Projecto POO 2024
POO Financial Systems

Vasco Guilherme Alves, nº 2022228207
Miguel Angelo Silva, nº 2023221244

Programação Orientada a Objectos — LEI 2024

Conteúdo

1 Introdução **2**

2 Manual do Utilizador **2**

3 Funcionalidades em Detalhe **3**

4 Mapa de Classes **3**

5 UML **4**

6 Conclusão e Observações **5**

1 Introdução

O presente trabalho propõe o desenvolvimento de uma aplicação de gestão financeira, designada *POO Financial Services* (POOFS), com o objetivo de permitir a uma empresa registar e emitir faturas aos seus clientes. Estas faturas devem conter produtos de dois tipos: alimentares e de farmácia, que diferem nas suas categorias, certificações, se são prescritos e nos cálculos da taxa IVA.

Por fim, a aplicação deverá oferecer uma interface eficiente e intuitiva, permitindo aos utilizadores a inserção e consulta de dados, bem como a exportação e importação através de ficheiros.

2 Manual do Utilizador

Início do Programa

Para começar a usar o programa:

1. Certifique-se de que os ficheiros `dados.ser` ou `dados-iniciais.txt` estão no mesmo diretório do programa (caso existam).
2. Compile e execute o ficheiro `POOFS.java`.
3. O programa irá carregar os dados guardados anteriormente (se disponíveis) e mostrar o menu principal.

Menu Principal

O menu principal oferece várias opções que cobrem as principais funcionalidades da aplicação:

1. **Criar/Editar Cliente**: Cria ou edita informações de clientes.
2. **Listar Clientes**: Mostra todos os clientes registados.
3. **Criar/Editar Fatura**: Cria ou atualiza faturas associadas a clientes.
4. **Listar Faturas**: Exibe todas as faturas emitidas.
5. **Consultar Fatura**: Permite ver os detalhes de uma fatura específica.
6. **Importar Faturas**: Carrega dados de clientes e faturas a partir de ficheiros externos.
7. **Exportar Faturas**: Grava os dados de clientes num ficheiro de texto.
8. **Estatísticas**: Apresenta dados gerais sobre clientes, faturas e produtos.
9. **Sair**: Guarda os dados e fecha o programa.

3 Funcionalidades em Detalhe

Criar/Editar Cliente — Mostra um menu que permite criar ou editar clientes. Se escolher criar um cliente novo vai pedir o nome do cliente e as restantes informações. Caso já exista um cliente com o nome inserido permite ao utilizador escolher editar esse cliente. Se escolher a segunda opção, editar, será mostrada um menu com todos os clientes onde pode escolher qual editar.

Listar Clientes — É impressa a listagem de todos os clientes.

Criar/Editar Fatura — Mostra um menu onde escolhe-se o cliente a que diz respeito a fatura, depois o utilizador pode escolher entre criar uma fatura nova para esse cliente ou editar uma das faturas já existentes. No mesmo modo que a primeira opção, o utilizador pode escolher editar uma fatura que já existe se ao criar inserir um ID que já existe. Ao editar a fatura, podemos também adicionar e remover produtos novos.

Listar Faturas — São apresentadas todas as faturas de todos os clientes registados, incluindo toda a informação que diz respeito aos produtos, incluindo o IVA, o total da fatura, a ainda informação sobre o cliente e a data.

Consultar Fatura — Permite consultar uma fatura especifica- Mostra todas as informações de uma fatura, incluindo o total com e sem IVA.

Importar Faturas — Carrega dados de um ficheiro indicado pelo utilizador, em formato de texto ou serializado.

Exportar Faturas — Permite gravar os dados de clientes num ficheiro para facilitar backups e consultas externas.

Estatísticas — Apresenta o total de clientes registados, o número de faturas emitidas, os produtos listados e os valores dos produtos faturados (com e sem IVA).

Sair — Guarda os dados no ficheiro `dados.ser` e termina o programa.

4 Mapa de Classes

Estrutura Geral (*Packages*)

