2023-01

Professor:	RADAMÉS PEREIRA

Atributos de uma boa especificação de Projeto de Software:

- Clareza
- Não Ambígua
- Completa
- Simples
- Bem escrita

UNIVERSIDADE COMUNITÁRIA REGIONAL DE CHAPECÓ - UNOCHAPECÓ
ÁREA DE CIÊNCIAS EXATAS E AMBIENTAIS
SISTEMAS DE INFORMAÇÃO

Estudo de caso:
DELÍCIAS GOURMET

Equipe: **Heitor Zatti**

Introdução

1.1 Resumo do Projeto

Estudo de Caso: Sistema da Delícias Gourmet. Em uma indústria alimentícia chamada Delícias Gourmet, o controle de materiais é um processo-chave para garantir a qualidade e a eficiência dos produtos finais. Nesta história, vamos explorar como diferentes atores desempenham funções importantes em seus respectivos setores, interagindo e colaborando para manter o sistema de controle de materiais funcionando sem problemas. A história começa com João, o gerente de compras, que é responsável por adquirir os ingredientes e os materiais necessários para a produção dos alimentos. Ele trabalha em estreita colaboração com os fornecedores para garantir a qualidade dos produtos e negocia os melhores preços para reduzir os custos de produção. Maria, a supervisora de estoque, trabalha em conjunto com João. Ela é responsável por gerenciar o armazenamento dos materiais e garantir que todos os itens estejam devidamente organizados e estocados. Quando os ingredientes e materiais chegam ao armazém, Maria confere e verifica se tudo está em conformidade com os pedidos feitos por João. Dentro da fábrica, Pedro, o coordenador de produção, coordena a utilização dos ingredientes e materiais no processo de fabricação. Ele colabora com Maria para garantir que os itens necessários sejam entregues no momento certo e na quantidade adequada. Pedro também trabalha com os líderes de cada linha de produção para planejar e otimizar a utilização dos materiais. Na área de controle de qualidade, Ana, a analista de qualidade, monitora o processo de produção, verificando se todos os ingredientes e materiais estão dentro dos padrões exigidos. Ela também é responsável por conduzir inspeções regulares e garantir que a produção esteja em conformidade com as normas de segurança alimentar e regulamentações governamentais. Por fim, a história se completa com Carlos, o responsável pelas vendas e distribuição dos produtos. Ele coordena a entrega dos produtos acabados aos clientes e mantém um registro de todas as vendas realizadas. Carlos também fornece informações sobre a demanda do mercado e as preferências dos clientes, o que ajuda João na seleção de fornecedores e ingredientes para futuras compras. Esta história ilustra como os diferentes atores em uma indústria alimentícia colaboram e desempenham funções cruciais para manter o sistema manual de controle de materiais funcionando de forma eficiente. A interdependência entre as atividades de João, Maria, Pedro, Ana e Carlos é fundamental para garantir que a Delícias Gourmet; mantenha sua reputação de qualidade e satisfação do cliente.

1.2 Plataforma de desenvolvimento

Para o desenvolvimento da ferramenta usaremos o vscode com a extensão PlantUml

1.3 Plataforma de operação

Sistema de Gestão de Compras

Sistema de Gestão de Estoque

Automação da Linha de Produção

Sistema de Controle de Qualidade

Sistema de Gestão de Vendas e Distribuição

1.4 Definições e siglas

ER: Especificação de Requisitos

Delícias Gourmet: Nome fictício da indústria alimentícia em questão

João: Nome fictício do gerente de compras

Maria: Nome fictício da supervisora de estoque

Pedro: Nome fictício do coordenador de produção

Ana: Nome fictício da analista de qualidade

Carlos: Nome fictício do responsável pelas vendas e distribuição

1.5 Perspectiva do produto

1.5.1 Modos de operação

O desenvolvimento é feito em front-end pois está gerando formas gráficas e imagens prontas para o consumidor

1.5.2 Requisitos de adaptação ao ambiente

Definem-se aqui possíveis requisitos de adaptação do produto aos ambientes particulares onde ele será implantado. Por exemplo, parâmetros e métodos de configuração requeridos para ambientes específicos devem ser descritos aqui.

