REVISTA TÓPICOS
ISSN 2965-6672 | Qualis A2

CONFORMIDADE COM A LGPD NO DESENVOLVIMENTO DE SOFTWARE ASSISTIDO POR INTELIGÊNCIA ARTIFICIAL: UMA REVISÃO SISTEMÁTICA DA LITERATURA SOBRE VULNERABILIDADES DE PRIVACIDADE E VAZAMENTO DE DADOS

COMPLIANCE WITH THE LGPD IN ARTIFICIAL INTELLIGENCE-ASSISTED SOFTWARE DEVELOPMENT: A SYSTEMATIC LITERATURE REVIEW ON PRIVACY VULNERABILITIES AND DATA LEAKAGE

Ciências Exatas e da Terra, Engenharias • 26/11/2026

REGISTRO DOI: 10.70773/revistatopicos/795569470

Yago da Costa Jardim Alves Braga[1]

Caio Cezar Jotta Nogueira[2]

Daniel Monteiro de Carvalho[3]

Gabriel Victor Martins Carvalho[4]

Bernardo Cardador Lobo[5]

Christiane Martins de Vasconcellos Silveira[6]

Wellington Ávila[7]

RESUMO

Assistentes de codificação baseados em grandes modelos de linguagem (LLMs) aceleram o desenvolvimento de software, mas trazem riscos de vazamento de dados e código inseguro. Esta Revisão Sistemática da Literatura (RSL) investiga vulnerabilidades técnicas e riscos de privacidade diante da Lei Geral de Proteção de Dados Pessoais (LGPD – Lei nº 13.709/2018). A revisão seguiu o protocolo PRISMA 2020, de janeiro de 2022 a abril de 2026, nas bases IEEE Xplore, ACM Digital Library, OpenAlex e SBC OpenLib, além de snowballing e fontes normativas. Dos 1.405 registros, 22 compuseram o corpus final: 18 estudos empíricos primários (15 internacionais e 3 nacionais) e 4 documentos normativos e técnicos. Embora reduzam o tempo de codificação em 55,8% em experimento controlado, os assistentes trazem viés de superconfiança, revisão mais lenta em alterações de segurança e falta de proteção nos *prompts*: 32,1% dos profissionais brasileiros nunca anonimizam dados nas consultas, e os termos de uso transferem a responsabilidade ao usuário. No código gerado, repetem-se fraquezas da taxonomia CWE (CWE-89, CWE-78, CWE-22, CWE-502, CWE-330, CWE-798 e CWE-532). Pelo mapeamento técnico-normativo proposto, essas falhas podem configurar tratamento irregular de dados (art. 44) e ferir a segurança desde a concepção (art. 46), enquanto o envio de dados pessoais em *prompts* a servidores no exterior pode caracterizar transferência internacional (art. 33). A resposta a esses riscos está em controles em esteiras *DevSecOps*: filtragem de dados sensíveis nos *prompts*, análise estática e dinâmica do código e revisão humana obrigatória.

Palavras-chave: Assistentes de Codificação por IA; DevSecOps; Privacy by Design; Vulnerabilidades de Software; Lei Geral de Proteção de Dados (LGPD); Revisão Sistemática da Literatura.

ABSTRACT

Coding assistants based on large language models (LLMs) speed up software development, but introduce data leakage risks and security vulnerabilities. This Systematic Literature Review (SLR) examines these tools' technical weaknesses and privacy risks under the Brazilian General Data Protection Law (LGPD – Law No. 13,709/2018). The review followed the PRISMA 2020 protocol, covering January 2022 to April 2026 across IEEE Xplore, ACM Digital Library, OpenAlex, and SBC OpenLib, complemented by snowballing and regulatory sources. Of 1,405 identified records, 22 documents formed the final corpus: 18 primary empirical studies (15 international and 3 Brazilian) and 4 normative and technical references. Results show that although coding assistants reduced task completion time by 55.8% in a controlled experiment, they bring developer overconfidence, longer security-change reviews, and weak protection of *prompt* inputs: 32.1% of surveyed Brazilian practitioners never anonymize data in *prompts*, and terms of service shift that responsibility to the user. Generated code flaws include SQL and command injection, path traversal, insecure deserialization, weak random number generation, hard-coded credentials, and sensitive data in logs (CWE-89, CWE-78, CWE-22, CWE-502, CWE-330, CWE-798, and CWE-532). Under the analytical mapping proposed, such flaws may constitute irregular data processing (Art. 44) and breach privacy-by-design obligations (Art. 46), whereas sending personal data in *prompts* to foreign servers may qualify as an international data transfer (Art. 33). The response to these risks lies in *DevSecOps* controls: *prompt* sanitization, static and dynamic code analysis, and mandatory human review.

Keywords: AI Coding Assistants; DevSecOps; Privacy by Design; Software Vulnerabilities; General Data Protection Law (LGPD); Systematic Literature Review.

1. INTRODUÇÃO

Assistentes de código baseados em grandes modelos de linguagem (large language models – LLMs), como GitHub Copilot, Cursor, ChatGPT e Claude, já fazem parte da rotina dos desenvolvedores. Funcionam no navegador ou dentro da IDE, e mudaram como se escreve, refatora e testa código. No experimento de Peng *et al.* (2023) com o GitHub Copilot, programadores assistidos pela ferramenta terminaram a tarefa 55,8% mais rápido que o grupo de controle. Isso ajudou a espalhar o *vibe coding*, prática descrita por Karpathy (2025) e analisada por Meske *et al.* (2025). O desenvolvedor delega blocos inteiros à IA, aceita as sugestões sem revisão adequada e só identifica a falha durante a execução.

Esse ganho de velocidade traz riscos em duas frentes: nos dados enviados ao modelo (fluxo upstream) e no código gerado em resposta (fluxo downstream). No upstream, a facilidade de interação em linguagem natural leva desenvolvedores a inserir dados corporativos, credenciais de acesso e dados pessoais em servidores de terceiros. Esse risco é agravado pelas próprias políticas comerciais das ferramentas, que costumam transferir contratualmente toda a responsabilidade por dados e segredos expostos para o próprio usuário (Al-Maamari, 2025). Além disso, um estudo experimental demonstra que modelos de código memorizam informações pessoais durante o treinamento e podem reproduzi-las caso recebam comandos específicos para isso (Niu *et al.*, 2023).

No fluxo downstream, o código retornado pelo modelo também traz vulnerabilidade. Os assistentes reproduzem, quando sugerem código, erros recorrentes nos repositórios públicos sobre os quais foram treinados, configurando, na interpretação dos autores, uma

herança de antipadrões dos dados de treinamento: injeções de SQL, comandos arbitrários, travessia de diretório, desserialização insegura, geradores aleatórios previsíveis e credenciais embutidas no código-fonte (hard-coded) (Cotroneo; Improta; Liguori, 2025; Fu *et al.*, 2025; Khoury *et al.*, 2023; Majdinasab *et al.*, 2024; Pearce *et al.*, 2022; Zhao *et al.*, 2025). O desenvolvedor tende a confiar excessivamente nessas sugestões e a considerar o código seguro, mesmo quando contém falhas graves (Perry *et al.*, 2023).

Na prática, grande parte do tempo de verificação é consumida avaliando e editando sugestões sucessivas, atividade que, na avaliação dos autores, concentra-se em fazer o código funcionar, e não em auditar sua segurança (Mozannar *et al.*, 2024). Mesmo em um estudo com usuários em linguagem C, as taxas de vulnerabilidades críticas de memória ficaram próximas às do grupo sem assistência (Sandoval *et al.*, 2023). Os modelos também alucinam pacotes inexistentes, o que ameaça a cadeia de suprimentos (Spracklen *et al.*, 2025), e as ferramentas de detecção automática geram falsos positivos e fadiga de alertas (Harzevili *et al.*, 2025; Santos *et al.*, 2026).

Referenciais internacionais, como o OWASP Top 10 for LLM Applications, tratam essas fragilidades sob uma perspectiva puramente técnica de cibersegurança (OWASP, 2025). No Brasil, as pesquisas em computação têm investigado a conformidade com a Lei Geral de Proteção de Dados Pessoais (LGPD – Lei nº 13.709/2018) (Brasil, 2018) por vários ângulos: auditorias de servidores expostos na internet (Ponce *et al.*, 2023), modelos de maturidade de governança (Saraiva; Souza; Soares, 2024), listas de verificação (checklists) de conformidade (Frej *et al.*, 2024) e a percepção de profissionais de desenvolvimento sobre riscos de privacidade no uso de LLMs

(Falcão; Canedo, 2024). Nenhum desses trabalhos, porém, cruza sistematicamente o catálogo de vulnerabilidades de código (CWEs) introduzidas por assistentes de IA com os dispositivos de segurança e de responsabilização da legislação nacional. Muitos programadores receiam compartilhar informações com esses assistentes (Madampe; Grundy; Arachchilage, 2025). Soma-se a isso a dificuldade de transformar exigências legais de privacidade em decisões concretas de código (Horstmann *et al.*, 2024). Persiste, portanto, a necessidade de articular essas duas dimensões: como o vazamento upstream e as falhas técnicas downstream geradas por IA afetam o tratamento irregular (art. 44), os deveres de segurança e a privacidade desde a concepção do software (privacy by design, art. 46), o envio de dados para servidores no exterior (art. 33) e a responsabilização civil em caso de vazamento (arts. 42 a 45).

Para preencher essa lacuna, este artigo apresenta uma Revisão Sistemática da Literatura (RSL), estruturada segundo as diretrizes de Kitchenham e Charters (2007) e o protocolo PRISMA 2020 (Page *et al.*, 2021). O trabalho é orientado pelo seguinte problema de pesquisa:

De que maneira as evidências empíricas da literatura sobre assistentes de código por IA caracterizam os riscos de exposição de dados e a geração de código vulnerável perante a LGPD, e quais contramedidas técnicas são recomendadas?

Para responder a essa questão, foram formuladas três perguntas de pesquisa (Research Questions – RQs):

- RQ1 (Fatores Comportamentais e Fluxo Upstream): Quais fatores humanos, vieses cognitivos e políticas comerciais das

ferramentas favorecem a exposição indevida de dados no desenvolvimento assistido por IA?

- RQ2 (Vulnerabilidades no Código Downstream): Quais padrões de falhas técnicas (CWEs) com impacto sobre a segurança e a privacidade dos dados são recorrentemente introduzidos no código gerado por LLMs?

- RQ3 (Enquadramento Legal e Mitigações): De que forma as fragilidades mapeadas se relacionam às exigências da LGPD e quais salvaguardas técnicas contribuem para a conformidade no ciclo de desenvolvimento de software?

Ao longo das próximas seções, o estudo: (i) mapeia os fatores comportamentais e contratuais que provocam vazamentos no envio de prompts (upstream); (ii) organiza uma taxonomia prática das vulnerabilidades (CWEs) presentes no código gerado por IA (downstream); e (iii) elabora uma proposta analítica preliminar de mapeamento técnico-normativo relacionando essas fraquezas técnicas aos artigos e deveres previstos na LGPD, propondo salvaguardas aplicáveis a esteiras de desenvolvimento e segurança (DevSecOps).

O restante do artigo está organizado da seguinte forma: a Seção 2 detalha a metodologia da revisão sistemática, incluindo critérios de seleção, buscas e avaliação de qualidade; a Seção 3 apresenta os resultados para cada pergunta de pesquisa; a Seção 4 discute os achados de forma integrada à realidade regulatória e de mercado; e a Seção 5 traz as considerações finais, limitações do estudo e propostas de trabalhos futuros.

2. METODOLOGIA

Esta pesquisa seguiu as diretrizes de Kitchenham e Charters (2007) para revisões sistemáticas em Engenharia de Software e o protocolo PRISMA 2020 (Page *et al.*, 2021), para garantir transparência e reprodutibilidade. O processo teve três etapas: (i) elaborar o protocolo e definir as perguntas de pesquisa; (ii) executar as buscas, a triagem e a avaliação de qualidade; e (iii) extrair e sintetizar os dados. As buscas foram iniciadas entre janeiro e fevereiro de 2026 e atualizadas até abril de 2026 para capturar publicações recentes, cobrindo o período de janeiro de 2022 a abril de 2026.

2.1. Critérios de Elegibilidade e Estratégia de Busca

As consultas estruturadas foram realizadas em quatro bases científicas principais: IEEE Xplore, ACM Digital Library, OpenAlex e SBC OpenLib (SOL), complementadas pela inclusão de documentos normativos oficiais da ANPD e da legislação federal, referenciais técnicos da OWASP e estudos identificados por rastreamento manual de referências (snowballing).

A estratégia de busca dividiu-se em três eixos terminológicos, aplicados aos campos de metadados de cada repositório (IEEE Xplore em All Metadata, ACM Digital Library apenas no campo Abstract, OpenAlex em título e resumo e SBC OpenLib em Busca Integrada), conforme sintetizado na Tabela 1.

Tabela 1. Estrutura das sub-strings de busca por eixo temático, escopo e bases consultadas.

