INTRODUÇÃO À PROGRAMAÇÃO

Projeto de Linguagens de Programação

Licenciatura em Sistemas de Informação para a Gestão

2022/2023

PROJETO FINAL

Grupo LSIG_EF_10

Avelino Almeida – 8220800

Índice

Lista de Siglas e Acrónimos

IP	Introdução à Programação
EF	Época Final
API	Application Programming Interface
JRE	Java Runtime Environment
JDK	Java Development Kit
OOP	Object Oriented Programming
IDE	Integrated Development Environment
JSE	Java Standard Edition

1. Introdução

Em suma: Neste capítulo pretende-se a apresentação e descrição do trabalho que irá ser efetuado.

1.1. Apresentação do tema de trabalho

Figura 1 - Radio Control

Neste trabalho foi proposto como tema principal desenvolver um software de pequena/média dimensão como elemento essencial do processo de aprendizagem individual com o foco num upgrade de uma aplicação já antiga, utilizada por uma loja de **Radio Control** que simplifique o atual registo e processamento manual de encomendas e venda de veículos e peças de substituição.

A aplicação deveria permitir, entre outras funcionalidades, o registo, alteração e eliminação de dados referente a cada cliente, a criação de encomendas para os produtos disponíveis na loja, a gestão de stock de produtos e peças, e a criação de relatórios de vendas e stock disponível.

1.2. Objetivos e motivação

Para alcançar os objetivos do ponto anterior, foi-nos desafiado a utilização a linguagem de programação Java, no seguimento da UC introdução à programação lecionada no 1º semestre. O meu desafio pessoal para este projeto passou upgrade da aplicação permitindo

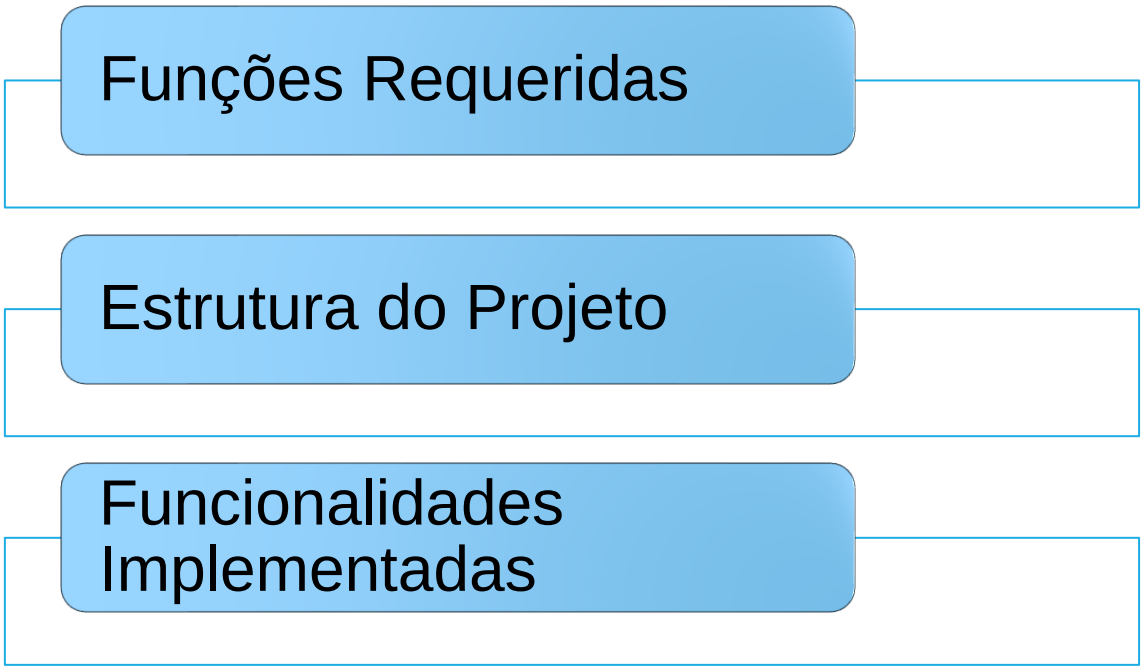
a simplificação e automatização de processos importantes para a loja de Radio Control, melhorando a eficiência e a precisão das operações realizadas.

1.3. Resultados esperados

Foram criadas classes para representar cada entidade relevante para a aplicação, como clientes, produtos, encomendas e componentes e foram implementadas as funcionalidades e atributos necessários para o registo de produtos e serviços.

O sistema foi testado e as operações que antes eram realizadas manualmente, agora são automatizadas, proporcionando maior eficiência e agilidade para a loja. Além disso, a geração de relatórios de vendas e stock agora é rápida e intuitiva, permitindo uma melhor tomada de decisão futura para a loja.

1.4. Estruturação do documento



2. Funcionalidades requeridas

Descrição da codificação do conjunto de funcionalidades (e correspondentes estruturas de dados) solicitadas.

