

**UNIVERSIDADE POLITÉCNICA**

**A POLITÉCNICA**

Escola Superior de Gestão, Ciências e Tecnologias.

**Engenharia Informática e de Telecomunicações**

**Disciplina:** Programação II

|  |  |
| --- | --- |
| **Discente:** | **Código:** |
| Amílcar Venâncio Mula | 379725 |
| Jose Marjane | 333121 |
| Edson Angelo Ngoenha | 446520 |
| Klisman Nhampule |  |

**Docente:**

segunda-feira, 28 de outubro de 2019

**Índice**

[Introdução 3](#_Toc23142368)

[Objectivos 4](#_Toc23142369)

[Objectivo geral 4](#_Toc23142370)

[Objectivo específico 4](#_Toc23142371)

[Metodologias 4](#_Toc23142372)

[Descrição do produto Metodologias 4](#_Toc23142373)

[Contexto de negócio 5](#_Toc23142374)

[Objetivos do produto 5](#_Toc23142375)

[Modelos do sistema 5](#_Toc23142376)

[Modelo conceitual 5](#_Toc23142377)

[Contextual 6](#_Toc23142378)

[Diagramação de classes 7](#_Toc23142379)

[Diagramação generalizada de classes com herança 7](#_Toc23142380)

[Modelo orientado a dados dos processos de registro 8](#_Toc23142381)

[Anexos 9](#_Toc23142382)

[Anexos I 9](#_Toc23142383)

[Anexos II 10](#_Toc23142384)

[Anexos III 13](#_Toc23142385)

# Introdução

Neste trabalho apresentaremos um sistema protótipo de ERP para gestão de pessoal(RH) e vendas para pequenas empresas que pretendam informatizar o seu negocio e optimizar os processos comerciais (maior controle).

Pretendemos também, através deste, aplicar conceitos anteriormente aprendidos em logica de programação e programação I, tais como os relacionados com POO e API’S java.

Será especificado aqui também detalhes do produto como: e o que ele é, qual é a sua finalidade, como é o seu funcionamento, a quem ele é destinado, fazer estimativas de uso de rentabilidade, etc.

# Objectivos

## Objectivo geral

Consolidar conceitos anteriormente aprendidos, e usar este trabalho como base de investigação de conceitos adicionais com relevância a bagagem do estudante como um futuro programador profissional assim, como engenheiro informático em diversas ares.

## Objectivo específico

Aplicar todos conhecimentos de programação na elaboração deste pequeno projecto com vista a desenvolver um produto de software tipo ERP(mini) para o caso de negocio de gestão e vendas. Para tal forma revisados conceitos como:

* Estruturas de controle de erros e excepções;
* Estruturas de Pacotes
* Classes abstratas;
* Estruturas de dados (Arrays e Vectores);
* Conceitos de orientação a objectos (Classes e Objectos, Métodos, encapsulamento e herança);
* Ficheiros;
* API’S.

# Metodologias

Na realização do presente trabalho tivemos como cominho para as nossas pesquisas manuais como apostilas e livros de programação em java, algumas pesquisas na internet e alguns dos conhecimentos retidos nas cadeiras anteriores relacionadas a programação.

# Descrição do produto Metodologias

Esta será uma aplicação para gestão de uma empresa de pequeno porte que faz a venda de produtos alimentares e pretende para o seu negocio um sistema para fazer o registro de vendas, encomendas, registro de fornecedores dos produtos alimentares, categorização dos produtos, registro de funcionários e sua divisão por departamentos, registro de clientes, transportadores das encomendas a serem entregues, etc…

# Contexto de negócio

Este produto de software terá aplicação a estabelecimentos de vendas de produtos alimentares. É um produto de software genérico que será vendido a quem quer que esteja interessado nele.

# Objetivos do produto

Apos estudo realizado na área de vendas, encontram-se pontos que levam a necessidade de criação de um sistema que auxiliara e optimizar o processo de vendas. Como tal, tornando mais eficaz e seguro a efetuação de transações comercias entre clientes e vendedores.

