

- **Título:** *Sistema de Pedidos de Restaurante em Kotlin*

**Autores:** João Vítor da Conceição de Almeida,  
**André Almeida Gomes Neto,**  
**Deivid Souza Dos Santos Oliveira.**

# Agenda

- **Objetivo da apresentação:**  
Apresentar o sistema desenvolvido e seu propósito no contexto do programa trainee.
- **1. Contexto e Justificativa**  
Explicação sobre o programa da FoodDelivery e a importância do desafio proposto.
- **2. Descrição da Atividade**  
Detalhamento da tarefa principal: desenvolvimento de um sistema de pedidos em Kotlin via console.
- **3. Estrutura do Sistema Desenvolvido**  
Apresentação do menu principal, itens, pedidos e fluxo de funcionamento.

# Contexto e Justificativa



- A FoodDelivery é uma empresa de tecnologia em expansão, conectando restaurantes, bares, mercados e farmácias aos consumidores.
- Como primeira etapa, foi proposto o desenvolvimento de um sistema de pedidos em Kotlin, via console, sem banco de dados, focado em lógica de programação e no domínio da linguagem
- .A justificativa é consolidar fundamentos essenciais de programação, versionamento com Git e trabalho em equipe, para etapas mais avançadas do projeto.

# Estrutura do Sistema



- **Menu Principal com 5 opções:**
- **Cadastrar Item** – adicionar produtos ao cardápio.
- **Atualizar Item** – editar informações de itens já cadastrados.
- **Criar Pedido** – montar pedido com 1 ou mais itens.
- **Atualizar Pedido** – alterar status do pedido.
- **Consultar Pedidos** – visualizar pedidos por status.

# Funcionalidades do Sistema



- **Itens:** Nome, Descrição, Preço, Estoque, Código gerado automaticamente.
- **Pedidos:**
  - Obrigatório conter ao menos 1 item.
  - Cupom de desconto simples (percentual fixo).
  - Pagamento considerado automático.
  - Fluxo de Status: **Aceito** → **Fazendo** → **Feito** → **Aguardando Entregador** → **Saiu para Entrega** → **Entregue**.

# Data classes

```
1  enum class Status{ 14 Usages  🧑 Deivid
2      ACEITO, 2 Usages
3      FAZENDO, 2 Usages
4      FEITO, 2 Usages
5      ESPERANDO_ENTREGADOR, 2 Usages
6      SAIU_PARA_ENTREGA, 2 Usages
7      ENTREGUE 2 Usages
8  }
9
10 data class Item( 5 Usages  🧑 Andre
11     val code: Int,
12     val name: String,
13     val description: String,
14     val price: Float,
15     val amount: Int
16 )
17
18 data class Order( 2 Usages  🧑 Deivid +1
19     val code: Int,
20     var status: Status,
21     val value: Double,
22     val itens: ArrayList<Item>,
23     val discount: Boolean
24 )
```

# 1 – Cadastrar Itens

```
when (option) {  
    1 -> {  
        print("Quantos itens deseja cadastrar no sistema: ")  
        val qtd = readln().toInt()  
        for (i in 1..qtd) {  
            println("Sobre o produto $i.")  
            itemCode++  
            print("Qual o nome do produto: ")  
            val name = readln()  
            print("Qual a descricao do produto: ")  
            val description = readln()  
            print("Qual o preco do produto: ")  
            val price = readln().toFloat()  
            print("Qual a quantidade em estoque: ")  
            val amount = readln().toInt()  
            menu.add(Item(itemCode, name, description, price, amount))  
            println("Item cadastrado com sucesso, codigo: $itemCode")  
        }  
    }  
}
```

## 2 – Atualizar Itens

```
72 2 -> {
73     if (menu.isEmpty()) {
74         println("Nenhum item cadastrado para atualizar.")
75         continue
76     }
77
78     for (item in menu) {
79         println(
80             "Codigo: ${item.code}, " +
81             "Nome: ${item.name}, " +
82             "Descrição: ${item.description}, " +
83             "Preço: ${item.price}, " +
84             "Quantidade em estoque: ${item.amount}"
85         )
86     }
87
88     print("Qual o código do item que deseja atualizar: ")
89     val thisItem = readln().toInt()
90     var ishere = false
91
92     for (i in 0 until menu.size) {
93         if (menu[i].code == thisItem) {
94             print("Qual o nome do novo produto: ")
95             val name = readln()
96             print("Qual a descrição do novo produto: ")
97             val description = readln()
98             print("Qual o preço do novo produto: ")
99             val price = readln().toFloat()
100            print("Qual a quantidade em estoque do novo produto: ")
101            val amount = readln().toInt()
102            val newItem = Item(thisItem, name, description, price, amount)
103            menu[i] = newItem
104            ishere = true
105            println("Item atualizado com sucesso, código: $thisItem")
106            break
107        }
108    }
109
110    if (!ishere) {
111        println("O código informado é inválido.")
112    }
113 }
```



