DIRETORIA ACADÊMICA

PLANO DE ENSINO

Curso: Análise e Desenvolvimento de Sistemas		
Unidade Curricular: Engenharia de Software		**Carga Horária:** 80h
Modalidade: Presencial (X) Semipresencial ( ) EAD ( )		
Período: 4º	**Ano/ Semestre Letivo:** 2025.2	
Professor (es): Walter Travassos Sarinho		

1. EMENTA

Princípios da Engenharia de Software. Modelos de ciclo de vida. Processos de software. Requisitos de software. Padrões de projetos de software. Projeto da interface com o usuário. Planejamento e Estimativas. Documentação. Qualidade e métricas. Gerenciamento de mudanças e configurações. Manutenção e evolução de software. Engenharia de Software para sistemas Web. Tópicos atuais de Engenharia de Software. A Pesquisa e o futuro da Engenharia de Software.

2. OBJETIVO (S)

2.1. Geral:

- Capacitar os estudantes a compreenderem e aplicarem os princípios, processos, técnicas e ferramentas da Engenharia de Software, possibilitando o desenvolvimento, manutenção e evolução de sistemas de software com qualidade, eficiência e aderência aos requisitos dos stakeholders.

2.2. Específicos:

- Compreender os fundamentos e modelos da Engenharia de Software.
- Desenvolver e gerenciar processos e projetos de software.
- Elicitar, analisar, documentar e manter requisitos de software.
- Aplicar padrões e técnicas de design de software.
- Garantir a qualidade e utilizar métricas de software.
- Desenvolver sistemas para a web e explorar inovações na Engenharia de Software.

3. CONTEÚDO PROGRAMÁTICO										
U	**Conteúdos**	**Conhecimentos**	**Habilidades**	**Atitudes**	**Nº Horas /Aulas**				
					T	P	L	SP	EAD
I	**1. Introdução à Engenharia de Software** - Princípios da Engenharia de Software - Importância e Histórico da Engenharia de Software **2. Modelos de Ciclo de Vida do Software** - Modelos Clássicos: Cascata, Incremental, Espiral - Modelos Ágeis e DevOps **3. Processos de Software** - Processos Tradicionais e Ágeis - Metodologias de Desenvolvimento: Scrum, Kanban, XP **4. Requisitos de Software** - Tipos de Requisitos: Funcionais e Não Funcionais - Técnicas de Elicitação e Análise de Requisitos **5. Padrões de Projetos de Software** - Design Patterns: Criação, Estrutural e Comportamental - Exemplos e Aplicações de Padrões de Projeto	- Conceitos básicos e históricos da Engenharia de Software. - Importância e papel da Engenharia de Software no desenvolvimento de sistemas complexos. - Diferentes modelos de ciclo de vida: Cascata, Incremental, Espiral, Ágil, DevOps. - Processos tradicionais e ágeis de desenvolvimento de software. - Metodologias de desenvolvimento: Scrum, Kanban, Extreme Programming (XP). - Tipos de requisitos: funcionais e não funcionais. - Técnicas de elicitação e análise de requisitos. - Design patterns: Criação, Estrutural e Comportamental. - Exemplos práticos de padrões de projeto.	- Identificar os fundamentos teóricos da Engenharia de Software. - Selecionar e aplicar o modelo de ciclo de vida mais adequado para diferentes tipos de projetos. - Comparar vantagens e desvantagens dos modelos de ciclo de vida. - Implementar processos de software em projetos reais. - Gerenciar projetos utilizando metodologias ágeis. - Coletar e documentar requisitos de software. - Analisar e priorizar requisitos de forma eficaz. - Aplicar padrões de projeto na resolução de problemas de design. - Adaptar padrões de projeto a diferentes contextos de software.	- Valorizar a importância da Engenharia de Software. - Demonstrar curiosidade e interesse pela evolução da disciplina. - Flexibilidade para adaptar-se a diferentes metodologias de desenvolvimento. - Compromisso com a escolha do modelo mais eficiente para o projeto. - Proatividade na adoção de boas práticas de desenvolvimento. - Colaboração em equipes de desenvolvimento ágil. - Atenção aos detalhes ao documentar requisitos. - Empatia para entender as necessidades dos usuários e stakeholders. - Criatividade na aplicação de padrões de projeto. - Compromisso com a qualidade e reutilização de soluções de software.	12	12			
	6. Projeto da Interface com o Usuário	- Princípios de design de interface.	- Projetar interfaces de usuário intuitivas e eficazes.	- Foco na usabilidade e satisfação do usuário.	12	12			

