Министерство образования и науки Республики Башкортостан

Государственное автономное профессиональное образовательное учреждение

Уфимский колледж статистики, информатики и вычислительной техники

Тема проекта «»

Практическая работа №2 «Разработка тестовых сценариев для аутентификации»

МДК.01.02 Поддержка и тестирование программных модулей

|  |  |
| --- | --- |
|  | Преподаватель  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Л.Р. Набиева  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г. |
|  | Студент гр. 22П-3  А. А. Ахмадуллин  «22» октября 2024 г. |

2024

***Цель работы:***

Разработать функционал в модулях включающий авторизацию, регистрацию, личный кабинет. В приложении должны быть реализованны востановление пароля, двухфакторная аутентификация, работа с капчей.

***Описание работы:***

**1. коректность логина/пароля:**

При вводе корректной почты и пароля мы переходим в личный кабинет или в окно подтверждения почты в случае, если у пользователя подключена двухфакторая аутентификация:

private void AvtorizationButton\_Click(object sender, RoutedEventArgs e)

{

Person myPerson = Person.getPerson(mailTextBox.Text);

string answer = validation(myPerson);

if (answer != "")

{

MessageBox.Show(answer);

return;

}

nextWindow(myPerson);

}

private void nextWindow(Person myPerson)

{

if (myPerson.TFA == true)

{

Autorization\_TFW\_Window autorization\_TFW\_Window = new Autorization\_TFW\_Window(myPerson);

autorization\_TFW\_Window.Show();

this.Close();

return;

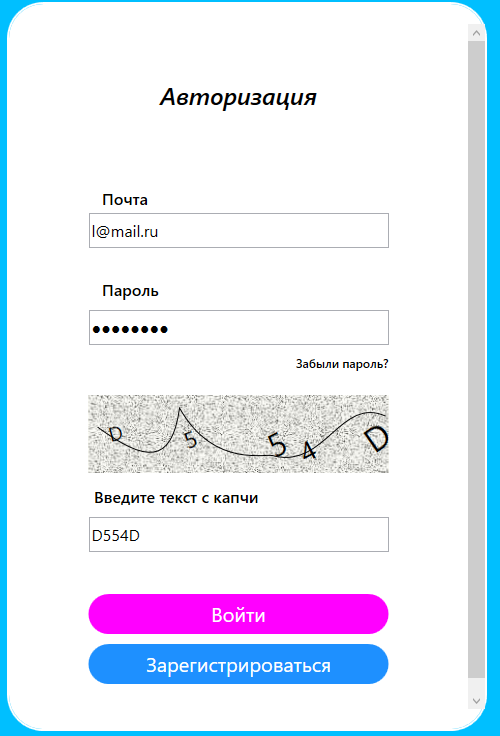
}

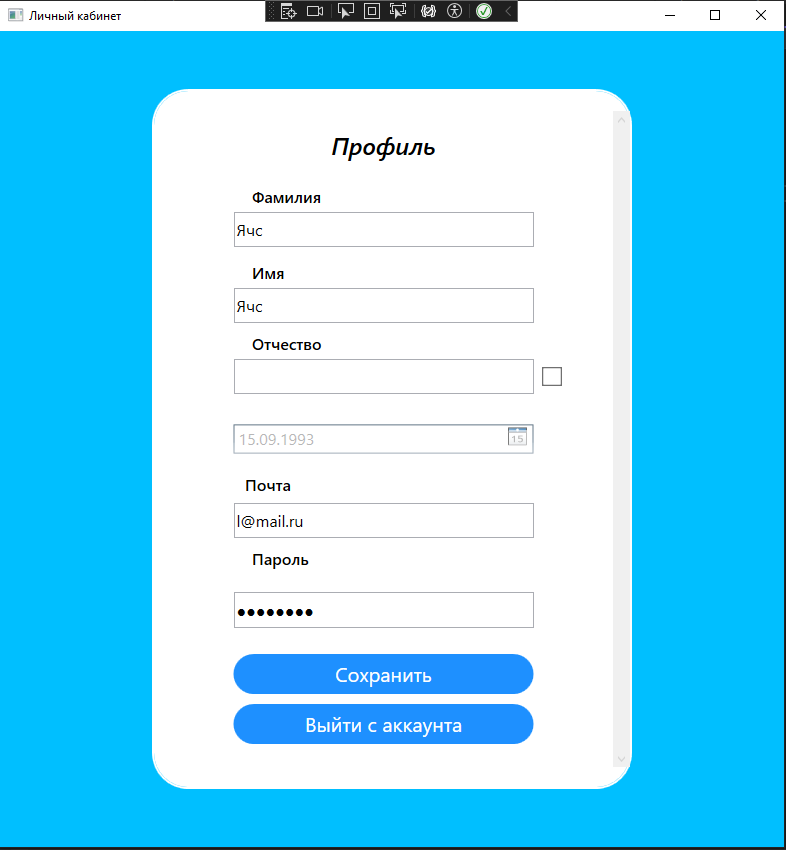
UserWindow userWindow = new UserWindow(myPerson);

userWindow.Show();

this.Close();

