GUSTAVO BERTI RA:22247815-2

A criação e estruturação de camadas em um projeto de software é um aspecto fundamental para a organização e manutenção do código. Mais do que simplesmente dividir o código em pastas, as camadas são definidas pelo relacionamento entre os componentes, visando facilitar não apenas a compreensão, mas também a escalabilidade e a testabilidade do sistema.

É crucial entender que, ao iniciar um projeto próprio, ganhamos valiosa experiência ao compreender a arquitetura desde o início. Muitas vezes, porém, a pressa em começar nos leva a pular etapas essenciais, como a realização de testes e a profunda compreensão do domínio do problema a ser resolvido. Esta abordagem pode comprometer negativamente o resultado final do projeto.

A aplicação de camadas na programação é essencial para isolar e conduzir diferentes tecnologias, evitando a mistura de responsabilidades e aumentando as chances de sucesso do projeto. Exemplos do passado mostram como a mistura de responsabilidades, como manipulação de interface gráfica, regras de negócio e acesso a banco de dados, resultava em sistemas complexos e difíceis de manter.

A mistura de responsabilidades no desenvolvimento de software pode causar problemas de execução e dificuldades de depuração. Por isso, separar responsabilidades e criar testes adequados são práticas essenciais para aumentar a produtividade e evitar erros no desenvolvimento.

Um dos desafios enfrentados é a distribuição de valores em sistemas de transações, que pode se tornar complexa ao misturar regras de negócio com operações de banco de dados. Separar essas responsabilidades de forma clara e precisa é crucial para facilitar testes e garantir a precisão dos resultados.

Quando o princípio de responsabilidade única é quebrado, surgem problemas de acoplamento indesejado entre diferentes partes do sistema, dificultando a manutenção e a evolução do código ao longo do tempo.

Para lidar com esses desafios, a adoção de uma Clean Architecture se mostra essencial. Separar as camadas de uma aplicação, evitando acoplamento e garantindo testabilidade e abstração adequada, facilita a manutenção e evita regras de negócio perdidas em camadas incorretas.

Além disso, o desacoplamento de camadas e a aplicação de padrões de design sólidos são cruciais para facilitar a manutenção e escalabilidade do sistema, tornando-o mais flexível e aberto para extensões.

Testar todas as camadas da aplicação é fundamental para garantir a qualidade do software, com ênfase na testagem no nível de unidade para garantir a consistência na aplicação.

Em última análise, desenvolver aplicações com potencial de manutenibilidade é essencial para garantir um ritmo sustentável de desenvolvimento responsável. Isso envolve uma estruturação adequada da aplicação, uma equipe saudável e estável, e um compromisso contínuo com o aprimoramento profissional.