Número de ordem	Requisito	Detalhes
1	Configuração de ticket de venda e da Nota Fiscal Eletrônica	Configuração dos campos de formulário com interface responsiva.

1.6 Funções do produto

Fornecer facilidade de uso no ambiente de trabalho, atendendo o objetivo principal no que ele foi desenvolvido

1.7 Características dos usuários

Indicado para a melhor usabilidade de funcionários, que é quem fará o uso direto da ferramenta. Nesse caso de uso, cozinheiros, engenheiros de alimentos e de produção.

1.8 Restrições

Restrições legais relacionadas à segurança alimentar e regulamentações governamentais podem limitar as opções dos desenvolvedores ao implementar o sistema de controle de materiais.

Regulamentações específicas sobre rotulagem, armazenamento e transporte de alimentos podem impor restrições na concepção do sistema.

1.9 Hipóteses de trabalho

A versão do ambiente operacional, como um sistema operacional específico, pode exigir modificações na ER para garantir a compatibilidade e a integração corretas do sistema de controle de materiais.

A escolha da plataforma de desenvolvimento, como uma linguagem de programação ou um framework específico, também pode afetar a especificação dos requisitos do sistema.

2 Requisitos específicos

2.1 Interfaces externas

2.1.1 Visão geral

Entradas: Pedidos de compra de ingredientes e materiais, informações sobre fornecedores, dados de estoque, requisitos de qualidade e segurança alimentar.

Saídas: Relatórios de compras, atualizações de estoque, programação de produção, relatórios de controle de qualidade, registros de vendas e distribuição, informações sobre demanda do mercado e preferências dos clientes.

2.1.2 Requisitos para interfaces gráficas de usuário

Sugere-se, no caso de interfaces gráficas, a inclusão dos seguintes elementos:

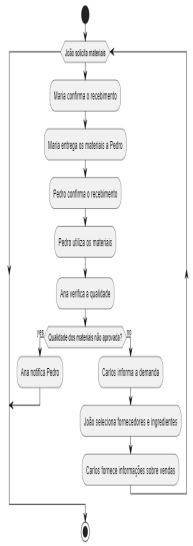
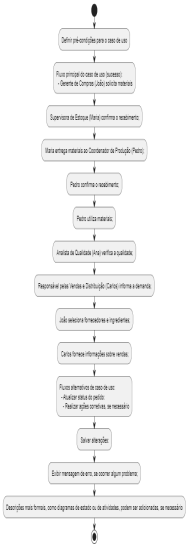
- Um esboço do layout gráfico sugerido para a interface;
- Uma descrição dos relacionamentos com outras interfaces;
- Um diagrama de estados/atividades, caso necessário para melhor entender-se o comportamento requerido da interface;
- Uma lista dos campos de dados da interface;
- Uma lista dos comandos da interface;
- BPM;

2.2 Requisitos funcionais

2.2.1 Diagramas de casos de uso

Incluir todos os casos de uso que se pretende implementar em uma liberação. Pode-se incluir ainda: um certo caso de uso e seus relacionamentos, todos os casos de uso para um certo ator.

2.2.2 Fluxos dos casos de uso

<table>
<tr>
<td> 
</td>
<td> 
</td>
</tr>
<tr>
<td>@startuml
start
:Definir pré-condições para o caso de uso;
:Fluxo principal do caso de uso (sucesso):
- Gerente de Compras (João) solicita materiais;
- Supervisora de Estoque (Maria) confirma o recebimento;
- Maria entrega materiais ao Coordenador de Produção (Pedro);
- Pedro confirma o recebimento;
- Pedro utiliza materiais;
- Analista de Qualidade (Ana) verifica a qualidade;
- Responsável pelas Vendas e Distribuição (Carlos) informa a demanda;
- João seleciona fornecedores e ingredientes;
- Carlos fornece informações sobre vendas;
:Fluxos alternativos de caso de uso:
- Atualizar status do pedido:
- Realizar ações corretivas, se necessário;
- Salvar alterações:
- Exibir mensagem de erro, se ocorrer algum problema;
:Descrições mais formais, como diagramas de estado ou de atividades, podem ser adicionadas, se necessário;
-right->[#black]-down->[#black]-left->[#black]-up->[#black]-stop</td>
<td>@startuml