}





Функция validation проверяет введеные в поля данные и возвращает пустую строку в случае, если все данные верны, или выводит строку с сообщением о неверных действиях со стороны пользователя

private string validation(Person myPerson)

{

if (mailTextBox.Text.Length == 0 || passwordTextBox.Password.Length == 0)

{

return "Введите почту и пароль!";

}

if (!MailClass.correctMail(mailTextBox.Text) || passwordRegex.IsMatch(passwordTextBox.Password))

{

return "Неверный формат логина/пароля!";

}

if (captchaTextBox.Text == "")

{

return "введите капчу!";

}

if (captchaTextBox.Text != Captcha.CaptchaText)

{

Captcha.CreateCaptcha(Captcha.LetterOption.Alphanumeric, 5);

captchaTextBox.Text = "";

return "неверная капча!";

}

if (myPerson == null)

{

return "неверная почта или пароль";

}

if (LoginAttempt.lockedOrOpen(myPerson.BlockedUntil))

{

return "аккаунт заблокирован, попробуйте зайти через 30 секунд";

}

if (myPerson.Password != passwordTextBox.Password)

{

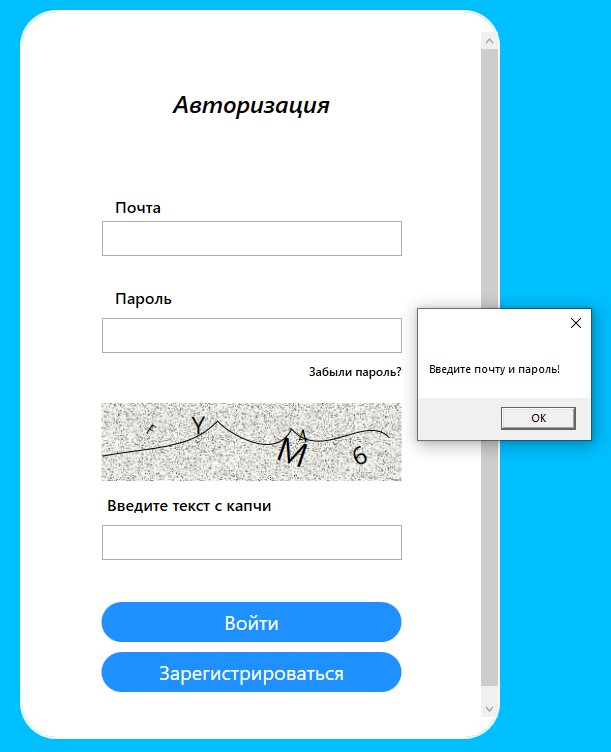
return UpdatefFiledAttempts(myPerson.Mail);

}

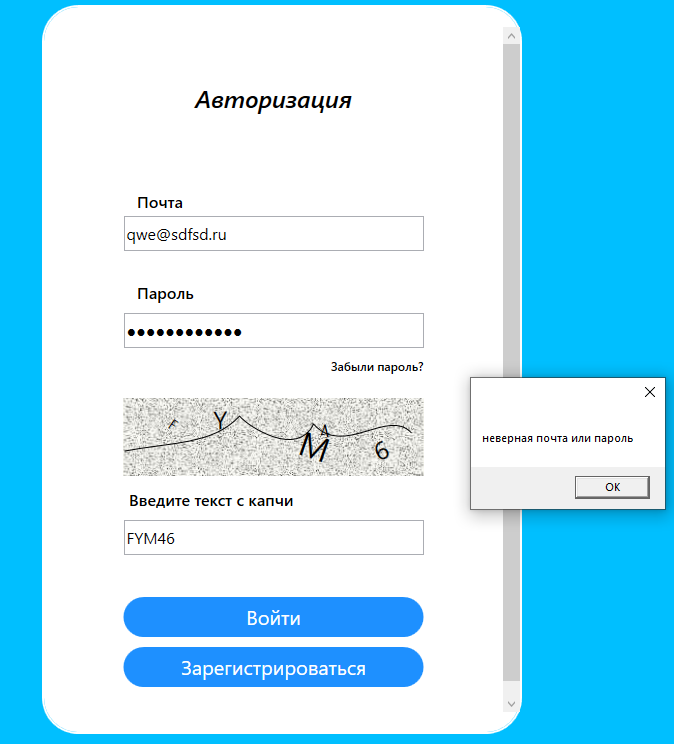
return "";

