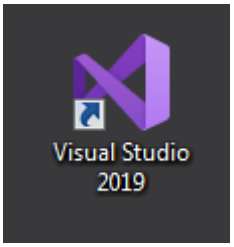
Autor: Edimar Soares Rodrigues Júnior

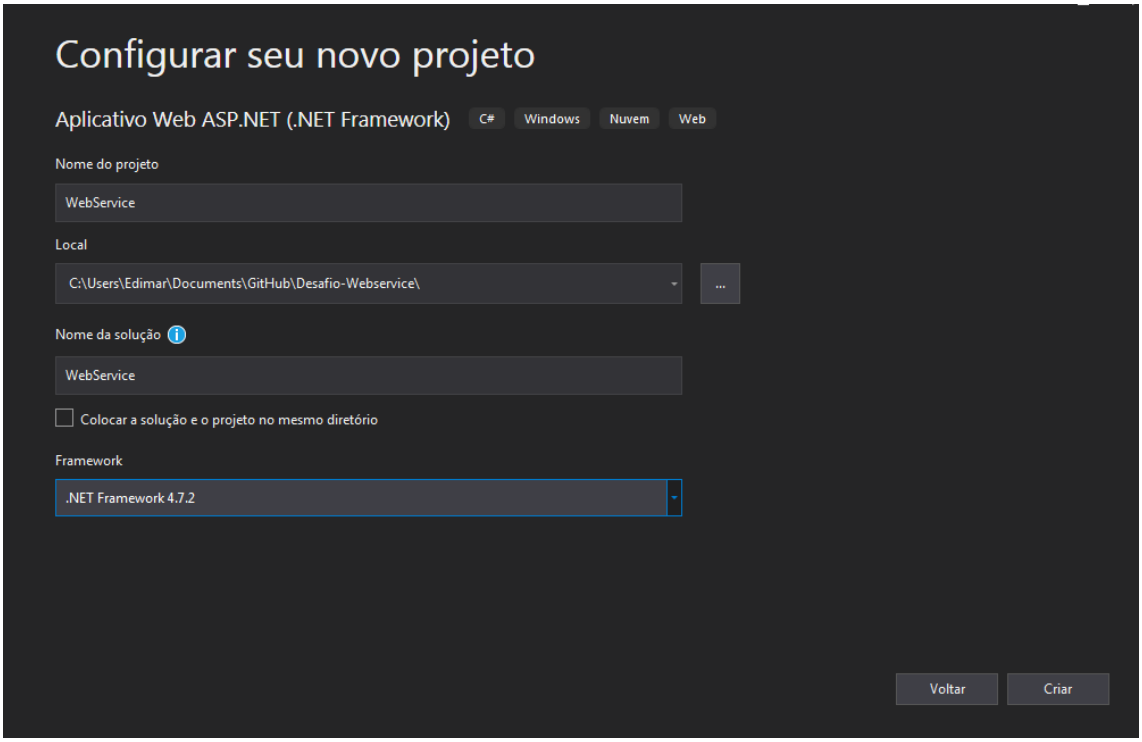
1) Ambiente de Desenvolvimento;

   Foi utilizado a IDE do Visual Studio 2019 para realizar o desenvolvimento da aplicação que realiza o upload de arquivos na linguagem C#.



2) Web Service

   Foi utilizado o modelo Aplicativo Web ASP.NET (.NET Framework) para realizar o desenvolvimento do Web Service, fazendo um localhost na máquina local que será consumido posteriormente pela aplicação.



Foi feito um bloco try/catch para realizar o tratamento de erro, retornando uma mensagem acusando se o arquivo escolhido para ser feito o upload foi bem-sucedido ou não.

```
    {
        [WebMethod]
        0 referências
        public string UploadArquivo(String nomeDoArquivo, byte[] arquivoByte)
        {
            try
            {
                String arquivo = Server.MapPath("~/Arquivos/") + nomeDoArquivo;
                System.IO.File.WriteAllBytes(arquivo, arquivoByte);
            }
            catch (Exception ex)
            {
                return "Erro ao realizar o Upload! (" + ex.Message + ")";
            }
            return "Upload realizado com sucesso!";
        }
    }
}
```

Foi criado uma pasta no diretório do Web Service chamada de Arquivos, onde será utilizado para armazenar os arquivos que serão *upados*.

Server.MapPath(“~”) retorna o caminho físico para a raiz do aplicativo, sendo assim, foi utilizado para acessar a pasta Arquivos.

