МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет автоматики и вычислительной техники

Кафедра АСУ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_



ОТЧЁТ

по практической работе № 7

Тема: Реализация проекта аутентификации и авторизации пользователей с использованием заданной модели безопасности

Дисциплина: Основы информационной безопасности

Направление подготовки 09.03.01 и 09.03.03

Информатика и вычислительная техника

Выполнили студенты группы АВТ-314: Проверил:

Гусейнов Р.А. ст. преподаватель

Карапетян А.А. Плеханов В.А.

Попов В.А.

2024

Оглавление

[Введение 3](#_Toc183399539)

[Принцип работы программы 5](#_Toc183399540)

[Демонстрация работы программы 8](#_Toc183399541)

[Заключение 11](#_Toc183399542)

[Приложение 12](#_Toc183399543)

# Введение

Данная работа посвящена разработке и реализации приложения, обеспечивающего безопасное хранение и обработку конфиденциальной информации. В условиях возрастающей значимости защиты данных, особенно в цифровом мире, актуальность подобных решений неоспорима. В рамках проекта мы рассматриваем задачу создания приложения, работающего с плоскими файлами, хранящимися на локальном диске операционной системы. Для обеспечения безопасности, будет реализован механизм защиты доступа к данным, включающий в себя надежное хранение паролей пользователей с использованием криптографически стойких хэш-функций.

**Основные цели:**

1. Исследование проблем безопасности информации.
2. Активное использование знаний предметной области.

3. Отработка приемов программирования на конкретных практических примерах.

Для успешного выполнения поставленных задач по обеспечению информационной безопасности, наша команда определит конкретные шаги, которые включают в себя:

1. **Определение Защищаемой Информации:**

* Идентификация текстового файла с именами пользователей как критической информации, требующей защиты.
* Создание перечня сведений конфиденциального характера, соответствующего защищаемой информации.

1. **Хранение Паролей с Использованием Хэш-Функции:**

* Разработка механизма безопасного хранения паролей пользователей с применением хэш-функции.
* Интеграция безопасного хранения паролей в плоские файлы на диске операционной системы.

1. **Логическая Защита Объектов:**

* Определение объектов, связанных с защищаемой информацией, подлежащих логической защите.
* Разработка стратегии логической защиты для эффективного контроля доступа к защищенным объектам.

1. **Инструменты Разработки:**

* Использование языка программирования C# для создания надежных механизмов безопасности.
* Разработка приложения с использованием Windows Forms для удобного управления доступом и защиты информации.

1. **Роли в Команде:**

* Попов Виктор – руководитель, бэк-энд разработчик
* Карапетян Ани – дизайнер, фронт-энд разработчик
* Гусейнов Рза - тестировщик

**Технология Регистрации Пользователей:**

Ввод Данных:

* Способ: Пользователь вводит свои данные (логин, пароль) через пользовательский интерфейс приложения.

Аутентификация:

* Способ: Система проверяет введенные данные на уникальность и корректность. Применяется хэширование пароля для безопасного хранения.

Хранение данных для каждого пользователя реализовано соответствующей директорией, хранящей в себе файлы, доступные только определенному пользователю для просмотра.

# Принцип работы программы

Программа состоит из двух форм (Form1 — registration\_window и Form2) и использует файл test.txt для хранения учетных данных пользователей. Работает она следующим образом:

**1. Форма registration\_window (Form1):**

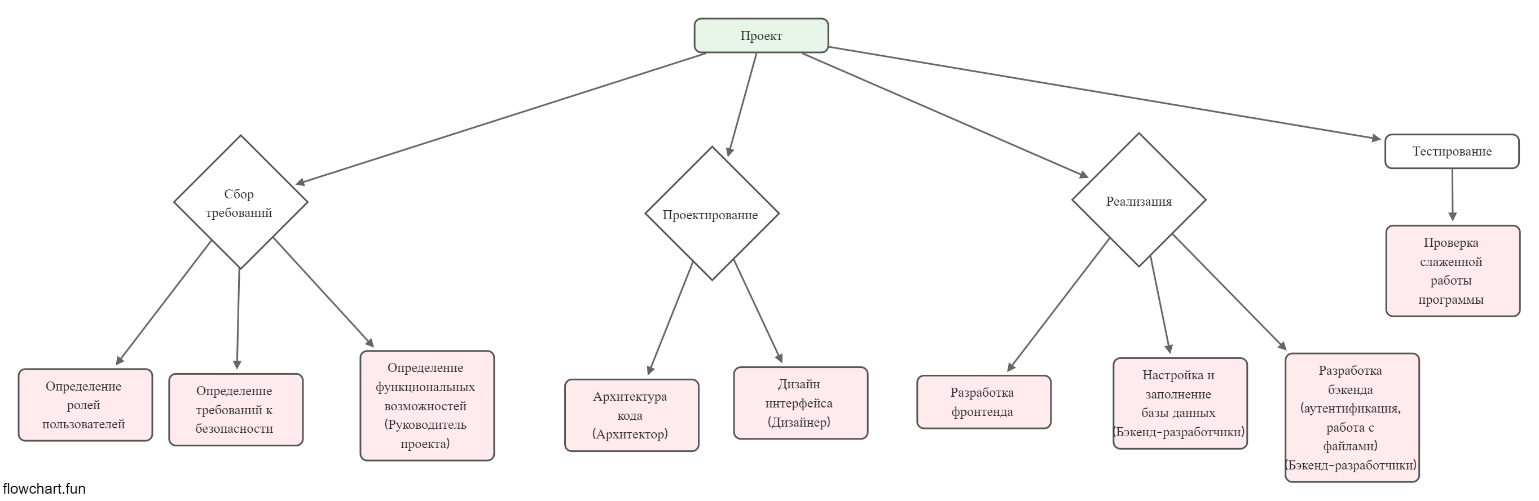
* Эта форма предназначена для входа пользователя в систему. Она содержит поля для ввода имени пользователя (user\_name) и пароля (user\_password).
* Кнопка “Войти” запускает обработчик события button1\_Click.
* В обработчике события button1\_Click:
  + Считываются введенные пользователем имя и пароль.
  + Вызывается метод isValidPassword для проверки введенных данных.
  + Если метод isValidPassword возвращает **true** (введенные данные верны):
  + Создается экземпляр формы Form2 (newF).
  + Form2 отображается, а текущая форма (Form1) скрывается.
  + Имя пользователя отображается в метке label2 на форме Form2.
  + Программа пытается получить список файлов из директории, имя которой совпадает с именем пользователя (D:\pr7\{0}\, где {0} – имя пользователя).
  + Список файлов отображается в элементе listView1 на форме Form2.
  + Если метод isValidPassword возвращает **false** (учетные данные неверны): ничего не происходит (нужно добавить обработку ошибки).
* Метод isValidPassword считывает файл test.txt, содержащий пары “имя пользователя пароль”, разделенные пробелами. Он последовательно проверяет, совпадает ли введенное пользователем имя с именем в файле, и использует класс PasswordHasher для проверки пароля.

**2. Форма Form2:**

* Эта форма отображает имя вошедшего пользователя и список файлов из его директории.
* Кнопка на Form2 позволяет вернуться к форме Form1 (registration\_window). Она очищает поля ввода и список файлов перед переходом.