}

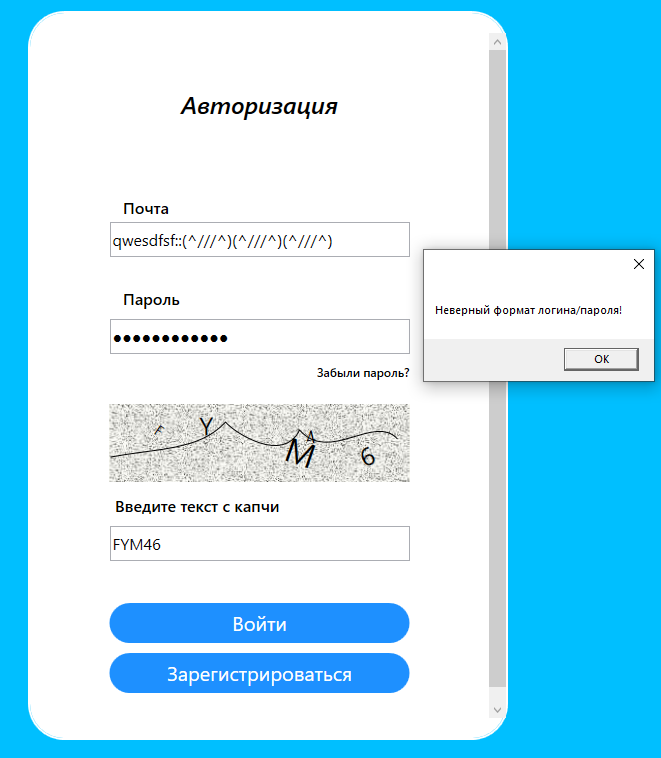
В случае неведенного логина или пароля выводиться сообщение “Введите почту и пароль”

****

В случае неверного ввода почты или пароля выводиться сообщение “неверная почта или пароль”



В случае ввода некорректной почты или пароля выводиться сообщение “Неверный формат логина пароля”



**2. блокировка аккаунта:**

имеется словарь, где ключ – это почта пользователя, а значение – это число неудачных попыток войти в систему с этим логином

private static Dictionary<string, int> failedAttempts = new Dictionary<string, int>();

в функции validation, при неправельно введенном пароле вызывается функция UpdateFiledAttempts, которая увеличивает количество неудачных попыток входа у пользователя. В случае, если кол-во неверных попыток достигло 5, то вызывается функция blockAccaunt, которая блокирует пользователя на 30 секунд:

static string UpdatefFiledAttempts(string personMail)

{

if (failedAttempts.ContainsKey(personMail))

{

failedAttempts[personMail]++;

if (failedAttempts[personMail] == 5)

{

failedAttempts.Remove(personMail);

LoginAttempt.blockAccaunt(personMail);

return "аккаунт был заблокирован!";

}

}

else

{

failedAttempts[personMail] = 1;

}

return "неверная почта или пароль";

}

public static void blockAccaunt(string personMail)

{

if (!File.Exists(Person.Path))

{

using (File.Create(Person.Path)) { }

return;

}

List<Person> persons = Person.deserialize();

Person person = persons.FirstOrDefault(x => x.Mail == personMail.ToLower());

if (person != null)

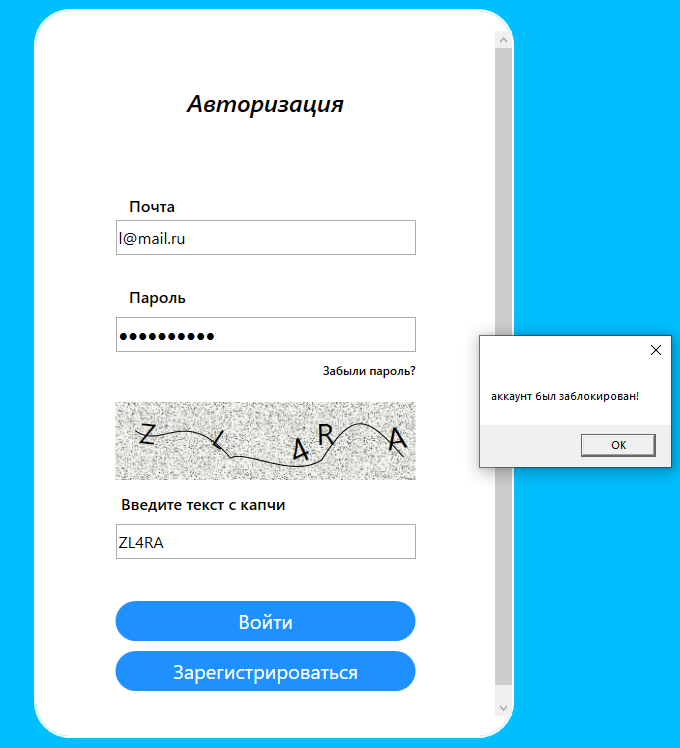
{

person.BlockedUntil = DateTime.Now.AddSeconds(30);

}

Person.SerializationListToJson(persons);

}

****

При попытки войти в аккаунт сразу после блокировки функция validation проверит заблокирован ли пользователь, и если да, выдаст соответсвующее сообщение:

if (LoginAttempt.lockedOrOpen(myPerson.BlockedUntil))

{

return "аккаунт заблокирован, попробуйте зайти через 30 секунд";

}

public static bool lockedOrOpen(DateTime blockedUntil)

{

if (blockedUntil == null || blockedUntil <= DateTime.Now)