left to right direction

actor João as "Gerente de Compras"
actor Maria as "Supervisora de Estoque"
actor Pedro as "Coordenador de Produção"
actor Ana as "Analista de Qualidade"
actor Carlos as "Responsável pelas Vendas e Distribuição"

rectangle "Sistema de Controle de Materiais na Indústria Alimentícia" {
João --> Maria: Solicitação de materiais
Maria <-- João: Confirmação de recebimento
Maria --> Pedro: Entrega de materiais
Pedro <-- Maria: Confirmação de recebimento
Pedro --> Ana: Utilização de materiais
Ana <-- Pedro: Verificação da qualidade
Carlos --> Pedro: Demanda do mercado
João --> Carlos: Seleção de fornecedores e ingredientes
Carlos <-- João: Informações sobre vendas
}

rectangle "Delícias</td>
</tr>
</table>

@enduml	Gourmet" {
	actor Usuario as "Funcionário"
	
	rectangle "Gerenciar Pedidos" {
	Usuario --> (1. Selecionar opção 'Gerenciar Pedidos')
	Usuario --> (2. Exibir lista de pedidos pendentes)
	Usuario --> (3. Selecionar um pedido)
	Usuario --> (4. Exibir detalhes do pedido)
	Usuario --> (5. Atualizar status do pedido)
	Usuario --> (6. Adicionar comentários adicionais)
	Usuario --> (7. Salvar alterações)
	Usuario --> (8. Exibir confirmação)
	}
	
	(5. Atualizar status do pedido) -down-> (5a. Realizar ações corretivas)
	(7. Salvar alterações) -down-> (7a. Exibir mensagem de erro)
	}
	
	@enduml

2.3 Requisitos não-funcionais

2.3.1 Requisitos de desempenho

Os requisitos de desempenho devem ser definidos de forma clara e mensurável, como tempo de resposta do sistema, capacidade de processamento, tempo de entrega de produtos, taxa de acurácia no controle de estoque, entre outros indicadores quantitativos.

2.3.2 Requisitos de dados persistentes

Descrevem-se aqui estruturas lógicas de dados persistentes (que mantém seu valor após a execução do programa) que sejam usadas pelo produto. Cada estrutura de dados pode ser, por exemplo, um arquivo convencional ou uma tabela em um banco de dados.

INCLUIR AQUI O MODELO DE BANCO DE DADOS

2.3.3 Restrições ao desenho

Padrões externos, como normas ISO relacionadas à qualidade e segurança alimentar, podem impor restrições de projeto que devem ser consideradas na implementação do sistema de controle de materiais.

A legislação aplicável ao setor alimentício, como regulamentações de rotulagem de alimentos, pode impor requisitos específicos que influenciam o projeto do sistema.

2.3.4 Atributos de Qualidade

A norma ISO/IEC 9126 fornece diretrizes para a qualidade do software. Os atributos de qualidade podem incluir funcionalidade, confiabilidade, usabilidade, eficiência, manutenção e portabilidade, entre outros, conforme definido por essa norma. Cada atributo pode ter sub características específicas que devem ser consideradas no projeto do sistema de controle de materiais.

