

PIZZA HUNT - SISTEMA DE GERENCIAMENTO DE PIZZARIA

Geslon Gisch^{*}

Jalisson Ternus^{**}

Thiago Thomas^{***}

Resumo

O sistema de gerenciamento de pizzarias oferece uma gama variada de recursos para atender às necessidades específicas do setor de alimentos e bebidas. Além de abordar o gerenciamento de clientes, cardápios, pedidos, funcionários e estoque, esse sistema também inclui ferramentas de análise e relatórios que possibilitam uma visão abrangente do desempenho do negócio.

Com uma interface intuitiva e fácil de usar, aliada a funcionalidades avançadas, como rastreamento de pedidos em tempo real e personalização de cardápios, este sistema proporciona uma experiência aprimorada para os clientes e otimiza a gestão interna, contribuindo para o crescimento e sucesso contínuo das pizzarias.

^{*}Discente do Curso de Ciência da Computação
Unoesc-Campus de São Miguel do Oeste
Rua Oiapoc, 2011. São Miguel do Oeste-SC
GeslonG@outlook.com

^{**} Discente do Curso de Ciência da Computação
Unoesc-Campus de São Miguel do Oeste
Rua Oiapoc, 2011. São Miguel do Oeste-SC
jalissonternus@gmail.com

^{***} Discente do Curso de Ciência da
ComputaçãoUnoesc-Campus de São Miguel do
Oeste
Rua Oiapoc, 2011. São Miguel do Oeste-SC
thiagothomas09@outlook.com

1 INTRODUÇÃO

No competitivo setor de serviços alimentícios, pizzarias enfrentam desafios em gerenciamento e atendimento ao cliente. A eficiência operacional é crucial, e a tecnologia desempenha um papel vital na modernização dessas operações. Este projeto desenvolve um sistema de gerenciamento integrado para pizzarias, utilizando tecnologias modernas e metodologias ágeis.

O sistema visa otimizar múltiplas funções essenciais: gerenciamento de clientes, cardápio, pedidos, funcionários e estoque. Utilizando Java Spring Boot para o back-end, PostgreSQL para gerenciamento de dados e HTML, CSS e JavaScript para o front-end, o projeto foca na eficiência e na experiência do usuário.

A metodologia ágil, com o uso do Trello para gerenciamento de projetos, permitiu colaboração eficiente e adaptabilidade às mudanças. Essa abordagem assegura um desenvolvimento dinâmico e responsivo, adaptado às necessidades de uma pizzeria moderna.

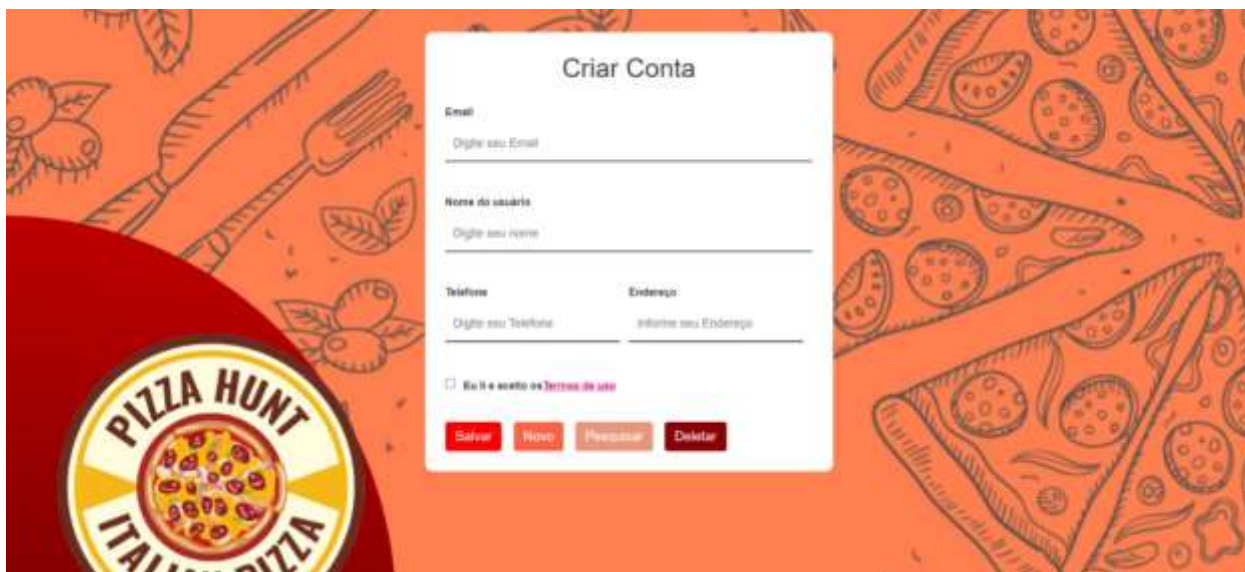
Este artigo detalha o desenvolvimento do sistema, abordando as soluções técnicas, funcionalidades e benefícios para as pizzarias.

2 DESENVOLVIMENTO

2.1 GERENCIAMENTO DE CLIENTES

O módulo de Gerenciamento de Clientes foi desenvolvido para armazenar e gerenciar informações essenciais dos clientes, como nome, endereço e histórico de pedidos. Implementado com Java Spring Boot e PostgreSQL, oferece funcionalidades como atualização e remoção de dados, além de coletar feedback dos clientes sobre serviços e produtos.

Imagem 1 - Tela de Cadastro de Clientes.



Fonte: Os autores(2023)

2.2 GERENCIAMENTO DE CARDÁPIO

No módulo de Gerenciamento de Cardápio, os usuários podem adicionar, atualizar e remover itens como pizzas, bebidas e sobremesas. Este módulo utiliza uma combinação de PostgreSQL para armazenamento de dados e HTML/CSS para apresentação, garantindo uma gestão eficiente e uma interface amigável.

2.3 GERENCIAMENTO DE PEDIDOS

Essencial para a operação da pizzeria, o módulo de Gerenciamento de Pedidos permite a criação, o rastreamento e a atualização do status dos pedidos. A integração com sistemas de pagamento facilita o processamento de transações, enquanto a aplicação de promoções e cupons de desconto é gerenciada de forma eficaz.

2.4 GESTÃO DE FUNCIONÁRIOS

O sistema também inclui um módulo para Gestão de Funcionários, permitindo o cadastro e a administração de informações dos colaboradores. Este módulo facilita o gerenciamento de dados como horários de trabalho, salários e cargos, utilizando Java Spring Boot para lógica de negócios e PostgreSQL para armazenamento de dados.

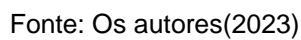
2.5 GESTÃO DE ESTOQUE

O módulo de Gestão de Estoque controla os ingredientes e suas quantidades, com alertas para estoque baixo e integração com fornecedores para reabastecimento automático. A eficiência deste módulo é crucial para evitar a escassez de ingredientes, garantindo a continuidade e qualidade dos serviços oferecidos.

2.6 DIAGRAMAS

Para fornecer uma visão clara do sistema proposto e suas funcionalidades, utilizamos ferramentas de modelagem de software que resultaram em dois diagramas fundamentais: o Diagrama de Caso de Uso e o Diagrama de Classe. O Diagrama de Caso de Uso oferece uma perspectiva externa, mostrando as interações dos usuários com o sistema, enquanto o Diagrama de Classe fornece uma visão interna, detalhando a estrutura lógica do sistema. Ambos são instrumentais para a fase de planejamento e desenvolvimento.

Made with
Visual Paradigm
For non-commercial use



```

classDiagram
    class Produto {
        nome : string
        descrição : string
        preço : float
        qtd_em_estoque : int
        + atualizar_estoque(quantidade)
    }
    class Cliente {
        nome : string
        endereço : string
        email : string
        telefone : string
    }
    class Pedido {
        data : date
        cliente : Cliente
        + adicionar_item(temPedido)
        + calcular_total()
    }
    class Estoque {
        + adicionar_produto(Produto)
        + remover_produto(Produto)
        + consultar_produto(nome_produto : string)
    }
    class Pagamento {
        valor : float
        data : date
        pedido : Pedido
    }
    class Item_pedido {
        quantidade : int
        produto : Produto
        + calcular_subtotal()
    }

    Produto "1" -- "1..*" Cliente : Comprado por
    Cliente "1" -- "1" Pedido : Realiza
    Pedido "0..*" -- "1" Estoque : contém
    Pedido "1..*" -- "1" Item_pedido : contém
    Item_pedido "1" -- "1" Pedido : Recebe
    Pedido "1" -- "1" Pagamento : pagamento do pedido
  
```

The diagram illustrates the following classes and their attributes/operations:

- Produto**: nome (string), descrição (string), preço (float), qtd_em_estoque (int), atualizar_estoque(quantidade).
- Cliente**: nome (string), endereço (string), email (string), telefone (string).
- Pedido**: data (date), cliente (Cliente), adicionar_item(temPedido), calcular_total().
- Estoque**: adicionar_produto(Produto), remover_produto(Produto), consultar_produto(nome_produto : string).
- Pagamento**: valor (float), data (date), pedido (Pedido).
- Item_pedido**: quantidade (int), produto (Produto), calcular_subtotal().

Relationships and Multiplicities:

- Produto** (1) is **Comprado por** **Cliente** (1..*).
- Cliente** (1) **Realiza** **Pedido** (1).
- Pedido** (0..*) **contém** **Estoque** (1).
- Pedido** (1..*) **contém** **Item_pedido** (1).
- Item_pedido** (1) **Recebe** **Pedido** (1).
- Pedido** (1) has a **pagamento do pedido** **Pagamento** (1).

Fonte: Os autores(2023)

```
1  -- Adicionando chaves estrangeiras para a tabela pedidos
2  ALTER TABLE pedidos
3      ADD FOREIGN KEY (id_cliente) REFERENCES clientes(id_cliente),
4      ADD FOREIGN KEY (id_feedback) REFERENCES feedback(id_feedback),
5      ADD FOREIGN KEY (id_funcionario) REFERENCES funcionario(id_funcionario);
6
7  -- Adicionando chaves estrangeiras para a tabela itempedido
8  ALTER TABLE itempedido
9      ADD FOREIGN KEY (id_pedido) REFERENCES pedidos(id_pedido),
10     ADD FOREIGN KEY (id_produto) REFERENCES produto(id_produto);
```

Fonte: Os autores(2023)

Imagem 3 – Script que gera Grupos de acesso com diferentes privilégios.

```
1      -- Cria usuários
2      CREATE USER user_cliente1 WITH PASSWORD 'senha1';
3      CREATE USER user_cliente2 WITH PASSWORD 'senha2';
4      CREATE USER user_funcionario1 WITH PASSWORD 'senha3';
5      CREATE USER user_funcionario2 WITH PASSWORD 'senha4';
6      CREATE USER user_administrador WITH PASSWORD 'senha5';
7
8
9      -- Cria grupos
10     CREATE GROUP grupo_clientes;
11     CREATE GROUP grupo_funcionarios;
12     CREATE GROUP grupo_administradores;
13
14
15     -- Atribui usuários aos grupos
16     GRANT grupo_clientes TO user_cliente1, user_cliente2;
17     GRANT grupo_funcionarios TO user_funcionario1, user_funcionario2;
18     GRANT grupo_administradores TO user_administrador;
19
20     -- Concessão de privilégios para o grupo de clientes na tabela clientes
21     GRANT SELECT ON clientes TO grupo_clientes;
22
23     -- Concessão de privilégios para o grupo de funcionários na tabela pedidos
24     GRANT SELECT, INSERT, UPDATE ON pedidos TO grupo_funcionarios;
25
26     -- Concessão de todos os privilégios para o grupo de administradores na tabela empresa
27     GRANT ALL PRIVILEGES ON empresa TO grupo_administradores;
```

