education sciences

MDPI

Systematic Review

Microlearning in Software Engineering Education: A Systematic Review of Initiatives and Curriculum Modernization

Franklin Parrales-Bravo [1,2,3]

[1] Artificial Intelligence Research Group, Universidad Bolivariana del Ecuador, Km 5 ½ vía Durán—Yaguachi, Durán 092405, Ecuador; franklin.parralesb@ug.edu.ec
[2] Grupo de Investigación en Inteligencia Artificial, Facultad de Ciencias Matemáticas y Físicas, Universidad de Guayaquil, Guayaquil 090514, Ecuador
[3] Faculty of Computer Science, Complutense University of Madrid, Av. Séneca, 2, 28040 Madrid, Spain

Abstract

This systematic review maps the landscape of microlearning research within software engineering education, critically examining how this pedagogical approach is being applied to develop the multifaceted competencies required of modern software professionals. Following PRISMA-ScR guidelines, the review synthesized 21 empirical studies from 2015 to 2026, analyzing their pedagogical approaches, technological integrations, curriculum coverage, and evidence of effectiveness. The findings reveal a field marked by creative experimentation yet significant fragmentation: while microlearning effectively engages students and conveys discrete programming and project management knowledge through gamified, mobile, and project-based formats, its application remains narrowly concentrated on introductory coding, leaving advanced competencies such as software architecture, requirements engineering, and testing strategies virtually unexplored. The review further exposes critical gaps in the evidence base, including the absence of longitudinal and transfer studies, the conflation of platform engagement with learning, and methodologically fragile claims of effectiveness. Enthusiasm for microcredentials and AI-personalized learning considerably outstrips empirical support, with implemented systems relying on rule-based logic rather than adaptive intelligence and credentialing frameworks lacking validation of employer recognition or employment outcomes. This review concludes that while microlearning holds genuine potential for just-in-time skill development in a rapidly evolving discipline, its role in software engineering education must be strategic and supplemental rather than comprehensive. The field must urgently move from promotional advocacy toward rigorous, comparative, and longitudinal research that assesses higher-order competencies and authentic professional capability, lest its promise remain unfulfilled.

Keywords: higher education; microcredentials; software engineering; pedagogical approaches; curriculum modernization

Check for updates

Academic Editor: Sonia M. Gómez Puente

Received: 19 February 2026
Revised: 9 March 2026
Accepted: 18 March 2026
Published: 20 March 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The concept of learning itself is undergoing a profound transformation, moving away from the monolithic structures of the past towards something more fluid, personalized, and immediate. At the heart of this shift is micro-learning, a pedagogical approach that delivers content in small, very specific bursts (Dixit et al., 2021). Learners are no longer asked to sit for hours-long lectures, but are instead engaged for three to ten minutes with a focused lesson, a single concept, or a practical task. The fundamental idea is simple yet powerful: the human brain is wired to absorb information more effectively in small,

manageable chunks (Boumalek et al., 2025). This method respects the learner's time and cognitive load, enabling focused concentration and significantly improving knowledge retention by avoiding the fatigue of information saturation (Monib et al., 2025). It is a direct response to how we now consume information in the digital age: quickly, contextually, and on-demand (Alias & Razak, 2025).

The importance of this model, particularly for a field as dynamic as software engineering, cannot be overstated. Software engineering is not a static discipline; it is a living, breathing entity that evolves with every new framework (Hridoy et al., 2023), language update (Murphy-Hill & Grossman, 2014), security vulnerability (Sinhabahu, 2020), and cloud service innovation (Kesavan, 2025). A curriculum learned in a four-year degree program can be partially obsolete by the time of graduation (Tunguz, 2025). In this environment, the ability to learn continuously is not just a soft skill (Schultheiss & Backes-Gellner, 2023); it is a core professional competency. Micro-learning aligns perfectly with this reality. It allows a developer to quickly learn a new Git workflow in the morning and apply it in the afternoon, or to understand a specific design pattern just in time to solve a problem they are facing. It bridges the gap between theoretical knowledge and practical, immediate application, fostering a culture of just-in-time learning that is essential for staying relevant and effective in the tech industry.

Globally, the educational landscape is grappling with a deep-seated crisis of relevance and engagement. Traditional educational institutions, with their rigid curricula and semester-based schedules, often struggle to keep pace with the velocity of change in sectors like technology (Alshawi, 2023). We see a rise in alternative credentialing—micro-credentials, nanodegrees, and professional certificates offered by industry giants like Google, Microsoft, and AWS, or platforms like Coursera and edX. These programs are, in essence, structured forms of micro-learning, designed to certify specific, job-ready skills (Oliver, 2022). This global shift reflects a change in learner behavior; people are turning to YouTube for a quick tutorial on a coding problem, browsing through technical blogs, or participating in short, focused coding challenges on platforms like LeetCode (LaPlante, 2024). The global education market is being disaggregated, with knowledge being pulled by learners in small pieces rather than being pushed in large, pre-packaged bundles by institutions (Álvarez, 2025; Kohler et al., 2021).

