REVISTA OWL (*OWL Journal*)
www.revistaowl.com.br – ISSN: 2965-2634

CLEAN CODE UNIVERSE: UMA PLATAFORMA ONLINE PARA O ENSINO DE CLEAN CODE

CLEAN CODE UNIVERSE: AN ONLINE PLATFORM FOR TEACHING CLEAN CODE

DOI: 10.5281/zenodo.22803742

Joel Victor Brandão dos Santos[1]
Anderson Jorge Serra da Costa[2]

RESUMO

A inclusão de estudantes com Transtorno do Espectro Autista (TEA) em escolas rurais representa um desafio pedagógico singular, agravado pela escassez de recursos tecnológicos e pela insuficiência deformação docente específica. Este artigo analisa as percepções de professores da educação básica acerca da utilização de recursos de baixa tecnologia como estratégia inclusiva no processo de ensino de estudantes com TEA em uma escola pública rural do município de Macapá/AP. A pesquisa fundamenta-se em abordagem qualitativa, de natureza aplicada e caráter descritivo-exploratório, com coleta de dados por meio de entrevistas semiestruturadas com seis docentes que atuam diretamente com estudantes diagnosticados com TEA. A análise dos dados orienta-se pela técnica de análise de conteúdo. Os resultados evidenciam que recursos como pictogramas, pranchas de Comunicação Aumentativa e Alternativa (CAA), rotinas visuais e materiais concretos adaptados constituem estratégias pedagógicas viáveis e potencialmente inclusivas em contextos marcados pela precariedade tecnológica. Conclui-se que a formação continuada é condição essencial para ampliar o repertório pedagógico dos docentes e fortalecer práticas inclusivas contextualizadas à realidade rural amazônica, superando a dependência exclusiva de tecnologias digitais de alto custo.

Palavras-chave: Clean Code; Qualidade de Software; Ensino de Clean Code.

1 Graduando em Engenharia de Software pela Universidade do Estado do Pará (UEPA), Belém, Pará, Brasil. E-mail: joelvictorbrandao@gmail.com

2 Doutor em Ciência da Computação pela Universidade Federal do Pará (UFPA), Belém, Pará, Brasil. E-mail: andersonjsc@uepa.br

ABSTRACT

This study aims to develop and apply the "Clean Code Universe" online platform, designed to teach Clean Code principles to software development students. The initiative arose from the difficulties students face in applying technical skills demanded by the job market—specifically, producing high-quality code through programming best practices. To achieve this, the research adopted the Design Science Research Methodology (DSRM) approach to develop the proposed platform. The study was conducted with a group of students from the Bachelor of Software Engineering (BES) program at the State University of Pará (UEPA). Three questionnaires were administered to the student group: the first supported requirements gathering, while the subsequent two provided feedback on the use of the pilot version and the final version of the platform, respectively. Evaluation of the final version yielded positive results regarding the platform's usability and the content of the lessons within each module. The study concludes that Clean Code Universe shows potential as a resource to support teaching, learning, and quality code development practices.

Keywords: Clean Code; Software Quality; Teaching Clean Code.

1. INTRODUÇÃO

No cenário atual da tecnologia, a crescente demanda por entregas de software com rapidez e qualidade, cria um alerta para lidar com as exigências da indústria por software de qualidade (Fernandes, 2025). Neste caso, a preocupação em buscar tratar o software com padrões e regras que visam manter sua estabilidade vem de longa data, uma vez que, sem qualquer padrão, ele pode rapidamente se tornar obsoleto e difícil de manter (Stryuk, 2018)

Nesse cenário, cresce a procura por profissionais de tecnologia qualificados, com conhecimentos teóricos e habilidade prática em projetos de desenvolvimento de software, utilizando tecnologias e ferramentas exigidas pelo mercado de trabalho (Santiago et al., 2023). Um desses conhecimentos é a escrita de código com qualidade, facilitando a escrita e leitura do código, e, consequentemente, facilitando o ciclo de vida do software. Visto que, qualquer ferramenta em uso, deve ser constantemente mantida e corrigida, integrando novas funcionalidades e melhorias ao sistema (Portos; Mendes, 2022).

No entanto, apesar da demanda da indústria, existem dificuldades vivenciadas por muitos educadores em relação aos métodos de ensino na preparação de seus estudantes para atuar no mercado de trabalho (Santos et al., 2024). Muitas pesquisas apontam essa realidade, na qual estudantes não se sentem preparados para ingressar no mercado de trabalho, havendo

uma lacuna entre o que é ensinado nas faculdades/universidades e o que a indústria exige (Gomes, 2021; Rolim et al., 2021).

Nesse sentido, a adoção de metodologias de ensino mais interativas para o aprendizado dos estudantes favorece o desenvolvimento de habilidades que alinham a formação acadêmica com as demandas do setor de tecnologia (Mohammed; Ozdamli, 2024). Como a indústria busca rapidez e qualidade na entrega de software, ensinar conceitos que possibilitam atender às exigências do mercado, como Clean Code (código limpo), são alguns dos conteúdos que podem ser trabalhados seguindo métodos de ensino mais interativos (Gomes, 2021).

Portanto, a utilização de alternativas para apoiar o ensino, por meio da adoção de conceitos importantes para a indústria e pela utilização de ferramentas interativas, é fundamental para promover uma aprendizagem alinhada às exigências do mercado (Gomes, 2021). Diante desse cenário, este trabalho considera como problema de pesquisa, a dificuldade na formação de profissionais com conhecimento em boas práticas de escrita de código, o que pode dificultar sua inserção no mercado de trabalho. Nesse contexto, esta pesquisa busca compreender de que forma o ensino de Clean Code pode contribuir para o desenvolvimento de competências técnicas de desenvolvimento de código de qualidade valorizadas pela indústria?

Desta forma, esse projeto tem como objetivo desenvolver e aplicar uma plataforma online para o ensino de boas práticas de escrita de código, com base nos princípios do Clean Code. A iniciativa permitiu desenvolver uma alternativa na busca por auxiliar no aprendizado dos estudantes da área de desenvolvimento de software. A plataforma, chamada de Clean Code Universe, representa uma proposta de uma plataforma online para auxiliar no ensino de Clean Code, aplicado com 17 estudantes do curso de Bacharelado em Engenharia de Software (BES) da Universidade do Estado do Pará (UEPA). A pesquisa foi desenvolvida seguindo a abordagem do Design Science Research Methodology (DSRM) (Peffers et al., 2007).

2. REFERENCIAL TEÓRICO

2.1. Qualidade de Código

Desde o seu surgimento, o software tem gerado grandes impactos, que vêm impulsionando a transformação na indústria (Joshua Andeme, 2025). O desenvolvimento de aplicações como Internet das Coisas (IoT), automação, segurança cibernética e Inteligência Artificial (IA) são alguns exemplos do impacto da Engenharia de Software, permitindo inovações significativas e a criação de novos modelos de negócios (Karabegovic; Isak, 2023).