Uma empresa de comercialização de veículos de radio control pretende que se desenvolva um sistema de gestão que simplifique o atual registo e processamento manual de encomendas e venda de veículos e peças de substituição.

A aplicação deverá permitir, entre outras funcionalidades:

- Registo
- Alteração
- Eliminação de dados referentes a cada cliente:
    - Código de cliente
    - Nome
    - Morada
    - NIF
    - País.

Por outro lado, deverá também ser possível

- **Registar, alterar, eliminar encomendas.**

Deve aplicação gerir uma base de dados de veículos com informações técnicas de componentes associados. Uma encomenda pode incluir um ou mais veículos completos ou componentes de substituição.

A aplicação deve facilitar a gestão das encomendas de peças de substituição permitindo pesquisar pelo componente ou pelo veículo, apresentando a lista dos componentes que o compõem. Após as encomendas serem realizadas, tanto pelos clientes quanto pelos funcionários, o stock dos veículos deverá atualizado. O administrador poderá adicionar peças ao sistema, simulando a aquisição de novos produtos para venda. Deverá existir sempre peças de substituição disponíveis.

3. Funcionalidades propostas

Descrição dos objetivos gerais das funcionalidades propostas pelo grupo, bem como da descrição da sua codificação (e correspondentes estruturas de dados).

Perfil Administrador

1. Gestão de Clientes
   1.1. Deverá ser possível criar, editar e remover clientes.
   1.2. Apenas deverá ser possível remover clientes que não tenham encomendas realizadas.
   1.3. Aos clientes que já efetuaram encomendas apenas deverá ser possível a alteração de estado para inativo.
2. **Gestão de Produtos**
   2.1. Deverá ser possível editar e remover produtos a comercializar.
   2.2. A informação a armazenar sobre os produtos compreende o código, nome, preço e componentes a usar (descrição e quantidade).
   2.3. Note que a remoção de um artigo apenas deverá ser possível se não tiver sido feita nenhuma encomenda previamente deste produto, e neste caso apenas deverá ser possível a alteração de estado para inativo.
3. **Perfil Cliente**
   3.1. Registo de uma encomenda
   3.2. Aplicação deverá ser capaz de registar uma encomenda de produtos para um determinado cliente identificando a data de entrega.
4. **Persistência de dados**
   4.1. Aplicação deverá permitir guardar/carregar dados em/de ficheiro, permitindo persisti-los ao longo de diferentes utilizações, de forma a possibilitar a leitura e gravação a qualquer momento através da respetiva escolha no menu de opções.
   4.2. A leitura dos dados de ficheiro implica a substituição da informação na memoria central do computador, enquanto a gravação para ficheiro deverá substituir a informação até então armazenada neste.
   4.3. A manipulação de ficheiros apenas deverá acontecer quando o utilizador escolher a opção de ler/gravar, todas as operações devem ser feitas com a informação carregada em memória.
5. **Listagens propostas**
   5.1. Deve ainda propor e implementar, no mínimo, 3 listagens/relatórios.
   5.2. Estas listagens/relatórios devem ser do interesse do cliente ou da empresa.
   5.3. O principal objetivo é avaliar a compreensão do problema bem como a capacidade de analisar os dados armazenados.

4. Estrutura analítica do projeto

Descrição dos objetivos gerais das funcionalidades propostas pelo grupo, bem como da descrição da sua codificação (e correspondentes estruturas de dados).

