SENAC

Campus Santo Amaro

.

TADS - Análise Desenvolvimento de Sistemas

POO-Programação Orientada a Objetos

Avaliação POO - N1

Professor: Veríssimo - carlos.hvpereira@sp.senac.br
Aluno: Alex araujo da silva
Turma: A

06/10/2023

Componentes da Equipe:

Alex araujo da silva responsável por toda criação do sistema e documentação.

Planejamento (Cronograma):

*Semana 1-2: Projeto e Análise de Requisitos
*Semana 3-4: Implementação do Backend
*Semana 5: Implementação da Interface de Linha de Comando
*Semana 6: Testes e Resolução de Bugs
*Semana 7: Documentação e Preparação para Entrega

Especificação do Sistema: Sistema de Gerenciamento de Clientes

O Sistema de Gerenciamento de Clientes é uma aplicação de software desenvolvida para auxiliar organizações a registrar, visualizar e gerenciar as demandas dos clientes de forma eficiente. Este sistema oferece funcionalidades para registrar clientes, adicionar demandas, visualizar demandas por código e deletar demandas. Além disso, o sistema possui a capacidade de lidar com demandas urgentes, marcando-as como urgentes quando necessário.

Funcionalidades Principais:

1. Registrar Cliente:
   - O sistema permite que os usuários registrem clientes fornecendo seu nome, telefone e senha.

2. Adicionar Demanda:
   - Usuários autenticados podem adicionar novas demandas, fornecendo uma descrição para a demanda. As demandas também podem ser marcadas como urgentes, se necessário.

3. Visualizar Demanda por Código:
   - Usuários podem visualizar detalhes de uma demanda específica fornecendo seu código. O sistema mostra a descrição da demanda e indica se é urgente ou não.

4. Deletar Demanda por Código:
   - Usuários autorizados podem deletar uma demanda existente fornecendo seu código.

Requisitos Técnicos:

- Linguagem de Programação: Java
- Banco de Dados: Não é utilizado neste sistema (os dados são armazenados em tempo de execução na memória)
- interface Gráfica: Não é utilizado neste sistema (a interação é feita via linha de comando)
- Autenticação: O sistema requer autenticação para adicionar demandas, garantindo a segurança dos dados dos clientes.

Cenário de Uso: Gerenciamento de Clientes

Descrição:* O sistema de Gerenciamento de Clientes é uma ferramenta desenvolvida para estabelecimentos que desejam automatizar o processo de registro, autenticação e gestão das informações dos clientes. Este sistema permite aos atendentes e gerentes adicionar novos clientes, visualizar detalhes dos clientes existentes e, se necessário, realizar a exclusão de registros de clientes.

Situação Problema:

O restaurante enfrenta desafios significativos em seu processo atual de gerenciamento de clientes. Utilizando um método manual para registrar pedidos e reclamações, o restaurante se depara com problemas frequentes, incluindo erros nos pedidos e dificuldades na resolução eficaz de reclamações dos clientes. A falta de um sistema organizado e automatizado tem impactos negativos na eficiência operacional e na satisfação do cliente.

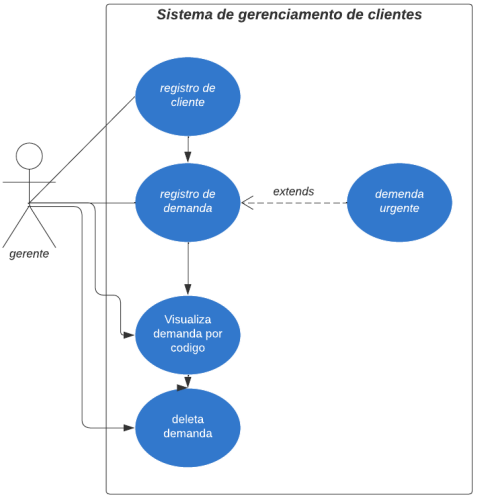
A situação atual exige uma solução que melhore a precisão dos pedidos, otimize a comunicação entre a equipe de atendimento e os gerentes, e forneça um meio eficiente para lidar com reclamações dos clientes. É fundamental implementar um sistema de gerenciamento de clientes que seja fácil de usar, livre de erros e que aprimore significativamente a experiência do cliente no restaurante.

Solução de Software (Lista dos Requisitos Implementados):

- Registro de clientes com nome, telefone e senha.
- Adição de demandas com descrição e opção para marcar como urgente.
- Visualização de demandas existentes por código, mostrando descrição e status de urgência.
- Deleção de demandas existentes por código.