This is where the true utility of micro-learning becomes a strategic tool for modernizing the software engineering curriculum in universities (Stankovski, 2025). Instead of viewing it as a competitor, curriculum designers can integrate it as a core pedagogical and structural component (Orman et al., 2023). The rigid, semester-long course can be reimagined as a learning journey composed of smaller, skill-based modules (Urem et al., 2025). For instance, a course on "Web Development" would not be a single narrative, but a series of micro-units: "HTML5 Semantics", "CSS Flexbox Layout", "JavaScript Promises and Async/Await", "Introduction to React Hooks", "Consuming REST APIs", and "Basic Web Security Headers" (Herath, 2024; Mangena et al., n.d.). Each unit has a clear objective and can be updated independently. When a new version of React is released, the university doesn't need to redesign the entire course; it simply updates that specific micro-module. This agility is the key to keeping the curriculum perpetually current.

Furthermore, micro-learning provides the perfect framework for integrating industry certifications into the academic path (Garcia & Stefany, 2023). A curriculum could be mapped to the objectives of a recognized certification (Catenazzi et al., 2025), like an AWS Certified Developer or a Meta Backend Developer certificate. Each unit of the university course would align with a section of the certification exam's study guide (Mangena et al., n.d.). Students could build their academic credits while simultaneously preparing for industry-validated credentials, making them incredibly competitive upon graduation (Kee-

ley, 2025). This blend of academic theory and practical, certifiable skill creates a powerful value proposition for the graduate.

The purpose of this systematic review is to synthesize and critically evaluate the existing body of empirical research on microlearning initiatives in software engineering education, with particular emphasis on how these approaches contribute to curriculum modernization and the development of industry-relevant competencies. By examining studies published between 2015 and 2026 that document the design, implementation, and evaluation of microlearning interventions across diverse educational contexts—ranging from vocational training programs to undergraduate courses and graduate-level professional development—this review seeks to identify prevalent pedagogical strategies, technological frameworks, and instructional design models that characterize successful microlearning implementations.

Furthermore, this review aims to assess the evidence regarding microlearning's effectiveness in enhancing student learning outcomes, engagement, motivation, and knowledge retention, while also exploring the integration of emerging technologies such as artificial intelligence, learning analytics, and gamification to personalize and enrich the learning experience. A central focus is placed on understanding how microlearning facilitates the alignment of academic curricula with industry demands through microcredentials, modular skill-based certification, and just-in-time learning paradigms that address the rapidly evolving nature of software engineering practice. Through this comprehensive analysis, the review aspires to provide educators, curriculum designers, and researchers with actionable insights into the opportunities, challenges, and best practices associated with leveraging microlearning as a strategic tool for modernizing software engineering education and better preparing students for the demands of the contemporary technology workforce.

This systematic review addresses the following research questions (RQs):

- RQ1: What technological approaches and pedagogical strategies characterize microlearning implementations in software engineering education?
- RQ2: Which software engineering competencies and knowledge areas are being taught through microlearning approaches?
- RQ3: How does microlearning impact student learning outcomes, engagement, motivation, and knowledge retention in software engineering courses?
- RQ4: What role do microcredentials and industry certifications play in microlearning-based curriculum modernization?
- RQ5: How are emerging technologies (AI, machine learning, learning analytics) being integrated into microlearning platforms for software engineering education?
- RQ6: What design frameworks and instructional models guide the development of microlearning content for software engineering?
- RQ7: How do microlearning initiatives address the integration of technical and professional competencies (soft skills) in software engineering education?
- RQ8: What challenges, limitations, and barriers exist in implementing microlearning approaches across diverse educational contexts (vocational, undergraduate, graduate, professional development)?
- RQ9: How do different student populations (vocational students, undergraduates, professionals) respond to microlearning interventions in software engineering education?

These questions systematically guide the evaluation of the selected studies, providing a structured framework for analyzing the diverse implementations, outcomes, and contextual factors reported in the literature.

The examination of microlearning in software engineering education is theoretically grounded in three complementary adult learning frameworks that illuminate both its potential and its limitations. First, Knowles' andragogical model (Knowles, 1984) provides foundational relevance, as microlearning's emphasis on self-directed, just-in-time, problem-centered units aligns directly with adult learners' preferences for immediate applicability and autonomy—particularly salient for the vocational students and industry professionals represented in this corpus. Second, Zimmerman's self-regulated learning theory (Zimmerman, 2002) offers analytical purchase on how learners navigate microlearning's fragmented structure, requiring metacognitive skills to set goals, monitor comprehension, and integrate discrete episodes into coherent understanding—processes that may be underdeveloped in novice learners. Third, Lave and Wenger's situated cognition (Lave & Wenger, 1991) provides a critical lens for examining whether microlearning's decontextualized knowledge chunks can support the development of authentic professional practice, or whether software engineering competency fundamentally requires participation in communities of practice that microlearning architectures rarely replicate. Together, these frameworks inform my analysis of why microlearning may effectively support discrete skill acquisition while potentially falling short in developing the integrated, context-sensitive judgment that characterizes expert software engineers. These theoretical lenses will inform the critical analysis of implemented interventions in the Discussion section.

2. Related Work

This section provides a comprehensive overview of existing literature reviews on microlearning and microcredentials, analyzing their scope, focus, and principal findings to establish the foundation and distinguish the unique contribution of the present systematic review. To facilitate this comparison, Table 1 synthesizes the most relevant recent reviews, detailing their authors, objectives, key outcomes, and the specific gaps that this review aims to address.

