Centro Universitário Católica de Santa Catarina - Joinville

Engenharia de Software

Eduardo Vinicios Klug

BarTab - Gestão de Contas e Consumo

de clientes em bares

Sumário

Resumo

Este projeto tem como objetivo o desenvolvimento do **BarTab**, uma solução voltada para pequenos estabelecimentos, como bares e botecos, que enfrentam desafios na gestão de contas e comandas. O foco está em oferecer um sistema simples, intuitivo e acessível para controle de vendas, gerenciamento de mesas e pagamentos, incluindo a funcionalidade de "Marcar Depois". Além da entrega do sistema, o projeto também engloba atividades como levantamento de requisitos, modelagem, implementação e testes, seguindo boas práticas de engenharia de software.

1 Introdução

1.1 Contexto

Pequenos estabelecimentos, como bares e botecos, geralmente utilizam métodos manuais para controle de consumo e fechamento de contas, como anotações em papel, cadernos ou simples calculadoras. Esse processo depende da memória dos atendentes ou de registros físicos sujeitos a extravio ou erros. Além disso, a comunicação entre diferentes membros da equipe (como garçom e caixa) costuma ser informal, o que pode gerar inconsistências nos lançamentos. Não há, nesses ambientes, um sistema centralizado que registre as movimentações de consumo em tempo real ou que permita acompanhar os valores pendentes por cliente de forma automatizada.

1.2 Justificativa

A ausência de um sistema digital dedicado para gerenciamento de contas abertas em pequenos comércios pode resultar em falhas de registro, dificuldades na organização financeira e perda de tempo no fechamento das contas. Mesmo com a existência de soluções de mercado, muitas são desenvolvidas para restaurantes maiores ou com estruturas mais complexas, o que pode dificultar sua adoção em negócios com rotinas mais simples e informais. Assim, existe uma lacuna entre os sistemas existentes e as necessidades reais de estabelecimentos menores. A proposta deste projeto parte da

identificação dessa lacuna e da necessidade de soluções mais aderentes à realidade desses empreendedores.

1.3 Objetivos

Objetivo Principal: Desenvolver uma solução digital para registrar, gerenciar e finalizar contas abertas em bares e botecos, de forma a reduzir erros manuais e simplificar a rotina, beneficiando diretamente os proprietários desses estabelecimentos com maior controle operacional.

Objetivos Secundários:

- Organizar os lançamentos de consumo por cliente em uma interface clara e acessível.
- Permitir a utilização de diferentes métodos de pagamento e registrar pendências por cliente.
- Facilitar o acesso ao histórico de consumo e pagamentos, promovendo maior rastreabilidade.
- Reduzir o tempo de fechamento de contas e minimizar a chance de erros de cálculo.
- Fornecer ao proprietário uma visão geral sobre contas ativas e valores a receber.

2 Descrição do Projeto

2.1 Tema do Projeto

O projeto consiste no desenvolvimento de um sistema web para controle de contas abertas e consumo em estabelecimentos de pequeno porte, com foco em bares e botecos. A solução será baseada em tecnologias web consolidadas, como React no frontend e NestJS no backend, garantindo compatibilidade com navegadores modernos e facilidade de implantação.

A simplicidade do sistema será definida por:

- Interface com fluxo direto: abertura de contas, adição de itens, visualização do total e encerramento da conta em até três cliques.
- Cadastro mínimo necessário: nome do cliente, itens e valor — evitando obrigatoriedade de dados como CPF, e-mail ou login/senha.
- Funcionalidades voltadas apenas para o essencial do negócio: controle de consumo, métodos de pagamento e pendência ("marcar depois").
- Operação offline-first opcional com sincronização posterior (a depender do tempo de desenvolvimento).

O sistema será projetado para ser executado localmente ou em servidores de hospedagem simples, sem necessidade de infraestrutura complexa ou configuração avançada.

2.2 Problemas a Resolver

Pequenos estabelecimentos enfrentam uma série de dificuldades operacionais na ausência de um sistema informatizado:

- **Risco de perda de informações:** contas são registradas em papel ou blocos físicos, que podem ser perdidos ou extraviados durante o atendimento.
- **Erros no cálculo final da conta:** somas manuais ou feitas rapidamente ao final do expediente aumentam a chance de falhas nos valores cobrados.
- **Falta de controle sobre pendências:** quando o cliente "marca para pagar depois", muitas vezes não há registro confiável da dívida. Isso compromete o controle financeiro e a cobrança futura.
- **Dificuldade na organização do atendimento:** quando há múltiplas contas abertas simultaneamente, o controle manual pode gerar confusões entre nomes semelhantes ou trocas de mesas.
- **Ausência de histórico:** sem um sistema digital, os dados de consumo e pagamento não são armazenados de forma estruturada, impedindo qualquer tipo de consulta futura ou análise do comportamento dos clientes.

