

Федеральное государственное автономное образовательное учреждение высшего образования

«Уральский федеральный университет имени первого Президента России Б.Н. Ельцина» (УрФУ) Институт радиоэлектроники и информационных технологий – РтФ

Оценка \_\_\_\_\_\_\_\_\_\_

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА ПО КУРСОВОЙ РАБОТЕ «ПАРОЛЬНЫЙ МЕНЕДЖЕР»**

**ПО ДИСЦИПЛИНЕ: «ЯЗЫКИ И МЕТОДЫ ПРОГРАММИРОВАНИЯ»**

|  |  |
| --- | --- |
| Преподаватель | Мирвода С.Г. |
| Студенты | Голоскоков А., Мосунов Р., Борисихин Л. |
| Специальность | Информационно-аналитические системы безопасности |
| Группа | РИ-311055 |

Екатеринбург

2023

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 3](#_Toc157014278)

[1 Цель и задачи проекта 3](#_Toc157014279)

[2 Видение проекта 3](#_Toc157014280)

[БЛОК ПРОЕКТИРОВАНИЯ 5](#_Toc157014281)

[1 Архитектура продукта 5](#_Toc157014282)

[2 Требования к системному ПО 7](#_Toc157014283)

[БЛОК РАЗРАБОТКИ 8](#_Toc157014284)

[ПРИЛОЖЕНИЕ 1 10](#_Toc157014285)

[1 Код приложения на C# 10](#_Toc157014286)

[1.1 Окно регистрации 10](#_Toc157014287)

[1.2 Окно авторизации 11](#_Toc157014288)

[1.3 Окно просмотра добавленных сервисов 12](#_Toc157014289)

[1.4 Окно добавления новых сервисов 15](#_Toc157014290)

[1.5 Окно просмотра банковских карт 17](#_Toc157014291)

[1.6 Окно добавления банковских карт 19](#_Toc157014292)

[1.7 Подключение к базе данных 21](#_Toc157014293)

[Приложение 2 23](#_Toc157014294)

[1 Код базы данных PostgreSQL 23](#_Toc157014295)

[1.1 Создание таблиц 23](#_Toc157014296)

[1.2 Политики безопасности 23](#_Toc157014297)

[1.3 Функции для взаимодействия базы данных и приложения 24](#_Toc157014298)

# ВВЕДЕНИЕ

**1 Цель и задачи проекта**

Цель: Создание безопасного и удобного в использовании менеджера паролей, который обеспечивает защиту персональных данных пользователей, предоставляет интуитивно понятный интерфейс для управления паролями, а также взаимодействует с зашифрованной базой данных для надежного хранения и управления учетными записями и паролями пользователей.

Задачи для выполнения поставленной цели:

1. Спроектировать работу приложения: как пользователь будет взаимодействовать с приложением и как приложение будет взаимодействовать с базой данных;
2. Спроектировать удобный интерфейс приложения и его функционал;
3. Выбрать язык программирования;
4. Создать базу данных, в которой будут храниться зашифрованные данные пользователя;
5. Создать оконное приложение «Менеджер паролей» и подключить его к базе данных.

**2 Видение проекта**

С ускоряющимся развитием интернет-бизнеса и рвением людей перенести всё в цифровой формат для удобства, появилась проблема, вытекающая из большого количества сервисов, которыми мы пользуемся – проблема безопасного хранения множества паролей от этих сервисов.

Наше приложение «Менеджер паролей» сможет шифровать не только данные о сервисах, но и данные о банковских картах (номер, CVC-код, дату). Также будут реализованы функции генерации надёжного пароля и проверки введённого пароля по базе с утёкшими паролями.

