# Министерство образования и науки Российской Федерации

## Федеральное государственное бюджетное образовательное учреждение высшего образования

## «Новосибирский государственный технический университет»

NSTU_Logo_blue

## Кафедра теоретической и прикладной информатики

### РГЗ по дисциплине «Информационная безопасность»

|  |  |  |
| --- | --- | --- |
| сигма градиент синий1 | Факультет: | ПМИ |
| Группа: | ПМИ-71 |
| Студент: | Арнольд Э. |
| Вариант: | 2 |
|  |  |
| Преподаватель: | Авдеенко Т.В. |

Новосибирск

2021

1. **Цель работы**

В рамках расчетно-графического задания (работы) по дисциплине студенты должны реализовать защищенное от нелегальной установки программное приложение.

1. **Задание**

При выполнении расчетно-графического задания (работы) студенты должны выбрать модель распространения программы (на выбор): Demoware, Trialware или Nagware. Для регистрации программы реализовать проверку введения регистрационного кода. Реализовать метод проверки регистрационного кода (на выбор): 1) алгоритмический, основанный на принципе "чёрного ящика"; 2) алгоритмический, основанный на математически сложной проблеме; 3) табличный. При необходимости в нужных местах следует применять шифрование. Алгоритм шифрования - на выбор в зависимости от шифруемых данных. Обязательные структурные части РГЗ. Обоснование выбора модель распространения программы Обоснование выбора метода проверки регистрационного кода Обоснование применения шифрование. Код программы Тестирование Отчет Демонстрация работы программы

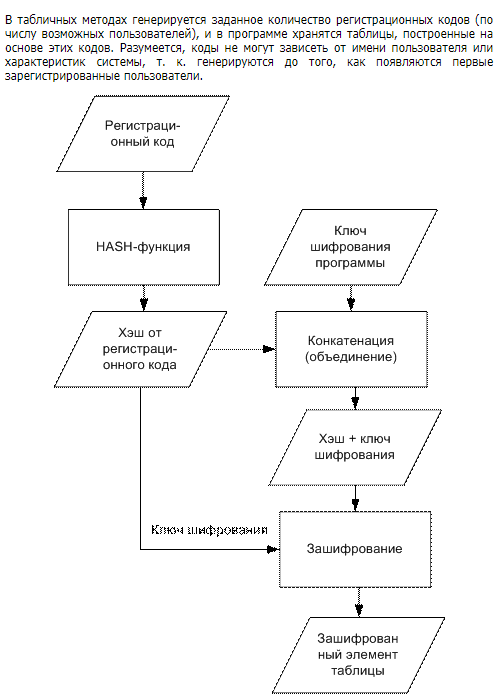
1. **Описание разработанного программного средства**

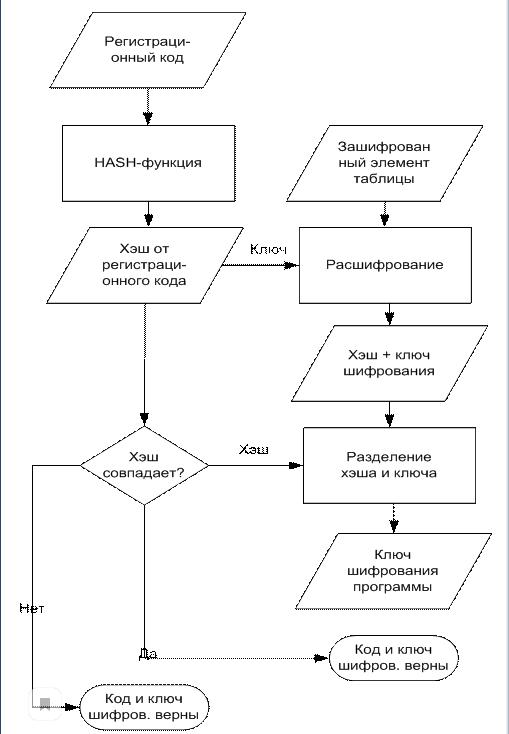
Обоснование выбора модель распространения программы :

**Nagware** - бесплатная программа, в которой во время использования появляется диалоговое окно с информацией о том, что мы пользуемся незарегистрированной версией программы. Такое окно можно убрать, оплатив программу.

