



Relatório

Programação Orientada a Objetos

Projeto Prático

Alunos:
Allan Sales Aleluia, a21990
Francisco Moreira Rebêlo, a16443
Rui Alexandre Borlido Magalhães, a22300

Docente:
Luís Ferreira

Escola Superior de Tecnologia
Licenciatura em Engenharia de Sistemas Informáticos - PL

Barcelos, 27 de dezembro de 2022

Índice

1. Introdução ao projeto – BusIPCA.....	6
2. Motivação	7
2.1. Pessoa	7
2.2. Local.....	8
2.3. Passe	8
2.4. Passes.....	9
2.5. Usuário	9
2.6. Program.....	10
2.7. Exportar listaPasses para ficheiro binário	10
2.8. Exception de verificação do número de telemóvel	11
3. Documentação de código	12
4. Conclusão.....	13

Índice de Imagens

Figura 1 - Exemplo do funcionamento das conexões.....	6
Figura 2 - Classe Pessoa.....	7
Figura 3 - Classe Local	8
Figura 4 - Classe Passe.....	8
Figura 5 - Classe Passes	9
Figura 6 - Classe Usuário	9
Figura 7 - Classe Program	10
Figura 8 - Este é o método que exporta a lista listaPasses para um ficheiro binário	10
Figura 9 - Esta exception vai verificar se o número de telemóvel tem 9 dígitos.....	11
Figura 10 - Documentação de código	12

1. Introdução ao projeto – BusIPCA

A BusIPCA é uma aplicação móvel que visa facilitar a utilização dos autocarros disponibilizados pelo IPCA. Nela será possível fazer aquisição de bilhetes, criação e carregamento de passes. Funções que hoje são feitas de maneira presencial e necessitam do deslocamento do utilizador até postos onde se possam fazer tais ações.

O tanto bilhete quanto passe iram gerar um QRCode, facilitando também a utilização e ajudando a preservar o meio ambiente com menos descarte de papel.

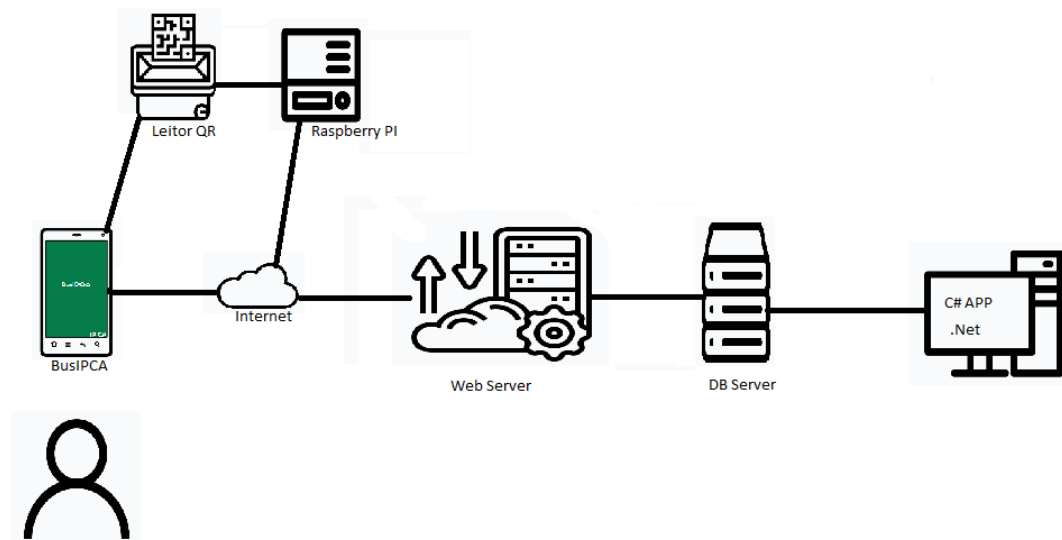
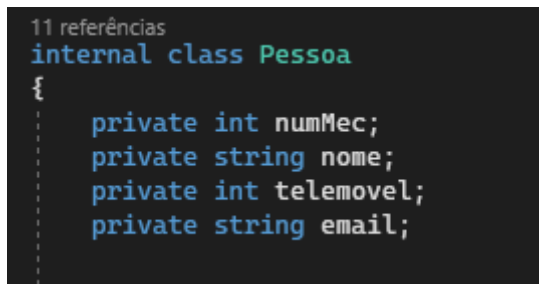


Figura 1 - Exemplo do funcionamento das conexões

2. Motivação

Nesta segunda fase, o objetivo era aprimorar o código do nosso projeto através de exceptions e novas classes que nos permitam uma implementar novas funcionalidades desde a entrega da primeira fase até ao momento.

2.1. Pessoa



```
11 referências
internal class Pessoa
{
    private int numMec;
    private string nome;
    private int telemovel;
    private string email;
```

Figura 2 - Classe Pessoa

A classe pessoa armazena a informação sobre os utilizadores da aplicação.