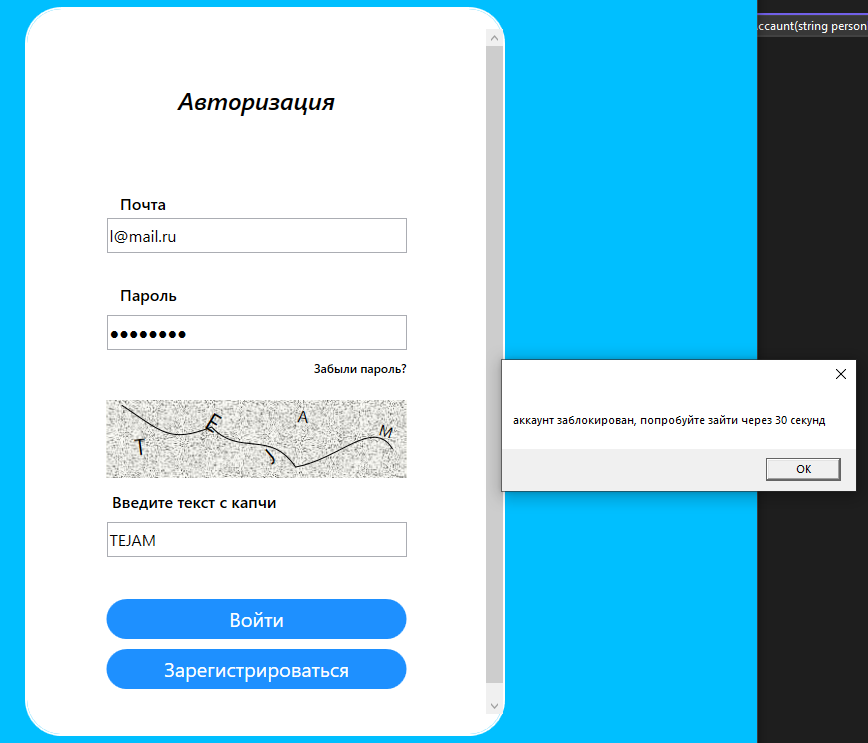
{

return false;

}

return true;

}

****

**3. востановление пароля:**

При отправке кода на почту сначало проверяется, корректная ли она, если да, то вызывает функцию sendCodeFunction

,private void sendCode\_Click(object sender, RoutedEventArgs e)

{

string isValid = validation();

if (isValid != "")

{

MessageBox.Show(isValid);

return;

}

sendCodeFunction();

}

private string validation()

{

if (mailTextBox.Text == "")

{

return "Введите почту!";

}

if (!MailClass.correctMail(mailTextBox.Text))

{

return "некорректный формат почты!";

}

if (!Person.personExists(mailTextBox.Text))

{

return "пользователя с такой почтой ненайден";

}

if (NoSpamCode.lockedOrOpen(mailTextBox.Text))

{

return "вы уже превысили максимальное количество попыток подтверждения\nПопробуйте через 30 секунд";

}

return "";

}

Функция sendCodeFunction записывает время, когда был отправлен код, изменяет состояние кнопок и вызывает функцию sendMessege, которая и отправляет код:

private void sendCodeFunction()

{

personMail = mailTextBox.Text;

NoSpamCode.blockSpamer(personMail, @"..\..\timeIsBakedJosn.json");

sendCodeButton.Content = "Код выслан";

sendCodeButton.Background = new SolidColorBrush(Colors.LightGray);

sendCodeButton.Foreground = new SolidColorBrush(Colors.Black);

sendCodeButton.IsEnabled = false;

confirmButton.Background = new SolidColorBrush(Colors.DodgerBlue);

confirmButton.Foreground = new SolidColorBrush(Colors.White);

confirmButton.IsEnabled = true;

code = new Random().Next(100000, 999999);

MailClass.sendMessege(personMail, code);

}

public static string sendMessege(string personMail, int code)

{

try

{

using (var smtp = new SmtpClient())

{

smtp.Connect("smtp.yandex.ru", 465, true);

smtp.Authenticate("testsharp1@yandex.ru", "acgazcvtbqynxndu");

var bodyBldr = new BodyBuilder();

bodyBldr.TextBody = $"{code}";

var msg = new MimeMessage()

{

Subject = "Код подтверждения",

Body = bodyBldr.ToMessageBody(),

};

msg.To.Add(MailboxAddress.Parse(personMail));

msg.From.Add(new MailboxAddress("SGH\_Corparation", "testsharp1@yandex.ru"));

smtp.Send(msg);

}

}

catch

{

return "Ошибка при отправке письма\nПроверте подключение к интернету";

}

return "";

При подтверждении введенного пользователем кода, вызывается функция validationCode, которая проверяет верный ли код и непросрочен ли он, если да, то выводиться соответсвующее сообщение, в случае корректного кода открывается окно востановление пароля:

private void confirmButton\_Click(object sender, RoutedEventArgs e)

{

string currentCode = validationCode();

if (currentCode != "")

{

MessageBox.Show(currentCode);

return;