No entanto, à medida que o software passou a estar cada vez mais presente no cotidiano das pessoas, aumentou também a necessidade de desenvolver código de qualidade, fundamentado em técnicas, padrões e boas práticas de desenvolvimento (Fernandes, 2025). Nesse contexto, a qualidade do código também deve estar relacionada à facilidade de ser compreendido e mantido por outros desenvolvedores, aspectos que contribuem para a evolução e a sustentabilidade dos sistemas de software (Gomes, 2021).

Dessa maneira, o software deve ser desenvolvido com profissionalismo, cabendo ao desenvolvedor assumir valores, princípios e responsabilidades relacionados ao exercício da sua profissão no mercado (Semerikov et al., 2020). Para que o desenvolvimento ocorra de forma eficiente e profissional, é fundamental seguir princípios que facilitem a compreensão do código por todos os profissionais envolvidos em sua manutenção e evolução (Portos; Mendes, 2022).

Nesse contexto, os padrões de desenvolvimento de software são essenciais, tais como o Princípio da Responsabilidade Única (SRP), o Princípio da Segregação de Interfaces (ISP) e o Princípio da Substituição de Liskov (LSP). Esses padrões são exemplos de práticas que ajudam a melhorar a escrita do código (Tyagi; Beg, 2022) e aumenta, consequentemente, a manutenibilidade do software. Dessa forma, conceitos como o de refatoração de código e Clean Code incluem diversas técnicas para a melhoria da escrita de código.

2.2. Clean Code

Uma das alternativas mais conhecidas de padrões de qualidade de código consiste no conjunto de conceitos chamado de Clean Code, também conhecido como Código Limpo, no português. Neste caso, trata-se de um estilo de desenvolvimento de software com foco na leitura e produção de código fácil de ler e escrever (Gomes, 2021).

O Clean Code é uma filosofia de desenvolvimento cujo objetivo é aplicar técnicas que facilitam a escrita, a leitura e a manutenção do código. Ele resulta da consolidação de padrões e do uso de boas práticas de programação, contribuindo para o desenvolvimento de software mais organizado e de melhor qualidade (Portos; Mendes, 2022).

O Clean Code se baseia no conhecimento de diversos profissionais experientes da área de desenvolvimento (Martin, 2009). Esses conhecimentos foram organizados por Robert C. Martin em seu livro Clean Code: A Handbook of Agile Software Craftsmanship publicado em 2009 (Martin, 2009).

No livro, são abordados aspectos que vão desde a escolha de nomes significativos para variáveis, funções e métodos até a organização geral do código. Também são apresentadas práticas como a implementação de testes no código, a eliminação de duplicações e a adoção de uma estrutura simples, evitando complexidade desnecessária (Martin, 2009). Essas práticas auxiliam na melhoria da estrutura interna de um código já existente, sem alterar seu comportamento ou adicionar novas funcionalidades, tornando-o mais organizado e fácil de manter (Portos; Mendes, 2022). O Clean Code pode ser utilizado tanto como padrão de desenvolvimento, desde o início da escrita do código, quanto parte de um processo de refatoração de código.

2.3. Ensino de Clean Code

O ensino de programação é uma prática adotada há décadas, com uma grande diversidade de instituições de ensino, que fazem uso de diversas metodologias. No entanto, de acordo com Gomes et al. (2017), muitas universidades enfrentam dificuldades no ensino

voltado para a qualidade de código e muitos estudantes acabam tendo dificuldades em aplicar os conceitos, por exemplo, de Clean Code em seus projetos. Isso acontece devido à complexidade no aprendizado de programação, em que alguns alunos apresentam dificuldades com fundamentos básicos, como algoritmos, linguagens de programação e paradigmas e acabam deixando de lado conceitos como a qualidade do código (Gomes et al., 2017).

Além disso, a falta de conhecimento prévio em programação tem sido um obstáculo para estudantes de cursos de TI. Em uma pesquisa realizada por Rolim et al., (2021), focada no curso de Engenharia de Software, mais de 75% dos estudantes do primeiro semestre relataram nunca ter tido contato com nenhuma linguagem de programação antes de ingressarem no curso de graduação. Essa lacuna inicial dificulta a adaptação dos alunos às disciplinas práticas e, consequentemente, ao aprendizado de conceitos mais avançados (Rolim et al., 2021).

O desenvolvimento de ferramentas para o ensino de qualidade de código tem evidenciado a busca por novas abordagens de ensino nos cursos superiores de TI. Nesse sentido, a aplicação dos princípios de Clean Code através dessas ferramentas, reduz a má escrita de código e diminui a ocorrência de vulnerabilidades (Araújo et al., 2020; Gomes et al., 2017). Essas pesquisas são essenciais, pois demonstram como novas abordagens de ensino podem contribuir para o aprendizado de boas práticas de Clean Code. No entanto, esses estudos ainda evidenciam a necessidade de mais alternativas de estudos que apliquem, por exemplo, ferramentas voltadas para o ensino de Clean Code (Araújo et al., 2020).

3. TRABALHOS CORRELATOS

3.1. Critérios de Seleção

Para a identificação de trabalhos relacionados a plataforma de ensino e aprendizagem de Clean Code aqui proposta, realizou-se uma revisão bibliográfica da literatura, conforme descrito na seção 4.1 da metodologia, adotando critérios de inclusão e exclusão, string de busca e palavras-chave definidos previamente. A partir dessa revisão bibliográfica, um

conjunto de trabalhos serviu de base teórica, sendo citados nas definições de teorias e argumentações do texto, enquanto que outro conjunto de trabalhos foi selecionado como trabalhos correlatos. A análise dos trabalhos selecionados com correlatos foi realizada com base na metodologia proposta por Bardin (2016), estruturada em três etapas: pré-análise, exploração de matérias e tratamento dos resultados.

Na etapa de pré-análise, buscaram-se definir os objetivos da análise, realizar uma leitura inicial dos trabalhos e organizar os materiais a serem analisados. A partir dessa análise, os estudos foram organizados para submissão às etapas subsequentes de exploração do material e análise dos resultados. Ao final do processo de seleção, quatro trabalhos foram considerados adequados para compor os trabalhos correlatos à pesquisa aqui proposta.

A partir disso, cada estudo foi apresentado nas seções seguintes, destacando sua relação com o tema pesquisado, além de identificar possíveis lacunas encontradas.

3.2.1. Teacher Mate: A Support Tool for Teaching Code Quality

Araújo et al. (2020) abordam em seu estudo os obstáculos enfrentados pelos professores para acompanhar individualmente as dificuldades de seus alunos, em relação aos conceitos associados à qualidade de código e à identificação da ausência de boas práticas de programação. Segundo os autores, esses problemas podem comprometer o desenvolvimento de habilidades práticas dos estudantes e, consequentemente, afetar a qualidade de seus futuros projetos.

Durante o estudo, os autores apresentaram e avaliaram uma ferramenta denominada de Teacher Mate, desenvolvida com o objetivo de apoiar o processo de ensino de qualidade de código no ambiente acadêmico. Araújo et al. (2020) mencionaram que a ferramenta funciona de forma integrada à plataforma SonarQube (https://www.sonarsource.com/products/sonarqube/). Onde os alunos podem submeter seus projetos ao Teacher Mate, atuando como um repositório para o armazenamento e gerenciamento dos códigos. Para mais tarde realizar a análise dos projetos. Neste caso, a ferramenta dispara uma instância do Sonar Scanner, responsável por analisar individualmente cada projeto. Durante esse processo, o Sonar

Scanner verifica o código com base em um conjunto predefinido de regras, denominado perfil de qualidade.