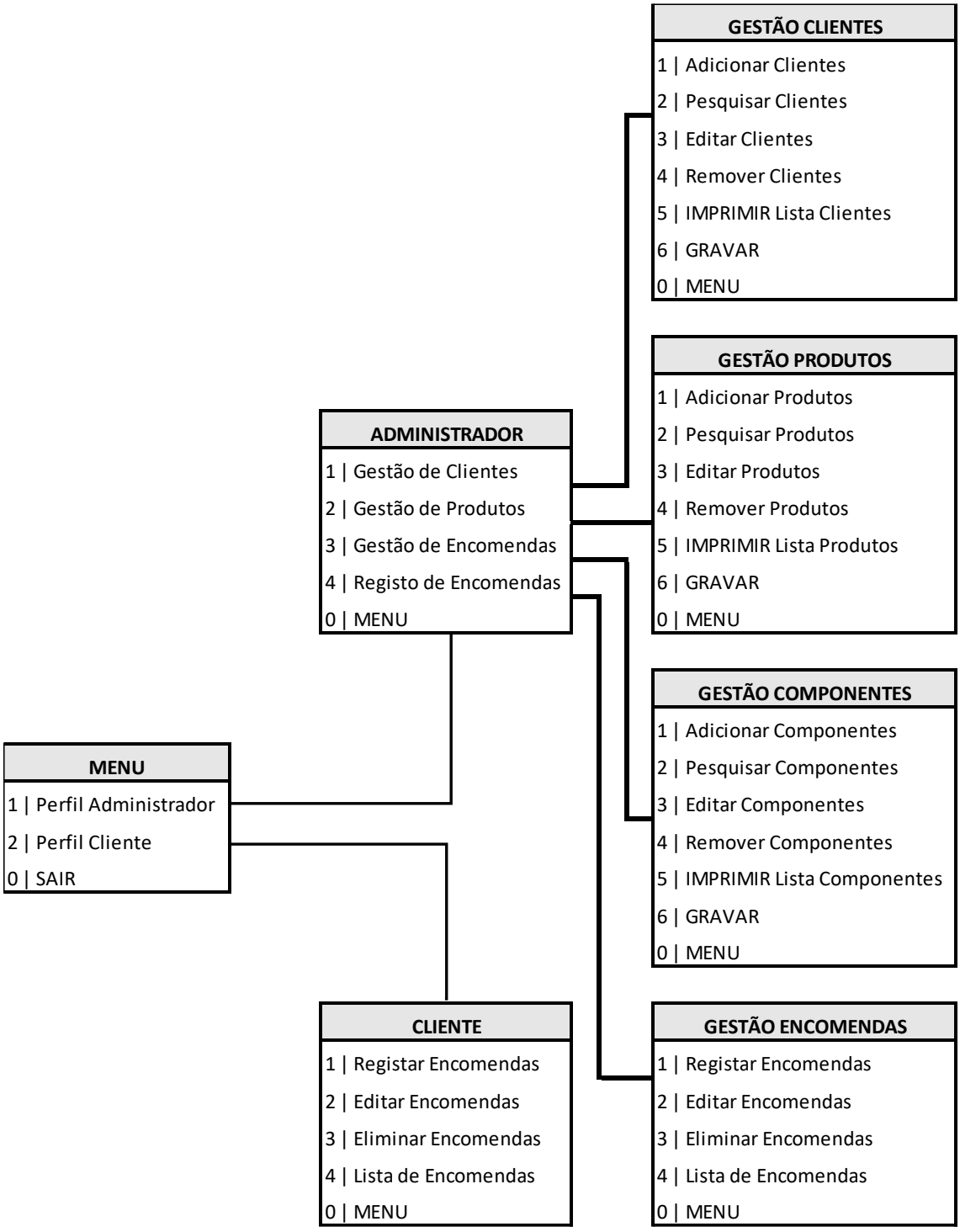


Figura 2 - Funcionalidades Propostas

5. Funcionalidades implementadas

Descrição das funcionalidades implementadas em função das funcionalidades propostas inicialmente. Devem ser indicadas as decisões tomadas no decorrer do desenvolvimento do projeto de modo a justificar o modo de execução da aplicação desenvolvida.

ESTRUTURAS DE REPETIÇÃO E MENUS

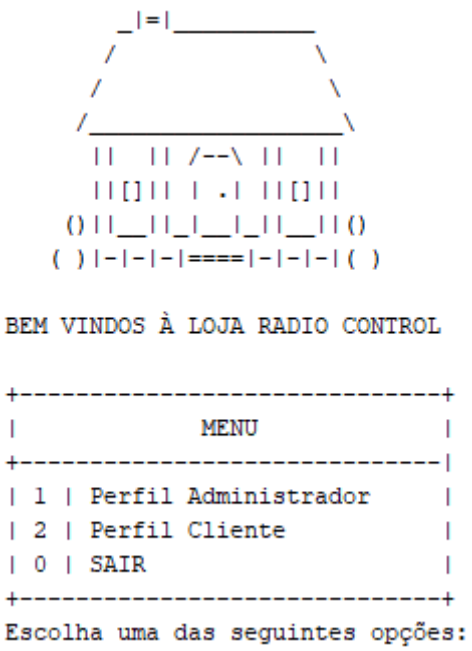


Figura 3 – Menu

Na figura 3 temos uma representação gráfica de um menu de opções para o utilizador. É um menu simples com 3 opções: "Perfil Administrador", "Perfil Cliente" e o "SAIR". O menu é apresentado no console do IDE com a ajuda do método *println*, que imprime uma *string* na tela e passa para a próxima linha. Depois do menu ser exibido, o programa pede para o utilizador escolher uma nova opção, através do método *print*.

