SOLID / Design Patterns

Foram utilizados 3 Design Patterns no projeto, dentre eles, Strategy, Factory e Singlethon. Strategy foi utilizado, pois percebeu-se que as funcionalidades de criar promoções, adicionar frete e taxa extra eram muito similares, pois alteravam o preço final do produto. Com esse padrão de projeto, diminuimos a quantidade de if statements e conseguimos desacoplar as classes, fazendo o uso do princípio Aberto/Fechado, pois é possível adicionar novas funcionalidades que alteram o preço final do produto sem a necessidade de alterar as implementações existentes.

Já o Factory foi utilizado para, a partir da entrada de dados do usuário, criar o manipular de repositório adequado. Foi necessária sua utilização, pois não se sabe de antemão quais dependências e objetos serão utilizados, pois variam a depender da entrada do usuário.Há a utilização do princípio Aberto/Fechado, pois novos tipos de entradas de dados podem ser adicionadas, sem que alterações no código já desenvolvido sejam necessárias, além do princípio da Responsabilidade Única, onde cada classe de manipulação de repositório é responsável por recuperar os dados da sua maneira. Por fim, foi utilizado Singlethon para garantir que só há uma instância do manipulador de repositório que pega os dados em memória.