Обоснование выбора метода проверки регистрационного кода:

**Табличный метод**Данныйметод был выбран потому, чтокоды не привязываются к конкретному пользователю и он относительно прост в реализации.





Обоснование применения шифрования :

SHA256 – хэш-функция

Используется для шифрования алгоритм Rijndael, т.к он является современным и гибким алгоритмом, соответствующий высоким стандартам криптостойкости.

Шифруются:

* Сообщения пользователя
* Регистрационный код + ключ

Алгоритм Rijndael(Стандарт AES) представляет собой итеративный блочный шифр, имеющий переменную длину блоков и различные длины ключей. Длина ключа и длина блока могут быть независимо друг от друга 128, 192 или 256 бит. В AES размер блока фиксирован и равен 128 бит, а в Rijndael поддерживаются различные длины ключей и блоков - от 128 до 256 бит, с шагом 32 бита. AES оперирует с блоком данных 16 байт, а Rijndael позволяет выбирать размер блока. Rijndael - быстрый и компактный алгоритм с простой математической структурой. Он продемонстрировал хорошую устойчивость к атакам на реализацию, при которых пытаются декодировать зашифрованное сообщение, анализируя внешние проявления алгоритма, в том числе уровень энергопотребления и время выполнения. Алгоритму присущ внутренний параллелизм, что позволяет без труда обеспечить эффективное использование процессорных ресурсов

Преимущества алгоритма Rijndael состоят в:

· высокая эффективность на любых платформах;

· высокий уровень защищенности;

· хорошо подходит для реализации в smart-картах из-за низких требований к памяти;

· быстрая процедура формирования ключа;

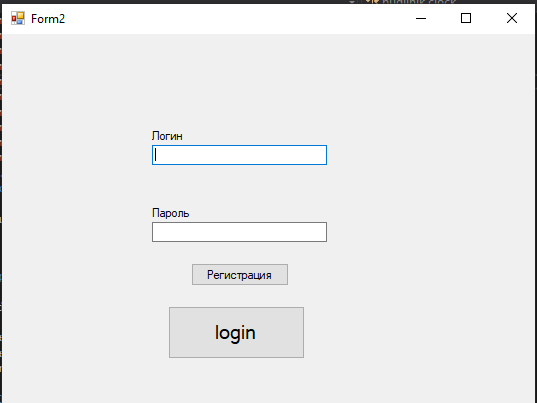
· хорошая поддержка параллелизма на уровне инструкций;

· поддержка различных длин ключа с шагом 32 бита.

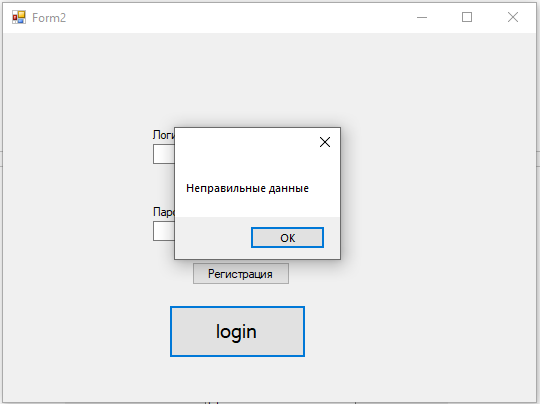
Недостатки:

· уязвим к анализу мощности.

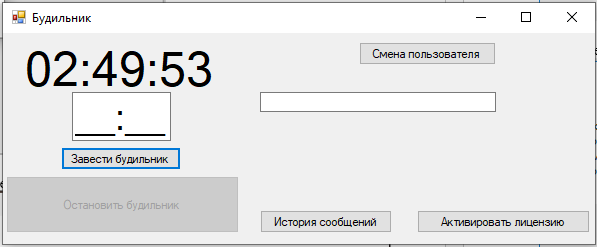
Окно аутентификации:

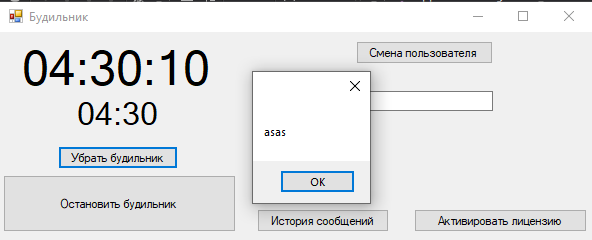


Результат не корректного ввода пароля:



Окно будильника:

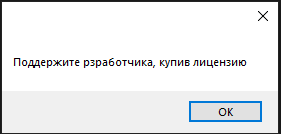




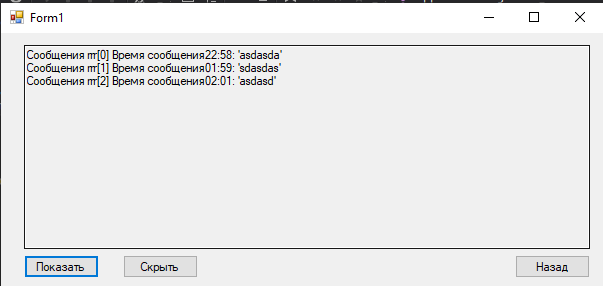
Окно будильника открывается после успешной аутентификации.

При этом, окно входа скрывается, чтобы избежать многократного входа.

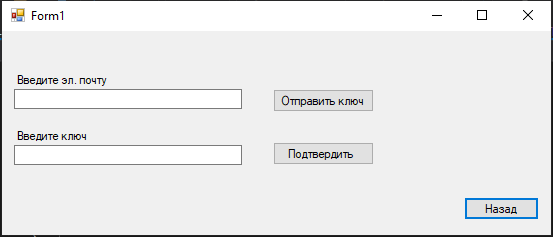
Окно с напоминанием, о том, чтобы активировать программу:



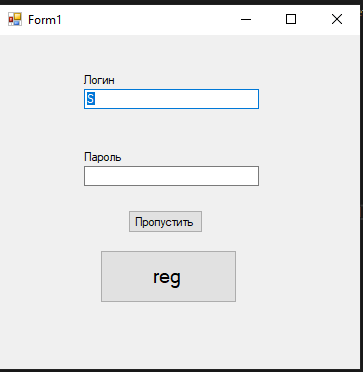
История сообщений пользователя:



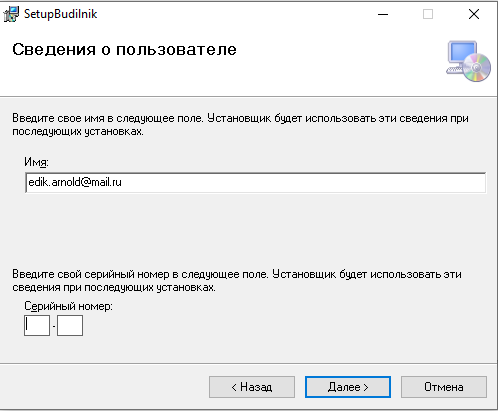
Окно активации платной версии:



Окно регистрации:



Окно установки:



1. **Код программы**

**CR.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Security.Cryptography;

namespace budilnik

{

class Cr

{

public byte[] SHA\_256(byte[] data) // генератор хэша на основе SHA1

{

byte[] res = new byte[1];

SHA256 SHA = new SHA256Managed();

res = SHA.ComputeHash(data);

return res;

}

}

}

**reg.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace budilnik

{

public partial class reg : Form

{

public reg()

{

InitializeComponent();

}

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

{

if (LoginFieldreg.Text == "")

{

MessageBox.Show("Введите логин");

return; }

if (PassFieldreg.Text == "")

{

MessageBox.Show("Введите пароль");

return;

}

if (user\_exist()) return;

String loguser = LoginFieldreg.Text;

String passuser = PassFieldreg.Text;

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("INSERT INTO `users` (`login`, `pass`, `demo`) VALUES(@loginreg, @passreg, `0`)", db.getconnection());

command.Parameters.Add("@loginreg", MySqlDbType.VarChar).Value = loguser;

command.Parameters.Add("@passreg", MySqlDbType.VarChar).Value = passuser;

db.openconnection();

if (command.ExecuteNonQuery() == 1)