Diagrama de caso de uso

Alex Araujo | October 13, 2023



Detalhamento do Caso de uso #1.1 – Cadastro

Nome do Caso de Uso:	Cadastro
Atores	Cliente
Trigger:	faz o cadastro
Pré-requisito	Estar online.
Fluxo de eventos	cliente entra no sistema e cadastrar seu nome, número, e cpf

1.a.1 Detalhamento do Caso de uso #1.2 – Cliente registra sua demanda.

Nome do Caso de Uso:	registro de demanda
Atores	Cliente
Trigger:	o cliente registra sua demanda para a empresa.
Pré-requisito	escolher sobre elogio, reclamação ou suporte.
Fluxo de eventos	ter se cadastrado no sistema

Caso de Uso:

- Ator Principal: Cliente

- Fluxo Principal:

1. registra cliente com nome, telefone e senha.

2. cliente adiciona uma nova demanda, fornecendo uma descrição e indicando se é urgente.

3. cliente visualiza detalhes de uma demanda existente fornecendo seu código.

4. cliente deleta uma demanda existente fornecendo seu código.

- Fluxo Alternativo:

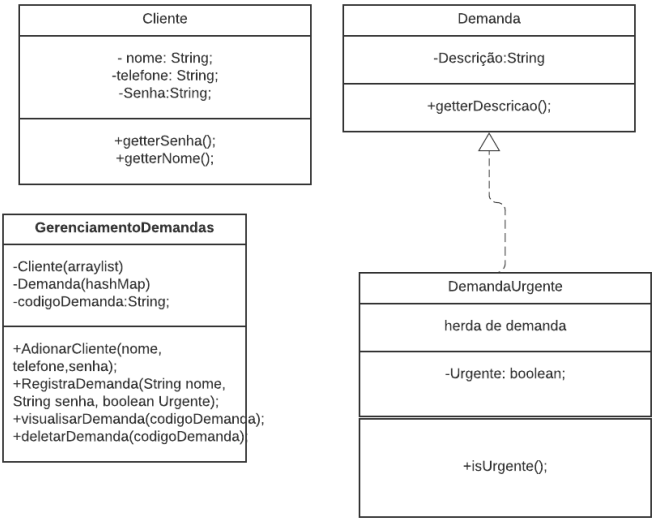
1. No passo 2 do Fluxo Principal, se a demanda é urgente, o sistema marca como "Urgente".

2. No passo 3 do Fluxo Principal, se a demanda não existe ou o código fornecido é inválido, o sistema exibe uma mensagem indicando que a demanda não foi encontrada.

3. No passo 4 do Fluxo Principal, se a demanda não existe ou o código

fornecido é inválido, o sistema exibe uma mensagem indicando que a demanda não foi encontrada.

Diagrama de Classes:


Cliente
- nome: String;
-telefone: String;
-Senha:String;
+getterSenha();
+getterNome();
Demanda
-Descrição:String
+getterDescricao();
GerenciamentoDemandas
-Cliente(arraylist)
-Demanda(hashMap)
-codigoDemanda:String;
+AdionarCliente(nome, telefone,senha);
+RegistraDemanda(String nome, String senha, boolean Urgente);
+visualisarDemanda(codigoDemanda);
+deletarDemanda(codigoDemanda);
DemandaUrgente
herda de demanda
-Urgente: boolean;
+isUrgente();

Código:

```
package OrientacaoObjeto;
public class Cliente {
private String nome;
private String telefone;
private String senha;
public Cliente(String nome, String telefone, String senha) {
this.nome = nome;
this.telefone = telefone;
this.senha = senha;
}
public String getNome() {
return nome;
}
public String getSenha() {
return senha;
}
}
```

~~~~~~

```
package OrientacaoObjeto;
public class Demanda {
private String descricao;
public Demanda(String descricao) {
this.descricao = descricao;
}
public String getDescricao() {
return descricao;
}
}
```

~~~~~~

```
package OrientacaoObjeto;
public class DemandaUrgente extends Demanda {
public DemandaUrgente(String descricao) {
super(descricao);
}
}
```

~~~~~~

