



Loja de Caleiras

Trabalho realizado no âmbito da Disciplina de Programação

Tiago Condeço Carvalho

Coimbra

2023

Índice

Introdução.....	2
Forms 1(Projeto).....	3
Menu.....	7
Serviços.....	7
Forms 2 e 3 (Projeto) - Serviços.....	9
Explicação do Código - Classes Serviços.....	10
Produtos.....	11
Carrinho.....	12
Botão Finalizar Compra.....	15
Interface De Utilizador.....	18
FeedBack.....	18
Faturas.....	19
Código do Load Form.....	21
Conclusão.....	22
Bibliografia.....	23

Introdução

O presente trabalho é sobre o desenvolvimento de uma aplicação em C#, para facilitar a gestão de serviços, produtos e carrinho de compras. Cada uma delas representada por botões na interface. Tem também um interface de Utilizador onde se vai poder ver as Faturas e onde poderá enviar o seu feedback.

O objetivo principal deste trabalho é criar uma aplicação intuitiva e funcional que permita ao usuário interagir de forma eficiente com os serviços oferecidos, os produtos disponíveis e gerir as suas compras por meio do carrinho integrado.

O trabalho está organizado em diversas partes, cada uma dedicada a um aspecto específico da aplicação:

- Temos a mensagem de bem-vindo.
- A parte de serviços.
- Gestão de produtos, apresentando as opções disponíveis..
- Implementação do carrinho de compras.
- Interface de Utilizador com parte de FeedBack e Faturas

Forms 1(Projeto)