O utilizador insere sua escolha e o programa continua a partir daí, dependendo da opção escolhida. Neste caso passamos para a opcção1

Funcionalidades do perfil Administrador

Exemplos:

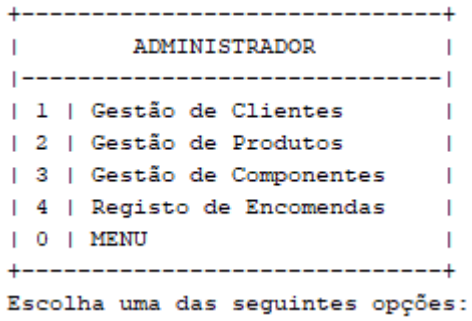


Figura 4 - Menu Administrador

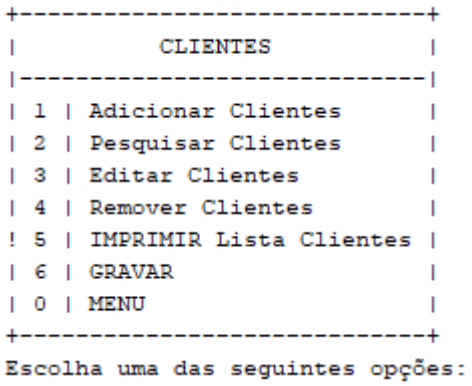


Figura 5 - Menu Clientes

No perfil de Administrador, tem-se acesso completo a todas as funcionalidades do sistema, incluindo a **gestão de clientes, produtos, componentes e registo de encomendas**. Isso permite, entre outras coisas, imprimir vários relatórios com todos os dados de clientes, produtos ou componentes.

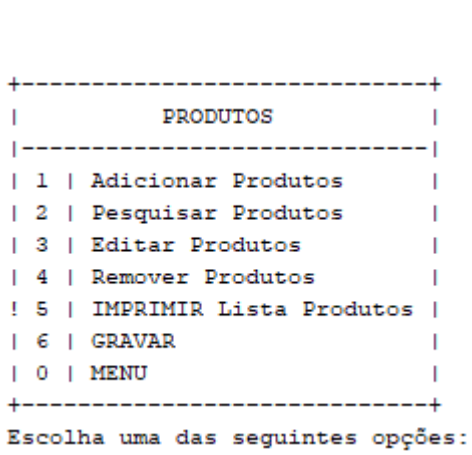


Figura 7 - Menu Produtos

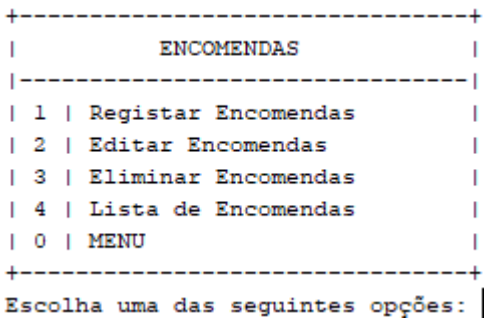


Figura 6 - Menu Encomendas

Linguagem Java

ARRAYS E MATRIZES

Começamos por criar uma instância da classe "*ListaClientes*" e inicializamos seu atributo "l_*clientes*" com um array de tamanho "*TAMANHO_INICIAL_CLIENTES*". O atributo "n_*clientes*" é inicializado com 0. Em seguida, a função "*lerFicheiroClientes*" é chamada passando "*listaclientes*" como parâmetro. Essa função provavelmente irá ler um arquivo de texto ou outra fonte de dados para preencher a lista de clientes.

```
ListaClientes listaclientes = new ListaClientes();
listaclientes.l_clientes = new Cliente[TAMANHO_INICIAL_CLIENTES];
listaclientes.n_clientes = 0;
lerFicheiroClientes(listaclientes);

ListaProdutos listaprodutos = new ListaProdutos();
listaprodutos.l_produtos = new Produto[TAMANHO_INICIAL_PRODUTOS];
listaprodutos.n_produtos = 0;
lerFicheiroProdutos(listaprodutos);

ListaComponentes listacomponentes = new ListaComponentes();
listacomponentes.l_componentes = new Componente[TAMANHO_INICIAL_COMPONENTES];
listacomponentes.n_componentes = 0;
lerFicheiroComponentes(listacomponentes);

ListaEncomendas listaencomendas= new ListaEncomendas();
listaencomendas.l_encomendas = new Encomenda[TAMANHO_INICIAL_ENCOMENDAS];
listaencomendas.n_encomendas= 0;
lerFicheiroEncomendas(listaencomendas);

Scanner sc = new Scanner(System.in);
int menu;
do {
    System.out.println("");
    System.out.println("        _|=|_________");
    System.out.println("       /              \\\");
    System.out.println("      /                \\\");
    System.out.println("     /__________________\\\");
    System.out.println("      ||  || /--\\ ||  ||");
    System.out.println("      ||[]|| | .| ||[]||");
    System.out.println("    ()||__||_|__|_||__||()");
    System.out.println("   ( )|-|-|-|====|-|-|-|( ) ");
    System.out.println("");
    System.out.println("BEM VINDOS À LOJA RADIO CONTROL\n");
    System.out.println("+------------------------------+");
    System.out.println("|             MENU             |");
    System.out.println("+------------------------------|");
    System.out.println("| 1 | Perfil Administrador     |");
    System.out.println("| 2 | Perfil Cliente           |");
    System.out.println("| 0 | SAIR                     |");
    System.out.println("+------------------------------+");
    System.out.print("Escolha uma das seguintes opções: ");
    menu = sc.nextInt();
    System.out.println("\n");
    switch (menu) {
```

