MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DA PARAÍBA

PLANO DE DISCIPLINA			
IDENTIFICAÇÃO			
CAMPUS: Cajazeiras			
CURSO: Superior de Tecnologia em Análise e Desenvolvimento de Sistemas			
DISCIPLINA: Fundamentos de Engenharia de Software		CÓDIGO DA DISCIPLINA: TEC.2417	
PRÉ-REQUISITO: -			
UNIDADE CURRICULAR: Obrigatória [X] Optativa [ ] Eletiva [ ]		SEMESTRE/ANO: 2025.1	
CARGA HORÁRIA			
TEÓRICA: 83h	PRÁTICA: 0	EaD[1]: 0	EXTENSÃO: 0
CARGA HORÁRIA SEMANAL: 5 h/a			
CARGA HORÁRIA TOTAL: 83h - 100h/a			
DOCENTE RESPONSÁVEL: Janderson Ferreira Dutra			

EMENTA

Fundamentos da Engenharia de Software: atividades fundamentais, fases e etapas, papéis e responsabilidades, artefatos e produtos. Fluxo de Produção de Software. Fábrica de Software. Qualidade de Software. Estimativas e Métricas. Requisitos. Modelagem de software.

OBJETIVOS

GERAL:

- Compreender os fundamentos da Engenharia de Software, abrangendo suas atividades, fases, papéis, artefatos e práticas essenciais para o desenvolvimento de software com qualidade e eficiência.

ESPECÍFICOS

- Identificar e descrever as principais atividades, fases e etapas do desenvolvimento de software.
- Compreender os papéis e responsabilidades dos profissionais envolvidos no processo de engenharia de software.
- Analisar os principais artefatos e produtos gerados ao longo do ciclo de vida do software.
- Explicar o fluxo de produção de software e os princípios de uma fábrica de software.
- Aplicar conceitos de qualidade de software, incluindo estimativas e métricas para avaliação de projetos.
- Estudar e aplicar técnicas de levantamento, análise e especificação de requisitos de software.
- Desenvolver habilidades na modelagem de software, utilizando notações e ferramentas adequadas.

CONTEÚDO PROGRAMÁTICO

1. Introdução à Engenharia de Software

- Conceitos fundamentais e evolução histórica
- Importância da Engenharia de Software na indústria
- Diferença entre desenvolvimento ad hoc e processos estruturados
- Organização do Trabalho: taylorismo; Fordismo; Toyotismo

2. Processo de Desenvolvimento de Software

- Fases e etapas do desenvolvimento de software

- Modelos de processo de software: cascata, evolutivo, iterativo, incremental, espiral, ágil e híbridos
- Introdução ao Rational Unified Process (RUP) e Scrum
- Papéis e responsabilidades no processo de desenvolvimento

3. Artefatos e Produtos do Desenvolvimento de Software

- Documentação técnica: visão geral e importância
- Principais artefatos: requisitos, modelos, código-fonte, testes, manuais
- Boas práticas para a criação e manutenção de artefatos

4. Fluxo de Produção e Fábrica de Software

- Organização e estrutura de uma fábrica de software
- Automação e integração no ciclo de desenvolvimento
- Ferramentas de suporte ao desenvolvimento e gestão
- Engenharia de Reuso de Software
- Linha de Produtos de Software
- Fluxos de Produção Puxada e Empurrada.

5. Qualidade de Software, Estimativas e Métricas

- Conceitos e princípios da qualidade de software
- Métricas de software: produtividade, complexidade, qualidade e manutenção
- Técnicas de estimativa de esforço e custo: PERT, COCOMO, Planning Poker
- Introdução a testes de software e estratégias de garantia de qualidade
- Principais métricas de qualidade
- Noções sobre KPI (Key Performance Indicator)
- Pontos de Função
- Cálculo de Custo de um Software
- Definição de Preço de Produtos de Software

6. Engenharia de Requisitos

- Tipos de requisitos: funcionais e não funcionais
- Técnicas de levantamento e análise de requisitos
- Especificação e validação de requisitos
- Requisitos em abordagens ágeis e tradicionais

7. Modelagem de Software

- Princípios de modelagem e abstração
- Introdução à UML (Unified Modeling Language)
- Diagramas UML: casos de uso, classes, sequência e atividades
- Modelagem ágil e boas práticas de design

8. Práticas Modernas de Engenharia de Software

- Desenvolvimento ágil e metodologias ágeis (Scrum, XP)
- Integração contínua, entrega contínua e DevOps
- Técnicas de refatoração e clean code
- Padrões de projeto e reutilização de software

METODOLOGIA DE ENSINO

A apresentação do conteúdo dar-se-á mediante aulas teóricas e práticas, apoiadas em recursos audiovisuais e computacionais, estabelecendo um ensino-aprendizagem significativo. As atividades incluirão exposição dialogada, estudos de caso, resolução de problemas, desenvolvimento de projetos e aplicação de metodologias ativas para engajamento dos alunos.

RECURSOS DIDÁTICOS

[X] Quadro
[X] Projetor
[ ] Vídeos/DVDs
[X] Periódicos/Livros/Revistas/Links
[ ] Equipamento de Som
[X] Laboratório
[X] Softwares[2]: Astah, Lucidchart, Figma, Planilha e Documentos eletrônicos.
[ ] Outros

CRITÉRIOS DE AVALIAÇÃO

- Serão contabilizadas 3 avaliações (AV1, AV2 e AV3). Durante o semestre o discente realizará várias atividades que valerá cada uma no máximo 100 pontos, será feita uma média aritmética com as notas dessas atividades, compondo assim a primeira avaliação. A segunda e terceira avaliações corresponderão a um projeto prático desenvolvido em duas etapas.
- A Média Semestral (MS) será a média aritmética obtida através das notas correspondentes às avaliações AV1, AV2 e AV3. Obterão a aprovação por média os alunos que atingirem a MS igual ou superior a 70 (setenta pontos). Será reprovado o discente que atingir a MS inferior a 40 (quarenta pontos).
- Os discentes que atingirem média inferior a 70 pontos e maior ou igual que 40 pontos poderão realizar uma Avaliação Final (AF). Esta avaliação valerá 100 pontos. A Média Final do Semestre (MF) será a média ponderada obtida pela fórmula:

$$MF = \frac{6 * MS + 4 * AF}{10}$$

Estará aprovado o discente que obtiver a Média Final maior ou igual a 50 pontos.

ATIVIDADE DE EXTENSÃO[4]

Não se aplica

BIBLIOGRAFIA[5]

Bibliografia Básica:

- PRESSMAN, R. S., MAXIN, B. R. **Engenharia de Software**: uma abordagem profissional. 9 ed. Porto Alegre: AMGH, 2016.
- SOMMERVILLE, I. **Engenharia de software**. 8 ed. São Paulo: Pearson, 2012.
- VALENTE, M. T. **Engenharia de software moderna**: Princípios e práticas para desenvolvimento de software com produtividade, 2020.

Bibliografia Complementar:

- BECK, K. **Programação extrema (XP) explicada**: acolha as mudanças. Porto Alegre: Bookman, 2004.
- KRUCHTEN, P. **Introdução ao RUP**: Rational Unified Process. Rio de Janeiro: Ciência Moderna, 2004.
- LARMAN, K. **Utilizando UML e padrões**: uma introdução à análise e ao projeto orientados a objetos. Porto Alegre: Bookman, 2007.
- SABBAGH, R. **Scrum**: Gestão ágil para projetos de sucesso. 2 ed. São Paulo: Casa do código, 2022.
- SCOTT, K. **O processo unificado explicado**. Porto Alegre: Bookman, 2003.

OBSERVAÇÕES

1 Para a oferta de disciplinas na modalidade à distância, integral ou parcial, desde que não ultrapassem os limites definidos em legislação.

2 Nesse ítem o professor deve especificar quais softwares serão trabalhados em sala de aula.

3 Nesse ítem o professor pode especificar outras formas de recursos utilizadas que não estejam citada.

4 Nesse item deve ser detalhado o PROJETO e/ou PROGRAMA DE EXTENSÃO que será executado na disciplina. Observando as orientações do Art. 10, Incisos I, II, III, IV, V, VI, VII e VIII, da Instrução Normativa que trata da construção do **Plano de Disciplina**.

5 Observar os mínimos de 3 (três) títulos para a bibliografia básica e 5 (cinco) para a bibliografia complementar.