**UNIVERSIDADE PAULISTA**

GUILHERME BUENO DA SILVA - D3189H-4

VINICIUS DE OLIVEIRA – D17315-4

EMERSON FRANSON BARROS – D44824-2

FREDERICO MAZZER MENDES JUNIOR - D32CIB-2

**PROCESSAMENTO BIOMÉTRICO**

PROCESSAMENTO DE IMAGENS

**SOROCABA**

**2019**

Sumário

[1. OBJETIVO DO TRABALHO 1](#_Toc19536009)

[2. INTRODUÇÃO 2](#_Toc19536010)

[3. FUNDAMENTOS DAS PRINCIPAIS TÉCNICAS BIOMÉTRICAS 4](#_Toc19536011)

[3.1 TIPOS DE RECONHECIMENTO 4](#_Toc19536012)

4. PLANO DE DESENVOLVIMENTO DA APLICAÇÃO...........................................8

4.1 LINGUAGEM C#.........................................................................................

4.2 BANCO DE DADOS SQL SERVER

4.3 FERRAMENTAS E RECURSOS

4.3.1. FILTRO PASSA-ALTA

4.3.2. TRATAMENTO DE IMAGEM

5. PROJETO DO PROGRAMA

6. RELATÓRIO DO PROGRAMA

# OBJETIVO DO TRABALHO

# 

O trabalho tem como objetivo aprofundar os conhecimentos na matéria de Processamento de Imagens, tendo como ponto de partida a criação de uma sistema de reconhecimento de biometria para identificação e autenticação. O grupo visa entender como funciona um sistema de biometria e o processamento que deve ser feito para que não ocorra falhas no ato de identificação e autenticação de cada usuário para cada informação correspondente.

Outro objetivo é estimular o trabalho em grupo, visando aplicar o que foi ensinado em sala de aula, de modo prático. A cooperação do grupo é primordial para o desenvolvimento do projeto, em todas as etapas.

Com a constante evolução da tecnologia, podemos observar que cada vez mais sistemas estão ficando automatizados e com autenticações biométricas, e é importante que profissionais de tecnologia saibam os princípios e fundamentos da biometria para que possam ficar familiarizados com este assunto.

A motivação para realizar este trabalho é a vontade de aprender coisas novas, o interesse em descobrir novas ferramentas tecnológicas que ajudam e facilitam o nosso cotidiano, entre outras coisas. Cada parte do processo é importante para o aprendizado, visto que é necessário uma compreensão geral do assunto e um esforço para realizar as tarefas propostas.

# INTRODUÇÃO

Os sistemas com biometria já estão sendo usados por muitas empresas para o controle de entrada e saída de funcionários ou até mesmo em domicílios e para outros fins como (condomínios, habilitação, relógio de ponto, para controlar o sigilo de informações etc). Hoje é mais comum existir esse tipo de identificação e autenticação pela facilidade de se identificar nos determinados locais, por exemplo, ao entrar em um condomínio que há instalado um sistema de identificação por biometria não será preciso lembrar de senhas ou ter cartões de acesso, mas sim simplesmente fazer a autenticação em um leitor biométrico que através de um sistema, faz cálculos para saber quem é você e o que você pode ter acesso.

O sistema será feito com a linguagem de programação C# e banco de dados SQL. O ponto de partida será criar um sistema de identificação para cada funcionário identificando assim cada qual funcionário poderá ter acesso a uma determinada informação. Vamos simular o cadastramento e a consulta do funcionário por meio da biometria cadastrada para garantir que cada funcionário só terá acesso a informação pelo seu nível ou cargo em uma determinada empresa.

Todos nós queremos a facilidade no momento da identificação, para auxiliar utilizaremos o sistema de biometria, para facilitar a identificação de pessoas nos lugares, identificar informações sigilosas que só algumas pessoas podem ter acesso.

O processamento de imagem é importante pois a imagem captada por um sensor ou um anexo deve ser tratada antes de poder ser analisada em uma consulta para identificar quem é a pessoa e o que ela pode ter acesso e onde pode ter acesso, assim que o sensor capita a impressão digital ela passará por vários processos para que seja possível analisar a impressão digital colocada no sensor, no ato da identificação ou autenticação e assim podendo ser feita uma análise do mesmo com as biometrias cadastradas no banco de dados.

O estudo de técnicas biométricas visa facilitar o desenvolvimento de sistemas que usam esse tipo de controle para restringir acessos e informações, assim diminuindo custos para fabricação de cartões de acesso, papéis de assinatura de presença, custos administrativos e a facilidade e rapidez de controle de acesso ou informação.

