Proposta de Trabalho – Desenvolvimento de API

Sistema: Controle de Pedidos em um Restaurante

Aluno: Pedro Gentil Roodes Rodriogues

Data de envio: 30/05/2025

Entrega final: 06/06/2025

Professor: Edécio

Descrição do Projeto

A proposta consiste no desenvolvimento de uma API para gerenciar os pedidos de um restaurante, incluindo o cadastro de pratos, clientes, pedidos, itens dos pedidos e pagamentos. O sistema permitirá operações CRUD completas e incluirá uma transação que abrange o registro de um pedido junto aos itens e pagamento.

Models (Tabelas)

- clientes
  - id (PK)
  - nome
  - email
  - telefone
- pratos
  - id (PK)
  - nome
  - descricao
  - preco
- pedidos

    - id (PK)
    - cliente_id (FK → clientes.id)
    - data
    - status

- itens_pedido
    - id (PK)
    - pedido_id (FK → pedidos.id)
    - prato_id (FK → pratos.id)
    - quantidade
    - subtotal

- pagamentos
    - id (PK)
    - pedido_id (FK → pedidos.id)
    - forma_pagamento
    - valor
    - data_pagamento

Transação Implementada

Ao realizar a inclusão de um novo pedido, o sistema irá:
1. Criar o registro do pedido.
2. Inserir os itens do pedido (relacionando com os pratos).
3. Registrar o pagamento associado.

Essa operação será executada como uma transação: se qualquer parte falhar, nenhuma das ações será persistida no banco de dados.

A exclusão de um pedido também será uma transação que excluirá:
- Os itens do pedido,
- O pagamento correspondente,
- E por fim, o próprio pedido.

Atividades Atendidas

1. Criação do banco de dados, aplicação e migrations das models
2. CRUD nas tabelas básicas (clientes, pratos)

3. Inclusão e listagem com transações (pedidos, itens_pedido, pagamentos)
4. Exclusão com transações (pedido, itens, pagamentos)
5. Envio de e-mail com histórico de pedidos ao cliente usando Nodemailer

Prints ou Definição das Models

Será adicionado print do diagrama relacional via DBeaver ou o schema gerado no ORM (ex: Prisma).

Exemplo em Prisma:

```
model Cliente {
  id       Int      @id @default(autoincrement())
  nome     String
  email    String
  telefone String
  pedidos  Pedido[]
}

model Prato {
  id       Int     @id @default(autoincrement())
  nome     String
  descricao String
  preco    Float
  itens    ItemPedido[]
}

model Pedido {
  id       Int          @id @default(autoincrement())
  cliente   Cliente      @relation(fields: [clienteId], references: [id])
  clienteId Int
  data      DateTime     @default(now())
  status    String
  itens     ItemPedido[]
  pagamento Pagamento?
}

model ItemPedido {
```

```
  id       Int     @id @default(autoincrement())
  pedido    Pedido  @relation(fields: [pedidoId], references: [id])
  pedidoId  Int
  prato     Prato   @relation(fields: [pratoId], references: [id])
  pratoId   Int
  quantidade Int
  subtotal  Float
}

model Pagamento {
  id           Int     @id @default(autoincrement())
  pedido       Pedido  @relation(fields: [pedidoId], references: [id])
  pedidoId     Int     @unique
  formaPagamento String
  valor        Float
  dataPagamento DateTime
}
```