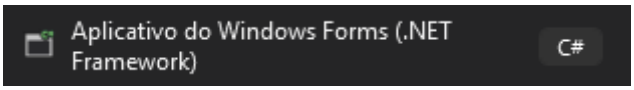
O método File.WriteAllBytes(String, Byte[]) cria um novo arquivo, grava a matriz de bytes especificada no arquivo e fecha o arquivo, o primeiro parâmetro deve conter o arquivo para gravação, e o segundo parâmetro os bytes a serem gravados no arquivo.

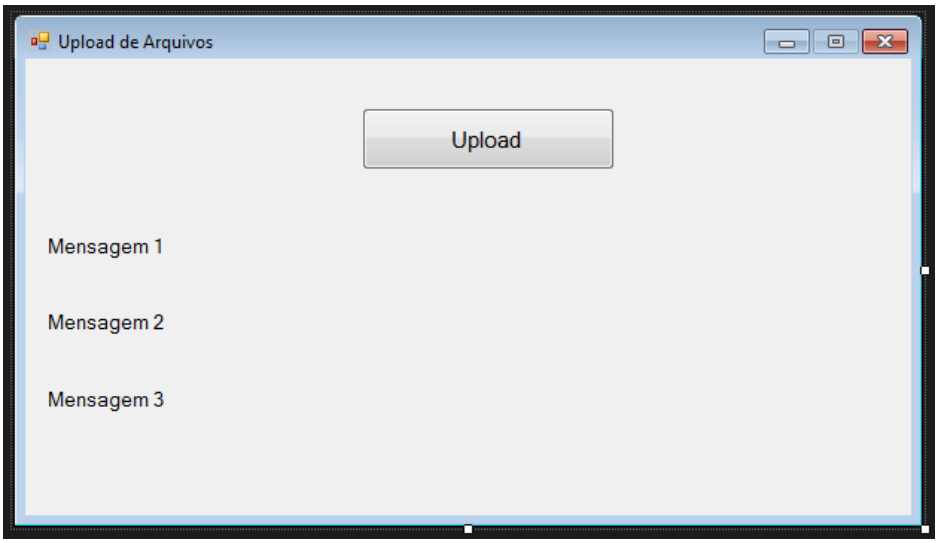
```
    {
        [WebMethod]
        0 referências
        public string UploadArquivo(String nomeDoArquivo, byte[] arquivoByte)
        {
            try
            {
                String arquivo = Server.MapPath("~/Arquivos/") + nomeDoArquivo;
                System.IO.File.WriteAllBytes(arquivo, arquivoByte);
            }
            catch (Exception ex)
            {
                return "Erro ao realizar o Upload! (" + ex.Message + ")";
            }
            return "Upload realizado com sucesso!";
        }
    }
}
```

O nome do método para fazer a chamada na aplicação foi UploadArquivo.

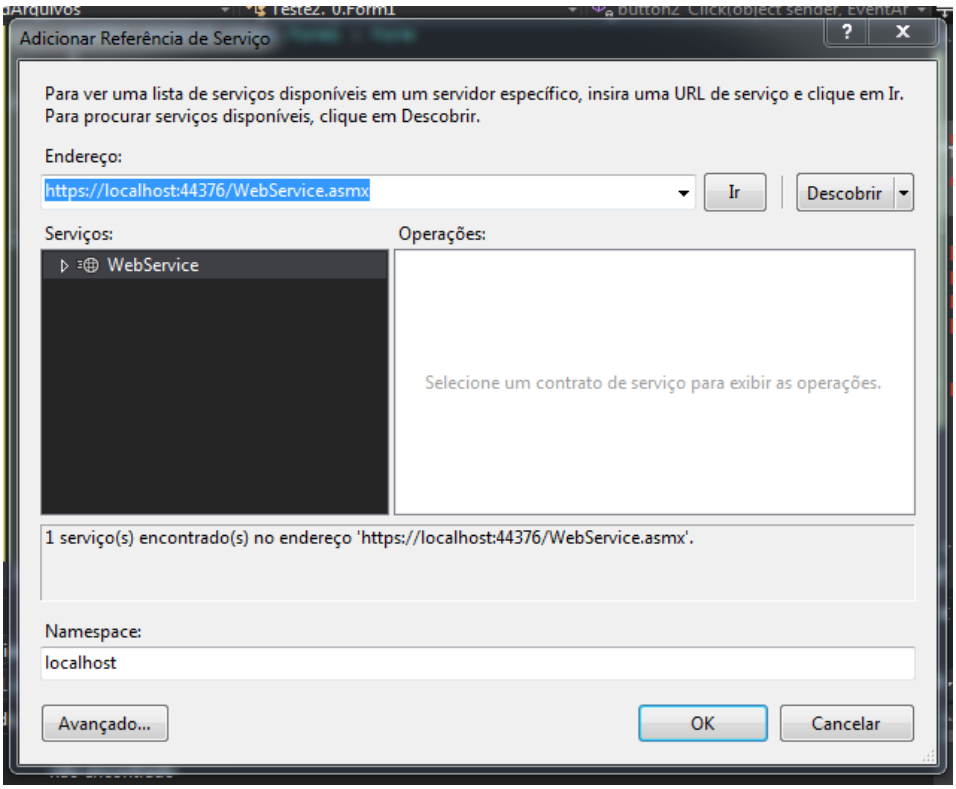
3) Implementação do Aplicativo

Foi utilizado o modelo Aplicativo do Windows Forms (.NET Framework) para realizar o desenvolvimento da aplicação Front-end, utilizando um botão e algumas labels para retornar mensagens para o usuário.





Foi implementado uma referência de Serviço que fará a chamada do Web Service que deve estar em execução.

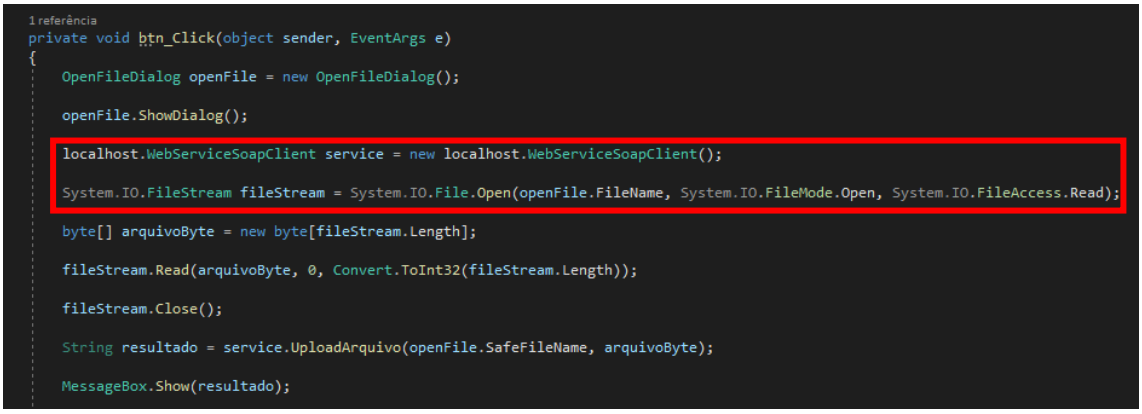


Foi programado o evento do botão quando ele for acionado. O controle OpenFileDialog é usado para localizar e selecionar arquivos do computador e será aberto pelo método ShowDialog.

```
1 referência
private void btn_Click(object sender, EventArgs e)
{
    OpenFileDialog openFile = new OpenFileDialog();

    openFile.ShowDialog();

    localhost.WebServiceSoapClient service = new localhost.WebServiceSoapClient();

    System.IO.FileStream fileStream = System.IO.File.Open(openFile.FileName, System.IO.FileMode.Open, System.IO.FileAccess.Read);

    byte[] arquivoByte = new byte[fileStream.Length];

    fileStream.Read(arquivoByte, 0, Convert.ToInt32(fileStream.Length));

    fileStream.Close();

    String resultado = service.UploadArquivo(openFile.SafeFileName, arquivoByte);

    MessageBox.Show(resultado);
```

O namespace do Web Service foi definido como localhost, então foi referenciado para fazer a chamada e ser utilizado.

Open(String, FileMode, FileAccess) abre um FileStream no caminho especificado, com o modo e o acesso especificados sem compartimento. O primeiro parâmetro será o arquivo a ser aberto, o segundo parâmetro será um valor FileMode que especifica se um arquivo deve ser criado caso não haja um e determina se o conteúdo dos arquivos existentes é mantido ou substituído e o terceiro parâmetro será um valor FileAccess que especifica as operações que podem ser executadas no arquivo.



```
1 referência
private void btn_Click(object sender, EventArgs e)
{
    OpenFileDialog openFile = new OpenFileDialog();

    openFile.ShowDialog();

    localhost.WebServiceSoapClient service = new localhost.WebServiceSoapClient();

    System.IO.FileStream fileStream = System.IO.File.Open(openFile.FileName, System.IO.FileMode.Open, System.IO.FileAccess.Read);

    byte[] arquivoByte = new byte[fileStream.Length];

    fileStream.Read(arquivoByte, 0, Convert.ToInt32(fileStream.Length));

    fileStream.Close();

    String resultado = service.UploadArquivo(openFile.SafeFileName, arquivoByte);

    MessageBox.Show(resultado);
```

