***Plano de Aula*: Code Smells e Refatoração**:

***Dados de Identificação*:**
Professor (a):
Aluno PAE:
Disciplina:
Turma:
Período:

Tema: Ensino de identificação de code smells e refatoração de código

Objetivo geral: Entendimento do conceito de code smells, clean code e refatoração.

Objetivos específicos:

ao nível de conhecimento – Identificar code smells em códigos orientado a objetos; identificar características que caracterizem os code smells; entender possíveis soluções de refatoração para os code smels identificados; Avaliar se possíveis soluções proposta por ferramentas de IA são adequadas para os code smells identificados.

ao nível de aplicação – importar projetos do GitHub para aplicação local; analisar o código usando ferramentas como Jdedorant e Project Usus; criar prompts de interação com gpt para identificar code smells e refatoração.

ao nível de solução de problemas – analisar o código e identificar code smells, julgar se as ferramentas identificaram code smells adequadamente; propor soluções para os problemas encontrados nos códigos, julgar se as soluções propostas pelas ferramentas são adequadas.

Conteúdo:

Aula 1 - Introdução à clean code, code smells, refatoração e dívida técnica. Identificando code smells e técnicas de refatoração de código.

Aula 2 – Code Smells (Alternative Classes with Different Interfaces, Comments, Data Class, Data Clumps, Divergent Change, Duplicated Code, Feature Envy, Inappropriate Intimacy, Incomplete Library Class, Large Class, Lazy Class), apresentando a descrição do code smell, razões porque acontecem e exemplos.

Aula 3 – Code Smells (Long Method, Long Parameter List, Message Chains, Middle Man, Parallel Inheritance Hierarchies, Primitive Obsession, Refused Bequest, Shotgun Surgery, Speculative Generality, Switch Statements, Temporary Field), apresentando, descrição do code smell, razões porque acontecem e exemplos.

Aula 4 – Exercícios de identificação inicial de code smells, manualmente, usando ferramentas como JDeodorant e IA.

Aula 5 - Estratégias de refatoração para cada code smells, com possíveis tratamentos. Exercícios de refatoração, manualmente e usando IA.

Aula 6 - Estratégias de refatoração para cada code smells, com possíveis tratamentos. Exercícios de refatoração, manualmente e usando IA.

Aula 7 – Introdução a métricas de software.

Recursos didáticos: Quadro, projetor, computador e softwares específicos.

Avaliação: realização das atividades descrita no Anexo I e II

Anexo I – Trabalho – Parte 1

O objetivo deste trabalho é o entendimento mais profundo sobre *code smells* e refatoração, trabalhando tanto no nível conceitual quanto prático.

O trabalho consiste em analisar o projeto atribuído e identificar *code smells* e refatorá-lo. Para tanto, os seguintes passos devem ser seguidos:

1) Realizar a análise do código para encontrar *code smells*. Essa análise pode ser feita por meio de ferramenta, como o JDeodorant e IA ou manualmente ou combinando as abordagens.
    a. Realizar um relatório detalhado dos codes smells identificados. Qual o code smell, ferramenta utilizada. Se a identificação não foi feita de forma manual, descrever se concorda ou não com a identificação do code smell.
2) Para os *code smells* identificado refatorar o código. Essa etapa deve ser feita usando algum GPT, e se necessário também manual ou com auxílio de ferramentas.
    a. Descrever como o code smell foi refatorado e se a técnica utilizada resolve o problema ou não. Caso tenha sido refatorado por alguma ferramenta, analisar se a refatoração foi eficiente. Caso tenha utilizado mais de uma ferramenta para a mesma refatoração, comparar as duas.
3) Fazer um relatório do processo, identificando os *code smells*, a técnica de refatoração utilizada e a sua percepção sobre o resultado alcançado.
    a. Quais os *code smells* mais comuns encontrados?
    b. Foi difícil a refatoração?
    c. Considera que houve melhoraria no código, explique.

Por fim, responda o questionário no endereço: Suprimido pelo blind review.

Anexo II - Trabalho – Parte 2

Considerando o projeto original e o projeto refatorado, faça um relatório, sobre o que mudou no projeto, e se considera que o projeto refatorado teve melhoria que o projeto original, e em que aspectos. Além disso, inclua os seguintes pontos na discussão:

1) Número de classes do projeto original e o refatorado.
2) Número de linha de códigos do projeto original e refatorado.
3) Analise as seguintes métricas dos dois projetos, e discuta o que significa a medida e o impacto nos projetos o resultado da medida (Utilize o plugin Project Usus para capturar as métricas):
    a. Average componente dependency
    b. Class Size
    c. Cyclomatic complexity
    d. Lack of cohesion of classes
    e. Method length
    f. Number of non-static, non-final public fields
    g. Package size
    h. Package with cyclic dependencies
    i. Unreferenced classes

Por fim, responda o questionário no endereço: *Suprimido pelo blind review*.