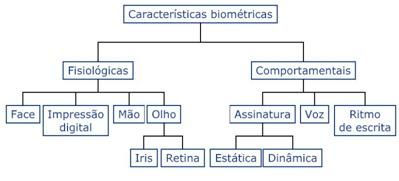
Os primeiros relatos da biometria como meio de identificação estão associados à pré-história, mais especificamente na Babilônia. Na China, há relatos de artesãos que imprimiam suas digitais em vasos como forma de associar suas identidades às obras, como uma assinatura única.

No entanto, antes mesmo de confirmarem o uso real com evidências científicas quanto à legitimidade desse meio de validação, o ato de imprimir a marca da digital já era utilizado como forma de comprometimento. Na colonização inglesa da India, Sir William Herschel, radicado no distrito de Hooghly, fazia com que seus parceiros de negócios imprimissem a digital nos contratos, além de firmá-los. Ele acreditava que tal ato “assustaria todo e qualquer pensamento de repúdio à assinatura”.

Tudo indica que, em um futuro próximo, a tecnologia de identificação de íris, retina e afins, se tornará cada vez mais comum e acessível, podendo ser implementada em celulares desde os modelos mais simples, até os mais avançados. Visto que é uma tecnologia muito segura, poderá ser usada em simples controladores de acesso em residências de baixo custo.

Então chegamos a conclusão que o processo de identificação por biometria, seja qual for a forma. Não é algo novo, surgindo na pré-história ainda possui muitos traços daquilo que podemos chamar de “base”, a finalidade ainda é distinguir e acima de tudo, ser seguro.

# **FUNDAMENTOS DAS PRINCIPAIS TÉCNICAS BIOMÉTRICAS**

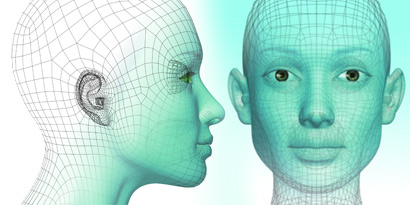
A biometria é uma das tecnologias disponíveis no mercado, quando o assunto é segurança, e ela vem se tornando cada vez mais popular entre as empresas que buscam por um sistema de autenticação. A biometria possui informações particulares, de um ser humano, tais como impressão digital, retina, íris, e a geometria da mão.

Em modo geral qualquer característica humana (Física ou comportamental) pode ser usada para identificação de pessoas, desde que obedeça as seguintes regras.

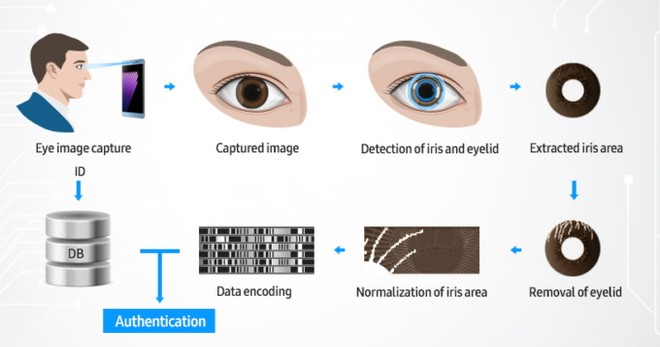
* Universalidade – Significa que todas as pessoas devem possuir a característica;
* Singularidade – Indica que esta característica não pode ser igual em pessoas diferentes;
* Permanência – Significa que a característica não deve variar com o tempo;
* Mensurabilidade – Indica que a característica pode ser medida quantitativamente.