Após o processamento dos códigos, o Teacher Mate consome a API do SonarQube para obter os relatórios referentes às regras violadas em cada projeto. Essas informações são então utilizadas para acompanhar o desempenho dos alunos e possibilitar o monitoramento visual, permitindo verificar se a ocorrência de problemas relacionados à qualidade do código aumenta ou diminui ao longo do tempo.

Em relação à abordagem pedagógica, a pesquisa foi baseada no diagnóstico contínuo por meio da inspeção estática automatizada de código para gerar percepções das dificuldades. A partir dessas informações obtidas, o professor recebe um diagnóstico sobre as principais dificuldades apresentadas pelos alunos, podendo adaptar às suas aulas e reforçar os conteúdos em que a turma apresenta mais dificuldade.

A pesquisa apresenta uma alternativa promissora para aproximar alunos e professores, especialmente diante da dificuldade de acompanhar a qualidade dos projetos de cada aluno. Porém, ainda falta uma integração direta com recursos que forneçam feedback imediato ao aluno, auxiliando na correção de erros durante o desenvolvimento dos projetos, o que poderia ser ampliado para melhores feedbacks para os estudantes, conseguindo melhorar o processo de ensino-aprendizagem.

3.2.2. Teaching Software Quality via Source Code Inspection Tool

Gomes et al. (2017) relataram problemas relacionados ao ensino de qualidade de software no contexto educacional. Embora os estudantes sejam incentivados e orientados a aprimorar a qualidade de seus códigos durante as disciplinas introdutórias do curso, o contato com conceitos relacionados à qualidade de software ocorre, de forma mais sistemática, apenas em semestres posteriores.

Para apoiar as dificuldades encontradas sobre a identificação da qualidade de código durante as etapas iniciais da formação, os autores apresentaram a ferramenta SMaRT (Software Maintenance, Report and Tracker), desenvolvida como um mecanismo de revisão automática de código executado localmente na própria máquina do estudante. A ferramenta é integrada ao SonarQube Scanner em sua arquitetura, permitindo a análise local do código, e utiliza dentro do ambiente de desenvolvimento baseado no Eclipse Plug-in Development Environment, integrando-se diretamente à interface de programação já utilizada pelos alunos.

A abordagem pedagógica utilizada consistiu na realização de exercícios propostos pelos professores, nos quais os alunos deveriam refatorar códigos que continham erros intencionalmente inseridos pelos docentes, utilizando indicadores visuais de desempenho, representados por notas de A a E, para demonstrar o nível de qualidade e desempenho dos códigos desenvolvidos pelos estudantes.

A proposta desenvolvida demonstra um estudo promissor, apresentando-se como uma alternativa para a melhoria da escrita de código pelos estudantes. Entretanto, ainda são necessárias avaliações mais abrangentes, na qual não foi avaliado os impactos de longo prazo da utilização do SMaRT na formação dos estudantes. Além disso, devido aos conceitos e conhecimentos prévios exigidos pela ferramenta, estudantes iniciantes podem apresentar dificuldades em sua utilização. O que leva a entender que outras propostas, que foquem na prática de conhecimentos básicos do Clean Code, são bem vindas para complementar essa ferramenta.

3.2.3. Clean Code Tutoring: Makings of a Foundation

Na pesquisa de Luburić et al. (2017), é apresentada a ferramenta Clean CaDET, um Tutor Inteligente projetado para ensinar boas práticas de código limpo e refatoração a engenheiros de software. Na pesquisa, os autores destacam o cenário de dificuldade de ensinar e avaliar a qualidade do código devido à ambiguidade inerente ao conceito de Clean Code.

O Clean CaDET funciona de forma integrada com recursos de Inteligência Artificial (IA), funcionando através de um plugin conectado a IDE do aluno, no qual o estudante pode carregar e resolver desafios práticos de refatoração. O núcleo inteligente da plataforma é composto por um Sistema Tutor Inteligente, que utiliza algoritmos e sistemas de recomendação baseados em regras para emular, de forma dinâmica, a tomada de decisão semelhante a um instrutor humano.

Em relação à abordagem pedagógica, a avaliação do Clean CaDET foi realizada em duas etapas, combinando métodos qualitativos e quantitativos. Inicialmente, a ferramenta foi avaliada e testada por engenheiros de software de nível intermediário, com o objetivo de verificar sua utilização e identificar aspectos relacionados à sua aplicação. Em seguida, foi conduzido um experimento com maior quantidade de estudantes do terceiro ano de Engenharia de Software, divididos em dois grupos: um grupo experimental, que utilizou o tutor inteligente, e um grupo de controle, que utilizou materiais educacionais disponibilizados por meio de um LMS (Learning Management System) tradicional.

A pesquisa apresentada propôs uma abordagem alternativa para o ensino de Clean Code por meio de um tutor inteligente baseado em Inteligência Artificial (IA), especialmente diante da dificuldade de acompanhar e avaliar a qualidade dos projetos desenvolvidos individualmente pelos estudantes. Entretanto, ainda são necessários mais testes para avaliar a real eficácia da ferramenta proposta em outros contextos.

3.2.4. Teaching Clean Code

Dietz et al. (2018), em sua pesquisa, destacam a lacuna existente entre o conhecimento abordado no contexto acadêmico e as exigências da indústria de software. Embora haja uma demanda significativa por desenvolvedores capazes de aplicar práticas relacionadas à qualidade do código, observa-se que essas habilidades ainda são pouco exploradas no processo de formação dos estudantes.

Dessa forma, os autores demonstram a necessidade de ampliar o ensino de práticas de qualidade de código no ambiente acadêmico. Como resposta a essa necessidade, foi

apresentada uma meta-ferramenta de análise estática de código. A meta-ferramenta foi configurada para executar uma seleção de 107 regras de qualidade de código e centralizar os problemas identificados em um único arquivo de saída no formato CSV.

Sobre a abordagem pedagógica adotada pelos autores, as aulas foram divididas em três fases. A primeira consistiu em aulas expositivas interativas, que alternavam blocos de conteúdo teórico e atividades práticas. A segunda foi através de atividades práticas em grupo, nas quais três estudantes trabalhavam na resolução de problemas de maior complexidade, e em seguida, recebendo correções e revisões detalhadas de seus códigos. Por fim, foi realizado um exame oral individual, no qual cada aluno era submetido a perguntas teóricas relacionadas ao projeto prático desenvolvido.

A ferramenta apresenta uma alternativa promissora ao buscar alinhar o ensino às exigências do mercado, em relação à formação de profissionais com habilidades em qualidade de código. Porém, ainda são necessárias avaliações mais abrangentes, pois os dados que foram coletados envolveram uma amostra pequena de participantes, durante um único semestre acadêmico. Além disso, as regras utilizadas pela ferramenta não são capazes de identificar todos os possíveis problemas relacionados à qualidade de um software, indicando a necessidade de complementar a meta-ferramenta através de outras abordagens.

