Feature 3: Programação Orientada a Objetos

Nesta feature, seu projeto ganhará estrutura e organização, elevando a maneira como vocês pensam e constroem software. Aqui, vocês deverão modelar o "mundo" do seu projeto utilizando os pilares da Programação Orientada a Objetos (POO): classes e objetos.

1. Introdução à POO

- **Objeto e classe**

À medida que o projeto avança, o negócio de vocês também evolui. Para representar essa complexidade, criaremos uma **nova classe** que tenha um relacionamento significativo com uma classe já existente. Isso demonstra como diferentes partes do negócio se interligam e como a POO nos ajuda a modelar essas conexões.

- **Novos tipos e atributos:** além dos tipos primitivos e `String`, utilizaremos:

Enum: para representar um conjunto fixo de constantes nomeadas; r**elacionamentos:** classes se relacionarão entre si. Um atributo de uma classe pode ser um objeto de outra classe. Isso é fundamental para que as classes façam mais sentido e representem o negócio de forma integrada; **coleção:** para representar múltiplos objetos relacionados.

- **Métodos**

Criem métodos que recebam parâmetros para realizar ações mais específicas e flexíveis. Implementem o método `toString()` nas suas classes para fornecer uma representação textual significativa do objeto, facilitando a depuração e exibição de informações no console. Utilizem métodos de classes padrão do Java que podem apoiar a lógica de negócio do seu projeto. Implementem a sobrecarga de métodos, criando múltiplos métodos com o mesmo nome, mas com diferentes listas de parâmetros (número, tipo ou ordem dos parâmetros). Isso permite que um método realize ações semelhantes de maneiras ligeiramente diferentes, dependendo dos dados fornecidos.

- **Construtores**

Desenvolvam tanto o construtor padrão (sem argumentos) quanto construtores que recebem parâmetros para inicializar objetos com um estado inicial específico. Explorem a possibilidade de um construtor chamar outro construtor da mesma classe para evitar duplicação de código. Demonstrem no seu código como objetos podem ser instanciados de diversas maneiras, utilizando os diferentes construtores criados, refletindo as necessidades de inicialização do seu sistema.

- **Encapsulamento**

O uso de Getters e Setters é fundamental para controlar como os atributos de um objeto são acessados e modificados, sendo o primeiro passo para o encapsulamento, um dos pilares da POO. Eles permitem validar dados antes de atribuí-los e manter a integridade interna do objeto.

- **Testando o comportamento:**

No método `main` da sua aplicação, instanciem os objetos das classes criadas e invoquem seus métodos para testar seus comportamentos em diversos cenários. Isso valida se a lógica da classe está funcionando corretamente e se os objetos se comportam conforme o esperado, especialmente com a criação de novos relacionamentos e tipos de atributos.