# **TIPOS DE RECONHECIMENTO**

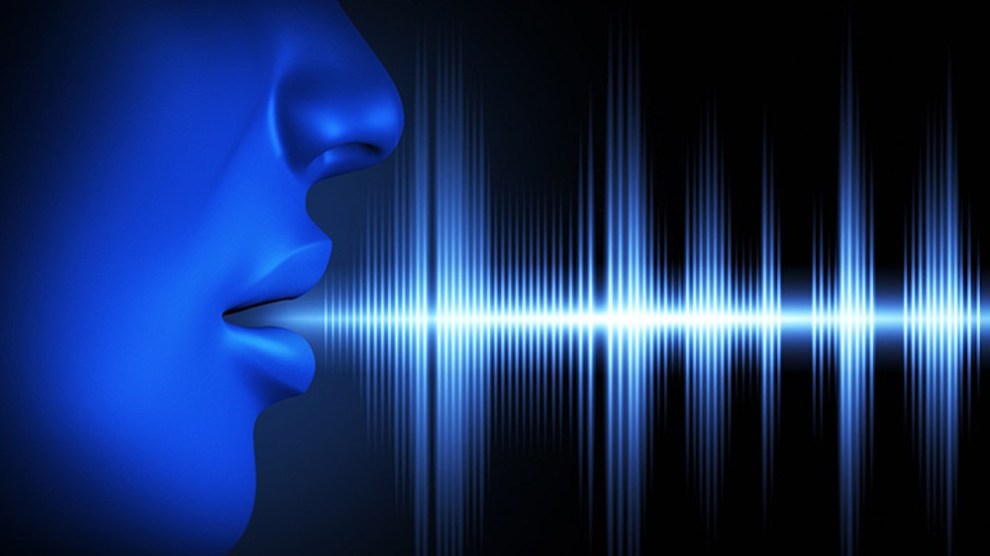
**Reconhecimento da Face** - Esse tipo de reconhecimento permite, através de uma imagem de um rosto humano, identificar vários pontos e traços da face. Sendo possível fazer uma descrição em detalhes deste rosto, coletando informações de tamanho, distância e formas, comparando a imagem com o que há registrado no banco de dados. Um dos fatores de risco, ou falha, deste tipo de autenticação, está no fato de que ao passar dos anos, o rosto sofre grandes alterações, como por exemplo a barba, cabelo, óculos de sol, entre outros itens, podendo prejudicar o reconhecimento, tornando o processo muito mais lento ou até mesmo não ser possível reconhecer.



* **Geometria da mão** - Tem como base o princípio de que não existem no mundo duas pessoas com mãos idênticas, e que com o passar dos anos, a mão não sofrerá grandes mudanças. Através de imagens registradas, serão capturados pontos específicos da mão que serão utilizados pelo sistema para distinguir se é ou não permitido o acesso.
* **Identificação da íris** - Utilizando um feixe de luz, é capturada uma imagem muito complexa da íris humana e posteriormente enviado a um banco de dados. Este tipo de identificação é um dos autenticadores mais seguros e confiáveis, devido à unicidade da íris de cada pessoa.



* **Identificação de retina** - Notamos neste tipo de identificação, muita semelhança com a anterior, íris. A principal diferença está em que serão extraídos da imagem apenas os vasos sanguíneos nos olhos, e não da íris. Os dispositivos que fazem a analise, gerenciam os padrões de vasos sanguíneos, fazendo uso de um laser de pouca intensidade e uma câmera de alta definição.

**Reconhecimento de voz** - Este método de autenticação é muito fácil de utilizar e é considerada não intrusiva, pois a voz de uma pessoa não é um atributo sigiloso, porém, não é uma técnica 100% confiável, visto que uma voz pode ser imitada ou falseada. O programa responsável pela identificação faz uma análise dos padrões harmônicos presentes na voz, e não apenas uma comparação entre diferentes reproduções de uma mesma fala.

**Reconhecimento da assinatura manuscrita** - O sistema se baseia no que é capturado durante a assinatura manuscrita do utilizador. São considerados dados diversos pontos, tais como a pressão exercida sobre a caneta, o tempo de escrita, movimentos da mão durante o processo e alguns pontos em que a caneta não se encontra na assinatura.

**Impressão digital** - A particularidade das impressões digitais é reconhecida em todo mundo. é utilizada desde o século XIX. Possivelmente a tecnologia mais desenvolvida quando o assunto é biometria, sendo necessário, apenas um dispositivo de alta precisão que possa identificar os mínimos detalhes de uma impressão digital. Já o sistema fica responsável por transformar esta simples imagem em informação útil, que será utilizado em algum momento para o reconhecimento digital.



1. PLANO DE DESENVOLVIMENTO DA APLICAÇÃO

Neste tópico iremos citar os elementos e ferramentas que utilizamos tais como linguagem de programação, sistema de gerenciamento de banco de dados, filtros e recursos, demonstrando em curtas palavras o que foi utilizado e para que foi. Serão exibidos fórmulas que estão presentes no algoritmo, comandos, entre outras finalidades que envolvem a aplicação.

