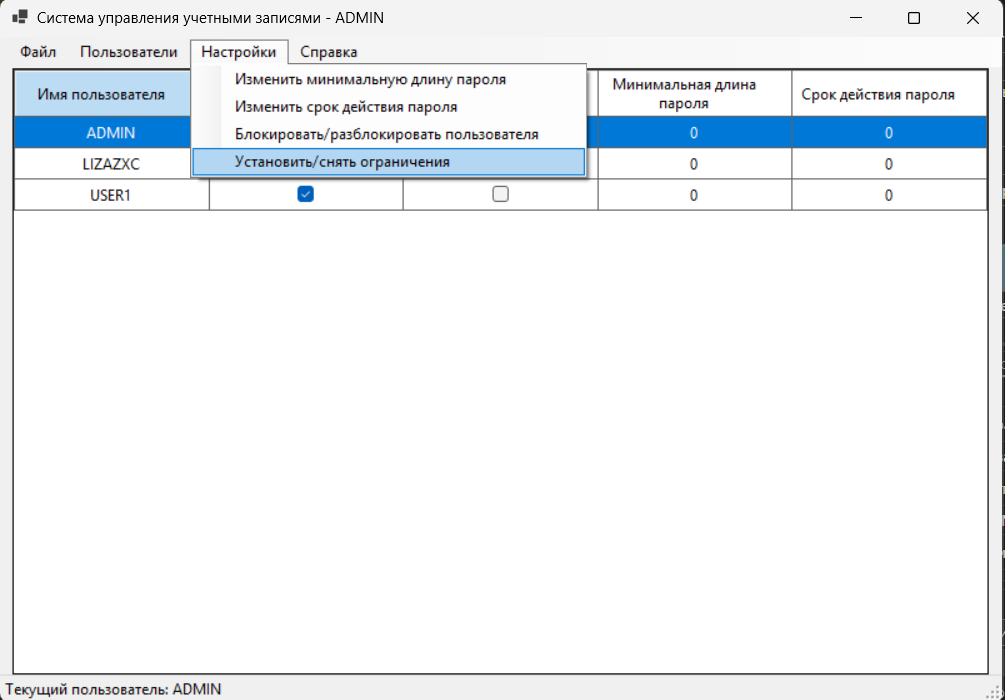


Рисунок 19 – Процесс включения/снятия ограничений на пароль

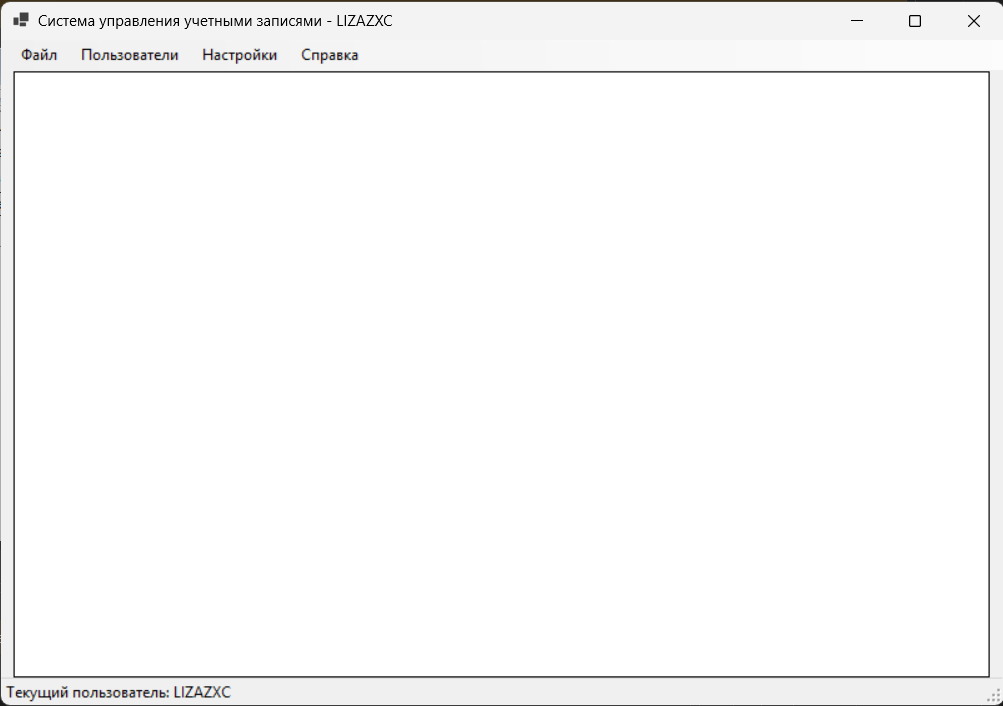


Рисунок 20 - Главная форма от лица пользователя

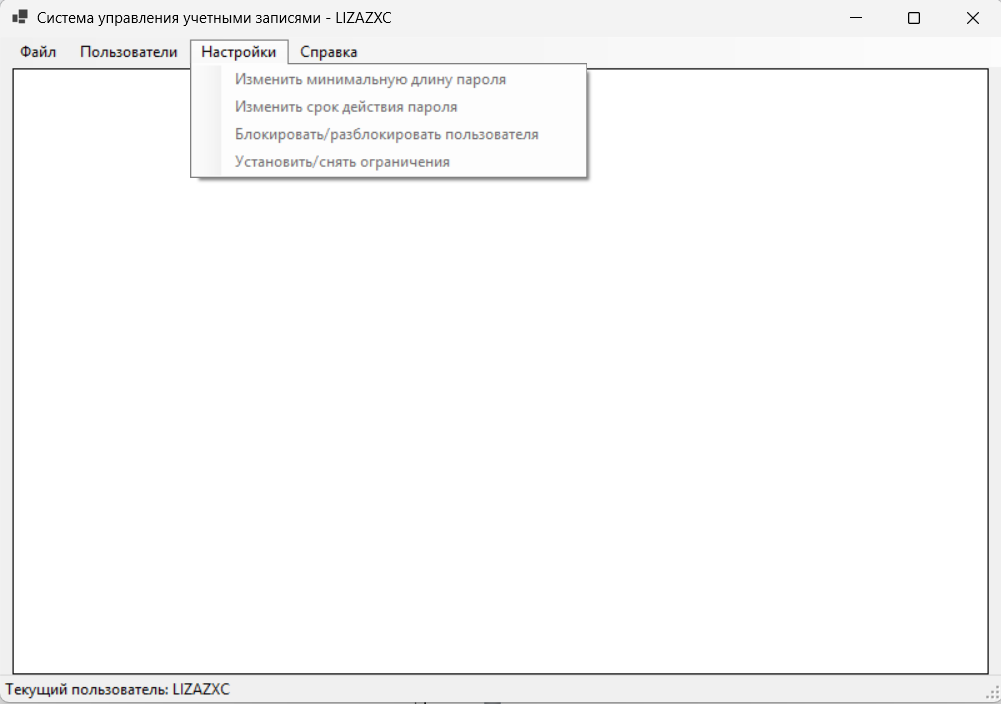


Рисунок 21 - Демонстрация ограничений для обычного пользователя

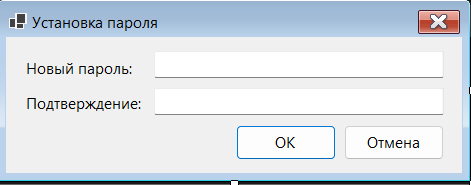


Рисунок 22 - Форма для задания пароля администратора при первом входе

**Листинг кода с комментариями**

// File: InfSec.sln

﻿

Microsoft Visual Studio Solution File, Format Version 12.00

# Visual Studio Version 17

VisualStudioVersion = 17.11.35222.181

MinimumVisualStudioVersion = 10.0.40219.1

Project("{FAE04EC0-301F-11D3-BF4B-00C04F79EFBC}") = "InfSec", "InfSec\InfSec.csproj", "{9222CC8A-06F7-4566-9550-11F10FCA4163}"

EndProject

Global

GlobalSection(SolutionConfigurationPlatforms) = preSolution

Debug|Any CPU = Debug|Any CPU

Release|Any CPU = Release|Any CPU

EndGlobalSection

GlobalSection(ProjectConfigurationPlatforms) = postSolution

{9222CC8A-06F7-4566-9550-11F10FCA4163}.Debug|Any CPU.ActiveCfg = Debug|Any CPU

{9222CC8A-06F7-4566-9550-11F10FCA4163}.Debug|Any CPU.Build.0 = Debug|Any CPU

{9222CC8A-06F7-4566-9550-11F10FCA4163}.Release|Any CPU.ActiveCfg = Release|Any CPU

{9222CC8A-06F7-4566-9550-11F10FCA4163}.Release|Any CPU.Build.0 = Release|Any CPU

EndGlobalSection

GlobalSection(SolutionProperties) = preSolution

HideSolutionNode = FALSE

EndGlobalSection

GlobalSection(ExtensibilityGlobals) = postSolution

SolutionGuid = {51C6E198-096B-44A9-B5B1-2231881C5C15}

EndGlobalSection

EndGlobal

// File: .git\ms-persist.xml

<?xml version="1.0" encoding="utf-8"?>

<PendingCommit>

<CommitComment />

<PinnedBranches />

<PublishPrompt Enabled="True" />

<ActiveAccountPrompt Enabled="True" />

<RepositorySorts />

<CreatePullRequest\_DefaultTargetBranch />

</PendingCommit>

// File: .vs\InfSec\v17\DocumentLayout.backup.json

{

"Version": 1,

"WorkspaceRootPath": "C:\\ulstu\\4\\InfSec\\InfSec\\",

"Documents": [

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}|Form",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}|Form"

