МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО «МГТУ «СТАНКИН»)

Институт информационных систем и технологий

Кафедра информационных технологий и вычислительных систем

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №3

ПО ДИСЦИПЛИНЕ

«Защита информации»

СТУДЕНТА 4 КУРСА Бакалавриата ГРУППЫ ИДБ-20-02

Ердогана Дениза Ердаловича

НА ТЕМУ

Использование функций криптографического интерфейса Windows для защиты информации

Вариант № 8

Направление:	09.03.01 Информатика и вычислительная техника
Профиль подготовки:	«Программное обеспечение средств вычислительной техники и автоматизированных систем»

Отчет сдан «_____» _____ 2023 г.

Оценка _____

Преподаватель_____Шевляков К.А_____

(подпись)

МОСКВА 2023

Задание

1. В программу, разработанную при выполнении лабораторной работы № 1, добавить средства защиты от несанкционированного доступа к файлу с учетными данными зарегистрированных пользователей.
2. Файл с учетными данными должен быть зашифрован при помощи функций криптографического интерфейса операционной системы Windows (CryptoAPI) с использованием сеансового ключа, генерируемого на основе вводимой администратором (пользователем) парольной фразы.
3. При запуске программы файл с учетными данными должен расшифровываться во временный файл (или в файл в оперативной памяти), который после завершения работы программы должен быть снова зашифрован для отражения возможных изменений в учетных записях пользователей. «Старое» содержимое файла учетных записей при этом стирается.
4. После ввода парольной фразы при запуске программы, генерации ключа расшифрования и расшифрования файла с учетными данными зарегистрированных пользователей правильность введенной парольной фразы определяется по наличию в расшифрованном файле учетной записи администратора программы.
5. При вводе неправильной парольной фразы или отказе от ее ввода работа программы должна завершаться с выдачей соответствующего сообщения.
6. Временный файл на диске с расшифрованными учетными данными после завершения работы программы удаляется.
7. Варианты использования алгоритмов шифрования и хеширования выбираются в соответствии с выданным преподавателем заданием.

Индивидуальное задание

Тип симметричного шифрования – потоковый;

Используемый режим шифрования - -;

Добавление к ключу случайного значения – нет;

Используемый алгоритм хеширования – MD2.

Работа выполнена на языке C# с использованием Windows Forms, SQL, MAMP, Wincrypt.

Запросы на работу с текстовым файлом

1) Класс SECURITY KEY.cs – ввод ключа шифрования:

Необходимые библиотеки:

```
using MySql.Data.MySqlClient;
using System;
using System.Collections.Generic;
using System.Data;
using System.Text;
using System.Security.Cryptography;
```

Структура для шифрованного пользователя:

```
public struct SecureNode  // NODE FOR ENCRYPTION
{
    public string userLogin;
    public string userPassword;
    public string userLimits;
    public string userBlocked;
    public string userRules;
};
```

Необходимые переменные и ключ шифрования:

```
private static readonly string passwordKey = "123";  // ENCRYPTION KEY
readonly List<SecureNode> usersList = new List<SecureNode>();
```

Метод преобразования типов:

```
public int FromStringToInt(string value)  // METHOD FOR BOOL CELLS OF SQL
{
    if (String.Equals("True", value))
        return 1;
    else
        return 0;
}
```