}

MessageBox.Show("Аккаунт подтвержден");

toGoNewPasswordWindwow();

}

private string validationCode()

{

if (!NoSpamCode.lockedOrOpen(personMail, @"..\..\timeIsBakedJosn.json"))

{

sendCodeButton.Content = "Выслать код";

sendCodeButton.Background = new SolidColorBrush(Colors.DodgerBlue);

sendCodeButton.Foreground = new SolidColorBrush(Colors.White);

sendCodeButton.IsEnabled = true;

confirmButton.Background = new SolidColorBrush(Colors.LightGray);

confirmButton.Foreground = new SolidColorBrush(Colors.Black);

confirmButton.IsEnabled = false;

return "Код восстановления просрочен";

}

if (confirmationCodeTextBox.Text != code.ToString())

{

if (counfirmButtonClickCount == 4)

{

LoginAttempt.blockAccaunt(personMail);

NoSpamCode.blockSpamer(personMail);

counfirmButtonClickCount = 0;

return "Обнаружена попытка взлома, аккаунт заблокирован на 30 секунд!";

}

counfirmButtonClickCount++;

return "Неверный код подтверждения!";

}

return "";}

private void toGoNewPasswordWindwow()

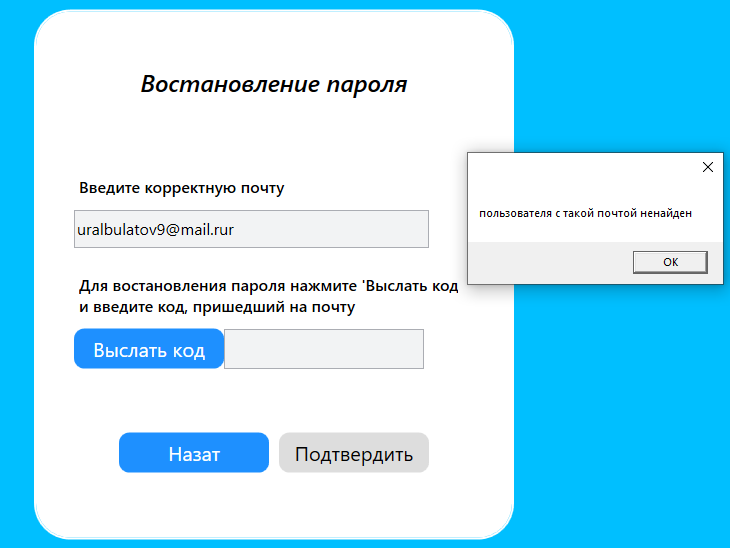
{

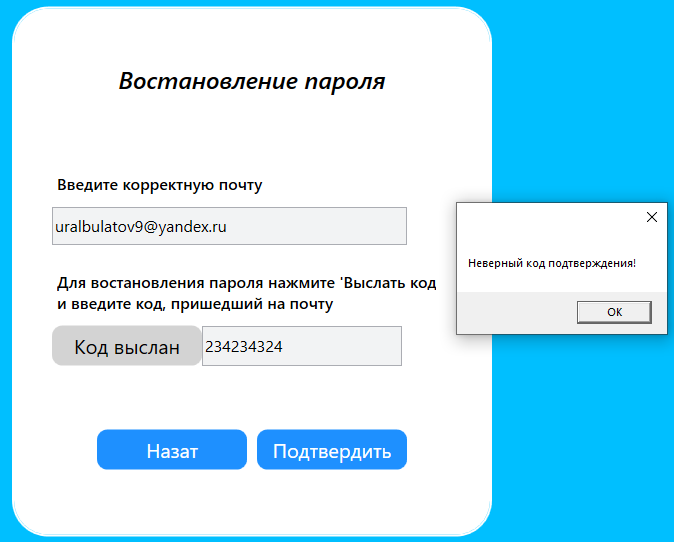
NewPasswordWindow newPasswordWindow = new NewPasswordWindow(personMail);

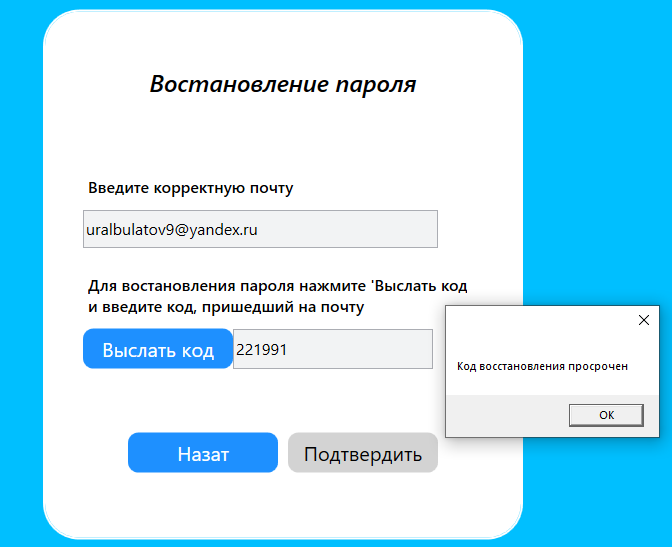
newPasswordWindow.Show();