},

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}"

},

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\databasemanager.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\databasemanager.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}"

}

],

"DocumentGroupContainers": [

{

"Orientation": 0,

"VerticalTabListWidth": 256,

"DocumentGroups": [

{

"DockedWidth": 200,

"SelectedChildIndex": 3,

"Children": [

{

"$type": "Bookmark",

"Name": "ST:0:0:{b1e99781-ab81-11d0-b683-00aa00a3ee26}"

},

{

"$type": "Bookmark",

"Name": "ST:0:0:{1c4feeaa-4718-4aa9-859d-94ce25d182ba}"

},

{

"$type": "Bookmark",

"Name": "ST:128:0:{116d2292-e37d-41cd-a077-ebacac4c8cc4}"

},

{

"$type": "Document",

"DocumentIndex": 0,

"Title": "MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeDocumentMoniker": "InfSec\\MainForm.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"RelativeToolTip": "InfSec\\MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T15:19:23.108Z",

"EditorCaption": " [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]"

},

{

"$type": "Document",

"DocumentIndex": 1,

"Title": "MainForm.cs",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeDocumentMoniker": "InfSec\\MainForm.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeToolTip": "InfSec\\MainForm.cs",

"ViewState": "AgIAACYAAAAAAAAAAAAAADkAAAAXAAAAAAAAAA==",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T14:39:45.47Z",

"EditorCaption": ""

},

{

"$type": "Document",

"DocumentIndex": 2,

"Title": "DatabaseManager.cs",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\DatabaseManager.cs",

"RelativeDocumentMoniker": "InfSec\\DatabaseManager.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\DatabaseManager.cs",

"RelativeToolTip": "InfSec\\DatabaseManager.cs",

"ViewState": "AgIAABMCAAAAAAAAAAA9wKIBAAArAAAAAAAAAA==",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T12:59:55.479Z",

"EditorCaption": ""

}

]

}

]

}

]

}

// File: .vs\InfSec\v17\DocumentLayout.json

{

"Version": 1,

"WorkspaceRootPath": "C:\\ulstu\\4\\InfSec\\InfSec\\",

"Documents": [

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}|Form",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}|Form"

},

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\mainform.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}"

},

{

"AbsoluteMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|c:\\ulstu\\4\\infsec\\infsec\\infsec\\databasemanager.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}",

"RelativeMoniker": "D:0:0:{9222CC8A-06F7-4566-9550-11F10FCA4163}|InfSec\\InfSec.csproj|solutionrelative:infsec\\databasemanager.cs||{A6C744A8-0E4A-4FC6-886A-064283054674}"

}

],

"DocumentGroupContainers": [

{

"Orientation": 0,

"VerticalTabListWidth": 256,

"DocumentGroups": [

{

"DockedWidth": 200,

"SelectedChildIndex": 3,

"Children": [

{

"$type": "Bookmark",

"Name": "ST:0:0:{b1e99781-ab81-11d0-b683-00aa00a3ee26}"

},

{

"$type": "Bookmark",

"Name": "ST:0:0:{1c4feeaa-4718-4aa9-859d-94ce25d182ba}"

},

{

"$type": "Bookmark",

"Name": "ST:128:0:{116d2292-e37d-41cd-a077-ebacac4c8cc4}"

},

{

"$type": "Document",

"DocumentIndex": 0,

"Title": "MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeDocumentMoniker": "InfSec\\MainForm.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"RelativeToolTip": "InfSec\\MainForm.cs [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T15:19:23.108Z",

"EditorCaption": " [\u041A\u043E\u043D\u0441\u0442\u0440\u0443\u043A\u0442\u043E\u0440]"

},

{

"$type": "Document",

"DocumentIndex": 1,

"Title": "MainForm.cs",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeDocumentMoniker": "InfSec\\MainForm.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\MainForm.cs",

"RelativeToolTip": "InfSec\\MainForm.cs",

"ViewState": "AgIAACYAAAAAAAAAAAAAADkAAAAXAAAAAAAAAA==",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T14:39:45.47Z",

"EditorCaption": ""

},

{

"$type": "Document",

"DocumentIndex": 2,

"Title": "DatabaseManager.cs",

"DocumentMoniker": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\DatabaseManager.cs",

"RelativeDocumentMoniker": "InfSec\\DatabaseManager.cs",

"ToolTip": "C:\\ulstu\\4\\InfSec\\InfSec\\InfSec\\DatabaseManager.cs",

"RelativeToolTip": "InfSec\\DatabaseManager.cs",

"ViewState": "AgIAABMCAAAAAAAAAAA9wKIBAAArAAAAAAAAAA==",

"Icon": "ae27a6b0-e345-4288-96df-5eaf394ee369.000738|",

"WhenOpened": "2025-10-18T12:59:55.479Z",

"EditorCaption": ""

}

]

}

]

}

]

}

// File: InfSec\AddUserDialog.cs

﻿using System;

using System.Windows.Forms;

namespace InfSec

{

public partial class AddUserDialog : Form

{

public string Username { get; private set; }

public AddUserDialog()

{

InitializeComponent();

this.Text = "Добавление нового пользователя";

}

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

{

// Считывание имени

string username = txtUsername.Text.Trim();

// Если поле пустое

if (string.IsNullOrEmpty(username))

{

MessageBox.Show("Введите имя пользователя", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Проверка на повтор имени пользователя

if (DatabaseManager.UserExists(username))

{

MessageBox.Show("Пользователь с таким именем уже существует", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Закрытие формы

Username = username;

DialogResult = DialogResult.OK;

Close();

}

// Закрытие формы по клику

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

{

DialogResult = DialogResult.Cancel;

Close();

}

}

}

// File: InfSec\AddUserDialog.Designer.cs

﻿namespace InfSec

{

partial class AddUserDialog

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label1 = new Label();

txtUsername = new TextBox();

btnOK = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(13, 23);

label1.Margin = new Padding(4, 0, 4, 0);

label1.Name = "label1";

label1.Size = new Size(55, 20);

label1.TabIndex = 0;

label1.Text = "Логин:";

//

// txtUsername

//

txtUsername.Location = new Point(72, 19);

txtUsername.Margin = new Padding(4, 5, 4, 5);

txtUsername.Name = "txtUsername";

txtUsername.Size = new Size(265, 27);

txtUsername.TabIndex = 1;

//

// btnOK

//

btnOK.Location = new Point(129, 56);

btnOK.Margin = new Padding(4, 5, 4, 5);

btnOK.Name = "btnOK";

btnOK.Size = new Size(100, 35);

btnOK.TabIndex = 2;

btnOK.Text = "ОК";

btnOK.UseVisualStyleBackColor = true;

btnOK.Click += btnOK\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(237, 56);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 3;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// AddUserDialog

//

AcceptButton = btnOK;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(359, 106);

Controls.Add(btnCancel);

Controls.Add(btnOK);

Controls.Add(txtUsername);

Controls.Add(label1);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "AddUserDialog";

StartPosition = FormStartPosition.CenterScreen;

Text = "Добавление нового пользователя";

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox txtUsername;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\ChangePasswordDialog.cs

﻿using System;

using System.Windows.Forms;

namespace InfSec

{

public partial class ChangePasswordDialog : Form

{

private string username;

public ChangePasswordDialog(string username)

{

InitializeComponent();

this.username = username;

this.Text = $"Смена пароля для пользователя {username}";

}

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

{

// Запись паролей

string oldPassword = txtOldPassword.Text;

string newPassword1 = txtNewPassword1.Text;

string newPassword2 = txtNewPassword2.Text;

// Если поля заполнены не верно

if (!DatabaseManager.ValidateUser(username, oldPassword))

{

MessageBox.Show("Неверный старый пароль", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (string.IsNullOrEmpty(newPassword1))

{

MessageBox.Show("Введите новый пароль", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (newPassword1 != newPassword2)

{

MessageBox.Show("Новые пароли не совпадают", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Проверяем, если минимальная длина больше нуля, то проверяем длину нового пароля

int minLength = DatabaseManager.GetUserMinLength(username);

if (minLength > 0 && newPassword1.Length < minLength)

{

MessageBox.Show($"Пароль должен быть не короче {minLength} символов", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Если ограничения на пароль включены для пользователя

if (DatabaseManager.GetUserRestrictions(username))

{

// Проверяем требования к паролю

if (!ValidatePasswordRequirements(newPassword1))

{

MessageBox.Show("Пароль не соответствует требованиям:\n" +

"- Наличие цифр\n" +

"- Наличие знаков препинания\n" +

"- Наличие знаков арифметических операций",

"Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

}

// Запись нового пароля

DatabaseManager.SetUserPassword(username, newPassword1);

// Закрытие формы

DialogResult = DialogResult.OK;

Close();

}

/// <summary>

/// Метод валидации пароля по индивидуальному заданию

/// </summary>

/// <param name="password">Пароль для валидации</param>

/// <returns>True, есди пароль прошел валидацию, иначе - False</returns>

private bool ValidatePasswordRequirements(string password)

{

// Проверяем наличие цифр

bool hasDigits = false;

foreach (char c in password)

{

if (char.IsDigit(c))

{

hasDigits = true;

break;

}

}

// Проверяем наличие знаков препинания

bool hasPunctuation = false;

foreach (char c in password)

