Relatório de Implementação: Sistema de Compra de Pizzas Customizáveis

1. Introdução

Este relatório descreve o desenvolvimento de um sistema de compra de pizzas customizáveis utilizando JavaFX. O sistema permite que os usuários selecionem pizzas, sabores, métodos de pagamento e revisem seus pedidos por meio de uma interface gráfica interativa. Para instalar o JavaFX SDK, faça download aqui: https://gluonhq.com/products/javafx.

2. Estrutura de Classes

2.1 Classes de Modelo

As classes de modelo representam as entidades principais do sistema e contêm a lógica relacionada aos dados e operações do pedido de pizza. Elas se mantiveram basicamente as mesmas da etapa anterior.

Order

- Responsável por armazenar informações sobre o pedido, incluindo pizzas, as informações do cliente, endereço, pagamento e se todas as informações do pedido já foram preenchidas.

Client

- Guarda informações sobre o cliente que está realizando o pedido: seu nome, sobrenome, endereço, a forma de pagamento que está sendo utilizada e se suas informações já estão registradas.

Flavour

- Representa um sabor de pizza. Ele pode ser doce ou salgado, e contém uma ArrayList de Pair que são os ingredientes que o compõem.

Address

- Representa o endereço de entrega que o cliente fornece.
    - Atributos: street, number, city, zipcode, complement.

Payment

- Representa a informação de pagamento do pedido.
    - Atributos: type, value, card (opcional).

Card

- Representa os detalhes de um cartão de crédito.
    - Atributos: number, validity, cvv, name.

2.2 Classes de armazenamento

Essas classes servem para fornecer as informações constantes que serão utilizadas: os nomes dos sabores e os preços das opções. Os métodos dentro delas servem para selecionar esses valores.

Pair

- É a base das classes de armazenamento. Ela contém uma opção (String), e um preço (double) associado a essa opção.

IngredientsMenu

- É uma classe abstrata que implementa algumas das funções utilizadas em SaltyMenu e SugaryMenu para selecionar valores
- Métodos importantes:
    - double getPrice(): dado o tipo de um ingrediente (fruta, cobertura, condimento, proteína, queijo, folhas verdes ou vegetal) e o nome de um ingrediente específico desse tipo, retorna o preço dele
    - double getFirstFromType(): dado um tipo de ingrediente e uma array list de ingredientes, retorna o primeiro ingrediente daquela array list que é do tipo informado
    - abstract Pair[] getIngredientsByType();
    - abstract String findType();

SaltyMenu

- Representa todo o menu de sabores salgados. Os sabores salgados estão divididos em 4 categorias: queijos, proteínas, vegetais e folhas verdes. Cada uma dessas categorias é uma lista constante de valores do tipo Pair.
- Métodos importantes:

    - Pair[] getIngredientsByType(): dado um tipo (proteína, queijo, folhas verdes ou vegetal), retorna todos os ingredientes desse tipo existentes, associados com seus preços.
    - abstract String findType(): dado um ingrediente, busca pelo seu tipo e o retorna (proteína, folhas verdes, queijo ou vegetal).

SugaryMenu

- Representa todo o menu de sabores doces. Os sabores doces estão divididos em 3 categorias: coberturas, condimentos e fruta. Cada uma dessas categorias é uma lista constante de valores do tipo Pair.
- Métodos importantes:
    - Pair[] getIngredientsByType(): dado um tipo (condimento, cobertura ou fruta), retorna todos os ingredientes desse tipo existentes, associados com seus preços
    - abstract String findType(): dado um ingrediente, busca pelo seu tipo e o retorna (condimento, cobertura ou vegetal)

PizzaInfo

- Guarda as opções e preços de características das pizzas além dos ingredientes disponíveis para formarem seus sabores. As 3 categorias que estão inclusas incluem os tamanhos disponíveis, o preço extra pago pela inserção de mais sabores na pizza e o custo de colocar borda recheada nela.
- Métodos importantes:
    - double getPrice(): dado uma propriedade (tamanho, borda, número de sabores) e uma opção existente para essa propriedade, retorna o preço dessa opção ou 0 se ela não existir
    - Pair[] getProperties(): dado uma propriedade (tamanho, borda, número de sabores), retorna todas as opções possíveis para ela, e os preços associados a essas opções

2.3 Classes de Controle

As classes de controle gerenciam a interação entre a interface gráfica e as classes de modelo. Todas as controllers compartilham a classe **SharedControl** para acesso centralizado ao pedido.

SharedControl

- Responsável por manter uma referência única ao pedido em andamento e compartilhá-lo entre diferentes controladores. Para garantir isso, a classe SharedControl possui uma instância de si mesma como atributo de classe, e todos os controladores acessam essa mesma instância.

SceneNavigator

- Responsável por administrar a navegação entre telas.

InitialController

- Controla a tela inicial.
    - Ações: adicionar nome e sobrenome do cliente, iniciar um novo pedido.

ChoosePizzaController

- Controla a tela de escolha do tamanho, do número de sabores e da borda da pizza.
    - Ações: selecionar o tamanho da pizza, sua borda e seu número de sabores. Atualizar dinamicamente o preço total do pedido conforme as opções vão sendo selecionadas. Guardar os valores selecionados na pizza sendo montada ao avançar para a escolha de sabores.

ChooseFlavorController

- Classe abstrata que serve como base para a implementação dos controladores da tela de escolha de sabores doce e da tela de escolha de sabores salgados, implementando funções comuns a elas.

ChooseSaltyFlavourController

- Controla a tela de escolha de ingredientes para a confecção de um sabor salgado.