O método FileStream.Read(Byte[], Int32, Int32) lê um bloco de bytes do fluxo e grava os dados em um buffer específico. O primeiro parâmetro contém a matriz de bytes especificada com valores entre offset e (offset + count – 1) substituídos pelos bytes lidos da fonte atual, o segundo parâmetro será o offset que é o deslocamento de bytes no array no qual os bytes lidos serão colocados, e o terceiro parâmetro será o count que será o número máximo de bytes a serem lidos.

```
1 referência
private void btn_Click(object sender, EventArgs e)
{
    OpenFileDialog openFile = new OpenFileDialog();

    openFile.ShowDialog();

    localhost.WebServiceSoapClient service = new localhost.WebServiceSoapClient();

    System.IO.FileStream fileStream = System.IO.File.Open(openFile.FileName, System.IO.FileMode.Open, System.IO.FileAccess.Read);

    byte[] arquivoByte = new byte[fileStream.Length];

    fileStream.Read(arquivoByte, 0, Convert.ToInt32(fileStream.Length));

    fileStream.Close();

    String resultado = service.UploadArquivo(openFile.SafeFileName, arquivoByte);

    MessageBox.Show(resultado);
```

Através do método UploadArquivo definida na Web Service, será mostrada se a operação foi bem-sucedida ou não por meio de uma caixa de mensagem.

```
1 referência
private void btn_Click(object sender, EventArgs e)
{
    OpenFileDialog openFile = new OpenFileDialog();

    openFile.ShowDialog();

    localhost.WebServiceSoapClient service = new localhost.WebServiceSoapClient();

    System.IO.FileStream fileStream = System.IO.File.Open(openFile.FileName, System.IO.FileMode.Open, System.IO.FileAccess.Read);

    byte[] arquivoByte = new byte[fileStream.Length];

    fileStream.Read(arquivoByte, 0, Convert.ToInt32(fileStream.Length));

    fileStream.Close();

    String resultado = service.UploadArquivo(openFile.SafeFileName, arquivoByte);

    MessageBox.Show(resultado);
```

A primeira ação depois do upload bem-sucedido será apresentar o nome do arquivo que foi feito o upload, pegando a informação em openFile.SafeFileName e colocando na primeira label na aplicação nomeada como lblMsg1.

```
lblMsg1.Text = $"Nome do Arquivo: {openFile.SafeFileName}";
lblMsg1.Visible = true;
```

A segunda ação será apresentar o tamanho do arquivo que foi feito o upload, pegando a informação em arquivoByte.Length que retornará o valor em Bytes, então foi utilizada condições para mostrar o tamanho em B, Kb ou em MB e colocando na segunda label na aplicação nomeada como lblMsg2.

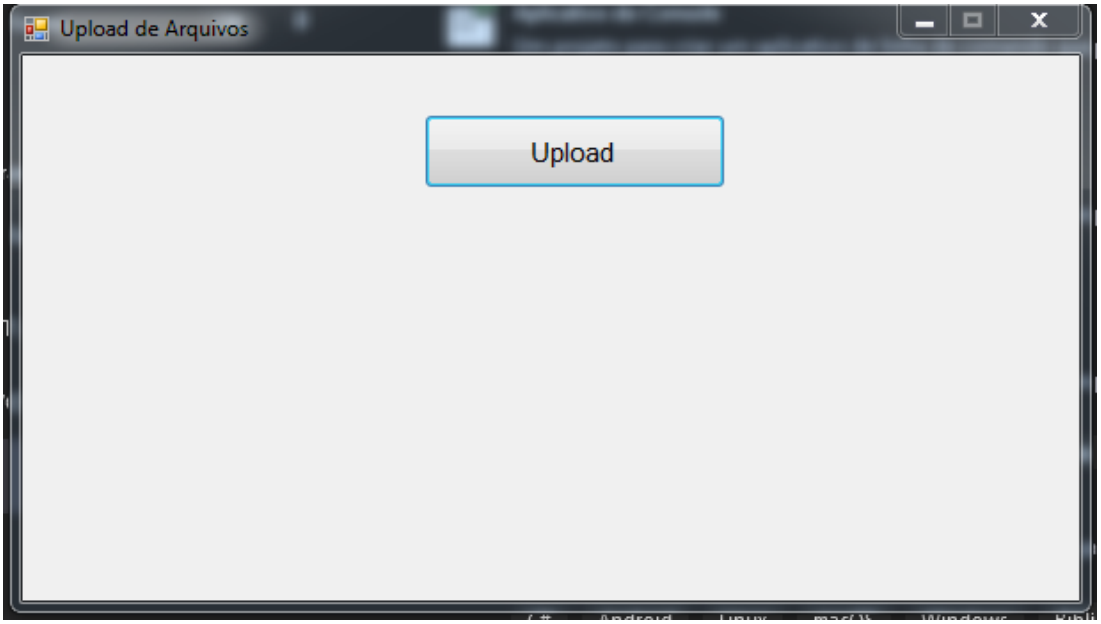
```
if (arquivoByte.Length <= 999)
{
    lblMsg2.Text = $"Tamanho do Arquivo: {arquivoByte.Length}B";
    lblMsg2.Visible = true;
}
else if (arquivoByte.Length <= 999999)
{
    lblMsg2.Text = $"Tamanho do Arquivo: {(arquivoByte.Length) / 1000}kB";
    lblMsg2.Visible = true;
}
else
{
    lblMsg2.Text = $"Tamanho do Arquivo: {(arquivoByte.Length) / 1000000}MB";
    lblMsg2.Visible = true;
}
```