Table 1. Summary of recent literature reviews on microlearning or microcredentials, highlighting scope, focus, and differences with the present review.

Authors	Goal	Main Findings	Differences with this Review
Boumalek et al. (2025)	To conduct a systematic literature review on micro-learning design and micro-course structuring, analyzing existing frameworks and models to identify theories and metrics crucial for developing well-structured micro-courses in formal educational settings.	The review analyzes 56 articles (2019–2023) to reveal strategies for effective micro-learning methodologies. It uncovers a "rich tapestry of strategies empowering both learners and teachers" while also addressing inherent limitations, underscoring the critical importance of thoughtful design in creating effective learning environments for technology-driven learners.	This review focuses on general micro-learning design principles and micro-course structuring across formal educational settings, while my review specifically examines microlearning within software engineering education, analyzing its role in developing competencies across the software development process.
Leong et al. (2021)	To review the trends of microlearning by analyzing publication trends (Scopus, 2006–2019) and Internet search trends (Google Trends) to serve as a reference for promoting, designing, and using microlearning.	Microlearning is a relatively new and emerging global topic. Publications increased 47× from 2006–2019, with conference proceedings as the primary source type, reflecting its novelty. It is strongly connected to e-learning/mobile learning, with higher education as the most mentioned level and language learning as a key subject. Increasing publication trends align with growing Internet searches, indicating rising practical demand.	This review focuses on macro-level publication and search trends of microlearning across all fields, while my review specifically examines the application, pedagogical approaches, and curriculum coverage of microlearning within software engineering education.

Table 1. *Cont.*

Authors	Goal	Main Findings	Differences with this Review
Lee (2023)	To conduct a systematic literature review to identify the trends, impacts, and challenges of mobile microlearning (MML) research from 2015 to 2020.	Mixed-method approaches are a growing trend in MML studies. MML is widely applied in both formal and informal settings, particularly in just-in-time workplace learning. It enhances learner achievement, motivation, skills, and knowledge retention. Key challenges include the need for meaningful microcontent design for small screens and usability issues.	This review focuses specifically on mobile microlearning across general educational and workplace contexts, while my review examines microlearning within software engineering education, analyzing its role in developing competencies across the software development process, with mobile delivery being one of several technological approaches considered.
Moore et al. (2024)	To systematically review the empirical literature on mobile-based microlearning in adult learning contexts (workplace and higher education) published between 2015 and 2021, focusing on implementation contexts and key findings.	Mobile-based microlearning is implemented across diverse disciplines (e.g., medicine, IT, physics, music) and shows positive impacts on learning outcomes and student perceptions. Key themes include effectiveness (improved knowledge, skills, engagement) and design principles (e.g., task-specific, concise, contextual). The approach effectively provides just-in-time, flexible learning that complements existing instruction.	This review focuses specifically on mobile-based microlearning across general adult learning contexts, while my review examines microlearning within software engineering education, analyzing its role in developing competencies across the software development process, including but not limited to mobile delivery.
Shatte and Teague (2020)	To synthesize the literature on technology-based microlearning in higher education, mapping its application and impact on both objective (e.g., academic performance) and subjective (e.g., motivation) student outcomes.	Microlearning, particularly micro-lectures, improves academic performance, participation, and student motivation. Benefits are seen across disciplines, but research is predominantly small-scale and focused on computer science and language learning, with emerging areas like personalization showing promise.	This review broadly examines technology-based microlearning across all higher education disciplines, while my review specifically focuses on its application within software engineering education, analyzing its role in developing competencies across the software development process.
Rahutomo et al. (2023)	To identify and analyze the fundamental components of microlearning from existing research and implementations in e-learning platforms to address the biased definitions of the concept.	The study found 15 fundamental components, with "Content's variety" being the most discussed, followed by "Evaluation and feedback" and "Short learning duration". It concluded that the focus of recent research is on transforming learning content, while advanced technologies like chatbots and learning analytics are the least discussed but are required for specialized applications.	This review focuses on the general, fundamental components of microlearning across all e-learning contexts, while my review specifically examines the application, pedagogical approaches, and coverage of software engineering competencies within the software development process.
Monib et al. (2025)	To conduct a systematic review on the effectiveness of microlearning on learning outcomes across higher education and corporate sectors, addressing gaps in its definition, didactics, and instructional design.	Microlearning positively impacts cognitive (knowledge, skills, competence), behavioural (engagement, collaboration), and affective (motivation, satisfaction, self-efficacy) outcomes. The study proposes a novel framework integrating individual/situational/subject differences, guiding principles (e.g., bite-sized content, specific objectives), and Bloom's Taxonomy-based outcomes.	This review focuses on microlearning's effects on learning outcomes across general higher education and corporate contexts, while my review specifically examines its application within software engineering education, analyzing its role in developing competencies across the software development process.
Silva et al. (2025)	To systematically review the literature on the contributions of microlearning in basic education, identifying the most used digital technologies, relevant learning theories, and the role of social technologies from a Science, Technology, Society, and Environment (STSE) perspective.	Microlearning, integrated with online platforms, mobile apps, and short videos, enhances student motivation, performance, and interaction. It is grounded in theories like Self-Determination Theory, Constructionism, and Constructivism, and promotes active, meaningful learning. From an STSE perspective, social technologies foster autonomy, social interaction, and ethical-environmental awareness.	This review focuses on the application of microlearning in basic education (K-12), while my review specifically examines its use within software engineering education at the higher education and professional level, analyzing its role in developing competencies across the software development process.