O sistema devera focar-se nos seguintes aspectos:

1. Cada produto tem um nome e um preço unitário;
2. Os produtos são divididos em categorias;
3. Produtos são fornecidos por outras empresas. Para cada fornecedor deve ser mantida informações acima de tudo, para seu contacto;
4. Os funcionários da empresa devem constar na base de dados. Os vendedores são um caso partículas, pois deve-se guardar informações relativas as encomendas que cada um faz;
5. Os clientes da empresa fazem varias encomendas. Os seus dados de contacto devem ser acessíveis;
6. Uma encomenda contem vários produtos, cada qual numa determinada quantidade e um determinado desconto. A encomenda é efetuada numa determinada data e deve conter registro de envio para o cliente.

# Modelos do sistema

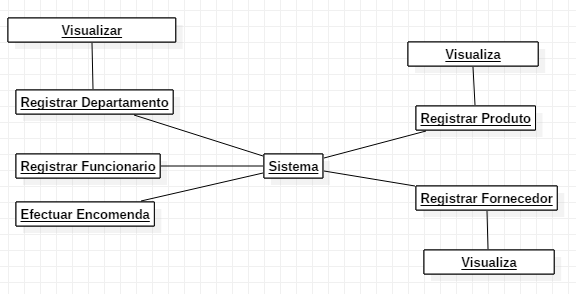
## Modelo conceitual

O sistema devera fazer registro de dados sobre as seguintes classes:

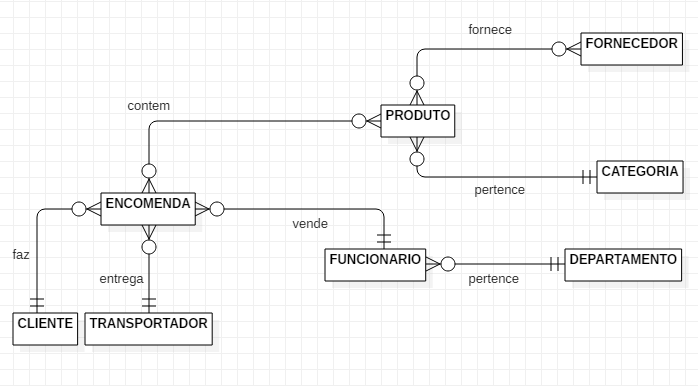
|  |  |  |  |
| --- | --- | --- | --- |
| Produto: | Categoria: | Departamento: | Funcionário |
| Código | Código | Código | Código |
| Nome | Tipo | Nome | Nome |
| Validade |  |  | Departamento |
| Categoria |  |  |  |
| Outros |  |  | **Outros** |

|  |  |  |  |
| --- | --- | --- | --- |
| Cliente: | Fornecedor: | Transportador: | Encomenda: |
| Código | Código | Código | Código |
| Nome | Nome | Nome | Código funcionário |
| Endereço | Endereço | Endereço | Código cliente |
| Contacto | Pais | Contacto | Código transportador |
|  | Contacto |  | Estado de envio |
|  |  |  | Lista e produtos |
| Outros | **Outros** | **Outros** | **Outros** |