Eixo Temático	**Strings Exatas Executadas e Campos Consultados por Base (Período: 2022–abril/2026)**	**Idioma**	**Bases Consultadas**

Eixo 1: Fatores Comportamentais, Cognitivos e Produtividade (RQ1)	String principal (IEEE Xplore [All Metadata], ACM DL [campo Abstract] e OpenAlex [title_and_abstract.search]): ("AI coding assistant" OR "AI-based coding assistants" OR "AI-powered coding assistants" OR "AI-based programming assistants" OR "GitHub Copilot" OR "LLM-assisted programming") AND (productivity OR "developer behavior" OR "cognitive load" OR "overreliance" OR "human factors" OR "privacy" OR "user study")	Inglês	IEEE Xplore (n = 84), ACM Digital Library (n = 37), OpenAlex (n = 523)
Eixo 2: Vulnerabilidades Técnicas e CWEs em Código de LLMs (RQ2)	String principal (IEEE Xplore [All Metadata], ACM DL [campo Abstract] e OpenAlex [title_and_abstract.search]): ("GitHub Copilot" OR "GitHub's Copilot" OR "AI coding assistant" OR "coding agents" OR "code generated by ChatGPT" OR "AI-generated code") AND ("CWE" OR "Common Weakness Enumeration" OR "security weaknesses" OR "vulnerability" OR "vulnerabilities" OR "insecure code")	Inglês	IEEE Xplore (n = 69), ACM Digital Library (n = 23), OpenAlex (n = 360)
Eixo 3: Governança, Proteção de Dados e LGPD (RQ3)	OpenAlex [title_and_abstract.search]: ("LGPD" OR "Lei Geral de Proteção de Dados" OR "Brazilian General Data Protection Law") AND	Português e Inglês	SBC OpenLib – SOL (n = 50), OpenAlex (n = 252)

	("software engineering" OR "engenharia de software" OR "software development" OR "desenvolvimento de software" OR "indústria de software" OR "sistema web" OR "vulnerabilidades") SBC OpenLib – SOL [Busca Integrada, filtro 2022–2026, adaptada à sintaxe da base]: LGPD software		

Fonte: *Elaborada pelos autores (2026).*

As consultas nas quatro bases retornaram 1.398 registros; 7 registros complementares foram adicionados (3 via snowballing, conforme Wohlin (2014): Harzevili *et al.* (2025), Horstmann *et al.* (2024) e Santos *et al.* (2026); e 4 de repositórios oficiais: LGPD, Guia e Resolução ANPD, OWASP Top 10 LLM), totalizando 1.405 identificados. Após a remoção de 265 duplicatas, 1.133 registros únicos das bases foram triados por título e resumo, e os 7 complementares ingressaram diretamente na leitura integral (1.140 no total); 46 registros das bases mais os 7 complementares, 53 no total, avançaram para a elegibilidade, dos quais 22 compuseram o corpus final. O fluxo completo e os quantitativos por etapa constam na Figura 1 e nas Tabelas 2 e 3.

As quatro bases foram escolhidas porque se complementam: a IEEE Xplore e a ACM Digital Library concentram os principais veículos de Engenharia de Software e Segurança da Informação; a SBC OpenLib cobre a produção científica nacional; e a OpenAlex amplia a cobertura para periódicos e anais de múltiplas editoras, com buscas abertas e reprodutíveis. A OpenAlex é um agregador aberto, sucessor do Microsoft Academic Graph, que consolida metadados

de Crossref, PubMed e outras fontes abertas e cobre a maior parte da literatura também indexada por Scopus e Web of Science. A sobreposição entre as fontes aparece nas 141 duplicatas interbasais, 10,09% dos 1.398 registros retornados, indicando que as bases se complementam sem serem integralmente redundantes. Scopus e Web of Science ficaram fora por decisão de escopo do protocolo, que priorizou os principais veículos da área (IEEE e ACM), a produção nacional (SBC) e um agregador aberto de ampla cobertura (OpenAlex), e não por falta de acesso. Essa ausência é uma limitação declarada desta revisão, e replicar as buscas nessas bases fica como agenda de continuidade.

Para a seleção dos documentos, estabeleceu-se a seguinte regra lógica de elegibilidade:

Regra de Elegibilidade = CI1 E (CI2 OU CI3)

Isto é, o documento deveria cumprir obrigatoriamente o critério temporal e de publicidade formal (CI1) somado a pelo menos um dos eixos temáticos (o eixo técnico-empírico de assistentes de IA, CI2, ou o eixo normativo de governança e proteção de dados, CI3). A definição formal dos critérios de inclusão (CI) e exclusão (CE) compreendeu:

Critérios de Inclusão (CI):

- CI1 (Obrigatório): Artigos científicos revisados por pares publicados em periódicos ou anais de conferências indexadas, bem como relatórios técnicos e preprints depositados no repositório arXiv vinculados a instituições acadêmicas ou laboratórios de pesquisa em Engenharia de Software e Segurança, publicados entre janeiro de 2022 e abril de 2026,

redigidos em português ou inglês, ressalvando-se marcos legislativos e normativos fundamentais que balizam a matéria (como a Lei nº 13.709/2018 e o Guia de Segurança da ANPD de 2021);

- CI2 (Eixo Técnico-Empírico de Assistentes de IA): Estudos primários que avaliem empiricamente assistentes de codificação ou modelos de linguagem na geração de código ou na tomada de decisão de desenvolvedores (produtividade, dinâmicas comportamentais, políticas contratuais de privacidade, segurança do código gerado ou introdução de CWEs);

- CI3 (Eixo Normativo e Governança de Proteção de Dados): Documentos normativos oficiais, diretrizes regulatórias, referenciais técnicos de segurança e pesquisas empíricas primárias sobre governança da segurança da informação, privacidade e conformidade com a LGPD no desenvolvimento de software e em infraestruturas computacionais brasileiras.

Critérios de Exclusão (CE):

- CE1: Estudos publicados fora do recorte temporal de janeiro de 2022 a abril de 2026 (exceto marcos normativos) ou trabalhos desprovidos de conexão direta com a geração de código por LLMs ou com a governança da LGPD (incluindo trabalhos focados exclusivamente em ensino introdutório de programação, reparo automatizado de código, defesa contra ataques adversariais a modelos ou uso de LLMs apenas como substitutos de ferramentas SAST de detecção);

- CE2: Artigos de opinião, notas informativas, editoriais curtos, revisões secundárias da literatura, ensaios conceituais/arquiteturais sem experimentação empírica primária concluída, relatos preliminares de pesquisa em andamento ou estudos com perda/incompletude dos registros experimentais reportada na coleta;

- CE3: Trabalhos cujo texto completo não estivesse disponível para verificação independente no período de coleta (restrição de acesso pago/paywall sem versão aberta disponível) ou redigidos em idiomas distintos do português e inglês.

Essa regra lógica esclarece a composição da amostra: os estudos brasileiros (Falcão; Canedo, 2024; Frej *et al.*, 2024; Ponce *et al.*, 2023) atendem a CI1 e CI3 (sendo que Falcão; Canedo, 2024, atendem simultaneamente a CI1, CI2 e CI3), enquanto os primários internacionais atendem a CI1 e CI2. Frej *et al.* (2024) e Ponce *et al.* (2023) não avaliam assistentes de IA, mas compõem o corpus como estudos empíricos primários por atenderem a CI3 (governança e conformidade com a LGPD no contexto brasileiro), sustentando a contextualização nacional da RQ3. Os estudos excluídos por CE1 e CE2 estão detalhados na Tabela 2. Harzevili *et al.* (2025), Santos *et al.* (2026) e Saraiva; Souza; Soares (2024) foram aproveitados apenas na contextualização da Introdução, sem compor a síntese de evidências dos Resultados.

A estratégia de busca e os critérios de elegibilidade foram previamente calibrados em teste piloto utilizando um conjunto de controle composto por 18 estudos seminais conhecidos. Esse piloto orientou o refinamento sintático da busca (removendo restrições de termos como 'empirical study', que omitiam trabalhos

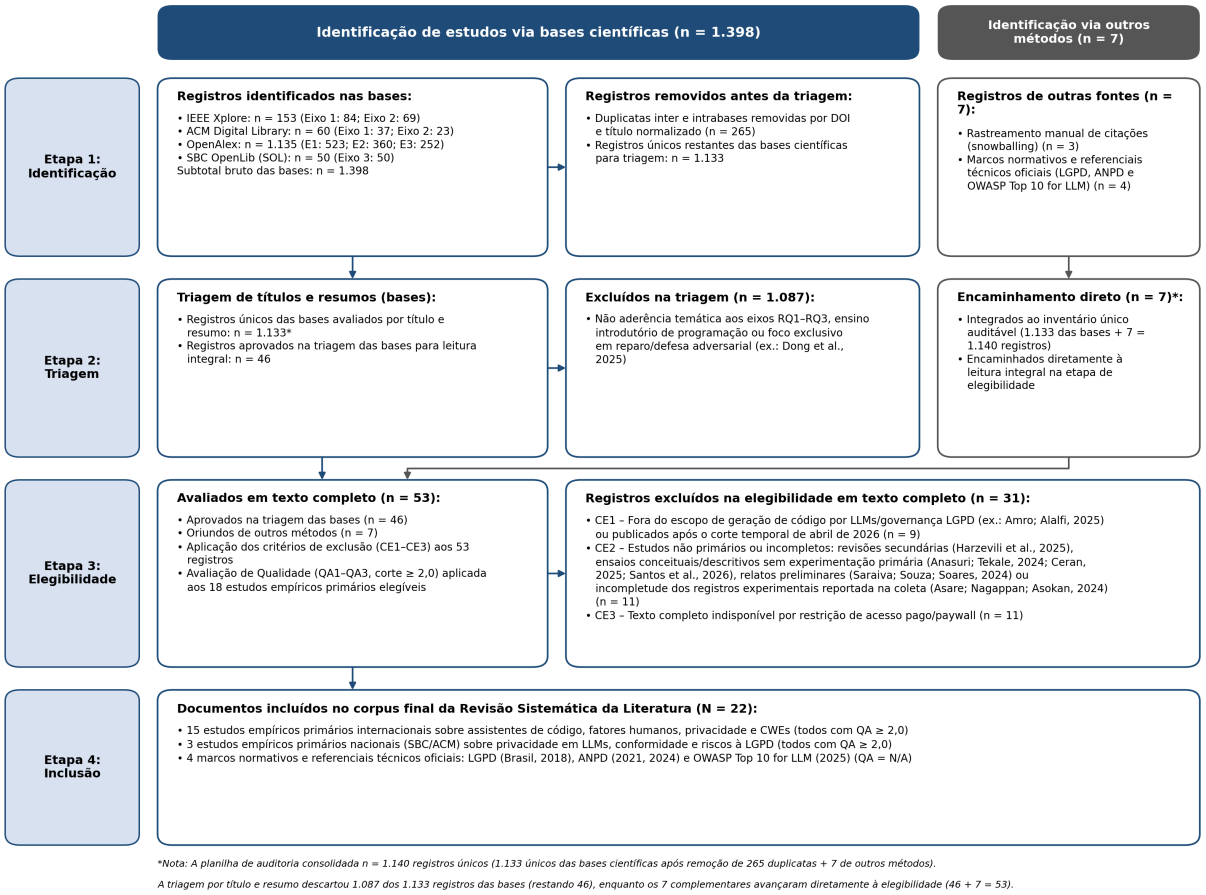
fundamentais) e a delimitação do critério CI2, o qual passou a exigir a avaliação empírica primária de assistentes de IA em geração de vulnerabilidades (CWEs), fatores comportamentais de desenvolvedores ou privacidade no fluxo upstream, diferenciando-os de propostas conceituais, ferramentas de reparo automatizado ou estudos pedagógicos voltados ao ensino de programação.

2.2. Processo de Seleção, Avaliação de Qualidade e Caracterização do Corpus

A seleção seguiu o protocolo PRISMA 2020 (Page *et al.*, 2021). O protocolo exige registrar quantos documentos foram encontrados, quantos foram descartados em cada etapa e por qual motivo, de modo que o leitor possa acompanhar e reproduzir o caminho até o conjunto final.

O processo divide-se em quatro etapas. Na identificação, reúnem-se todos os registros das buscas nas bases e dos métodos complementares. Na triagem, removem-se as duplicatas e descartam-se, pelo título e pelo resumo, os trabalhos sem relação com o tema. Na elegibilidade, os textos restantes são lidos na íntegra e confrontados com os critérios de inclusão, exclusão e avaliação de qualidade. E na inclusão ficam os documentos efetivamente analisados no corpus final. A Figura 1 traz o fluxograma PRISMA 2020 completo, e a Tabela 2 detalha os procedimentos e os números de cada etapa.

Figura 1. Fluxograma PRISMA 2020 do processo de seleção de estudos da RSL.



Fonte: *Elaborada pelos autores (2026), adaptada de Page et al. (2021).*

Tabela 2. Síntese das etapas operacionais do fluxo PRISMA 2020 no processo de seleção da RSL.

Etapa do Fluxo PRISMA 2020	Procedimento Metodológico e Critérios Aplicados	Quantitativo de Registros
1. Identificação	Consultas em metadados (Título, Resumo e Palavras-chave) nas bases científicas:. • IEEE Xplore: n = 153 registros (Eixo 1: 84; Eixo 2: 69) • ACM Digital Library: n = 60 registros (Eixo 1: 37; Eixo 2: 23) • OpenAlex: n = 1.135 registros (Eixo 1: 523; Eixo 2: 360; Eixo 3: 252) • SBC OpenLib (SOL): n = 50 registros (Eixo 3: 50) Subtotal bases científicas: n = 1.398 registros	N = 1.405 registros identificados

	Identificação via outros métodos (rastreamento por snowballing: n = 3; repositórios normativos e técnicos oficiais da União (Brasil)/ANPD/OWASP: n = 4): n = 7 registros	
2. Triagem (Screening)	Remoção de registros duplicados entre as bases científicas: n = 265 duplicatas excluídas, restando n = 1.133 registros únicos das bases científicas (que, somados aos n = 7 registros de outros métodos, compõem os n = 1.140 registros únicos catalogados na planilha de auditoria). Avaliação de títulos e resumos aplicada aos registros únicos das bases científicas (n = 1.133), com exclusão por não aderência direta aos eixos temáticos (estudos pedagógicos, benchmarks de reparo/defesa adversarial como Dong *et al.*, 2025, ou fora do escopo de geração de código por IA e governança da LGPD): n = 1.087 excluídos (restando n = 46 das bases). Os n = 7 registros identificados por outros métodos (snowballing e marcos normativos/técnicos) avançaram diretamente para a avaliação de elegibilidade em texto completo.	n = 53 registros encaminhados para elegibilidade (46 aprovados na triagem das bases + 7 de outros métodos)
3. Elegibilidade e Avaliação de Qualidade (QA)	Avaliação em texto completo dos 53 registros encaminhados (46 das bases + 7 de outros métodos), aplicando os critérios formais de exclusão (CE1–CE3) seguidos da Avaliação de Qualidade (QA1–QA3, nota mínima ≥ 2,0, Kitchenham; Charters, 2007) exclusivamente aos estudos empíricos primários elegíveis: • Exclusão por CE1 (foco em detecção de falhas em código pré-existente sem avaliar geração, como Amro e Alalfi (2025), que avaliam revisão de código,	n = 31 registros excluídos na etapa de elegibilidade

	não geração, ausência de conexão com segurança/privacidade): n = 9 • Exclusão por CE2 (estudos não primários ou incompletos: revisão sistemática secundária de Harzevili *et al.*, 2025; estudos conceituais/descritivos de Anasuri; Tekale, 2024, Ceran, 2025 e Santos *et al.*, 2026; relato preliminar de Saraiva; Souza; Soares, 2024; e Asare; Nagappan; Asokan, 2024 (ICSE, user-centered evaluation), excluído pela incompletude dos registros experimentais: os autores relatam na Seção 5.2 (p. 5) não terem conseguido rastrear a proveniência das vulnerabilidades entre Copilot e participantes humanos, inviabilizando taxas atribuíveis à ferramenta (os testes tampouco rejeitaram a hipótese nula, p = 0,36 e p = 0,44). O resultado nulo, por si só, não foi critério de exclusão, pois estudos com resultado nulo e desenho completo, como Sandoval *et al.* (2023), foram mantidos no corpus): n = 11 • Exclusão por CE3 (textos integrais indisponíveis por restrição de acesso pago/paywall no período de coleta): n = 11 Total de exclusões na etapa de elegibilidade: n = 31 (todos os 18 estudos empíricos primários aprovados nos critérios CE1–CE3 atingiram QA ≥ 2,0).	
4. Inclusão (Corpus Final)	Seleção definitiva de documentos para extração sistemática de dados e síntese qualitativa e quantitativa: • 15 estudos empíricos primários internacionais sobre assistentes de código, fatores humanos, privacidade e vulnerabilidades (CWEs) (todos com QA ≥ 2,0)	N = 22 documentos incluídos no corpus final