Figura 8 - Linguagem Java

Gestão de Clientes

Cada opção é implementada como um case na estrutura de seleção switch. Quando o usuário seleciona uma opção, o código correspondente é executado. Por exemplo, se o usuário selecionar a opção 1, o código para adicionar um novo cliente será executado. Se o usuário selecionar a opção 2, o código para pesquisar clientes será executado, e assim por diante. Se a opção selecionada for inválida, será exibida uma mensagem de erro.

```
case 1:
    int menuclientes;
    do {
        System.out.println("+-----------------------------+");
        System.out.println("|       GESTÃO CLIENTES       |");
        System.out.println("|-----------------------------|");
        System.out.println("| 1 | Adicionar Clientes      |");
        System.out.println("| 2 | Pesquisar Clientes      |");
        System.out.println("| 3 | Editar Clientes         |");
        System.out.println("| 4 | Remover Clientes        |");
        System.out.println("! 5 | IMPRIMIR Lista Clientes |");
        System.out.println("| 6 | GRAVAR                  |");
        System.out.println("| 0 | MENU                    |");
        System.out.println("+-----------------------------+");
        System.out.print("Escolha uma das seguintes opções: ");
        menuclientes =sc.nextInt();
        System.out.println("\n");
        sc.nextLine();
        switch (menuclientes) {
            case 1:
                try {
                    System.out.println("----- ADICIONAR CLIENTES -----");
                    Cliente temp = new Cliente();
                    temp.id = nCliente++;
                    boolean nifExiste = true;
                    while (nifExiste) {
                        System.out.println("ID: "+temp.id);
                        System.out.print("NIF: ");
                        try {
                            temp.NIF = Integer.parseInt(sc.nextLine());
                            nifExiste = verificarNIF(listaclientes, temp.NIF);
                            if(nifExiste) {
                                System.out.println("NIF já existente. Por favor insira outro.");
                            }
                        } catch (NumberFormatException e) {
                            System.out.println("Insira apenas números para o NIF.");
                        }
                    }
                    System.out.print("Nome: ");
                    temp.nome = sc.nextLine()   ;
                    System.out.print("Morada: ");
                    temp.morada = sc.nextLine();
                    System.out.print("Pais: ");
                    temp.pais = sc.nextLine();
                    if (adicionarCliente(listaclientes, temp)){
                        System.out.println("Cliente adicionado com sucesso!");
                    }
                } catch (Exception e) {
                }
                break;
```

Figura 9 - Gestão de Clientes(1)

```
case 2:
    System.out.println("----- PESQUISAR CLIENTES ----- ");
    System.out.print("Inserir o NIF do cliente: ");
    int nif = sc.nextInt();
    sc.nextLine();
    localizarCliente(listaclientes, nif);
    try {
        System.out.println("Pressione 'Enter' para voltar ao MENU");
        System.in.read();
    } catch (IOException e) {
    }
    break;
case 3:
    System.out.println("----- EDITAR CLIENTES ----- ");
    editarClientes(sc, listaclientes);
    break;
case 4:
    System.out.println("----- REMOVER CLIENTES -----");
    System.out.print("Inserir NIF do Cliente: ");
    int idRemover = sc.nextInt();
    removerCliente(listaclientes, idRemover);
    break;
case 5:
    System.out.println("----- LISTA DE CLIENTES ----- ");
    imprimirClientes(listaclientes);
    break;
case 6:
    System.out.println("----- GRAVAR FICHEIRO -----");
    criarFicheiroClientes(listaclientes);
    System.out.println("As alteracoes foram gravadas com sucesso!!");
    break;
case 0:
    System.out.println("----- SAIR -----");
    System.out.println("MENU");
    break;
default:
    System.out.println("Opção inválida!");
```

Figura 10 - Gestão de Clientes(1)

