Centro Universitário de Excelência
Sistemas de Informação

Sistema de gerenciamento de pedidos FoodDelivery

Autores:
Juary Dos Santos Lima Júnior
Herick Marcio Matos Brito
Lucas Ferreira Lan

Feira de Santana, 2025

Agenda

1. Estrutura Principal

2. Modelagem de Dados

3. Funcionalidades

~~Objetivo da apresentação~~
Oferecer experiência completa de gestão de produtos e pedidos, simulando funcionalidades reais de aplicativos de entrega, sistema interativo com menu principal.

Estrutura Principal

← o tamanho da fonte deveria ser maior.

Menu principal : Entrelaçado com do while com as opções:

Cadastrar Item → inclusão de novos produtos no cardápio.

Atualizar Item → modificação de informações de produtos existentes.

Criar Pedido → montagem de um novo pedido.

Atualizar Pedido → alteração do status de pedidos em andamento.

Consultar Pedidos → Exibição de pedidos realizados, com opção de filtro por status.

deveria ser um print da tela inicial

o ideal seria focar na experiência do usuário e explicar isso

Modelagem de dados

```
data class Pedido (  4 Usages  juary-junior
    var numeroPedido: Int,
    var itens: MutableList<Produto>,
    var pagamento: String,
    var status: OrderStatus,
    var valor: Float
)
```

```
data class Produto (  6 Usages  Lucas
    var nome: String,
    var descricao: String,
    var preco: Float,
    var estoque: Int,
    var codigo: Int
)
```

```
enum class OrderStatus{  13 Usages  juary-junior
    ACEITO,  2 Usages
    FAZENDO,  2 Usages
    FEITO,  2 Usages
    ESPERANDO_ENTREGADOR,  2 Usages
    SAIU_PARA_ENTREGA,  2 Usages
    ENTREGUE  2 Usages
}
```

O ideal aqui seria apresentar essa entidades com fundo branco.

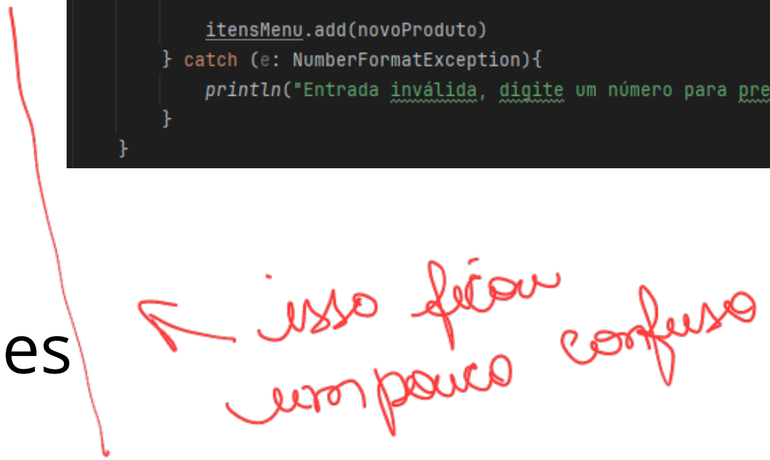
misturas de inglês e português

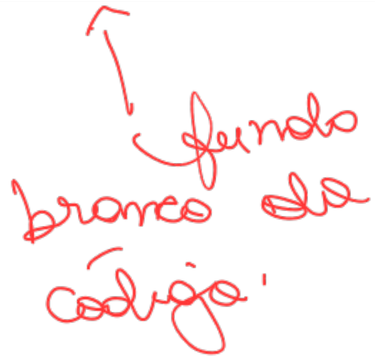
Funcionalidades

- Cadastro de itens:
  - Try catch
  - Leitura dos dados
  - Criação do produto

- Atualização de itens
  - Try catch
  - Verificações de entrada
  - Leitura dos dados
  - Atualização das informações

