**RELATÓRIO (MODELO)**

**Responsável: André Cararo Lowcke**

Nesse relatório, encontram-se as propostas de melhorias no código do projeto de Supermercado.

**SUGESTÕES DA EQUIPE DE TESTE**

Abaixo, encontram-se os problemas identificados pela equipe de teste e as sugestões de melhorias propostas pelo responsável.

|  |
| --- |
| **SUGESTÃO #1: Nomeação adequada e legibilidade (Exemplo)** |
| **Arquivo:** CarrinhoDeCompras.java  **Código atual:**    **Sugestão de melhoria:**    **Justificativa**: os parâmetros “p” e “q” foram renomeados para “preço” e “quantidade” com o objetivo de deixar o código mais claro sobre o que está acontecendo. Além disso, o método foi simplificado para deixa-lo mais curto e melhorar a legibilidade. |

|  |
| --- |
| **SUGESTÃO #2: Métodos duplicados** |
| **Arquivo:** Cliente.java  **Código atual:**    **Sugestão de melhoria:**    **Justificativa**: Manter métodos duplicados pode ser justificável quando há a necessidade de garantir compatibilidade com APIs externas ou sistemas legados, evitando que alguma integração que depende de um nome específico pare de funcionar. Além disso, pode fazer sentido em termos de clareza dentro do código, já que um método como getNomeCliente() explicita que se trata do nome do cliente, evitando ambiguidades em sistemas que lidam com múltiplos tipos de nomes, como de funcionários ou produtos. Em algumas situações, a duplicação pode até facilitar a manutenção, deixando a intenção do código mais evidente para quem o utiliza. |

|  |
| --- |
| **SUGESTÃO #3: Substituição de atributo por objeto** |
| **Arquivo:** Pedido.java  **Código atual:**    **Sugestão de melhoria:**  **package supermercado;**  **public class Pedido {**  **private int numeroPedido;**  **private CarrinhoDeCompras carrinho;**  **private Cliente cliente; // Substituição feita aqui**    **public int getNumeroPedido() {**  **return this.numeroPedido;**  **}**  **public String getNomeCliente() {**  **return cliente.getNome();**  **}**  **public String getCpfCliente() {**  **return cliente.getCpf();**  **}**  **public String getEmailCliente() {**  **return cliente.getEmail();**  **}**  **public void fecharPedido() {**  **System.out.println("Numero do pedido: " + numeroPedido);**  **System.out.println("Cliente: " + cliente.getNome());**  **System.out.println("Total do pedido: " + this.carrinho.calcularTotal());**  **System.out.println("=====================================");**  **}**  **public CarrinhoDeCompras getCarrinho() {**  **return carrinho;**  **}**  **public void setCarrinho(CarrinhoDeCompras carrinho) {**  **this.carrinho = carrinho;**  **}**  **public void setNumeroPedido(int numeroPedido) {**  **this.numeroPedido = numeroPedido;**  **}**  **public Cliente getCliente() {**  **return cliente;**  **}**  **public void setCliente(Cliente cliente) {**  **this.cliente = cliente;**  **}**  **}**  **Justificativa**: **vita duplicação de dados**: Os dados do cliente já estão representados na classe Cliente. Repetir isso em Pedido é redundante. **Facilita manutenção**: Qualquer mudança na estrutura de Cliente (ex: adicionar um telefone) só precisa ser feita na classe Cliente, não em Pedido. **Reflete melhor o domínio real**: Um pedido pertence a um cliente, então faz sentido que Pedido tenha um objeto Cliente, e não apenas campos soltos. **Organização orientada a objetos**: Torna o código mais coeso, limpo e alinhado aos princípios de **POO**. |

|  |
| --- |
| **SUGESTÃO #4: Espaços em branco e recuo** |
| **Arquivo:** Supermercado.java  **Código atual:**    **Sugestão de melhoria:**  **package supermercado;**  **public class Supermercado {**  **// Teste**  **public static void main(String[] args) {**    **// Carrinho**  **CarrinhoDeCompras c = new CarrinhoDeCompras();**  **c.adicionarProduto("Arroz", 1, 3.99);**  **c.adicionarProduto("Filé de frango sassami", 2, 21.99);**  **c.calcularTotal();**  **// Pedido**  **Pedido p = new Pedido();**  **p.setNumeroPedido(1);**  **p.setCarrinho(c);**  **p.fecharPedido();**  **}**  **}**  **Justificativa**: Foi aplicado o conceito de **espaços em branco e recuo** no código para melhorar a **legibilidade, organização e profissionalismo**. A **indentação com recuo de 4 espaços** foi utilizada conforme o padrão Java, especialmente dentro do método main(), facilitando a leitura das instruções. Foram inseridos **espaços em branco entre blocos lógicos**, como entre a criação do carrinho, o cálculo do total e a criação do pedido, permitindo uma melhor visualização das etapas do processo. Além disso, os **comentários foram iniciados com letra maiúscula**, seguindo uma convenção comum de escrita mais clara e profissional.  Essas boas práticas são importantes porque tornam o código mais **fácil de entender rapidamente (legibilidade)**, facilitam sua **manutenção futura**, seja por você ou por outros desenvolvedores, e demonstram um **nível de cuidado esperado em ambientes profissionais e acadêmicos**. |

