Semana 13: Linguagens de Programação Back-end

- TDD e Testes Automatizados

Esta semana aborda a importância do Test-Driven Development (TDD) e dos testes automatizados no desenvolvimento back-end, focado em garantir a qualidade e robustez das aplicações. Aulas exploram a introdução ao TDD, a aplicação de testes de integração para verificar o fluxo de dados entre componentes e a relevância dos testes de performance para assegurar a escalabilidade e responsividade dos sistemas.

Estrutura com Temas e Subtemas

Semana 13: Linguagens de Programação Back-end

TDD e Testes Automatizados

- **Aula 1: Introdução ao TDD**
    - **Definição e Princípios do TDD:**
        - *test-Driven Development* (TDD) é uma prática que ajuda a reduzir erros ao priorizar a escrita de testes antes da implementação de código.
        - Ciclo Red-Green-Refactor: Escrever um teste que falha (Red), escrever o código mínimo para o teste passar (Green), e refatorar o código (Refactor).
        - Benefícios: Melhoria na qualidade do código, documentação implícita, detecção precoce de bugs.
    - **Exemplo Prático (Conceitual):**
        - Desenvolvimento de uma função para somar dois números.
        - Passo 1: Escrever um teste que espera que `soma(2, 3)` retorne `5`. O teste falha.
        - Passo 2: Implementar a função `soma(a, b)` que retorna `a + b`. O teste passa.
        - Passo 3: Refatorar (se necessário), garantindo que o código esteja limpo e otimizado.
    - Código da aula: [SIS]ANO2C2B2S13A1

- **Aula 2: Testes de Integração**
    - **Conceito e Finalidade:**
        - Verificar a comunicação e o fluxo de dados entre diferentes módulos ou componentes de um sistema (ex: API, banco de dados, serviços externos).
        - Garantir que as partes do sistema funcionam corretamente em conjunto.
    - **Tipos de Testes de Integração:**
        - Testes de integração de banco de dados.
        - Testes de integração de API.
        - Testes de integração com serviços de terceiros.
    - **Exemplo Prático (Conceitual):**
        - Um sistema de e-commerce com módulos de usuário, produto e pedido.
        - Teste de integração: Verificar se, ao criar um pedido, o sistema corretamente:
            - Cadastra o pedido no banco de dados.
            - Associa os produtos corretos.
            - Atualiza o estoque dos produtos.
            - Envia uma notificação ao usuário.
    - Código da aula: [SIS]ANO2C2B2S13A2

- **Aula 3: Testes de Performance**
    - **O que são e Por que são Importantes:**
        - Avaliar a velocidade, escalabilidade e estabilidade de uma aplicação sob diferentes cargas de trabalho.
        - Identificar gargalos e prever o comportamento do sistema em situações de alta demanda.
    - **Métricas Comuns:**
        - Tempo de resposta, taxa de transferência (throughput), utilização de recursos (CPU, memória), número de usuários concorrentes.
    - **Tipos de Testes de Performance:**
        - Teste de carga: Simular um número esperado de usuários.
        - Teste de estresse: Levar o sistema ao limite para encontrar o ponto de falha.
        - Teste de escalabilidade: Avaliar como o sistema se comporta ao aumentar a carga.

- **Exemplo Prático (Conceitual):**
  - Um endpoint de API de busca de produtos em um e-commerce.
  - Teste de performance:
    - Simular 1000 requisições simultâneas ao endpoint.
    - Medir o tempo médio de resposta.
    - Verificar a utilização da CPU do servidor.
    - Observar se há erros ou lentidão excessiva.
- Código da aula: [SIS]ANO2C2B2S13A3