2.3 Limitações

- O sistema não abordará o controle de estoque detalhado.
- Não haverá integração com sistemas bancários externos.

3 Especificação Técnica

3.1 Requisitos de Software

Requisitos Funcionais:

- RF01 - O sistema deve permitir a criação e o controle de contas abertas.
- RF02 - O sistema deve adicionar e remover itens da conta do cliente.
- RF03 - O sistema deve exibir resumos de consumo e o total da conta.
- RF04 - O sistema deve permitir diferentes formas de pagamento.
- RF05 - O sistema deve armazenar o histórico de contas e saldos pendentes.
- RF06 - O sistema deve permitir adicionar, alterar e remover itens do sistema.
- RF07 - O sistema deve implementar um sistema de "Marcar Depois", onde clientes podem deixar contas pendentes e consultá-las posteriormente.
- RF08 - O sistema deve criar uma tela de controle de dívidas para listar clientes com saldo negativo e permitir registros de pagamento.
- RF09 - O sistema deve registrar o histórico de pagamentos e permitir pagamentos parciais.
- RF10 - O sistema deve permitir o cadastro, edição, listagem e exclusão de clientes.

Requisitos Não-Funcionais:

- RNF01 - O sistema deve possuir uma interface responsiva e intuitiva.

- RNF02 - O sistema deve utilizar um banco de dados seguro para armazenar informações financeiras.
- RNF03 - A arquitetura do sistema deve permitir a escalabilidade para até 25 contas abertas simultaneamente, com até 100 operações por hora e até 5.000 transações por mês, sem degradação perceptível de desempenho.
- RNF04 - O sistema deve permitir envio de notificações via WhatsApp, SMS ou pop-ups no sistema.

Diagrama de Caso de Uso

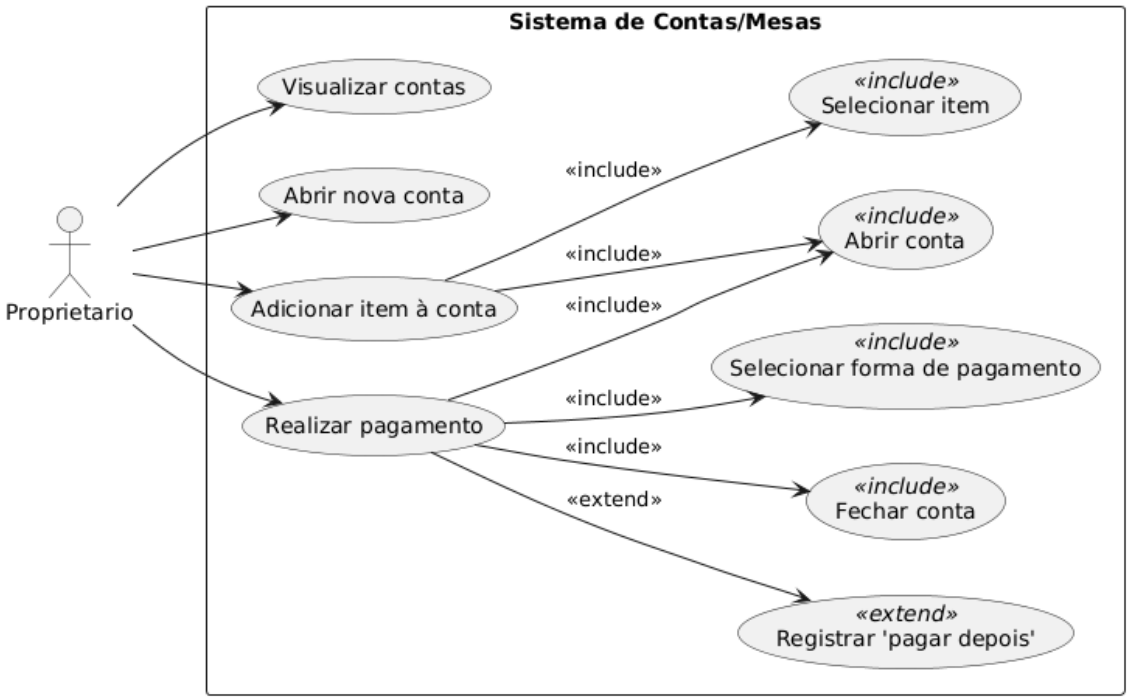

Figura 1: Diagrama de Caso de Uso do Sistema de Contas/Mesas

3.2 Considerações de Design

Visão Inicial da Arquitetura:

- Backend em NestJS
- Frontend em React com TypeScript
- Banco de dados PostgreSQL

Padrões de Arquitetura:

- Aplicar Clean Code e SOLID
- Seguir modelo de Microsserviços para futuras expansões

Diagrama C4

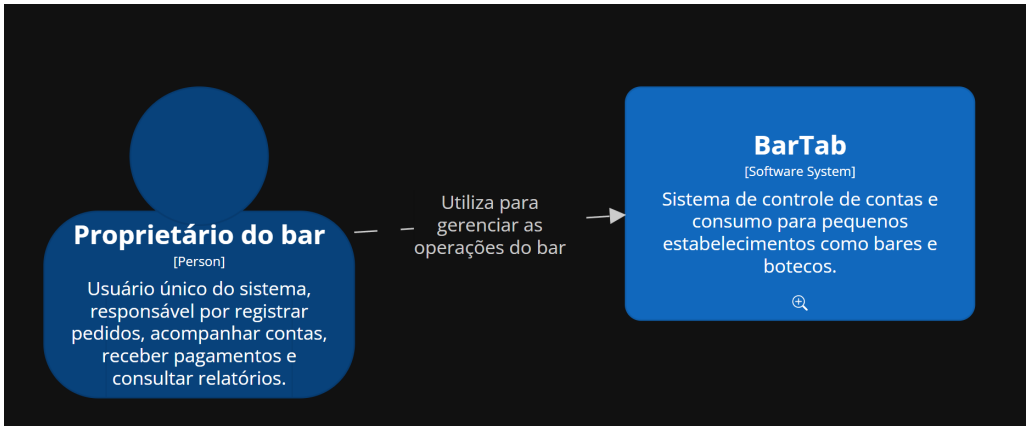

Figura 2: Diagrama de Contexto - BarTab

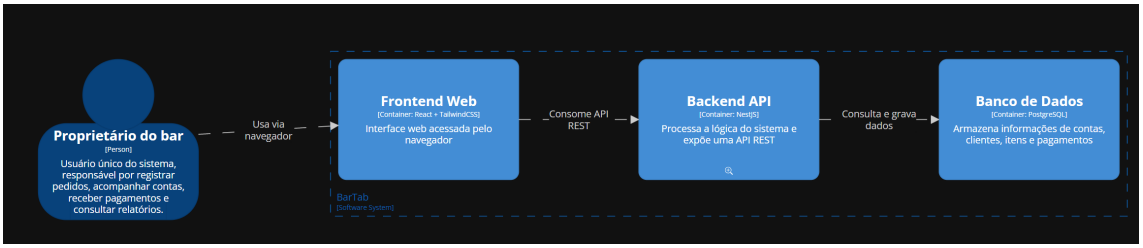

Figura 3: Diagrama de Contêineres - BarTab

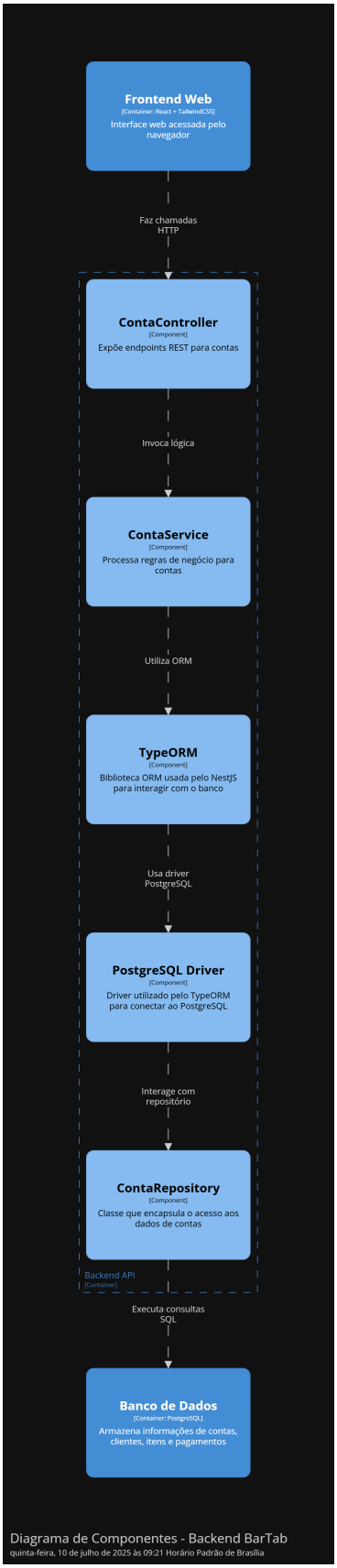

Figura 4: Diagrama de Componentes do Backend - BarTab

3.3 Stack Tecnológica

- Linguagem: TypeScript
- Frontend: React + TailwindCSS
- Backend: NestJS
- Banco de Dados: PostgreSQL
- Gerenciador de Dependências: Yarn

3.4 Considerações de Segurança

- Implementar autenticação e controle de acesso.
- Proteger endpoints sensíveis do backend.
- Realizar validações no frontend e backend para evitar injeções e acessos indevidos.

3.5 Regras para "Marcar Depois"

Cadastro e Identificação do Cliente:

- Ao abrir uma conta, o usuário deve inserir um nome para consulta.
- Se o cliente já existir, o sistema vincula a conta a ele.
- Caso não exista, o sistema cria um novo registro automaticamente.

Armazenamento de Saldos:

- Cada cliente terá um saldo registrado no banco de dados, podendo ser positivo ou negativo.
- Se o cliente optar por pagar depois, o valor será adicionado ao saldo devedor.

Tela de Controle de Dívidas:

- O sistema apresentará uma tela listando todos os clientes com saldo pendente.

- Possibilidade de marcar pagamentos (parciais ou totais) e atualizar o saldo do cliente.
- Histórico de pagamentos acessível para cada cliente.
- Opção para enviar lembretes automáticos via WhatsApp, SMS ou notificação.

4 Próximos Passos

O projeto será desenvolvido utilizando a metodologia ágil **SCRUM**, com sprints quinzenais e entregas incrementais. Esta abordagem garante organização, ritmo constante de evolução e qualidade no processo de desenvolvimento, em conformidade com os direcionamentos da instituição.