|  |
| --- |
| **SUGESTÃO #5: Comentários do código** |
| **Arquivo:** Supermercado.java  **Código atual:**    **Sugestão de melhoria:**  package supermercado;  public class Supermercado {  public static void main(String[] args) {    // Criação do carrinho de compras e adição de produtos  CarrinhoDeCompras c = new CarrinhoDeCompras();  c.adicionarProduto("Arroz", 1, 3.99);  c.adicionarProduto("Filé de frango sassami", 2, 21.99);  // Cálculo do total dos produtos no carrinho  c.calcularTotal();  // Criação do pedido com número e carrinho associado  Pedido p = new Pedido();  p.setNumeroPedido(1);  p.setCarrinho(c);  // Finalização do pedido e exibição dos dados  p.fecharPedido();  }  }  **Justificativa**:  Os comentários do código foram melhorados para deixar de ser genéricos, como "teste" ou "carrinho", passando a descrever de forma clara o que está acontecendo em cada bloco. Foram utilizados **verbos de ação**, como "Criação", "Adição" e "Finalização", para indicar a intenção de cada trecho, tornando o entendimento mais direto. Além disso, os comentários foram **alinhados ao contexto do código**, o que facilita a leitura e compreensão por qualquer pessoa que venha a analisar ou manter o sistema. |

|  |
| --- |
| **SUGESTÃO #6: Nomeação adequada** |
| **Arquivo:** Supermercado.java  **Código atual:**    **Sugestão de melhoria:package supermercado;**  **public class Supermercado {**  **public static void main(String[] args) {**    **// Criação do carrinho de compras e adição de produtos**  **CarrinhoDeCompras carrinho = new CarrinhoDeCompras();**  **carrinho.adicionarProduto("Arroz", 1, 3.99);**  **carrinho.adicionarProduto("Filé de frango sassami", 2, 21.99);**  **// Cálculo do total dos produtos no carrinho**  **carrinho.calcularTotal();**  **// Criação do pedido com número e carrinho associado**  **Pedido pedido = new Pedido();**  **pedido.setNumeroPedido(1);**  **pedido.setCarrinho(carrinho);**  **// Finalização do pedido e exibição dos dados**  **pedido.fecharPedido();**  **}**  **}**  **Justificativa**: No código da classe Supermercado, foram aplicadas **nomeações adequadas** aos identificadores, seguindo as convenções de nomenclatura da linguagem Java e boas práticas de clareza semântica. A variável c foi renomeada para carrinho e p para pedido, pois nomes curtos e genéricos dificultam o entendimento, especialmente em códigos maiores. Com nomes descritivos, o código se torna mais **autoexplicativo**, reduzindo a necessidade de comentários excessivos e facilitando a **manutenção e leitura** por parte de outros desenvolvedores. Nomear variáveis com clareza é essencial para que o propósito de cada objeto seja compreendido rapidamente dentro do contexto do programa. |

**SUGESTÕES DO DESENVOLVEDOR**

Abaixo, encontram-se os problemas identificados e as sugestões de melhorias propostas pelo responsável.

|  |
| --- |
| **💡** Esse é um espaço reservadopara sugestões extras. Se esse espaço estiver em branco, nenhuma possível melhoria foi identificada. |

|  |
| --- |
| **SUGESTÃO #7:** |
| **Arquivo:**  **Código atual:**  **Sugestão de melhoria: criação da classe PedidoView, responsável apenas pela exibição dos dados do pedido, separando assim a lógica de visualização da lógica de negócio.**  **Justificativa**:  Pedido agora contém **apenas dados e regras de negócio**.PedidoView contém a **responsabilidade exclusiva da apresentação**.Supermercado orquestra tudo de forma clara e modular. |