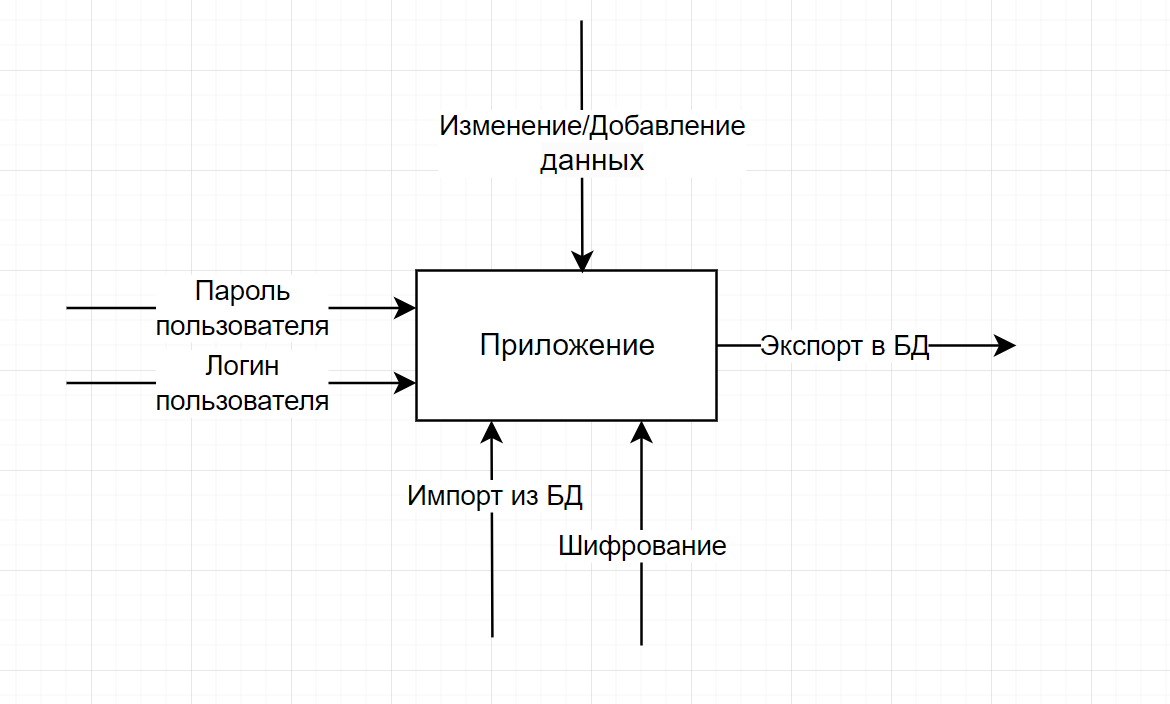


Рисунок 1 - Функциональная диаграмма продукта как он видится

На вход приложению поступают данные от пользователя: логин и пароль от аккаунта, а также, после входа в аккаунт, данные о сервисах и кредитных картах (пароли, названия сервисов/номера карт). На выход приложение создаёт запрос к базе данных на изменение или добавление информации, а приложение обрабатывает подобные запросы.

**БЛОК ПРОЕКТИРОВАНИЯ**

**1 Архитектура продукта**

При проектировании продукта было решено сделать интерфейс удобным и интуитивно понятным. При использовании приложения пользователи могут взаимодействовать с базой данных, посредством добавления и изменения записей о сервисах и банковских карт, которыми они пользуются.

При проектировании продукта были созданы следующие архитектурные диаграммы:

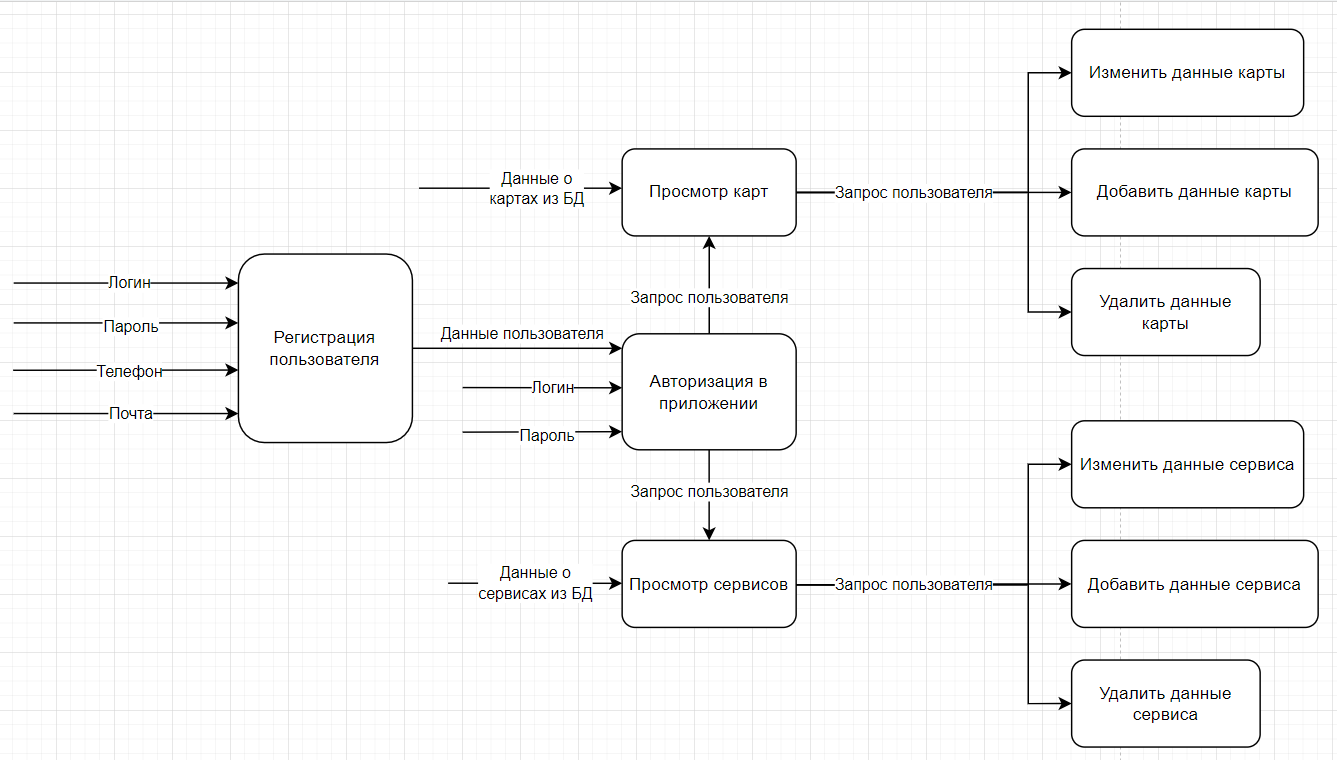


Рисунок 2 – Функциональная диаграмма

Изображение выглядит как диаграмма, линия, План, Технический чертеж

Автоматически созданное описание

Рисунок 3 – Архитектурная диаграмма

Изображение выглядит как текст, диаграмма, снимок экрана, Параллельный

Автоматически созданное описание

Рисунок 4 – Диаграмма Классов

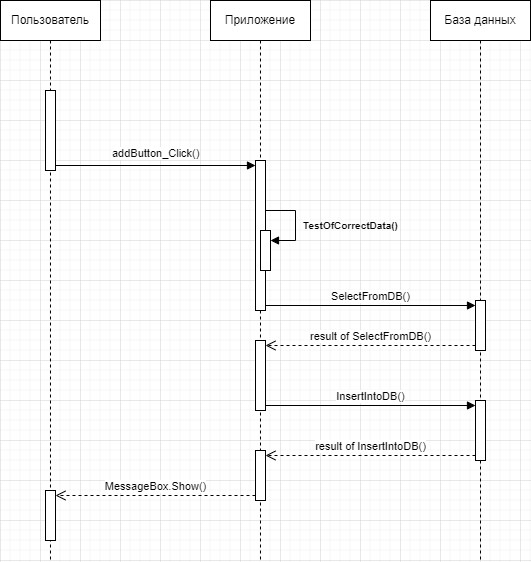


Рисунок 5 – Диаграмма последовательности (для функции добавления записи)

Изображение выглядит как текст, снимок экрана, диаграмма, Параллельный

Автоматически созданное описание

Рисунок 6 - Концептуальная модель БД

**2 Требования к системному ПО**

Приложение создано для его использования на персональных компьютерах. Рекомендуется использовать операционные системы Windows 10 и Windows 11, так как на более старых версиях Windows приложение может работать с ограничениями.