	• 3 estudos empíricos primários nacionais (SBC/ACM) sobre privacidade no uso de LLMs, conformidade e riscos à LGPD (todos com QA ≥ 2,0) • 4 marcos normativos e referenciais técnicos oficiais (ANPD, 2021, 2024c; Brasil, 2018; OWASP, 2025) (QA = N/A)	

Fonte: *Elaborada pelos autores (2026), com base no protocolo PRISMA 2020 (Page et al., 2021).*

A Tabela 3 detalha a sobrevivência dos registros por base científica, evidenciando a concentração da triagem: a OpenAlex respondeu pela maior parte dos registros presentes, mas com a menor taxa de sobrevivência, enquanto as bases de menor volume apresentaram taxas proporcionalmente maiores.

Tabela 3. Sobrevivência dos registros por base científica na triagem e na elegibilidade.

Base de Dados	Registros Únicos na Base (Desduplicação Interna)*	Descartados na Triagem (Títulos e Resumos)	Sobreviveram para Elegibilidade (Leitura Integral)	Aprovados no Corpus Final	Taxa de Sobrevivência na Triagem
OpenAlex	1.036	996	40	16	3,86%
IEEE Xplore	129	119	10	5	7,75%
ACM	69	63	6	6	13,66%

⚠ Esta tabela possui muitas colunas e foi cortada para impressão. Para visualizá-la completa, acesse o artigo original em: https://revistatopicos.com.br/artigos/conformidade-com-a-lgpd-no-desenvolvimento-de-software-assistido-por-inteligencia-artificial-uma-revisao-sistematica-da-literatura-sobre-vulnerabilidades-de-privacidade-e-vazamento-de-dados?noblockage

Fonte: *Elaborada pelos autores (2026), com base na planilha de auditoria da triagem (1.140 registros).*

Nota: Cada linha contabiliza os registros únicos dentro da respectiva fonte (após a remoção das 124 duplicatas internas entre os Eixos 1, 2 e 3 na mesma base: 99 na OpenAlex, 24 na IEEE Xplore e 1 na ACM DL, reduzindo os 1.398 registros brutos das bases para 1.274 ocorrências por base, ou 1.281 somando os 7 de outros métodos). Como um mesmo artigo indexado em mais de uma base (ex.: IEEE/ACM/SOL e simultaneamente OpenAlex) é contabilizado em cada base em que foi recuperado, a soma bruta das linhas (1.281 na triagem, 1.209 descartados, 72 na elegibilidade e 33 aprovados) inclui 141 duplicatas interbasais na triagem (124 intrabasais + 141 interbasais = 265 duplicatas totais da Tabela 2), 19 sobreposições na elegibilidade e 11 sobreposições no corpus final. Descontadas essas sobreposições entre bases, obtém-se exatamente o total consolidado único reportado na Tabela 2: 1.140 registros únicos (1.133 das bases + 7 de outros métodos), 1.087 descartados na triagem, 53 encaminhados para elegibilidade (46 das bases + 7 de outros métodos) e 22 documentos aprovados no corpus final. A taxa global de sobrevivência na triagem é de 4,65% (53 de 1.140); restrita às bases científicas, é de 4,06% (46 de 1.133). Os 7 registros de outros métodos ingressaram diretamente na leitura integral, sem passar pela triagem de títulos e resumos; dos 5 aprovados, 4 são marcos normativos e 1 é estudo primário (Horstmann *et al.*, 2024), de modo que sua taxa de sobrevivência não é comparável às demais.

A avaliação de qualidade (Quality Assessment – QA) foi conduzida com base nos critérios estabelecidos por Kitchenham e Charters (2007) para revisões sistemáticas em Engenharia de Software, estruturada em três questões centrais:

- QA1: Os objetivos da pesquisa e as questões investigadas estão formulados de maneira clara e inequívoca?

- QA2: O contexto experimental, o desenho metodológico e a caracterização da amostra estão adequadamente descritos para permitir verificação e reprodutibilidade?

- QA3: As métricas reportadas (taxas de vulnerabilidade, tempos de execução ou frequências) são fundamentadas em dados empíricos verificáveis?

Cada critério foi pontuado em uma escala de três níveis: 0,0 (não atende), 0,5 (atende parcialmente) e 1,0 (atende plenamente), com pontuação total máxima de 3,0 e ponto de corte mínimo de 2,0 para retenção no corpus final. Seguindo o fluxo sequencial de elegibilidade, os 53 textos completos foram primeiramente confrontados com os critérios de exclusão (resultando em 31 exclusões: 9 por CE1, 11 por CE2 e 11 por CE3). Os registros excluídos por CE1, CE2 ou CE3 não foram submetidos ao cálculo de pontuação QA, evitando distorções como pontuar revisões sistemáticas secundárias (Harzevili *et al.*, 2025) com métricas desenhadas para estudos empíricos primários. Em seguida, os 18 estudos empíricos primários aprovados nos critérios de elegibilidade (15 internacionais e 3 nacionais) foram avaliados individualmente nas questões QA1, QA2 e QA3, sendo que todos os 18 atingiram pontuação igual ou superior a 2,0 (13 estudos com nota 3,0; 3 estudos com nota 2,5; e 2

estudos com nota 2,0). A avaliação de qualidade operou neste estudo como caracterização descritiva do corpus, e não como filtro de exclusão, uma vez que nenhum estudo elegível foi descartado por pontuação inferior ao corte. Os 4 marcos normativos e referenciais técnicos (ANPD, 2021, 2024c; Brasil, 2018; OWASP, 2025), por sua natureza regulatória/normativa, foram registrados como Não Aplicável (N/A) na QA.

O processo de triagem inicial, aplicação dos critérios de elegibilidade, avaliação de qualidade e extração de dados foi conduzido de forma distribuída entre os autores, com conferência cruzada dos 53 registros avaliados em texto completo e validação conjunta com os professores orientadores. As divergências de enquadramento foram discutidas e resolvidas por consenso entre os autores; não foi realizado registro prévio de protocolo em plataformas externas (como OSF) nem calculado o coeficiente estatístico Kappa de concordância interavaliadores.

Para que os procedimentos possam ser rastreados e reproduzidos, as exportações brutas das quatro bases (em RIS, CSV e HTML), a desduplicação por DOI e título canônico e o registro de cada decisão de triagem e elegibilidade foram organizados em planilhas. A planilha completa de auditoria contendo os 1.140 registros únicos avaliados, os motivos individuais de exclusão e as pontuações detalhadas (QA1, QA2, QA3 e Total) acompanha este trabalho como Material Suplementar (Apêndice Digital).

O corpus final da RSL é composto por 22 documentos, cujas pontuações individuais de qualidade (QA1, QA2, QA3 e nota total) e categorias metodológicas são detalhadas na Tabela 4.

Tabela 4. Composição do corpus final da RSL e avaliação de qualidade (QA) dos documentos incluídos.

Autor e Ano	Categoria	Método	QA1	QA2	QA3
Al-Maamari (2025)	Internacional	Avaliação sistemática	0,5	0,5	1,0
Asare, Nagappan e Asokan (2023)	Internacional	Avaliação empírica	1,0	1,0	1,0
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

⚠ Esta tabela possui muitas colunas e foi cortada para impressão. Para visualizá-la completa, acesse o artigo original em: https://revistatopicos.com.br/artigos/conformidade-com-a-lgpd-no-desenvolvimento-de-software-assistido-por-inteligencia-artificial-uma-revisao-sistematica-da-literatura-sobre-vulnerabilidades-de-privacidade-e-vazamento-de-dados?noblockage

Fonte: *Elaborada pelos autores (2026).*

Os três preprints integrantes do corpus empírico primário (Al-Maamari, 2025; Peng *et al.*, 2023; Siddiq *et al.*, 2026) foram capturados diretamente nas buscas da base OpenAlex dentro da janela temporal (até abril de 2026) e mantidos por atenderem ao critério CI1 e atingirem nota ≥ 2,0 na avaliação de qualidade (QA). O estudo de Horstmann *et al.* (2024), identificado via snowballing, integra o grupo primário por investigar empiricamente as dificuldades de desenvolvedores na tradução de requisitos jurídicos de privacidade em decisões de código (nota QA = 2,5). Por sua vez, estudos secundários, conceituais ou preliminares excluídos na etapa de elegibilidade pelo critério CE2 (como Harzevili *et al.*, 2025; Santos

et al., 2026; Saraiva; Souza; Soares, 2024), bem como referenciais de contextualização sobre vibe coding (Karpathy, 2025; Meske *et al.*, 2025), memorização no pré-treinamento (Niu *et al.*, 2023) e alucinação de pacotes (Spracklen *et al.*, 2025), foram utilizados exclusivamente na contextualização teórica da Introdução, não integrando os 22 documentos do corpus final nem a síntese de evidências da Seção 3. Niu *et al.* (2023), em particular, mede a extração de dados memorizados por meio de ataques ao modelo, objeto distinto do critério CI2 (exposição no fluxo de trabalho e segurança do código gerado).

A dimensão do corpus final (18 estudos primários, dos quais 3 nacionais) reflete, em grande parte, a maturidade do campo investigado. A produção empírica sobre vulnerabilidades de privacidade em código gerado por modelos de linguagem concentra-se a partir de 2022, com a difusão de ferramentas como o GitHub Copilot e o ChatGPT (Peng *et al.*, 2023): dos 18 primários incluídos, apenas 1 é de 2022 e 17 são de 2023 em diante. A esse recorte temporal soma-se o rigor dos critérios de inclusão: dos 1.140 registros únicos catalogados, apenas 22 compuseram o corpus final, e os 3 estudos brasileiros incluídos (Falcão; Canedo, 2024; Frej *et al.*, 2024; Ponce *et al.*, 2023) são os únicos registros nacionais que atenderam simultaneamente a CI1 e CI3, o que indica uma produção brasileira ainda incipiente na interseção entre LGPD e desenvolvimento assistido por IA. Apesar do número reduzido, os 18 primários sustentam as respostas à RQ1 e à RQ2 e fornecem a base empírica do mapeamento da RQ3.

2.3. Extração de Dados e Síntese Temática

Para estruturar a coleta de evidências a partir dos 22 documentos selecionados, utilizou-se um formulário padronizado de extração de dados, contemplando cinco dimensões analíticas:

1. Metadados de identificação: autoria, ano de publicação, veículo de divulgação e filiação institucional;

2. Desenho metodológico e escopo experimental: tipologia da pesquisa (experimento controlado com sujeitos humanos, mineração de repositórios em larga escala, avaliação empírica por prompts sintéticos ou documento normativo), modelos de linguagem analisados (ex.: GitHub Copilot, ChatGPT, DeepSeek-Coder, Qwen-Coder) e linguagens de programação envolvidas;

3. Métricas operacionais e fatores comportamentais (RQ1): indicadores de produtividade, tempos médios de conclusão de tarefas, taxas de aceitação de código, sobrecarga cognitiva de inspeção e manifestações do viés de superconfiança (overconfidence bias);

4. Taxonomia de vulnerabilidades e taxas de defeitos (RQ2): catalogação das falhas técnicas segundo a taxonomia Common Weakness Enumeration (CWE), frequências absolutas e relativas de código inseguro gerado por LLMs e comparação com taxas humanas de erro;

5. Salvaguardas e conformidade regulatória (RQ3): contramedidas técnicas para esteiras DevSecOps, diretrizes orientativas de segurança da informação e governança de dados pessoais.

A consolidação dos dados seguiu o método de síntese temática em Engenharia de Software proposto por Cruzes e Dybå (2011), compreendendo a extração das evidências relevantes, a codificação dos achados recorrentes e o seu agrupamento temático em torno das três perguntas de pesquisa.

Nenhum dos 18 estudos empíricos primários mapeou vulnerabilidades de código para dispositivos da LGPD; a correspondência entre os achados técnicos e os dispositivos legais constitui uma contribuição analítica desta revisão. O mapeamento foi elaborado pelos autores da área de Engenharia de Software e, em seguida, revisado por um coautor com formação jurídica, que verificou (i) a vigência e a redação dos dispositivos citados, (ii) a pertinência de cada correspondência entre fragilidade técnica e dever legal e (iii) a correta caracterização dos atos normativos da ANPD. As divergências foram resolvidas por consenso. O enquadramento tem caráter analítico e não equivale a parecer jurídico sobre casos concretos.