{

if (char.IsPunctuation(c))

{

hasPunctuation = true;

break;

}

}

// Проверяем наличие знаков арифметических операций

bool hasArithmetic = false;

foreach (char c in password)

{

if (c == '+' || c == '-' || c == '\*' || c == '/')

{

hasArithmetic = true;

break;

}

}

return hasDigits && hasPunctuation && hasArithmetic;

}

// Закрытие формы по клику

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

{

DialogResult = DialogResult.Cancel;

Close();

}

}

}

// File: InfSec\ChangePasswordDialog.Designer.cs

﻿namespace InfSec

{

partial class ChangePasswordDialog

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label1 = new Label();

label2 = new Label();

label3 = new Label();

txtOldPassword = new TextBox();

txtNewPassword1 = new TextBox();

txtNewPassword2 = new TextBox();

btnOK = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(13, 16);

label1.Margin = new Padding(4, 0, 4, 0);

label1.Name = "label1";

label1.Size = new Size(119, 20);

label1.TabIndex = 0;

label1.Text = "Старый пароль:";

//

// label2

//

label2.AutoSize = true;

label2.Location = new Point(13, 54);

label2.Margin = new Padding(4, 0, 4, 0);

label2.Name = "label2";

label2.Size = new Size(115, 20);

label2.TabIndex = 1;

label2.Text = "Новый пароль:";

//

// label3

//

label3.AutoSize = true;

label3.Location = new Point(13, 91);

label3.Margin = new Padding(4, 0, 4, 0);

label3.Name = "label3";

label3.Size = new Size(124, 20);

label3.TabIndex = 2;

label3.Text = "Подтверждение:";

//

// txtOldPassword

//

txtOldPassword.Location = new Point(145, 14);

txtOldPassword.Margin = new Padding(4, 5, 4, 5);

txtOldPassword.Name = "txtOldPassword";

txtOldPassword.PasswordChar = '\*';

txtOldPassword.Size = new Size(293, 27);

txtOldPassword.TabIndex = 3;

//

// txtNewPassword1

//

txtNewPassword1.Location = new Point(145, 51);

txtNewPassword1.Margin = new Padding(4, 5, 4, 5);

txtNewPassword1.Name = "txtNewPassword1";

txtNewPassword1.PasswordChar = '\*';

txtNewPassword1.Size = new Size(293, 27);

txtNewPassword1.TabIndex = 4;

//

// txtNewPassword2

//

txtNewPassword2.Location = new Point(145, 88);

txtNewPassword2.Margin = new Padding(4, 5, 4, 5);

txtNewPassword2.Name = "txtNewPassword2";

txtNewPassword2.PasswordChar = '\*';

txtNewPassword2.Size = new Size(293, 27);

txtNewPassword2.TabIndex = 5;

//

// btnOK

//

btnOK.Location = new Point(230, 125);

btnOK.Margin = new Padding(4, 5, 4, 5);

btnOK.Name = "btnOK";

btnOK.Size = new Size(100, 35);

btnOK.TabIndex = 6;

btnOK.Text = "ОК";

btnOK.UseVisualStyleBackColor = true;

btnOK.Click += btnOK\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(338, 125);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 7;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// ChangePasswordDialog

//

AcceptButton = btnOK;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(456, 174);

Controls.Add(btnCancel);

Controls.Add(btnOK);

Controls.Add(txtNewPassword2);

Controls.Add(txtNewPassword1);

Controls.Add(txtOldPassword);

Controls.Add(label3);

Controls.Add(label2);

Controls.Add(label1);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "ChangePasswordDialog";

StartPosition = FormStartPosition.CenterScreen;

Text = "Смена пароля";

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.TextBox txtOldPassword;

private System.Windows.Forms.TextBox txtNewPassword1;

private System.Windows.Forms.TextBox txtNewPassword2;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\ChangeQuantityDialog.cs

﻿using System;

using System.Data.Common;

using System.Windows.Forms;

namespace InfSec

{

public partial class ChangeQuantityDialog : Form

{

public string Username { get; private set; }

public string ParameterType { get; set; } // "expiry" или "minlength"

public int Value { get; private set; }

public ChangeQuantityDialog(string username, string parameterType)

{

InitializeComponent();

this.Username = username;

this.ParameterType = parameterType;

this.Text = parameterType == "expiry" ? "Изменение срока действия пароля" : "Изменение минимальной длины пароля";

// Настройка заголовка и метки в зависимости от типа параметра

if (parameterType == "expiry")

{

label2.Text = "Срок действия пароля (месяцы):";

this.Text = "Изменение срока действия пароля";

}

else

{

label2.Text = "Минимальная длина пароля:";

this.Text = "Изменение минимальной длины пароля";

}

}

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

{

// Считывание и проверка параметра

int value;

if (!int.TryParse(newQuantity.Text, out value))

{

MessageBox.Show("Введите корректное значение", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (value < 0)

{

MessageBox.Show("Значение не может быть отрицательным", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Передача параметра и закрытие формы

Value = value;

DialogResult = DialogResult.OK;

Close();

}

// Закрытие формы по клику

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

{

DialogResult = DialogResult.Cancel;

Close();

}

}

}

// File: InfSec\ChangeQuantityDialog.Designer.cs

﻿namespace InfSec

{

partial class ChangeQuantityDialog

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label2 = new Label();

newQuantity = new TextBox();

btnOK = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label2

//

label2.AutoSize = true;

label2.Location = new Point(13, 12);

label2.Margin = new Padding(4, 0, 4, 0);

label2.Name = "label2";

label2.Size = new Size(207, 20);

label2.TabIndex = 2;

label2.Text = "Новое значение параметра:";

//

// newQuantity

//

newQuantity.Location = new Point(228, 9);

newQuantity.Margin = new Padding(4, 5, 4, 5);

newQuantity.Name = "newQuantity";

newQuantity.Size = new Size(118, 27);

newQuantity.TabIndex = 3;

//

// btnOK

//

btnOK.Location = new Point(138, 46);

btnOK.Margin = new Padding(4, 5, 4, 5);

btnOK.Name = "btnOK";

btnOK.Size = new Size(100, 35);

btnOK.TabIndex = 4;

btnOK.Text = "ОК";

btnOK.UseVisualStyleBackColor = true;

btnOK.Click += btnOK\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(246, 46);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 5;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// ChangeQuantityDialog

//

AcceptButton = btnOK;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(359, 95);

Controls.Add(btnCancel);

Controls.Add(btnOK);

Controls.Add(newQuantity);

Controls.Add(label2);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "ChangeQuantityDialog";

StartPosition = FormStartPosition.CenterScreen;

Text = "Изменение срока действия пароля";

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox newQuantity;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\DatabaseManager.cs

﻿using System;

using System.Data;

using System.Data.SQLite;

using System.IO;

using System.Security.Cryptography;

using System.Text;

namespace InfSec

{

public static class DatabaseManager

{

private static string encryptedDbPath = "users\_encrypted.db";

private static string tempDbPath = "temp\_users.db";

private static string currentDbPath = "";

private static string currentPasswordPhrase = "";

private const int SALT\_SIZE = 8; // Размер соли для DES

/// <summary>

/// Метод, который при первом запуске инициирует создание БД и записи администратора.

/// При последующих запусках метод расшифрует существующую БД и проверит

/// начилие записи администратора

/// </summary>

/// <param name="passwordPhrase">Введенная пользователем парольная фраза</param>

public static void InitializeDatabase(string passwordPhrase)

{

currentPasswordPhrase = passwordPhrase;

if (!File.Exists(encryptedDbPath))

{

// Первый запуск

CreateDatabase(tempDbPath);

currentDbPath = tempDbPath;

// создаём ADMIN только в новой базе

CreateAdminUser();

// Сохраняем и шифруем базу с новой солью

SaveChangesAndEncrypt(); // Это создаст зашифрованный файл с солью

Console.WriteLine("Новая зашифрованная база данных создана.");

}

else

{

// Повторный запуск: расшифровка

if (!DecryptDatabase()) throw new Exception("Неверная парольная фраза");

currentDbPath = tempDbPath;

// Проверка наличия ADMIN

if (!UserExists("ADMIN"))

{

throw new Exception("Неверная парольная фраза (учетная запись ADMIN не найдена)");

}

}

}

/// <summary>

/// Метод создания БД с единственной сущностью User

/// </summary>

/// <param name="path">Путь для сохранения БД</param>

private static void CreateDatabase(string path)

{

if (File.Exists(path)) File.Delete(path);

SQLiteConnection.CreateFile(path);

using (var connection = new SQLiteConnection($"Data Source={path};Version=3;"))

{

connection.Open();

string createTableSql = @"CREATE TABLE IF NOT EXISTS users (

id INTEGER PRIMARY KEY AUTOINCREMENT,

username TEXT UNIQUE NOT NULL,

password\_hash TEXT,

blocked BOOLEAN DEFAULT 0,

restrictions\_enabled BOOLEAN DEFAULT 0,

min\_length INTEGER DEFAULT 0,

expiry\_months INTEGER DEFAULT 0

)";

using (var command = new SQLiteCommand(createTableSql, connection)) command.ExecuteNonQuery();

}

}

/// <summary>

/// Метод для создания пользователя ADMIN, если он не существует

/// </summary>

public static void CreateAdminUser()

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

// Создаём ADMIN только если его ещё нет

string sql = "INSERT OR IGNORE INTO users (username, password\_hash) VALUES ('ADMIN', '')";

using (var command = new SQLiteCommand(sql, connection))

command.ExecuteNonQuery();