MessageBox.Show("Succesfully");

else

MessageBox.Show("Fail");

db.closeconnection();

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

clock c2 = new clock();

c2.Show();

Hide();

}

}

public Boolean user\_exist()

{

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `users` WHERE `login` = @uL", db.getconnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = LoginFieldreg.Text;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Такой логин занят");

return true;

}

else return false;

}

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

{

author a2 = new author();

a2.Show();

Hide();

}

private void reg\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

**Sym\_alg.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

using System.Security.Cryptography;

using System.Collections;

using System.Threading;

namespace budilnik

{

class Sym\_alg

{

public byte[] key\_gen() // генератор 128-битного ключа

{

byte[] key = new byte[16]; // основной ключ

var rng = new RNGCryptoServiceProvider();

rng.GetBytes(key);

return key;

}

public class RijndaelAlgorithm

{

public static string Encrypt

(

string plainText,

string passPhrase,

string saltValue,

string hashAlgorithm,

int passwordIterations,

string initVector,

int keySize

)

{

//Преобразование строк в байтовые массивы.

//Предположим, что строки содержат только коды ASCII.

//Если строки содержат символы Юникода, используйте Юникод, UTF7 или UTF8

//Кодировки.

byte[] initVectorBytes = Encoding.Default.GetBytes(initVector);

byte[] saltValueBytes = Encoding.Default.GetBytes(saltValue);

//Преобразование открытого текста в массив байтов.

byte[] plainTextBytes = Encoding.Default.GetBytes(plainText);

// Во-первых, мы должны создать пароль, из которого будет получен ключ.

//Этот пароль будет создан из указанной парольной фразы и

//соли. Пароль будет создан с использованием указанного хэша

//алгоритма. Создание пароля может выполняться в нескольких итерациях.

PasswordDeriveBytes password = new PasswordDeriveBytes

(

passPhrase,

saltValueBytes,

hashAlgorithm,

passwordIterations

);

//Используйте пароль для создания псевдослучайных байтов для шифрования

//ключа. Укажите размер ключа в байтах (вместо битов).

byte[] keyBytes = password.GetBytes(keySize / 8);

//Создать неинициализированный объект шифрования Rijndael.

RijndaelManaged symmetricKey = new RijndaelManaged();

symmetricKey.Mode = CipherMode.CBC;

//Создать шифратор из существующих байтов ключа и инициализации

//вектор. Размер ключа определяется на основе количества байтов ключа.

ICryptoTransform encryptor = symmetricKey.CreateEncryptor

(

keyBytes,

initVectorBytes

);

//Определите поток памяти, который будет использоваться для хранения зашифрованных данных.

MemoryStream memoryStream = new MemoryStream();

//Определите криптографический поток (всегда используйте режим записи для шифрования).

CryptoStream cryptoStream = new CryptoStream

(

memoryStream,

encryptor,

CryptoStreamMode.Write

);

//Начните шифровать.

cryptoStream.Write(plainTextBytes, 0, plainTextBytes.Length);

//Шифровка конца.

cryptoStream.FlushFinalBlock();

//Преобразование зашифрованных данных из потока памяти в массив байтов.

byte[] cipherTextBytes = memoryStream.ToArray();

//Закройте оба потока.

memoryStream.Close();

cryptoStream.Close();

//Преобразование зашифрованных данных в строку в кодировке base64.

string cipherText = System.Text.Encoding.Default.GetString(cipherTextBytes);

//Возвратите зашифрованную последовательность.

return cipherText;

}

public static string Decrypt

(

string cipherText,

string passPhrase,

string saltValue,

string hashAlgorithm,

int passwordIterations,

string initVector,

int keySize

)

{

//Преобразование строк, определяющих характеристики ключа шифрования, в байтовые массивы.

byte[] initVectorBytes = Encoding.Default.GetBytes(initVector);

byte[] saltValueBytes = Encoding.Default.GetBytes(saltValue);

//Преобразование нашего зашифрованного текста в массив байтов.

byte[] cipherTextBytes = System.Text.Encoding.Default.GetBytes(cipherText);

//Во-первых, мы должны создать пароль, из которого будет получен ключ

//Этот пароль будет создан из указанной парольной фразы и значения соли.