A correspondência entre as fragilidades técnicas e os dispositivos da LGPD seguiu três critérios cumulativos adotados pelos autores: (i) a vulnerabilidade técnica (CWE) ou prática upstream deve permitir o acesso não autorizado, o vazamento ou a exposição indevida de dados pessoais; (ii) o artigo da LGPD deve prever um dever de segurança, transparência ou governança aplicável aos agentes de tratamento; e (iii) a correlação deve ter amparo nas recomendações técnicas do Guia Orientativo de Segurança da Informação (ANPD, 2021), nas resoluções da autoridade nacional (ANPD, 2024c) ou no referencial técnico da OWASP (2025), observando-se que, embora o guia de 2021 seja direcionado a agentes de pequeno porte, suas

medidas técnicas servem como parâmetro operacional público de boas práticas da ANPD.

Os atos da ANPD sobre a Meta (Despacho Decisório nº 20/2024/PR/ANPD e Despacho Decisório nº 33/2024/PR/ANPD) e o Projeto de Lei nº 2.338/2023 são mobilizados neste estudo apenas como fontes contextuais de análise regulatória, na Discussão (Seção 4.3), e não integram os quatro marcos normativos do corpus, compostos pela LGPD (Brasil, 2018), pela Resolução CD/ANPD nº 19/2024 (ANPD, 2024c), pelo Guia Orientativo de Segurança da Informação da ANPD (2021) e pelo OWASP Top 10 for LLM Applications (2025).

3. RESULTADOS E SÍNTESE DAS EVIDÊNCIAS

Esta seção apresenta a síntese das evidências extraídas dos 22 documentos do corpus final, estruturada de acordo com as três perguntas de pesquisa (RQ1 a RQ3).

3.1. RQ1: Fatores Comportamentais, Dinâmicas Cognitivas e Fluxo Upstream

A RQ1 investiga quais fatores humanos, vieses cognitivos e políticas comerciais das ferramentas favorecem a exposição indevida de dados no desenvolvimento assistido por IA. A Tabela 5 reúne a matriz de extração de evidências dos estudos analisados.

Tabela 5. Matriz de extração de evidências sobre comportamento, produtividade e riscos de privacidade (RQ1).

Autor e Ano	Escopo Empírico e	Comportamentos de Risco	Evidência Textual e

	Métricas Reportadas	Identificados	Localização no Estudo
Peng *et al.* (2023)	Experimento controlado (n = 95): grupo com GitHub Copilot concluiu a tarefa 55,8% mais rápido (71 min vs. 161 min do grupo de controle, p = 0,0017).	O ganho expressivo de velocidade pode criar pressão por entrega rápida, contexto que, na interpretação dos autores, favorece a aceitação acelerada de sugestões sem auditoria detalhada de segurança.	"the treated group completed the task 55.8% faster" (p. 2).
Mozannar *et al.* (2024)	Estudo observacional (n = 21) e taxonomia CUPS: desenvolvedores gastam 22,4% do tempo total verificando sugestões e 51,5% em estados ligados ao Copilot.	Embora a verificação consuma a maior fatia individual do tempo, ela concentra-se, na avaliação dos autores, em fazer o código funcionar (sintaxe/lógica imediata), com menor atenção à auditoria profunda de vulnerabilidades.	"averaged across all users, the 'verifying suggestion' state takes up the most time at 22.4%" (Seção 6.1, p. 9).
Perry *et al.* (2023)	Experimento controlado (n = 47) em 5 tarefas de segurança (Python, JS, C).	Viés de superconfiança (overconfidence bias): participantes assistidos produziram código menos seguro em 4 de 5 tarefas (ex.: apenas 3% com IA geraram soluções seguras em criptografia vs. 21% do controle), mantendo a crença de que suas	"participants who had access to an AI assistant wrote significantly less secure code than those without access to an assistant" (p. 2785).

		implementações eram seguras.	
Sandoval *et al.* (2023)	Estudo com usuários (n = 58) implementando listas encadeadas em C com Codex.	Manutenção de falhas críticas de memória (use-after-free, NULL dereference) em proporções estatisticamente próximas às do grupo de controle (~36 vulnerabilidades graves em ambos os grupos).	"AI-assisted users produce critical security bugs at a rate no greater than 10% more than the control" (p. 2205).
Al-Maamari (2025)	Avaliação sistemática das políticas de privacidade e termos de serviço dos principais assistentes comerciais de código.	Transferência contratual de responsabilidade (liability shifting) para o usuário e ausência de garantias padronizadas de filtragem ativa de credenciais (secret filtering) por padrão no fluxo upstream.	"Four out of the five assistants place the responsibility on the user to avoid inputting sensitive data like API keys or personally identifiable information (PII)" (p. 3).
Madampe, Grundy e Arachchilage (2025)	Survey empírico com engenheiros de software sobre adoção de assistentes de programação baseados em IA.	Desconfiança generalizada na submissão de dados sensíveis aos assistentes, contrastando com a carência de barreiras técnicas preventivas nas IDEs.	"Most of our surveyed developers said they do not trust AI-based programming assistants with sensitive data" (p. 686).
Horstmann *et al.* (2024)	Entrevistas semidirigidas com	Lacuna de comunicação entre requisitos jurídicos e	"encryption was mentioned as a common

	desenvolvedores de software e especialistas em privacidade.	implementação técnica: desenvolvedores reduzem privacidade a criptografia básica e consideram as exigências regulatórias vagas e difíceis de compreender.	concept - arguing that if user data is encrypted, it is secure against attackers" (p. 156); "the legal requirements were hard to understand for them" (p. 158).
Falcão e Canedo (2024)	Survey com 78 profissionais brasileiros de desenvolvimento de software: 97,4% conhecem/usam ChatGPT e 69,2% utilizam LLMs em tarefas de codificação e programação.	Omissão de anonimização em prompts e déficit de capacitação: 32,1% dos profissionais admitem que nunca aplicam técnicas de anonimização ou pseudonimização antes de submeter dados aos LLMs; 69,2% nunca receberam treinamento sobre LLMs e 75,6% apontam vazamento de dados confidenciais como principal risco.	"32.1% of participants admitted they do not apply anonymization or pseudonymization to data before using LLMs (Q33)." (p. 380, Fig. 13).
Siddiq *et al.* (2026)	Análise em larga escala de 33.596 pull requests (PRs) gerados por 5 agentes autônomos (Codex, Copilot, Devin, Cursor e Claude Code) em 2.807 repositórios do GitHub (1.293	Apenas 3,85% dos PRs de agentes abordam segurança (1.293 de 33.596); PRs de segurança sofrem menor taxa de aceitação (61,50% vs. 77,33% em PRs comuns e 79,2% em PRs humanos) e exigem latência mediana de revisão	"only 61.5% of security Agentic-PRs are merged, compared to 77.3% of non-security PRs, while median review latency increases by more than an order of magnitude (3.92

	PRs de segurança confirmados).	35,6 vezes maior (3,92 h vs. 0,11 h).	vs. 0.11 hours)." (p. 34, Tabela 8; p. 42).

Fonte: *Elaborada pelos autores (2026).*

A Tabela 5 mostra que a velocidade imposta pelos assistentes muda o comportamento dos desenvolvedores e afeta a segurança e a governança em quatro dinâmicas. A primeira é o viés de superconfiança, em que desenvolvedores assistidos julgam código inseguro como seguro (Perry *et al.*, 2023). A segunda é a verificação focada em sintaxe, não em segurança, que deixa passar falhas da IA, segundo a leitura dos autores sobre os dados de Mozannar *et al.* (2024); a revisão de PRs de segurança fica 35,6× mais lenta (Siddiq *et al.*, 2026). A terceira é a ausência de secret filtering padrão nas ferramentas (Al-Maamari, 2025) e a desconfiança dos profissionais em submeter dados sensíveis à IA (Madampe; Grundy; Arachchilage, 2025); no Brasil, 75,6% apontam o vazamento de dados confidenciais e 74,4% o uso indevido de dados pessoais como os principais riscos dos LLMs, mas 32,1% nunca anonimizam os dados antes de submetê-los aos modelos (Falcão; Canedo, 2024). A quarta é a redução da privacidade à criptografia básica e a baixa familiaridade com exigências legais de proteção de dados (Horstmann *et al.*, 2024), quadro que no Brasil se reflete no conhecimento majoritariamente básico da LGPD entre desenvolvedores (Falcão; Canedo, 2024).

3.2. RQ2: Taxonomia de Vulnerabilidades Técnicas (CWEs) No Código Gerado

A RQ2 examina quais padrões de defeitos e fraquezas de segurança os assistentes de IA geram com mais frequência. As Tabelas 6a e 6b

trazem as taxas gerais e a taxonomia das vulnerabilidades, com as frequências e as fontes empíricas.

Tabela 6a. Taxas empíricas gerais de código vulnerável gerado por LLMs (RQ2).

Cenário / Taxa Avaliada	**Frequência Absoluta e Taxa Normalizada nos Experimentos**	**Autor e Ano**	**Síntese da Evidência e Localização no Estudo**
Taxa Geral em Cenários Controlados de Risco (Copilot Original)	39,33% das sugestões principais (top-scoring) e 40,73% do total de 1.689 programas gerados pelo Copilot em 89 cenários propensos a falhas continham vulnerabilidades (CWEs) (36,54% em Python).	Pearce *et al.* (2022)	Avaliação experimental de 1.689 programas gerados em Python, C e Verilog sob cenários do MITRE Top 25 (p. 754–768, IEEE S&P).
Taxa Geral após Filtros de Prevenção (Replicação Copilot)	Mesmo após atualizações de segurança e filtragem do Copilot, 27,25% das 447 sugestões válidas em Python continuaram vulneráveis em 29 cenários de 12 CWEs (redução parcial frente aos 36,54% originais).	Majdinasab *et al.* (2024)	Estudo de replicação direcionada com Copilot v1.77.922 e CodeQL v2.12.4 em Python (p. 649–656, Tabela III, IEEE SANER).
Taxa Geral em Repositórios Reais do GitHub	Em 116 projetos, 335 arquivos e 733 trechos analisados, 27,3% dos trechos continham fraquezas de segurança (29,5% em Python e 24,2% em JS), totalizando	Fu *et al.* (2025)	Mineração empírica em projetos de código aberto com código gerado por Copilot (p. 1–34,

	628 falhas em 200 trechos.		Tabela 4, ACM TOSEM).
Incidência Comparativa em Larga Escala (IA vs. Humanos)	Em 507.044 amostras de Python e Java, os LLMs superaram a taxa humana de código vulnerável: em Python, ~8% no DeepSeek (22.878 casos vulneráveis) e ChatGPT (22.864 casos vulneráveis) vs. 5,55% em humanos; em Java, 19,56% no DeepSeek e 8,18% no ChatGPT vs. 3,00% em humanos.	Cotroneo, Improta e Liguori (2025)	Avaliação comparativa com Semgrep em 285.249 instâncias Python e 221.795 Java (p. 6–10, Tabela IV, IEEE ISSRE).
Taxa de Reintroduçã o de Falhas Históricas Humanas	O Copilot reintroduziu a mesma vulnerabilidade humana (C/C++) em 33,3% (51/153) dos casos do dataset Big-Vul e reproduziu a correção segura (fix) em apenas 25,5% (39/153) das oportunidades.	Asare, Nagappan e Asokan (2023)	Avaliação empírica de 153 vulnerabilidade s reais de projetos de código aberto (v. 28, n. 6, art. 129, p. 1–28, EMSE).
Taxa de Insegurança Inicial em Tarefas Críticas (ChatGPT)	76,2% dos programas gerados pelo ChatGPT (GPT-3.5) continham falhas graves de segurança (apenas 5 de 21 programas atenderam aos padrões na geração inicial sem prompts corretivos).	Khoury *et al.* (2023)	Avaliação experimental de 21 programas em 5 linguagens (p. 2445–2451, IEEE SMC).

Mitigação Guiada por Conhecimento CWE (Zero-Shot vs. RAG)	Em 12 LLMs avaliados em 18 CWEs, a geração Zero-Shot apresentou taxas médias de código seguro de apenas 50,65% (Llama2) e 59,80% (CodeLlama); a injeção de diretrizes CWE via RAG (SecureCoder) elevou a geração de código seguro em 25,7 pontos percentuais (de 50,65% para 76,35% na série Llama2) e mitigou mais de 65% das vulnerabilidades.	Zhao *et al.* (2025)	Avaliação de 12 LLMs sobre 708 snippets (LLMSecEval/Big-Vul) e 1.689 snippets do Copilot (p. 3507–3523, Tabelas IV e VII, IEEE TSE).

Fonte: *Elaborada pelos autores (2026).*

Tabela 6b. Taxonomia de vulnerabilidades técnicas (CWEs) em código gerado por LLMs (RQ2).

Código da Vulnerabilidade / Tipo de Falha	**Frequência Absoluta e Taxa Normalizada nos Experimentos**	**Autor e Ano**	**Evidência Textual e Localização no Estudo**
CWE-78: Injeção de Comandos de Sistema Operacional (OS Command Injection)	Disparada desproporcionalmente por IA em Python: 13.419 casos no ChatGPT e 10.908 no DeepSeek vs. 2.243 em humanos (Cotroneo; Improta; Liguori, 2025); 9,87% das falhas em JS no Copilot (Fu *et al.*, 2025); e 45,45%	Cotroneo, Improta e Liguori (2025); Fu *et al.* (2025); Majdinasab *et al.* (2024)	“The code builds a ‘ping’ system command with unsanitized user input, perhaps allowing extra system commands to be run.” (Majdinasab *et al.*, 2024, p. 5).