}

}

/// <summary>

/// Метод проверки правильности пароля пользователя для входа

/// Введенный пароля, хэширутеся и сравнивается с хэшем в БД

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="password">Введенный пароль</param>

/// <returns>True, если пароль верен, иначе - False</returns>

public static bool ValidateUser(string username, string password)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT password\_hash FROM users WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

var result = command.ExecuteScalar();

// Если в базе null, трактуем это как пустую строку

string storedHash = result == null || result == DBNull.Value ? "" : result.ToString();

// Если пароль пустой вход разрешён только при пустом хэше

if (string.IsNullOrEmpty(password))

{

return string.IsNullOrEmpty(storedHash);

}

// Проверка хэша

string inputHash = ComputeMD5Hash(password);

return storedHash == inputHash;

}

}

}

/// <summary>

/// Метод проверки изменения пароля пользователя для входа

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="password">Новый пароль</param>

public static void SetUserPassword(string username, string password)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "UPDATE users SET password\_hash = @password\_hash WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

command.Parameters.AddWithValue("@password\_hash", ComputeMD5Hash(password));

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод для получения всех пользователей

/// </summary>

/// <returns>DataTable - список пользователей</returns>

public static DataTable GetAllUsers()

{

DataTable dt = new DataTable();

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT username, blocked, restrictions\_enabled, min\_length, expiry\_months FROM users";

using (var adapter = new SQLiteDataAdapter(sql, connection))

adapter.Fill(dt);

}

return dt;

}

/// <summary>

/// Метод проверки на существование пользователя с таким именем пользователя

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <returns>True, если пользователя существует, иначе - False</returns>

public static bool UserExists(string username)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT COUNT(\*) FROM users WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

var result = command.ExecuteScalar();

return Convert.ToInt32(result) > 0;

}

}

}

/// <summary>

/// Метод добавления пользователя в БД с пустым паролем

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <returns>True, если пароль верен, иначе - False</returns>

public static void AddUser(string username)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "INSERT INTO users (username, password\_hash) VALUES (@username, '')";

using (var command = new SQLiteCommand(sql, connection))

{

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод проверки заюлокирован пользователь или нет по имени пользователя

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <returns>True, если пользователь заблокирован, иначе - False</returns>

public static bool IsUserBlocked(string username)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT blocked FROM users WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

var result = command.ExecuteScalar();

if (result == null || result == DBNull.Value)

return false;

return Convert.ToBoolean(result);

}

}

}

/// <summary>

/// Метод блокировки/разблокировки пользователя

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="blocked">Поле, указывающее заблокировать или разблокировать</param>

public static void SetUserBlocked(string username, bool blocked)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "UPDATE users SET blocked = @blocked WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод включения/выключения проверки на ограничения для задания пароля

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="enabled">Поле,отображающее включить ограничения или снять</param>

public static void SetUserRestrictions(string username, bool enabled)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "UPDATE users SET restrictions\_enabled = @enabled WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод проверки включены ли ограничения на пароля для пользователя

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <returns>True, если ограничения для пользователя включены, иначе - False</returns>

public static bool GetUserRestrictions(string username)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT restrictions\_enabled FROM users WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

var result = command.ExecuteScalar();

if (result == null || result == DBNull.Value)

return false;

return Convert.ToBoolean(result);

}

}

}

/// <summary>

/// Метод пзадания минимальной длины пароля для пользователя

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="minLength">Новая минимальная длина пароля</param>

/// <returns>True, если пароль верен, иначе - False</returns>

public static void SetUserMinLength(string username, int minLength)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "UPDATE users SET min\_length = @min\_length WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

command.Parameters.AddWithValue("@min\_length", minLength);

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод проверки правильности пароля полязователя для входа

/// Введенный пароля, хэширутеся и сравнивается с хэшем в БД

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <param name="password">Введенный пароль</param>

/// <returns>True, если пароль верен, иначе - False</returns>

public static void SetUserExpiry(string username, int expiryMonths)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "UPDATE users SET expiry\_months = @expiry\_months WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

command.Parameters.AddWithValue("@expiry\_months", expiryMonths);

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

command.ExecuteNonQuery();

}

}

}

/// <summary>

/// Метод расшифрования базы данных с использованием алгоритма DES в режиме ECB.

/// Считывает соль из начала зашифрованного файла.

/// </summary>

/// <returns>True, если расшифровка успешна, иначе False</returns>

private static bool DecryptDatabase()

{

try

{

if (File.Exists(tempDbPath))

{

File.Delete(tempDbPath);

}

if (!File.Exists(encryptedDbPath))

{

return false;

}

byte[] encryptedBytes = File.ReadAllBytes(encryptedDbPath);

// Проверяем длину файла (должна быть >= размеру соли)

if (encryptedBytes.Length < SALT\_SIZE)

{

Console.WriteLine("Ошибка: Зашифрованный файл слишком короткий.");

return false;

}

// Читаем соль из начала файла

byte[] salt = new byte[SALT\_SIZE];

Array.Copy(encryptedBytes, 0, salt, 0, SALT\_SIZE);

// Остальная часть - зашифрованные данные

int dataLength = encryptedBytes.Length - SALT\_SIZE;

byte[] encryptedData = new byte[dataLength];

Array.Copy(encryptedBytes, SALT\_SIZE, encryptedData, 0, dataLength);

// Генерируем ключ, используя соль

byte[] key = GenerateKeyFromPhraseWithSalt(currentPasswordPhrase, salt);

// Расшифровываем данные

byte[] decryptedBytes = DecryptDataWithKey(encryptedData, key);

File.WriteAllBytes(tempDbPath, decryptedBytes);

currentDbPath = tempDbPath;

return true;

}

catch (Exception ex) // Лучше ловить конкретные исключения, например CryptographicException

{

Console.WriteLine("Ошибка при расшифровке: " + ex.Message);

return false;

}

}

/// <summary>

/// Метод сохранения изменений и шифрования базы данных при завершении работы программы.

/// Генерирует случайную соль и добавляет её в начало зашифрованного файла.

/// </summary>

public static void SaveChangesAndEncrypt()

{

try

{

if (File.Exists(tempDbPath))

{

// Читаем содержимое временной базы

byte[] fileBytes = File.ReadAllBytes(tempDbPath);

// Генерируем случайную соль

byte[] salt = GenerateRandomSalt();

// Генерируем ключ, используя соль

byte[] key = GenerateKeyFromPhraseWithSalt(currentPasswordPhrase, salt);

// Шифруем данные

byte[] encryptedData = EncryptDataWithKey(fileBytes, key);

// Создаем общий массив: соль + зашифрованные данные

byte[] finalEncryptedBytes = new byte[SALT\_SIZE + encryptedData.Length];

Array.Copy(salt, 0, finalEncryptedBytes, 0, SALT\_SIZE);

Array.Copy(encryptedData, 0, finalEncryptedBytes, SALT\_SIZE, encryptedData.Length);

// Записываем в зашифрованный файл

File.WriteAllBytes(encryptedDbPath, finalEncryptedBytes);

// Удаляем временную БД

File.Delete(tempDbPath);

}

}

catch (Exception ex)

{

Console.WriteLine("Ошибка при сохранении: " + ex.Message);

throw; // Перебрасываем исключение, чтобы вызывающий код мог обработать его

}

}

/// <summary>

/// Метод шифрования данных с использованием алгоритма DES в режиме ECB.

/// </summary>

/// <param name="data">Данные для шифрования</param>

/// <param name="key">Ключ шифрования</param>

/// <returns>Зашифрованные данные (без соли)</returns>

private static byte[] EncryptDataWithKey(byte[] data, byte[] key)

{

using (DESCryptoServiceProvider des = new DESCryptoServiceProvider())

{

des.Mode = CipherMode.ECB;

des.Padding = PaddingMode.PKCS7;

using (ICryptoTransform encryptor = des.CreateEncryptor(key, new byte[8])) // IV не используется в ECB, но нужен для создания трансформа

{

return encryptor.TransformFinalBlock(data, 0, data.Length);

}

}

}

/// <summary>

/// Метод расшифрования данных с использованием алгоритма DES в режиме ECB.

/// </summary>

/// <param name="encryptedData">Зашифрованные данные (без соли)</param>

/// <param name="key">Ключ шифрования</param>

/// <returns>Расшифрованные данные</returns>

private static byte[] DecryptDataWithKey(byte[] encryptedData, byte[] key)

{

using (DESCryptoServiceProvider des = new DESCryptoServiceProvider())

{

des.Mode = CipherMode.ECB;

des.Padding = PaddingMode.PKCS7;

using (ICryptoTransform decryptor = des.CreateDecryptor(key, new byte[8])) // IV не используется в ECB

{

return decryptor.TransformFinalBlock(encryptedData, 0, encryptedData.Length);

}

}

}

/// <summary>

/// Метод генерации случайной соли.

/// </summary>

/// <returns>Случайный массив байт заданного размера</returns>

private static byte[] GenerateRandomSalt()

{

byte[] salt = new byte[SALT\_SIZE];

using (var rng = RandomNumberGenerator.Create())

{

rng.GetBytes(salt);

}

return salt;

}

/// <summary>

/// Метод генерации ключа шифрования из парольной фразы и соли.

/// Простое объединение фразы и соли перед генерацией ключа.