O numMec corresponde ao número mecanográfico do utilizador (aluno/professor)

São também armazenados o nome, telemóvel e email do utilizador na mesma classe.

2.2. Local

```
3 referências
internal class Local
{
    private string nome;
    private int codigo;
    List<Local> listaDeParagens = new List<Local>();
}
```

Figura 3 - Classe Local

A classe Local vai armazenar a informação sobre as paragens.

Tem Get e Set como propriedades para permitir que elementos públicos possam aceder aos valores armazenados na classe privada.

Tem como construtores o construtor por definição e o construtor que recebe todos os valores que correspondem às variáveis da classe Local.

2.3. Passe

```
16 referências
internal class Passe
{
    /// <summary>
    /// atributos da classe passe
    /// </summary>
    #region ATRIBUTOS
    private Usuario usuPasse; // atributo que utiliza a classe usuario para acessar os dados do passe
    private int numero;
    private int numeroAluno;
    private string nome;
    private bool status;
    private int ultimoNumero;
    private List<Passe> listaDePASSES;
}
```

Figura 4 - Classe Passe

A classe passe armazena a informação sobre os passes dos utilizadores da aplicação.

O usuPasse é o atributo que utiliza a classe usuario para aceder aos dados do passe.

Os restantes atributos (numero, numeroAluno, nome, status, ultimoNumero) são armazenados no array listaDePASSES.

2.4. Passes

```
8 referências
internal class Passes
{
    #region ATRIBUTOS
    private int idPasse;
    private List<Passes> listaPasses;
    private Passe dadosPasse;
    private int totalPasses;
    private int ultimoIdPasse;
    #endregion
}
```

Figura 5 - Classe Passes

A classe passes armazena e gera a lista de passes na aplicação.

São gerados posteriormente a lista de opções de passes na aplicação.

2.5. Usuário

```
7 referências
internal class Usuario : Pessoa
{
    #region ATRIBUTOS
    private Passe atribuirPasse;
}
```

Figura 6 - Classe Usuário

A classe Usuário é uma classe filha da classe Pessoa. Nela existe a mecânica da atribuição um passe a uma pessoa.

2.6. Program

```
namespace BusIPCA
{
    internal class Program
    {
        static void Main(string[] args)
        {
            //Cria um utilizador
            Usuario u1 = new Usuario(21990, "Allan", 999999999, "email@xpto");

            //Faz a atribuição de um passe a um utilizador
            u1.AtribuirPasse = new Passe(1, 1, 21990, "Allan", false, 0);
        }
    }
}
```

Figura 7 - Classe Program

A classe program, é o nosso "main". Nela são criadas as pessoas e os passes com os seus respetivos valores. Nela também é chamado o método que aloca um passe a um utilizador.

2.7. Exportar listaPasses para ficheiro binário

```
/// salvando a lista de passes em ficheiro binário
public bool SaveAll(string docpasses)
{
    try
    {
        Stream stream = File.Open(docpasses, FileMode.OpenOrCreate);
        BinaryFormatter bin = new BinaryFormatter();
        bin.Serialize(stream, listaPasses);
        stream.Close();
        return true;
    }
    catch (IOException e)
    {
        throw e;
    }
}
```

Figura 8 - Este é o método que exporta a lista listaPasses para um ficheiro binário

2.8. Exception de verificação do número de telemóvel

```
public int Telemovel
{
    get { return telemovel; }
    set
    {
        //Exception que verifica se o numero de telemóvel tem 9 dígitos
        try
        {
            if (value.ToString().Length != 9)
            {
                throw new Exception(message: "O número de telemóvel tem de ter 9 dígitos!");
            }
            this.telemovel = value;
        }
        catch (Exception ex)
        {
            //Se não
            Console.WriteLine(ex.Message);
        }
    }
}
```

Figura 9 - Esta exception vai verificar se o número de telemóvel tem 9 dígitos

3. Documentação de código

O nosso código foi documentado com recurso à ferramenta DoxyGen onde pode ser consultada toda a informação resumida sobre as classes e os seus respetivos métodos. Essa informação pode ser encontrada na pasta "Documentação".

BusIPCA.Local Class Reference
Classe LOCAL - Para criação das paragens More...
Public Member Functions
Local (string nome, int id) Construtor que recebe por parâmetro os valores de Local More...
Local () Construtor por defeito
Properties
string Nome [get, set]
int Codigo [get, set]
Private Attributes
string nome
int codigo
List< Local > ListaDeParagens = new List< Local >()
Detailed Description
Classe LOCAL - Para criação das paragens

Figura 10 - Documentação de código

4. Conclusão

Com a falta de tempo por parte dos elementos do grupo e falta de prática de escrever código levou a que não fosse implementado o sistema de camadas, pois o nosso código não abrangia toda a base de dados que vai ser criada posteriormente para o projeto de base de dados. Sabemos que o correto para aplicação das camadas é que cada camada seja um projeto diferente, onde as bibliotecas(dll) estarão reunidas e através delas será feita a comunicação entre as camadas, sem que uma aceda a outras diretamente.