II	- Princípios de Design de Interface - Usabilidade e Experiência do Usuário (UX) **7. Planejamento e Estimativas de Projetos** - Técnicas de Estimativa: PERT, COCOMO, Planning Poker - Planejamento de Recursos e Cronogramas **8. Documentação de Software** - Tipos de Documentação: Técnica, de Usuário, de Sistema - Boas Práticas de Documentação **9. Qualidade e Métricas de Software** - Conceitos de Qualidade de Software - Métricas e Indicadores de Qualidade.	- Conceitos de usabilidade e experiência do usuário (UX). - Técnicas de estimativa: PERT, COCOMO, Planning Poker. - Planejamento de recursos e cronogramas. - Tipos de documentação: técnica, de usuário, de sistema. - Boas práticas de documentação. - Conceitos de qualidade de software. - Métricas e indicadores de qualidade.	- Avaliar e melhorar a experiência do usuário com base em feedback. - Desenvolver estimativas precisas de esforço, custo e tempo. - Criar e gerenciar planos de projeto detalhados. - Produzir documentação clara e completa. - Manter a documentação atualizada ao longo do ciclo de vida do software. - Medir e avaliar a qualidade do software. - Implementar melhorias com base nas métricas de qualidade.	- Abertura para melhorias contínuas no design de interface. - Disciplina na elaboração e seguimento de planos de projeto. - Responsabilidade no cumprimento de prazos e orçamentos. - Organização na criação e manutenção de documentação. - Valorização da clareza e precisão na comunicação técnica. - Compromisso com a excelência e a melhoria contínua. - Rigor na aplicação de padrões de qualidade.					
III	**10. Gerenciamento de Mudanças e Configurações** - Controle de Versões e Configuração de Software - Processos de Gerenciamento de Mudanças **11. Manutenção e Evolução de Software** - Tipos de Manutenção: Corretiva, Adaptativa, Evolutiva - Estratégias para Evolução de Software	- Controle de versões e configuração de software. - Processos de gerenciamento de mudanças. - Tipos de manutenção: corretiva, adaptativa, evolutiva. - Estratégias para evolução de software. - Especificidades do desenvolvimento web. - Arquiteturas e tecnologias para sistemas web	- Gerenciar mudanças de forma eficaz e controlada. - Realizar manutenção e atualização de sistemas de software. - Planejar a evolução contínua dos sistemas. - Desenvolver e implementar sistemas web robustos e escaláveis. - Utilizar tecnologias adequadas para o desenvolvimento web.	- Adaptabilidade a mudanças e novas demandas. - Responsabilidade no gerenciamento de alterações. - Persistência na resolução de problemas de manutenção. - Visão de longo prazo para a evolução dos sistemas. - Interesse por novas tecnologias e tendências web. - Compromisso com a segurança e desempenho de sistemas web.	12	12			

	12. Engenharia de Software para Sistemas Web - Especificidades do Desenvolvimento Web - Arquiteturas e Tecnologias para Sistemas Web								
IV	**13. Tópicos Atuais e Futuro da Engenharia de Software** - Pesquisa e Inovações na Engenharia de Software - Futuro da Engenharia de Software: Inteligência Artificial, Blockchain e outras tendências.	- Inovações e pesquisas recentes na Engenharia de Software. - Impacto de novas tecnologias como IA e Blockchain.	- Analisar e aplicar novas tendências e tecnologias na prática. - Realizar pesquisas e investigações na área de Engenharia de Software.	- Curiosidade e abertura para aprender sobre inovações. - Visão estratégica sobre o futuro da Engenharia de Software.	10	10			
Subtotal da Carga Horária					**40**	**40**			
Total da Carga Horária						**80**			