Table 1. *Cont.*

Authors	Goal	Main Findings	Differences with this Review
Λυγούρα (2024)	To conduct a systematic literature review to define and compare digital (micro-)credentials for learning, analyzing definitions of micro-credentials, digital badges, digital credentials, and open badges from academic and grey literature (2013–2023).	The review analyzed 281 publications, locating 230 definitions (139 original, 91 adopted). Key elements of micro-credentials include outcomes/assessment and duration, while digital badges emphasize certification and security. The term "micro-credential" lacks a standard definition, with variations even in European Commission working definitions. The study highlights the need for a common vocabulary to enable recognition across educational and labor market contexts.	This review focuses on defining and comparing digital credential terminology across all learning contexts, while my review specifically examines microlearning as a pedagogical approach within software engineering education, analyzing its role in developing competencies across the software development process.
Al-Bhloly et al. (2024)	To provide a comprehensive survey of micro-learning (ML), reviewing its importance, principles, dimensions, characteristics, techniques, platforms, applications, and future trends to bridge the research gap and highlight its value in modern education.	The paper identifies 27 key characteristics of micro-learning (e.g., effective, efficient, motivating, interactive) and compares it to traditional and e-learning. It details the technological architecture for ML and lists numerous applications across diverse fields like language learning, professional development, healthcare, and programming, concluding that ML is a highly flexible and effective modern educational trend.	This survey provides a broad, general overview of micro-learning's characteristics, technologies, and applications across all fields. My review, in contrast, has a specific focus on the application of microlearning within software engineering education, analyzing its role in developing competencies across the software development process.
Alias and Razak (2025)	To conduct a systematic literature review to identify effective pedagogical practices of microlearning in teaching and learning in higher education between 2014 and 2023.	The analysis revealed two key themes for effective microlearning: the design of microlearning content (small, bite-sized, focused, and relevant) and the instructional flow of the lesson (clear objectives, varied media, peer interaction, practical tasks, and instant feedback).	This review focuses on general pedagogical design principles for microlearning in higher education, while my review specifically examines the application, technological integration, and coverage of software engineering competencies within the software development process.

Table 1 synthesizes a diverse body of recent literature reviews, each with distinct goals that collectively map the broader landscape of microlearning research from which the present study emerges. Several reviews adopt a wide-angle lens, seeking to define the fundamental characteristics, technological architectures, and cross-disciplinary applications of microlearning as a modern educational trend (Al-Bhloly et al., 2024; Rahutomo et al., 2023). Others focus on mapping publication trends and the growth of the field, establishing microlearning as an emerging global topic with strong connections to e-learning and mobile learning (Leong et al., 2021). A significant cluster of reviews concentrates on the pedagogical "how"—investigating effective design principles, instructional frameworks, and the structuring of micro-courses to enhance learning in formal educational settings, higher education, and corporate contexts (Alias & Razak, 2025; Boumalek et al., 2025; Monib et al., 2025). This focus on design is often coupled with an examination of impact, with studies like those by Monib et al. (2025) and Silva et al. (2025) systematically evaluating the effects of microlearning on cognitive, behavioral, and affective outcomes, such as knowledge retention, student motivation, and engagement.

A further layer of specialization is evident in reviews that narrow their scope by delivery modality or educational level. For instance, Lee (2023) and Moore et al. (2024) specifically examine mobile-based microlearning in adult learning contexts, highlighting its role in just-in-time workplace training and its positive impact on learner achievement. In contrast, Silva et al. (2025) focus on the application of microlearning in basic education (K-12), exploring its integration with digital technologies and its grounding in learning theories. Finally, the related but distinct field of microcredentials is addressed by Λυγούρα (2024), whose goal is to bring conceptual clarity to the varied and often inconsistent definitions of digital credentials, a necessary step for enabling their recognition across educational and labor market contexts. Collectively, these reviews have successfully charted

the macro-trends, established foundational design principles, and provided preliminary evidence for microlearning's effectiveness across a broad spectrum of educational settings.

While these prior reviews provide an essential foundation, they exhibit a critical gap that the present study is uniquely positioned to address: the specific application of microlearning within the discipline of software engineering. As detailed in the "Differences with this Review" column of Table 1, the existing literature operates at a level of generality that overlooks the unique pedagogical demands and competency landscape of software engineering education. For example, while Boumalek et al. (2025) explore general micro-learning design principles for formal education, and Monib et al. (2025) examine its effectiveness across higher education and corporate sectors, neither study delves into how these principles apply to teaching and learning activities like requirements analysis, architectural design, debugging, or software testing. Similarly, reviews focused on mobile delivery (Lee, 2023; Moore et al., 2024) or basic education (Silva et al., 2025) treat software engineering as merely one application context among many, rather than as a complex professional field with its own epistemological challenges. Even the broad survey by Al-Bhloly et al. (2024), which mentions programming as an application area, does so only superficially, listing it without any deep analysis of how microlearning can support the development of competencies across the entire software development lifecycle.