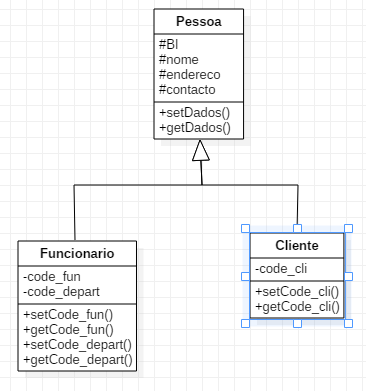
## Contextual



## Diagramação de classes



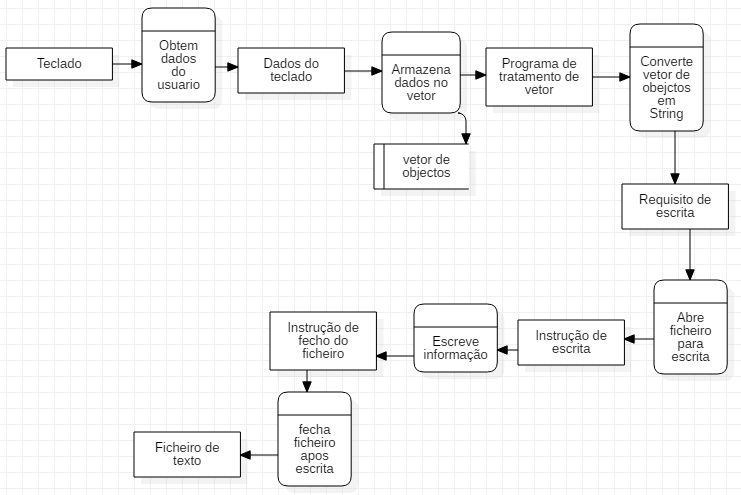
## Diagramação generalizada de classes com herança



## Modelo orientado a dados dos processos de registro

Tendo em conta que todas actividade de registro tem mesmo procedimento ate o momento que os dados são salvos no ficheiro, aqui ilustraremos um exemplo base desse registro.

O processo de registro consiste em fazer leitura dos dados via teclado e repassa-los as respetivas classes de encapsulamento, onde todos dados serão armazenados numa variável do tipo objecto de classe. Assim, sendo todos objectos de classe são armazenados em um vector de objectos, onde para seu posterior armazenamento do ficheiro de texto, este vector é tratado de tal forma que o condicione para que como texto os dados sejam salvos.



# Anexos

## Anexos I

O sistema é organizado em pacotes de modo que as entidades (classes) de forma a deixa explicita as características de cada classe. Assim temos os seguintes pacotes:

Departamento, encomenda, fornecedor, funcionário, de entrada e saída de dados para ficheiros, de produtos e de clientes.

|  |  |
| --- | --- |
|  |  |

## Anexos II

public class Pessoa {

protected String BI;

protected String nome;

protected String endereco;

protected String contacto;

public String getBI() {

return BI;

}

public void setBI(String BI) {

this.BI = BI;

}

…

}

public interface ControladorFunc {

public abstract Funcionarioo coletaDados();

public abstract void registraDados();

}

public class Funcionarioo extends Pessoa implements ControladorFunc{

private int code\_fun;

private int code\_depart;

private int getConde\_fun() {

return code\_fun;

}

private void setConde\_fun(int conde\_fun) {

this.code\_fun = conde\_fun;

}

private int getConde\_depart() {

return code\_depart;

}

private void setConde\_depart(int conde\_depart) {

this.code\_depart = conde\_depart;

}

/\*

@Override

public String toString() {

return "Funcionario {" + "BI: " + BI + ", Nome: " + nome +

", Endereco: " + endereco + ", Contacto: "+contacto+

", Cogido de Departamento: "+code\_depart+", Codigo de Funcionario: "+code\_fun+"}";

}

\*/

@Override

public String toString() {

return BI + "," + nome +"," + endereco + ","+contacto+ ","+code\_depart+","+code\_fun;

}

@Override

public Funcionarioo coletaDados() {

this.setBI(JOptionPane.showInputDialog(null, "Introduza o N.BI", "Registro", JOptionPane.INFORMATION\_MESSAGE));

…

return this;

}

@Override

public void registraDados() {

try {

VectorDeDados v = new VectorDeDados(2);

v.adiciona(coletaDados());

JOptionPane.showMessageDialog(null, "Registro com Sucesso!", "Registro", JOptionPane.INFORMATION\_MESSAGE);

System.out.println(v);

Escrita es = new Escrita();

es.regFuncionario(v.toString());

} catch (IOException ex) {

Logger.getLogger(Funcionarioo.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

## Anexos III

Sistema:

