## 

|  |
| --- |
|  |

## 

Nome do projeto?

Versão 1.0

Projeto: curso/ Senac/ Full stack

Instrutor: Gustavo Blima

Equipe: Iôkim Diego

Marcos Vinicius

David Luan

Eliane daniel

Leonardo

lincoln

2024

**ÍNDICE DETALHADO**

**PREFÁCIO…………………………………………………………………………………………………………………4**

**1.**  **INTRODUÇÃO AO DOCUMENTO…………………………………………………………………………5**

1.1. TEMA………………………………………………………………………………………………………………………………………..5

1.2. OBJETIVO DO PROJETO……………………………………………………………………………………………………………..5

1.3. DELIMITAÇÃO DO PROBLEMA…………………………………………………………………………………………………..5

**2. DESCRIÇÃO GERAL DO SISTEMA…………………………………………………………………………..6**

2.1. PRINCIPAIS ENVOLVIDOS E SUAS CARACTERÍSTICAS………………………………………………………………..6

2.2. REGRAS DE NEGÓCIO……………………………………………………………………………………………………………….6

**3. REQUISITOS DO SISTEMA…………………………………………………………………………………………….7**

3.1. REQUISITOS FUNCIONAIS………………………………………………………………………………………………………..7

3.2. REQUISITOS NÃO-FUNCIONAIS………………………………………………………………………………………………..7

3.3. PROTÓTIPO……………………………………………………………………………………………………………………………...7

3.4. MÉTRICAS E CRONOGRAMA……………………………………………………………………………………………………..7

**4. ANÁLISE E DESIGN…………………………………………………………………………………………………………8**

4.1. ARQUITETURA DO SISTEMA ....................................................................................................................8

4.2. MODELO DO DOMÍNIO………………………………………………………………………………………………………………8

4.3. DIAGRAMAS DE CASO DE USO…………………………………………………………………………………………………..8

4.4. DIAGRAMA DE CLASSES……………………………………………………………………………………………………………..8

4.5. DIAGRAMA DE ATIVIDADE………………………………………………………………………………………………………….8

4.5. DIAGRAMA DE COMPONENTES…………………………………………………………………………………………………..8

4,6. MODELO DE DADOS……………………………………………………………………………………………………………………8

4.7. MODELO LOGICO DA BASE DE DADOS……………………………………………………………………………………….8

4.8. DICIONÁRIO DE DADOS……………………………………………………………………………………………………………...8

4.9. AMBIENTE DE DESENVOLVIMENTO…………………………………………………………………………………………….8

4.10. SISTEMAS E COMPONENTES EXTERNOS UTILIZADOS……………………………………………………………….8

**5. IMPLEMENTAÇÃO……………………………………………………………………………………………….17**

**6. TESTES………………………………………………………………………………………………………………..18**

6.1 PLANO DE TESTES……………………………………………………………………………………………………………………..18

6.2 EXECUÇÃO DO PLANO DE TESTES…………………………………………………………………………………………….18

**7. IMPLANTAÇÃO……………………………………………………………………………………………………19**

7.1 DIAGRAMA DE IMPLANTAÇÃO………………………………………………………………………………………………….19

7.2 MANUAL DE IMPLANTAÇÃO………………………………………………………………………………………………………19

**8. MANUAL DO USUÁRIO…………………………………………………………………………………………20**

.

**PREFÁCIO**

Este é um projeto ambicioso de jovens programadores em sua acessão ao mundo da programação, desejamos por meio desse projeto não só provar ao mundo nossas habilidade como programadores, desejamos provar a todos que com vontade e determinação podemos criar, fazer, construir nossos desejos. programa não é fácil como diz muita gente, mas não é impossível como muitos pensam, programação é determinação, mais ação do que planos, nas páginas a frente você vai encontrar um projeto que não é novidade, já existem alguns projetos parecidos por aí, porém o nosso projeto tem um diferencial, criamos algo fácil de usar, de fácil manutenção, uma solução para pequenos e grandes empreendedores, desejamos que nossa aplicação caia no gosto popular de todos clientes e empresários.!

**1.**  **INTRODUÇÃO AO DOCUMENTO**

Quando você vai com a família ao restaurante, gostaria de não depender de um atendente ou ter que ir até o caixa pegar uma fila, pra e realizar o pedido?