The critical distinction, therefore, lies in the level of analysis and disciplinary focus. Prior reviews have built a robust understanding of what microlearning is and that it can work in general terms. However, they have not systematically investigated how it can be purposefully designed and implemented to cultivate the specific, multi-faceted competencies required of a software engineer—from technical coding skills to abstract architectural thinking and collaborative project management. By concentrating on the intersection of microlearning and the software development process, this review moves beyond generic claims of effectiveness to provide a targeted analysis of curriculum coverage, technological integration, and pedagogical approaches unique to the field. It interrogates not just whether microlearning engages students, but whether it can effectively teach them to design, build, and maintain complex software systems. In doing so, it addresses a significant lacuna in the literature, offering a critical, domain-specific perspective that complements and extends the broader, foundational work summarized in Table 1.

3. Methodology

This literature review was conducted using a systematic approach to identify peer-reviewed journal articles published between 2015 and 2026 that focus on the application of microlearning and microcredentials within software engineering education. The review specifically examines studies investigating pedagogical approaches, technological integration, and competency development across the software development process, including programming, software design, project management, testing, requirements engineering, architecture, and professional skills.

It is important to mention that the final report followed the PRISMA-ScR (Preferred Reporting Items for Systematic reviews and Meta-Analyses extension for Scoping Reviews) guidelines (Tricco et al., 2018). The protocol was pre-registered in the Open Science Framework (OSF) portal (Available in https://osf.io/ck97n/overview?view_only=c9d311e1b6a547a0b46c456b22799723) (accessed on 19 February 2026).

3.1. Databases and Search Strategy

Table 2 presents the company names and addresses (city, country) of the databases and software used. The search strategy was designed to systematically identify studies investigating the application of microlearning and microcredentials within software engineering education. A comprehensive literature search was conducted across multiple academic databases, including Scopus, and IEEE Xplore, following best practices established in prior systematic reviews. In addition, we have considered Google Scholar because it provides a comprehensive and broad-reaching search of scholarly literature across multiple disciplines and sources, including peer-reviewed papers, theses, books, and conference proceedings, which helps to capture a wider range of potentially relevant studies that may not be indexed in more specialized databases.

Table 2. Details of scientific databases considered in this review.

Database	Company/Organization	Address	Software
Scopus	Elsevier	Radarweg 29, Amsterdam, Netherlands (HQ); 230 Park Ave, New York, NY, USA	https://scopus.com/ (accessed on 23 December 2025)
IEEE Xplore	Institute of Electrical and Electronics Engineers (IEEE)	445 Hoes Lane, Piscataway, NJ, USA	https://ieeexplore.ieee.org/ (accessed on 23 December 2025)
Google Scholar	Google LLC	1600 Amphitheatre Parkway, Mountain View, CA, USA	https://scholar.google.com/ (accessed on 23 December 2025)

The search strategy, presented in Table 3, employed a combination of keywords related to the core concepts, specifically: ("microlearn" OR "micro-learn" OR "bite-sized" OR "microcontent") AND ("software engineering" OR "programming" OR "coding" OR "computer science education") to capture microlearning studies, complemented by a parallel string for microcredentials: ("microcredential" OR "micro-credential" OR "digital badge" OR "micro-certification") AND ("software engineering" OR "programming" OR "computer science") during the period 2015–2026 (until 13 February 2026). Searches were limited to peer-reviewed articles and conference proceedings published in English or Spanish. This structured approach ensured a rigorous and reproducible identification of relevant literature, forming the foundation for the subsequent screening and synthesis phases.

Table 3. Number of studies retrieved per diagnostic approach and database using the systematic search strategy.

Database	Search Strategy	Results
Scopus	TITLE-ABS-KEY (("microlearn" OR "micro-learn" OR "micro content" OR "bite-sized") AND ("software engineering" OR "programming" OR "coding" OR "computer science education")) OR TITLE-ABS-KEY (("microcredential" OR "micro-credential" OR "digital badge" OR "micro-certif") AND ("software engineering" OR "programming" OR "coding" OR "computer science"))	88
IEEE Xplore	("microlearning" OR "micro-learning" OR "microcontent" OR "bite-sized learning") AND ("software engineering education" OR "programming education" OR "computer science education") OR ("microcredentials" OR "micro-credentials" OR "digital badges") AND ("software engineering" OR "programming" OR "computer science")	25
Google Scholar	(microlearning OR micro-learning OR "bite-sized learning") AND ("software engineering" OR programming) OR (microcredentials OR "digital badges") AND ("software engineering" OR "computer science")	5710
Total		5823

3.2. Inclusion Criteria

For the present study, we selected articles between 2015 and 2026 (until 19 February 2026) that met the criterion of Table 4.

Table 4. Inclusion Criteria.