Gestão de Produtos e Componentes

O seguinte código apresenta as funcionalidades propostas de gestão de produtos e componentes nesta aplicação. Permite adicionar, pesquisar, editar, remover e listar produtos. Além disso, ele também permite a criação de um ficheiro com a lista de produtos. As opções são exibidas num menu de escolha e o utilizador pode selecionar a ação desejada. Cada opção é implementada como uma opção (switch case) e é executada quando selecionada. A aplicação também tem tratamento de exceções para garantir a integridade dos dados.

```
int menuprodutos;
do {
    System.out.println("+-----------------------------+");
    System.out.println("|          PRODUTOS           |");
    System.out.println("|-----------------------------|");
    System.out.println("| 1 | Adicionar Produtos      |");
    System.out.println("| 2 | Pesquisar Produtos      |");
    System.out.println("| 3 | Editar Produtos         |");
    System.out.println("| 4 | Remover Produtos        |");
    System.out.println("! 5 | IMPRIMIR Lista Produtos |");
    System.out.println("| 6 | GRAVAR                  |");
    System.out.println("| 0 | MENU                    |");
    System.out.println("+-----------------------------+");
    System.out.print("Escolha uma das seguintes opções: ");
    menuprodutos = sc.nextInt();
    System.out.println("\n");
    switch (menuprodutos) {
        case 1:

                System.out.println("----- ADICIONAR PRODUTOS -----");
                Produto temp = new Produto();
                temp.codigo = nProduto++;
                sc.nextLine();
                System.out.println("ID: "+temp.codigo);
                System.out.print("Nome: ");
                temp.nome = sc.nextLine()   ;
                System.out.print("Dimensões(CxLxA): ");
                temp.dimensoes = sc.nextLine();
                System.out.print("Preço: ");
                temp.preco = sc.nextInt();
                sc.nextLine();
                System.out.print("Tipo: ");
                temp.tipo = sc.nextLine();
                System.out.print("Quantidade: ");
                temp.quantidade = sc.nextInt();
                System.out.println("");
                System.out.print("Deseja inserir componentes? (1 - sim, 0 - não): ");
                menuprodutos = sc.nextInt();
                sc.nextLine();
                Componente comp = new Componente();
                int menucomponentes = 1;
                    try{
                        while (menucomponentes == 1) {
                            System.out.println("----- ADICIONAR COMPONENTES -----");
                            comp.cod_componente = nComponente++;
                            System.out.println("ID: "+comp.cod_componente);
                            System.out.print("Descrição: ");
                            comp.descricao = sc.nextLine()   ;
                            System.out.print("Quantidade: ");
                            comp.quantidade = sc.nextInt();
                            sc.nextLine();
                            System.out.print("Unidade: ");
                            comp.unidade = sc.nextLine();
                            System.out.println("Componente adicionado com sucesso!\n");
                            System.out.print("Deseja inserir mais componentes? (1 - sim, 0 - não): ");
                            menucomponentes = sc.nextInt();
                            sc.nextLine();
                            if (adicionar_componente(listaprodutos, menucomponentes, comp)){
                                System.out.println("Componente adicionado com sucesso!");
                            }
```

Figura 11 - Gestão de Produtos(1)

```
case 2:
    System.out.println("----- PESQUISAR PRODUTOS ----- ");
    System.out.print("Inserir o codigo do produto: ");
    int codigo = sc.nextInt();
    sc.nextLine();
    localizarProdutos(listaprodutos, codigo);
    try {
        System.out.println("Pressione 'Enter' para voltar ao MENU");
        System.in.read();
    } catch (IOException e) {
    }
    break;
case 3:
    System.out.println("----- EDITAR PRODUTOS ----- ");
    editarProdutos(sc, listaprodutos);
    break;
case 4:
    System.out.println("----- REMOVER PRODUTOS -----");
    System.out.print("Inserir Codigo do Produto: ");
    int codigoRemover = sc.nextInt();
    removerProdutos(listaprodutos, codigoRemover);
    break;
case 5:
    System.out.println("----- LISTA DE PRODUTOS ----- ");
    imprimirProdutos(listaprodutos);
    break;
case 6:
    System.out.println("----- GRAVAR FICHEIRO -----");
    criarFicheiroProdutos(listaprodutos);
    System.out.println("As alteracoes foram gravadas com sucesso!!");
    break;
case 0:
    System.out.println("MENU");
    break;
default:
    System.out.println("Opção inválida!");
```

Figura 12 - Gestão de Produtos(2)

```
int menucomponentes;
do {
    System.out.println("+--------------------------------+");
    System.out.println("|          COMPONENTES           |");
    System.out.println("|--------------------------------|");
    System.out.println("| 1 | Adicionar Componentes      |");
    System.out.println("| 2 | Pesquisar Componentes      |");
    System.out.println("| 3 | Editar Componentes         |");
    System.out.println("| 4 | Remover Componentes        |");
    System.out.println("! 5 | IMPRIMIR Lista Componentes |");
    System.out.println("| 6 | GRAVAR                     |");
    System.out.println("| 0 | MENU                       |");
    System.out.println("+--------------------------------+");
    System.out.print("Escolha uma das seguintes opções: ");
    menucomponentes = sc.nextInt();
```