//Пароль будет создан с использованием указанного алгоритма хэша. Создание пароля может выполняться в нескольких итерациях.

PasswordDeriveBytes password = new PasswordDeriveBytes

(

passPhrase,

saltValueBytes,

hashAlgorithm,

passwordIterations

);

//Используйте пароль для создания псевдослучайных байтов для шифрования

//ключа. Укажите размер ключа в байтах (вместо битов).

byte[] keyBytes = password.GetBytes(keySize / 8);

//Создать неинициализированный объект шифрования Rijndael.

RijndaelManaged symmetricKey = new RijndaelManaged();

//Целесообразно установить режим шифрования "Цепочка блоков шифрования"

//(CBC). Используйте параметры по умолчанию для других симметричных ключевых параметров.

symmetricKey.Mode = CipherMode.CBC;

//Создать дешифратор из существующих байтов ключа и инициализации

//вектора. Размер ключа определяется на основе номера ключа

//байты.

ICryptoTransform decryptor = symmetricKey.CreateDecryptor

(

keyBytes,

initVectorBytes

);

//Определите поток памяти, который будет использоваться для хранения зашифрованных данных.

MemoryStream memoryStream = new MemoryStream(cipherTextBytes);

//Определите криптографический поток (всегда используйте режим чтения для шифрования).

CryptoStream cryptoStream = new CryptoStream

(

memoryStream,

decryptor,

CryptoStreamMode.Read

);

byte[] plainTextBytes = new byte[cipherTextBytes.Length];

//Начните расшифровывать.

int decryptedByteCount = cryptoStream.Read

(

plainTextBytes,

0,

plainTextBytes.Length

);

//Закройте оба потока.

memoryStream.Close();

cryptoStream.Close();

//Преобразование расшифрованных данных в строку.

//Предположим, что исходная строка открытого текста была UTF8-encoded.

string plainText = Encoding.Default.GetString

(

plainTextBytes,

0,

decryptedByteCount

);

//Возвратите расшифрованную последовательность.

return plainText;

}

}

}

}

**Aktiv.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Net.Mail;

using System.Net;

using MySql.Data.MySqlClient;

using static budilnik.Sym\_alg;

namespace budilnik

{

public partial class aktiv : Form

{

public aktiv()

{

InitializeComponent();

}

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

{

clock c2 = new clock();

c2.Show();

Hide();

}

private void button1\_Click\_1(object sender, EventArgs e)

{

if (textBoxemail.Text == "")

{ MessageBox.Show("Введите почту");

return; }

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM `userskod` WHERE `usageble` = 1 LIMIT 1", db.getconnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

DataRow row = table.Rows[0];

String kodget = row.Field<string>("kodus");

MySqlCommand command22 = new MySqlCommand("UPDATE `userskod` SET `usageble` = 0 WHERE `id` = "+ row.Field<int>("id") +" ", db.getconnection());

db.openconnection();

if (command22.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка обновления статуса кода");

db.closeconnection();

byte[] b\_mess = new byte[1]; // сообщение в байтах

byte[] hash\_arr = new byte[1]; // хэш

Cr hash = new Cr();

b\_mess = System.Text.Encoding.Default.GetBytes(kodget);

hash\_arr = hash.SHA\_256(b\_mess);

String hashstr = System.Text.Encoding.Default.GetString(hash\_arr);

Sym\_alg SM = new Sym\_alg();

byte[] k = SM.key\_gen();

byte[] iv = SM.key\_gen();

String kstr = System.Text.Encoding.Default.GetString(k);

String ivstr = System.Text.Encoding.Default.GetString(iv);

String sumhk = hashstr + kstr;

byte[] encypt = new byte[1]; // шифротекст

byte[] sumhkbt = System.Text.Encoding.Default.GetBytes(sumhk);

//encypt = SM.Rijndaelt\_Encrypt(sumhkbt, k, iv);

string encyptstr = RijndaelAlgorithm.Encrypt

(

sumhk,

hashstr,

"salt",

"SHA256",

2,

ivstr,

256

);