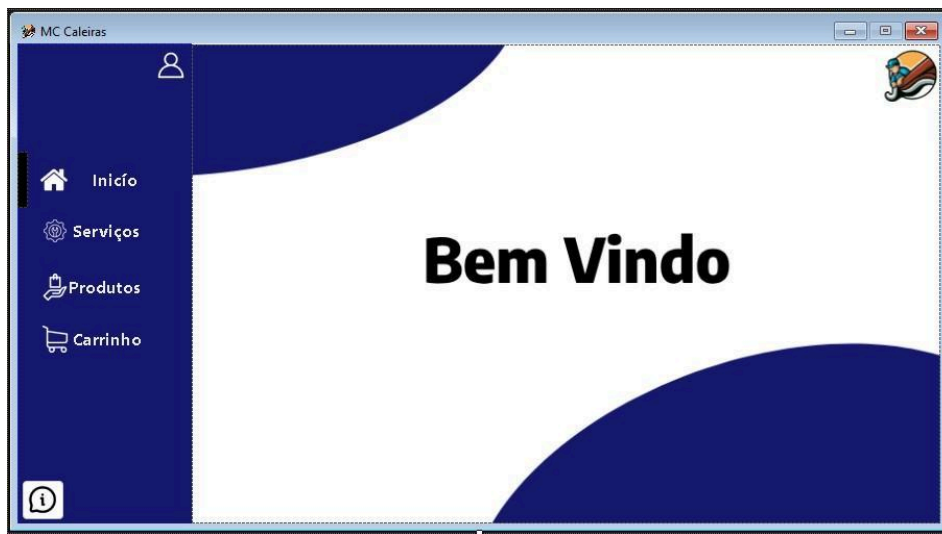


Figura 1- Início

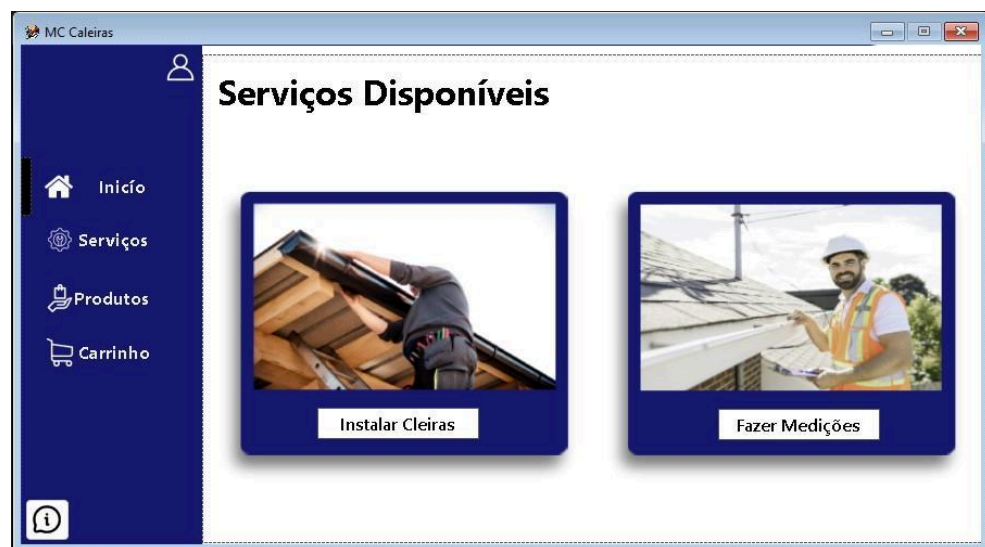


Figura 2- Serviços



Figura 3- Produtos

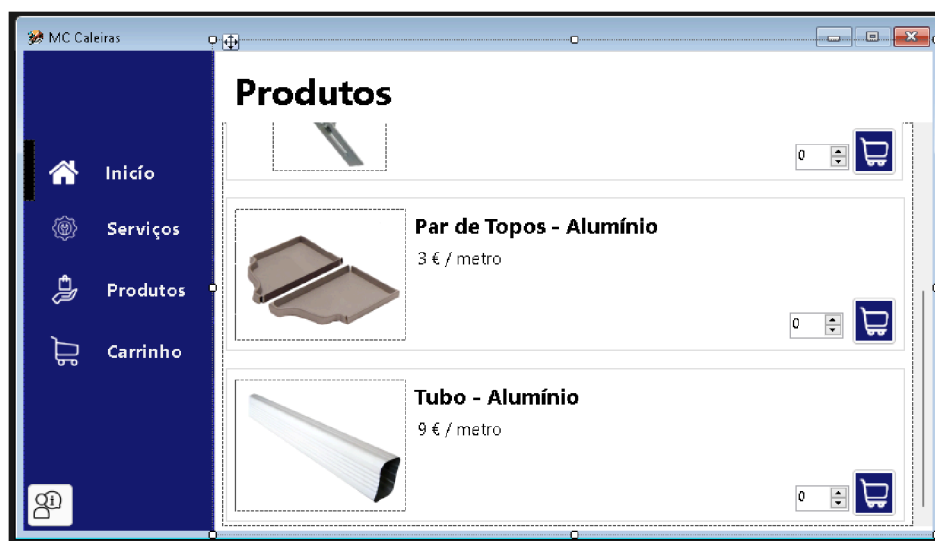


Figura 4- Produtos 2



Figura 5- Carrinho

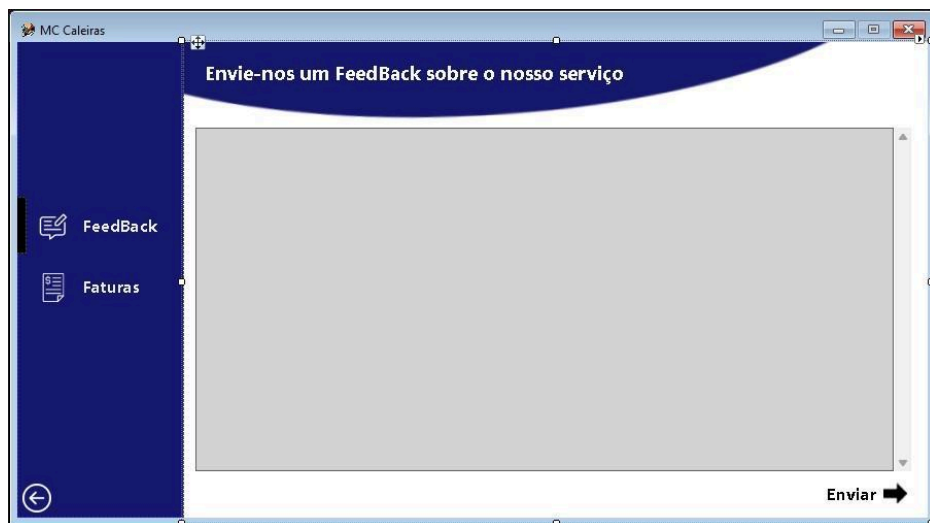


Figura 6- Utilizador FeedBack



Figura 7- Utilizador Faturas

No primeiro forms eu tenho um menu feito por botões, que estão dentro de um painel. Nesse mesmo painel tenho uma barra preta (painel), que serve para embelezar um pouco mais o menu, também serve para “dizer” em que parte do menu está. Falando mais dos botões temos os botões de Início, Serviços, Produtos , Carrinho e um Acerca de... e um botão de Utilizador. Todos esses botões têm uma parte que lhes corresponde,tudo no mesmo forms, como se pode ver nas imagens acima.

Menu

No menu eu tenho dois Painéis e 5 botões.

```
//Menu
1 referência
private void button1_Click(object sender, EventArgs e)
{
    barrapreta.Height = button1.Height;
    barrapreta.Top = button1.Top;
    groupBox1.Visible = true;

    //Por tudo o que n é serviço invisível
    groupBox2.Visible = false;
    groupBox3.Visible = false;
    groupBox8.Visible = false;
}
```

Figura 8- Menu

```
//Botão Acerca de...
1 referência
private void button5_Click(object sender, EventArgs e)
{
    MessageBox.Show("App feita por Tiago Condeço Cavalho 🐼 \n\n Tabalhador Mário Carvalho (Meu Pai)");
}
```

Figura 9- Botão Acerca de...

Assim que inicia o programa dos os layout's são postos invisíveis e o único visível é o do Bem-Vindo. Mostrando agora como funciona a lógica dos botões, eu vou apenas mostrar o botão de "Início" e "Acerca de...", visto que os outros dois funcionam da mesma forma, exceto o do "Carrinho", mas esse eu vou mostrar mais para a frente.