Cronograma de Sprints

Sprint	Período Estimado	Entregas / Objetivos Principais
0	Semana 0	Planejamento do projeto, definição do backlog inicial, setup do repositório e ferramentas.
1	Semanas 1–2	Design das interfaces (protótipos), modelagem de dados, diagrama C4 (nível contexto e contêineres).
2	Semanas 3–4	Implementação do CRUD de clientes e CRUD de itens de consumo.
3	Semanas 5–6	Tela inicial com cards das contas abertas e funcionalidade para abrir novas contas.
4	Semanas 7–8	Tela de detalhes da conta: exibição dos itens consumidos, total e botão de pagamento.
5	Semanas 9–10	Fluxo de pagamento: dinheiro, débito, crédito, pix e opção “pagar depois”.
6	Semanas 11–12	Integração do fluxo de “pagar depois” com o saldo devedor do cliente.
7	Semanas 13–14	Testes automatizados (unitários, integração e smoke) com abordagem TDD.
8	Semanas 15–16	Refatorações, ajustes finais, documentação completa e deploy utilizando CI/CD.
9	Semana 17	Apresentação final, entrega oficial e avaliação pelos professores.

Tabela 1: Cronograma de desenvolvimento utilizando SCRUM

Atividades Contínuas

- **Daily Logs**: Registro individual semanal de progresso e obstáculos.
- **Revisão de Sprint**: Avaliação dos objetivos alcançados ao final de cada sprint e replanejamento do backlog.
- **Retrospectiva de Sprint**: Discussão dos pontos positivos, negativos e me-

lhorias no processo.

- **Documentação viva**: Utilização de Wiki no repositório GitHub como fonte centralizada de documentação do projeto.

5 Referências

- Documentação oficial do React: `https://react.dev`
- Documentação oficial do NestJS: `https://docs.nestjs.com`
- PostgreSQL Documentation: `https://www.postgresql.org/docs/`
- Tailwind CSS Docs: `https://tailwindcss.com/docs`
- GitHub - CatolicaSC Portfolio Directions: `https://github.com/CatolicaSC-Portfolio`

6 Avaliações de Professores

Edicarsia Barbiero Pillon

Luiz Carlos Camargo

Claudinei Dias