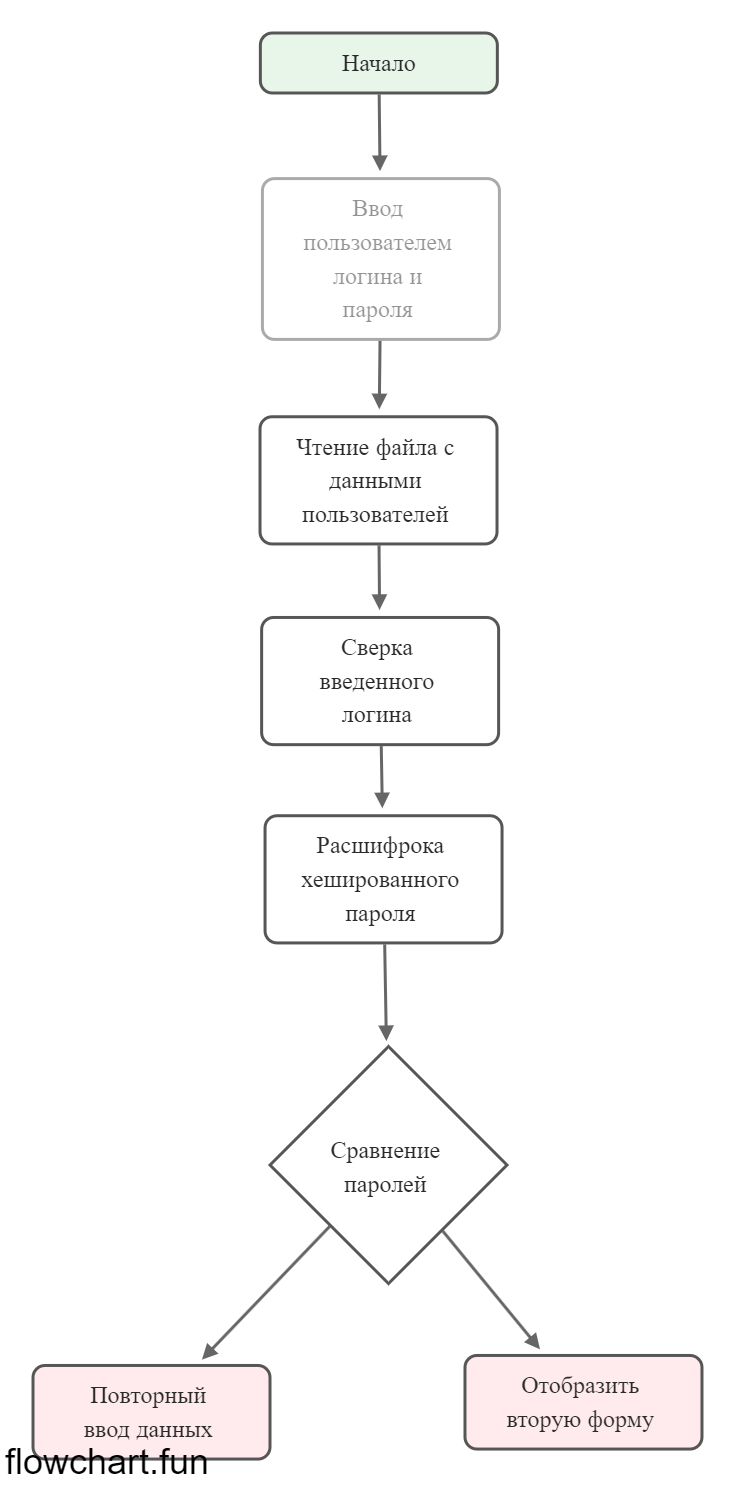
**3. Класс PasswordHasher:**

* Этот класс (рассмотрен ранее) отвечает за хеширование и проверку паролей с использованием библиотеки BCrypt.Net. Он обеспечивает (теоретически) криптографически безопасный способ хранения паролей.

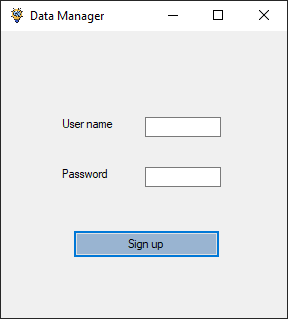


# 

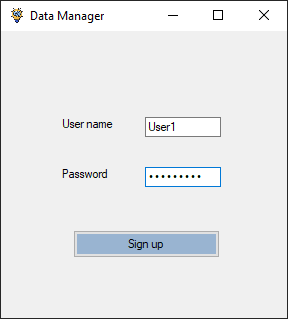
**Проверка пользователя**



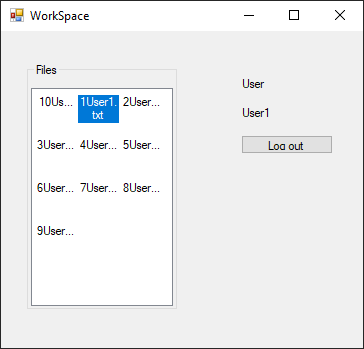
# Демонстрация работы программы

**Начальное окно**

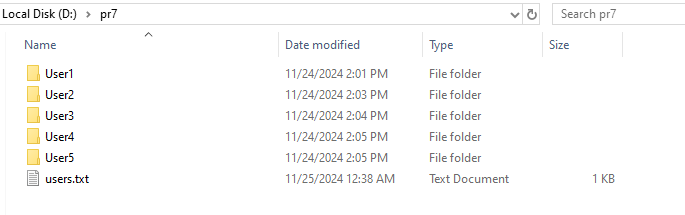
**Ввод соответствующих данных**



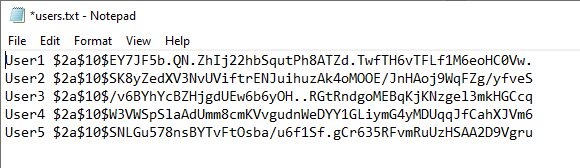
**Рабочее пространство авторизованного пользователя**