4. METODOLOGIA

Este trabalho adotou a metodologia Design Science Research Methodology (DSRM), quanto a sua abordagem. A escolha por essa metodologia se deu por ela focar na criação e avaliação de soluções tecnológicas, orientando o processo desde a identificação do problema até a produção de conhecimento científico e teórico (Peffers et al., 2007), o que se alinha com o objetivo d e criar uma solução tecnológica para apoiar o ensino de Clean Code.

O modelo DSRM foi aplicado seguindo seis etapas fundamentais: a identificação do problema e sua motivação; a definição dos objetivos da solução; o desenvolvimento da plataforma; a demonstração; avaliação; e a comunicação dos resultados. Assim como em outros estudos que empregaram essas mesmas etapas do DSRM (Orisa et al., 2023; Sabri,

2025), esse modelo serviu de base no processo de desenvolvimento e avaliação deste trabalho. As seções a seguir apresentam mais detalhes sobre cada uma dessas seis etapas do DSRM.

4.1. Identificação do problema

A primeira etapa do DSRM consiste em definir o problema de pesquisa e explicar por que a solução proposta é importante, uma vez que essa definição serve de base para todo o processo do desenvolvimento da plataforma (Peffers et al., 2007).

Neste trabalho, a identificação do problema ocorreu a partir de duas abordagens, uma através da revisão bibliográfica e a outra através de um questionário aplicado aos estudantes. Na revisão bibliográfica, foram analisados estudos nas bases SciELO e Google Acadêmico.

Para a realização das buscas, foram utilizados termos de busca que abordassem os principais temas relacionados à pesquisa sobre Clean Code, como qualidade de código, ao ensino e aprendizagem de Clean Code. A partir desses termos, foi definida a seguinte string de busca:

clean code AND software AND (teaching OR learning) AND code quality.

O objetivo foi compreender as principais dificuldades e lacunas relacionadas ao ensino de qualidade de código (Gomes, 2021; Araújo et al., 2020). Em seguida, os trabalhos encontrados foram submetidos a uma análise inicial de seus títulos, palavras-chave e resumos, seguindo os critérios de inclusão e exclusão previamente definidos, a fim de verificar sua relevância e relação com o tema da pesquisa.

Em relação aos critérios de inclusão, foram adotados os seguintes critérios:

- trabalhos que envolvam a qualidade de código;
- trabalhos relacionados aos princípios de Clean Code;
- trabalhos voltados ao ensino ou à aprendizagem de Clean Code;
- trabalhos com foco em plataforma de ensino e aprendizagem de Clean Code;
- Estudos publicados entre 2016 e 2026.

Entre os critérios de exclusão, foram considerados:

- trabalhos sem relação direta com o tema da pesquisa;
- trabalhos publicados antes de 2016;
- trabalhos que não apresentassem informações suficientes ou relevantes para os objetivos da pesquisa;
- Trabalhos que não estejam escritos em Inglês ou Português;
- Trabalhos inacessíveis em suas versões completas.

Após a aplicação dos critérios estabelecidos, os trabalhos foram submetidos à etapa de exploração do material, na qual foram analisadas suas principais características, contribuições e relações com o ensino e a aprendizagem de Clean Code. Com base nos termos de busca, os critérios de seleção, a base de dados e as palavras chaves, foram selecionados diversos trabalhos com objetivo de analisar suas principais contribuições para o ensino e a aprendizagem de Clean Code.

O Quadro 1 apresenta a quantidade de artigos encontrados para cada área da pesquisa, bem como o número de trabalhos efetivamente selecionados.

Quadro 1: Áreas da pesquisa dos artigos encontrados e selecionados

Áreas encontradas	Artigos encontrados	Artigos selecionados
Qualidade de Código	1.000.000	10
Clean Code and Código Limpo	1.000	8
Aprendizagem and Clean Code	4.900	6
Ensino and Clean Code	5.000	6
Plataforma and ensino and Clean Code	1.500	4

Fonte: Google Acadêmico; SciELO; Elicit, 2026

Após a fase da revisão bibliográfica, foi aplicado um questionário inicial a 16 estudantes do curso de Bacharelado em Engenharia de Software (BES) da UEPA. O

questionário buscou identificar as dificuldades dos estudantes na aplicação dos conceitos de Clean Code, seu nível de conhecimento sobre boas práticas de programação e sua percepção sobre a abordagem desses conteúdos em sala de aula. Os resultados do questionário são detalhados na seção 5.3.

A análise conjunta dos dados coletados permitiu identificar o problema de pesquisa neste trabalho. A proposta concentrou-se na dificuldade da formação de profissionais com conhecimento em boas práticas de escrita de código, o que pode dificultar sua inserção no mercado de trabalho. Essa definição orientou todas as etapas subsequentes do desenvolvimento da plataforma Clean Code Universe.

4.2. Definição do Objetivo da Solução

Os dados levantados na etapa anterior foram discutidos e analisados com o orientador deste trabalho em uma reunião online, com duração de uma hora, contribuindo para a definição do objetivo da solução. Durante a reunião, foram analisados estudos e pesquisas científicas, que contribuíram para o direcionamento da pesquisa. A partir dessa discussão, foi possível definir o escopo e o objetivo específico da solução proposta. O objetivo da pesquisa está apresentado na seção 5.2.

4.3. Desenvolvimento da Solução

A primeira etapa do desenvolvimento da solução consistiu na coleta de dados por meio de um questionário aplicado com 16 estudantes do curso de Bacharelado em Engenharia de Software (BES) da Universidade do Estado do Pará (UEPA), tendo como finalidade levantar informações iniciais sobre o perfil de cada participante, seu nível de domínio em programação, as dificuldades enfrentadas, o conhecimento prévio sobre Clean Code, sua importância, se os conteúdos abordados em aula eram suficientes para aplicar Clean Code, bem como a percepção sobre a relevância de uma plataforma de apoio ao aprendizado. Também foram investigados outros aspectos como a linguagem de programação que poderia ser utilizada e quais funcionalidades a plataforma poderia incluir.

A partir das informações obtidas durante a coleta de dados, foram definidos os principais requisitos para a plataforma. Entre eles, destacam-se a necessidade de uma interface de fácil navegação, com conteúdos claros e objetivos, acompanhados de exemplos práticos que facilitem o aprendizado. A plataforma também deverá disponibilizar exercícios práticos e criativos, buscando fugir do modelo tradicional de ensino de programação. Além disso, deverá oferecer dicas e orientações durante as atividades, de modo a auxiliar o usuário na resolução dos exercícios.

Nesse momento, a plataforma teve sua primeira interface, sendo um protótipo minimalista com poucos elementos visuais, utilizando tecnologias básicas da web, como JavaScript, HTML e CSS. Nessa fase, a ferramenta se chamava Clean Code Academy e tinha como objetivo validar a proposta inicial da interface.