/// </summary>

/// <param name="phrase">Парольная фраза</param>

/// <param name="salt">Соль</param>

/// <returns>Ключ шифрования</returns>

private static byte[] GenerateKeyFromPhraseWithSalt(string phrase, byte[] salt)

{

byte[] phraseBytes = Encoding.UTF8.GetBytes(phrase);

// Объединяем фразу и соль (можно сделать иначе, например, через HMAC)

byte[] combined = new byte[phraseBytes.Length + salt.Length];

Array.Copy(phraseBytes, 0, combined, 0, phraseBytes.Length);

Array.Copy(salt, 0, combined, phraseBytes.Length, salt.Length);

// Вычисляем хэш объединенного значения

using (SHA256 sha256 = SHA256.Create())

{

byte[] hash = sha256.ComputeHash(combined);

// Берем первые 8 байт хэша как ключ (требование DES)

byte[] key = new byte[8];

Array.Copy(hash, 0, key, 0, Math.Min(8, hash.Length));

return key;

}

}

/// <summary>

/// Метод вычисления хэша MD5 для пароля.

/// </summary>

/// <param name="input">Входная строка</param>

/// <returns>Хэш в виде строки</returns>

private static string ComputeMD5Hash(string input)

{

using (MD5 md5 = MD5.Create())

{

byte[] inputBytes = Encoding.UTF8.GetBytes(input);

byte[] hashBytes = md5.ComputeHash(inputBytes);

StringBuilder sb = new StringBuilder();

foreach (byte b in hashBytes) sb.Append(b.ToString("x2"));

return sb.ToString();

}

}

/// <summary>

/// Метод получения минимальной длины пароля для пользователя.

/// </summary>

/// <param name="username">Имя пользователя</param>

/// <returns>Минимальная длина пароля</returns>

public static int GetUserMinLength(string username)

{

using (var connection = new SQLiteConnection($"Data Source={currentDbPath};Version=3;"))

{

connection.Open();

string sql = "SELECT min\_length FROM users WHERE username = @username";

using (var command = new SQLiteCommand(sql, connection))

{

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

var result = command.ExecuteScalar();

if (result == null || result == DBNull.Value)

return 0;

return Convert.ToInt32(result);

}

}

}

}

}

// File: InfSec\InfSec.csproj

﻿<Project Sdk="Microsoft.NET.Sdk">

<PropertyGroup>

<OutputType>WinExe</OutputType>

<TargetFramework>net8.0-windows</TargetFramework>

<Nullable>enable</Nullable>

<UseWindowsForms>true</UseWindowsForms>

<ImplicitUsings>enable</ImplicitUsings>

<PlatformTarget>x64</PlatformTarget>

</PropertyGroup>

<ItemGroup>

<PackageReference Include="System.Data.SQLite.Core" Version="1.0.119" />

</ItemGroup>

<ItemGroup>

<Compile Update="DatabaseManager.cs">

<CopyToOutputDirectory>Always</CopyToOutputDirectory>

</Compile>

<Compile Update="Program.cs">

<CopyToOutputDirectory>Always</CopyToOutputDirectory>

</Compile>

</ItemGroup>

<ItemGroup>

<None Update="app.config">

<CopyToOutputDirectory>Always</CopyToOutputDirectory>

</None>

</ItemGroup>

</Project>

// File: InfSec\LoginForm.Designer.cs

﻿namespace InfSec

{

partial class LoginForm

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label1 = new Label();

label2 = new Label();

txtUsername = new TextBox();

txtPassword = new TextBox();

btnLogin = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(16, 38);

label1.Margin = new Padding(4, 0, 4, 0);

label1.Name = "label1";

label1.Size = new Size(55, 20);

label1.TabIndex = 0;

label1.Text = "Логин:";

//

// label2

//

label2.AutoSize = true;

label2.Location = new Point(16, 85);

label2.Margin = new Padding(4, 0, 4, 0);

label2.Name = "label2";

label2.Size = new Size(65, 20);

label2.TabIndex = 1;

label2.Text = "Пароль:";

//

// txtUsername

//

txtUsername.Location = new Point(95, 34);

txtUsername.Margin = new Padding(4, 5, 4, 5);

txtUsername.Name = "txtUsername";

txtUsername.Size = new Size(265, 27);

txtUsername.TabIndex = 2;

//

// txtPassword

//

txtPassword.Location = new Point(95, 80);

txtPassword.Margin = new Padding(4, 5, 4, 5);

txtPassword.Name = "txtPassword";

txtPassword.PasswordChar = '\*';

txtPassword.Size = new Size(265, 27);

txtPassword.TabIndex = 3;

//

// btnLogin

//

btnLogin.Location = new Point(153, 134);

btnLogin.Margin = new Padding(4, 5, 4, 5);

btnLogin.Name = "btnLogin";

btnLogin.Size = new Size(100, 35);

btnLogin.TabIndex = 4;

btnLogin.Text = "Войти";

btnLogin.UseVisualStyleBackColor = true;

btnLogin.Click += btnLogin\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(261, 134);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 5;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

//

// LoginForm

//

AcceptButton = btnLogin;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(379, 188);

Controls.Add(btnCancel);

Controls.Add(btnLogin);

Controls.Add(txtPassword);

Controls.Add(txtUsername);

Controls.Add(label2);

Controls.Add(label1);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "LoginForm";

StartPosition = FormStartPosition.CenterScreen;

Text = "Вход в систему";

FormClosing += LoginForm\_FormClosing;

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox txtUsername;

private System.Windows.Forms.TextBox txtPassword;

private System.Windows.Forms.Button btnLogin;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\MainForm.cs

﻿using System;

using System.Data;

using System.Windows.Forms;

namespace InfSec

{

public partial class MainForm : Form

{

private string currentUser;

private bool isAdmin;

public MainForm(string username)

{

InitializeComponent();

this.currentUser = username;

this.isAdmin = (username.ToUpper() == "ADMIN");

this.Text = $"Система управления учетными записями - {username}";

UpdateStatus();

LoadUsersList();

// Сокрытие недоступных пунктов меню для обычных пользователей

if (!isAdmin)

{

menuChangePassword.Enabled = true;

menuAddUser.Enabled = false;

menuViewUsers.Enabled = false;

menuBlockUser.Enabled = false;

menuSetRestrictions.Enabled = false;

menuChangeMinLength.Enabled = false;

menuChangeExpiry.Enabled = false;

}

else

{

menuChangePassword.Enabled = true;

menuAddUser.Enabled = true;

menuViewUsers.Enabled = true;

menuBlockUser.Enabled = true;

menuSetRestrictions.Enabled = true;

menuChangeMinLength.Enabled = true;

menuChangeExpiry.Enabled = true;

}

}

// Изменение отображения текущего пользователя

private void UpdateStatus()

{

toolStripStatusLabel1.Text = $"Текущий пользователь: {currentUser}";

}

/// <summary>

/// Метод для загрузки пользователей и настройка таблицы

/// </summary>

private void LoadUsersList()

{

if (!string.Equals(currentUser, "ADMIN"))

{

return;

}

DataTable users = DatabaseManager.GetAllUsers();

lstUsers.DataSource = users;

lstUsers.Columns["username"].HeaderText = "Имя пользователя";

lstUsers.Columns["blocked"].HeaderText = "Блокировка";

lstUsers.Columns["restrictions\_enabled"].HeaderText = "Условие пароля";

lstUsers.Columns["min\_length"].HeaderText = "Минимальная длина пароля";

lstUsers.Columns["expiry\_months"].HeaderText = "Срок действия пароля ";

lstUsers.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

lstUsers.ColumnHeadersDefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;

lstUsers.DefaultCellStyle.Alignment = DataGridViewContentAlignment.MiddleCenter;

}

// Закрытие приложения по клику

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

{

Application.Exit();

}

/// <summary>

/// Метод для отображения справчной информации

/// </summary>

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

{

MessageBox.Show(

"Автор: Барсуков Павел ПИбд-41\n" +

"Индивидуальное задание: Наличие цифр, знаков препинания и знаков арифметических операций\n" +

"Используемый режим шифрования алгоритма DES для шифрования файла: ECB\n" +

"Добавление к ключу случайного значения: Да\n" +

"Используемый алгоритм хеширования пароля: MD5\n",

"О программе",

MessageBoxButtons.OK,

MessageBoxIcon.Information);

}

/// <summary>

/// Метод для смены пароля текущего пользователя

/// </summary>

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

{

ChangePasswordDialog changePwdDialog = new ChangePasswordDialog(currentUser);

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

{

MessageBox.Show("Пароль успешно изменен", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для создания нового пользователя админитсратором

/// </summary>

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

{

if (!isAdmin) return;

AddUserDialog addUserDialog = new AddUserDialog();

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

{

string username = addUserDialog.Username;

DatabaseManager.AddUser(username);

LoadUsersList();

MessageBox.Show($"Пользователь {username} добавлен", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для простомтра всех пользователей по одному

/// </summary>

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

{

if (!isAdmin) return;

ViewUsersDialog viewUsersDialog = new ViewUsersDialog();

viewUsersDialog.ShowDialog();

}

/// <summary>

/// Метод для блокировки и разблокировки выделенного пользователя

/// </summary>

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

{

if (!isAdmin) return;

// Проверяем, есть ли выбранная строка

if (lstUsers.SelectedRows.Count == 0 || lstUsers.SelectedRows[0].Cells["username"].Value == null)

