|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Professor:** | | **Componente Curricular:** | | **Ano/Turma:** | **Bimestre:** |
| **João Vitor Yokada** | | **Processos de Desenvolvimento de Software** | | **2ºDS** | **3º** |
| **Justificativa** | | | | | |
| Este Guia de Aprendizagemvisa desenvolver as competências e habilidades do Currículo Paulista e os princípios do Programa Ensino Integral: Pedagogia da Presença, Protagonismo, os Quatro Pilares da Educação e Educação Interdimensional. | | | | | |
| **Aproximação com a realidade do estudante** | | | | | |
| Aproximar os conteúdos propostos com o contexto vivido pelos estudantes de forma intencional, exemplificando situações, profissões e transformações onde os sujeitos possam assumir uma postura ativa, crítica e criativa frente ao mundo do qual fazem parte. | | | | | |
| **Título** | **Conteúdos** | | **Objetivos** | | |
| **Processos de Desenvolvimento de Software** | Com base no conteúdo apresentado, os objetivos da disciplina visam capacitar os alunos da aplicação prática dos princípios da orientação a objetos e no domínio de boas práticas de desenvolvimento de software. Os conteúdos abordados permitem ao aluno:   * Compreender e aplicar técnicas de validação e verificação de requisitos, essenciais para garantir que o sistema atenda às expectativas do cliente e manutenção do código. * Empregar o conceito de encapsulamento na construção de projetos práticos, promovendo organização, segurança e manutenção do código. * Desenvolver e executar testes não funcionais, avaliando aspectos como desempenho, segurança e usabilidade do software. * Construir e interpretar diagramas de estado e de atividades utilizando modelagem visual com UML, facilitando a comunicação de processos internos e fluxos do sistema. * Compreender o conceito de herança, promovendo reutilização de código e organização hierárquica de classes, bem como implementá-la em projetos práticos. * Aprender técnicas de gerenciamento de mudanças de requisitos, que auxiliam na adaptação contínua do software às necessidades do cliente. * Aplicar de forma prática o polimorfismo, consolidando os pilares da orientação a objetos para construção de sistemas flexíveis e extensíveis.   Ao final do bimestre, o estudante deverá compreender os processos e habilidades apresentados. | | Processos de Desenvolvimento de Software e Metodologias Ágeis tem como objetivo capacitar os alunos a aplicarem conceitos fundamentais da engenharia de software com foco prático em orientação a objetos e boas práticas de desenvolvimento. Entre os temas abordados estão a validação e verificação de requisitos, o uso do encapsulamento para organização e segurança do código, o desenvolvimento e execução de testes não funcionais, e a modelagem de comportamentos com diagramas de estado e de atividades em UML. Os alunos também aprendem a reutilizar código por meio da herança e da estruturação hierárquica de classes, a gerenciar mudanças em requisitos com eficácia, e a aplicar, na prática, os conceitos de herança e polimorfismo para construção de sistemas flexíveis, reutilizáveis e de alta qualidade. | | |
| **Metodologias** | | | **Ambientes de Aprendizagem** | | |
| Ambiente virtual de aprendizagem (AVA)  Utilização dos materiais digitais, impressos e tecnológicos  Leitura de materiais, livros e notícias  Metodologias ativas, gamificação  Visitas técnicas, projetos  Debates, palestras | | | Sala de aula  Sala de leitura  Sala Maker | | |
| **Critérios de Avaliação** | | | | | |
| Prova Paulista  Atividades – AVA  Avaliação bimestral - MINITCC | | | | | |
| **Fontes de pesquisa para o estudante** | | | | | |
| **Bibliografia básica**   * PRESSMAN, Roger S. *Engenharia de software: uma abordagem profissional*. 7. ed. Porto Alegre: AMGH, 2016. * SCHWABER, Ken; SUTHERLAND, Jeff. *Guia do Scrum – um guia definitivo para o Scrum: as regras do jogo*. [atualizado em 2020]. Disponível em: <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Portuguese-European.pdf>. Acesso em: 15 nov. 2023. * SOMMERVILLE, Ian. *Engenharia de Software*. 9. ed. São Paulo: Pearson Addison Wesley, 2011.   **Bibliografia complementar**   * BECK, Kent. *Extreme Programming Explained: Embrace Change*. 2. ed. Boston: Addison-Wesley, 2004. * COHN, Mike. *User Stories Applied: For Agile Software Development*. Boston: Addison-Wesley Professional, 2004. * FOWLER, Martin. *Agile Software Development: Principles, Patterns, and Practices*. Boston: Addison-Wesley Professional, 2002. * FOWLER, Martin. *Refactoring: Improving the Design of Existing Code*. Boston: Addison-Wesley Professional, 2018. * RUMBAUGH, James; JACOBSON, Ivar; BOOCH, Grady. *Unified Modeling Language Reference Manual*. 2. ed. Boston: Addison-Wesley, 2004. | | | | | |

 **GUIA DE APRENDIZAGEM**



**Ensino Médio**