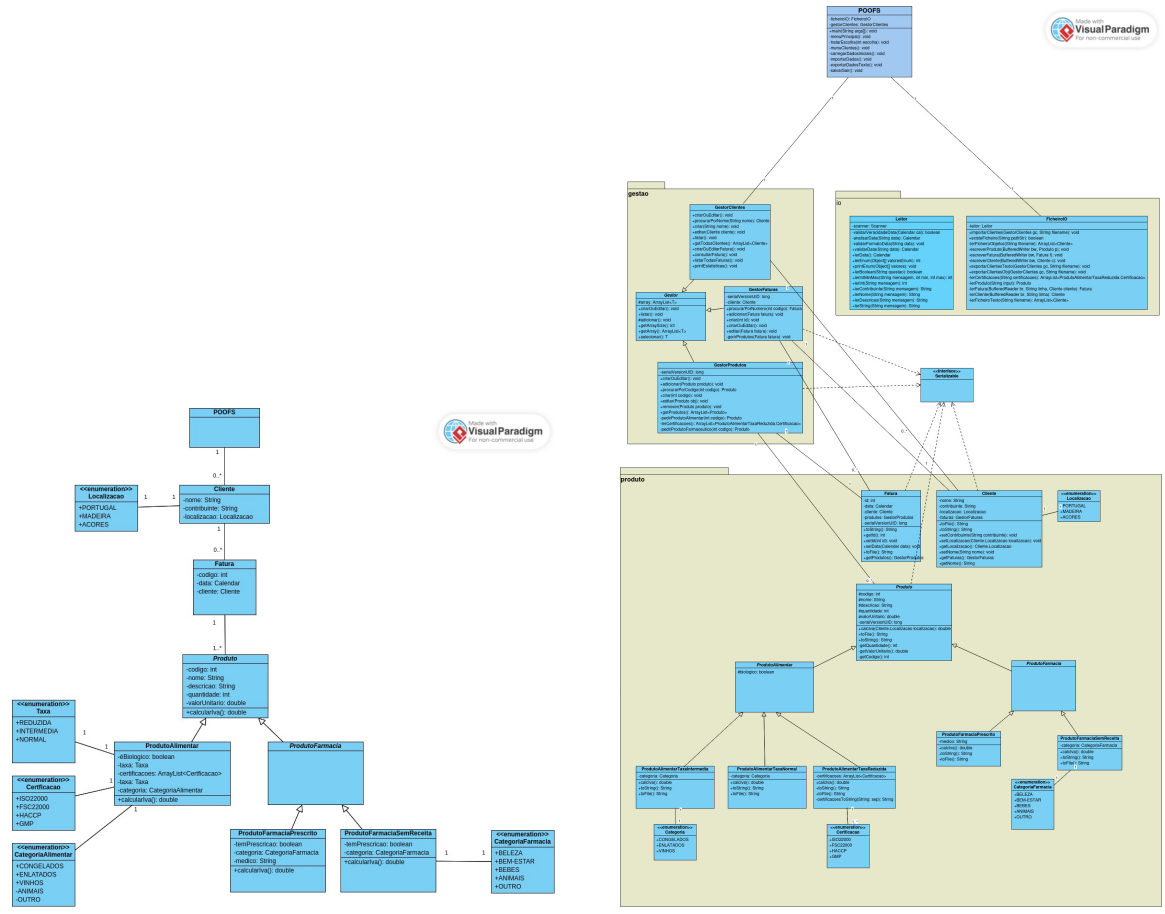
O programa está dividido em três pacotes principais:

- **produto**: Contém classes relacionadas aos clientes, faturas e todos os tipos produtos.
- **gestao**: Responsável pela gestão de clientes, faturas e produtos. Isto inclui a criação, edição, remoção e a listagem dos itens de cada cada gestor.
- **io**: Inclui classes utilitárias para entrada e validação de dados do utilizador tal como a importação e exportação de ficheiros.

Descrição das Classes

- **POOFS**: Classe principal que gere o menu e as operações principais.
- **Cliente**: Armazena informações de clientes e as suas faturas.
- **Fatura**: Representa uma fatura emitida, com uma lista de produtos.
- **Produto**: Classe genérica para produtos.
- **ProdutoAlimentar**: Subclasse de `Produto`, dividida em categorias baseadas na taxa de IVA.
- **ProdutoFarmacia**: Subclasse de `Produto` para produtos farmacêuticos.
- **GestorClientes**, **GestorFaturas** e **GestorProdutos**: Gerem a lógica das respetivas entidades.
- **Leitor**: Facilita a entrada e validação de dados pelo utilizador.
- **FicheiroIO**: Gere importação e exportação de dados.

5 UML



(a) UML Projetado (b) UML do Projecto Final

Figura 1: UML Projetado Vs. UML Final

O UML de classes final, têm 17 classes, 4 das quais são `abstract`, comparado com o diagrama projectado que tinha apenas 7 classes, sendo duas `abstract`. Este aumento em

complexidade deve-se sobretudo à criação dos pacotes `gestão` e `io` para auxiliar, respectivamente, com a gestão (criação, edição, listagem, remoção, etc.) de dados e com a validação, importação e exportação de dados.

Mesmo assim, a estrutura original foi preservada na sua maioria dentro do pacote `produto`. A classe `produtoAlimentar` passou a ser `abstract` para dar lugar à criação de três subclasses, cada uma correspondente a um tipo de taxamento (reduzido, intermédio ou normal).

6 Conclusão e Observações

O programa **POOFS** (*POO Financial Services*) foi concebido para ajudar empresas a organizar as suas operações financeiras. Ele regista clientes, cria faturas, e oferece dados detalhados de forma clara e prática.

Principais Benefícios:

- Menu intuitivo que permite gerir clientes e faturas com facilidade tal como obter informações especificas.
- Suporte para produtos alimentares e farmacêuticos, com cálculos automáticos de IVA.
- Exportação e importação de dados para segurança e partilha.

Considerações Finais

- O sistema tem uma arquitetura bem estruturada, que facilita manutenção e melhorias futuras.