Legenda: U - unidade **T –** quantidade de aula(s) teórica(s) por unidade(s); **P-** quantidade de aula(s) práticas(s) por unidade(s); **L - quantidade de aula(s) por unidade(s) de Prática Pedagógica (exclusivo para o Curso de Educação Física - Licenciatura**); **SP** - quantidade de aulas(s) semipresenciais; **EAD** - quantidade de aula(s) à distância.

4. METODOLOGIA

O componente curricular será ministrado de forma dinâmica, agregando estratégias de metodologias ativas: aulas expositivas e dialogadas; exibição de vídeos; discussão de casos; experiências e situações do mercado de trabalho, através da metodologia de situações-problema; projetos de software; atividades práticas; atividade integrada (Prática Integradora), seguindo o modelo proposto para o período letivo do curso.

5. RECURSOS DIDÁTICOS

Projetor multimídia, vídeos, documentários, quadro branco e pincel que subsidiarão as aulas expositivo-dialogadas, os debates, as discussões e a atividade integrada. Para as sessões de filmes e vídeos, projetor de mídia e notebook. Para as atividades de estudo e pesquisa, livros didáticos disponíveis na biblioteca ou disponibilizados pelo docente, bem como artigos científicos disponíveis nas Bases de Dados (EBSCO e Portal CAPES). Ainda, será utilizado o Ambiente Virtual de Aprendizagem (AVA) do SECTRAS *On Line*.

6. AVALIAÇÃO (ÕES)

A avaliação do discente será contínua e processual. Se dará mediante o somatório de diferentes atividades, que constituirão uma nota final para cada uma das três principais verificações de aprendizagem (uma por UNIDADE) previstas pela Instituição. A média final será obtida a partir dos resultados de cada Unidade. Estão previstas 3 avaliações (formativas e somativas), dispostas da seguinte forma:

- 1ª Unidade: avaliação escrita (8,00) e atividade prática (2,00).

- 2ª Unidade: avaliação escrita (8,00) e atividade prática (2,00).
- 3ª Unidade: avaliação escrita (6,00), atividade prática (2,00) e Prática Integradora (2,00).
Além disso, haverá 6 avaliações diagnósticas para obter juízo de valor relativo ao rendimento do aluno sobre os temas e nas atividades práticas.

7. BIBLIOGRAFIA

7.1. Básica:

MASSARI, V.L. **Gerenciamento Ágil de Projetos.** 2ª Ed. Rio de Janeiro: Brasport, 2018. [acervo digital]
MORAIS, Izabelly Soares de (org.). **Engenharia de software.** São Paulo: Pearson Education do Brasil, 2017. [acervo digital]
SOMMERVILLE, Ian. **Engenharia de Software.** 10ª Ed. São Paulo: Pearson Education, 2019. [acervo digital]

7.2. Complementar:

BRAGA, P.H.C. **Teste de Software.** São Paulo: Pearson, 2016. [acervo digital]
GALLOTTI, G.M.A. **Arquitetura de Software.** 1ª Ed. São Paulo: Pearson Education do Brasil, 2016.
GALLOTTI, G.M.A. **Qualidade de Software.** São Paulo: Pearson, 2016. [acervo digital]
KERR, E. S. **Gerenciamento de requisitos.** 1ª Ed. São Paulo: Pearson Education do Brasil, 2015. [acervo digital]
VASQUES, C.E.; SIMÕES, G.S. **Engenharia de requisitos: software orientado ao negócio.** 1ª Ed. Rio de Janeiro: Brasport, 2016. [acervo digital]