Após a criação da primeira versão, buscou-se aprimorar a identidade visual da plataforma. Em que uma das inspirações veio da capa do livro de Robert C. Martin, cuja representação remete à ideia de universo, o que motivou a mudança do nome para Clean Code Universe (Martin, 2009). Com essa nova proposta visual, foi implementado o uso da biblioteca Three.js, que criou o fundo animado com efeito de estrelas em movimento, reforçando a temática espacial. Além disso, o ambiente de desenvolvimento Antigravity IDE contribuiu para a construção da interface, enquanto que a utilização de Inteligência Artificial Generativa Gemini 3.1 Pro e o Claude IA Sonnet 5 auxiliaram na revisão do código, na identificação de problemas durante o desenvolvimento do software, na sugestão de melhorias, na elaboração de exercícios e explicações, na correção de bugs no código e na geração de imagens usadas na plataforma (Google, 2026; Anthropic, 2026).

Além disso, foram incluídos vídeos curtos do YouTube relacionados aos temas trabalhados em cada um dos módulos. Os vídeos foram integrados à plataforma por meio da API do YouTube, utilizando players dentro das lições dos módulos, sempre referenciando os canais e autores dos vídeos. A utilização dos vídeos teve como finalidade apresentar explicações complementares aos textos, imagens e exercícios presentes durante as lições dos módulos (Google, 2026).

A versão inicial da plataforma foi submetida a um teste-piloto com 5 estudantes do curso de BES da UEPA. As questões buscaram verificar a clareza dos exemplos e explicações didáticas sobre Clean Code, a contribuição da plataforma para o início da aplicação desses conceitos no dia a dia, a percepção sobre sua capacidade de introduzir os fundamentos da temática e a facilidade de navegação. As respostas obtidas orientaram em melhorias que foram implementadas no sistema. A partir dessas alterações, foi possível desenvolver a versão final da plataforma.

4.4. Demonstração

Com base nas sugestões coletadas na etapa anterior, foi desenvolvida a versão final da plataforma online Clean Code Universe. A plataforma foi disponibilizada aos estudantes do curso de Bacharelado em Engenharia de Software (BES) da Universidade do Estado do Pará (UEPA), que puderam acessá-la e testar seus recursos de forma remota. O convite para participação foi encaminhado ao grupo de alunos do curso e 17 estudantes se disponibilizaram a utilizar o sistema, o acesso ocorreu de forma online, permitindo que os participantes navegassem livremente pelos conteúdos e realizassem as atividades propostas.

4.5. Avaliação

Após a utilização da plataforma, foi realizada a avaliação da versão definitiva da plataforma, correspondendo à etapa de Avaliação da Design Science Research Methodology (DSRM). Essa etapa teve como objetivo verificar a percepção dos usuários em relação ao sistema desenvolvido, especialmente quanto à facilidade de navegação, à presença de dificuldades durante a utilização, à clareza dos exemplos e explicações e à contribuição da plataforma para a aplicação dos princípios de Clean Code.

A avaliação foi realizada por meio de um questionário online, aplicado aos 17 estudantes que utilizaram a versão definitiva da plataforma. O instrumento contemplou questões de caracterização dos participantes, incluindo idade, gênero, experiência em

programação e familiaridade com os princípios de Clean Code, além de questões relacionadas à usabilidade e à percepção dos usuários sobre a plataforma.

Para possibilitar a comparação entre as respostas obtidas no teste-piloto e na avaliação da versão final do sistema, foram mantidas as mesmas perguntas em ambas as duas etapas. Dessa forma, tornou-se possível analisar possíveis mudanças na percepção dos participantes após a implementação das melhorias realizadas na plataforma.

4.6. Comunicação dos Resultados

A etapa de comunicação refere-se ao processo de apresentação e divulgação dos dados obtidos ao longo do desenvolvimento do estudo, juntamente com a interpretação e discussão dos resultados encontrados. A comunicação dos resultados será realizada por meio da apresentação pública do Trabalho de Conclusão de Curso (TCC) e da publicação do estudo em revista científica, possibilitando a divulgação do trabalho e do conhecimento produzido, contribuindo para sua divulgação no meio acadêmico e científico.

5. RESULTADOS

Nesta seção, são apresentados os resultados obtidos em cada etapa do Design Science Research Methodology (DSRM). A organização segue as etapas descritas na metodologia, contemplando a identificação do problema, a definição dos objetivos, o desenvolvimento da solução, a demonstração e a avaliação. A comunicação dos resultados não será apresentada, já que este artigo representa parte dessa comunicação, bem como a defesa do trabalho de conclusão de curso que será apresentado após a submissão deste trabalho.

5.1. Identificação do Problema

A revisão bibliográfica indicou que os princípios de Clean Code são de fato relevantes para a formação de estudantes de programação, mas que ainda existe uma carência de recursos didáticos interativos e organizados especificamente para apoiar o ensino desse conteúdo (Araújo et al., 2020; Dietz et al., 2018; Gomes et al., 2017).

Nesse sentido, observa-se que muitos estudantes do curso de Tecnologia da Informação (TI) enfrentam dificuldades para aprender programação básica, o que dificulta ainda mais no desenvolvimento de habilidades relacionadas à criação de códigos de qualidade, como é o caso do Clean Code (Rolim et al., 2021).

Embora o conteúdo teórico seja também importante, muitos alunos apresentam limitações para aplicar esses princípios na hora de escrever códigos mais legíveis e de melhor qualidade (Dietz et al., 2018). Desta forma, a análise dos trabalhos encontrados contribuiu para confirmar a necessidade de uma ferramenta educacional que reunisse explicações teóricas, exemplos práticos e exercícios sobre código limpo.

5.2. Definição do Objetivo

A partir da identificação do problema, definiu-se como objetivo principal: desenvolver e aplicar uma plataforma online voltada ao ensino de Clean Code com um grupo de alunos do curso de Engenharia de Software da Universidade do Estado do Pará (UEPA).

5.3. Desenvolvimento da Solução

Conforme explicado na seção 4.3, todos os 16 participantes ouvidos na etapa de levantamento de requisitos consideraram que um código organizado é importante para o desenvolvimento de software. Quanto à preocupação em manter o código claro e bem estruturado, entre os 16 participantes, 11 apresentaram um posicionamento favorável, enquanto apenas um participante discordou e 4 permaneceram neutros.

Sobre se os conteúdos estudados em salas de aula ou em cursos livres online eram suficientes para aplicar conceitos do Clean Code no desenvolvimento de software, observou-se que 7 alunos indicaram que os conteúdos não são suficientes, enquanto 6 permaneceram neutros e 3 indicaram que são suficientes. Esse resultado indica que parte significativa dos estudantes não considerava os conteúdos disponíveis suficientes para a aplicação prática dos princípios de Clean Code.

Além disso, entre os 16 participantes, 14 consideraram que uma plataforma poderia auxiliar no processo de ensino e aprendizagem, enquanto 2 estudantes permaneceram neutros.

Quando questionados sobre qual linguagem de programação tinham maior interesse em aprender ou aprimorar, Python foi a mais escolhida, sendo indicada por sete participantes. Em seguida, Java apresentou o segundo maior nível de interesse, com seis participantes. As linguagens C, Kotlin e JavaScript também foram mencionadas, porém com menor frequência. Dessa forma, essa preferência foi considerada na definição da linguagem de programação utilizada nos exemplos e exercícios propostos na plataforma desenvolvida.