Criterion	Description	Rationale
Population/Context	Studies focusing on software engineering education, computer science education, or programming instruction at any level (vocational high school, undergraduate, graduate, or professional development)	Aligns with the review's focus on software engineering curriculum modernization
Intervention	Studies implementing, evaluating, or proposing microlearning approaches, micro-credentials, or related pedagogical strategies (e.g., gamified microlearning, mobile microlearning, automated assessment with micro-content)	Directly addresses the core research questions
Content Domain	Studies covering software engineering competencies across the development process, including programming, software design, project management, testing, requirements engineering, architecture, and soft skills for software professionals	Ensures comprehensive coverage of the software engineering curriculum.
Outcomes	Studies reporting on learning outcomes, engagement metrics, competency development, implementation challenges, or effectiveness evidence	Enables synthesis of findings across the corpus
Publication Type	Peer-reviewed journal articles and conference proceedings	Ensures scholarly rigor and quality
Language	English or Spanish	Aligns with the authors' linguistic capabilities and the geographical scope of included studies (e.g., Latin American contexts in Table 8)
Time Frame	No initial date restrictions, but searches limited to studies published up to the search date	Maximizes breadth while maintaining currency

3.3. Exclusion Criteria

In this study, we discarded the articles that contained any of the criterion of Table 5.

Table 5. Exclusion Criteria.

Criterion	Description	Rationale
Irrelevant Population	Studies focusing exclusively on non-software engineering disciplines (e.g., pure chemistry education; medical education; humanities) without explicit connection to software engineering or programming	Maintains disciplinary focus
Irrelevant Intervention	Studies addressing general e-learning, traditional instruction, or other pedagogical approaches without explicit microlearning or microcredential components	Ensures relevance to the research questions
Conceptual Only	Papers presenting purely conceptual frameworks, position papers, or opinion pieces without empirical data or implementation evidence	Focuses on evidence-based findings, though conceptual papers may be retained for background discussion
Publication Type	Editorials, prefaces, book reviews, newsletters, non-peer-reviewed magazine articles, and preprints	Ensures quality control
Language	Studies published in languages other than English or Spanish	Aligns with review team's language proficiency
Duplicate Publications	Duplicate reports of the same study across multiple venues	Prevents double-counting of evidence
Full-Text Unavailable	Studies for which full text cannot be obtained through institutional access or interlibrary loan	Enables thorough analysis

3.4. Quality Appraisal and Data Extraction Process

To ensure methodological rigor and transparency in synthesizing the evidence, two independent reviewers conducted quality appraisal and data extraction for all studies meeting inclusion criteria. Given the scoping review's aim to map the breadth of microlearning research in software engineering education while also examining evidence of effectiveness, a dual approach to quality assessment was adopted.

Each of the 21 included studies underwent critical appraisal using established criteria appropriate to study design. For experimental and quasi-experimental studies, we applied an adapted version of the MERSQI: Medical Education Research Study Quality Instrument (Jaros & Beck Dallaghan, 2024), available in Appendix A, which evaluates

study design, sampling, data type, validity evidence, data analysis, and outcomes. For qualitative studies, we employed the Critical Appraisal Skills Programme (CASP) Qualitative Checklist (CASP, 2018). Descriptive studies and case reports were assessed using criteria adapted from the Joanna Briggs Institute Checklist for Case Reports. Two reviewers (initials) independently evaluated each study, with disagreements resolved through consensus discussion or consultation with a third reviewer (initials). While quality scores were not used as exclusion criteria—consistent with the scoping review's objective to comprehensively map the literature—the appraisal process served to characterize the evidentiary strength of included studies and inform the critical interpretation of findings presented in the Discussion section. This approach allows readers to contextualize claims of effectiveness against methodological robustness, acknowledging that studies with weaker designs (e.g., single-group pre-post tests without controls) provide less compelling evidence than randomized controlled trials or rigorous quasi-experimental designs.

Data extraction was conducted independently by two reviewers using the form available in Appendix B, which is a standardized extraction form developed iteratively based on the nine research questions guiding this review. The form captured seven categories of information: bibliographic information, study context, intervention characteristics, software engineering competencies, outcome measures, key findings, and methodological limitations. Prior to full extraction, both reviewers independently piloted the form on three randomly selected studies, achieving 90.1% agreement on initial extractions. Discrepancies were discussed, and the form was refined for clarity. Subsequently, all 21 studies were extracted independently by both reviewers.

Inter-rater reliability was calculated using Cohen's κ for all extraction categories. Two independent reviewers achieved substantial to almost perfect agreement across all categories, with an overall κ of 0.899 (95% CI [0.877, 0.921]), indicating almost perfect agreement. Category-specific κ values ranged from 0.58 to 0.94, with percentage agreement ranging from 71.42% to 100%. Table 6 presents the top 10 categories by agreement.

Table 6. Top 10 Categories by Agreement.

Category Label	Section	Kappa (κ)	Agreement (%)
country	A: Bibliographic	0.911	100.0
language	A: Bibliographic	0.920	100.0
outcome timing	E: Outcomes	0.936	100.0
publication type	A: Bibliographic	0.933	100.0
retention measured	E: Outcomes	0.920	100.0
sample size category	B: Study Context	0.947	100.0
delivery mode	C: Intervention	0.925	95.2
effect size reported	F: Findings	0.935	95.2
funding source	A: Bibliographic	0.930	95.2
primary pedagogy	C: Intervention	0.945	95.2

Moreover, Table 7 presents the section-level reliability. By section, agreement was strongest for "A: Bibliographic" (κ = 0.92) and weakest for "D: Competencies" (κ = 0.81), reflecting the greater interpretive judgment required when classifying software engineering competencies. All discrepancies were resolved through consensus discussion between the two reviewers, with unresolved cases adjudicated by a third reviewer. The final extracted data represent the agreed-upon interpretation and form the foundation for the thematic analysis presented in Table 8 and the Discussion section (Section 5).