4.1. LINGUAGEM C#

O trabalho consiste em uma aplicação C#, que possui tela principal, tela de cadastro e pesquisa, aonde o usuário poderá utilizar todos os recursos do programa, selecionando a tela que deseja utilizar. Trata-se de um programa responsável por identificar diferentes níveis de acesso, podendo assim, distinguir níveis de acesso de um determinado grupo de pessoas.

No módulo de cadastro, é possível cadastrar novos funcionários. Durante este processo, é importante todas as informações que são relacionadas ao usuário, portanto, sendo todos os campos obrigatórios.

No campo “Nome”, o usuário deve inserir seu nome completo, respeitando o limite de caracteres.

No campo “CPF”, deve ser inserido o CPF com 11 dígitos, sem a necessidade de caracteres especiais, pois o campo já possui uma formatação.

No campo “E-mail”, deve ser inserido um e-mail válido, desta forma: [nomedousuario@provedor.com](mailto:nomedousuario@provedor.com).

No campo “Tipo de Acesso”, podemos definir o nível de acesso do usuário, trata-se de um controle ComboBox, que utiliza propriedade DropDownStyle.

É necessário selecionar um dos 3 níveis de acesso, sendo que no nível 1 não é possível acessar informações de nível 2, que podem ser acessados apenas por Diretores, ou no nível 3 apenas pelo ministro do ambiente.

Para concluir o cadastro, o usuário deve inserir uma imagem com a sua impressão digital, que será utilizado, como método de biometria, esta imagem irá passar por um filtro de tratamento, aonde serão selecionados pontos específicos, para diferenciar uma digital de outra.

Por último, mas não menos importante, temos o botão “Salvar”, que irá salvar todas as informações inseridas em uma nova linha de tabela, no banco de dados que utilizamos, SQL. Tais informações poderão ser pesquisadas logo após serem salvas.

4.2. BANCO DE DADOS SQL SERVER

O SQL Server é um banco de dados comumente utilizado, pela sua eficiência e praticidade, e é por esse motivo que decidimos implementá-lo em nosso projeto.

Para armazenar as imagens das impressões digitais, e os dados inseridos pelo usuário, fizemos o uso deste banco de dados. Na tela de cadastro, após ser inserido os dados da pessoa e a imagem da impressão digital, é possível salvar os dados, clicando no botão “Salvar”, e, então, esses dados serão enviados para o banco.

No banco estes dados serão salvos na tabela “Acesso” que possui a informação de qual nível de acesso este usuário possui, esta por vez tem uma chave primaria “CPF”, presente também na tabela “Pessoa” do banco, com a finalidade de identificar o usuário.

Após identificação, os dados são retirados do SQL e exibidos na pagina ao usuário, tudo isso é possível graças aos recursos de comunicação do C# com o SQL Server, que permitem utilizar os dados, sendo assim possível, fazer pesquisas e consultar informações em um determinado banco.

1. 4.3. FERRAMENTAS E RECURSOS

Neste tópico mostraremos ferramentas e recursos utilizados para tratamento de imagem e decodificação.

4.3.1. FILTRO PASSA-ALTA

Em nosso trabalho, utilizamos o filtro passa-alta, para realçar as bordas das impressões digitais, para que estas sejam reconhecidas pelo sistema de biometria. Para esclarecer a parte teórica, iremos mostrar os conceitos deste filtro, como ele realiza suas transformações, através de cálculos.

O filtro passa-alta faz uso do circuito RC-Série, ou seja, possui resistores e capacitores em série, em que a tensão de saída é obtida no resistor. Como podemos ver na figura 1.

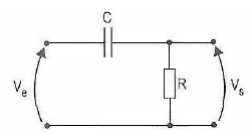


Figura 1

No entanto, para as ondas senoidais, com frequências altas, a reatância capacitiva passara a possuir valores baixos comparado a resistência. Sendo assim, a tensão de saída será semelhante à tensão de entrada. Já com frequências baixas, a mesma reatância capacitiva passa a possuir desta vez, valores altos comparado ao valor da resistência, estreitando a tensão de saída para um valor quase nulo. Então, este filtro permitirá apenas a passagem de altas frequências, e por esta característica é denominado por filtro passa-alta.

Para utilizar este filtro, será necessário a realização das seguintes fórmulas:

𝑉𝑒 = √𝑉𝑅 ² + 𝑉𝐶 ²

Onde:

VR = VC = Vs → 𝑉𝑒 = √𝑉𝑆 ² + 𝑉𝑆 ²

𝑉𝑒 = √2𝑉𝑠²

Ve = Vs√2 ou Vs= Ve √2

Essa frequência, como observado acima, é denominada frequência de corte (fC) e esta pode ser calculada igualando o valor da reatância com o valor da resistência.