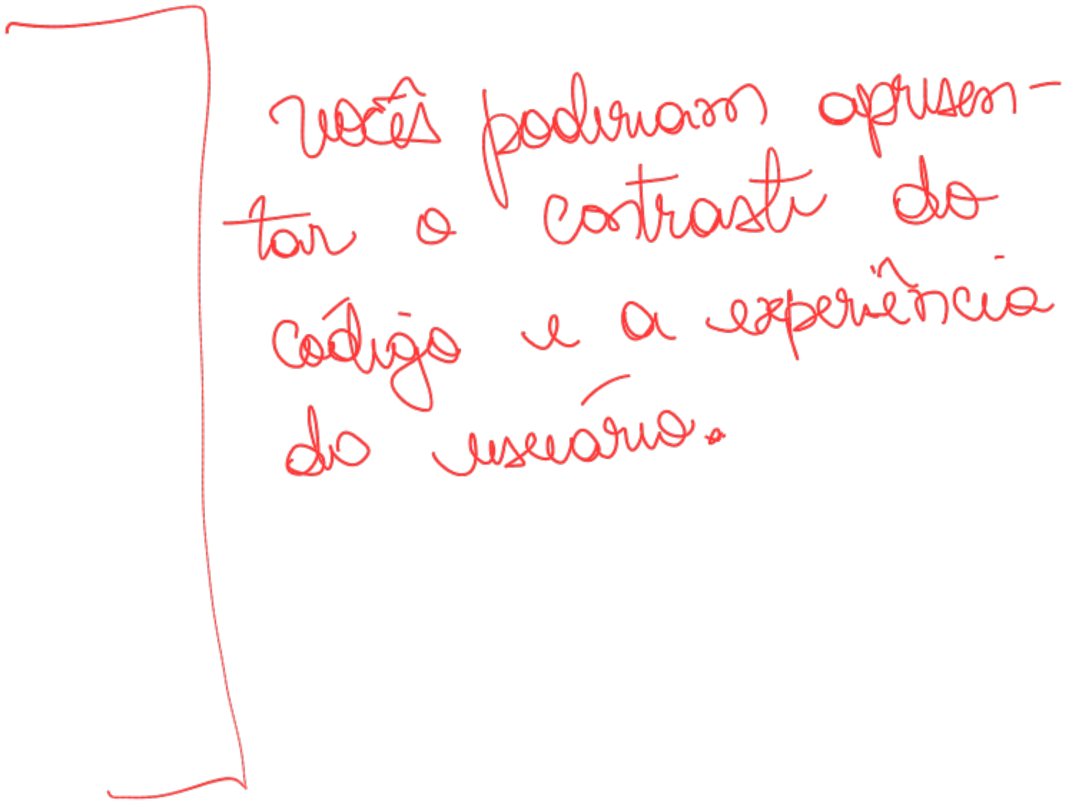
```
when(opcaoEscolhida) {
    1 -> {
        try {
            println("---CADASTRO DE ITEM---")
            println("Nome do item")
            val nomeItem: String = readln()

            println("Descrição do item")
            val descricaoItem: String = readln()

            println("Preço")
            val precoItem: Float = readln().toFloat()

            println("Quantidade em estoque")
            val estoqueItem: Int = readln().toInt()

            val novoProduto = Produto(nome = nomeItem, descricao = descricaoItem, preco = precoItem, estoque = estoqueItem,
                codigo = itensMenu.size + 1)

            itensMenu.add(novoProduto)
        } catch (e: NumberFormatException){
            println("Entrada inválida, digite um número para preço e estoque")
        }
    }
```





Funcionalidades

- Criar pedido
  - Do...while
  - Funções como adicionar itens dinamicamente, limpar carrinho, finalizar pedido e cupom de desconto
  - Volta do estoque, caso pedido seja desfeito