Table 7. Section-level reliability.

Section	Kappa (κ)	Agreement (%)	n
A: Bibliographic	0.920	98.8	84
B: Study Context	0.880	86.7	105
C: Intervention	0.850	91.8	147
D: Competencies	0.810	86.7	105
E: Outcomes	0.860	91.3	126
F: Findings	0.870	89.3	84
G: Limitations	0.840	84.1	63

3.5. Data Extraction and Synthesis

Figure 1 presents the PRISMA flowchart documenting the study selection process according to the PRISMA guidelines (Tricco et al., 2018). A total of 5823 records were initially identified through database searching: Scopus (n = 88), IEEE Xplore (n = 25), and Google Scholar (n = 5710). After duplicate removal, 4856 records proceeded to title and abstract screening. Of these, 4782 records were excluded based on irrelevance to the research questions, leaving 74 full-text articles for detailed eligibility assessment. During full-text review, 53 articles were excluded for the following reasons: irrelevant population (n = 12), irrelevant intervention (n = 15), purely conceptual papers without empirical data (n = 18), languages other than English or Spanish (n = 5), and full-text unavailability (n = 3). Consequently, 21 studies met all inclusion criteria and were included in the final synthesis, as detailed in Table 8. This selection process ensures a rigorous and reproducible foundation for the systematic review's findings.

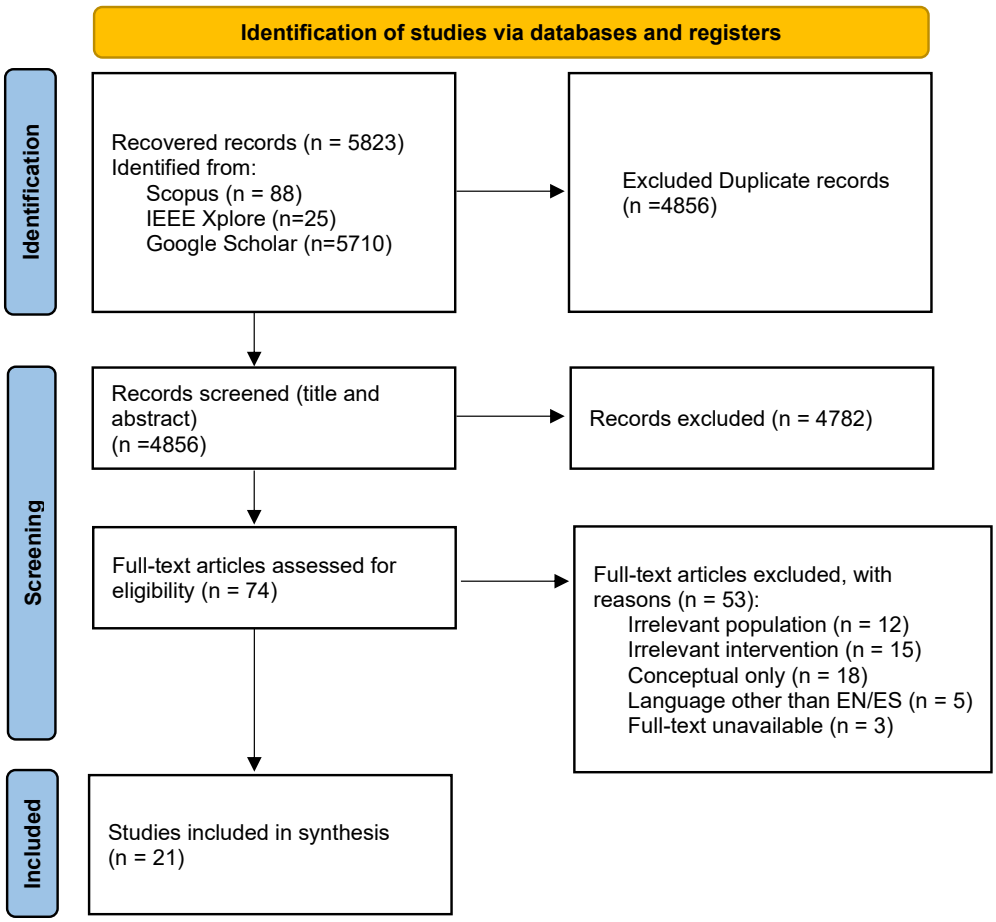

Figure 1. Flowchart of study selection according to the PRISMA guidelines.

Table 8. Studies related to initiatives of microlearning in software engineering education. The bold words serve as a quick visual indicator for the reader to immediately identify the core area(s) of the curriculum being addressed by that particular study.