Шифрование:

```
public string Decryption(string argument)  // DECRYPTION USER
{
    string answer = "";  // FINAL RESULT

    if (!string.IsNullOrEmpty(argument))
    {
        byte[] data = Convert.FromBase64String(argument);

        using (MD5CryptoServiceProvider md5 = new MD5CryptoServiceProvider())  // CREATE ALGORITHM FUNCTION
        {
            byte[] keys = md5.ComputeHash(UTF8Encoding.UTF8.GetBytes(passwordKey));

            using (TripleDESCryptoServiceProvider tripDes = new TripleDESCryptoServiceProvider()  // CLASS OF CIPHER TOOLS
            {
                // ПОТОКОВОСТЬ ЗАДАЁТСЯ РАЗМЕРОМ ПАКЕТОВ ДАННЫХ И МОДОМ, КОТОРЫЙ ИСПОЛЬЗУЕТ XOR
                FeedbackSize = 8,  // SIZE OF INFORMATION BLOCK
                Key = keys,  // INPUT KEY
                Mode = CipherMode.ECB
                //Mode = CipherMode.CBC  // CIPHER MODE - https://learn.microsoft.com/ru-ru/dotnet/api/system.security.cryptography.ciphermode?view=net-7.0
            })
            {
                ICryptoTransform transform = tripDes.CreateDecryptor();
                byte[] results = transform.TransformFinalBlock(data, 0, data.Length);
                answer = UTF8Encoding.UTF8.GetString(results);
            }
        }
    }

    return answer;
}
```

Дешифрование:

```
public void AddDecryption()  // DECRYPTION DATABASE
{
    SecureNode encryptionUser = new SecureNode();
    Database database = new Database();
    DataTable table = new DataTable();
    MySqlDataAdapter adapter = new MySqlDataAdapter();
    MySqlCommand getTable = new MySqlCommand("SELECT * FROM `secureusers`", database.GetConnection());

    adapter.SelectCommand = getTable;
    adapter.Fill(table);

    foreach (DataRow row in table.Rows)  // DECRYPTION ALL CELLS
    {
        encryptionUser.userLogin = Decryption(row["login"] as string);
        encryptionUser.userPassword = Decryption(row["password"] as string);
        encryptionUser.userLimits = Decryption(row["limits"] as string);
        encryptionUser.userBlocked = Decryption(row["blocked"] as string);
        encryptionUser.userRules = Decryption(row["root"] as string);

        usersList.Add(encryptionUser);
    }

    string connectionString = "server=localhost;user=root;database=usersdatabase;password=root;";
    MySqlConnection connection = new MySqlConnection(connectionString);
    connection.Open();

    string createTable = $"CREATE TABLE `usersdatabase`.`users` ( `id` INT(10) UNSIGNED NOT NULL AUTO_INCREMENT , " +
                         $"`login` VARCHAR(64) CHARACTER SET utf8 COLLATE utf8_general_ci NOT NULL , `password` VARCHAR(32) CHARACTER " +
                         $"SET utf8 COLLATE utf8_general_ci NOT NULL , `limits` TINYINT(1) NOT NULL , `blocked` TINYINT(1) NOT NULL , " +
                         $"`root` TINYINT(1) NOT NULL DEFAULT '0' , PRIMARY KEY (`login`), INDEX (`id`)) ENGINE = InnoDB;";
    MySqlCommand commandOne = new MySqlCommand(createTable, connection);
    commandOne.ExecuteScalar();

    foreach (SecureNode user in usersList)  // ENCRYPTION USERS ADD
    {
        string createEncryptionUser = $"INSERT INTO `users` (`id`, `login`, `password`, `limits`, `blocked`, `root`) VALUES (NULL, " +
           $"'{user.userLogin}', '{user.userPassword}', '{FromStringToInt(user.userLimits).ToString()}', " +
           $"'{FromStringToInt(user.userBlocked).ToString()}', '{FromStringToInt(user.userRules).ToString()}');";
        MySqlCommand loginCommand = new MySqlCommand(createEncryptionUser, connection);
        loginCommand.ExecuteScalar();
    }

    connection.Close();
}
```

Нажатие кнопки “CLOSE” – закрытия окна:

```
private void CloseButton_Click(object sender, System.EventArgs e)  // CLOSE APPLICATION
{
    System.Windows.Forms.Application.Exit();
}
```

Нажатие кнопки “OK” – ввода ключа шифрования/дешифрования:

```
private void OkButton_Click(object sender, System.EventArgs e)  // TRY TO DECRYPT TABLES
{
    string inputPassword = PasswordTextBox.Text;

    if (string.IsNullOrEmpty(inputPassword))  // IF USER DON'T INPUT ANYTHING
    {
        ErrorPage emptyInputPassword = new ErrorPage();
        emptyInputPassword.GetData(this.Location);
        emptyInputPassword.ChangeLabelText("YOU DON'T INPUT PASSWORD!");
        emptyInputPassword.ShowDialog();
    }
    else if (!String.Equals(inputPassword, passwordKey))  // INCORRECT PASSWORD
    {
        ErrorPage incorrectPassword = new ErrorPage();
        incorrectPassword.GetData(this.Location);
        incorrectPassword.ChangeLabelText("YOU INPUT INCORRECT PASSWORD!");
        incorrectPassword.ShowDialog();
    }
    else
    {
        AddDecryption();

        this.Hide();

        HomePage homePage = new HomePage();
        homePage.GetPassword(passwordKey);
        homePage.Show();
    }
}
```

Внешний вид программы

Рис. № 1 пустая база данных до первого запуска программы.

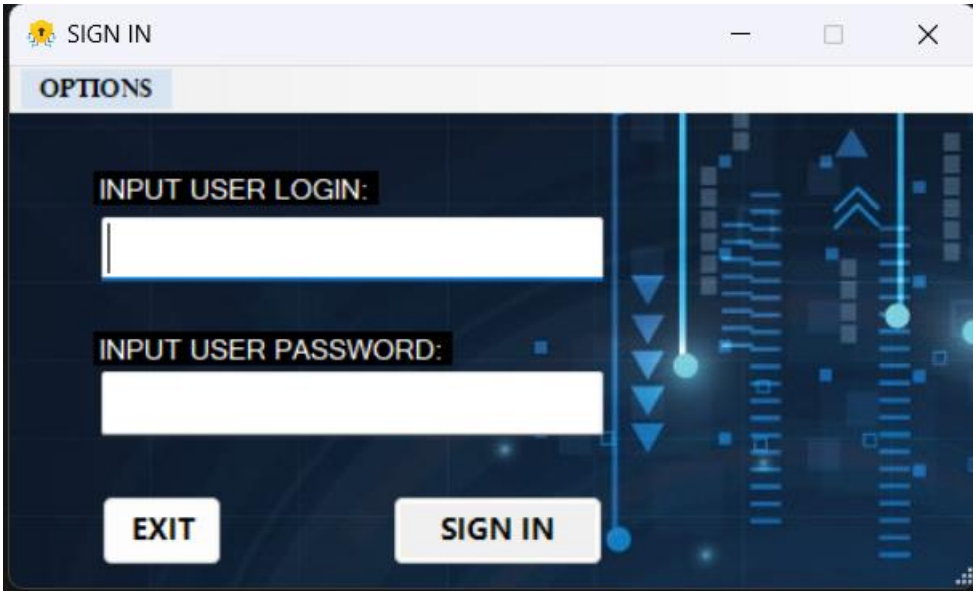

Рис. № 2 программа при первом запуске

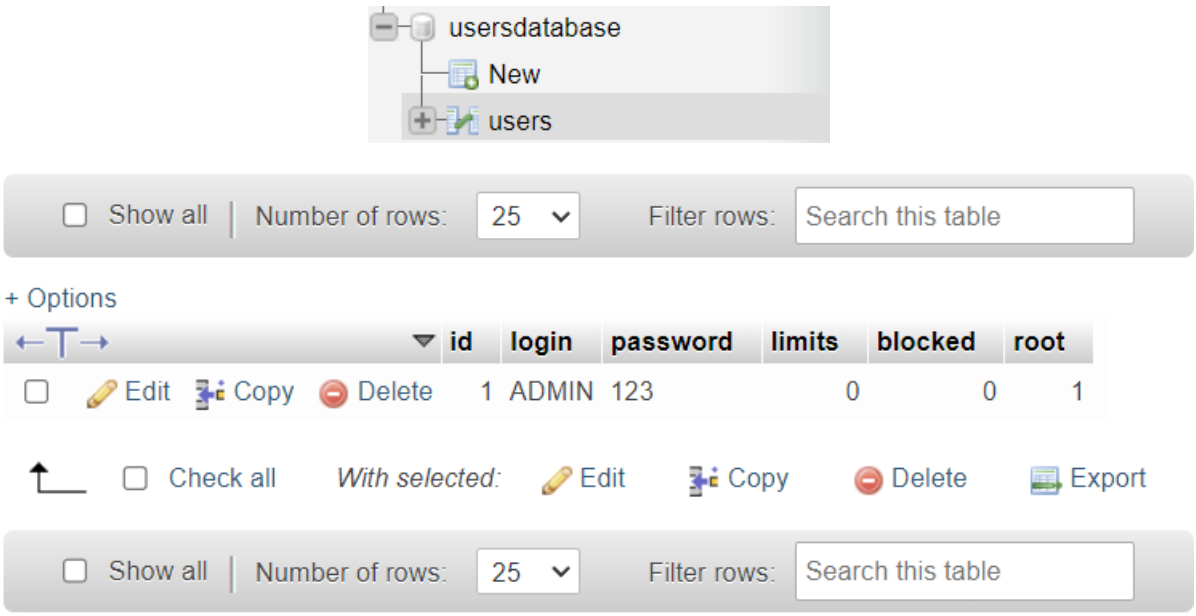

Рис. № 3, 4 база данных во время работы программы

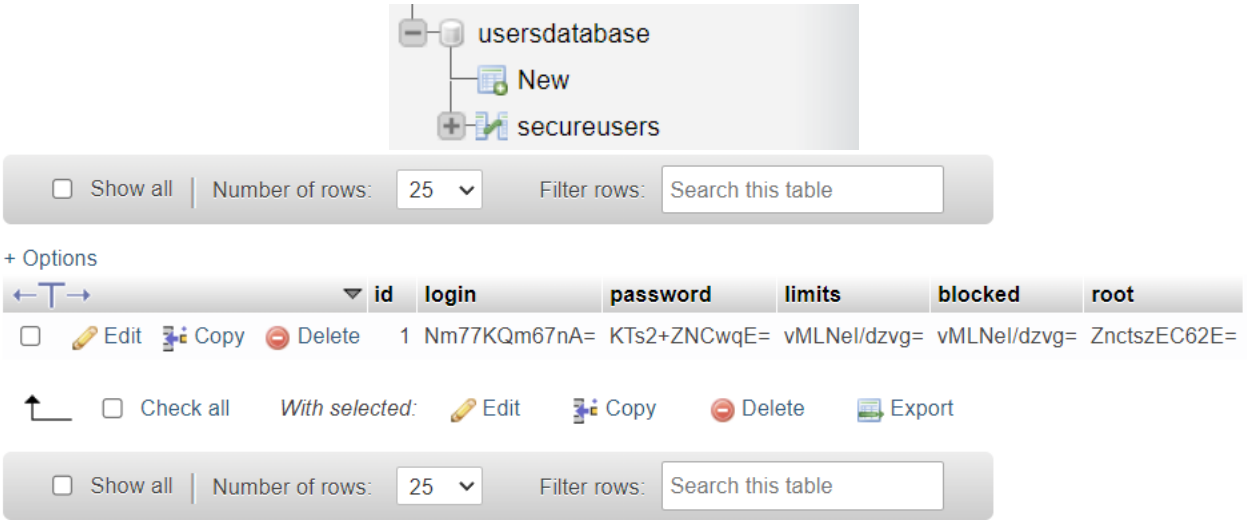

Рис. № 5, 6 база данных после завершения работы с программой

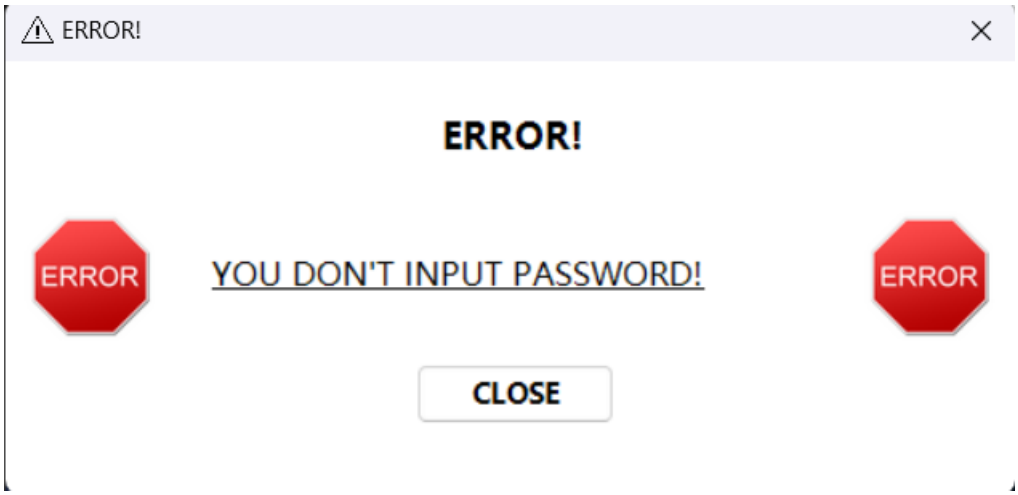

Рис. № 7 неправильный ввод ключа дешифрования

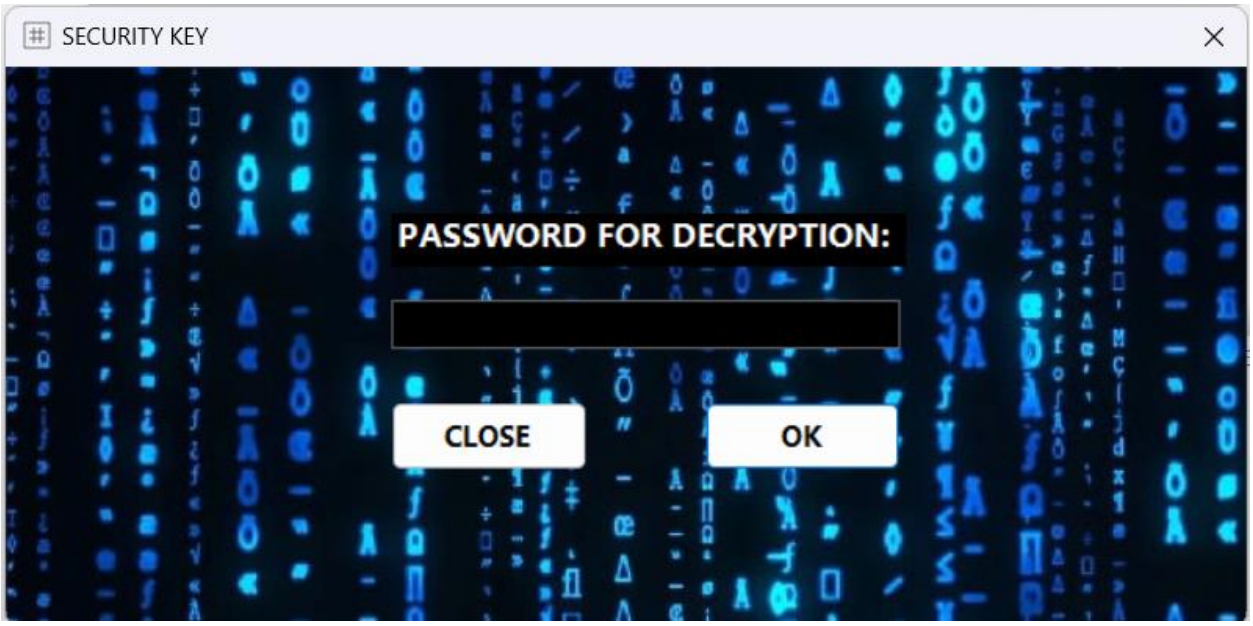

Рис. № 8 окно ввода ключа дешифрования

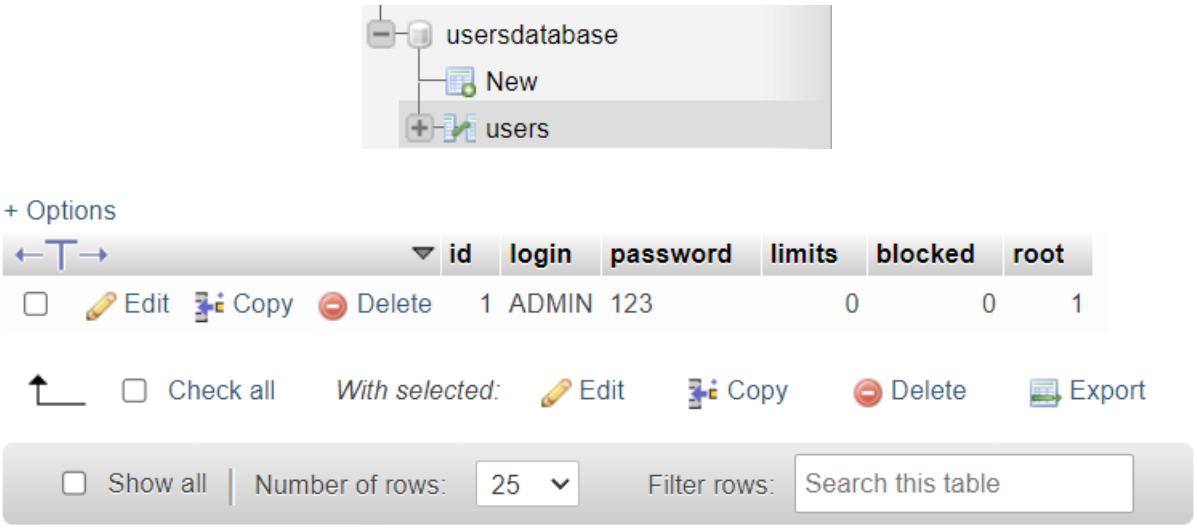

Рис. № 9, 10 база данных при правильном вводе ключа дешифрования

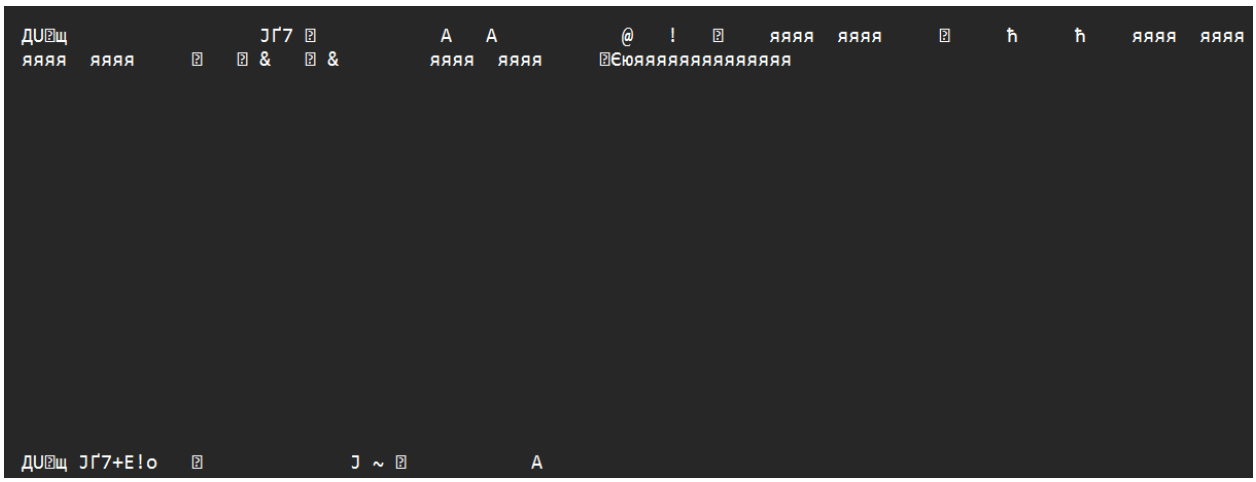

Рис. № 11 физический файл базы данных