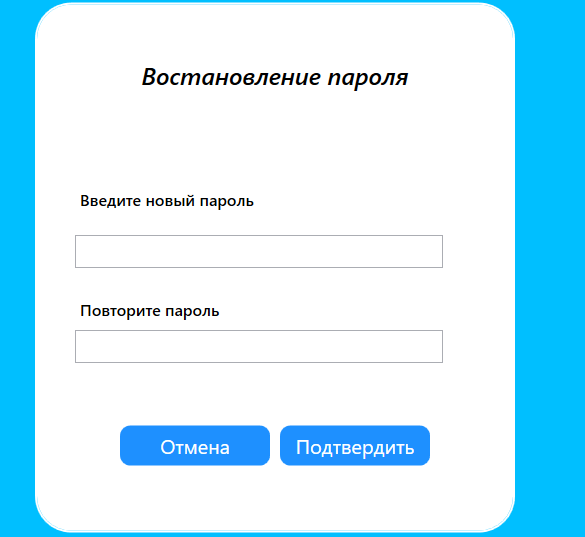
this.Close();

}









При нажатии на кнопку подтверждения проверяется валидация пароля, если валидация проходит, то вызывается функция newPassword, которая обновляет пароль и возваращает на главное окно:

private void confirmButton\_Click(object sender, RoutedEventArgs e)

{

if (!validation())

return;

newPassword();

}

private bool validation()

{

if (passwordPasswordBox.Password == "" || repeatePasswordBox.Password == "")

{

MessageBox.Show("Заполните все поля!");

return false;

}

if (passwordPasswordBox.Password.Length < 8)

{

MessageBox.Show("длинна пароля должна быть не меньше 8 символов!");

return false;

}

if (repeatePasswordBox.Password != passwordPasswordBox.Password)

{

MessageBox.Show("пароли не совпали!");

return false;

}

return true;

}

private void newPassword()

{

List<Person> persons = Person.deserialize();

Person person = persons.FirstOrDefault(x => x.Mail == personMail);

if (person != null)

{

person.Password = passwordPasswordBox.Password;

Person.SerializationListToJson(persons);

}

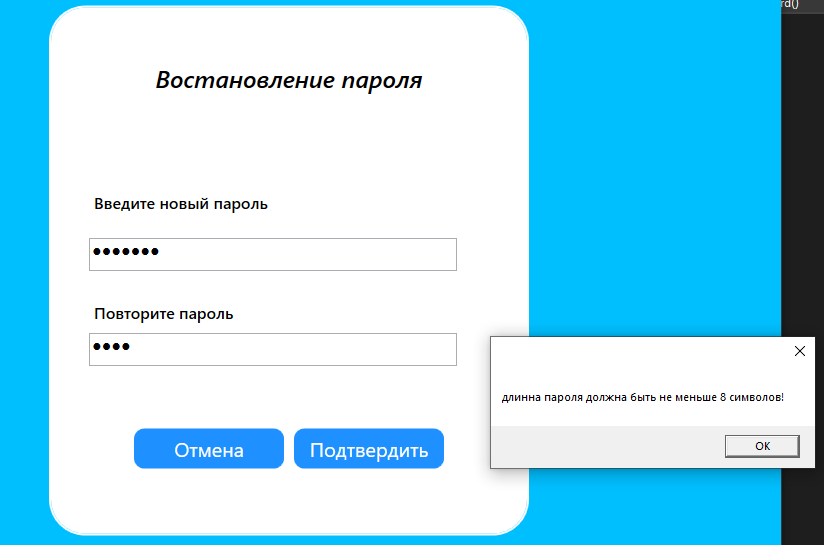
MessageBox.Show("Пароль успешно обнавлен!");

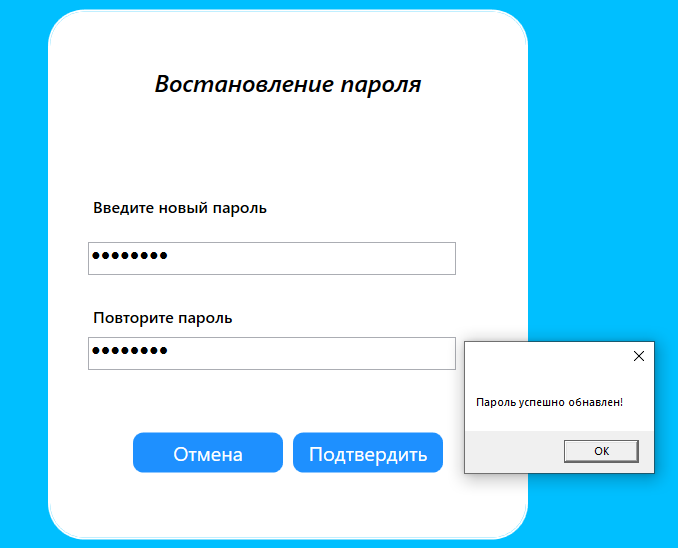
MainWindow autorizationWindow = new MainWindow();

autorizationWindow.Show();

this.Close();