    - Ações: possibilitar a troca para a escolha de um sabor doce. Escolher os ingredientes do sabor salgado. Exibir dinamicamente o preço total do pedido conforme as opções vão sendo selecionadas. Possibilitar a ida para um próximo sabor bem como a volta para uma tela anterior e edição de escolhas. Ao sair da tela com algum ingrediente selecionado, atualizar a pizza que está sendo confeccionada colocando o sabor montado nela e atualizar o preço do pedido.

ChooseSugaryFlavourController

- Controla a tela de escolha de ingredientes para a confecção de um sabor doce.
    - Ações: possibilitar a troca para a escolha de um sabor doce. Escolher os ingredientes do sabor salgado. Atualizar dinamicamente o preço total do pedido conforme as opções vão sendo selecionadas. Possibilitar a ida para um próximo sabor bem como a volta para uma tela anterior e edição de escolhas. Ao sair da tela com algum ingrediente selecionado, atualizar a pizza que está sendo confeccionada colocando o sabor montado nela e atualizar o preço do pedido.

PaymentController

- Controla a tela de escolha de pagamento.
    - Ações: selecionar o método de pagamento, inserir dados do endereço, inserir dados do cartão (se aplicável), navegar para a revisão do pedido ou voltar para a edição dos sabores. Ao sair da tela, atualizar 0os valores do endereço e do pagamento do cliente conforme as informações fornecidas.

ReviewController

- Controla a tela de revisão do pedido.
    - Ações: exibir o pedido completo: todas as pizzas com seus sabores e preços além dos dados do cliente. Permitir a exclusão de pizzas do pedido. Permitir a edição de pizzas já confeccionadas. Finalizar o pedido ou voltar para a tela de escolha de pagamento.

FinalController

- Controla a tela final.
    - Ações: exibir mensagem de confirmação, navegar de volta à tela inicial e resetar o pedido, excluindo todas as informações existentes sobre o cliente

2.4 Extra: classe de implementação de componente de interface

ChangeableButton

- Implementa um botão que pode ser selecionado e “desselecionado” a partir da classe RadioButton do JavaFX.

3. Comparação com a etapa 1

3.1 Requisitos

RF-1: O cliente consegue personalizar até 4 sabores em uma pizza. Para cada sabor, ele pode escolher até 1 ingrediente em cada uma das categorias disponíveis para aquela opção de sabor (para sabores doces, as categorias são condimento, fruta, cobertura; para sabores salgados, as categorias são proteína, vegetal, folhas verdes e queijo)

RF-2: O sistema exibe todas as possibilidades possíveis de ingredientes para a opção de sabor escolhida (doce ou salgado) e permite que o cliente troque entre essas opções a qualquer momento

RF-3: Não só o sistema mostra o preço atualizado conforme o cliente adiciona os ingredientes, como também mostra na escolha dos atributos da pizza (tamanho, número de sabores e se tem borda)

RF-4: O cliente pode escolher entre pagar com dinheiro, cartão ou pix. Se escolher pagar com cartão, deve preencher as informações dele. Se escolher pagar com pix, um link será mostrado a ele na tela de revisão

RF-5: O sistema está limitando o cliente a 5 pizzas, mas essa é apenas uma limitação para cumprir o requisito. Na verdade, o sistema foi desenvolvido para que não haja limitação no número de pizzas em um pedido.

RNF-1: O app foi desenvolvido para isso.

RNF-2: O tempo de resposta não excede 1 segundo.

RNF-3: Após o pedido ser finalizado, todos os dados do cliente são apagados e eles não são salvos em nenhum lugar.

RNF-4: O sistema conta com um esquema de cores e botões para a interação bastante intuitivo.

RNF-5: Para adicionar novos ingredientes ou modificar o preço de algum deles, basta acrescentá-los nas listas de SaltyMenu ou SugaryMenu.

3.2 Classes e suas relações

Mudanças nas Classes de Modelo

As classes de modelo passaram por alterações pontuais em comparação à etapa 1, com a principal mudança relacionada à estrutura de armazenamento dos sabores e ingredientes. Antes, cada pizza possuía uma ArrayList de sabores, sendo esses sabores instâncias de uma classe abstrata (herdada pelas classes Doce e Salgado). Os ingredientes eram representados como ArrayList de *strings*.

Agora, a modelagem foi simplificada e aprimorada:

- A classe Flavour (Sabor) tornou-se concreta, contendo uma ArrayList de pares ingrediente-preço (Pair), que facilita o gerenciamento dos dados e reduz a complexidade.
- A criação e atribuição desses pares são feitas através de métodos definidos nas classes IngredientsMenu (abstrata), SaltyMenu e SugaryMenu. Essas classes também armazenam os valores fixos dos ingredientes.

Essa reformulação torna o sistema mais coeso e facilita a expansão futura.

Reestruturação nas Classes de Controle

As classes de controle passaram por mudanças significativas. A abordagem inicial previa uma classe de estado para cada tela, com uma classe abstrata definindo um padrão para esses estados, enquanto o pedido (Order) seria responsável por armazenar o estado atual.

Essa lógica foi alterada para simplificar o sistema:

- Foi criada uma classe de controle compartilhada, SharedControl, que registra o pedido por meio de um atributo estático acessado por todos os controladores.
- A classe abstrata para estados individuais não foi implementada, pois os controladores das telas apresentavam estruturas muito diferentes. A única exceção foram os controladores das telas de escolha de ingredientes:
    - ChooseSugaryFlavourController (sabores doces) e ChooseSaltyFlavourController (sabores salgados) compartilham métodos similares, o que motivou a criação da classe abstrata ChooseFlavourController.

Nova Implementação: ChangeableButton

Uma nova classe, não prevista na fase de planejamento, foi implementada para atender a uma necessidade específica da interface: ChangeableButton. Esse botão permite ao usuário selecionar e "deselecionar" opções de forma intuitiva, aprimorando a experiência de interação.