MySqlCommand command23 = new MySqlCommand("UPDATE `userskod` SET `handk` = @uh, `keyforh` = @uk, `iv` = @uiv WHERE `id` = " + row.Field<int>("id") + " ", db.getconnection());

command23.Parameters.Add("@uh", MySqlDbType.VarChar).Value = encyptstr;

command23.Parameters.Add("@uk", MySqlDbType.VarChar).Value = kstr;

command23.Parameters.Add("@uiv", MySqlDbType.VarChar).Value = ivstr;

db.openconnection();

if (command23.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка обновления хэша");

db.closeconnection();

SmtpClient Smtp = new SmtpClient("smtp.mail.ru", 25);

Smtp.EnableSsl = true;

Smtp.Credentials = new NetworkCredential("rrtt.dcvv@mail.ru", "18120508fdcnhbz");

MailMessage Message = new MailMessage();

Message.From = new MailAddress("rrtt.dcvv@mail.ru");

Message.To.Add(new MailAddress(textBoxemail.Text));

Message.Subject = "Код активации";

Message.Body = kodget;

try

{

Smtp.Send(Message);

MessageBox.Show("Успешно отправлено");

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

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

{

String fromtextbox = maskedTextBox1.Text;

maskedTextBox1.Clear();

byte[] b\_mess1 = new byte[1]; // сообщение в байтах

byte[] hash\_arr2 = new byte[1]; // хэш

Cr hash1 = new Cr();

b\_mess1 = System.Text.Encoding.Default.GetBytes(fromtextbox);

hash\_arr2 = hash1.SHA\_256(b\_mess1);

String hashstrfromuser = System.Text.Encoding.Default.GetString(hash\_arr2);

byte[] decypt = new byte[1]; // дешифрованное сообщение

Sym\_alg SM = new Sym\_alg();

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM `userskod` WHERE `usageble` = 0", db.getconnection());

adapter.SelectCommand = command2;

adapter.Fill(table);

for (int i = 0; i < table.Rows.Count; ++i)

{

DataRow row = table.Rows[i];

String hashandkget = row.Field<string>("handk");

String kget = row.Field<string>("keyforh");

String ivget = row.Field<string>("iv");

byte[] ivgetbt = System.Text.Encoding.Default.GetBytes(ivget);

byte[] kgetbt = System.Text.Encoding.Default.GetBytes(kget);

byte[] hashandkgetbt = System.Text.Encoding.Default.GetBytes(hashandkget);

string hashandkget2 = RijndaelAlgorithm.Decrypt

(

hashandkget,

hashstrfromuser,

"salt",

"SHA256",

2,

ivget,

256

);

//decypt = SM.Rijndaelt\_Decrypt(hashandkgetbt, kgetbt, ivgetbt);

string razdel = hashandkget2.Substring(0, 32);

if (razdel == hashstrfromuser)

{

DataTable table1 = new DataTable();

MySqlDataAdapter adapter1 = new MySqlDataAdapter();

MySqlCommand command13 = new MySqlCommand("SELECT \* FROM `users` WHERE `login` = @uL", db.getconnection());

command13.Parameters.Add("@uL", MySqlDbType.VarChar).Value = Work\_sess.loguser;

adapter1.SelectCommand = command13;

adapter1.Fill(table1);

if (table1.Rows.Count > 0)

{

DataRow row1 = table1.Rows[0];

UInt32 userid = row1.Field<UInt32>("id");

MySqlCommand command24 = new MySqlCommand("UPDATE `users` SET `demo` = 1 WHERE `id` = " + userid + "", db.getconnection());

db.openconnection();

if (command24.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка обновления статуса аккаунта");

db.closeconnection();

clock c2 = new clock();

c2.Show();

Hide();

}

}

else

MessageBox.Show("Ошибка подтверждения");

}

}

private void aktiv\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

**Clock.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Media;

using MySql.Data.MySqlClient;

using static budilnik.Sym\_alg;

namespace budilnik

{

public partial class clock : Form

{

String soobshen;