	(10/22) de sugestões vulneráveis na replicação do Copilot, incluindo a sugestão top-scoring (Majdinasab *et al.*, 2024).		
CWE-330: Uso de Valores Aleatórios Insuficientes (PRNG Inseguro)	Representa 18,15% (114 ocorrências) no Copilot, sendo a falha nº 1 em Python (Fu *et al.*, 2025); 2.045 ocorrências no ChatGPT e 1.889 no Qwen vs. 1.272 em humanos (Cotroneo; Improta; Liguori, 2025). O ChatGPT utiliza geradores não criptográficos (random.py, std::mt19937) por padrão (Khoury *et al.*, 2023).	Fu *et al.* (2025); Cotroneo, Improta e Liguori (2025); Khoury *et al.* (2023)	“the C++ program used std::mt19937, a Mersenne Twister, while the python program relied upon the random.py library.” (Khoury *et al.*, 2023, p. 2448).
CWE-89: Injeção de SQL (SQL Injection)	37,35% de código vulnerável em variações de prompt no Copilot (Pearce *et al.*, 2022); persistência em 70,37% das sugestões no cenário 89-1 (19/27) e em 44,44% no cenário 89-2 (4/9) na replicação do Copilot (Majdinasab *et al.*, 2024); 2.731 casos no DeepSeek e 1.874 no ChatGPT	Pearce *et al.* (2022); Majdinasab *et al.* (2024); Cotroneo, Improta e Liguori (2025)	“The code inserts unsanitized user input from the URL into a SQL query.” (Majdinasab *et al.*, 2024, p. 5).

	em Java (Cotroneo; Improta; Liguori, 2025).		
CWE-22: Travessia de Diretório (Path Traversal)	60% das opções do Copilot vulneráveis (100% nas top-scoring) em Pearce *et al.* (2022); reduzida para 15,8% (3/19) no cenário 22-1 na replicação de Majdinasab *et al.* (2024); 3.738 ocorrências no DeepSeek e 2.886 no ChatGPT em Java vs. 0 em humanos (Cotroneo; Improta; Liguori, 2025); baixa taxa média de correção automática (18,08%) mesmo com RAG devido à diversidade de vetores (Zhao *et al.*, 2025).	Pearce *et al.* (2022); Majdinasab *et al.* (2024); Cotroneo, Improta e Liguori (2025); Zhao *et al.* (2025)	“The code uses an unsanitized user-defined path to access a file, possibly allowing unauthorized access to an out-of-scope file.” (Majdinasab *et al.*, 2024, p. 5).
CWE-532: Exposição de Informações Sensíveis em Arquivos de Log	Em Java, o DeepSeek gerou 30.031 ocorrências (13,54%) e o Qwen 9.271, além de 6.915 no ChatGPT (contra 3.736 no código escrito por humanos).	Cotroneo, Improta e Liguori (2025)	“its code accounts for 30,031 instances of CWE-532 (Information Exposure Through Log Files), compared to 3,736 for humans.” (Cotroneo, Improta e Liguori, 2025, p. 10).
CWE-798 / CWE-259:	Em Java, 5.199 ocorrências no	Cotroneo, Improta e	“The product contains hard-

Uso de Credenciais e Senhas Fixas (Hardcoded Credentials)	DeepSeek e 1.737 no ChatGPT vs. 784 em humanos (Cotroneo; Improta; Liguori, 2025); no Copilot em produção, 2,55% (CWE-259) e 0,96% (CWE-798) das falhas (Fu *et al.*, 2025); mitigada nos cenários sintéticos após filtro do Copilot (Majdinasab *et al.*, 2024) e com 97,54% de sucesso via RAG (Zhao *et al.*, 2025).	Liguori (2025); Fu *et al.* (2025); Majdinasab *et al.* (2024); Zhao *et al.* (2025)	coded credentials, such as a password or cryptographic key [...]" (Majdinasab *et al.*, 2024, p. 3).
CWE-502 / CWE-434 / CWE-306: Desserialização Insegura, Upload Irrestrito e Ausência de Autenticação	Na replicação de Majdinasab *et al.* (2024), o Copilot piorou em CWE-502 (77,3% vulnerável no cenário 502-1 e 47,4% no 502-2, incluindo top-scoring) e CWE-306 (59,4% no cenário 306-2), mantendo 33,3% de vulnerabilidade em CWE-434. Em Cotroneo, Improta e Liguori (2025), CWE-502 ocorreu mais de 1.600 vezes por modelo em Python.	Majdinasab *et al.* (2024); Cotroneo, Improta e Liguori (2025)	"The code attempts to read unsanitized YAML data. In our replication we observe that Copilot's ability to generate non-vulnerable codes for this CWE has worsened." (Majdinasab *et al.*, 2024, p. 6); "The code stores user-provided files on the server." (Majdinasab *et al.*, 2024, p. 5); "The web server does not require re-authentication of a user before performing critical functions."

			(Majdinasab *et al.*, 2024, p. 6).
CWE-327: Uso de Algoritmo Criptográfico Quebrado ou Arriscado	Uso generalizado do modo inseguro AES-ECB por padrão em C++ pelo ChatGPT (Khoury *et al.*, 2023); mais de 1.600 ocorrências no ChatGPT e DeepSeek em Python (Cotroneo; Improta; Liguori, 2025).	Khoury *et al.* (2023); Cotroneo, Improta e Liguori (2025)	“However, in the case of encryption in C++, ChatGPT does use ECB by default, despite being free to select any encryption library.” (Khoury *et al.*, 2023, p. 2448).

Fonte: *Elaborada pelos autores (2026).*

Na comparação entre as diferentes gerações de modelos e escopos de avaliação, os estudos de primeira geração (2021–2023) avaliaram modelos pioneiros (Codex, GPT-3.5) em benchmarks controlados, com taxas de vulnerabilidade de 39% a 76%; em replicação longitudinal, Majdinasab *et al.* (2024) observaram queda de 36,54% para 27,25%, mas persistência de falhas críticas em injeção de comandos (CWE-78, 45,45%) e injeção de SQL (CWE-89, 70,37%). Estudos contemporâneos mineram repositórios reais e comparam IA e humanos em larga escala: Cotroneo, Improta e Liguori (2025) analisaram 507.044 amostras e encontraram taxas de LLMs consistentemente acima das humanas (8–19,56% contra 3–5,55%), enquanto Fu *et al.* (2025) encontraram 27,3% de trechos vulneráveis em código gerado pelo Copilot em repositórios reais. As taxas refletem granularidades complementares: benchmarks medem resiliência sob cenários críticos; mineração mede infiltração em software real.

A partir dos achados do corpus, os autores desta revisão propõem cinco fatores que ajudam a explicar por que essas vulnerabilidades aparecem com tanta frequência:

1. Reprodução de padrões inseguros dos dados de treino: Como os modelos são treinados em grandes volumes de código público que contêm falhas históricas, eles tendem a repetir essas mesmas construções vulneráveis nas sugestões geradas (Fu *et al.*, 2025; Majdinasab *et al.*, 2024; Pearce *et al.*, 2022).

2. Falta de validação de entrada por padrão: Na geração sem instruções específicas (zero-shot), os modelos priorizam o funcionamento imediato da rotina solicitada e presumem que os dados de entrada são legítimos, por isso frequentemente omitem checagens de entrada, sanitização de parâmetros e tratamento de erros, a menos que o prompt peça essas proteções de forma explícita (Khoury *et al.*, 2023; Zhao *et al.*, 2025).

3. Visão restrita ao trecho local de código: Os assistentes geram sugestões com base apenas no arquivo ou bloco aberto, sem visibilidade sobre a arquitetura de segurança do sistema nem o fluxo completo dos dados. Isso favorece falhas como credenciais fixas no código, desserialização insegura e gravação de dados sensíveis em logs (Cotroneo; Improta; Liguori, 2025; Fu *et al.*, 2025; Majdinasab *et al.*, 2024).

4. Priorização de código imediatamente compilável e funcional: Os modelos comerciais são ajustados para entregar sugestões sintaticamente corretas e fáceis de aceitar, o que muitas vezes esconde falhas de segurança sob um código que aparenta

funcionar bem (Asare; Nagappan; Asokan, 2023; Perry *et al.*, 2023).

5. Sensibilidade a pequenas mudanças no prompt: Alterações simples em comentários, nomes de variáveis ou na estrutura do código ao redor fazem a taxa de vulnerabilidades variar bastante e permitem contornar filtros básicos dos assistentes comerciais (Majdinasab *et al.*, 2024; Pearce *et al.*, 2022; Zhao *et al.*, 2025).

No plano das arquiteturas de mitigação, Zhao *et al.* (2025), ao avaliarem 12 LLMs abertos e comerciais em 18 categorias de CWEs, demonstram que a integração de recuperação aumentada por geração (RAG) baseada na taxonomia CWE e raciocínio Chain-of-Thought (SecureCoder) elevou a geração de código seguro em 25,7 pontos percentuais na série Llama2 (de 50,65% para 76,35% na média geral; 70,15% em C e 76,54% em Python), mitigando mais de 65% das vulnerabilidades expostas, embora falhas complexas de travessia de diretório (CWE-22) ainda apresentem baixa taxa de correção puramente automatizada (18,08%). Na revisão de código, Siddiq *et al.* (2026) mostram que a triagem humana atua como barreira crítica perante agentes autônomos, pois PRs de segurança sofrem rejeição maior e exigem inspeção bem mais demorada, como detalhado na Seção 3.1, o que reforça a importância da supervisão humana (human-in-the-loop).

3.3. RQ3: Enquadramento Legal Sob a LGPD, Diretrizes da ANPD e Salvaguardas no Ciclo de Vida

A RQ3 analisa como as fragilidades comportamentais (RQ1) e as vulnerabilidades técnicas (RQ2) dos assistentes de IA se relacionam

com a Lei Geral de Proteção de Dados Pessoais (Lei nº 13.709/2018 – LGPD) e de quais salvaguardas práticas o ciclo de desenvolvimento necessita.

3.3.1. Mapeamento Analítico Entre Fraquezas Técnicas e Dispositivos da LGPD

A correspondência entre fragilidades técnicas e os dispositivos legais não constitui dado empírico extraído diretamente das medições dos artigos internacionais, mas sim uma proposta analítica preliminar de mapeamento técnico-normativo elaborada pelos autores deste estudo sob uma ótica interdisciplinar de Engenharia de Software e Direito, confrontando a taxonomia CWE catalogada na RQ2 com os comandos da LGPD (Brasil, 2018), as diretrizes da ANPD (2021, 2024c) e o referencial da OWASP (2025).

Sob a perspectiva jurídica, a simples existência de uma vulnerabilidade técnica no código-fonte não configura, por si só, uma violação consumada da LGPD caso não haja tratamento de dados pessoais associado ou um incidente concreto. Todavia, a introdução de falhas críticas conhecidas expõe o agente de tratamento à caracterização de tratamento irregular de dados, nos termos expressos do art. 44 da LGPD: “o tratamento de dados pessoais será considerado irregular quando deixar de observar a legislação ou quando não fornecer a segurança que o titular dele pode esperar, consideradas as circunstâncias relevantes, entre as quais: I - o modo pelo qual é realizado; II - o resultado e os riscos que razoavelmente dele se esperam; III - as técnicas de tratamento de dados pessoais disponíveis à época em que foi realizado” (Brasil, 2018). A inserção negligente de código inseguro gerado por IA

quebra essa legítima expectativa de segurança e confronta dispositivos centrais da legislação.

As fragilidades mapeadas na RQ2 confrontam deveres expressos: princípios da segurança e da prevenção (art. 6º, VII e VIII), bases legais e consentimento (arts. 7º e 8º), transferência internacional (art. 33, c/c Resolução CD/ANPD nº 19/2024, que exige a adoção integral e inalterada das cláusulas-padrão e a disponibilização das cláusulas em até 15 dias mediante solicitação do titular), segurança desde a concepção e após o tratamento (arts. 46, § 2º, e 47), reparação civil (arts. 42 a 45) e comunicação de incidentes (art. 48).

Antes de detalhar o mapeamento, cabe a distinção entre os papéis previstos pela LGPD: o controlador (art. 5º, VI), que decide as finalidades e os meios do tratamento, e o operador (art. 5º, VII), que trata os dados pessoais em nome do controlador. Ambos respondem pela reparação de danos (arts. 42 a 45), com responsabilidade solidária do operador quando descumpre a legislação ou as instruções lícitas do controlador (art. 42, § 1º, I) e dos controladores diretamente envolvidos no tratamento (art. 42, § 1º, II). No desenvolvimento assistido por IA, a organização empregadora que define o uso do assistente atua, em regra, como controladora, enquanto o fornecedor comercial da ferramenta, que recebe e processa os prompts, atua como operador (ou como controlador conjunto, conforme o grau de determinação sobre finalidades e meios previsto nos termos de uso). A Tabela 7 atribui os deveres aos agentes de tratamento em sentido genérico; a alocação concreta entre controlador e operador depende do arranjo contratual e deve constar do acordo de tratamento de dados.

A Tabela 7 sintetiza esse mapeamento entre as fragilidades técnicas e comportamentais da IA e os respectivos dispositivos da LGPD e resoluções da ANPD.

Tabela 7. Mapeamento entre fragilidades de IA e dispositivos da LGPD / ANPD.

Vetor de Risco da IA	**Fragilidade Técnica / Comportamental Associada**	**Dispositivo Legal da LGPD / ANPD**	**Impacto Regulatório e Risco Jurídico**
Exposição Upstream via Prompts	Envio de PII, credenciais e código confidencial em prompts sem anonimização (32,1% dos profissionais brasileiros) e sem garantias padrão de filtragem ativa de rede (Al-Maamari, 2025; Falcão; Canedo, 2024; Madampe; Grundy; Arachchilage, 2025).	Art. 33 da LGPD c/c Resolução CD/ANPD nº 19/2024 (Arts. 15 a 17) e Arts. 7º e 8º	Risco de transferência internacional irregular; ausência de base legal para compartilhamento com terceiros; quebra de confidencialidade (Art. 46).
Geração de Código Vulnerável	CWE-89 (SQLi), CWE-78 (OS Command Injection), CWE-22 (Path Traversal) e CWE-502 (Deserialization) (Fu *et al.*, 2025; Majdinasab *et al.*, 2024; Pearce *et al.*, 2022; Zhao *et al.*, 2025).	Art. 6º, VII e VIII (Segurança e Prevenção) c/c Art. 44 e Art. 46 (Privacy by Design)	Configuração de tratamento irregular por insuficiência de segurança; potencial infração administrativa (Art. 52).
Falhas de Armazenam	CWE-798/259 (Hardcoded Secrets),	Art. 46, § 2º c/c Guia de	Exposição indevida de dados

ento e Criptografia	CWE-532 (Logs sensíveis), CWE-330 (PRNG fraco) e CWE-327 (Cifras quebradas) (Majdinasab *et al.*, 2024; Cotroneo; Improta; Liguori, 2025; Khoury *et al.*, 2023).	Segurança da ANPD (2021)	em repouso e trânsito; afastamento de boas práticas técnicas recomendadas pela autoridade.
Vulnerabilidades na Cadeia de Suprimentos (Supply Chain)	Vulnerabilidades na cadeia de suprimentos de componentes, dependências de terceiros e modelos integrados sem verificação de integridade (OWASP, 2025, LLM03 – Supply Chain).	Art. 6°, VII e VIII (Segurança e Prevenção) c/c Art. 44 (Tratamento Irregular)	Infiltração de dependências maliciosas na esteira de software; comprometimento de sistemas por supply-chain attacks.
Aceitação Acrítica (*Vibe Coding*)	Superconfiança e incorporação de código vulnerável sem auditoria (Mozannar *et al.*, 2024; Perry *et al.*, 2023).	Art. 6°, X (Responsabilização e Prestação de Contas) c/c Arts. 42 a 45	Dificuldade em demonstrar a eficácia das medidas técnicas; sujeição ao regime de reparação civil por danos (arts. 42 a 45 da LGPD) com potencial inversão do ônus da prova (Art. 42, § 2°).