Seria muito bom chegar senta na mesa e por meio de seu smartphone, ter acesso ao cardápio do dia, fazer seu pedido, é só esperar ficar pronto, essa aplicação tem como objetivo resolver esse essa situação, nos tópicos a frente veja como ela vai funciona, e como vai ser desenvolvido.

**1.1. TEMA**

O tema da aplicação é cardapio facil, esse tema foi escolhido através de uma observação do espectro cotidiano, onde foi analisado a complexidade que ainda é fazer um simples pedido, tem lugares que o número de atendente são poucos, outros que é necessário pegar filas, às vezes você só quer sentar e relaxar, esperando a comida chegar.

**1.2. OBJETIVO DO PROJETO**

Nosso objetivo é criar uma aplicação acessível, fácil de trabalhar, de baixo custo, que seja uma boa experiência para o usuário, que o empreendedor tenha um bom retorno, o objetivo não é só gerar lucros, mas proporcionar uma boa experiência para todos, visando o conforto, a praticidade, e o acesso de todos.

**1.3. DELIMITAÇÃO DO PROBLEMA**

Não estamos criando um sistema robusto cheio de funções**,** com banco de dados gigantes, difícil de rodar, com alto consumo de dados, mas uma aplicação simples, de fácil manutenção, uma interface interativa, básica, e intuitiva.

**2. DESCRIÇÃO GERAL DO SISTEMA**

**2.1. PRINCIPAIS ENVOLVIDOS E SUAS CARACTERÍSTICAS**

**2.2. REGRAS DE NEGÓCIO**

**3. REQUISITOS DO SISTEMA**

**3.1. REQUISITOS FUNCIONAIS**

**3.2. REQUISITOS NÃO-FUNCIONAIS**

**3.3. PROTÓTIPO**

**3.4. MÉTRICAS E CRONOGRAMA**

**4. ANÁLISE E DESIGN**

**4.1. ARQUITETURA DO SISTEMA**

**4.2. MODELO DO DOMÍNIO**

**4.3. DIAGRAMAS DE CASO DE USO**

**Descrição do Diagrama:**

**1. \*\*Cliente:\*\*** Inicia a interação escaneando o QR code disponível no estabelecimento com seu dispositivo móvel.

**2. \*\*Sistema:\*\*** Após a leitura do QR code, o sistema redireciona o cliente para o cardápio online.

**3. \*\*Cliente:\*\*** Navega pelo cardápio, visualiza os itens disponíveis e seleciona os produtos desejados.

**4. \*\*Sistema:\*\*** Registra os itens escolhidos e exibe um resumo do pedido para o cliente confirmar.

**5. \*\*Cliente:\*\*** Confirma o pedido.

**6. \*\*Sistema:\*\*** Gera uma confirmação de pedido e inicia o processo de execução.

**7. \*\*Cozinha/Preparação:\*\*** Recebe o pedido do sistema e começa a preparar os itens solicitados.

**8. \*\*Sistema:\*\*** Atualiza o status do pedido em tempo real, indicando a preparação em andamento.

**9. \*\*Cliente:\*\*** Pode acompanhar o status do pedido através do sistema, verificando se está em processo de preparação, pronto para entrega ou entregue.

**10. \*\*Entrega:\*\*** Quando o pedido está pronto, o atendente é notificado, e o sistema atualiza o status para "Pronto para Entrega".