{

MessageBox.Show("Выберите пользователя из списка", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Получаем значение из ячейки "username" выбранной строки

string usernameToBlock = lstUsers.SelectedRows[0].Cells["username"].Value.ToString();

if (!DatabaseManager.UserExists(usernameToBlock))

{

MessageBox.Show("Пользователь не найден", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Проверяем текущее состояние blocked

bool isBlocked = DatabaseManager.IsUserBlocked(usernameToBlock);

string actionText = isBlocked ? "разблокировать" : "заблокировать";

DialogResult result = MessageBox.Show(

$"Вы действительно хотите {actionText} пользователя {usernameToBlock}?",

"Управление блокировкой",

MessageBoxButtons.YesNo,

MessageBoxIcon.Question

);

if (result == DialogResult.Yes)

{

DatabaseManager.SetUserBlocked(usernameToBlock, !isBlocked);

LoadUsersList();

string status = isBlocked ? "разблокирован" : "заблокирован";

MessageBox.Show($"Пользователь {usernameToBlock} {status}", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для включения и снятия ограничений на пароль для выделенного пользователя

/// </summary>

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

{

if (!isAdmin) return;

// Проверяем, что есть выделенная строка в списке пользователей

if (lstUsers.SelectedRows.Count == 0 || lstUsers.SelectedRows[0].Cells["username"].Value == null)

{

MessageBox.Show("Выберите пользователя из списка", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Получаем имя пользователя из выделенной строки

string username = lstUsers.SelectedRows[0].Cells["username"].Value.ToString();

if (!DatabaseManager.UserExists(username))

{

MessageBox.Show("Пользователь не найден", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Проверяем текущее состояние blocked

bool isRestrictionsOn = DatabaseManager.GetUserRestrictions(username);

string actionText = isRestrictionsOn ? "выключить ограничения" : "включить ограничения";

DialogResult result = MessageBox.Show(

$"Вы действительно хотите {actionText} пользователю {username}?", "Управление блокировкой",

MessageBoxButtons.YesNo, MessageBoxIcon.Question

);

if (result == DialogResult.Yes)

{

DatabaseManager.SetUserRestrictions(username, !isRestrictionsOn);

LoadUsersList();

string status = isRestrictionsOn ? "ограничения для пароля сняты" : "ограничения для пароля включены";

MessageBox.Show($"Для пользователя {username} {status}", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для задания минимальной длины пароля выделенного пользователя

/// </summary>

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

{

if (!isAdmin) return;

// Проверяем, что есть выделенная строка в списке пользователей

if (lstUsers.SelectedRows.Count == 0 || lstUsers.SelectedRows[0].Cells["username"].Value == null)

{

MessageBox.Show("Выберите пользователя из списка", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Получаем имя пользователя из выделенной строки

string username = lstUsers.SelectedRows[0].Cells["username"].Value.ToString();

if (!DatabaseManager.UserExists(username))

{

MessageBox.Show("Пользователь не найден", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Показываем диалог для ввода новой минимальной длины

ChangeQuantityDialog paramDialog = new ChangeQuantityDialog(username, "minlength");

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

{

int minLength = paramDialog.Value;

DatabaseManager.SetUserMinLength(username, minLength);

LoadUsersList();

MessageBox.Show($"Минимальная длина пароля для пользователя {username} установлена в {minLength}", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для задания срока действия пароля в месяцах для выделенного пользователя

/// </summary>

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

{

if (!isAdmin) return;

// Проверяем, что есть выделенная строка в списке пользователей

if (lstUsers.SelectedRows.Count == 0 || lstUsers.SelectedRows[0].Cells["username"].Value == null)

{

MessageBox.Show("Выберите пользователя из списка", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Получаем имя пользователя из выделенной строки

string username = lstUsers.SelectedRows[0].Cells["username"].Value.ToString();

if (!DatabaseManager.UserExists(username))

{

MessageBox.Show("Пользователь не найден", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Показываем диалог для ввода нового срока действия

ChangeQuantityDialog paramDialog = new ChangeQuantityDialog(username, "expiry");

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

{

int expiryMonths = paramDialog.Value;

DatabaseManager.SetUserExpiry(username, expiryMonths);

LoadUsersList();

MessageBox.Show($"Срок действия пароля для пользователя {username} установлен в {expiryMonths} месяцев", "Успех", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

/// <summary>

/// Метод для звкрытия формы и приложения

/// </summary>

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

{

// При закрытии основной формы сохраняем изменения и шифруем базу данных

try

{

DatabaseManager.SaveChangesAndEncrypt();

}

catch (Exception ex)

{

MessageBox.Show($"Ошибка при сохранении базы данных: {ex.Message}", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

Application.Exit();

}

}

}

// File: InfSec\MainForm.Designer.cs

﻿namespace InfSec

{

partial class MainForm

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

menuStrip1 = new MenuStrip();

файлToolStripMenuItem = new ToolStripMenuItem();

menuChangePassword = new ToolStripMenuItem();

menuExit = new ToolStripMenuItem();

пользователиToolStripMenuItem = new ToolStripMenuItem();

menuAddUser = new ToolStripMenuItem();

menuViewUsers = new ToolStripMenuItem();

настройкиToolStripMenuItem = new ToolStripMenuItem();

menuChangeMinLength = new ToolStripMenuItem();

menuChangeExpiry = new ToolStripMenuItem();

menuBlockUser = new ToolStripMenuItem();

menuSetRestrictions = new ToolStripMenuItem();

справкаToolStripMenuItem = new ToolStripMenuItem();

menuAbout = new ToolStripMenuItem();

statusStrip1 = new StatusStrip();

toolStripStatusLabel1 = new ToolStripStatusLabel();

lstUsers = new DataGridView();

menuStrip1.SuspendLayout();

statusStrip1.SuspendLayout();

((System.ComponentModel.ISupportInitialize)lstUsers).BeginInit();

SuspendLayout();

//

// menuStrip1

//

menuStrip1.ImageScalingSize = new Size(20, 20);

menuStrip1.Items.AddRange(new ToolStripItem[] { файлToolStripMenuItem, пользователиToolStripMenuItem, настройкиToolStripMenuItem, справкаToolStripMenuItem });

menuStrip1.Location = new Point(0, 0);

menuStrip1.Name = "menuStrip1";

menuStrip1.Padding = new Padding(8, 3, 0, 3);

menuStrip1.Size = new Size(917, 30);

menuStrip1.TabIndex = 0;

menuStrip1.Text = "menuStrip1";

//

// файлToolStripMenuItem

//

файлToolStripMenuItem.DropDownItems.AddRange(new ToolStripItem[] { menuChangePassword, menuExit });

файлToolStripMenuItem.Name = "файлToolStripMenuItem";

файлToolStripMenuItem.Size = new Size(59, 24);

файлToolStripMenuItem.Text = "Файл";

//

// menuChangePassword

//

menuChangePassword.Name = "menuChangePassword";

menuChangePassword.Size = new Size(207, 26);

menuChangePassword.Text = "Сменить пароль";

menuChangePassword.Click += menuChangePassword\_Click;

//

// menuExit

//

menuExit.Name = "menuExit";

menuExit.Size = new Size(207, 26);

menuExit.Text = "Выход";

menuExit.Click += menuExit\_Click;

//

// пользователиToolStripMenuItem

//

пользователиToolStripMenuItem.DropDownItems.AddRange(new ToolStripItem[] { menuAddUser, menuViewUsers });

пользователиToolStripMenuItem.Name = "пользователиToolStripMenuItem";

пользователиToolStripMenuItem.Size = new Size(122, 24);

пользователиToolStripMenuItem.Text = "Пользователи";

//

// menuAddUser

//

menuAddUser.Name = "menuAddUser";

menuAddUser.Size = new Size(346, 26);

menuAddUser.Text = "Добавить пользователя";

menuAddUser.Click += menuAddUser\_Click;

//

// menuViewUsers

//

menuViewUsers.Name = "menuViewUsers";

menuViewUsers.Size = new Size(346, 26);

menuViewUsers.Text = "Просмотреть список пользователей";

menuViewUsers.Click += menuViewUsers\_Click;

//

// настройкиToolStripMenuItem

//

настройкиToolStripMenuItem.DropDownItems.AddRange(new ToolStripItem[] { menuChangeMinLength, menuChangeExpiry, menuBlockUser, menuSetRestrictions });

настройкиToolStripMenuItem.Name = "настройкиToolStripMenuItem";

настройкиToolStripMenuItem.Size = new Size(98, 24);

настройкиToolStripMenuItem.Text = "Настройки";

//

// menuChangeMinLength

//

menuChangeMinLength.Name = "menuChangeMinLength";

menuChangeMinLength.Size = new Size(402, 26);

menuChangeMinLength.Text = "Изменить минимальную длину пароля";

menuChangeMinLength.Click += menuChangeMinLength\_Click;

//

// menuChangeExpiry

//

menuChangeExpiry.Name = "menuChangeExpiry";

menuChangeExpiry.Size = new Size(402, 26);

menuChangeExpiry.Text = "Изменить срок действия пароля";

menuChangeExpiry.Click += menuChangeExpiry\_Click;

//

// menuBlockUser

//

menuBlockUser.Name = "menuBlockUser";

menuBlockUser.Size = new Size(402, 26);

menuBlockUser.Text = "Блокировать/разблокировать пользователя";

menuBlockUser.Click += menuBlockUser\_Click;

