Centro Universitário de Excelência
Sistemas de Informação

Desenvolvimento Web Orientado a Objetos FoodDelivery

Equipe: Paris

Autores:
Érica Araujo de Jesus
Mickael Cedraz Alencar
Monyc Luisa Almeida de Cerqueira
Nalbert de Souza Santana
Pedro César Paixão de Jesus

Feira de Santana, 2025

Visão Geral

A apresentação tem como objetivo explicar de forma clara a implementação do projeto *FoodDelivery*, destacando os conceitos de Programação Orientada a Objetos, o processo de desenvolvimento, os resultados obtidos e as conclusões práticas.

1. Resumo do Projeto:

- Qual o objetivo do sistema?
- O que ele resolve?

2. Fundamentação Teórica:

- Conceitos principais da POO (classes, atributos, métodos, construtores, diagrama de classes).

3. Desenvolvimento do Sistema:

- Estrutura do código em Java.
- Funcionalidades implementadas.

4. Conclusão e Resultados:

- O que foi alcançado.
- Principais aprendizados.
- Possíveis melhorias futuras.

Fundamentação Teórica:

A Programação Orientada a Objetos é a base do projeto *Lu Delivery*, permitindo organizar o sistema em estruturas claras e reutilizáveis. Seus principais conceitos:

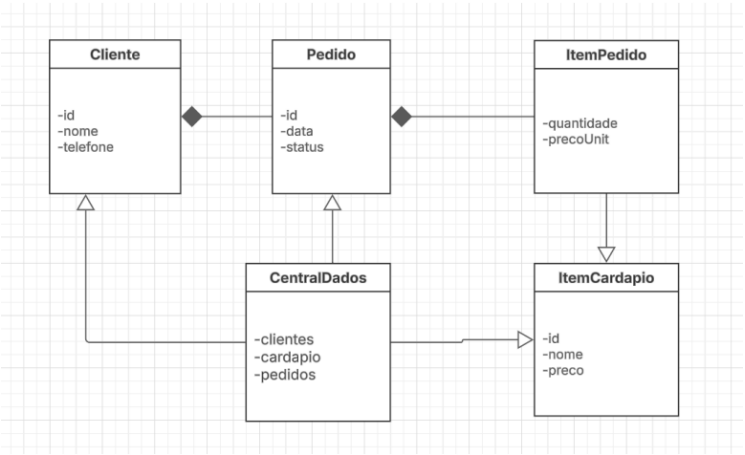
- **Classes** → moldes que definem os objetos do sistema (ex.: Cliente, Pedido, Restaurante).
- **Atributos** → variáveis internas que guardam informações dos objetos (ex.: nome, preço, endereço).
- **Métodos** → funções que representam os comportamentos (ex.: fazerPedido(), calcularTotal()).
- **Construtores** → responsáveis por criar e inicializar os objetos corretamente.
- **Diagrama de Classes** → ferramenta visual que mostra a estrutura do sistema e os relacionamentos entre suas partes.

Desenvolvimento do Sistema:

Estrutura do Código em Java

- **CentralDados** → Singleton que armazena clientes, cardápio e pedidos.
- **Cliente** → representa os usuários (nome, telefone, id).
- **ItemCardapio** → representa pratos/itens do cardápio.
- **ItemPedido** → associa item + quantidade.
- **Pedido** → representa um pedido, com status e itens.
- **ServicoCadastro** → lógica de cadastro (clientes e itens).
- **ServicoPedido** → lógica de criação e controle dos pedidos.
- **Main** → interface textual do sistema (menus e relatórios).

Diagrama de Classes



Funcionalidades Implementadas

- Cadastro de clientes e itens do cardápio.
- Consulta de clientes e cardápio.
- Criação e atualização de pedidos.
- Controle de status do pedido (fluxo ACEITO → ENTREGUE).
- Relatórios de vendas (simples e detalhado).

Conclusão e Resultados

O que foi alcançado

- Sistema funcional para simular um aplicativo de Food Delivery.
- Cadastro de clientes, cardápio e pedidos implementados com sucesso.
- Relatórios de vendas e controle de status de pedidos funcionando.

Principais Aprendizados

- Aplicação prática dos conceitos de **POO** (classes, objetos, herança, composição, encapsulamento).
- Importância da organização em **camadas** (dados, serviços, interface).
- Experiência no desenvolvimento em **Java** com uso de listas, enums e construtores.

Possíveis Melhorias Futuras

- Criar uma interface **gráfica** para tornar o sistema mais amigável.
- Implementar **persistência em banco de dados** para salvar os pedidos e clientes.
- Expandir funcionalidades: promoções, cálculo de entrega, acompanhamento em tempo real.

Referências

- **CARVALHO, Thiago Leite.** Orientação a Objetos: aprenda seus conceitos e suas aplicabilidades de forma efetiva. São Paulo: Casa do Código, 2016.

- **MORAES, Caique Vinicius de.** Programação Orientada a Objetos: domine os fundamentos da POO. São Paulo: Novatec, 2021.

- **ORACLE.** Java Platform, Standard Edition Documentation. Disponível em: https://docs.oracle.com/en/java/. Acesso em: 28 ago. 2025.

- **REFACTORING GURU.** Singleton. Refactoring Guru, s.d. Disponível em: https://refactoring.guru/pt-br/design-patterns/singleton. Acesso em: 28 ago. 2025.