**БЛОК РАЗРАБОТКИ**

Для создания нашего приложения были выбраны следующие языки программирования:

1. **C# -** является отличным выбором, так как удобен для разработки приложений, благодаря своему чистому синтаксису, высокой читаемости кода, платформенной независимости в рамках .NET, обширным библиотекам и фреймворкам а также мощным инструментам разработки, таким как Visual Studio.
2. **PostgreSQL** - для работы с базами данных. Этот язык обладает выдающейся надежностью, расширяемостью и поддержкой стандартов SQL.

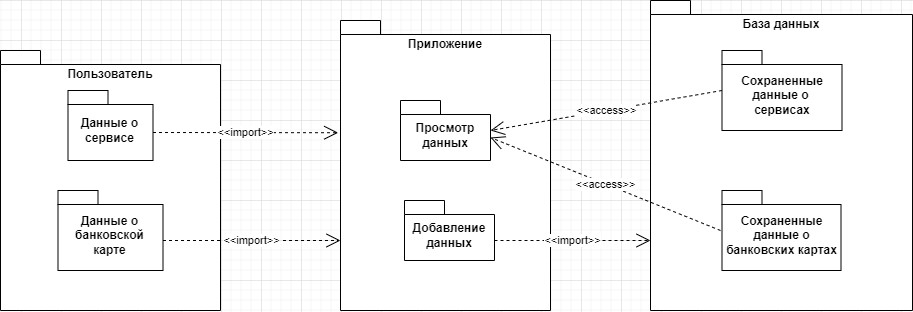


Рисунок 7 – Диаграмма пакетов

Изображение выглядит как текст, диаграмма, Параллельный, снимок экрана

Автоматически созданное описание

Рисунок 8 – Физическая модель БД

Таблица users предназначена для хранения данных о пользователе. В нее входят колонки:

* user\_id – id пользователя;
* login – login пользователя;
* email – электронная почта пользователя;
* passwd – мастер-пароль пользователя;
* phone\_number - номер телефона пользователя.

Таблица services предназначена для хранения данных о сервисах пользователя. В нее входят колонки:

* service\_id – id сервиса;
* user\_id – id пользователя;
* service\_name – название сервиса;
* login\_service – login, используемый в сервисе;
* passwd\_service – пароль от сервиса.

Таблица credits предназначена для хранения данных о банковских картах пользователя. В нее входят колонки:

* credit\_id – id карты;
* user\_id – id пользователя;
* credit\_number – номер банковской карты;
* credit\_date – дата, до которой пригодна карта;
* credit\_code – CVC код.

# ПРИЛОЖЕНИЕ 1

## Код приложения на C#

### Окно регистрации

Ниже представлена часть кода, содержащая основную логику работы приложения:

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

{

string login = loginField.Text;

string passwd = passwdField.Text;

string email = EmailField.Text;

string phone = PhoneField.Text;

Regex phoneRegex = new Regex("^\\+?[1-9][0-9]{7,14}$");

Regex emailRegex = new Regex("^\\w+@[a-zA-Z\_]+?\\.[a-zA-Z]{2,6}$");

string passwordHash = BCrypt.Net.BCrypt.EnhancedHashPassword(passwd, 13);

if (EmailField.Text == "Email" || emailRegex.IsMatch(EmailField.Text) == false)

{

MessageBox.Show("Неверный формат почты");

Email.ForeColor = Color.Red;

phoneNumber.ForeColor = SystemColors.HotTrack;

Login.ForeColor = SystemColors.HotTrack;

Password.ForeColor = SystemColors.HotTrack;

RepeatPassword.ForeColor = SystemColors.HotTrack;

}

else if (PhoneField.Text == "Phone Number" || phoneRegex.IsMatch(PhoneField.Text) == false)

{

MessageBox.Show("Неверный формат телефона");

phoneNumber.ForeColor = Color.Red;

Email.ForeColor = SystemColors.HotTrack;

Login.ForeColor = SystemColors.HotTrack;

Password.ForeColor = SystemColors.HotTrack;

RepeatPassword.ForeColor = SystemColors.HotTrack;

}

else if (loginField.Text == "" || loginField.Text == "Login")

{

MessageBox.Show("Введите логин");

Login.ForeColor = Color.Red;

Email.ForeColor = SystemColors.HotTrack;

phoneNumber.ForeColor = SystemColors.HotTrack;

Password.ForeColor = SystemColors.HotTrack;

RepeatPassword.ForeColor = SystemColors.HotTrack;

}

else if (passwdField.Text == "Password")

{

MessageBox.Show("Недопустимое занчение пароля");

Password.ForeColor = Color.Red;

Email.ForeColor = SystemColors.HotTrack;

phoneNumber.ForeColor = SystemColors.HotTrack;

Login.ForeColor = SystemColors.HotTrack;

RepeatPassword.ForeColor = SystemColors.HotTrack;

}

else if (passwdField.Text != passwdField2.Text)