//

// menuSetRestrictions

//

menuSetRestrictions.Name = "menuSetRestrictions";

menuSetRestrictions.Size = new Size(402, 26);

menuSetRestrictions.Text = "Установить/снять ограничения";

menuSetRestrictions.Click += menuSetRestrictions\_Click;

//

// справкаToolStripMenuItem

//

справкаToolStripMenuItem.DropDownItems.AddRange(new ToolStripItem[] { menuAbout });

справкаToolStripMenuItem.Name = "справкаToolStripMenuItem";

справкаToolStripMenuItem.Size = new Size(81, 24);

справкаToolStripMenuItem.Text = "Справка";

//

// menuAbout

//

menuAbout.Name = "menuAbout";

menuAbout.Size = new Size(187, 26);

menuAbout.Text = "О программе";

menuAbout.Click += menuAbout\_Click;

//

// statusStrip1

//

statusStrip1.ImageScalingSize = new Size(20, 20);

statusStrip1.Items.AddRange(new ToolStripItem[] { toolStripStatusLabel1 });

statusStrip1.Location = new Point(0, 683);

statusStrip1.Name = "statusStrip1";

statusStrip1.Padding = new Padding(1, 0, 19, 0);

statusStrip1.Size = new Size(917, 26);

statusStrip1.TabIndex = 1;

statusStrip1.Text = "statusStrip1";

//

// toolStripStatusLabel1

//

toolStripStatusLabel1.Name = "toolStripStatusLabel1";

toolStripStatusLabel1.Size = new Size(151, 20);

toolStripStatusLabel1.Text = "toolStripStatusLabel1";

//

// lstUsers

//

lstUsers.AllowUserToAddRows = false;

lstUsers.AllowUserToDeleteRows = false;

lstUsers.BackgroundColor = SystemColors.ButtonHighlight;

lstUsers.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.AutoSize;

lstUsers.Location = new Point(12, 33);

lstUsers.MultiSelect = false;

lstUsers.Name = "lstUsers";

lstUsers.ReadOnly = true;

lstUsers.RowHeadersVisible = false;

lstUsers.RowHeadersWidth = 51;

lstUsers.SelectionMode = DataGridViewSelectionMode.FullRowSelect;

lstUsers.Size = new Size(893, 647);

lstUsers.TabIndex = 2;

//

// MainForm

//

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(917, 709);

Controls.Add(lstUsers);

Controls.Add(statusStrip1);

Controls.Add(menuStrip1);

MainMenuStrip = menuStrip1;

Margin = new Padding(4, 5, 4, 5);

Name = "MainForm";

StartPosition = FormStartPosition.CenterScreen;

Text = "Система управления учетными записями";

FormClosing += MainForm\_FormClosing;

menuStrip1.ResumeLayout(false);

menuStrip1.PerformLayout();

statusStrip1.ResumeLayout(false);

statusStrip1.PerformLayout();

((System.ComponentModel.ISupportInitialize)lstUsers).EndInit();

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.MenuStrip menuStrip1;

private System.Windows.Forms.ToolStripMenuItem файлToolStripMenuItem;

private System.Windows.Forms.ToolStripMenuItem menuChangePassword;

private System.Windows.Forms.ToolStripMenuItem menuExit;

private System.Windows.Forms.ToolStripMenuItem пользователиToolStripMenuItem;

private System.Windows.Forms.ToolStripMenuItem menuAddUser;

private System.Windows.Forms.ToolStripMenuItem menuViewUsers;

private System.Windows.Forms.ToolStripMenuItem настройкиToolStripMenuItem;

private System.Windows.Forms.ToolStripMenuItem menuChangeMinLength;

private System.Windows.Forms.ToolStripMenuItem menuChangeExpiry;

private System.Windows.Forms.ToolStripMenuItem справкаToolStripMenuItem;

private System.Windows.Forms.ToolStripMenuItem menuAbout;

private System.Windows.Forms.StatusStrip statusStrip1;

private System.Windows.Forms.ToolStripStatusLabel toolStripStatusLabel1;

private DataGridView lstUsers;

private ToolStripMenuItem menuBlockUser;

private ToolStripMenuItem menuSetRestrictions;

}

}

// File: InfSec\PasswordDialog.cs

﻿using System;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.StartPanel;

namespace InfSec

{

public partial class PasswordDialog : Form

{

public string NewPassword { get; private set; }

public PasswordDialog()

{

InitializeComponent();

this.Text = "Установка пароля";

}

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

{

// Запись паролей

string password1 = txtPassword1.Text;

string password2 = txtPassword2.Text;

// Если поля заполнены не верно

if (string.IsNullOrEmpty(password1))

{

MessageBox.Show("Введите пароль", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

if (password1 != password2)

{

MessageBox.Show("Пароли не совпадают", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Проверяем, если минимальная длина больше нуля, то проверяем длину нового пароля

int minLength = DatabaseManager.GetUserMinLength("ADMIN");

if (minLength > 0 && password1.Length < minLength)

{

MessageBox.Show($"Пароль должен быть не короче {minLength} символов", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Если ограничения для пользователя включены

if (DatabaseManager.GetUserRestrictions("ADMIN"))

{

// Проверяем требования к паролю

if (!ValidatePasswordRequirements(password1))

{

MessageBox.Show("Пароль не соответствует требованиям:\n" +

"- Наличие цифр\n" +

"- Наличие знаков препинания\n" +

"- Наличие знаков арифметических операций",

"Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

}

//Закрытие формы

NewPassword = password1;

DialogResult = DialogResult.OK;

Close();

}

/// <summary>

/// Метод валидации пароля по индивидуальному заданию

/// </summary>

/// <param name="password">Пароль для валидации</param>

/// <returns>True, есди пароль прошел валидацию, иначе - False</returns>

private bool ValidatePasswordRequirements(string password)

{

// Проверяем наличие цифр

bool hasDigits = false;

foreach (char c in password)

{

if (char.IsDigit(c))

{

hasDigits = true;

break;

}

}

// Проверяем наличие знаков препинания

bool hasPunctuation = false;

foreach (char c in password)

{

if (char.IsPunctuation(c))

{

hasPunctuation = true;

break;

}

}

// Проверяем наличие знаков арифметических операций

bool hasArithmetic = false;

foreach (char c in password)

{

if (c == '+' || c == '-' || c == '\*' || c == '/')

{

hasArithmetic = true;

break;

}

}

return hasDigits && hasPunctuation && hasArithmetic;

}

// Закрытие формы по клику

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

{

DialogResult = DialogResult.Cancel;

Close();

}

}

}

// File: InfSec\PasswordDialog.Designer.cs

﻿namespace InfSec

{

partial class PasswordDialog

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label1 = new Label();

label2 = new Label();

txtPassword1 = new TextBox();

txtPassword2 = new TextBox();

btnOK = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(16, 21);

label1.Margin = new Padding(4, 0, 4, 0);

label1.Name = "label1";

label1.Size = new Size(115, 20);

label1.TabIndex = 0;

label1.Text = "Новый пароль:";

//

// label2

//

label2.AutoSize = true;

label2.Location = new Point(16, 56);

label2.Margin = new Padding(4, 0, 4, 0);

label2.Name = "label2";

label2.Size = new Size(124, 20);

label2.TabIndex = 1;

label2.Text = "Подтверждение:";

//

// txtPassword1

//

txtPassword1.Location = new Point(148, 14);

txtPassword1.Margin = new Padding(4, 5, 4, 5);

txtPassword1.Name = "txtPassword1";

txtPassword1.PasswordChar = '\*';

txtPassword1.Size = new Size(290, 27);

txtPassword1.TabIndex = 2;

//

// txtPassword2

//

txtPassword2.Location = new Point(148, 51);

txtPassword2.Margin = new Padding(4, 5, 4, 5);

txtPassword2.Name = "txtPassword2";

txtPassword2.PasswordChar = '\*';

txtPassword2.Size = new Size(290, 27);

txtPassword2.TabIndex = 3;

//

// btnOK

//

btnOK.Location = new Point(230, 88);

btnOK.Margin = new Padding(4, 5, 4, 5);

btnOK.Name = "btnOK";

btnOK.Size = new Size(100, 35);

btnOK.TabIndex = 4;

btnOK.Text = "ОК";

btnOK.UseVisualStyleBackColor = true;

btnOK.Click += btnOK\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(338, 88);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 5;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// PasswordDialog

//

AcceptButton = btnOK;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(456, 136);

Controls.Add(btnCancel);

Controls.Add(btnOK);

Controls.Add(txtPassword2);

Controls.Add(txtPassword1);

Controls.Add(label2);

Controls.Add(label1);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "PasswordDialog";

StartPosition = FormStartPosition.CenterScreen;

Text = "Установка пароля";

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox txtPassword1;

private System.Windows.Forms.TextBox txtPassword2;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\PasswordPhraseForm.cs

﻿using System;

using System.Windows.Forms;

namespace InfSec

{

public partial class PasswordPhraseForm : Form

{

public string PasswordPhrase { get; private set; }

public bool Cancelled { get; private set; }

public PasswordPhraseForm()

{

InitializeComponent();

this.Text = "Ввод парольной фразы";

}

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

{

// Считывание парольной фразы

string phrase = txtPasswordPhrase.Text.Trim();

// Проверка парольной фразы на пустоту

if (string.IsNullOrEmpty(phrase))

{

MessageBox.Show("Введите парольную фразу", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Warning);

return;

}

// Запись введенной парольной фразы

PasswordPhrase = phrase;

//Звкрытие формы

DialogResult = DialogResult.OK;