No botão temos, nas primeiras linhas de código o try-catch, que vai aparecer ao longo deste trabalho mas explicando já como funciona, serve para que caso o haja algum problema no código não tenha de fechar e apenas mostra um mensagem de erro numa MessageBox e não executa a ação. A seguir temos uma linhas que servem para quando clicarmos no botão a barra preta fique ao lado do botão, para que a barra preta tenha a mesma altura e a mesma posição vertical que o botão. Coloca-se a "GroupBox" que corresponde ao botão neste caso a do Bem-Vindo. E as ultimas linhas servem para por o todas as outras "GroupBox" invisíveis.(Serviços, Produtos e Carrinho)

Botão "Acerca de...", nesse botão apenas uso uma "MessaBox" que diz quem fez o trabalho e quem é que faz as caleiras, que é o meu pai.

Serviços

Nos serviços eu utilizo uma “Label” uma “PictureBox” e dois Botões

```
1 referência
private void button6_Click(object sender, EventArgs e)
{
    Instalacao instalacao = new Instalacao();
    instalacao.Show();
}

1 referência
private void button7_Click(object sender, EventArgs e)
{
    Medicao medicao = new Medicao();
    medicao.Show();
}
```

Figura 10- Código Forms's

Nos “Serviços” eu decidi fazer uns “forms” à parte que são literalmente iguais só muda o nome. O código dos botões da parte do “Serviços” é para abrir os formes que criei.

Forms 2 e 3 (Projeto) - Serviços

Destes dois “Forms” usei uma “PictureBox” (uma em cada), quatro “textbox” (em cada) e um botão (em cada).

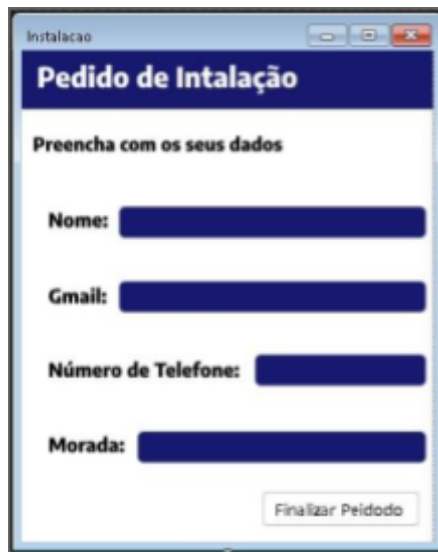


Figura 11- Forms Instalação

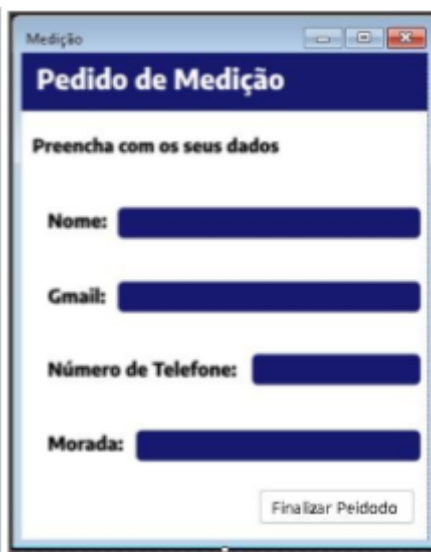


Figura 12- Forms Manutenção

```
Serviços cla1 = new Serviços();
Serviços2 cla2 = new Serviços2();

1 referência
public Instalacao()
{
    InitializeComponent();
}

1 referência
private void button1_Click(object sender, EventArgs e)
{
    cla1.Nome = Convert.ToString(textBox1.Text);
    cla1.Gmail = Convert.ToString(textBox2.Text);
    cla2.Morada = Convert.ToString(textBox4.Text);
    cla2.Tele = Convert.ToInt32(textBox3.Text);

    DialogResult result = MessageBox.Show(cla2.Meng(cla1.Nome, cla1.Gmail), "Formulário Instalação", MessageBoxButtons.OK, MessageBoxIcon.Information);

    if (result == DialogResult.OK)
    {
        Close();
    }
}
```

Figura 13- Botão Finalizar pedido

Apenas vou explicar um visto que o código doutro é igual só muda o nome. Na imagem acima eu crio uma variável associada às classes que criei. Já dentro dos botões do botão tenho o “cla1.Nome” e o “cla1.Gmail” que serve para ler e guardar o que tiver escrito nas “textBox” na classe Serviços (classe1) e o cla2.Morada e o cla2.Tele é a mesma coisa mas vão para a classe Serviços 2 (classe2). Pós isso tenho uma variável que vai guardar a resposta da “MessageBox”. Já na “MessageBox” tenho para mostrar o método cla2.Meng (vou explicar a seguir), nesse método eu mando as variáveis Nome e Gmail, depois o nome da “MessageBox” por fim o botão e o ícone que vão aparecer. No fim tenho um if que simplesmente se o resultado for OK (que vai ser porque era a única opção) ele fecha os forms.