Fonte: Os autores(2023)

Imagem 4 – Script de criação das Procedures requisitadas.

```
1      -- procedimento para verificar o estoque de um produto
2      CREATE OR REPLACE PROCEDURE verificar_estoque_produto(
3          IN produto_id INTEGER,
4          OUT estoque_disponivel NUMERIC
5      )
6      LANGUAGE plpgsql
7      AS $$
8      BEGIN
9          SELECT quantidade_estoque INTO estoque_disponivel
10         FROM Estoque
11         WHERE ingrediente_id = produto_id;
12     END;
13     $$;
14
15
16     -- procedimento para atualizar o status de um pedido
17     CREATE OR REPLACE PROCEDURE atualizar_status_pedido(
18         IN pedido_id INTEGER,
19         IN novo_status VARCHAR(50)
20     )
21     LANGUAGE plpgsql
22     AS $$
23     BEGIN
24         UPDATE pedidos
25         SET status_pedido = novo_status
26         WHERE id_pedido = pedido_id;
27     END;
28     $$;
```

Fonte: Os autores(2023)

3 MATERIAIS E MÉTODOS

3.1 AMBIENTE DE DESENVOLVIMENTO

O sistema foi desenvolvido em um ambiente que integra várias ferramentas e tecnologias. O back-end foi implementado utilizando Java Spring Boot, escolhido por sua robustez e eficiência em aplicações empresariais. Para o gerenciamento de banco de dados, optamos pelo PostgreSQL, devido à sua confiabilidade e compatibilidade

com diversas plataformas. No front-end, utilizamos HTML, CSS e JavaScript, proporcionando uma interface de usuário responsiva e atraente.

3.2 DESIGN E ARQUITETURA DO SISTEMA

A arquitetura do sistema foi projetada para ser modular e escalável, facilitando a manutenção e a futura expansão. Seguindo o padrão MVC (Model-View-Controller), o design separa a lógica de negócios da interface de usuário, promovendo a organização do código e a eficiência no desenvolvimento.

