Reflexões Individuais - Seminário POO em C#

Membro 1

Durante esse trabalho percebi como o conceito de encapsulamento é fundamental para proteger os dados de uma aplicação. Antes, eu tinha a ideia de que bastava declarar atributos como `public` e usá-los livremente, mas isso abre espaço para erros difíceis de rastrear. Ao projetar a classe ContaFinanceira, vi que as invariantes funcionam como guardiãs do estado, evitando que valores inválidos sejam inseridos. Minha principal aprendizagem foi entender que não é apenas sobre “fazer o código rodar”, mas sim garantir consistência a longo prazo.

Membro 2

O ponto que mais me marcou foi o uso de exceções como mecanismo de comunicação de erro. No começo, eu tendia a pensar nelas como algo “pesado” ou “indesejado”, mas no projeto vi como tornam o código mais expressivo. Quando uma regra de negócio é quebrada, a exceção deixa isso evidente e impede que o erro seja ignorado silenciosamente. Também percebi a diferença entre tratar regras de negócio no próprio objeto (orientação a objetos) e deixar a validação espalhada no código (estruturado). Essa comparação me mostrou como a abordagem OO gera mais clareza e segurança.

Membro 3

O aprendizado mais relevante para mim foi perceber os limites de arrays simples dentro de uma classe. Ao modelar gastos ou lançamentos, tivemos que pensar em como manter a simplicidade sem perder clareza. Isso reforçou a ideia de que, mesmo em um escopo restrito, ainda é possível aplicar boas práticas de design. Além disso, ficou claro que os objetos não servem apenas para armazenar dados, mas para proteger regras e guiar o uso correto do sistema. Acredito que essa experiência vai me ajudar a escrever código mais robusto em projetos futuros.