**11**. **\*\*Sistema:\*\*** Rastreia o status da entrega em tempo real.

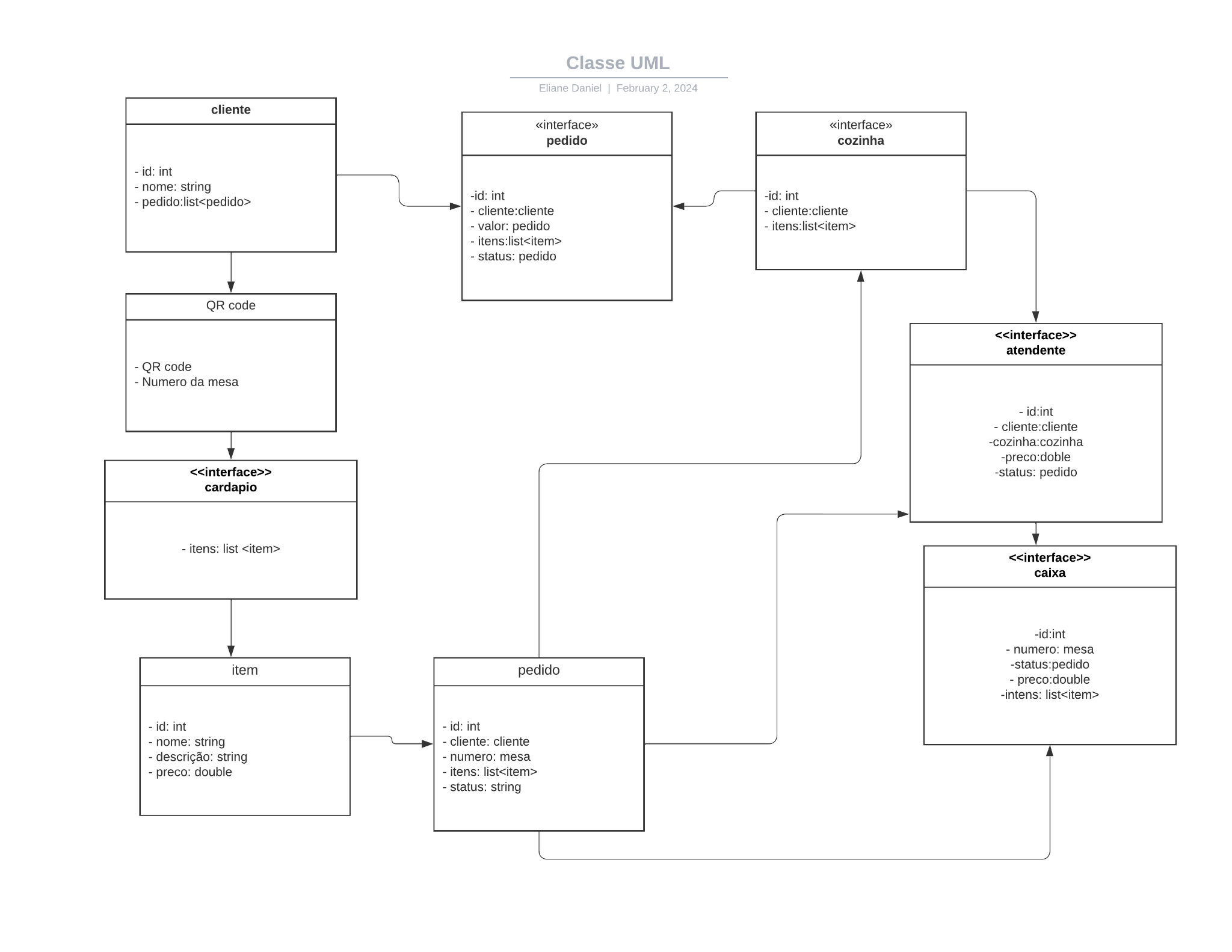
**12.** **\*\*Cliente:\*\*** Recebe a notificação de que o pedido está a caminho.

**13.** **\*\*atendente:\*\*** Entrega o pedido ao cliente.

**14. \*\*Cliente:\*\*** Recebe a entrega e confirma a conclusão do pedido no sistema.

Esse diagrama ilustra a interação do cliente com o sistema desde o acesso ao cardápio online por meio do QR code até a entrega do pedido, incluindo o acompanhamento em tempo real do status do pedido.

**4.4. DIAGRAMA DE CLASSES**



**1. cliente:**

Classe/Entidade: Cliente

Atributos:

* id: int
* nome: string
* numero\_mesa: string
* pedidos: List<Pedido>

**2. Atendente:**

Classe/Entidade: Atendente

Atributos:

* id: int
* nome: string
* cargo: string
* pedidosAtendidos List<Pedido>

**3. cozinha:**

Classe/Entidade: Cozinha

Atributos:

* id: int
* nome: string
* pedidosEmPreparo: List<Pedido>

**4. caixa:**

Classe/Entidade: Caixa

Atributos:

* id: int
* nome: string
* totalVendas:double
* transacoes:: List<transacao>

**5. Pedido:**

Classe/Entidade: Pedido

Atributos:

* id: int
* cliente:cliente
* itens:list<itemPedido>
* status: string

**6. itemPedido:**

Classe/Entidade: itemPedido

Atributos:

* item:item
* quantidade:int

**7.item:**

Classe/Entidade: item

Atributos:

* id: int
* nome: string
* descricao: string
* preco:double

**8. Transacao:**

Classe/Entidade: Transacao

Atributos:

* id: int
* valor:double
* data:DateTime

Cada classe é representada por uma entidade no sistema, e os atributos são as características ou propriedades dessas entidades. Por exemplo, a classe "Cliente" tem atributos como "nome" e "numero\_mesa", enquanto a classe "Pedido" tem atributos como "itens" e "status". Essas classes e atributos ajudam a modelar e organizar as informações no sistema.

**4.5. DIAGRAMA DE COMPONENTES**

| **ClienteApp**  **--------------------------------------------**  - interface gráfica  - lógica de Cliente  - conexão com QRcode  -comunicação com pedidoService | **AtendenteApp**  **--------------------------------------------**  - interface Gráfica  - lógica de atendimento  - conexão com Cliente  - comunicação com a  cozinhaService | **CozinhaApp**  **-------------------------------------------**  - interface Gráfica  - lógica de cozinha  - conexão com o pedido  - comunicação com ClienteServico |
| --- | --- | --- |

| CaixaApp  -------------------------------------  - Interface Gráfica  - Lógica de Caixa  - Conexão com  CaixaService | Sistema de Pedidos  -------------------------------------  - Lógica Principal  - Coordena as interações entre os componentes |
| --- | --- |
| CaixaService  -------------------------------------  - Lógica de Pagamento  - Comunicação com Banco de Dados  - Conexão com Sistema de Pedidos | PedidoService  -------------------------------------  - Lógica de Pedidos  - Comunicação com  CozinhaService |

**Descrição do Diagrama de Componentes**

**1. <Aplicações Cliente, Atendente e Cozinha:>**

- Cada aplicação representa a interface gráfica e lógica específica para o cliente, atendente e cozinha.

- Cada aplicação se comunica com o componente de serviço correspondente.

**2. <Sistema de Pedidos:>**

- Representa a lógica principal do sistema, coordenando as interações entre os diferentes componentes.

- Contém a lógica central de processamento de pedidos e atualização de status.

**3. <CaixaApp e CaixaService:>**

- Representam a interface gráfica e a lógica de caixa.

- O serviço de caixa lida com a lógica de pagamento e se comunica com o Banco de Dados e o Sistema de Pedidos.

**4. <PedidoService:>**

- Lida com a lógica de pedidos, incluindo o processamento de pedidos, comunicação com a cozinha e atualização de status.

- Comunica-se com o Sistema de Pedidos e o CozinhaService.

**5. <CozinhaApp e CozinhaService:>**

- Representam a interface gráfica e a lógica da cozinha.

- O serviço de cozinha lida com a preparação dos itens do pedido e comunica-se com o PedidoService.

**6. <ClienteService e QRCode:>**

- Lida com a lógica do cliente, incluindo a comunicação com o QRCode para acessar o cardápio.

- Comunica-se com o Sistema de Pedidos.

Este diagrama de componentes ilustra a estrutura geral do sistema, mostrando como as diferentes partes interagem entre si para possibilitar o funcionamento do sistema de pedidos, desde a realização do pedido até a conclusão do pagamento.

**4,6. MODELO DE DADOS**

Para modela o banco de dados desse sistema, utilizamos o modelo conceitual simples, usando a modelagem Entidade-Relacionamento(ER)

**Modelo Conceitual:**

* **Entidades:**

**Cliente:**

* + id (Chave Primária)
  + nome
  + numero\_mesa

**Pedido:**

* + id (Chave Primária)
  + id\_cliente (Chave Estrangeira referenciando Cliente)
  + status

**Item:**

* + id (Chave Primária)
  + nome
  + descricao
  + preco

**ItemPedido:**

* + id\_pedido (Chave Estrangeira referenciando Pedido)
  + id\_item (Chave Estrangeira referenciando Item)
  + quantidade