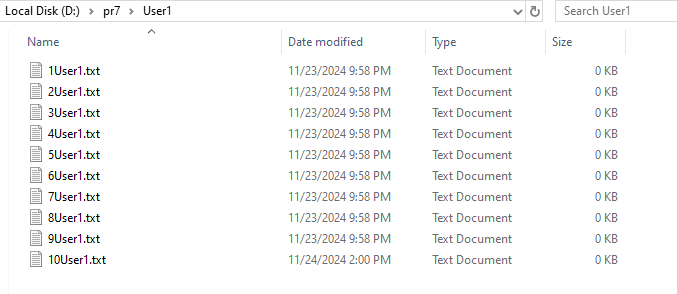
**Файловая система**



**Хранение данных о пользователях (логин пароль)**



**Пользовательские файлы**



# Заключение

В ходе данной работы мы подробно рассмотрели и разработали систему обеспечения информационной безопасности, фокусируясь на технологии регистрации пользователей и методах хранения необходимых данных. Также были выделены субъекты и объекты доступа, определены роли пользователей и системы в этом контексте.

Технология регистрации, основанная на аутентификации, обеспечивает надежность и безопасность процесса. Введение хэширования паролей и логической защиты объектов дополнительно повышает уровень безопасности.

# Приложение

**Исходный код программы**

**Class Form1**

|  |
| --- |
| using System;  using System.IO;  using System.Windows.Forms;  using System.Xml.Linq;  using static System.Windows.Forms.VisualStyles.VisualStyleElement;  namespace PRACTICA\_OIB  {  public partial class registration\_window : Form  {  public registration\_window()  {  InitializeComponent();  }  private void button1\_Click(object sender, EventArgs e)  {  string name = user\_name.Text;  string password = user\_password.Text;  bool isValid = isValidPassword(name, password);    if (isValid)  {  Form2 newF = new Form2();  newF.Show();  this.Hide();  newF.label2.Text = name;  string directoryPath = String.Format(@"D:\pr7\{0}\", name);  try  {  string[] files = Directory.GetFiles(directoryPath);  foreach (string file in files)  {  ListViewItem item = new ListViewItem(Path.GetFileName(file));  newF.listView1.Items.Add(item);  }  }  catch (DirectoryNotFoundException)  {  Console.WriteLine($"Directory not found: {directoryPath}");  }  }  }  public bool isValidPassword(string name, string password)  {  try  {  using (StreamReader sr = new StreamReader(@"D:\pr7\users.txt", System.Text.Encoding.Default))  {  string line;  while ((line = sr.ReadLine()) != null)  {  string[] user = line.Split(' ');  if (name.Equals(user[0]) && PasswordHasher.VerifyPassword(password, user[1]))  {  return true;  }  }  }  }  catch (FileNotFoundException ex)  {  Console.WriteLine(ex.ToString());  }  return false;  }  private void Form1\_Load(object sender, EventArgs e){}  private void textBox2\_TextChanged(object sender, EventArgs e){}  }  } |

**Class Form2**

|  |
| --- |
| using System;  using System.Linq;  using System.Windows.Forms;  namespace PRACTICA\_OIB  {  public partial class Form2 : Form  {  public Form2()  {  InitializeComponent();  }  private void groupBox1\_Enter(object sender, EventArgs e){}  private void button1\_Click\_1(object sender, EventArgs e)  {  registration\_window form1 = Application.OpenForms.OfType<registration\_window>().FirstOrDefault();  if (form1 != null)  {  form1.Show();  this.Hide();  form1.user\_name.Clear();  form1.user\_password.Clear();  listView1.Items.Clear();  }  else  {  MessageBox.Show("Ошибка: первая форма не найдена.");  }  }  }  } |

**Class PasswordHasher**

|  |
| --- |
| using System;  namespace PRACTICA\_OIB  {  public class PasswordHasher  {  public static string HashPassword(string password)  {  string salt = BCrypt.Net.BCrypt.GenerateSalt(); // Генерируем соль с помощью BCrypt  return BCrypt.Net.BCrypt.HashPassword(password, salt); // Хешируем с использованием соли  }  public static bool VerifyPassword(string password, string hashedPassword)  {  return BCrypt.Net.BCrypt.Verify(password, hashedPassword);  }  }  } |

**Class Program**

|  |
| --- |
| using System;  using System.IO;  using System.Text;  using System.Windows.Forms;  namespace PRACTICA\_OIB  {  public static class Program  {  [STAThread]  public static void Main()  {  Application.EnableVisualStyles();  Application.SetCompatibleTextRenderingDefault(false);  Application.Run(new registration\_window());  }  }  } |