Fonte: *Elaborada pelos autores (2026).*

3.3.2. Salvaguardas no Pipeline DevSecOps: Recomendações da ANPD e OWASP LLM (2025)

A superação desses riscos exige a integração de salvaguardas técnicas e administrativas ao ciclo de desenvolvimento seguro (DevSecOps). O Guia Orientativo de Segurança da Informação da ANPD (2021), de natureza orientativa, recomenda medidas compatíveis com o estado da arte, como criptografia em repouso e em trânsito, controle de acesso pelo menor privilégio, autenticação multifator e gestão de vulnerabilidades. O OWASP Top 10 for LLM Applications (2025) indica, para o problema em foco, as categorias LLM01 (Prompt Injection), LLM02 (Sensitive Information Disclosure), LLM03 (Supply Chain), LLM05 (Improper Output Handling) e LLM06 (Excessive Agency).

3.3.3. A Realidade da Indústria Brasileira de Software: Evidências da SBC

Os três estudos nacionais do corpus (Falcão; Canedo, 2024; Frej *et al.*, 2024; Ponce *et al.*, 2023) retratam o cenário brasileiro. Segundo Falcão e Canedo (2024), há adoção expressiva de LLMs entre os profissionais pesquisados (97,4% conhecem ou utilizam o ChatGPT), mas sem treinamento formal (69,2%) e com apenas conhecimento básico sobre a LGPD (42,3%). Embora a maioria aponte o vazamento de dados confidenciais e o uso indevido de dados pessoais como os maiores riscos dos LLMs, 32,1% nunca anonimizam dados antes de submetê-los aos modelos. A infraestrutura nacional tem milhares de bases expostas sem autenticação (1.480 MySQL, 779 InfluxDB, 506 Redis e 3.677 compartilhamentos SMB) e 85% das vulnerabilidades em serviços web persistentes por anos sem correção (Ponce *et al.*, 2023). Há ainda dificuldade de traduzir os preceitos jurídicos abstratos da LGPD em requisitos, controles e testes concretos de implementação no código-fonte (Frej *et al.*, 2024). Na interpretação dos autores, essa barreira se acentua quando desenvolvedores

recorrem a assistentes de IA treinados em contextos estrangeiros, que desconhecem as exigências regulatórias nacionais.

4. DISCUSSÃO E ANÁLISE DOS DADOS

4.1. Produtividade e Responsabilidade na Revisão do Código

Os achados do corpus mostram uma tensão na engenharia de software atual: os assistentes de IA aceleram o desenvolvimento, mas criam fragilidades cognitivas e operacionais novas. Por um lado, os modelos reduzem o tempo de escrita de código funcional em 55,8% (Peng *et al.*, 2023), estimulando a delegação de tarefas à IA.

Contudo, essa percepção de produtividade acarreta efeitos adversos, a começar pelo viés de superconfiança (overconfidence bias): programadores assistidos por IA tendem a superestimar a solidez do que escrevem, considerando o código seguro mesmo quando contém defeitos graves em autenticação, validação ou criptografia (Perry *et al.*, 2023). Isso acontece porque as sugestões são sintaticamente fluidas: compilam sem advertências e cumprem a função solicitada, ocultando falhas estruturais subjacentes.

Associado a essa superconfiança, o esforço de revisão humana muda de foco. Embora os desenvolvedores gastem muito tempo interagindo com as ferramentas e inspecionando trechos gerados (Mozannar *et al.*, 2024), na avaliação dos autores essa verificação concentra-se predominantemente na resolução de problemas imediatos de sintaxe e comportamento funcional. Os achados sugerem que o tempo poupado com a automação da digitação não é convertido em auditoria preventiva de requisitos não funcionais nem em testes de robustez de segurança (Mozannar *et al.*, 2024). Quando a automação escala para agentes autônomos submetendo

pull requests completos, Siddiq *et al.* (2026) demonstram que a revisão de PRs de segurança gerados por agentes leva mais de uma ordem de grandeza a mais de tempo que a de PRs comuns. A velocidade de geração sintática, portanto, não se traduz em igual velocidade de revisão, o que fragiliza qualquer equivalência entre gerar rápido e gerar com segurança.

Essa dinâmica cognitiva repercute diretamente na dimensão upstream: a interação em linguagem natural no próprio editor leva o desenvolvedor a colar trechos de código, credenciais e dados corporativos nos prompts. Como os modelos comerciais transferem contratualmente a responsabilidade preventiva ao usuário e carecem de filtros ativos de tráfego por padrão (Al-Maamari, 2025; Madampe; Grundy; Arachchilage, 2025), e 32,1% dos desenvolvedores brasileiros avaliados admitem nunca anonimizar ou pseudonimizar dados antes de enviá-los aos LLMs (Falcão; Canedo, 2024), dados pessoais identificáveis e segredos ficam expostos ao compartilhamento indevido com provedores externos. Esse comportamento é consistente com o distanciamento apontado por Horstmann *et al.* (2024), no qual desenvolvedores concebem a segurança de maneira restrita e relatam dificuldade em compreender e traduzir diretrizes jurídicas formais de privacidade em requisitos técnicos.

4.2. Da Fragilidade Técnica Ao Risco de Tratamento Irregular Sob a LGPD

Cruzar as evidências de código vulnerável (RQ2) com o ordenamento jurídico brasileiro é uma proposta analítica preliminar, feita pelos autores sob uma ótica interdisciplinar técnico-jurídica, e não uma relação causal medida pelos estudos internacionais. Ao analisar a

taxonomia de falhas técnicas à luz da LGPD, descarta-se a ideia simplista de que qualquer defeito de software seja, por si só, uma ilicitude jurídica consumada.

O enquadramento normativo adequado repousa no Art. 44 da LGPD, que define o conceito de tratamento irregular de dados a partir da quebra da expectativa legítima de segurança que o titular pode esperar, considerando os riscos razoáveis e o estado da técnica disponível à época do tratamento (Brasil, 2018). Quando uma organização permite que assistentes de IA introduzam no software vulnerabilidades severas e amplamente conhecidas, tais como injeção de SQL (CWE-89), injeção de comandos (CWE-78), desserialização insegura (CWE-502) e geradores aleatórios inseguros (CWE-330), cuja persistência e gravidade são atestadas tanto em cenários controlados e replicações (Majdinasab *et al.*, 2024; Pearce *et al.*, 2022; Zhao *et al.*, 2025) quanto em projetos reais e comparações diretas com código humano (Cotroneo; Improta; Liguori, 2025; Fu *et al.*, 2025), ela falha em fornecer a segurança razoável exigida pelo ordenamento, configurando risco concreto de tratamento irregular.

A materialização dessas fraquezas viola deveres legais expressos:

1. Princípios da Segurança, Prevenção e Privacy by Design (Arts. 6º, VII e VIII, e 46): A recorrência de falhas estruturais demonstra a omissão de medidas técnicas preventivas desde as fases iniciais do ciclo de desenvolvimento, contrariando o mandamento de segurança desde a concepção (privacy by design, art. 46, § 2º);

2. Garantia de Segurança da Informação (Arts. 46 e 47 da LGPD): A inclusão de credenciais fixas no código (CWE-798/259) e o

registro descontrolado de dados pessoais em arquivos de log (CWE-532) comprometem os mecanismos de autenticação e expõem informações sensíveis a acessos indevidos, contrariando o dever de adotar medidas de segurança aptas a proteger os dados (art. 46) e de garantir essa segurança mesmo após o término do tratamento (art. 47);

3. Riscos na Cadeia de Suprimentos (Art. 44 da LGPD): A incorporação de dependências externas, bibliotecas de terceiros ou artefatos sugeridos por modelos sem validação de proveniência e integridade (OWASP, 2025, LLM03) abre vetores para comprometimento da cadeia de suprimentos de software e exposição sistêmica das aplicações;

4. Transferência Internacional de Dados (Art. 33 da LGPD): A transmissão de dados pessoais em prompts para infraestruturas estrangeiras desprovidas de respaldo em Cláusulas-Padrão Contratuais ou garantias equivalentes aprovadas pela autoridade (ANPD, 2024c, arts. 15 a 17) caracteriza transferência internacional em desconformidade com a lei;

5. Sujeição ao Regime de Reparação Civil por Danos (Arts. 42 a 45 da LGPD): Caso as vulnerabilidades geradas por IA resultem em vazamento ou acesso indevido com danos patrimoniais, morais, individuais ou coletivos aos titulares, os agentes de tratamento submetem-se ao regime de reparação civil dos arts. 42 a 45 da LGPD. Não há nexo causal automático entre a falha no código e o dano indenizável: a reparação exige a demonstração do dano e do nexo com o tratamento, admitindo-se a inversão do ônus da prova a juízo do

magistrado quando for verossímil a alegação ou hipossuficiente o titular (art. 42, § 2º), além da imposição de sanções administrativas (art. 52) e do dever de comunicação tempestiva à autoridade e aos titulares (art. 48).

4.3. O Contexto Corporativo Nacional e as Perspectivas Regulatórias

No contexto brasileiro, esses riscos recaem sobre um ambiente que já apresenta fragilidades de governança e infraestrutura (Falcão; Canedo, 2024; Frej *et al.*, 2024; Ponce *et al.*, 2023). A adoção expressiva de LLMs ocorre sem treinamento formal e com omissão frequente de anonimização, sobre uma infraestrutura nacional com passivos históricos de cibersegurança (Ponce *et al.*, 2023), e persiste a dificuldade de traduzir os preceitos jurídicos da LGPD em requisitos e testes de código (Frej *et al.*, 2024).

Nesse cenário de fragilidades estruturais nas organizações, a adoção sem governança dos assistentes de codificação amplia os riscos. Embora técnicas de mitigação via RAG e Chain-of-Thought elevem a geração de código seguro (Zhao *et al.*, 2025), elas não eliminam a totalidade dos defeitos complexos (como CWE-22 e CWE-502). Além disso, quando o desenvolvimento recorre a agentes autônomos submetendo pull requests, a latência mediana de revisão humana para alterações de segurança é substancialmente maior (Siddiq *et al.*, 2026), tornando indispensável a combinação de verificação estática/dinâmica determinística e supervisão humana (human-in-the-loop).

No âmbito regulatório, o Brasil passou a tratar o tema com medidas concretas em 2024. Na esfera legislativa, aprovado no Senado em

dezembro de 2024 e em tramitação na Câmara dos Deputados, onde aguarda parecer do relator na comissão especial, o Projeto de Lei nº 2.338/2023 (Marco Legal da Inteligência Artificial) (Brasil, 2023) propõe o estabelecimento de exigências rigorosas de gestão de riscos, governança de dados, explicabilidade algorítmica e supervisão humana para sistemas de IA de alto risco e, caso aprovado, imporia deveres objetivos tanto para desenvolvedores quanto para agentes que implementam essas tecnologias no ambiente produtivo.

Na atuação fiscalizatória, a Autoridade Nacional de Proteção de Dados (ANPD) adotou medidas preventivas expressivas em 2024 perante o uso de dados pessoais em modelos generativos. Por meio do Despacho Decisório nº 20/2024/PR/ANPD (Processo Administrativo nº 00261.004509/2024-36) (ANPD, 2024a), a autoridade determinou inicialmente a suspensão preventiva imediata do uso de dados pessoais de titulares brasileiros para treinamento de modelos generativos pela Meta Platforms, apontando riscos de tratamento irregular por falta de transparência e de bases legais adequadas. A eficácia dessa medida cautelar apenas foi suspensa pelo Despacho Decisório nº 33/2024/PR/ANPD (ANPD, 2024b) após a empresa submeter e aprovar um plano estrito de conformidade perante a autarquia, o qual exigiu notificações prévias com 30 dias de antecedência, mecanismos facilitados de oposição (opt-out) diretamente nas interfaces dos aplicativos e a exclusão integral de dados pertencentes a contas de crianças e adolescentes.

Esse histórico regulatório indica que o emprego corporativo de assistentes de IA no Brasil demanda governança técnica ativa, sob pena de responsabilização administrativa e civil. Usar ferramentas

que capturam dados corporativos em prompts ou inserem código sem garantias razoáveis de segurança expõe as organizações à fiscalização e a penalidades severas, e exige alinhar as práticas de engenharia de software com a governança jurídica.