XC = R ou \_\_1\_\_ = R → fC = \_\_1\_\_\_

2πfCC 2πRC

A característica da tensão de saída em função da frequência de um filtro passa-alta pode ser vista na figura a seguir.

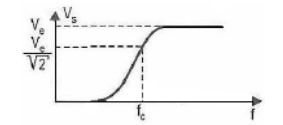


Figura 5

4.3.2. TRATAMENTO DE IMAGEM

Foi utilizado o recurso de conversão para bytes, na imagem, através do algoritmo implementado no C#, para transformar a impressão digital salva como imagem, em binário. Este, por sua vez, será salvo em uma das colunas da tabela (Pessoas) do banco de dados SQL Server, e será utilizado como forma de identificação e criptografia do usuário.

O algoritmo é capaz de realizar diversas conversões de imagem até que a mesma se transforme em informação, para que o sistema possa encontrar este dado futuramente. E este, por sua vez, fará referência a um usuário específico, com precisão.

A informação que está sendo tratada, passando por este processo de conversão, resultando em binários. Já a imagem sofrerá grandes mudanças, como é possível notar comparando a figura 6 antes do filtro, e figura 7 após o filtro, e no final do processo, possuirá apenas as informações relevantes para a identificação.



Figura 6 Figura 7

1. PROJETO/ESTRUTURA DO PROGRAMA

Nesta seção mostraremos os detalhes do projeto, ilustrado com imagens do programa em funcionamento, para facilitar a compreensão.



Figura 8: Tela principal da aplicação

Na tela principal, podemos ver as opções existentes, e é esperado que o usuário selecione uma delas. Ao fundo há um papel de parede referente ao tema do trabalho, para fins ilustrativos.

No momento em que o usuário seleciona “Cadastrar” ou “Pesquisar”, novas opções aparecerão na tela, e devem ser preenchidos com os dados requeridos para concluir a operação.



Figura 9: Tela de cadastro da aplicação

Quando é clicado em “Cadastrar”, novas caixas de texto aparecerão, solicitando dados pessoais e a imagem da impressão digital, como pode ser observado na figura 9.

Após preencher todos os campos, com os dados necessários, além da imagem da impressão digital, como o exemplo da figura 10, é possível clicar em “Salvar” para que o programa salve os dados no banco de dados.



Figura 10: Tela de cadastro preenchida

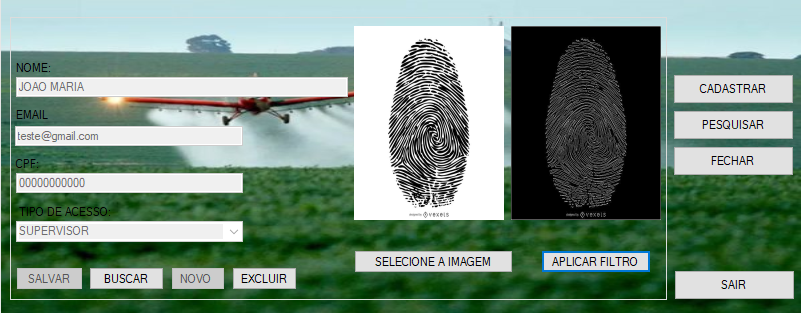


Figura 11: Tela de pesquisa da aplicação

A aplicação possui um módulo de pesquisa que é possível acessá-lo pelo botão “Pesquisar”, que se encontra no menu posicionado à direita da aplicação.

Nesta janela, como é possível observar na figura 11, os campos não estão disponíveis para alteração, sendo possível apenas fornecer a impressão digital, selecionando a imagem, sendo assim, liberada a opção de aplicar o filtro passa-alta através do botão “Aplicar Filtro”.

Após a aplicação do filtro passa-alta, que tem como função realçar as bordas da imagem, e retirar todas as baixas frequências que a imagem possui, é possível observar o resultado pós-filtro.

O programa foi criado através do Visual Basic, na linguagem de programação C#, podendo ser executado em qualquer versão do Windows, por um arquivo .exe, junto às pastas raízes do programa.

A aplicação possui tratamento de erros, em casos específicos, onde o usuário é notificado que um campo foi preenchido incorretamente, ou não foi preenchido. Então, se o usuário não preencher os campos, será exibida a seguinte mensagem: “Preencha todas as informações” e “Campos em branco”, indicando que o campo deve ser preenchido obrigatoriamente, para que sejam salvos seus dados.