A segunda ação será apresentar o código HASH (sha256) do arquivo que foi feito o upload, essa função não foi implementada 100%, que será implementada posteriormente, contudo está retornando o código HASH no nome do arquivo e não do conteúdo do arquivo.

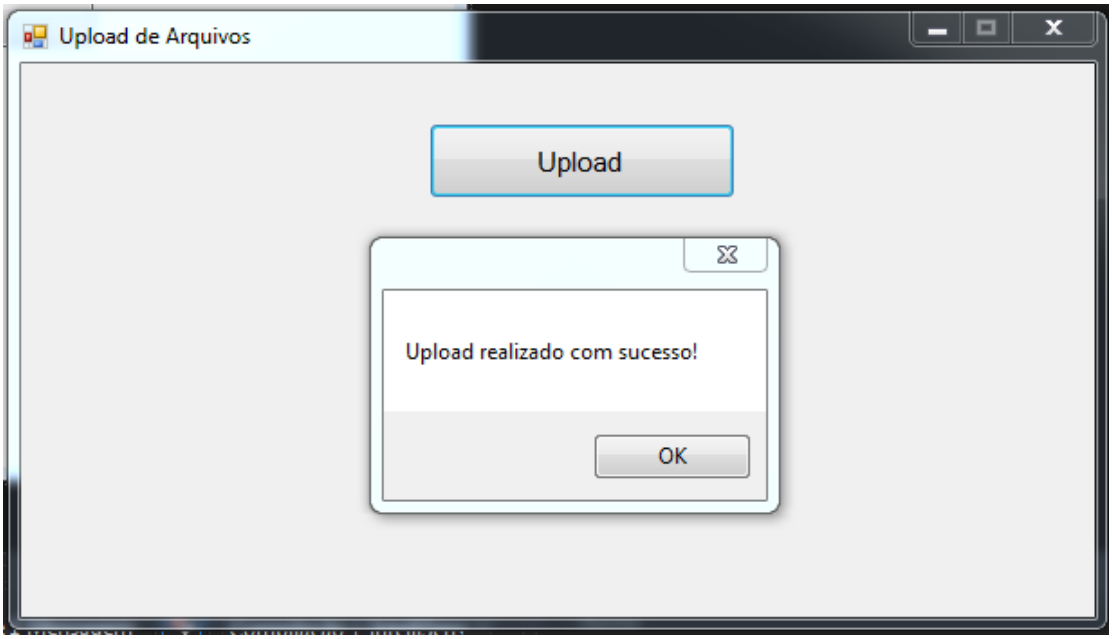
```
byte[] bytes = Encoding.UTF8.GetBytes(openFile.SafeFileName);
SHA256Managed hashstring = new SHA256Managed();
byte[] hash = hashstring.ComputeHash(bytes);
string hashString = string.Empty;
foreach (byte x in hash)
{
    hashString += String.Format("{0:x2}", x);
}
lblMsg3.Text = $"Código HASH: {hashString}";
lblMsg3.Visible = true;
```

4) Execução da Aplicação

1 – Ao clicar o botão de upload mostrará o explorador de arquivos, onde o usuário deve selecionar o arquivo para ser *upado*;



2 – Ser o upload for bem-sucedido aparecerá uma caixa de mensagem acusando a confirmação;



3 – Será mostrado na tela da aplicação o nome do arquivo que foi feito o upload, seu tamanho e seu código HASH.