Em relação às funcionalidades desejadas na plataforma, os participantes destacaram a necessidade de receberem dicas de boas práticas de codificação, contar com recursos de Inteligência Artificial para auxiliar na análise e nos comentários sobre o código escrito, realizar exercícios práticos e aprender a escrever código de forma mais limpa por meio de atividades práticas, com menor ênfase em conteúdos muito teóricos. Essas informações contribuíram para adoção de recursos pedagógicos e tecnologia para plataforma online. A utilização de Inteligência Artificial ficou definida para trabalhos futuros.

5.3.1. Resultados do Teste-Piloto

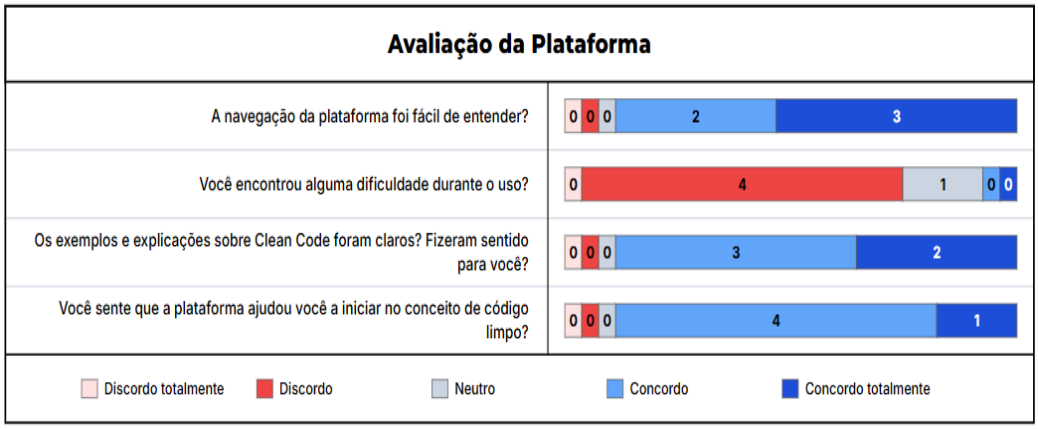
Conforme apresentado na seção 4.3, 5 alunos participaram do teste-piloto, no intuito de avaliar a primeira versão do sistema, antes de aplicá-lo com mais alunos. Neste sentido, o Gráfico 1 apresenta um resumo dos principais resultados obtidos.

Sobre a facilidade de navegação da plataforma, 2 participantes marcaram “concordo totalmente”, enquanto 3 “concordaram”, o que mostra que nenhum aluno teve dificuldades em navegar na ferramenta.

Na questão da existência de dificuldades durante o uso da plataforma, 1 participante permaneceu “neutro” enquanto 4 “discordaram”, o que indica que a maioria não encontrou obstáculos significativos na utilização da ferramenta.

Quanto à clareza dos exemplos e explicações, 2 participantes assinalaram “concordo totalmente” e 3 marcaram “concordo”, indicando uma avaliação positiva do conteúdo apresentado.

Gráfico 1. Avaliação do Teste-Piloto



Fonte: autor, 2026

Sobre a contribuição da plataforma para que os estudantes começassem a aplicar os princípios de Clean Code no dia a dia, 1 participante respondeu “concordo totalmente” e 4 assinalaram “concordo”, demonstrando que conseguiriam aplicar os conceitos do Clean Code.

Apesar da avaliação favorável, os participantes também indicaram oportunidades de melhorias. Entre as principais respostas apresentadas pelos participantes, destacaram-se:

- A necessidade de um menu de navegação mais acessível que permitisse acesso mais rápido aos tópicos de cada módulo, uma vez que na versão do teste-piloto o deslocamento ocorria apenas por meio das setas de página;
- A possibilidade de reduzir a quantidade de páginas em cada módulo, simplificando e tornando a leitura mais prática e menos monótona;
- A inclusão de mais conteúdos sobre as técnicas e os princípios de Clean Code, para atender isso foi adicionado um novo módulo com a adição de novas técnicas e princípios sobre Clean Code.

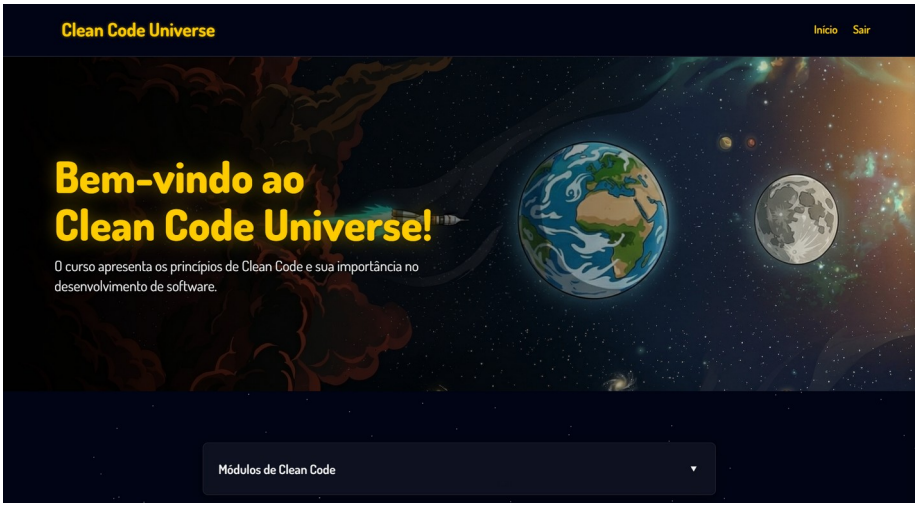
5.4. Demonstração

Como indicado na seção 4.4, alunos do curso de Bacharelado em Engenharia de Software da UEPA foram convidados a participarem do estudo, com 17 estudantes se disponibilizando a realizarem os testes da versão final da solução proposta. Com isso, o link da ferramenta foi disponibilizado de forma online, permitindo que os participantes utilizassem livremente a ferramenta, lendo os conteúdos, que foram apresentados de forma breve e simples, como sugerido no piloto e com apoio de vídeos e exercícios práticos.

A Figura 1 apresenta a tela inicial da plataforma Clean Code Universe, que possui um plano de fundo com a temática espacial, composto por planetas, estrelas, além de uma mensagem de boas-vindas ao usuário. Na parte inferior da tela, é possível visualizar o painel lateral com os módulos da plataforma, que permite a navegação entre os diferentes módulos da plataforma.

A versão final foi organizada em quatro módulos principais: Módulo 1 – Introdução ao Clean Code; Módulo 2 – Princípio da Simplicidade; Módulo 3 – Comentários Limpos; e Módulo 4 – Erros Claros.

Figura 1. Tela Inicial da Plataforma



Fonte: Autor, 2026

Cada módulo foi estruturado com diferentes recursos didáticos, incluindo textos autoexplicativos, imagens ilustrativas, vídeos curtos relacionados aos temas abordados e exercícios de fixação.

Na Figura 2, é apresentado o formato das páginas de conteúdo dos módulos da plataforma. Como pode ser observado, a página contém um título, um texto explicativo e uma imagem, além de funcionalidades complementares que facilitam a navegação. Para facilitar a navegação, a plataforma disponibiliza botões para avançar e voltar entre as páginas, a indicação do número da página e um sumário, que permite acessar rapidamente diferentes partes do conteúdo. Além disso, as imagens podem ser ampliadas ao clicar sobre elas com o cursor do mouse.

Figura 2. Conteúdo de Ensino da Plataforma

Fonte: Autor, 2026

Em relação à figura 3, é mostrada uma das telas de exercícios propostos nos módulos, cujo objetivo é refatorar ou reescrever o código seguindo as boas práticas ensinadas de Clean Code. Pode-se observar que o exercício é composto por um editor de código em Python, acompanhado de um ícone de ajuda que fornece dicas e orientações sobre o que deve ser realizado. Além disso, há o botão Executar, que permite ao usuário executar o código refatorado e verificar seu funcionamento.

Na Figura 3, também é possível visualizar o botão de navegação localizado no lado direito da tela, mudado para a coloração em vermelha. Essa mudança indica que o botão está

temporariamente indisponível, sinalizando que o exercício ainda não foi concluído. Para habilitar a navegação e permitir o avanço para as lições seguintes, o estudante deve primeiro finalizar a atividade presente no módulo.

Dessa forma, os estudantes puderam estudar os conceitos apresentados, visualizar exemplos e testar seus conhecimentos ao final de cada módulo. A versão final incorporou melhorias identificadas no teste-piloto, como a reorganização da navegação e o aprimoramento da estrutura dos conteúdos.

Figura 3. Exercício de Refatoração do Código

Fonte: Autor, 2026

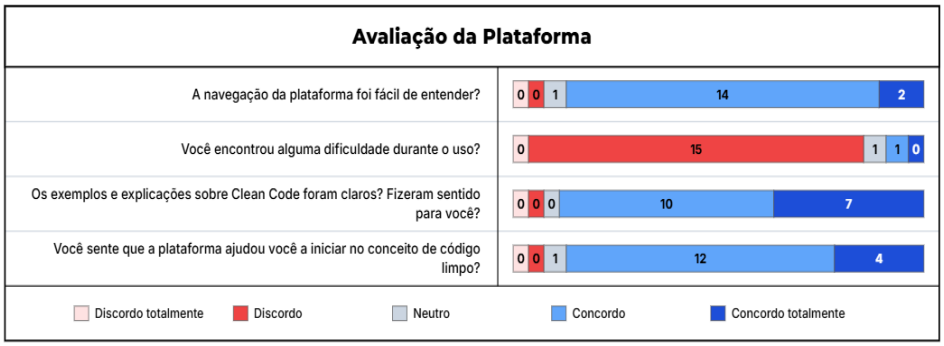
5.5. Avaliação da Versão Final

Em relação a avaliação da versão definitiva da plataforma, conforme apresentado na seção 4.5, participaram, ao todo, 17 estudantes, que responderam ao terceiro questionário, representando um número maior de participantes em comparação com a etapa anterior, que

contou com apenas 5 estudantes. Todos os estudantes que se disponibilizaram a utilizar a plataforma responderam ao questionário , possibilitando a obtenção de dados referentes à percepção dos usuários sobre a versão final do sistema.

Dessa forma, o Gráfico 2 apresenta as mesmas perguntas analisadas no Gráfico 1 do teste-piloto, porém com um número maior de resposta, permitindo avaliar a evolução da plataforma a partir da versão final.

Gráfico 2. Avaliação da Versão Final



Fonte: autor, 2026

Em relação à facilidade de navegação da plataforma, 2 participantes marcaram “concordo totalmente”, enquanto 14 apenas “concordaram” e 1 permaneceu “neutro”. Já sobre a existência de dificuldades durante o uso da plataforma, 1 participante assinalou “concordo”, 1 permaneceu “neutro” e 15 “discordaram”, indicando que novamente a plataforma não apresentou obstáculos significativos para os alunos.

Quanto à clareza dos exemplos e explicações, 7 participantes assinalaram “concordo totalmente” e 10 marcaram “concordo”, indicando uma avaliação positiva do conteúdo apresentado. Em relação à contribuição da plataforma para começar a aplicar Clean Code no dia a dia, 4 participantes responderam “concordo totalmente”, 12 assinalaram “concordo” e 1 permaneceu “neutro", mostrando, também, uma avaliação positiva do conteúdo abordado sobre Clean Code na plataforma.

6. CONCLUSÃO

O estudo conduzido neste trabalho destacou a relevância da qualidade do código na indústria de software, especialmente para o desenvolvimento de aplicações de fácil manutenção. Apesar de sua importância, observa-se que o ensino tradicional nem sempre oferece aos estudantes oportunidades suficientes para desenvolver, de forma prática, competências relacionadas à produção de código de qualidade. Nesse contexto, a plataforma online Clean Code Universe foi desenvolvida como uma alternativa educacional baseada no ensino dos princípios de Clean Code.

A plataforma Clean Code Universe tem como diferencial oferecer um ambiente de aprendizagem controlado e estruturado, que vai além de uma simples ferramenta de exercícios. Cada módulo combina imagens, vídeos, textos explicativos e atividades práticas, organizados de forma progressiva para apoiar o processo de aprendizagem. Além disso, a plataforma utiliza diferentes formatos de atividades interativas, como movimentação de blocos, associação de conceitos e tarefas que estimulam o raciocínio e a aplicação dos conteúdos abordados, sempre com foco em simplicidade e objetividade na apresentação dos conteúdos (poucos textos e vídeos curtos, por exemplo).

Além disso, a plataforma também permite que o estudante possa escrever e compilar código diretamente no ambiente, recebendo feedback imediato sobre a execução e a qualidade do código. Durante a realização das atividades, são disponibilizadas dicas contextualizadas e sugestões de ajustes, auxiliando o estudante a refletir sobre suas escolhas e a desenvolver gradualmente suas práticas de Clean Code.

A avaliação da versão final, realizada com 17 estudantes, apresentou resultados positivos quanto à facilidade de navegação, à ausência de dificuldades significativas durante o uso, à clareza dos exemplos e das explicações e à contribuição da plataforma para o início da aplicação dos princípios de Clean Code no dia a dia. Esses resultados indicam o potencial da plataforma como recurso de apoio ao processo de ensino e –aprendizagem, embora não

permitam afirmar, isoladamente, a efetiva consolidação desses conhecimentos pelos participantes.

Como limitação, destaca-se o número reduzido de participantes e o fato da avaliação ter sido baseada principalmente na percepção dos estudantes sobre a experiência de uso da plataforma, não permitindo verificar diretamente a aprendizagem ou a capacidade de aplicação prática dos princípios de Clean Code. Assim, para trabalhos futuros, recomenda-se realizar estudos com uma quantidade maior de estudantes, além de aplicar instrumentos capazes de verificar a aprendizagem e a aplicação prática dos princípios de Clean Code.

Também seria relevante acompanhar os participantes por um período mais longo e comparar seu desempenho antes e depois da utilização da plataforma, ampliando a validade dos resultados obtidos. Além disso, propõe-se a utilização de recursos de Inteligência Artificial como uma possibilidade para trabalhos futuros, visando investigar seu potencial como ferramenta de apoio à aprendizagem e à aplicação dos princípios de Clean Code.