Para finalizar este tópico, podemos observar que não é um programa complexo, pois realiza funções relativamente simples, não necessitando de uma programação extensa. Porém, é um projeto bem organizado, com cada parte funcionando corretamente, que possui apenas recursos funcionais, que são utilizados ao longo do programa. A interface simples, de fácil entendimento pelo usuário, e o código compacto, tornam este programa acessível para todo e qualquer dispositivo, exigindo o mínimo de hardware, pois, a única finalidade da aplicação será consultas e cadastros em um banco de dados externo, podendo ser comparado a um sistema ERP.

1. RELATÓRIO DO PROGRAMA

\*\*\*FORMULARIO PRINCIPAL\*\*\*

using System;

using System.Windows.Forms;

using System.Drawing;

using System.Data.Entity;

using System.ComponentModel;

using PROCESSAMENTO\_IMAGEM.MODELS;

using System.IO;

using PROCESSAMENTO\_IMAGEM.BLL;

using System.Linq;

namespace PROCESSAMENTO\_IMAGEM.GUI

{

public partial class frmPrincipal : Form

{

public Contexto db = new Contexto();

static string nomeArquivo;

public frmPrincipal()

{

InitializeComponent();

}

protected override void OnLoad(EventArgs e)

{

base.OnLoad(e);

db.Pessoas.Load();

db.Acessos.Load();

}

protected override void OnClosing(CancelEventArgs e)

{

base.OnClosing(e);

this.db.Dispose();

}

private void frmPrincipal\_Btn\_Exit\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void frmPrincipal\_Btn\_Register\_Click(object sender, EventArgs e)

{

frmPrincipal\_Gbx\_Data.Visible = true;

frmPrincipal\_Btn\_Procurar.Enabled = false;

frmPrincipal\_Btn\_Delete.Enabled = false;

frmPrincipal\_btn\_New.Enabled = true;

frmPrincipal\_Btn\_Save.Enabled = true;

frmPrincipal\_Txt\_Name.Enabled = true;

frmPrincipal\_Txt\_Cpf.Enabled = true;

frmPrincipal\_Txt\_Email.Enabled = true;

frmPrincipal\_Cbx\_Access.Enabled = true;

frmPrincipal\_Txt\_Name.Clear();

frmPrincipal\_Txt\_Cpf.Clear();

frmPrincipal\_Txt\_Email.Clear();

pictureBox1.Image = null;

pictureBox2.Image = null;

pictureBox2.Visible = false;

frmPrincipal\_Btn\_Filtro.Visible = false;

}

private void frmPrincipal\_Btn\_Search\_Click(object sender, EventArgs e)

{

frmPrincipal\_Gbx\_Data.Visible = true;

frmPrincipal\_btn\_New.Enabled = false;

frmPrincipal\_Btn\_Save.Enabled = false;

frmPrincipal\_Btn\_Procurar.Enabled = true;

frmPrincipal\_Txt\_Name.Enabled = false;

frmPrincipal\_Txt\_Cpf.Enabled = false;

frmPrincipal\_Txt\_Email.Enabled = false;

frmPrincipal\_Cbx\_Access.Enabled = false;

frmPrincipal\_Btn\_Delete.Enabled = false;

frmPrincipal\_Txt\_Name.Clear();

frmPrincipal\_Txt\_Cpf.Clear();

frmPrincipal\_Txt\_Email.Clear();

pictureBox1.Image = null;

pictureBox2.Image = null;

}

private void frmPrincipal\_Btn\_Close\_Click(object sender, EventArgs e)

{

frmPrincipal\_Gbx\_Data.Visible = false;

}

private void frmPrincipal\_Btn\_Save\_Click(object sender, EventArgs e)

{

if (string.IsNullOrWhiteSpace(frmPrincipal\_Txt\_Name.Text) ||

string.IsNullOrWhiteSpace(frmPrincipal\_Txt\_Email.Text) ||

string.IsNullOrWhiteSpace(frmPrincipal\_Cbx\_Access.Text) ||

string.IsNullOrWhiteSpace(nomeArquivo) ||

string.IsNullOrWhiteSpace(frmPrincipal\_Txt\_Cpf.Text))

{

MessageBox.Show("Preencha todas as informações", "Campos em branco", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

Pessoa p = new Pessoa();