```
package OrientacaoObjeto;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
public class GerenciadorDemandas {
private ArrayList<Cliente> clientes;
private Map<Integer, Demanda> demandas;
private int codigoDemanda;
public GerenciadorDemandas() {
clientes = new ArrayList<>();
demandas = new HashMap<>();
codigoDemanda = 1;
}
public void adicionarCliente(String nome, String telefone, String
senha) {
Cliente cliente = new Cliente(nome, telefone, senha);
clientes.add(cliente);
}
public int registrarDemanda(String nome, String senha, String
descricao, boolean urgente) {
for (Cliente cliente : clientes) {
if (cliente.getNome().equals(nome) &&
cliente.getSenha().equals(senha)) {
Demanda demanda;
if (urgente) {
demanda = new DemandaUrgente(descricao);
} else {
demanda = new Demanda(descricao);
}
demandas.put(codigoDemanda, demanda);
return codigoDemanda++;
}
```
~~~~~~

```
}
return -1; // Cliente não encontrado ou senha incorreta
}
public Demanda visualizarDemanda int codigoDemanda) {
return demandas.get(codigoDemanda);
}
public void deletarDemanda int codigoDemanda) {
demandas.remove(codigoDemanda);}
}
```

~~~~~~

```
package OrientacaoObjeto;
import java.util.Scanner;
public class Main {
public static void main String[] args) {
Scanner scanner = new Scanner System.in);
GerenciadorDemandas gerenciador = new GerenciadorDemandas();
while (true) {
System.out.println("\nMenu:");
System.out.println("1 - Registrar Cliente");
System.out.println("2 - Adicionar Demanda");
System.out.println("3 - Visualizar Demanda por Código");
System.out.println("4 - Deletar Demanda por Código");
System.out.println("5 - Sair");
System.out.print("Escolha uma opção: ");
int opcao = scanner.nextInt();
switch (opcao) {
case 1:
System.out.print("Digite o nome do cliente: ");
String nome = scanner.next();
System.out.print("Digite o telefone do cliente: ");
String telefone = scanner.next();
System.out.print("Digite a senha do cliente: ");
String senha = scanner.next();
```
~~~~~~

```
gerenciador.adicionarCliente(nome, telefone, senha);
System.out.println("Cliente registrado com sucesso!");
break;
case 2:
System.out.print("Digite o nome do cliente: ");
String nomeCliente = scanner.next();
System.out.print("Digite a senha do cliente: ");
String senhaCliente = scanner.next();
System.out.print("Digite a descrição da demanda: ");
String descricao = scanner.next();
System.out.print("A demanda é urgente? (true/false): ");
boolean urgente = scanner.nextBoolean();
int codigoDemanda = gerenciador.registrarDemanda(nomeCliente,
senhaCliente, descricao, urgente);
if (codigoDemanda != -1) {
System.out.println("Demanda registrada com sucesso! Código da
Demanda: " + codigoDemanda);
} else {
System.out.println("Cliente não encontrado ou senha incorreta. Não
foi possível registrar a demanda.");
}
break;
case 3:
System.out.print("Digite o código da demanda: ");
int codigoVisualizacao = scanner.nextInt();
Demanda demanda =
gerenciador.visualizarDemanda(codigoVisualizacao);
if (demanda != null) {
System.out.println("Demanda: " + demanda.getDescricao());
} else {
System.out.println("Demanda não encontrada.");
}
break;
case 4:
System.out.print("Digite o código da demanda a ser deletada: ");
int codigoDeletar = scanner.nextInt();
gerenciador.deletarDemanda(codigoDeletar);
System.out.println("Demanda deletada com sucesso!");
break;
case 5:
System.out.println("Saindo do programa. Até mais!");
```

```
System.exit(0);
break;
default:
System.out.println("Opção inválida. Por favor, escolha uma opção
válida.");
}
}
}
}
```

Conclusão:

O Sistema de Gerenciamento de Clientes apresentou uma solução eficaz para o restaurante, permitindo um registro organizado e uma gestão eficiente das informações dos clientes. A automação do processo resultou em maior precisão, melhor comunicação entre atendentes e gerentes, além de proporcionar um serviço mais ágil e personalizado aos clientes.

Esta solução é altamente adaptável e pode ser facilmente expandida para incorporar funcionalidades adicionais no futuro, como análises de desempenho e feedback dos clientes. Com este software de gerenciamento de clientes em funcionamento, o restaurante está bem preparado para oferecer um serviço de qualidade aos seus clientes, fortalecendo sua reputação e garantindo a fidelidade dos clientes existentes. A eficácia desta solução demonstra o valor da automação no setor de restaurantes, contribuindo significativamente para a satisfação do cliente e para o sucesso do negócio.