Timer timer01 = new Timer();

Timer timer03 = new Timer();

SoundPlayer sp = new SoundPlayer(@"C:\Users\edika\Downloads\budilnik\budilnik\1.wav");

bool b = false;

public clock()

{

InitializeComponent();

}

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

{

button2.Enabled = false;

timer01.Interval = 1000;

timer01.Tick += new EventHandler(timer1\_Tick);

timer03.Interval = 60000;

timer03.Tick += new EventHandler(timer3\_Tick);

timer01.Start();

timer03.Start();

}

private void timer1\_Tick(object sender, EventArgs e)

{

label1.Text = DateTime.Now.Hour.ToString("00") + ":" + DateTime.Now.Minute.ToString("00") + ":" + DateTime.Now.Second.ToString("00");

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `users` WHERE `login` = @uL AND `pass` = @uP AND `demo` = 0", db.getconnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = Work\_sess.loguser;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = Work\_sess.passuser;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count == 0)

{

buttonaktiv.Visible = false;

}

}

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

{

if (b == false)

{

label2.Text = maskedTextBox1.Text;

timer2.Start();

maskedTextBox1.Visible = false;

button1.Text = "Убрать будильник";

b = true;

soobshen = textBox1.Text.ToString();

textBox1.Clear();

}

else if (b == true)

{

label2.Text = "00:00";

timer2.Stop();

maskedTextBox1.Visible = true;

button1.Text = "Завести будильник";

b = false;

}

}

private void timer2\_Tick(object sender, EventArgs e)

{

if (label1.Text == label2.Text + ":00")

{

String vremya = label2.Text;

button2.Enabled = true;

sp.Play();

MessageBox.Show(soobshen);

byte[] b\_mess = new byte[1]; // сообщение в байтах

b\_mess = System.Text.Encoding.Default.GetBytes(soobshen);

byte[] encypt = new byte[1]; // шифротекст

String KEY ; // ключ

String IV; // вектор инициализации

Sym\_alg SM = new Sym\_alg();

byte[] k = SM.key\_gen();

byte[] iv = SM.key\_gen();

KEY = System.Text.Encoding.Default.GetString(k);

IV = System.Text.Encoding.Default.GetString(iv);

//encypt = SM.Rijndaelt\_Encrypt(b\_mess, KEY, IV);

string encyptstr = RijndaelAlgorithm.Encrypt

(

soobshen,

KEY,

"salt",

"SHA256",

2,

IV,

256

);

DB db = new DB();

MySqlCommand command = new MySqlCommand("INSERT INTO `soobshenias` (`messages`, `login`, `k`,`iv`,`timemessage`) VALUES(@usmes, @login, @k, @iv,@uvr)", db.getconnection());

command.Parameters.Add("@login", MySqlDbType.VarChar).Value = Work\_sess.loguser;

command.Parameters.Add("@usmes", MySqlDbType.VarChar).Value = encyptstr;

command.Parameters.Add("@k", MySqlDbType.VarChar).Value = KEY;

command.Parameters.Add("@iv", MySqlDbType.VarChar).Value = IV;

command.Parameters.Add("@uvr", MySqlDbType.VarChar).Value = vremya;

db.openconnection();

if (command.ExecuteNonQuery() != 1)

MessageBox.Show("Ошибка отправки сообщения");

db.closeconnection();

}

}

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

{

sp.Stop();

button2.Enabled = false;

maskedTextBox1.Visible = true;

button1.Text = "Завести будильник";

b = false;

}

private void nameuserlog\_TextChanged(object sender, EventArgs e)

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

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

{

aktiv ak2 = new aktiv();

ak2.Show();

Hide();

}

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

{

author a2 = new author();

a2.Show();

Hide();

}

private void timer3\_Tick(object sender, EventArgs e)

{

DateTime.Now.Minute.ToString("00");

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `users` WHERE `login` = @uL AND `pass` = @uP AND `demo` = 0", db.getconnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = Work\_sess.loguser;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = Work\_sess.passuser;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Поддержите рзработчика, купив лицензию");

}

}

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

{

history h2 = new history();

h2.Show();