Funcionalidades

- Atualizar pedido
  - Verificações
  - Filtro com Find
  - Transição entre os status

```
4 -> {
    println("---ATUALIZAÇÃO DO STATUS DO PEDIDO---")

    if(pedidos.isEmpty()) {
        println("Sem pedidos por aqui\n")
    } else {
        println("Escolha o pedido para atualizar o status\n")
        println("PEDIDOS:")
        for (pedido in pedidos) {
            println("Número do pedido: ${pedido.numeroPedido} - ${pedido.status}\n - total: R$${pedido.valor}")
        }

        val codigoPedidoEscolhido: Int = readln().toInt()

        val pedidoEscolhido: Pedido? = pedidos.find {
            it.numeroPedido == codigoPedidoEscolhido
        }

        if (pedidoEscolhido == null) {
            println("Pedido não encontrado")
        } else {
            println("Deseja atualizar o pedido para qual Status? Status atual " +
                    "do pedido ${pedidoEscolhido.numeroPedido}: ${pedidoEscolhido.status}\n")

            println("1 - FAZENDO")
            println("2 - FEITO")
            println("3 - ESPERANDO ENTREGADOR")
            println("4 - SAIU PARA ENTREGA")
            println("5 - ENTREGUE")

            val statusEscolhido: Int = readln().toInt()

            val numeroPedido = pedidoEscolhido.numeroPedido
```

Deixar um pouco mais claro isso, além disso o contraste do código e da experiência era legal

Funcionalidades

- Consultar pedidos
  - do...while
  - Filtro com find e when
  - Filtragem por status

```
5 -> {
    println("---CONSULTA DE PEDIDOS---")
    var estaConsultando = true
    do {

        if(pedidos.isEmpty()) {
            println("Sem pedidos por enquanto!\n")
            break
        }

        println("Vizualizar pedidos:")
        println("1 - VER TODOS")
        println("2 - ACEITO")
        println("3 - FAZENDO")
        println("4 - FEITO")
        println("5 - ESPERANDO ENTREGADOR")
        println("6 - SAIU PARA ENTREGA")
        println("7 - ENTREGUE")
        println("8 - Sair da consulta")

        val opcaoEscolhida: Int = readln().toInt()
        var pedidosPorStatus: List<Pedido> = listOf()

        when (opcaoEscolhida) {
            1 -> {
                for (pedido in pedidos) {
                    println("Pedido ${pedido.numeroPedido} - Itens: ${pedido.itens} - VALOR: R$${pedido.valor} -
                }
            }
            2 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.ACEITO }
            3 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.FAZENDO }
            4 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.FEITO }
            5 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.ESPERANDO_ENTREGADOR }
            6 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.SAIU_PARA_ENTREGA }
            7 -> pedidosPorStatus = pedidos.filter { it.status == OrderStatus.ENTREGUE }
            8 -> {
                println("Saindo...")
                break
```

→ faltou considerações finais

Referências

KOTLIN. Conditions and loops. Disponível em: kotlinlang.org/docs/control-flow.html. Acesso em: 23 ago. 2025.

Curso de Kotlin | #00 - O primeiro contato. YouTube, 2025. Disponível em: https://www.youtube.com/watch?si=L8BsPtOFXHBkZ9Fl&v=Kd3msE3lMuc&feature=youtu.be. Acesso em: 23 ago. 2025.

ROCKETSEAT. Formação Android com Kotlin. Disponível em: rocketseat.com.br/formacao/kotlin. Acesso em: 23 ago. 2025.

Apresentação:

As descrições dos slides estão presentes. Aqui temos algumas observações:

→ A apresentação deveria focar na experiência do usuário com prints

→ Vocês também devem apresentar as estruturas de dados escolhidas e as escolhas de projeto.

→ Os slides não apresentam um número e também tem alguns problemas de estilo.

→ A validação dos dados poderiam ser feitas com do..While no próximo desafio. Criem uma função genérica para leitura do dado.

→ Vocês podem melhorar o uso do enum na opção de consulta e atualização de pedido.

A apresentação foi feita com clareza, apesar da necessidade de uma abstração alta do ouvinte

A equipe demonstrou um domínio no entendimento, compreensão e aplicação dos conceitos.

Estamos desenvolvendo um domínio na análise e avaliação dos códigos.

Estouramos o tempo da apresentação. É importante se atentar ao tempo limite!!!

A apresentação oral foi de bom para muito bom com peso de:

2.40 pontos