Figura 13 - Gestão de Produtos(3)

```
switch (menucomponentes) {
    case 1:
        try {
            System.out.println("----- ADICIONAR COMPONENTES -----");
            Componente temp = new Componente();
            temp.cod_componente = nComponente++;
            sc.nextLine();
            System.out.println("ID: "+temp.cod_componente);
            System.out.print("Descrição: ");
            temp.descricao = sc.nextLine()   ;
            System.out.print("Quantidade: ");
            temp.quantidade = sc.nextInt();
            sc.nextLine();
            System.out.print("Unidade: ");
            temp.unidade = sc.nextLine();
            System.out.println("");
            if (adicionarComponentes(listacomponentes, temp)){
                System.out.println("Componente adicionado com sucesso!");
            }
        } catch (Exception e) {
        }
        break;
    case 2:
        System.out.println("----- PESQUISAR COMPONENTES ----- ");
        System.out.print("Inserir o codigo do componente: ");
        int codigo = sc.nextInt();
        sc.nextLine();
        localizarComponentes(listacomponentes, codigo);
        try {
            System.out.println("Pressione 'Enter' para voltar ao MENU");
            System.in.read();
        } catch (IOException e) {
        }
        break;
    case 3:
        System.out.println("----- EDITAR COMPONENTES ----- ");
        editarComponentes(sc, listacomponentes);
        break;
    case 4:
        System.out.println("----- REMOVER COMPONENTES -----");
        System.out.print("Inserir Codigo do Componente: ");
        int codigoRemover = sc.nextInt();
        removerComponentes(listacomponentes, codigoRemover);
        break;
    case 5:
        System.out.println("----- LISTA DE COMPONENTES ----- ");
        imprimirComponentes(listacomponentes);
        break;
    case 6:
        System.out.println("----- GRAVAR FICHEIRO -----");
        criarFicheiroComponentes(listacomponentes);
        System.out.println("As alteracoes foram gravadas com sucesso!!");
        break;
    case 0:
        System.out.println("MENU");
        break;
    default:
        System.out.println("Opção inválida!");
`
```

Figura 14 - Gestão de Produtos(4)

Registo de uma encomenda

(Esta função é partilhada de igual modo pelo funcionário da loja como pelo cliente se pretender realizar uma encomenda)

Nesta parte de código representa um sistema de gestão de encomendas. Exibe um menu com as opções para registar, editar, excluir e listar encomendas. O utilizador pode escolher uma dessas opções inserindo um número correspondente. O código verifica a escolha do usuário através do comando "switch" e realiza a ação correspondente. Por exemplo, se o usuário escolher a opção 1, o sistema registará uma nova encomenda. Se o usuário escolher a opção 4, o sistema exibirá a lista de encomendas existentes. O código continua a ser executado até que o usuário selecione a opção 0 para sair do menu de encomendas.

```
System.out.println("+--------------------------------+");
System.out.println("|          ENCOMENDAS            |");
System.out.println("|--------------------------------|");
System.out.println("| 1 | Registar Encomendas        |");
System.out.println("| 2 | Editar Encomendas          |");
System.out.println("| 3 | Eliminar Encomendas        |");
System.out.println("| 4 | Lista de Encomendas        |");
System.out.println("| 0 | MENU                       |");
System.out.println("+--------------------------------+");
System.out.print("Escolha uma das seguintes opções: ");
menuencomendas = sc.nextInt();
System.out.println("\n");
switch (menuencomendas) {
    case 1:
        System.out.println("----- REGISTAR ENCOMENDAS ----- ");
        try {
            Encomenda temp = new Encomenda();
            Produto produto = new Produto();
            temp.id_encomenda = nEncomenda++;
            System.out.print("Insira o NIF do cliente: ");
            temp.nif_cliente = sc.nextInt();
            sc.nextLine();
            adicionarEncomenda(listaencomendas, temp);
            int pos = pesquisarCliente(listaclientes, temp.nif_cliente);
            if (pos != -1) {
                int opcao = 1;
                while (opcao == 1) {
                System.out.print("Insira o ID Produto: ");
                produto.codigo = sc.nextInt();
                System.out.print("Inserir a quantidade pretendida: ");
                produto.quantidade = sc.nextInt();
                adicionarProdutosEncomenda(listaencomendas, pos, produto);
                System.out.println("Encomenda adicionada com sucesso!\n");
                System.out.print("Pretende inserir mais produtos? (1 - sim, 0 - não): ");
                opcao = sc.nextInt();
                sc.nextLine();
            }
            } else {
                System.out.println("O NIF introduzido não está registado no sistema");
            }
        } catch (Exception e) {
        }
        criarFicheiroEncomenda(listaencomendas);
        break;
```

Figura 15 - Gestão de Encomendas(1)

```
case 2:
    System.out.println("----- EDITAR ENCOMENDAS -----");
    editarEncomendas(sc, listaencomendas);
    break;
case 3:
    System.out.println("----- ELIMINAR ENCOMENDAS -----");
    System.out.print("Inserir ID da Encomenda: ");
    int idRemover = sc.nextInt();
    removerEncomendas(listaencomendas, idRemover);
    break;
case 4:
    System.out.println("----- LISTA ENCOMENDAS -----");
    imprimirEncomendas(listaencomendas);
    break;
case 0:
    System.out.println("MENU");
    break;
default:
    System.out.println("Opção inválida!");
```

Figura 16 - Gestão de Encomendas(2)

Persistência de dados

A persistência de dados é uma característica crucial em qualquer aplicação, pois permite que os dados sejam armazenados e mantidos mesmo após a finalização do programa. Em aplicações que trabalham com grandes quantidades de informação, a persistência de dados é ainda mais importante para garantir a integridade e a disponibilidade dos dados ao longo do tempo.