Explicação do Código - Classes Serviços

```
abstract internal class Serviços
{
    0 referências
    public Serviços()
    {
        nome = "";
        gmail = "";
        obg = "\n\nObrigado por nos escolher!!";
    }

    string nome, gmail;
    public string obg;

    4 referências
    public string Nome
    {
        get { return nome; }
        set { nome = value; }
    }

    4 referências
    public string Gmail
    {
        get { return gmail; }
        set { gmail = value; }
    }

    1 referência
    protected string MengInicio(string n)
    {
        return "Em breve iremos contactar consigo Sr./Sr.ª " + n + "\n\nOs seus dados: ";
    }

    3 referências
    public abstract string Meng(string n, string g);
}

0 referências
~Serviços()
{ }
```

Figura 14- Classe Serviços

```
6 referências
internal class Serviços2 : Serviços
{
    string morada;
    int tele;

    3 referências
    public string Morada
    {
        get { return morada; }
        set { morada = value; }
    }

    3 referências
    public int Tele
    {
        get { return tele; }
        set { tele = value; }
    }

    3 referências
    public override string Meng(string n, string g)
    {
        return MengInicio(n) + "\n\nNome: " + n + "\nGmail: " + g + "\nMorada: " + Morada + "\nTelemóvel: " + Tele + obg;
    }

    2 referências
    public Serviços2()
    { }

    0 referências
    ~Serviços2()
    { }
}
```

Figura 15- Classe Serviços 2

Nas classes como era de esperar tenho as variáveis como privadas (nome, mail) porque era um dos pontos perdidos, uma pública (obg) e também o get e set para ler e atribuir o valor à variável Morada por exemplo.

Na classe Serviços que é uma classe abstrata tenho o construtor a inicializar as variáveis, um método "protected" (MengInicio), que contém o início do texto que vai ser apresentado na "MessageBox", neste método estou a receber uma string "n" que é o "Nome" e uma string "n" que vem do método "Meng" este método está como abstract.

Na classe Serviços 2 que é uma classe derivada da classe Serviços, nesta tenho mais duas classes privadas (morada e tele), e para cada uma o seu respectivo get, set. O método Meng que era a função abstrata e uso o "override" como polimorfismo de maneira a que eu possa usar o mesmo nome, está a receber dos forms as variáveis string "n" e string "g", dentro do método é o que vai aparecer no fim da MenssageBox, chamo o método "MengInicio", depois é o resto do texto e chamar as variáveis e para poder usar a herança uso uma a variável pública que tenho na classe Serviços. Por fim os construtores e os destrutores.

Produtos

Usei um “grupbox” que engloba tudo, uma “Label” como título, um “flowLayoutPanel1” para fazer poder ter aquela barrinha para puxar a página pa cima e para baixo (nas propriedades o AutoScroll está True), dentro dele tenho mais 4 “grupbox’s” e em cada tenho uma “pictureBox”, um botão um numeric uptodown e duas “Label’s”

```
1 referência
private void button8_Click(object sender, EventArgs e)
{
    cla3.valor = 12;

    button8.Enabled = false;

    numericUpDown1.Value = 1;

    cla3.cont = 1;
}

1 referência
private void numericUpDown1_ValueChanged(object sender, EventArgs e)
{
    cla3.quant = (int)numericUpDown1.Value;

    cla3.Quantidade();

    if (cla3.quant == 0)
    {
        button8.Enabled = true;
        numericUpDown1.Enabled = false;

        cla3.valor = 0;
        cla3.cont = 0;
    }

    else if (cla3.quant != 0)
    {
        numericUpDown1.Enabled = true;
    }
}
```

Figura 16 - Botão carrinho e NumericUpDown

```

2 referências
internal class Class3
{
    public double total;

    public double valor, valor2, valor3, valor4;

    public int quant, quant2, quant3, quant4;

    public int cont, cont2, cont3, cont4;

    2 referências
    public double Quantidade()
    {
        return valor = quant * 12;
    }

    2 referências
    public double Quantidade2()
    {
        return valor2 = Math.Round(quant2 * 2.5, 2);
    }
}

```

Figura 17 - Classe3, Variáveis e Método Quantidade

Como funciona a lógica dos botões do numericUpDown, vou explicar apenas um porque nos outros a lógica é a mesma. No botão com a imagem de carrinho, sempre que ele é premido vai ser guardado na “Classe3” na variável valor o valor do produto neste caso é 12, em seguida o botão fica desabilitado e o “numeric” fica com o valor de 1 (ou seja que foi adicionado um produto ao carrinho) e a variável cont passa a 1 (o cont vou explicar a seguir para que serve). No “NumericUpDown” guardamos o valor dele no “quant” (quantidade) que vai servir para multiplicar pelo valor caso o utilizador queira mais do que um produto e esse cálculo vai acontecer na “Classe3” nos métodos Quantidade, são todos iguais exceto na Quantidade2 que eu adicionei um “Math.Round” para por duas casas decimais (que neste caso não era muito preciso). No forms é só chamar essa função. Adicionei também um “if” e um “if else” que se o “quant” for igual a 0 o botão para adicionar o primeiro produto ao carrinho, volta a estar ativo. *Reseta* as outras duas variáveis voltar a ficar a 0. Mas se “quant” for diferente de 0 o “numericUpDown” continua ativo.