**Atendente:**

* + id (Chave Primária)
  + nome
  + cargo

**Cozinha:**

* + id (Chave Primária)
  + nome

**Caixa:**

* + id (Chave Primária)
  + nome
  + total\_vendas

**4.7. MODELO LÓGICO DA BASE DE DADOS**

* **Cliente:**
  + Atributos: id\_cliente (PK), nome, numero\_mesa.
* **Pedido:**
  + Atributos: id\_pedido (PK), status, id\_cliente (FK).
  + Relacionamentos: Cliente (1:N).
* **Item:**
  + Atributos: id\_item (PK), nome, descricao, preco.
* **ItemPedido:**
  + Atributos: id\_pedido (PK, FK), id\_item (PK, FK), quantidade.
  + Relacionamentos: Pedido (N:N), Item (N:N).
* **Atendente:**
  + Atributos: id\_atendente (PK), nome, cargo.
* **Cozinha:**
  + Atributos: id\_cozinha (PK), nome.
* **Caixa:**
  + Atributos: id\_caixa (PK), nome, total\_vendas.

No banco de dados será criado 7 tabelas ao total:

CREATE TABLE cliente

CREATE TABLE pedido

CREATE TABLE item

CREATE TABLE itemPedido

CREATE TABLE atendente

CREATE TABLE cozinha

CREATE TABLE caixa

o nome das tabelas são todo em letra minúscula, os nomes não deve ser mudados, para não criar erro quando o banco de dados for conectado ao sequelize no node.js. o banco foi modelado no conceito Entidade-Relacionamento, porém na programação utilizaremos o conceito ORM ( mapeamento Objeto-Relacional) O modelo ORM é uma técnica para mapear objetos do código para entidades no banco de dados de forma transparente.

Permite que os desenvolvedores usem objetos e classes em seu código para representar dados, e o ORM cuida automaticamente da persistência desses objetos no banco de dados.

O objetivo é abstrair as complexidades das consultas SQL e facilitar a interação entre o código e o banco de dados.

**4.8. DICIONÁRIO DE DADOS**

1. Cliente:

* + id\_cliente (PK): Identificador único do cliente.
  + nome: Nome do cliente.
  + numero\_mesa: mesa que o cliente escolheu.

2. Pedido:

* + id\_pedido (PK): Identificador único do pedido.
  + status: Status atual do pedido.
  + id\_cliente (FK): Chave estrangeira referenciando o Cliente.

3.Item:

* + id\_item (PK): Identificador único do item.
  + nome: Nome do item.
  + descricao: Descrição do item.
  + preco: Preço do item.

4.ItemPedido:

* + id\_pedido (PK, FK): Chave estrangeira referenciando o Pedido.
  + id\_item (PK, FK): Chave estrangeira referenciando o Item.
  + quantidade: Quantidade do item no pedido.

5. Atendente:

* + id\_atendente (PK): Identificador único do atendente.
  + nome: Nome do atendente.
  + cargo: Cargo do atendente.

6. Cozinha:

* + id\_cozinha (PK): Identificador único da cozinha.
  + nome: Nome da cozinha.

7. Caixa:

* + id\_caixa (PK): Identificador único do caixa.
  + nome: Nome do caixa.
  + total\_vendas: Total de vendas registradas no caixa.

Observações:

* PK: Chave Primária.
* FK: Chave Estrangeira.

Este dicionário de dados fornece uma descrição detalhada de cada tabela, identificando as chaves primárias, chaves estrangeiras e atributos associados a cada entidade, o dicionário será atualizado conforme a estrutura do banco de dados sofrer mudanças.

**4.9. AMBIENTE DE DESENVOLVIMENTO**

Será utilizado o ambiente de desenvolvimento Node.js, no node.js utilizaremo o drive oficial do MYSQL o “mysql2”. o mysql2 é uma versão moderna e melhorada do antigo”mysql” oferece melhor desempenho e suporte para várias consultas simultâneas ……………..continua

**4.10. SISTEMAS E COMPONENTES EXTERNOS UTILIZADOS**

**5. IMPLEMENTAÇÃO**

**6. TESTES**

**6.1 PLANO DE TESTES**

**6.2 EXECUÇÃO DO PLANO DE TESTES**

**7. IMPLANTAÇÃO**

**7.1 DIAGRAMA DE IMPLANTAÇÃO**

**7.2 MANUAL DE IMPLANTAÇÃO**

**8. MANUAL DO USUÁRIO**