{

MessageBox.Show("Пароли не совпадают");

Password.ForeColor = Color.Red;

RepeatPassword.ForeColor = Color.Red;

Email.ForeColor = SystemColors.HotTrack;

phoneNumber.ForeColor = SystemColors.HotTrack;

Login.ForeColor = SystemColors.HotTrack;

}

else

{

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT \* FROM users WHERE email = @um or phone\_number = @uph or login = @ul", db.GetConnection("postgres", "Slayanin2003"));

command.Parameters.Add("@um", NpgsqlTypes.NpgsqlDbType.Text).Value = email;

command.Parameters.Add("@uph", NpgsqlTypes.NpgsqlDbType.Text).Value = phone;

command.Parameters.Add("@ul", NpgsqlTypes.NpgsqlDbType.Text).Value = login;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Пользователь с такими данными уже существует");

}

else

{

command = new Npgsql.NpgsqlCommand("INSERT INTO users (email, passwd, phone\_number, login) values (@um, @up, @uph, @ul)", db.GetConnection("postgres", "Slayanin2003"));

command.Parameters.Add("@um", NpgsqlTypes.NpgsqlDbType.Text).Value = email;

command.Parameters.Add("@up", NpgsqlTypes.NpgsqlDbType.Text).Value = passwordHash;

command.Parameters.Add("@uph", NpgsqlTypes.NpgsqlDbType.Text).Value = phone;

command.Parameters.Add("@ul", NpgsqlTypes.NpgsqlDbType.Text).Value = login;

adapter.SelectCommand = command;

adapter.Fill(table);

command = new Npgsql.NpgsqlCommand("SELECT create\_user(@ul, @up)", db.GetConnection("postgres", "Slayanin2003"));

command.Parameters.Add("@ul", NpgsqlTypes.NpgsqlDbType.Text).Value = login;

command.Parameters.Add("up", NpgsqlTypes.NpgsqlDbType.Text).Value = passwd;

adapter.SelectCommand = command;

adapter.Fill(table);

MessageBox.Show("Успешная Регистрация");

this.Hide();

PasswdManager loginform = new PasswdManager();

loginform.Show();

}

}

}

### Окно авторизации

Ниже представлена часть кода, содержащая основную логику работы приложения:

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

{

login = loginField.Text;

password = passwdField.Text;

string passwordHash = "";

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT \* FROM users WHERE login = @ul", db.GetConnection("postgres", "Slayanin2003"));

command.Parameters.Add("@ul", NpgsqlTypes.NpgsqlDbType.Text).Value = login;

adapter.SelectCommand = command;

adapter.Fill(table);

DataRow[] rows = table.Select();

if (table.Rows.Count == 1)

{

passwordHash = $"{rows[0]["passwd"]}";

if (BCrypt.Net.BCrypt.EnhancedVerify(password, passwordHash) == true)

{

this.Hide();

MainMenu mainMenu = new MainMenu();

mainMenu.Show();

}

else

{

MessageBox.Show("Ошибка авторизации!");

}

}

else

MessageBox.Show("Ошибка авторизации!");

}

### Окно просмотра добавленных сервисов

Ниже представлена часть кода, содержащая основную логику работы приложения:

public MainMenu()

{

InitializeComponent();

int addService\_X = 86, addService\_Y = 100;

int serviceName\_X = 97;

int serviceName\_Y = 102;

int serviceLogin\_Y = 102;

int servicePasswd\_Y = 102;

int pSN\_X = 90;

int pSN\_Y = 100;

int pSL\_Y = 100;

int pSP\_Y = 100;

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT service\_name, login\_service, passwd\_service FROM services", db.GetConnection(PasswdManager.login, PasswdManager.password));

adapter.SelectCommand = command;

adapter.Fill(table);

DataRow[] rows = table.Select();

if (table.Rows.Count > 0 )

{

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

{

Label serviceName = new Label();

Label serviceLogin = new Label();

Label servicePasswd = new Label();

Panel pSN = new Panel();

Panel pSL = new Panel();

Panel pSP = new Panel();

serviceName.Location = new Point(serviceName\_X, serviceName\_Y);

serviceName.AutoSize = true;

serviceName.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

serviceName.ForeColor = Color.DimGray;

serviceName.Name = $"serviceName{i}";

serviceName.Text = $"{rows[i]["service\_name"]}";

serviceName.BackColor = SystemColors.GradientInactiveCaption;

serviceName.MouseEnter += new System.EventHandler(Object\_MouseEnter);

serviceName.MouseLeave += new System.EventHandler(Object\_MouseLeave);

serviceName.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

this.Controls.Add(serviceName);

pSN.Location = new Point(pSN\_X, pSN\_Y);

pSN.Name = $"panelNumberCard{i}";

pSN.Size = new Size(serviceName.Size.Width + 13, serviceName.Size.Height + 5);

pSN.BackColor = SystemColors.GradientInactiveCaption;

this.Controls.Add(pSN);

int serviceLogin\_X = pSN.Size.Width + 100;

serviceLogin.Location = new Point(serviceLogin\_X, serviceLogin\_Y);

serviceLogin.AutoSize = true;

serviceLogin.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

serviceLogin.ForeColor = Color.DimGray;

serviceLogin.Name = $"serviceLogin{i}";

serviceLogin.Text = $"{rows[i]["login\_service"]}";

serviceLogin.BackColor = SystemColors.GradientInactiveCaption;

serviceLogin.MouseEnter += new System.EventHandler(Object\_MouseEnter);

serviceLogin.MouseLeave += new System.EventHandler(Object\_MouseLeave);

serviceLogin.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

this.Controls.Add(serviceLogin);

int pSL\_X = pSN.Size.Width + 94;

pSL.Location = new Point(pSL\_X, pSL\_Y);