Close();

}

// Закрытик формы по клику

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

{

Cancelled = true;

DialogResult = DialogResult.Cancel;

Close();

}

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

{

txtPasswordPhrase.Focus();

}

}

}

// File: InfSec\PasswordPhraseForm.Designer.cs

﻿namespace InfSec

{

partial class PasswordPhraseForm

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

label1 = new Label();

txtPasswordPhrase = new TextBox();

btnOK = new Button();

btnCancel = new Button();

SuspendLayout();

//

// label1

//

label1.AutoSize = true;

label1.Location = new Point(13, 17);

label1.Margin = new Padding(4, 0, 4, 0);

label1.Name = "label1";

label1.Size = new Size(136, 20);

label1.TabIndex = 0;

label1.Text = "Парольная фраза:";

//

// txtPasswordPhrase

//

txtPasswordPhrase.Location = new Point(157, 14);

txtPasswordPhrase.Margin = new Padding(4, 5, 4, 5);

txtPasswordPhrase.Name = "txtPasswordPhrase";

txtPasswordPhrase.PasswordChar = '\*';

txtPasswordPhrase.Size = new Size(286, 27);

txtPasswordPhrase.TabIndex = 1;

//

// btnOK

//

btnOK.Location = new Point(235, 51);

btnOK.Margin = new Padding(4, 5, 4, 5);

btnOK.Name = "btnOK";

btnOK.Size = new Size(100, 35);

btnOK.TabIndex = 2;

btnOK.Text = "ОК";

btnOK.UseVisualStyleBackColor = true;

btnOK.Click += btnOK\_Click;

//

// btnCancel

//

btnCancel.Location = new Point(343, 51);

btnCancel.Margin = new Padding(4, 5, 4, 5);

btnCancel.Name = "btnCancel";

btnCancel.Size = new Size(100, 35);

btnCancel.TabIndex = 3;

btnCancel.Text = "Отмена";

btnCancel.UseVisualStyleBackColor = true;

btnCancel.Click += btnCancel\_Click;

//

// PasswordPhraseForm

//

AcceptButton = btnOK;

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(456, 103);

Controls.Add(btnCancel);

Controls.Add(btnOK);

Controls.Add(txtPasswordPhrase);

Controls.Add(label1);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "PasswordPhraseForm";

StartPosition = FormStartPosition.CenterScreen;

Text = "Ввод парольной фразы";

Load += PasswordPhraseForm\_Load;

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox txtPasswordPhrase;

private System.Windows.Forms.Button btnOK;

private System.Windows.Forms.Button btnCancel;

}

}

// File: InfSec\ViewUsersDialog.cs

﻿using System;

using System.Data;

using System.Windows.Forms;

namespace InfSec

{

public partial class ViewUsersDialog : Form

{

private DataTable usersTable;

private int currentIndex = 0;

public ViewUsersDialog()

{

InitializeComponent();

this.Text = "Список пользователей";

LoadUsers();

}

/// <summary>

/// Метод загрузки всех пользователей

/// </summary>

private void LoadUsers()

{

// Получение всех пользователей

usersTable = DatabaseManager.GetAllUsers();

// Проверка на кол-во пользователей

if (usersTable.Rows.Count > 0)

{

DisplayUser(currentIndex);

}

else

{

lblUsername.Text = "Нет пользователей";

lblBlocked.Text = "";

lblRestrictions.Text = "";

lblMinLength.Text = "";

lblExpiry.Text = "";

}

}

/// <summary>

/// Метод отображения пользователя

/// </summary>

/// <param name="index">Номер пользователя</param>

private void DisplayUser(int index)

{

if (usersTable.Rows.Count <= 0) return;

DataRow row = usersTable.Rows[index];

lblUsername.Text = row["username"].ToString();

lblBlocked.Text = Convert.ToBoolean(row["blocked"]) ? "Заблокирован" : "Активен";

lblRestrictions.Text = Convert.ToBoolean(row["restrictions\_enabled"]) ? "Ограничения включены" : "Ограничения отключены";

lblMinLength.Text = $"Минимальная длина: {row["min\_length"]}";

lblExpiry.Text = $"Срок действия: {row["expiry\_months"]} месяцев";

}

/// <summary>

/// Метод для отображения следующего пользователя

/// </summary>

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

{

if (usersTable.Rows.Count > 0)

{

currentIndex++;

if (currentIndex >= usersTable.Rows.Count)

{

currentIndex = 0;

}

DisplayUser(currentIndex);

}

}

/// <summary>

/// Метод для отображения предыдущего пользователя

/// </summary>

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

{

if (usersTable.Rows.Count > 0)

{

currentIndex--;

if (currentIndex < 0)

{

currentIndex = usersTable.Rows.Count - 1;

}

DisplayUser(currentIndex);

}

}

// Закрытие формы по клику

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

{

Close();

}

}

}

// File: InfSec\ViewUsersDialog.Designer.cs

﻿namespace InfSec

{

partial class ViewUsersDialog

{

/// <summary>

/// Required designer variable.

/// </summary>

private System.ComponentModel.IContainer components = null;

/// <summary>

/// Clean up any resources being used.

/// </summary>

/// <param name="disposing">true if managed resources should be disposed; otherwise, false.</param>

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

#region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

private void InitializeComponent()

{

lblUsername = new Label();

lblBlocked = new Label();

lblRestrictions = new Label();

lblMinLength = new Label();

lblExpiry = new Label();

btnPrev = new Button();

btnNext = new Button();

SuspendLayout();

//

// lblUsername

//

lblUsername.AutoSize = true;

lblUsername.Font = new Font("Microsoft Sans Serif", 12F, FontStyle.Bold, GraphicsUnit.Point, 204);

lblUsername.Location = new Point(16, 38);

lblUsername.Margin = new Padding(4, 0, 4, 0);

lblUsername.Name = "lblUsername";

lblUsername.Size = new Size(207, 25);

lblUsername.TabIndex = 0;

lblUsername.Text = "Имя пользователя";

//

// lblBlocked

//

lblBlocked.AutoSize = true;

lblBlocked.Location = new Point(16, 85);

lblBlocked.Margin = new Padding(4, 0, 4, 0);

lblBlocked.Name = "lblBlocked";

lblBlocked.Size = new Size(140, 20);

lblBlocked.TabIndex = 1;

lblBlocked.Text = "Статус блокировки";

//

// lblRestrictions

//

lblRestrictions.AutoSize = true;

lblRestrictions.Location = new Point(16, 120);

lblRestrictions.Margin = new Padding(4, 0, 4, 0);

lblRestrictions.Name = "lblRestrictions";

lblRestrictions.Size = new Size(179, 20);

lblRestrictions.TabIndex = 2;

lblRestrictions.Text = "Ограничения на пароль";

//

// lblMinLength

//

lblMinLength.AutoSize = true;

lblMinLength.Location = new Point(16, 155);

lblMinLength.Margin = new Padding(4, 0, 4, 0);

lblMinLength.Name = "lblMinLength";

lblMinLength.Size = new Size(210, 20);

lblMinLength.TabIndex = 3;

lblMinLength.Text = "Минимальная длина пароля";

//

// lblExpiry

//

lblExpiry.AutoSize = true;

lblExpiry.Location = new Point(16, 191);

lblExpiry.Margin = new Padding(4, 0, 4, 0);

lblExpiry.Name = "lblExpiry";

lblExpiry.Size = new Size(165, 20);

lblExpiry.TabIndex = 4;

lblExpiry.Text = "Срок действия пароля";

//

// btnPrev

//

btnPrev.Location = new Point(16, 231);

btnPrev.Margin = new Padding(4, 5, 4, 5);

btnPrev.Name = "btnPrev";

btnPrev.Size = new Size(153, 35);

btnPrev.TabIndex = 5;

btnPrev.Text = "Предыдущий";

btnPrev.UseVisualStyleBackColor = true;

btnPrev.Click += btnPrev\_Click;

//

// btnNext

//

btnNext.Location = new Point(177, 231);

btnNext.Margin = new Padding(4, 5, 4, 5);

btnNext.Name = "btnNext";

btnNext.Size = new Size(150, 35);

btnNext.TabIndex = 6;

btnNext.Text = "Следующий";

btnNext.UseVisualStyleBackColor = true;

btnNext.Click += btnNext\_Click;

//

// ViewUsersDialog

//

AutoScaleDimensions = new SizeF(8F, 20F);

AutoScaleMode = AutoScaleMode.Font;

ClientSize = new Size(340, 285);

Controls.Add(btnNext);

Controls.Add(btnPrev);

Controls.Add(lblExpiry);

Controls.Add(lblMinLength);

Controls.Add(lblRestrictions);

Controls.Add(lblBlocked);

Controls.Add(lblUsername);

FormBorderStyle = FormBorderStyle.FixedSingle;

Margin = new Padding(4, 5, 4, 5);

MaximizeBox = false;

MinimizeBox = false;

Name = "ViewUsersDialog";

StartPosition = FormStartPosition.CenterScreen;

Text = "Список пользователей";

ResumeLayout(false);

PerformLayout();

}

#endregion

private System.Windows.Forms.Label lblUsername;

private System.Windows.Forms.Label lblBlocked;

private System.Windows.Forms.Label lblRestrictions;

private System.Windows.Forms.Label lblMinLength;

private System.Windows.Forms.Label lblExpiry;

private System.Windows.Forms.Button btnPrev;

private System.Windows.Forms.Button btnNext;

}

}