REFERÊNCIAS

ANTHROPIC. *Anthropic*. 2026. Disponível em: https://www.anthropic.com/. Acesso em: 10 maio 2026.

ARAÚJO, E. G. Teacher Mate: a support tool for teaching code quality. In: INTERNATIONAL CONFERENCE ON INTELLIGENT TUTORING SYSTEMS (ITS), 16., 2020. p. 1–6.

DIETZ, Linus; MANNER, Johannes; HARRER, Simon; LENHARD, Jörg. Teaching *Clean Code*. 2018.

FERNANDES, Ramon Santos. Modelos de processos de engenharia de software. *Aurum Editora*, [S. l.], p. 112–122, 2025. DOI: https://doi.org/10.63330/aurumpub.005-011. Disponível em: https://aurumpublicacoes.com/index.php/editora/article/view/144.

GOOGLE. *Google Acadêmico*. 2026. Disponível em: https://scholar.google.com/. Acesso em: 10 maio 2026.

GOMES, P. H. et al. Teaching software quality via source code inspection tool. In: IEEE FRONTIERS IN EDUCATION CONFERENCE (FIE), 2017. *Proceedings...* IEEE, 2017. p. 1–8.

GOMES, P. H. D. A. Inspeção de código-fonte como subsídio. 2021. p. 1–145.

JOSHUA, Jonah Vincent; ANDEME, Diosdado Asumu Ndong. The evolution of software engineering: from prehistoric beginnings to the age of artificial intelligence. *International Journal of Computer and Information Technology*, v. 14, n. 1, 2025. DOI: https://doi.org/10.24203/8rbykd84. Disponível em: https://www.ijcit.com/index.php/ijcit/article/view/497.

KARABEGOVIĆ, Isak (org.). *Proceedings of International Scientific Conference: Basic Technologies and Models for Implementation of Industry 4.0*. Sarajevo: Academy of Sciences and Arts of Bosnia and Herzegovina, 2023.

LIUTAK, O. M.; BAULA, O. V.; SHULYAK, A. P. The influence of global world market transformations information and communication technologies for international competitiveness. *Aktual'ni Problemy Ekonomiky = Actual Problems in Economics*, Kiev, ed. 231, p. 86–95, 2020. DOI: https://doi.org/10.32752/19936788201912318695.

LUBURIĆ, Nikola; VIDAKOVIĆ, Dragan; SLIVKA, Jelena; PROKIĆ, Simona; GRUJIĆ, Katarina; GLORIJA, Kovacevic; KOVACEVIC, Aleksandar; SLADIĆ, Goran. *Clean Code tutoring: makings of a foundation*. 2022. p. 137–148. DOI: https://doi.org/10.5220/0010800900003182.

MARTIN, Robert C. *Clean Code: a handbook of agile software craftsmanship*. 1. ed. Upper Saddle River: Prentice Hall, 2009.

MOHAMMED, F. S.; OZDAMLI, F. A systematic literature review of soft skills in information technology education. *Behavioral Sciences*, v. 14, n. 10, p. 894, 2024. DOI: https://doi.org/10.3390/bs14100894.

NUROLLAHIAN, Sara; KEUNING, Hieke; WIESE, Eliane. Teaching well-structured code: a literature review of instructional approaches. 2025. Preprint. DOI: https://doi.org/10.48550/arXiv.2502.11230.

ORISA, Mira; FAISOL, Ahmad; ASHARI, Mochammad Ibrahim. Perancangan website company profile menggunakan Design Science Research Methodology (DSRM). *Jurnal Informatika Teknologi dan Sains (JINTEKS)*, v. 5, n. 1, p. 160–164, 2023.

OTTER.AI. *Elicit*. 2026. Disponível em: https://elicit.com/. Acesso em: 30 mar. 2026.

PEFFERS, Ken; TUUNANEN, Tuure; ROTHENBERGER, Marcus A.; CHATTERJEE, Samir. A design science research methodology for information systems research. *Journal of Management Information Systems*, v. 24, n. 3, p. 45–77, 2007. DOI: https://doi.org/10.2753/MIS0742-1222240302.

PINHEIRO SANTIAGO, C.; MENDONÇA MENEZES, J. W.; ALVES DE AQUINO, F. J. Proposta e avaliação de uma metodologia de aprendizagem baseada em projetos em disciplinas de engenharia de software através de uma sequência didática. *Revista Brasileira de Informática na Educação*, v. 31, p. 31–59, 2023. DOI: https://doi.org/10.5753/rbie.2023.2817. Disponível em: https://journalssol.sbc.org.br/index.php/rbie/article/view/2817.

PORTO, M. Código limpo: a importância da refatoração. 2022. p. 56–64.

ROLIM, M. C. T. Investigando as dificuldades e perspectivas sobre um curso de engenharia de software de dois ciclos: um survey com visão do discente. 26 abr. 2021. p. 55–65.

SABRI, Mohammad Omar. The embracement of Design Science Research Methodology in the development of digital transformation models. *WSEAS Transactions on Business and Economics*, v. 22, p. 12–19, 2025. DOI: https://doi.org/10.37394/23207.2025.22.2.

SANTANA, A. C. de A.; NARCISO, R. Pilares da pesquisa educacional: autores e metodologias científicas em destaque. *Aracê*, [S. l.], v. 7, n. 1, p. 1577–1590, 2025. DOI: https://doi.org/10.56238/arev7n1-095. Disponível em: https://periodicos.newsciencepubl.com/arace/article/view/2782.

SANTOS, E. D.; OLIVEIRA, S. R. B. Análise da diversidade no ensino de engenharia de requisitos de software: uma revisão sistemática da literatura. *Caderno Pedagógico*, v. 21, n. 4, e3724, 2024. DOI: https://doi.org/10.54033/cadpedv21n4079.

SCIELO. *Scientific Electronic Library Online*. 2026. Disponível em: https://scielo.org/. Acesso em: 30 mar. 2026.

SEMERIKOV, Serhiy; STRIUK, Andrii; STRIUK, Larysa; STRIUK, Mykola; SHALATSKA, Hanna. Sustainability in software engineering education: a case of general professional competencies. *E3S Web of Conferences*, v. 166, p. 10036, 2020. DOI: https://doi.org/10.1051/e3sconf/202016610036.

SOFTWARE ENGINEERING: FIRST 50 YEARS OF FORMATION AND DEVELOPMENT. In: STUDENT WORKSHOP ON COMPUTER SCIENCE & SOFTWARE ENGINEERING – CS&SE@SW, 1., 2018, Kryvyi Rih, Ukraine. *Proceedings...* CEUR Workshop Proceedings, v. 2292, 2018.

TYAGI, Bhaumik; BEG, Yusra. An experimental assessment on effects of SOLID design principles on the quality of software using CKJM metric analysis. *International Journal for Research in Applied Science and Engineering Technology*, v. 10, n. 9, p. 1260–1266, 2022.

VALENTAITĖ, Karolina; BALIUTĖ, Asta. Makroekonominių veiksnių poveikis finansinių technologijų verslo modelių plėtrai COVID-19 pandemijos metu. 2022.

Recebido em: 05/08/2026
Aprovado em: 24/08/2026
Publicado em: 16/09/2026