4.4. Implicações Práticas para a Engenharia de Software Segura (DevSecOps)

Para conciliar os benefícios de produtividade da inteligência artificial com a observância rigorosa das exigências legais, as equipes de desenvolvimento devem estabelecer esteiras seguras (DevSecOps) integrando salvaguardas nos fluxos upstream e downstream:

1. Higienização de prompts, anonimização e filtragem ativa de segredos (upstream - LLM02): Implementação de ferramentas de sanitização (data scrubbing), anonimização/pseudonimização preventiva (Falcão; Canedo, 2024) e detecção de credenciais (secret filtering) operando na borda das IDEs, impedindo que tokens, chaves de API, senhas ou dados pessoais identificáveis sejam enviados nos contextos de consulta a modelos externos;

2. Abordagem zero-trust, engenharia de prompt guiada por CWE e verificação contínua (downstream - LLM05 e LLM06): Adoção de desconfiança sistemática em relação ao código sugerido por assistentes de IA, combinando diretrizes de codificação segura injetadas via RAG no contexto de geração (Zhao *et al.*, 2025) com submissão obrigatória a ferramentas de análise estática (SAST) e dinâmica (DAST) configuradas para barrar injeções de código (CWE-89, CWE-78), desserialização insegura (CWE-502), geradores aleatórios fracos (CWE-330) e

segredos embutidos (CWE-798), além da sanitização rigorosa das saídas com consultas preparadas (prepared statements) e codificação de saídas (output encoding);

3. Gestão de dependências e SBOM (LLM03): Rastreabilidade e verificação de integridade de bibliotecas de terceiros mediante inventários de componentes (Software Bill of Materials – SBOM e ML-BOM), mitigando vulnerabilidades na cadeia de suprimentos de software e modelos (OWASP, 2025);

4. Alinhamento às recomendações orientativas da ANPD (2021): Adoção de controle de acesso rigoroso baseado no menor privilégio (need-to-know), autenticação multifator (MFA) em todos os repositórios corporativos, aplicação de criptografia robusta para dados em repouso e em trânsito com base nos padrões recomendados pelo mercado (TLS 1.3 e AES-256-GCM), celebração de acordos formais de tratamento de dados com fornecedores de tecnologia e treinamento técnico periódico das equipes;

5. Revisão humana obrigatória (human-in-the-loop - LLM01 e LLM06): Fixação de diretrizes organizacionais que vedem a aprovação automática de solicitações de integração (pull requests) geradas por agentes autônomos em módulos críticos de segurança e privacidade, exigindo validação determinística e aprovação por revisores humanos (OWASP, 2025; Siddiq *et al.*, 2026).

5. CONSIDERAÇÕES FINAIS

5.1. Síntese das Respostas às Perguntas de Pesquisa

Esta Revisão Sistemática da Literatura (RSL) seguiu as diretrizes de Kitchenham e Charters (2007) e o protocolo PRISMA 2020 (Page *et al.*, 2021) para investigar o impacto dos assistentes de codificação baseados em IA generativa na conformidade com a Lei Geral de Proteção de Dados Pessoais (LGPD – Lei nº 13.709/2018) e as normas da autoridade reguladora. A tese central é que a busca por velocidade no desenvolvimento assistido por IA não pode se sobrepor às exigências de segurança, prevenção e privacidade que a lei impõe aos agentes de tratamento.

Na RQ1, os estudos sobre o fluxo upstream e o comportamento humano mostram que a velocidade que a IA proporciona alimenta o *vibe coding*, associado a vieses cognitivos: o viés de superconfiança (overconfidence bias), a revisão superficial de código e a sobrecarga de revisar pull requests de agentes autônomos (Mozannar *et al.*, 2024; Perry *et al.*, 2023; Siddiq *et al.*, 2026). Ao mesmo tempo, as ferramentas comerciais não trazem filtragem prévia de segredos (secret filtering) por padrão (Al-Maamari, 2025), e 32,1% dos profissionais pesquisados no Brasil não anonimizam os dados antes de enviar os prompts (Falcão; Canedo, 2024), o que favorece a exposição acidental de credenciais e dados pessoais a servidores de terceiros.

Na RQ2, a análise do código downstream mostra que os modelos de linguagem geram uma proporção expressiva de código com fraquezas críticas da taxonomia CWE (27,25% mesmo após as atualizações dos filtros preventivos do Copilot, segundo Majdinasab *et al.*, 2024), acima da incidência de vulnerabilidades do código escrito por humanos em comparações de larga escala (Cotroneo; Improta; Liguori, 2025; Fu *et al.*, 2025). As falhas mais recorrentes são injeção de comandos (CWE-78), injeção de SQL (CWE-89), travessia

de diretório (CWE-22), desserialização insegura (CWE-502), geradores pseudoaleatórios inseguros (CWE-330), exposição de dados em logs (CWE-532) e credenciais fixas (CWE-798). Para mitigá-las, é preciso combinar geração guiada por diretrizes CWE via RAG (Zhao *et al.*, 2025) com verificação determinística em esteiras DevSecOps.

Na RQ3, a pesquisa propôs um mapeamento técnico-normativo preliminar entre as fragilidades do corpus e os dispositivos da LGPD. A conclusão é que incorporar código vulnerável sem revisão não é, por si só, uma violação automática, mas pode configurar risco de tratamento irregular nos termos do Art. 44 da LGPD, ferindo os princípios da segurança, prevenção e responsabilização (Art. 6º, VII, VIII e X), os deveres de privacy by design (Art. 46) e as regras de transferência internacional de dados (Art. 33 c/c ANPD, 2024c, arts. 15 a 17), e sujeitando os agentes à reparação civil por danos (arts. 42 a 45 da LGPD) e a sanções administrativas (art. 52).

5.2. Principais Contribuições do Trabalho

O estudo contribui em três frentes:

Primeiro, o trabalho apresenta uma proposta analítica preliminar de mapeamento técnico-normativo, ligando as falhas técnicas recorrentes do código gerado por IA (CWE-89, CWE-78, CWE-22, CWE-502, CWE-330, CWE-798, CWE-532) e os riscos de exposição upstream à LGPD e às resoluções da ANPD. Esse enquadramento inicial ajuda equipes de engenharia de software a compreender como defeitos de codificação podem configurar riscos de tratamento irregular perante a lei brasileira.

Segundo, a pesquisa sistematiza o impacto dos fatores comportamentais e cognitivos no desenvolvimento assistido,

mostrando como a ilusão de produtividade do *vibe coding* e o viés de superconfiança diluem a responsabilidade técnica e levam à incorporação de código vulnerável sem revisão, expondo os agentes de tratamento a reparação civil e sanções administrativas.

Por fim, o estudo consolida um catálogo de salvaguardas para esteiras de DevSecOps, articulando as boas práticas da ANPD com as diretrizes do OWASP Top 10 for LLM Applications (2025). São prescrições práticas para higienizar prompts, realizar análise determinística Zero-Trust e garantir revisão humana obrigatória (human-in-the-loop) no ciclo de desenvolvimento.

5.3. Ameaças à Validade

As limitações desta revisão seguem os quatro tipos de ameaças à validade usados na Engenharia de Software:

Validade Interna (Internal Validity): refere-se à consistência da seleção e da extração dos dados e ao possível viés de interpretação dos revisores. Para mitigar essa ameaça, adotou-se o processo descrito na Seção 2.2, com registro integral das justificativas de decisão para as 1.140 obras únicas na planilha de auditoria (Material Suplementar). Reconhece-se como limitação metodológica a ausência de registro prévio do protocolo em bases externas (como OSF) e a resolução de divergências por consenso direto sem o cálculo estatístico do coeficiente Kappa de concordância interavaliadores. Adicionalmente, 13 dos 18 estudos primários atingiram a nota máxima de qualidade (3,0), o que limita o poder discriminativo da avaliação de qualidade, visto que a QA descreve a qualidade do corpus, mas diferencia pouco os estudos entre si.

Validade de Construto (Construct Validity): avalia se as variáveis, os termos de busca e os conceitos usados são adequados ao problema de pesquisa. A literatura sobre modelos de linguagem e assistentes de código evolui em ritmo acelerado, gerando terminologias emergentes contínuas. Para mitigar eventuais omissões conceituais, desenhou-se uma estratégia de busca estruturada em três eixos temáticos complementares com consultas restritas aos campos de metadados, associada ao rastreamento ativo de citações (snowballing) e à inclusão manual de resoluções oficiais da ANPD e diretrizes técnicas da OWASP. A cobertura das bases constitui limitação adicional: a ausência de Scopus e Web of Science, por decisão de escopo do protocolo, pode ter deixado de fora literatura indexada exclusivamente nessas bases, e a replicação das buscas nelas é proposta como agenda de continuidade. Os 11 registros excluídos por indisponibilidade do texto integral (CE3, restrição de paywall) constituem limitação de cobertura.

Validade Externa (External Validity): examina em que medida os resultados da literatura internacional valem para a realidade das empresas brasileiras. Nenhum dos quinze estudos empíricos primários internacionais investigou diretamente o ordenamento jurídico brasileiro. Embora os dados empíricos sobre CWEs derivem de repositórios abertos globais e experimentos controlados, a contextualização para o cenário nacional foi subsidiada pelos três estudos empíricos primários nacionais do corpus (Falcão; Canedo, 2024; Frej *et al.*, 2024; Ponce *et al.*, 2023), os quais indicam a omissão de anonimização de dados em prompts por parte de desenvolvedores no país, dificuldades práticas na tradução de requisitos da LGPD para o código e defasagens históricas de segurança em ativos de rede no Brasil.

Validade de Conclusão (Conclusion Validity): verifica se as evidências coletadas sustentam as inferências feitas na RQ3. O mapeamento entre as fraquezas técnicas (CWEs) e os dispositivos da LGPD constitui um enquadramento técnico-jurídico analítico proposto pelos autores. Embora as correspondências normativas tenham passado por revisão jurídica interna, o modelo não foi submetido à validação empírica por painel externo independente de especialistas (como o método Delphi) nem a levantamento jurisprudencial exaustivo. A existência isolada de uma falha de software não configura ilicitude automática em abstrato, sendo indispensável a presença de tratamento de dados pessoais, ausência de segurança razoável (art. 44) e ocorrência de dano efetivo para ensejar reparação civil (arts. 42 a 45 da LGPD).

5.4. Trabalhos Futuros

Para dar continuidade à investigação em engenharia de software segura e governança de dados, sugerem-se as seguintes direções de pesquisa:

1. Desenvolvimento de ferramentas de análise estática adaptadas às diretrizes da ANPD: Projetar, implementar e avaliar ferramentas automatizadas de análise estática (linters e extensões para ambientes de desenvolvimento como VS Code, Cursor e IntelliJ) capazes de verificar, em tempo de escrita, a conformidade de requisições a LLMs com as normas da ANPD, executando data scrubbing preventivo em prompts e bloqueando a inserção de código com CWEs críticas;

2. Investigações empíricas na indústria de software nacional: Conduzir estudos de campo e pesquisas aplicadas (surveys e

estudos de caso) com empresas brasileiras de desenvolvimento de software, quantificando a incidência de falhas técnicas decorrentes de assistentes de IA em código de produção e avaliando a maturidade organizacional na adoção de salvaguardas de DevSecOps e conformidade com a LGPD;

3. Validação empírica e interdisciplinar do mapeamento técnico-normativo (método Delphi e jurisprudência da ANPD): Submeter a proposta analítica preliminar de mapeamento entre a taxonomia CWE e os dispositivos da LGPD (Tabela 7) à validação empírica por meio de painéis interdisciplinares de especialistas em Direito Digital, Proteção de Dados e Segurança da Informação (empregando o método Delphi ou grupos focais), combinada à análise sistemática de processos administrativos sancionatórios e decisões da ANPD.

REFERÊNCIAS BIBLIOGRÁFICAS

AL-MAAMARI, A. Can you trust your copilot? A privacy scorecard for AI coding assistants. arXiv preprint arXiv:2509.20388, [S. l.], 2025. DOI: 10.48550/arXiv.2509.20388. Disponível em: https://doi.org/10.48550/arXiv.2509.20388. Acesso em: 26 abr. 2026.

AMRO, A.; ALALFI, M. H. GitHub's Copilot Code Review: Can AI Spot Security Flaws Before You Commit? arXiv preprint arXiv:2509.13650, [S. l.], 2025. DOI: 10.48550/arXiv.2509.13650. Disponível em: https://doi.org/10.48550/arXiv.2509.13650. Acesso em: 26 abr. 2026.

ANASURI, S.; TEKALE, K. M. Securing AI-assisted cloud engineering: Guardrails for Copilot-generated IaC and CI/CD changes to prevent vulnerability injection. International Journal of Scientific Research and Engineering Trends, [S. l.], v. 10, n. 6, p. 1120–1128, 2024. DOI:

10.5281/zenodo.18594686. Disponível em: https://doi.org/10.5281/zenodo.18594686. Acesso em: 26 abr. 2026.

ANPD – Autoridade Nacional de Proteção de Dados. Despacho Decisório nº 20/2024/PR/ANPD, de 2 de julho de 2024. Processo nº 00261.004509/2024-36. Diário Oficial da União: seção 1, Brasília, DF, 2 jul. 2024a, p. 180. Disponível em: https://www.in.gov.br/web/dou/-/despacho-decisorio-n-20/2024/pr/anpd-569297245. Acesso em: 26 abr. 2026.

ANPD – Autoridade Nacional de Proteção de Dados. Despacho Decisório nº 33/2024/PR/ANPD, de 30 de agosto de 2024. Processo nº 00261.004529/2024-36. Diário Oficial da União: seção 1, Brasília, DF, 30 ago. 2024b. Disponível em: https://www.in.gov.br/en/web/dou/-/despacho-decisorio-n-33/2024/pr/anpd-581192714. Acesso em: 26 abr. 2026.

ANPD – Autoridade Nacional de Proteção de Dados. Guia orientativo de segurança da informação para agentes de tratamento de pequeno porte. Brasília: ANPD / Governo Federal, 2021. Disponível em: https://www.gov.br/anpd/pt-br/documentos-e-publicacoes/guia-vf.pdf. Acesso em: 26 abr. 2026.

ANPD – Autoridade Nacional de Proteção de Dados. Resolução CD/ANPD nº 19, de 23 de agosto de 2024. Regulamento de Transferência Internacional de Dados. Diário Oficial da União: seção 1, Brasília, DF, 26 ago. 2024c. Disponível em: https://www.in.gov.br/en/web/dou/-/resolucao-cd/anpd-n-19-de-23-de-agosto-de-2024-580095396. Acesso em: 26 abr. 2026.

ASARE, O.; NAGAPPAN, M.; ASOKAN, N. A user-centered security evaluation of Copilot. In: IEEE/ACM INTERNATIONAL CONFERENCE

ON SOFTWARE ENGINEERING (ICSE), 46., 2024, Lisbon. Proceedings [...]. New York: ACM, 2024. Artigo 161, p. 1–11. DOI: 10.1145/3597503.3639154. Disponível em: https://doi.org/10.1145/3597503.3639154. Acesso em: 26 abr. 2026.

ASARE, O.; NAGAPPAN, M.; ASOKAN, N. Is GitHub's Copilot as bad as humans at introducing vulnerabilities in code? Empirical Software Engineering, [S. l.], v. 28, n. 6, artigo 129, p. 1–28, 2023. DOI: 10.1007/s10664-023-10380-1. Disponível em: https://doi.org/10.1007/s10664-023-10380-1. Acesso em: 26 abr. 2026.