3.3 METODOLOGIA DE DESENVOLVIMENTO

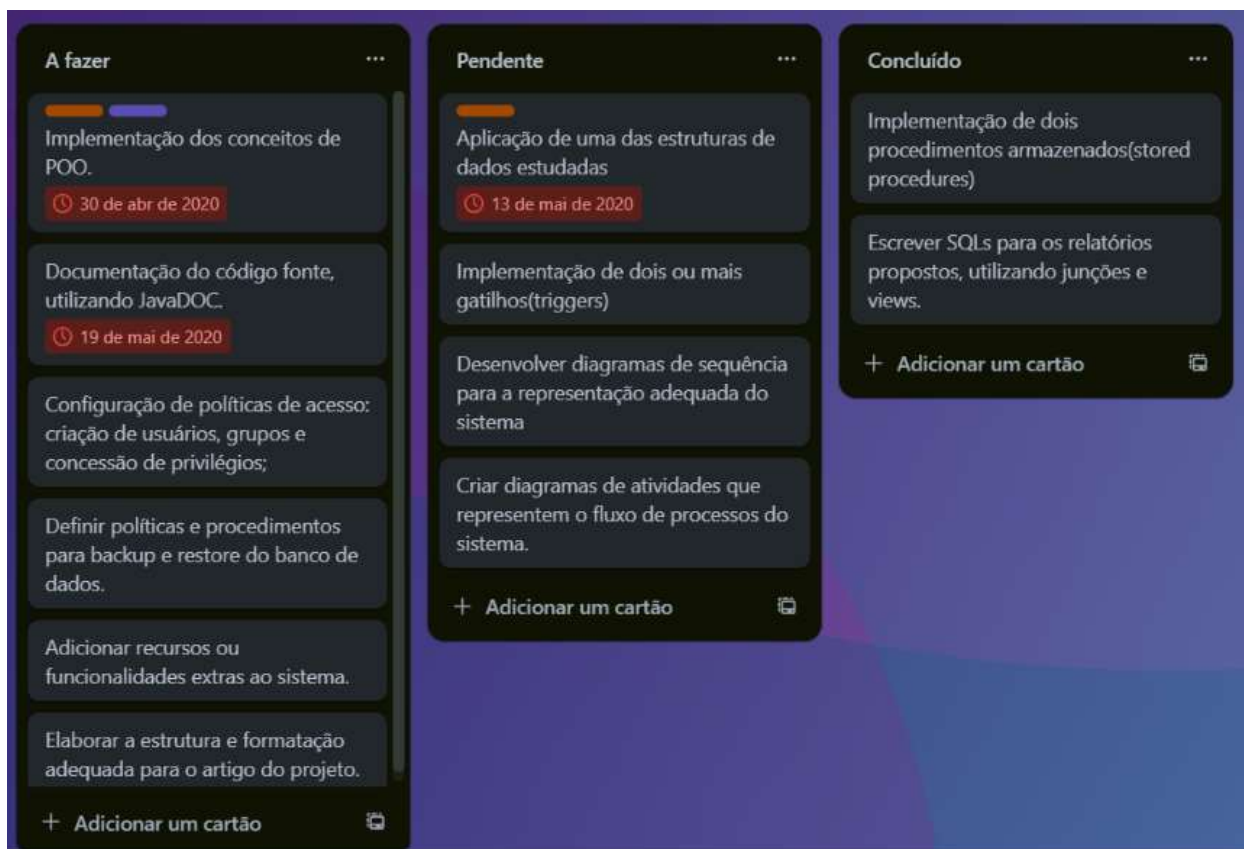
O desenvolvimento do sistema de gerenciamento de pizzarias foi organizado e monitorado usando a ferramenta de gerenciamento de projetos Trello. Esta ferramenta permitiu à equipe acompanhar o progresso, priorizar tarefas e colaborar eficientemente. As imagens a seguir ilustram o quadro do Trello em diferentes estágios do projeto, refletindo a implementação da metodologia ágil, a evolução das tarefas desde a concepção até a conclusão e as mudanças no planejamento ao longo do tempo.

Imagem 5 – Início do trabalho com trello.