Carrinho

Usei uma pictureBox (para a imagem de fundo) duas "Label's", uma "textBox" para apresentar o valor total, um "flowLayoutPanel" dentro desse tenho 4 "grupbox's" dentro da cada uma delas tenho duas "Labe's", um "pictureBox" e um botão e mais um botão para finalizar o pedido

```
1 referência
private void button4_Click(object sender, EventArgs e)
{
    barrapreta.Height = button4.Height;
    barrapreta.Top = button4.Top;

    //Por tudo o que n é carrinho invisivel
    groupBox1.Visible = false;
    groupBox2.Visible = false;
    groupBox3.Visible = false;

    //Mostrar Coisas do Carrinho
    groupBox8.Visible = true;

    //Ocultar tudo o resto
    groupBox9.Visible = false;
    groupBox10.Visible = false;
    groupBox11.Visible = false;
    groupBox12.Visible = false;

    //Verificar se o botão foi usado
    if (cla3.cont == 1)
    {
        groupBox9.Visible = true;
    }

    if (cla3.cont2 == 1)
    {
        groupBox10.Visible = true;
    }

    if (cla3.cont3 == 1)
    {
        groupBox11.Visible = true;
    }

    if (cla3.cont4 == 1)
    {
        groupBox12.Visible = true;
    }

    textBox1.Text = Convert.ToString(cla3.Total() + "€");
}
```

Figura 18 - Código do Carrinho

```
5 referências
public double Total()
{
    return Math.Round(Quantidade() + Quantidade2() + Quantidade3() + Quantidade4(), 2);
}
```

Figura 19 - Classe3, Método Total

O código do carrinho eu coloquei no próprio botão do menu. O código começa por ocultar todas as outras “groupBox’s” que eu tinha, põe-se visível a groupBox do carrinho, oculta-se as “grupBox’s” dos produtos, a seguir tenho uns “if’s” que vão usar a variável “cont ” que eu tinha criado, para se o cont for igual a 1 (ou seja que o botão foi usado) ele põe a respetiva “gurpoBox” do produto visível.

Na última linha é para apresentar o método “Total” na “textBox” (é usado o Convert.ToString para converter para texto e não haver problema ao apresentar na “textBox”), as contas são feitas na “Classe3”, uso um “Math.Rounbd” (como disse à pouco este caso não é muito preciso porque os valores são simples), dentro dele é só somar os métodos “Quantidades” (somar os valores com já com a quantidade de cada um).

```
//Remover
1 referência
private void button12_Click(object sender, EventArgs e)
{
    groupBox9.Visible = false;

    cla3.total = Math.Round(cla3.Total() - cla3.valor, 2);

    textBox1.Text = Convert.ToString(cla3.total);

    numericUpDown1.Value = 0;
    cla3.valor = 0;
}
```

Figura 20 - Código Botão Remover

Por último o botão de remover, ao clicar oculta-se a “groupBox” do respetivo produto, guarda-se na variável “cla3.total” o valor do “Total” menos o valor do produto removido, mostro o valor da viável total e no fim volto a pôr o repativo “numericUpDown” e o “valor” a 0 ou seja como tava nu início.

Botão Finalizar Compra

```
1 referência
private void button22_Click(object sender, EventArgs e)
{
    try
    {
        if (cla3.cont == 1 || cla3.cont2 == 1 || cla3.cont3 == 1 || cla3.cont4 == 1)
        {
            i += 1;

            string prod1 = "", prod2 = "", prod3 = "", prod4 = "";
            double totalFat = cla3.Total();

            if (cla3.cont == 1)
            {
                prod1 = "\n\nCaleira - Alumino 12€ / metro" + "\nQuantidade:" + cla3.quant;
            }
            else if (cla3.cont == 0)
            {
                totalFat -= cla3.Quantidade();
            }

            if (cla3.cont2 == 1)
            {
                prod2 = "\n\nSuporte - Alto Inox 2,50€" + "\nQuantidade:" + cla3.quant2;
            }
            else if (cla3.cont2 == 0)
            {
                totalFat -= cla3.Quantidade2();
            }

            if (cla3.cont3 == 1)
            {
                prod3 = "\n\nPar de Topos -Aluminio 3€" + "\nQuantidade:" + cla3.quant3;
            }
            else if (cla3.cont3 == 0)
            {
                totalFat -= cla3.Quantidade3();
            }

            if (cla3.cont4 == 1)
            {
                prod4 = "\n\nTubo - Alumino 9€ / metro" + "\nQuantidade:" + cla3.quant4;
            }
            else if (cla3.cont4 == 0)
            {
                totalFat -= cla3.Quantidade4();
            }

            //Nome do Ficheiro e da pasta
            string fatura = "Fatura(" + i + ")";
            string nomePasta = "Faturas";
        }
    }
}
```

Figura 21 - Código Botão Finalizar Pedido

```

string pastaProg = Environment.CurrentDirectory; //caminho da pasta onde está a ser executado

//Para por tudo num caminho só
string caminhoFinal = Path.Combine(pastaProg, nomePasta);
string caminhoFicheiro = Path.Combine(caminhoFinal, fatura + ".txt");

FileStream fs = new FileStream(caminhoFicheiro, FileMode.OpenOrCreate);

string texto = "\tMc Caleiras" + prod1 + prod2 + prod3 + prod4 + "\n\nTotal: " + totalFat + "\n\nObrigado pela sua compra :)";

if (fs.Length == 0)
{
    //ficheiro vazio
    StreamWriter sw = new StreamWriter(fs);

    MessageBox.Show("Compra finalizado com sucesso !!", ":)", MessageBoxButtons.OK, MessageBoxIcon.Information);

    sw.Write(texto);

    sw.Close();

    //FAZER RESET
    groupBox9.Visible = false;
    groupBox10.Visible = false;
    groupBox11.Visible = false;
    groupBox12.Visible = false;

    textBox1.Clear();

    numericUpDown1.Value = 0;
    numericUpDown2.Value = 0;
    numericUpDown3.Value = 0;
    numericUpDown4.Value = 0;

    cla3.cont = 0;
    cla3.cont2 = 0;
    cla3.cont3 = 0;
    cla3.cont4 = 0;
}
fs.Close();

dataGridView1.Rows.Add(fatura);
}
else
{
    MessageBox.Show("Impossível finalizar compra não tem nada no carrinho", "Erro catastrófico", MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
}
}
catch (Exception)
{
    MessageBox.Show("\n\nErro ao finalizar pedido!!", "Erro");
}
}

```

Figura 22 - Código Botão Finalizar Pedido Continuação

Dentro do try, há um “if” para determinar se algum produto foi selecionado para compra (cla3.cont, cla3.cont2, cla3.cont3 ou cla3.cont4 se for igual a 1) tal como na parte do carrinho. As variáveis “prod” que vão servir para guardar os “dados” de cada produto. Ou seja caso tenha sido adicionado ao carrinho também é guardado os dados nas variáveis. Depois o total para fazer o cálculo do total para por na fatura. O “else if” que funciona com o botão de remover caso se tire do carrinho também subtrai o preço na parte da fatura.

Variáveis que vou usar bastante:

fatura - serve para guardar o nome do Ficheiro e o “i” para casoso se criem duas, não ficarem com o mesmo nome

nomePasta = guarda o nome (Faturas) da pasta criada manualmente no bin debug

pastaProg = guarda o caminho de onde o programa está a ser executado

caminhoFinal = Combina o caminho pastaProg e nome da pasta criada num só criando assim um caminho para a pasta das Faturas

caminhoFicheiro= Combina o caminho da pasta (caminhoFinal) com o nome do ficheiro (fatura) a criar

Após isso o “FileStream” que se encarrega de criar o ficheiro dentro da pasta, a variável “texto” que guarda tudo para se conseguir escrever no ficheiro. O “if” que verifica que o ficheiro está vazio e podendo assim usar o “StreamWriter” para escrever no ficheiro, por fim a “MessageBox” que vai mostrar uma mensagem ao utilizador para lhe confirmar que a compra foi feita com sucesso, terminamos a escrita. As linhas a baixo servem para “Resetar” as coisas para por como estavam pondo as “groupBox” invisíveis os contadores a 0 os “numeric” também a 0. Já fora do “if”, fecha-se o ficheiro e vou adicionar à “dataGridView” o nome do ficheiro que foi criado (exemplo: Fatura(1)). Os respectivos “else” e “catch”, com mensagens de erro para caso algo não funcione.

Interface De Utilizador

```
1 referência
private void button18_Click(object sender, EventArgs e)
{
    panel2.Visible = false;
}
```

Figura 23 - Código Botão Menu Utilizador Voltar

```
//feedback
1 referência
private void button21_Click(object sender, EventArgs e)
{
    barra Preta2.Height = button21.Height;
    barra Preta2.Top = button21.Top;

    panel5.Visible = true;
    panel4.Visible = false;
}
```

Figura 24 - Código Botão Menu Utilizador FeedBack

```
//Faturas
1 referência
private void button20_Click(object sender, EventArgs e)
{
    barra Preta2.Height = button20.Height;
    barra Preta2.Top = button20.Top;

    panel5.Visible = false;
    panel4.Visible = true;
}
```

Figura 25 - Código Botão Menu Utilizador Faturas

Nesta parte vou falar da parte de Utilizador onde se pode ver as faturas e enviar um feedBack, por ser uma “janela” nova tem também um menu com três botões um do FeedBack, o das faturas e o de voltar para a página principal. Nesta janela de utilizador eu optei por utilizar painéis em vez de groupbox. O código dos botões do menu é igual ao outro, a barra Preta 2 que serve para o utilizador saber onde está, o código serve para que ela fique ao mesmo nível que o botão, e depois tanto estes botões como no de voltar o painel “.Visible= true / false” para ficar visível ou não consoante onde o utilizador clica.

FeedBack

Nesta parte eu usei um painel que tem como fundo uma imagem, uma label, uma text box e um botão.

```
private void button19_Click(object sender, EventArgs e)
{
    try
    {
        if (textBox2.Text == "")
        {
            MessageBox.Show("Ups parece que não escreveu nada no seu feedback", ":", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        else
        {
            //Nome do Ficheiro
            i += 1;

            string feedb = "FeedBack(" + i + ")";

            string pastaProg = Environment.CurrentDirectory; //Serve pa encontrar o caminho em que o programa está a ser executado
            //Para por tudo num caminho só
            string caminhoFicheiro = Path.Combine(pastaProg, feedb + ".txt");

            FileStream fs = new FileStream(caminhoFicheiro, FileMode.OpenOrCreate);

            string texto = textBox2.Text;

            if (fs.Length == 0)
            { //ficheiro vazio
                MessageBox.Show("A enviar...", "Info", MessageBoxButtons.OK, MessageBoxIcon.Information);

                StreamWriter sw = new StreamWriter(fs);

                sw.Write(texto);

                sw.Close();
            }

            fs.Close();

            textBox2.Clear();
        }
    }
    catch (Exception)
    {
        MessageBox.Show("\n\nErro ao enviar dados!!", "Erro");
        textBox2.Clear();
    }
}
```

Figura 26 - Código Botão Enviar FeedBack

Deste código começo com try / catch para caso algo corra mal, em seguida um “if” que caso o utilizador não tenha escrito nada, manda um ameng a dizer para ele escrever algo. No “else” tenho a mesma lógica que na criação da fatura. Criei um ficheiro com um nome de feedback, só é preciso “pegar” naquilo que o utilizador põe na textbox numa variável, escrever no ficheiro e é só. Esse ficheiro vai ficar no bin debug, para ficar “escondido”, porque quem vai ver o feedback é quem fez a app ou quem o dono da loja. No um Clear para limpar a textBox.

Faturas

Nesta parte eu uso dois botões, um dataDridView, uma richTextBox e uma textBox

```
1 referência
private void button23_Click(object sender, EventArgs e)
{
    textBox3.Clear();
    dataGridView1.ClearSelection();
}
```

Figura 27 - Código Botão Limpar a TextBox de ver faturas

Este é o botão limpar o que está escrito na richTextBox e a desseleccionar qualquer coisa que estiver seleccionada na dataGridView.

```
private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        // Verifica se a célula clicada é válida
        if (e.RowIndex >= 0 && e.ColumnIndex >= 0)
        {
            // Obtém o valor da célula clicada
            string valorCelula = dataGridView1.Rows[e.RowIndex].Cells[e.ColumnIndex].Value.ToString();

            string nomePasta = "Faturas";

            string pastaProg = Environment.CurrentDirectory; //caminho da pasta onde está a ser executado

            //Para por tudo num caminho só
            string caminhoFinal = Path.Combine(pastaProg, nomePasta);
            string caminhoFicheiro = Path.Combine(caminhoFinal, valorCelula + ".txt");

            if (valorCelula != null)
            {
                FileStream fs = new FileStream(caminhoFicheiro, FileMode.Open);

                if (fs.Length != 0)
                { //ficheiro não está vazio

                    StreamReader sr = new StreamReader(fs);

                    string texto;

                    texto = sr.ReadToEnd();

                    richTextBox1.Text = texto;

                    sr.Close();
                }
            }
            else
            {
                // Se o valor da célula for nulo, limpa a TextBox
                richTextBox1.Clear();
            }
        }
    }
    catch
    {
        MessageBox.Show("Erro, essa célula já não deve existir");
    }
}
```

Figura 28- Código dataGridView

Aqui eu tenho o try/catch dentro dele um (nesta parte precisei da ajuda do chat)“if” que serve para verificar se tem alguma coisa seleccionada na “data...”. “valorCelula” que guarda o nome da fatura que o utilizador seleccionou, as variáveis que já expliquei em cima. Em seguida um outro “if ” que se a variável for diferente de null ou seja se tiver algum valor atribuído vai usar o StreamWriter para ler o que tem no ficheiro e escrever na richTextBox, por fim os “else” para limpar a “rich” e o catch que mostra uma mensagem de erro.

```

1 referência
private void button24_Click(object sender, EventArgs e)
{
    try
    {
        DialogResult result = MessageBox.Show("Eliminar uma fatura é ILEGAL", "SEU BURRO", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
        if (result == DialogResult.Yes)
        {
            if (textBox4.Text == "")
            {
                MessageBox.Show("Tem introduzir o nome da pasta !!", "", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            else
            {
                string nomePasta = "Faturas";

                string pastaProg = Environment.CurrentDirectory; //caminho da pasta onde está a ser executado

                string apagarF = textBox4.Text;

                //Para por tudo num caminho só
                string caminhoFinal = Path.Combine(pastaProg, nomePasta);
                string caminhoFicheiro = Path.Combine(caminhoFinal, apagarF + ".txt");

                if (File.Exists(caminhoFicheiro))
                {
                    File.Delete(caminhoFicheiro);
                    MessageBox.Show("Ficheiro eliminado com sucesso!");
                }
                else
                {
                    MessageBox.Show("O ficheiro não existe.");
                }

                dataGridView1.Rows.Clear();
                richTextBox1.Clear();
                textBox4.Clear();
            }
        }
        else
        {
            MessageBox.Show("Boa escolha :)", "Uff");
        }
    }
    catch (Exception)
    {
        MessageBox.Show("\n\nErro ao eliminar o ficheiro!!" + "Erro");
    }
}

```

Figura 29- Código apagar Ficheiro

Começo com o try/catch que dentro dele tenho um “if” para se a resposta da MessageBox acima for sim ele apaga o ficheiro caso contrário não apaga e mostra outra MessageBox. Um outro “if” e “else” que serve só para garantir que a pessoa escreva o nome do ficheiro que quer apagar, quando o fizer tenho as variáveis dos caminhos. Após isso, mais um “if” para verificar se o ficheiro existe, se sim ele usa o “File.Delete” para apagar o ficheiro e mostra uma MessageBox a dizer que foi apagado. Caso contrário um MessageBox a dizer que o ficheiro não existe. No final de tudo o cath com uma mensagem caso não funcione o botão.

Código do Load Form

```

1 referência
private void Form1_Load(object sender, EventArgs e)
{
    string nomePasta = "Faturas";

    string pastaProg = Environment.CurrentDirectory;

    string caminhoPasta = Path.Combine(pastaProg, nomePasta);

    if (!Directory.Exists(caminhoPasta))
    {
        Directory.CreateDirectory(caminhoPasta);
    }
    else
    {
        //
    }
}

```

Figura 30- Código ao Iniciar o Programa

Esta parte do código é simples apenas serve para quando o programa iniciar cria uma pasta para as faturas no Bin/Debug.

Diagrama de Classes

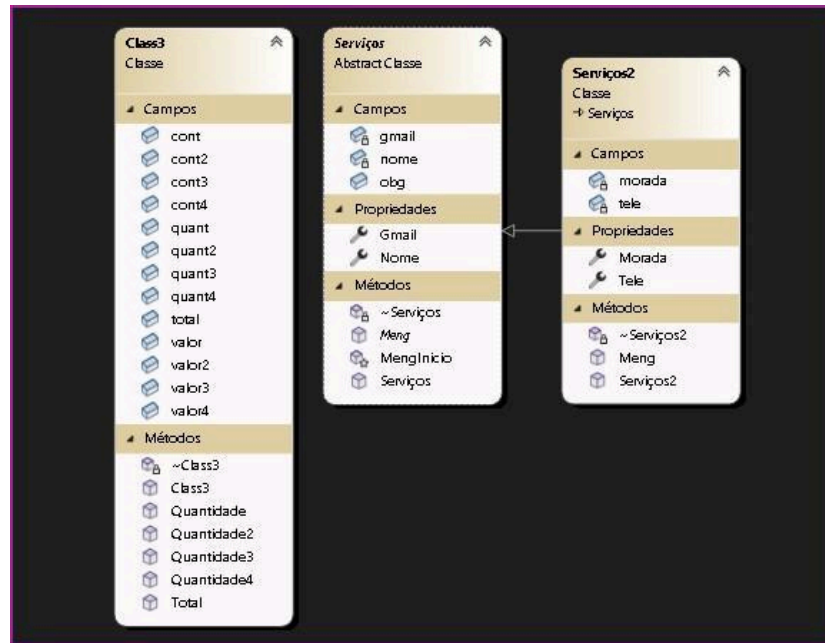


Figura 18 - Código Botão Remover

Aqui está o diagrama de classes de menina a que possamos ver o que essencial utilizado nas classes e de forma simplificada. Na parte de campos são as variáveis utilizadas nas propriedades são os Get e Set usados e nos métodos são os métodos.

Conclusão

Este projeto de desenvolvimento em C# para gestão de serviços, produtos e carrinho de compras proporcionou uma imersão completa nos princípios da programação orientada a objetos avançada e na criação de interfaces. Desde a construção detalhada do menu inicial até a implementação prática do carrinho de compras, cada etapa demonstrou não apenas uma compreensão sólida dos conceitos fundamentais, mas também a habilidade de aplicá-los de forma eficiente e escalável. Além disso, a colaboração com o Chat GPT em momentos-chave do projeto ressaltou a importância da busca por conhecimento em diversas fontes e a capacidade de adaptar soluções para desafios específicos. Em suma, esta jornada não apenas fortaleceu nosso conhecimento técnico, mas também consolidou habilidades essenciais para a construção de aplicações robustas e funcionais em C#.

Bibliografia

- Exercícios anteriormente feitos na aula;
- Auxílio do Chat GPT, numa parte do código que eu tinha *int quant = Convert.ToInt32(numericUpDown1);* tava a dar erro e o Chat sugeriu que usa-se *int quant = (int)numericUpDown1.Value;* E na parte das Faturas como referi;

Vídeos da plataforma YouTube:

- Para saber como fazer a página do produtos:
[C# - Scrolling a Flow layout Panel using Buttons in WinForm App](#)
- Para aprender como fazer o menu:
[C# - Designing a Flat desktop Application of a Fast Food Restaurant](#)