BRASIL. Câmara dos Deputados. Projeto de Lei nº 2.338, de 2023. Dispõe sobre o uso da inteligência artificial. Brasília, DF: Câmara dos Deputados, 2023. Disponível em: https://www.camara.leg.br/proposicoesWeb/fichadetramitacao?idProposicao=2487262. Acesso em: 26 abr. 2026.

BRASIL. Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD). Diário Oficial da União: seção 1, Brasília, DF, 15 ago. 2018. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm. Acesso em: 26 abr. 2026.

CERAN, O. Security evaluation of AI-generated code: A comparative study of ChatGPT, Copilot, and Gemini through static and dynamic analysis. Gazi Mühendislik Bilimleri Dergisi, [S. l.], v. 11, n. 2, p. 304–320, 2025. DOI: 10.30855/gmbd.070525A13. Disponível em: https://doi.org/10.30855/gmbd.070525A13. Acesso em: 26 abr. 2026.

COTRONEO, D.; IMPROTA, C.; LIGUORI, P. Human-written vs. AI-generated code: A large-scale study of defects, vulnerabilities, and complexity. In: IEEE INTERNATIONAL SYMPOSIUM ON SOFTWARE

RELIABILITY ENGINEERING (ISSRE), 36., 2025, São Paulo. Proceedings [...]. Piscataway: IEEE, 2025. p. 252–263. DOI: 10.1109/ISSRE66568.2025.00035. Disponível em: https://doi.org/10.1109/ISSRE66568.2025.00035. Acesso em: 26 abr. 2026.

CRUZES, D. S.; DYBÅ, T. Recommended steps for thematic synthesis in software engineering. In: INTERNATIONAL SYMPOSIUM ON EMPIRICAL SOFTWARE ENGINEERING AND MEASUREMENT (ESEM), 5., 2011, Banff. Proceedings [...]. Piscataway: IEEE, 2011. p. 275–284. DOI: 10.1109/ESEM.2011.36. Disponível em: https://doi.org/10.1109/ESEM.2011.36. Acesso em: 26 abr. 2026.

DONG, J. *et al.* Securing AI code generation: A prompt rectification approach for mitigating cyber risks. In: IEEE INTERNATIONAL CONFERENCE ON TRUST, SECURITY AND PRIVACY IN COMPUTING AND COMMUNICATIONS (TrustCom), 24., 2025, Guiyang. Proceedings [...]. Piscataway: IEEE, 2025. p. 1–10. DOI: 10.1109/TrustCom66490.2025.00106. Disponível em: https://doi.org/10.1109/TrustCom66490.2025.00106. Acesso em: 26 abr. 2026.

FALCÃO, F. D. S.; CANEDO, E. D. Investigating software development teams members' perceptions of data privacy in the use of large language models. In: BRAZILIAN SYMPOSIUM ON SOFTWARE QUALITY (SBQS), 23., 2024, Salvador. Proceedings [...]. New York: ACM, 2024. p. 373–382. DOI: 10.1145/3701625.3701675. Disponível em: https://doi.org/10.1145/3701625.3701675. Acesso em: 26 abr. 2026.

FREJ, M. *et al.* Um sistema web para auxiliar soluções na conformidade com a LGPD. In: SIMPÓSIO BRASILEIRO DE

ENGENHARIA DE SOFTWARE (SBES), 38., 2024, Curitiba. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2024. p. 713–719. DOI: 10.5753/sbes.2024.3558. Disponível em: https://doi.org/10.5753/sbes.2024.3558. Acesso em: 26 abr. 2026.

FU, Y. *et al.* Security weaknesses of Copilot-generated code in GitHub projects: An empirical study. ACM Transactions on Software Engineering and Methodology (TOSEM), New York, v. 34, n. 8, Artigo 218, p. 1–34, 2025. DOI: 10.1145/3716848. Disponível em: https://doi.org/10.1145/3716848. Acesso em: 26 abr. 2026.

HARZEVILI, N. S. *et al.* A systematic literature review on automated software vulnerability detection using machine learning. ACM Computing Surveys, New York, v. 57, n. 3, p. 1–36, 2025. DOI: 10.1145/3699711. Disponível em: https://doi.org/10.1145/3699711. Acesso em: 26 abr. 2026.

HORSTMANN, S. A. *et al.* “Those things are written by lawyers, and programmers are reading that.” Mapping the communication gap between software developers and privacy experts. Proceedings on Privacy Enhancing Technologies (PoPETs), [S. l.], v. 2024, n. 1, p. 151–170, 2024. DOI: 10.56553/popets-2024-0010. Disponível em: https://doi.org/10.56553/popets-2024-0010. Acesso em: 26 abr. 2026.

KARPATHY, A. Vibe coding. X (formerly Twitter), [S. l.], 4 fev. 2025. Disponível em: https://x.com/karpathy/status/1886192184808988973. Acesso em: 26 abr. 2026.

KHOURY, R. *et al.* How secure is code generated by ChatGPT? An empirical evaluation. In: IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN, AND CYBERNETICS (SMC), 2023, Oahu. Proceedings [...]. Piscataway: IEEE, 2023. p. 2445–2451. DOI:

10.1109/SMC53992.2023.10394237. Disponível em: https://doi.org/10.1109/SMC53992.2023.10394237. Acesso em: 26 abr. 2026.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing Systematic Literature Reviews in Software Engineering. Technical Report EBSE-2007-01, Keele University, 2007. Disponível em: https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf. Acesso em: 26 abr. 2026.

MADAMPE, K.; GRUNDY, J.; ARACHCHILAGE, N. How are we doing with using AI-based programming assistants for privacy-related code generation? The developers' experience. In: INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING (EASE), 29., 2025, Istanbul. Proceedings [...]. New York: ACM, 2025. p. 684–689. DOI: 10.1145/3756681.3757035. Disponível em: https://doi.org/10.1145/3756681.3757035. Acesso em: 26 abr. 2026.

MAJDINASAB, V. *et al.* Assessing the security of GitHub Copilot's generated code – A targeted replication study. In: IEEE INTERNATIONAL CONFERENCE ON SOFTWARE ANALYSIS, EVOLUTION AND REENGINEERING (SANER), 31., 2024, Rovaniemi. Proceedings [...]. Piscataway: IEEE, 2024. p. 649–660. DOI: 10.1109/SANER60148.2024.00051. Disponível em: https://doi.org/10.1109/SANER60148.2024.00051. Acesso em: 26 abr. 2026.

MESKE, C. *et al.* Vibe coding: Opportunities and challenges of coding with generative AI without programming knowledge. arXiv preprint arXiv:2502.15843, [S. l.], 2025. DOI: 10.48550/arXiv.2502.15843.

Disponível em: https://doi.org/10.48550/arXiv.2502.15843. Acesso em: 26 abr. 2026.

MOZANNAR, H. *et al.* Reading between the lines: Modeling user behavior and costs in AI-assisted programming. In: CHI CONFERENCE ON HUMAN FACTORS IN COMPUTING SYSTEMS (CHI '24), 2024, Honolulu. Proceedings [...]. New York: ACM, 2024. Artigo 142, p. 1–16. DOI: 10.1145/3613904.3641936. Disponível em: https://doi.org/10.1145/3613904.3641936. Acesso em: 26 abr. 2026.

NIU, L. *et al.* CodexLeaks: Privacy leaks from code generation language models in GitHub Copilot. In: USENIX SECURITY SYMPOSIUM, 32., 2023, Anaheim. Proceedings [...]. [S. l.]: USENIX Association, 2023. p. 2133–2150. Disponível em: https://www.usenix.org/conference/usenixsecurity23/presentation/niu. Acesso em: 26 abr. 2026.

OWASP Foundation. OWASP Top 10 for Large Language Model Applications. Open Web Application Security Project, 2025. Disponível em: https://owasp.org/www-project-top-10-for-large-language-model-applications/. Acesso em: 26 abr. 2026.

PAGE, M. J. *et al.* The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. BMJ, [S. l.], v. 372, n. 71, p. 1–9, 2021. DOI: 10.1136/bmj.n71. Disponível em: https://doi.org/10.1136/bmj.n71. Acesso em: 26 abr. 2026.

PEARCE, H. *et al.* Asleep at the keyboard? Assessing the security of GitHub Copilot's code contributions. In: IEEE SYMPOSIUM ON SECURITY AND PRIVACY (S&P), 43., 2022, San Francisco. Proceedings [...]. Piscataway: IEEE, 2022. p. 754–768. DOI:

10.1109/SP46214.2022.9833571. Disponível em: https://doi.org/10.1109/SP46214.2022.9833571. Acesso em: 26 abr. 2026.

PENG, S. *et al.* The impact of AI on developer productivity: Evidence from GitHub Copilot. arXiv preprint arXiv:2302.06590, [S. l.], 2023. DOI: 10.48550/arXiv.2302.06590. Disponível em: https://doi.org/10.48550/arXiv.2302.06590. Acesso em: 26 abr. 2026.

PERRY, N. *et al.* Do users write more insecure code with AI assistants? In: ACM SIGSAC CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY (CCS), 30., 2023, Copenhagen. Proceedings [...]. New York: ACM, 2023. p. 2785–2799. DOI: 10.1145/3576915.3623157. Disponível em: https://doi.org/10.1145/3576915.3623157. Acesso em: 26 abr. 2026.

PONCE, L. M. *et al.* Um arcabouço para processamento escalável de vulnerabilidades e caracterização de riscos à conformidade da LGPD. In: SIMPÓSIO BRASILEIRO DE CIBERSEGURANÇA (SBSEG), 23., 2023, Juiz de Fora. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2023. p. 15–28. DOI: 10.5753/sbseg.2023.233114. Disponível em: https://doi.org/10.5753/sbseg.2023.233114. Acesso em: 26 abr. 2026.

SANDOVAL, G. *et al.* Lost at C: A user study on the security implications of large language model code assistants. In: USENIX SECURITY SYMPOSIUM, 32., 2023, Anaheim. Proceedings [...]. [S. l.]: USENIX Association, 2023. p. 2205–2222. Disponível em: https://www.usenix.org/conference/usenixsecurity23/presentation/sandoval. Acesso em: 26 abr. 2026.

SANTOS, R. F. *et al.* Desenvolvimento de sistemas inteligentes para detecção de vulnerabilidades em tempo real com aplicações em

cibersegurança. Revista Tópicos, Rio de Janeiro, v. 4, n. 32, p. 1–27, 2026. DOI: 10.70773/revistatopicos/775413946. Disponível em: https://doi.org/10.70773/revistatopicos/775413946. Acesso em: 26 abr. 2026.

SARAIVA, J.; SOUZA, C. de; SOARES, S. Desafios de compliance da LGPD: Implantação na indústria de software brasileira. In: WORKSHOP SOBRE ASPECTOS SOCIAIS, HUMANOS E ECONÔMICOS DE SOFTWARE (WASHES), 9., 2024, Brasília. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2024. p. 193–198. DOI: 10.5753/washes.2024.3079. Disponível em: https://doi.org/10.5753/washes.2024.3079. Acesso em: 26 abr. 2026.

SIDDIQ, M. L. *et al.* Security in the age of AI teammates: An empirical study of agentic pull requests on GitHub. arXiv preprint arXiv:2601.00477, [S. l.], 2026. DOI: 10.48550/arXiv.2601.00477. Disponível em: https://doi.org/10.48550/arXiv.2601.00477. Acesso em: 26 abr. 2026.

SPRACKLEN, J. *et al.* We have a package for you! A comprehensive analysis of package hallucinations by code generating LLMs. In: USENIX SECURITY SYMPOSIUM, 34., 2025, Seattle. Proceedings [...]. [S. l.]: USENIX Association, 2025. Disponível em: https://www.usenix.org/conference/usenixsecurity25/presentation/spracklen. Acesso em: 26 abr. 2026.

WOHLIN, C. Guidelines for snowballing in systematic literature studies and a replication in software engineering. In: INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING (EASE), 18., 2014, London. Proceedings [...]. New York: ACM, 2014. Artigo 38, p. 1–10. DOI:

10.1145/2601248.2601268. Disponível em: https://doi.org/10.1145/2601248.2601268. Acesso em: 26 abr. 2026.

ZHAO, J. *et al.* Towards secure code generation with LLMs: A study on Common Weakness Enumeration. IEEE Transactions on Software Engineering, Piscataway, v. 51, n. 12, p. 3507–3523, 2025. DOI: 10.1109/TSE.2025.3619281. Disponível em: https://doi.org/10.1109/TSE.2025.3619281. Acesso em: 26 abr. 2026.

---

[1] Discente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID: https://orcid.org/0009-0007-8473-3955. E-mail: acesse o artigo original para visualizar o e-mail

[2] Discente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID: https://orcid.org/0009-0004-2650-8571. E-mail: acesse o artigo original para visualizar o e-mail

[3] Discente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID: https://orcid.org/0009-0004-9146-2012. E-mail: acesse o artigo original para visualizar o e-mail

[4] Discente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID:

https://orcid.org/0009-0007-8356-6948. E-mail: acesse o artigo original para visualizar o e-mail

[5] Bacharel em Direito pelo Centro Universitário de Brasília (UniCEUB). SEPN 707/907, Asa Norte, Brasília, DF, CEP 70790-075. Advogado (OAB/DF 79.262) no MFA Advogadas. ORCID: https://orcid.org/0009-0004-8905-2671. E-mail: acesse o artigo original para visualizar o e-mail

[6] Doutora em Bioquímica pela Universidade Federal do Rio de Janeiro. Docente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID: https://orcid.org/0000-0003-3825-7242. E-mail: acesse o artigo original para visualizar o e-mail

[7] Mestre em Gestão do Trabalho pela Universidade Santa Úrsula. Docente do Curso de Bacharelado em Engenharia de Software da Universidade de Vassouras, Campus Maricá. Av. Roberto Silveira, 1954, Flamengo, Maricá, RJ, CEP 24903-815. ORCID: https://orcid.org/0000-0002-9377-5993. E-mail: acesse o artigo original para visualizar o e-mail