Hide();

}

private void clock\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

**History.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

using static budilnik.Sym\_alg;

namespace budilnik

{

public partial class history : Form

{

public history()

{

InitializeComponent();

}

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

{

}

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

{

}

private void dataGridView1\_CellContentClick\_1(object sender, DataGridViewCellEventArgs e)

{

}

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

{

clock c2 = new clock();

c2.Show();

Hide();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

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

{

byte[] decypt = new byte[1]; // дешифрованное сообщение

Sym\_alg SM = new Sym\_alg();

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command2 = new MySqlCommand("SELECT \* FROM `soobshenias` WHERE `login` = @uL", db.getconnection());

command2.Parameters.Add("@uL", MySqlDbType.VarChar).Value = Work\_sess.loguser;

adapter.SelectCommand = command2;

adapter.Fill(table);

for (int i = 0; i < table.Rows.Count; ++i)

{

DataRow row = table.Rows[i];

String messget = row.Field<string>("messages");

String kget = row.Field<string>("k");

String ivget = row.Field<string>("iv");

String vremyaget = row.Field<string>("timemessage");

byte[] ivgetbt = System.Text.Encoding.Default.GetBytes(ivget);

byte[] kgetbt = System.Text.Encoding.Default.GetBytes(kget);

byte[] messgetbt = System.Text.Encoding.Default.GetBytes(messget);

//decypt = SM.Rijndaelt\_Decrypt(messgetbt, kgetbt, ivgetbt);

byte[] cipherTextBytes = System.Text.Encoding.Default.GetBytes(messget);

string messgetde2 = RijndaelAlgorithm.Decrypt

(

messget,

kget,

"salt",

"SHA256",

2,

ivget,

256

);

textBox1.Text += "Cообщения "+Work\_sess.loguser+"["+i+"] Время сообщения " + vremyaget + ": " + "'" + messgetde2 + "'" + "\r\n";

}

}

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

{

textBox1.Clear();

}

private void history\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

**Author.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using MySql.Data.MySqlClient;

namespace budilnik

{

public partial class author : Form

{

public author()

{

InitializeComponent();

}

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

{

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void label2\_Click(object sender, EventArgs e)

{

}

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

{

Work\_sess.loguser = LoginField.Text;

Work\_sess.passuser = PassField.Text;

DB db = new DB();

DataTable table = new DataTable();

MySqlDataAdapter adapter = new MySqlDataAdapter();

MySqlCommand command = new MySqlCommand("SELECT \* FROM `users` WHERE `login` = @uL AND `pass` = @uP",db.getconnection());

command.Parameters.Add("@uL", MySqlDbType.VarChar).Value = Work\_sess.loguser;

command.Parameters.Add("@uP", MySqlDbType.VarChar).Value = Work\_sess.passuser;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

clock c2 = new clock();

c2.Show();

Hide();

}

else MessageBox.Show("Неправильные данные");

}

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

{

reg g2 = new reg();

g2.Show();

Hide();

}

private void author\_FormClosed(object sender, FormClosedEventArgs e)

{

}

}

}

**Work\_sess**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace budilnik

{

public static class Work\_sess

{

public static String loguser; //логин

public static String passuser; //имя

public static int demo\_level; //уровень доступа

}

}

**DB.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using MySql.Data.MySqlClient;

namespace budilnik

{

class DB

{

MySqlConnection connection = new MySqlConnection("server=localhost; port=3306; username=root; password=root; database=auth");

public void openconnection()

{

if (connection.State == System.Data.ConnectionState.Closed)

connection.Open();

}

public void closeconnection()

{

if (connection.State == System.Data.ConnectionState.Open)

connection.Close();

}

public MySqlConnection getconnection()

{

return connection;

}

}

}

1. **Выводы**

В ходе выполненной расчетно-графического задания было разработано программное средство для демонстрации будильника с моделью распространения программы Nagware и табличным методом проверки регистрационных ключей. В приложении были применены средства защиты информации, такие, как:

1. Хэширование кодов
2. Кодирование сообщении и кодов.
3. Динамический SQL.
4. Использование ключа в установщике с целью ограничения копирования программы.