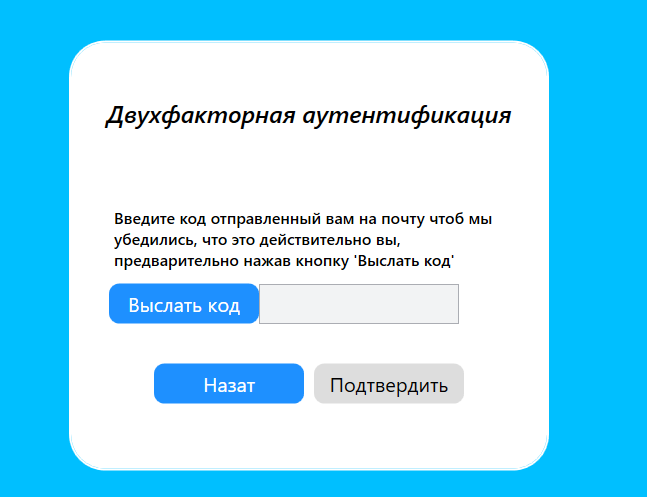
}





**4. двухфакторная аутентификация:**

При вводе корректной почты и пароля, при наличии у пользователя двухфакторной аутентификации может открыться соответсвующее окно:



При нажатии на кнопку “Выслать код” вызывается функция sendCodeFunction, которая изменяет состояние кнопок и высылает код на почту пользователя

private void sendCodeFunction()

{

sendCodeButton.Content = "Код выслан";

sendCodeButton.Background = new SolidColorBrush(Colors.LightGray);

sendCodeButton.Foreground = new SolidColorBrush(Colors.Black);

sendCodeButton.IsEnabled = false;

confirmButton.Background = new SolidColorBrush(Colors.DodgerBlue);

confirmButton.Foreground = new SolidColorBrush(Colors.White);

confirmButton.IsEnabled = true;

code = new Random().Next(100000, 999999);

string sendCode = MailClass.sendMessege(currentPerson.Mail, code);

if (sendCode != "")

MessageBox.Show(sendCode);

}

При нажатии на кнопку “Подтвердить” мы проверяем правельность кода и если он верный, то открываем профиль

private void confirmFunction()

{

if (confirmationCodeTextBox.Text != code.ToString())

{

if (counfirmButtonClickCount == 4)

{

LoginAttempt.blockAccaunt(currentPerson.Mail);

MessageBox.Show("аккаунт заблокирован на 30 секунд!");

counfirmButtonClickCount = 0;

return;

}

MessageBox.Show("Неверный код подтверждения!");

counfirmButtonClickCount++;

return;

}

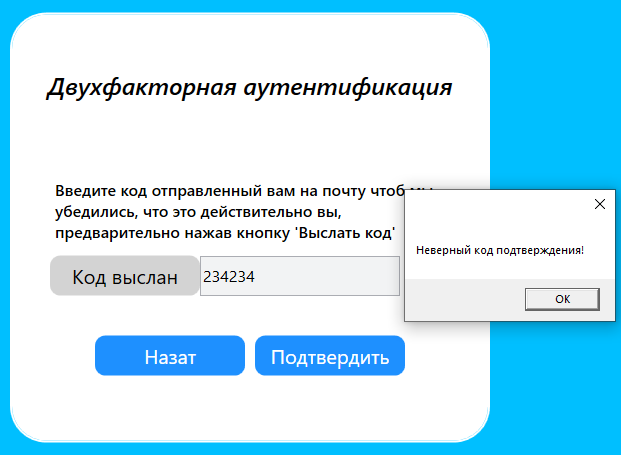
MessageBox.Show("Аккаунт подтвержден");

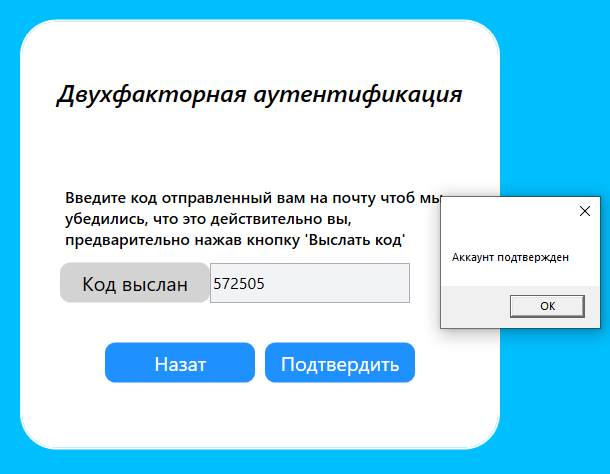
UserWindow userWindow = new UserWindow(currentPerson);

userWindow.Show();

this.Close();

}





**5. работа с капчей:**

При проверке капчи в функции validation вызывается метод checkCaptcha, который проверят правельно ли пользователь ввел капчу, уведомляя его об ошибках. Так же при неверно введенной капче функция генерирует новую. В случае если капча и другая валидация верна мы поподаем либо в профиль, либо в окно двухфакторной аутентификации:

private string checkCaptcha()

{

if (captchaTextBox.Text == "")

{

return "введите капчу!";

}

if (captchaTextBox.Text != Captcha.CaptchaText)

{

Captcha.CreateCaptcha(Captcha.LetterOption.Alphanumeric, 5);

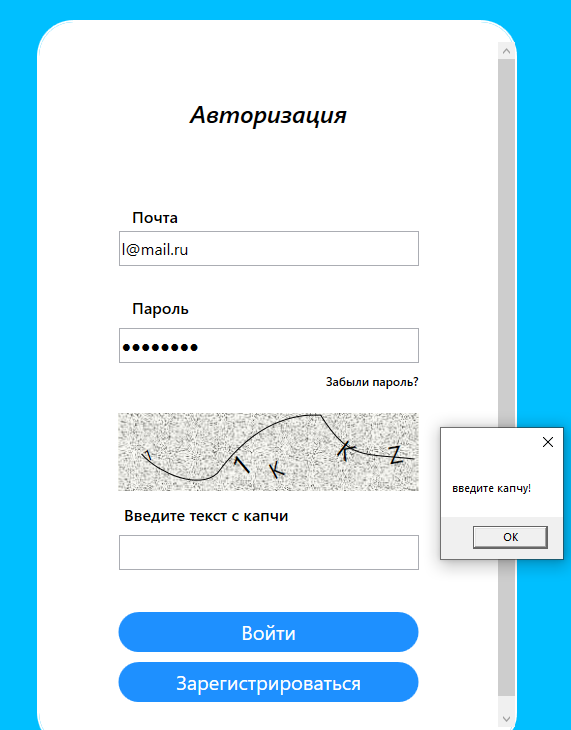
captchaTextBox.Text = "";

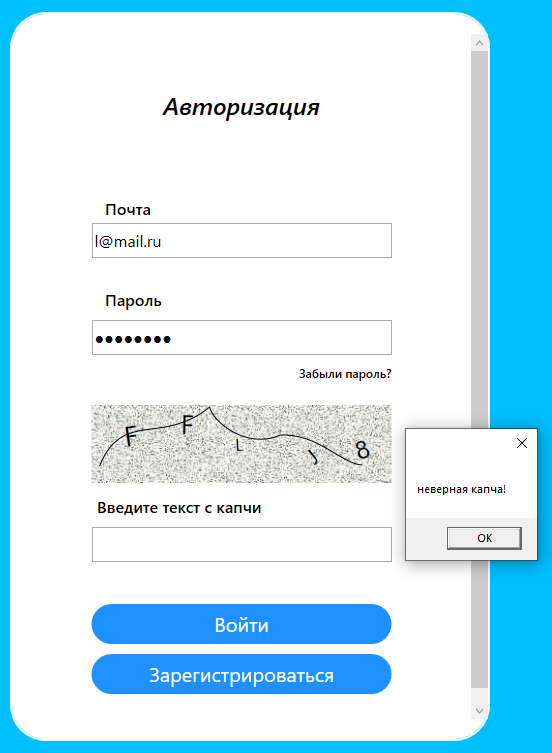
return "неверная капча!";

}

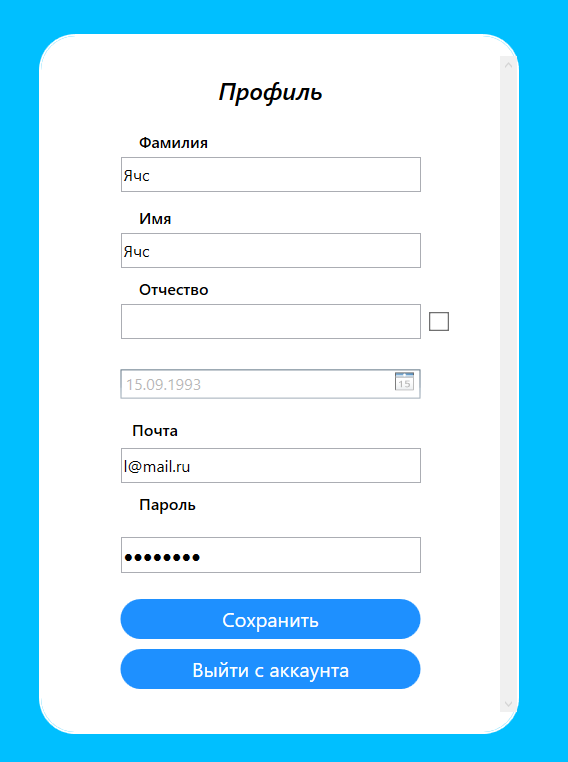
return "";

}





Примечание: поле ввода не было пустым, при вводе неверной капчи, функция его очищает.



***Вывод:***

Научился создавать приложения, которые поддерживают работу с капчей и с электронными почтовыми сервисами