### 3 – Criar pedido

```
142         val item = menu.find { it.code == code }
143         if (item != null) {
144             itensOrder.add(item)
145             value += item.price
146         } else {
147             println("O código do item inserido é inválido")
148         }
149     }
150
151     print("Você deseja usar um cupom de desconto? (S/N)")
152     when (readln().uppercase()) {
153         "S" -> {
154             hasDiscount = true
155             println("Você ganhou 10% de desconto.")
156             value *= 0.90
157         }
158         "N" -> {
159             hasDiscount = false
160             println("Você escolheu não usar o cupom.")
161         }
162     }
163
164     else -> println("Opção inválida")
165 }
166
167 value = Math.round(value * 100) / 100.0
168
169 if (itensOrder.isEmpty()) {
170     println("Pedido não criado, nenhum item válido selecionado.")
171     continue
172 }
173
174 orders.add(Order(orderCode, Status.ACEITO, value, itensOrder, hasDiscount))
175 println("O pedido de código $orderCode foi aceito, valor final: R$ $value")
176 }
```

## 4 – Atualizar Pedido

```
178 4 -> {
179     if {orders.isEmpty()} {
180         println("Nenhum pedido cadastrado para atualizar.")
181         continue
182     }
183
184     for (order in orders) {
185         println(
186             "Codigo: ${order.code} " +
187             "Status: ${order.status} " +
188             "Valor: ${order.value}"
189         )
190     }
191
192     println("Qual o código do pedido a ser atualizado: ")
193     val code = readln().toInt()
194     var ishere = false
195
196     for (i in 0 until orders.size) {
197         if {orders[i].code == code} {
198             println("Qual o novo Status do pedido: ")
199             println(" 1 - FAZENDO.")
200             println(" 2 - FEITO.")
201             println(" 3 - ESPERANDO ENTREGADOR.")
202             println(" 4 - SAIU PARA ENTREGA.")
203             println(" 5 - ENTREGUE.")
204             print("Digite a opção: ")
205             when {readln().toInt()} {
206                 1 -> orders[i].status = Status.FAZENDO
207                 2 -> orders[i].status = Status.FEITO
208                 3 -> orders[i].status = Status.ESPERANDO_ENTREGADOR
209                 4 -> orders[i].status = Status.SAIU_PARA_ENTREGA
210                 5 -> orders[i].status = Status.ENTREGUE
211             }
212             ishere = true
213             println("O pedido de código $code teve seu status atualizado para ${orders[i].status}")
214             break
215         }
216     }
217
218     if (!ishere) {
219         println("O código informado é inválido.")
220     }
221 }
```

## 5 – Consultar Pedidos

```
238     val status = readln().toInt()
239     var filter: Status? = null
240     var ishere = false
241
242
243     when (status) {
244         2 -> filter = Status.ACEITO
245         3 -> filter = Status.FAZENDO
246         4 -> filter = Status.FEITO
247         5 -> filter = Status.ESPERANDO_ENTREGADOR
248         6 -> filter = Status.SAIU_PARA_ENTREGA
249         7 -> filter = Status.ENTREGUE
250     }
251
252     for (order in orders) {
253         if (filter == null || order.status == filter) {
254             println("Codigo: ${order.code}")
255             println("Status: ${order.status}")
256             println("Valor: ${order.value}")
257             println("Itens[ ")
258             for (item in order.items) {
259                 print(
260                     "Codigo: ${item.code}, " +
261                     "Nome: ${item.name}, " +
262                     "Descrição: ${item.description}, " +
263                     "Preço: ${item.price} ]"
264                 )
265             }
266             println("Desconto: ${if (order.discount) "10%" else "0%"}")
267             ishere = true
268         }
269     }
270     if (!ishere){
271         println("Nenhum pedido encontrado com este STATUS.")
272     }
273 }
274
275 0 -> {
276     println("O Sistema será encerrado")
277 }
278 }
279 } while (option != 0)
```

# Referências



- Curso em Vídeo: Git
- Minicurso de Git: Código Fonte
- Curso de Kotlin: Rapadura Dev
- Documentação oficial do Android e Kotlin