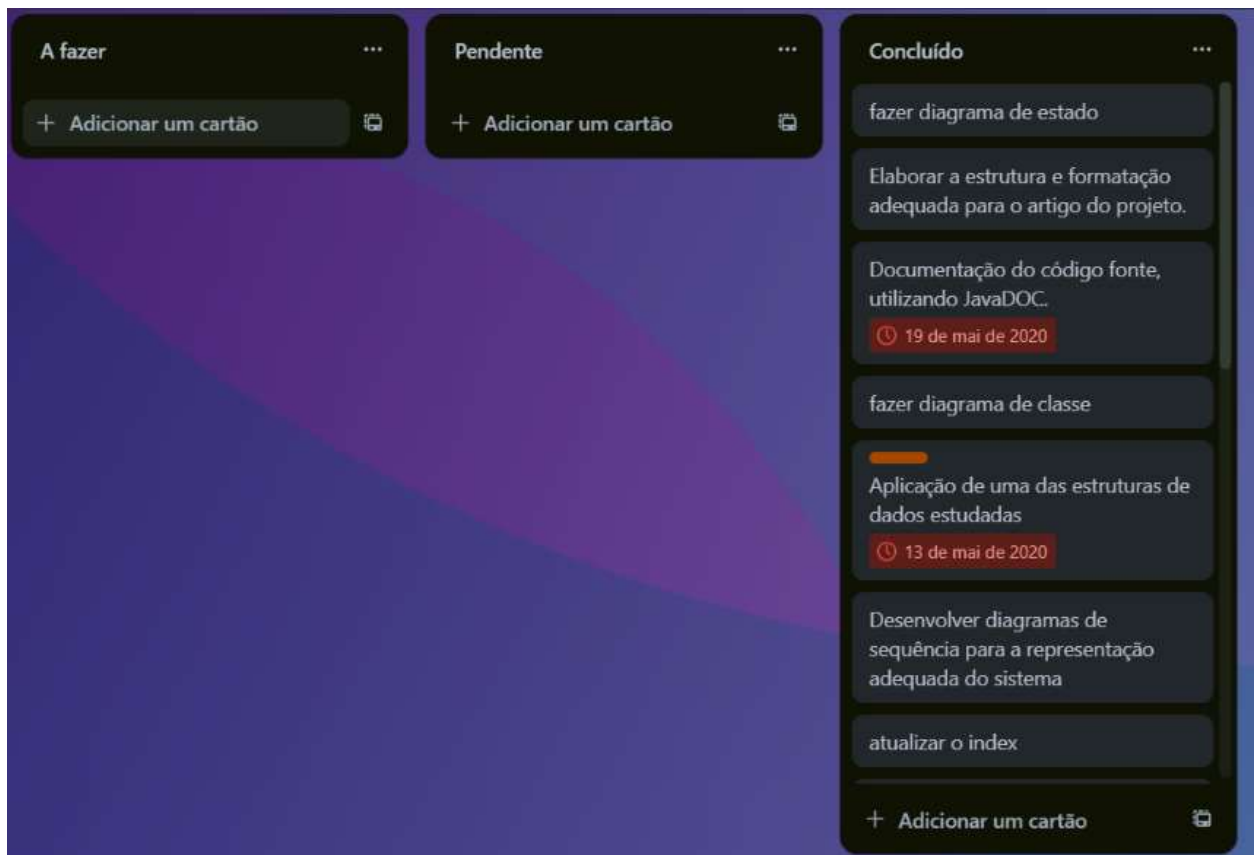
Fonte: Os autores(2023)

Imagem 6 - Última semana para a entrega.



Fonte: Os autores(2023)

Imagem 7 – Final.



Fonte: Os autores(2023)

3.4 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos foi uma etapa interna do projeto, realizada através de discussões detalhadas entre os membros do grupo. Analisamos e definimos as funcionalidades necessárias para um sistema de gerenciamento de pizzaria eficiente. Os requisitos funcionais e não funcionais foram determinados com base no conhecimento coletivo do grupo sobre processos de pizzarias e as práticas de desenvolvimento de software.

4 CONCLUSÃO

Este projeto de desenvolvimento de um sistema de gerenciamento para pizzarias, embora ainda em andamento, já demonstra um potencial significativo para transformar a operação e gestão de pizzarias. Através da implementação da página de cadastro de clientes, começamos a ver a aplicabilidade prática das tecnologias e metodologias escolhidas. Java Spring Boot, PostgreSQL, e as tecnologias de front-end, HTML, CSS e JavaScript, mostraram-se eficazes para a criação de uma solução robusta e amigável ao usuário.

Referencias

ALVES, Roberson J. F. Apostila de Banco de Dados. São Miguel do Oeste: Unoesc, 2023. Material didático em PDF.

BARBOSA, Otilia D. Spring Boot API. São Miguel do Oeste: Unoesc, 2023. Material didático em PDF.

BARBOSA, Otilia D. Parte 3 - Anotações, JavaDoc e JUnit. São Miguel do Oeste: Unoesc, 2023. Material didático em PDF.

PETRY, Franciele C. Modelagem UML: DIAGRAMA DE CLASSES. 2023. 26 slides. Apresentação de slides.

PETRY, Franciele C. Modelagem UML: DIAGRAMA DE ATIVIDADE E DE ESTADOS. 2023. 35 slides. Apresentação de slides.