No código criado e apresentado neste trabalho, a persistência de dados é realizada através da criação de arquivos para armazenar informações sobre as encomendas e os clientes. Isso é feito ao chamar as funções "***criarFicheiroEncomenda***" e "***criarFicheiroCliente***", que criam arquivos no disco para armazenar as informações.

Ao iniciar a aplicação, os arquivos de encomendas, clientes e produtos são lidos e os dados são carregados em listas para serem manipulados na memória. Quando o utilizador realiza ações que modificam as informações, como adicionar, editar ou remover encomendas, essas alterações são refletidas nas listas em memória e, posteriormente, salvas nos arquivos de disco.

Lista de Clientes

```
CLIENTE;1;11111;Avelino Almeida;Penafiel;Portugal;Ativo
CLIENTE;2;22222;Santiago Almeida;Penafiel;Portugal;Ativo
CLIENTE;3;33333;Matilde Almeida;Penafiel;Portugal;Ativo
CLIENTE;4;44444;Susana Mendes;Penafiel;Portugal;Ativo
```

Figura 17 - clientes.txt

Lista de Produtos

```
1  PRODUTO;1;Traxxas Carro Sledge 1/8 Truggy 4WD 6S BL Red;80x40x40;809;Carro;10;ativo
```

Figura 18 - produtos.txt

Lista de Componentes

```
COMPONENTE;1;motor sem escova Traxxas 2000kV;1;UN
COMPONENTE;2;roda alto desempenho;4;UN
COMPONENTE;3;Controle de vecolidade VXL-6s;1;UN
COMPONENTE;4;Guia rápido;1;UN
COMPONENTE;5;Sistema de rádio TQi 2.4GHz;1;UN
COMPONENTE;6;Kit ferramentas;1;UN
```

Figura 19 - componentes.txt

Lista de Encomendas

```
ENCOMENDA:1;11111;10
ENCOMENDA:2;123;10
ENCOMENDA:3;123456789;0
```

Figura 20 - encomendas.txt

6. Conclusão

Quaisquer considerações finais que o grupo julgue pertinentes para avaliação, nomeadamente tarefas não implementadas e o motivo válido para a sua não conclusão.

Na conclusão deste projeto possibilitou criar sistema utilizaria técnicas de persistência de dados para armazenar as informações de forma segura e acessível, garantindo a continuidade do gestão do mesmo após o encerramento da aplicação. Foi sem dúvida, o meu principal foco neste trabalho.

Esta aplicação apresentada foi desenvolvida com base na análise de todos os conteúdos lecionados durante o semestre tendo por objetivo a implementação das seguintes funcionalidades: Arrays, estruturas de controlo, funções e métodos estruturas de repetição, consola com input_output, estruturas de repetição e menus, tipos estruturados de dados, registos e ficheiros e por fim documentação

Principais Destaques

- Gestão e manipulação de listas clientes, produtos e componentes (totalmente funcional)
- Validação de dados através de conhecimentos adquiridos na UC
- Interface gráfica (menus)

Propostas de melhoria

- Melhoria significativa na gestão das encomendas (gestão de stocks)
- Criar listas conjuntas
- Introdução de matrizes

Referências

(s.d.). Obtido de w3schools: https://www.w3schools.com/

(s.d.). Obtido de Java Downloads: https://www.oracle.com/java/technologies/downloads/

Coelho, P. (s.d.). *Programação Java.*

ESTG - Escola Superior de Tecnologia e Gestão. (s.d.). Obtido de https://www.estg.ipp.pt

netbeans. (s.d.). Obtido de https://netbeans.apache.org/

Submissão de Trabalho

Expresso a minha gratidão à Escola Superior de Tecnologia e Gestão e à direção do curso de Sistemas de Informação para a Gestão por nos oferecerem a oportunidade valiosa de realizar estes projetos práticos, que são uma fonte enriquecedora de aprendizado para todos nós. Agradecemos pelo investimento no nosso desenvolvimento pessoal e profissional.

Gostaria de expressar meu sincero agradecimento ao professor João Ramos pela sua dedicação incansável em transmitir os seus conhecimentos técnicos, pelo tempo dedicado e paciência para nos ajudar fora do horário letivo.

Felgueiras, 29 de janeiro de 2023

Relatório[1] - Grupo LSIG_EF_10

Avelino Almeida – 8220800

[1] Época Final