2.4 Objetos/Classes

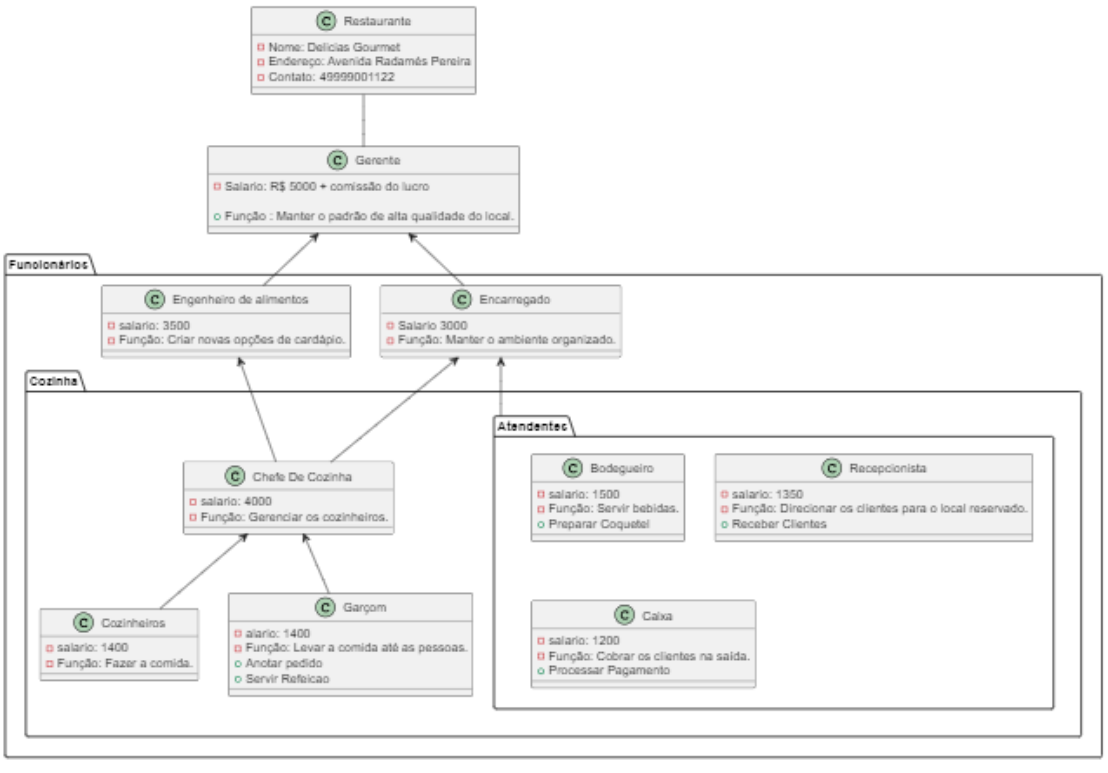
2.4.1 Modelo Conceitual/Classes de Análise/Modelo de Domínio

(Classes, Associações, nomes das associações,
Multiplicidades e Atributos)

```
@startuml

class Restaurante {
  - Nome: Delicias Gourmet
  - Endereço: Avenida Radamés Pereira
  - Contato: 49999001122

}



class Gerente {
  - Salario: R$ 5000 + comissão do lucro

  + Função : Manter o padrão de alta qualidade do local.
}

package "Funcionários" {
  class Encarregado {
    - Salario 3000
```

```
    - Função: Manter o ambiente organizado.
}

class "Engenheiro de alimentos" {
    - salario: 3500
    - Função: Criar novas opções de cardápio.
    }

package "Cozinha" {
  class "Chefe De Cozinha" {
    - salario: 4000
    - Função: Gerenciar os cozinheiros.
    }

  class "Cozinheiros" {
    - salario: 1400
    - Função: Fazer a comida.

  }

  class "Garçom" {
    - alario: 1400
    - Função: Levar a comida até as pessoas.
    + Anotar pedido
    + Servir Refeicao
}

package "Atendentes" {
  class "Bodegueiro" {
    - salario: 1500
    - Função: Servir bebidas.
    + Preparar Coquetel
  }

  class "Recepcionista" {
    - salario: 1350
    - Função: Direcionar os clientes para o local reservado.
    + Receber Clientes
  }

  class "Caixa" {
    - salario: 1200
```

```
      - Função: Cobrar os clientes na saída.
      + Processar Pagamento
    }
  }
}


"Chefe De Cozinha" <-- Garçom
"Chefe De Cozinha" <-- Cozinheiros
"Engenheiro de alimentos" --> Gerente

"Engenheiro de alimentos" <-- "Chefe De Cozinha"
Encarregado <-- "Chefe De Cozinha"
Encarregado <-- Atendentes

Restaurante -- Gerente
Gerente <-- Encarregado
@enduml
```



2.4.2 Eventos e Operações

2.4.3 DSS – Diagramas de Sequência do Sistema, Contratos

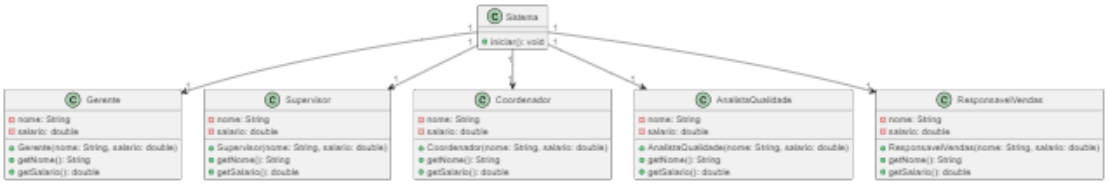
```
@startuml

actor Gerente
actor Supervisor
actor Coordenador
actor AnalistaQualidade
actor ResponsavelVendas

Gerente -> Supervisor: solicitar informações de estoque
Supervisor -> Supervisor: verificar estoque
Supervisor -> Coordenador: solicitar materiais necessários
Coordenador -> Coordenador: verificar disponibilidade dos
materiais
Coordenador -> AnalistaQualidade: solicitar análise de
materiais
AnalistaQualidade -> AnalistaQualidade: analisar materiais
Coordenador -> Coordenador: obter resultado da análise
Coordenador -> Coordenador: planejar uso dos materiais
Coordenador -> Supervisor: informar materiais a serem
adquiridos
Supervisor -> Gerente: solicitar autorização para compras
Gerente -> Gerente: analisar e aprovar solicitação de
compras
Gerente -> ResponsavelVendas: fornecer informações sobre
demanda do mercado
ResponsavelVendas -> Gerente: fornecer informações sobre
fornecedores
Gerente -> Gerente: selecionar fornecedores e ingredientes
Gerente -> Supervisor: aprovar compras
Supervisor -> Coordenador: realizar compras
Coordenador -> Supervisor: receber materiais adquiridos
Supervisor -> Supervisor: armazenar materiais
Supervisor -> Coordenador: informar disponibilidade dos
materiais

@enduml
```

2.4.4 Classes de Implementação - Diagrama de Classes

Multiplicidades, Atributos e Métodos). Atribuição de responsabilidades com GRASP (General Responsibility Assignment Software Patterns) que são um conjunto de princípios e diretrizes para atribuição de responsabilidades em projetos de software orientados a objetos.

```
@startuml

class Sistema {
  + iniciar(): void
}

class Gerente {
  - nome: String
  - salario: double
  + Gerente(nome: String, salario: double)
  + getNome(): String
  + getSalario(): double
}

class Supervisor {
  - nome: String
  - salario: double
  + Supervisor(nome: String, salario: double)
  + getNome(): String
  + getSalario(): double
}

class Coordenador {
  - nome: String
  - salario: double
  + Coordenador(nome: String, salario: double)
  + getNome(): String
  + getSalario(): double
}

class AnalistaQualidade {
  - nome: String
  - salario: double
  + AnalistaQualidade(nome: String, salario: double)
```

```
  + getNome(): String
  + getSalario(): double
}

class ResponsavelVendas {
  - nome: String
  - salario: double
  + ResponsavelVendas(nome: String, salario: double)
  + getNome(): String
  + getSalario(): double
}

Sistema "1" --> "1" Gerente
Sistema "1" --> "1" Supervisor
Sistema "1" --> "1" Coordenador
Sistema "1" --> "1" AnalistaQualidade
Sistema "1" --> "1" ResponsavelVendas

@enduml
```



3 Análise de UCP

As tabelas de escopo de valor do produto e tempo de desenvolvimento com Use Case Points - UCP.

Referências:

IEEE Std. 830 – 1993. IEEE Recommended Practice for Software Requirements Specifications.

IEEE ISO/IEC/IEEE 29148 – 2011. IEEE Systems and software engineering — Life cycle processes — Requirements engineering

OBSERVAÇÃO: Os itens deste modelo de especificação, recomendado pela IEEE, poderão ser complementados com novos itens caso sejam justificáveis.