ImagemEmBits image = new ImagemEmBits();

Filtro filtro = new Filtro();

p.Name = frmPrincipal\_Txt\_Name.Text;

p.Cpf = frmPrincipal\_Txt\_Cpf.Text;

p.Email = frmPrincipal\_Txt\_Email.Text;

switch (frmPrincipal\_Cbx\_Access.SelectedIndex)

{

case 0:

p.IdAccess = 1;

break;

case 1:

p.IdAccess = 2;

break;

case 2:

p.IdAccess = 3;

break;

}

p.Image = image.ConverterImagemParaBytes(nomeArquivo);

pictureBox1.ImageLocation = nomeArquivo;

Validate();

db.Pessoas.Add(p);

db.SaveChanges();

MessageBox.Show("Cadastro efetuado com sucesso", "Cadastrado", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

public Image byteArrayToImage(byte[] img)

{

using (var ms = new MemoryStream(img))

{

return Image.FromStream(ms);

}

}

private void FrmPrincipal\_Btn\_Image\_Click(object sender, EventArgs e)

{

frmPrincipal\_Ofd\_Image.FileName = "";

frmPrincipal\_Ofd\_Image.Title = "Imagens";

frmPrincipal\_Ofd\_Image.Filter = "Todas Imagens|\*.jpg; \*.bmp; \*.png";

frmPrincipal\_Ofd\_Image.ShowDialog();

if (frmPrincipal\_Ofd\_Image.FileName.ToString() != "")

{

nomeArquivo = frmPrincipal\_Ofd\_Image.FileName.ToString();

pictureBox1.Image = new Bitmap(nomeArquivo);

}

}

private void FrmPrincipal\_Btn\_Procurar\_Click(object sender, EventArgs e)

{

byte[] imagem;

ImagemEmBits image = new ImagemEmBits();

imagem = image.ConverterImagemParaBytes(nomeArquivo);

if (imagem != null)

{

var resultado = from x in db.Pessoas

where x.Image == imagem

select x;

if (resultado.Count() > 0)

{

foreach (var r in resultado)

{

frmPrincipal\_Txt\_Name.Text = r.Name;

frmPrincipal\_Txt\_Email.Text = r.Email;

frmPrincipal\_Txt\_Cpf.Text = r.Cpf;

frmPrincipal\_Cbx\_Access.SelectedIndex = r.IdAccess - 1;

textBox1.Text = r.ID.ToString();

frmPrincipal\_Btn\_Filtro.Visible = true;

pictureBox2.Visible = true;

frmPrincipal\_Btn\_Delete.Enabled = true;

}

}

else

{

MessageBox.Show("Imagem não encontrada no Banco de Dados", "Imagem não cadastrada", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

else

{

MessageBox.Show("Imagem não selecionada", "Selecionar imagem", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

private void Button1\_Click(object sender, EventArgs e)

{

ImagemEmBits image = new ImagemEmBits();

Filtro filtro = new Filtro();

pictureBox2.Image = filtro.FiltroLaplace(nomeArquivo);

MessageBox.Show("Filtro aplicado na imagem selecionada");

}

private void FrmPrincipal\_btn\_New\_Click(object sender, EventArgs e)

{

frmPrincipal\_Txt\_Name.Clear();

frmPrincipal\_Txt\_Email.Clear();

frmPrincipal\_Txt\_Cpf.Clear();

frmPrincipal\_Cbx\_Access.SelectedIndex = 0;

pictureBox1.Image = null;

}

private void FrmPrincipal\_Btn\_Delete\_Click(object sender, EventArgs e)

{

int ID = int.Parse( textBox1.Text);

foreach (Pessoa p in db.Pessoas.Where(x => x.ID == ID))

db.Pessoas.Remove(p);

db.SaveChanges();

frmPrincipal\_Txt\_Name.Clear();

frmPrincipal\_Txt\_Email.Clear();

frmPrincipal\_Txt\_Cpf.Clear();

frmPrincipal\_Cbx\_Access.SelectedIndex = 0;

pictureBox1.Image = null;

pictureBox2.Visible = false;

frmPrincipal\_Btn\_Filtro.Visible = false;

}

private void FrmPrincipal\_Txt\_Cpf\_KeyPress(object sender, KeyPressEventArgs e)

{

//aceita apenas números, tecla backspace.

if (!char.IsNumber(e.KeyChar) && !(e.KeyChar == (char)Keys.Back))

{

e.Handled = true;

}

}

}

}

\*\*\*MODELS\*\*\*

\*\*\*CLASSE ACESSO\*\*\*

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace PROCESSAMENTO\_IMAGEM.MODELS

{

[Table("Acessos")]

public class Acesso

{

[Key]

public int IdAccess { get; set; }

public string TipoAcesso { get; set; }

}

}

\*\*\*CLASSE PESSOA\*\*\*

using System.ComponentModel.DataAnnotations;

using System.ComponentModel.DataAnnotations.Schema;

namespace PROCESSAMENTO\_IMAGEM.MODELS

{

[Table("Pessoas")]

public class Pessoa

{

[Key]

public int ID { get; set; }

public int IdAccess { get; set; }

public string Name { get; set; }

public string Email { get; set; }

public string Cpf { get; set; }

public byte[] Image { get; set; }

}

}

\*\*\*CONTEXTO\*\*\*

using PROCESSAMENTO\_IMAGEM.MODELS;

using System.Data.Entity;

using System.Data.Entity.ModelConfiguration.Conventions;

namespace PROCESSAMENTO\_IMAGEM.BLL

{

public class Contexto : DbContext

{

public DbSet<Pessoa> Pessoas { get; set; }

public DbSet<Acesso> Acessos { get; set; }

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();

modelBuilder.Conventions.Remove<ManyToManyCascadeDeleteConvention>();

modelBuilder.Conventions.Remove<OneToManyCascadeDeleteConvention>();

}

}

}

\*\*\*FILTRO\*\*\*

using System;

using System.Collections.Generic;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PROCESSAMENTO\_IMAGEM.BLL

{

public class Filtro

{/// <summary>

/// Filtro Passa-Alta.

/// </summary>

/// <param name="imagem"></param>

/// <returns></returns>

public Image FiltroLaplace(string imagem)

{

Bitmap Imagem = new Bitmap(imagem);

int Largura = Imagem.Width;

int Altura = Imagem.Height;

int[,] Laplace = new int[,] { { 0, 1, 0 }, { 1, -4, 1 }, { 0, 1, 0 } };

int todosPixelsR = 0;

int todosPixelsG = 0;

int todosPixelsB = 0;

int i;

int j;

int aux, aux2;

Bitmap NovaImagem = new Bitmap(Largura, Altura); //para trabalhar com a imagem, novo bmp com o mesmo tamanho da img original

for (i = 0; i < Altura - 2; i++)

{

for (j = 0; j < Largura - 2; j++)

{

for (aux = 0; aux < 3; aux++)

{

for (aux2 = 0; aux2 < 3; aux2++)

{

todosPixelsR += Imagem.GetPixel(j + aux, i + aux2).R \* Laplace[aux2, aux];

todosPixelsG += Imagem.GetPixel(j + aux, i + aux2).G \* Laplace[aux2, aux];

todosPixelsB += Imagem.GetPixel(j + aux, i + aux2).B \* Laplace[aux2, aux];

}

}

if (todosPixelsR < 0)

{

todosPixelsR = 0;

}

else

{

if (todosPixelsR > 255)

{

todosPixelsR = 255;

}

}

if (todosPixelsG < 0)

{

todosPixelsG = 0;

}

else

{

if (todosPixelsG > 255)

{

todosPixelsG = 255;

}

}

if (todosPixelsB < 0)

{

todosPixelsB = 0;

}

else

{

if (todosPixelsB > 255)

{

todosPixelsB = 255;

}

}

NovaImagem.SetPixel(j + 1, i + 1, Color.FromArgb(255, todosPixelsR, todosPixelsG, todosPixelsB));

todosPixelsR = 0;

todosPixelsG = 0;

todosPixelsB = 0;

}

}

return NovaImagem;

}

}

}

\*\*\*IMAGEM EM BITS\*\*\*

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PROCESSAMENTO\_IMAGEM.BLL

{

public class ImagemEmBits

{

public byte[] ConverterImagemParaBytes(string caminhoImagem)

{

byte[] arraybytes = null;

if (caminhoImagem != null)

{

FileInfo informacoesFicnheiro = new FileInfo(caminhoImagem);

long numeroBytes = informacoesFicnheiro.Length;

FileStream fStream = new FileStream(caminhoImagem, FileMode.Open, FileAccess.Read);

BinaryReader br = new BinaryReader(fStream);

arraybytes = br.ReadBytes((int)numeroBytes);

return arraybytes;

}

return null;

}

}

}