pSL.Name = $"panelDateCard{i}";

pSL.Size = new Size(serviceLogin.Size.Width + 13, serviceLogin.Size.Height + 5);

pSL.BackColor = SystemColors.GradientInactiveCaption;

this.Controls.Add(pSL);

int servicePasswd\_X = pSN.Size.Width + pSL.Size.Width + 104;

servicePasswd.Location = new Point(servicePasswd\_X, servicePasswd\_Y);

servicePasswd.AutoSize = true;

servicePasswd.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

servicePasswd.ForeColor = Color.DimGray;

servicePasswd.Name = $"numberCard{i}";

servicePasswd.Text = $"{rows[i]["passwd\_service"]}";

servicePasswd.BackColor = SystemColors.GradientInactiveCaption;

servicePasswd.MouseEnter += new System.EventHandler(Object\_MouseEnter);

servicePasswd.MouseLeave += new System.EventHandler(Object\_MouseLeave);

servicePasswd.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

this.Controls.Add(servicePasswd);

int pSP\_X = pSN.Size.Width + pSL.Size.Width + 98;

pSP.Location = new Point(pSP\_X, pSP\_Y);

pSP.Name = $"panelCVC{i}";

pSP.Size = new Size(servicePasswd.Size.Width + 13, servicePasswd.Size.Height + 5);

pSP.BackColor = SystemColors.GradientInactiveCaption;

this.Controls.Add(pSP);

serviceName\_Y += 45;

serviceLogin\_Y += 45;

servicePasswd\_Y += 45;

pSN\_Y += 45;

pSL\_Y += 45;

pSP\_Y += 45;

addService\_Y = pSN\_Y;

if (pSN.Width + pSL.Width + pSP.Width + 66 > 518)

{

int scroll = 0;

hScrollBar.Visible = true;

hScrollBar.Scroll += new ScrollEventHandler(hScrollBar\_Scroll);

void hScrollBar\_Scroll(object sender, ScrollEventArgs e)

{

if (scroll > hScrollBar.Value)

{

serviceName.Location = new Point(serviceName.Location.X + 10, serviceName.Location.Y);

serviceLogin.Location = new Point(serviceLogin.Location.X + 10, serviceLogin.Location.Y);

servicePasswd.Location = new Point(servicePasswd.Location.X + 10, servicePasswd.Location.Y);

pSN.Location = new Point(pSN.Location.X + 10, pSN.Location.Y);

pSL.Location = new Point(pSL.Location.X + 10, pSL.Location.Y);

pSP.Location = new Point(pSP.Location.X + 10, pSP.Location.Y);

}

else if (scroll < hScrollBar.Value)

{

serviceName.Location = new Point(serviceName.Location.X - 10, serviceName.Location.Y);

serviceLogin.Location = new Point(serviceLogin.Location.X - 10, serviceLogin.Location.Y);

servicePasswd.Location = new Point(servicePasswd.Location.X - 10, servicePasswd.Location.Y);

pSN.Location = new Point(pSN.Location.X - 10, pSN.Location.Y);

pSL.Location = new Point(pSL.Location.X - 10, pSL.Location.Y);

pSP.Location = new Point(pSP.Location.X - 10, pSP.Location.Y);

}

scroll = hScrollBar.Value;

}

}

void Object\_MouseEnter(object sender, EventArgs e)

{

var label = (Label)sender;

label.ForeColor = Color.Black;

}

void Object\_MouseLeave(object sender, EventArgs e)

{

var label = (Label)sender;

label.ForeColor = Color.DimGray;

}

void Object\_MouseClick(object sender, EventArgs e)

{

Label label = (Label)sender;

if (label != null)

{

Clipboard.SetText(label.Text, TextDataFormat.UnicodeText);

labelCopy.Location = new Point(label.Location.X, label.Location.Y + 25);

labelCopy.Visible = true;

}

}

}

}

Button addService = new Button();

addService.BackColor = System.Drawing.SystemColors.HotTrack;

addService.Cursor = System.Windows.Forms.Cursors.Hand;

addService.Font = new Font("Segoe UI Semibold", 7.8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

addService.Location = new Point(addService\_X, addService\_Y);

addService.Name = "loginButton";

addService.Size = new Size(100, 27);

addService.TabIndex = 4;

addService.Text = "Add New";

addService.UseVisualStyleBackColor = false;

addService.MouseClick += new MouseEventHandler(addService\_MouseClick);

this.Controls.Add(addService);

}

### Окно добавления новых сервисов

Ниже представлена часть кода, содержащая основную логику работы приложения:

private void addButton\_MouseClick(object sender, MouseEventArgs e)

{

if (textServiceName.Text == "")

{

MessageBox.Show("Введите имя сервиса");

labelServiceName.ForeColor = Color.Red;

labelLogin.ForeColor = SystemColors.HotTrack;

labelPassword.ForeColor = SystemColors.HotTrack;

}

else if (textLogin.Text == "")

{

MessageBox.Show("Введите логин");

labelLogin.ForeColor = Color.Red;

labelServiceName.ForeColor = SystemColors.HotTrack;

labelPassword.ForeColor = SystemColors.HotTrack;

}

else if (textPassword.Text == "")

{

MessageBox.Show("Введите пароль");

labelPassword.ForeColor = Color.Red;

labelLogin.ForeColor = SystemColors.HotTrack;

labelServiceName.ForeColor = SystemColors.HotTrack;

}

else

{

string serviceName = textServiceName.Text;

string serviceLogin = textLogin.Text;

string servicePassword = textPassword.Text;

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT \* FROM credits WHERE service\_name = @sn and login\_service = @sl and passwd\_service = @sp", db.GetConnection(PasswdManager.login, PasswdManager.password));

command.Parameters.Add("@sn", NpgsqlTypes.NpgsqlDbType.Text).Value = serviceName;

command.Parameters.Add("@sl", NpgsqlTypes.NpgsqlDbType.Text).Value = serviceLogin;

command.Parameters.Add("@sp", NpgsqlTypes.NpgsqlDbType.Text).Value = servicePassword;

if (table.Rows.Count > 0)

{

MessageBox.Show($"Вы уже сохраняли эти данные для {serviceName}!");

}

else

{

command = new Npgsql.NpgsqlCommand("SELECT insert\_service(@sn, @sl, @sp)", db.GetConnection(PasswdManager.login, PasswdManager.password));

command.Parameters.Add("@sn", NpgsqlTypes.NpgsqlDbType.Text).Value = serviceName;

command.Parameters.Add("@sl", NpgsqlTypes.NpgsqlDbType.Text).Value = serviceLogin;

command.Parameters.Add("@sp", NpgsqlTypes.NpgsqlDbType.Text).Value = servicePassword;

adapter.SelectCommand = command;

adapter.Fill(table);

MessageBox.Show("Данные успешно добавлены!");

this.Hide();

MainMenu mainMenu = new MainMenu();

mainMenu.Show();

}

}

}

### Окно просмотра банковских карт

Ниже представлена часть кода, содержащая основную логику работы приложения:

public MainMenu2()

{

InitializeComponent();

int panelNumberCard\_X = 86;

int panelNumberCard\_Y = 100;

int panelNumberCard\_Size\_X = 191;

int panelNumberCard\_Size\_Y = 25;

int panelDateCard\_X = 290;

int panelDateCard\_Y = 100;

int panelDateCard\_Size\_X = 53;

int panelDateCard\_Size\_Y = 25;

int panelCVC\_X = 355;

int panelCVC\_Y = 100;

int panelCVC\_Size\_X = 36;

int panelCVC\_Size\_Y = 25;

int numberCard\_X = 97;

int numberCard\_Y = 102;

int dateCard\_X = 290;

int dateCard\_Y = 102;

int cvc\_X = 355;

int cvc\_Y = 102;

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT credit\_number, credit\_date, credit\_code FROM credits", db.GetConnection(PasswdManager.login, PasswdManager.password));

adapter.SelectCommand = command;

adapter.Fill(table);

DataRow[] rows = table.Select();

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

{

Label numberCard = new Label();

Label dateCard = new Label();

Label cvc = new Label();

Panel pNC = new Panel();

Panel pDC = new Panel();

Panel pCVC = new Panel();

numberCard.Location = new Point(numberCard\_X, numberCard\_Y);

numberCard.AutoSize = true;

numberCard.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

numberCard.ForeColor = Color.DimGray;

numberCard.Name = $"numberCard{i}";

numberCard.Text = $"{rows[i]["credit\_number"]}";

numberCard.BackColor = SystemColors.GradientInactiveCaption;

numberCard.MouseEnter += new System.EventHandler(Object\_MouseEnter);

numberCard.MouseLeave += new System.EventHandler(Object\_MouseLeave);

numberCard.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

dateCard.Location = new Point(dateCard\_X, dateCard\_Y);

dateCard.AutoSize = true;

dateCard.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

dateCard.ForeColor = Color.DimGray;

dateCard.Name = $"numberCard{i}";

dateCard.Text = $"{rows[i]["credit\_date"]}";

dateCard.BackColor = SystemColors.GradientInactiveCaption;

dateCard.MouseEnter += new System.EventHandler(Object\_MouseEnter);

dateCard.MouseLeave += new System.EventHandler(Object\_MouseLeave);

dateCard.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

cvc.Location = new Point(cvc\_X, cvc\_Y);

cvc.AutoSize = true;

cvc.Font = new Font("Segoe UI", 11F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));

cvc.ForeColor = Color.DimGray;

cvc.Name = $"numberCard{i}";

cvc.Text = $"{rows[i]["credit\_code"]}";

cvc.BackColor = SystemColors.GradientInactiveCaption;

cvc.MouseEnter += new System.EventHandler(Object\_MouseEnter);

cvc.MouseLeave += new System.EventHandler(Object\_MouseLeave);

cvc.MouseClick += new System.Windows.Forms.MouseEventHandler(Object\_MouseClick);

this.Controls.Add(numberCard);

this.Controls.Add(dateCard);

this.Controls.Add(cvc);

numberCard\_Y += 45;

dateCard\_Y += 45;

cvc\_Y += 45;

pNC.Location = new Point(panelNumberCard\_X, panelNumberCard\_Y);

pNC.Name = $"panelNumberCard{i}";

pNC.Size = new Size(panelNumberCard\_Size\_X, panelNumberCard\_Size\_Y);

pNC.BackColor = SystemColors.GradientInactiveCaption;

pDC.Location = new Point(panelDateCard\_X, panelDateCard\_Y);

pDC.Name = $"panelDateCard{i}";

pDC.Size = new Size(panelDateCard\_Size\_X, panelDateCard\_Size\_Y);

pDC.BackColor = SystemColors.GradientInactiveCaption;

pCVC.Location = new Point(panelCVC\_X, panelCVC\_Y);

pCVC.Name = $"panelCVC{i}";

pCVC.Size = new Size(panelCVC\_Size\_X, panelCVC\_Size\_Y);

pCVC.BackColor = SystemColors.GradientInactiveCaption;

this.Controls.Add(pNC);

this.Controls.Add(pDC);

this.Controls.Add(pCVC);

panelNumberCard\_Y += 45;

panelDateCard\_Y += 45;

panelCVC\_Y += 45;

void Object\_MouseEnter(object sender, EventArgs e)

{

var label = (Label)sender;

label.ForeColor = Color.Black;

}

void Object\_MouseLeave(object sender, EventArgs e)

{

var label = (Label)sender;

label.ForeColor = Color.DimGray;

}

void Object\_MouseClick(object sender, EventArgs e)

{

Label label = (Label)sender;

if (label != null)

{

Clipboard.SetText(label.Text, TextDataFormat.UnicodeText);

labelCopy.Location = new Point(label.Location.X,label.Location.Y+25);

labelCopy.Visible = true;

}

}

}

Button addCard = new Button();

addCard.BackColor = System.Drawing.SystemColors.HotTrack;

addCard.Cursor = System.Windows.Forms.Cursors.Hand;

addCard.Font = new Font("Segoe UI Semibold", 7.8F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(204)));

addCard.Location = new Point(panelNumberCard\_X, panelNumberCard\_Y);

addCard.Name = "loginButton";

addCard.Size = new Size(100, 27);

addCard.TabIndex = 4;

addCard.Text = "Add New";

addCard.UseVisualStyleBackColor = false;

addCard.MouseClick += new MouseEventHandler(addCard\_MouseClick);

this.Controls.Add(addCard);

}

### Окно добавления банковских карт

Ниже представлена часть кода, содержащая основную логику работы приложения:

private void addButton\_MouseClick(object sender, MouseEventArgs e)

{

string formatCardNumber = "";

string checkDate = "";

int count = 0;

Regex cardNumberRegex = new Regex("[0-9]{16}$");

Regex dateRegex = new Regex("[0-9][0-9]/[0-9][0-9]");

Regex cvcRegex = new Regex("[0-9]{3}$");

if (textCardNumber.Text.Length == 16)

{

formatCardNumber = textCardNumber.Text;

formatCardNumber = textCardNumber.Text.Insert(4, " ");

formatCardNumber = formatCardNumber.Insert(9, " ");

formatCardNumber = formatCardNumber.Insert(14, " ");

}

if (dateRegex.IsMatch(textDate.Text) == false)

{

MessageBox.Show("Неверная дата!");

labelDate.ForeColor = Color.Red;

labelNumberCard.ForeColor = SystemColors.HotTrack;

labelCVC.ForeColor = SystemColors.HotTrack;

return;

}

else

{

for (int i = 0; i < textDate.Text.Length; i++)

{

if (textDate.Text[i] != '/')

{

count += 1;

checkDate += Convert.ToString(textDate.Text[i]);

if (count == 2)

{

if (Convert.ToInt32(checkDate) > 12)

{

MessageBox.Show("Неверная дата!");

labelDate.ForeColor = Color.Red;

labelNumberCard.ForeColor = SystemColors.HotTrack;

labelCVC.ForeColor = SystemColors.HotTrack;

return;

}

checkDate = "";

}

else

{

if (Convert.ToInt32(checkDate) > 99)

{

MessageBox.Show("Неверная дата!");

labelDate.ForeColor = Color.Red;

labelNumberCard.ForeColor = SystemColors.HotTrack;

labelCVC.ForeColor = SystemColors.HotTrack;

return;

}

}

}

}

}

if (cardNumberRegex.IsMatch(textCardNumber.Text) == false)

{

MessageBox.Show("Неверный номер карты!");

labelNumberCard.ForeColor = Color.Red;

labelDate.ForeColor = SystemColors.HotTrack;

labelCVC.ForeColor = SystemColors.HotTrack;

}

else if (dateRegex.IsMatch(textDate.Text) == false)

{

MessageBox.Show("Неверная дата!");

labelDate.ForeColor = Color.Red;

labelNumberCard.ForeColor = SystemColors.HotTrack;

labelCVC.ForeColor = SystemColors.HotTrack;

}

else if (cvcRegex.IsMatch(textCVC.Text) == false || textCVC.Text.Length != 3)

{

MessageBox.Show("Неверный СVC код!");

labelCVC.ForeColor = Color.Red;

labelDate.ForeColor = SystemColors.HotTrack;

labelNumberCard.ForeColor = SystemColors.HotTrack;

}

else

{

string cardNumber = formatCardNumber;

string date = textDate.Text;

string cvc = textCVC.Text;

DB db = new DB();

DataTable table = new DataTable();

Npgsql.NpgsqlDataAdapter adapter = new Npgsql.NpgsqlDataAdapter();

Npgsql.NpgsqlCommand command = new Npgsql.NpgsqlCommand("SELECT \* FROM credits WHERE credit\_number = @cn and credit\_date = @cd and credit\_code = @cc", db.GetConnection(PasswdManager.login, PasswdManager.password));

command.Parameters.Add("@cn", NpgsqlTypes.NpgsqlDbType.Text).Value = cardNumber;

command.Parameters.Add("@cd", NpgsqlTypes.NpgsqlDbType.Text).Value = date;

command.Parameters.Add("@cc", NpgsqlTypes.NpgsqlDbType.Text).Value = cvc;

adapter.SelectCommand = command;

adapter.Fill(table);

if (table.Rows.Count > 0)

{

MessageBox.Show("Вы уже сохраняли эту карту!");

}

else

{

command = new Npgsql.NpgsqlCommand("SELECT insert\_credit(@cn, @cd, @cc)", db.GetConnection(PasswdManager.login, PasswdManager.password));

command.Parameters.Add("@cn", NpgsqlTypes.NpgsqlDbType.Text).Value = cardNumber;

command.Parameters.Add("@cd", NpgsqlTypes.NpgsqlDbType.Text).Value = date;

command.Parameters.Add("@cc", NpgsqlTypes.NpgsqlDbType.Text).Value = cvc;

adapter.SelectCommand = command;

adapter.Fill(table);

MessageBox.Show("Карта успешно добавлена!");

this.Hide();

MainMenu2 mainMenu2 = new MainMenu2();

mainMenu2.Show();

}

}

### Подключение к базе данных

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Web;

using Npgsql;

namespace Курсач

{

internal class DB

{

NpgsqlConnection conn;

public void openConnection(string user, string passwd)

{

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

{

conn.Open();

}

}

public void closeConnection()

{

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

conn.Open();

}

public NpgsqlConnection GetConnection(string user, string passwd)

{

conn = new NpgsqlConnection($"Host = localhost; Username = {user}; Password = {passwd}; Database = passmanager");

return conn;

}

}

}

# Приложение 2

## Код базы данных PostgreSQL

### 1.1 Создание таблиц

CREATE TABLE public.users (

"user\_id" serial NOT NULL UNIQUE,

"login" TEXT NOT NULL UNIQUE,

"email" TEXT NOT NULL UNIQUE,

"passwd" TEXT NOT NULL,

"phone\_number" TEXT NOT NULL UNIQUE,

CONSTRAINT "users\_pk" PRIMARY KEY ("user\_id")

) WITH (

OIDS=FALSE

);

CREATE TABLE public.services (

"service\_id" serial NOT NULL UNIQUE,

"user\_id" integer NOT NULL,

"service\_name" TEXT NOT NULL,

"login\_service" TEXT NOT NULL,

"passwd\_service" TEXT NOT NULL,

CONSTRAINT "services\_pk" PRIMARY KEY ("service\_id")

) WITH (

OIDS=FALSE

);

CREATE TABLE public.credits (

"credit\_id" serial NOT NULL UNIQUE,

"user\_id" integer NOT NULL,

"credit\_number" text NOT NULL,

"credit\_date" text NOT NULL,

"credit\_code" integer NOT NULL,

CONSTRAINT "credits\_pk" PRIMARY KEY ("credit\_id")

) WITH (

OIDS=FALSE

);

ALTER TABLE "services" ADD CONSTRAINT "services\_fk0" FOREIGN KEY ("user\_id") REFERENCES "users"("user\_id");

ALTER TABLE "credits" ADD CONSTRAINT "credits\_fk0" FOREIGN KEY ("user\_id") REFERENCES "users"("user\_id");

### Политики безопасности

СREATE ROLE APP\_USER;

CREATE ROLE APP\_GUEST;

GRANT USAGE ON SCHEMA public TO APP\_USER;

GRANT USAGE ON SCHEMA public TO APP\_GUEST;

GRANT SELECT, INSERT, UPDATE, DELETE ON users to APP\_USER;

GRANT SELECT, INSERT, UPDATE, DELETE ON credits to APP\_USER;

GRANT SELECT, INSERT, UPDATE, DELETE ON services to APP\_USER;

GRANT INSERT ON users to APP\_GUEST;

CREATE POLICY user\_access ON users

USING (login = current\_user);

CREATE POLICY credits\_access ON credits

USING (user\_id = (SELECT user\_id FROM users WHERE login = current\_user));

CREATE POLICY services\_access ON services

USING (user\_id = (SELECT user\_id FROM users WHERE login = current\_user));

ALTER TABLE users ENABLE ROW LEVEL SECURITY;

ALTER TABLE credits ENABLE ROW LEVEL SECURITY;

ALTER TABLE services ENABLE ROW LEVEL SECURITY;

### Функции для взаимодействия базы данных и приложения

CREATE FUNCTION create\_user(user\_login TEXT, user\_passwd TEXT)

RETURNS VOID AS $$

BEGIN

EXECUTE FORMAT('CREATE ROLE "%I" LOGIN PASSWORD %L', user\_login, user\_passwd);

EXECUTE 'GRANT APP\_USER TO ' || user\_login;

EXECUTE 'GRANT USAGE, SELECT ON SEQUENCE services\_service\_id\_seq TO ' || quote\_ident(user\_login);

EXECUTE 'GRANT USAGE, SELECT ON SEQUENCE credits\_credit\_id\_seq TO ' || quote\_ident(user\_login)

END;

END;

$$ LANGUAGE plpgsql;

CREATE FUNCTION insert\_user(

user\_login TEXT,

user\_email TEXT,

user\_passwd TEXT,

user\_phone\_number TEXT

) RETURNS VOID AS $$

BEGIN

INSERT INTO users (login, email, passwd, phone\_number)

VALUES (user\_login, user\_email, user\_passwd, user\_phone\_number);

END;

$$ LANGUAGE plpgsql;

CREATE FUNCTION insert\_credit(

credit\_number text,

credit\_date text,

credit\_code text

) RETURNS VOID AS $$

DECLARE

user\_id\_var INTEGER;

BEGIN

SELECT user\_id INTO user\_id\_var FROM users WHERE login = current\_user;

IF user\_id\_var IS NOT NULL THEN

INSERT INTO credits (user\_id, credit\_number, credit\_date, credit\_code)

VALUES (user\_id\_var, credit\_number, credit\_date, credit\_code);

ELSE

RAISE EXCEPTION 'Текущий пользователь не существует в таблице users';

END IF;

END;

$$ LANGUAGE plpgsql;

CREATE FUNCTION insert\_service(

service\_name TEXT,

login\_service TEXT,

passwd\_service TEXT

) RETURNS VOID AS $$

DECLARE

user\_id\_var INTEGER;

BEGIN

SELECT user\_id INTO user\_id\_var FROM users WHERE login = current\_user;

IF user\_id\_var IS NOT NULL THEN

INSERT INTO services (user\_id, service\_name, login\_service, passwd\_service)

VALUES (user\_id\_var, service\_name, login\_service, passwd\_service);

ELSE

RAISE EXCEPTION 'Текущий пользователь не существует в таблице users';

END IF;

END;

$$ LANGUAGE plpgsql;