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Wira et al. (2026)	To develop project-based learning media delivered through e-learning with a microlearning approach to improve learning outcomes in Object-Oriented Programming.	Vocational high school students (11th grade) in the Software Engineering (RPL) program at SMKN 1 Sukawati.	Project-Based Learning, E-learning (Moodle LMS), and Microlearning.	Object-Oriented Programming (OOP).	Object-Oriented Programming concepts such as encapsulation, inheritance, and polymorphism, delivered via videos, animations, interactive modules, quizzes, and small-scale projects.	**Coding (Back End)**: The course focuses on programming (coding) skills and understanding OOP concepts, which are fundamental to back-end development.	The developed media showed very high validity and practicality. Its implementation effectively improved student learning outcomes (N-Gain 0.52/moderate category) and helped students achieve the minimum mastery criterion (KKM) score of 75.
Hadiyana et al. (2026)	To design and engineer an AI-capable, mobile-based microlearning application that integrates short-form videos, learning analytics, and a rule-based recommendation system to support soft skill development in vocational higher education.	Vocational higher education students (in study programs related to the context).	Microlearning, Mobile Learning (m-learning), AI-based rule for personalization.	Soft Skills (for vocational contexts).	Soft skills modules for communication, collaboration, problem-solving, and time management. Each 60–180 s module includes a single learning objective, visual cues, a short example, a 2–3 item quiz, and a reflection prompt.	**Conception, Architecture Design, Software Design, Coding (Front End & Back End)**: The paper explicitly discusses the software engineering of the mobile application, including system architecture, performance, data instrumentation (learning analytics), and UI design, making it the core subject of the study.	The developed prototype (v0 with 10 modules) showed high content validity and good reliability. Pilot tests showed module completion rates of 66–78%, with average quiz scores of 72–81/100. System simulations indicated the architecture is ready for a medium-scale class ($\leq$300 concurrent users).
Wigati et al. (2024)	To analyze the learning needs of vocational high school students and explore the potential of mobile microlearning as a teaching material to improve their competence in software engineering.	2nd-grade vocational high school students (specifically in software engineering expertise) at SMKN 5 Bekasi.	Microlearning, Mobile Learning (m-learning), Collaborative Learning.	Creative Products and Entrepreneurship (for the Software Engineering expertise competence).	Not specified in detail, but focuses on material for Creative Products and Entrepreneurship that is currently perceived by students as "too many and complicated." The goal is to present it concisely.	**Requirement Engineering/Analysis**: The paper is exclusively a needs analysis for a future learning media development. It identifies the "what" and "why" for a mobile microlearning solution, which is the first step in a software project's lifecycle.	Students need concise, engaging, and easy-to-understand teaching materials accessible via mobile devices. They prefer project-based group assignments. Current materials are too numerous and complicated, and media is not varied. Mobile microlearning with collaborative assignments is identified as a potential solution.
Skalka (2018)	To describe a conceptual framework for teaching programming languages that combines microlearning, automated code validation, competitions, and community learning, with a focus on data processing methods for automated content development.	University students learning programming languages (the framework is designed for higher education).	Microlearning, Automated Assessment (Code Validation), Community Learning, Gamification (competitions).	Introductory Programming (Language agnostic, but examples imply Java/Python).	Educational content divided into micro-content (explanations) and micro-activities (fixation), followed by programming tasks. Content includes text, source code examples, and diagrams.	**Coding, Testing**: The framework focuses on writing source code and validating its correctness through automated assessment (I/O-based and acceptance testing).	The paper presents a conceptual framework, not an empirical study with findings. It proposes automated content development via text summarization and automated code validation (using VPL) as key elements to improve efficiency in creating microlearning content and providing immediate feedback to learners.

Table 8. *Cont.*

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Gasca-Hurtado et al. (2024)	To evaluate the motivation and learning outcomes of a microlearning strategy, supported by a mobile application called "Mr. Lottery," in the specific context of Software Project Management using the Scrum framework.	23 undergraduate students (7th semester) enrolled in a Software Engineering course at a Latin American higher education institution.	Microlearning, Gamification (via a lottery-style game), Mobile Learning.	Software Project Management (with a focus on the Scrum framework).	Concepts related to the Scrum framework, including its definition, roles, events, artifacts, and the rules that bind them, as defined in The Scrum Guide.	**Conception, Project Management**: The paper focuses on teaching the Scrum framework, which is a methodology for managing software projects. This covers the conception and management phases of a project, rather than technical coding or testing.	The microlearning strategy was positively received. Statistically significant differences were found in motivation, with students' mean scores for "Motivation to learn in the specific context" and "Stimulus for learning" being higher than the theoretical average. While posttest scores on a Scrum certification exam increased, the improvement was not statistically significant.
Suciu (2022)	To analyze the benefits of combining macro- and microlearning with gamification elements in an academic context and to understand how these methods help computer science students comprehend and expand their knowledge of project management.	Computer science students (undergraduate and master's) enrolled in "Software Project Management" and "Agile Software Development" courses at Babes-Bolyai University, Romania.	Microlearning, Gamification, E-learning platform (Code of Talent).	Software Project Management, Agile Software Development.	Frameworks, methodologies, processes, steps, practices, and tools specific to software project development, including both hard skills (project administration) and soft skills (team coordination, facilitation, communication).	**Project Management**: The paper explicitly focuses on teaching project management concepts, including traditional and agile methodologies, which is a cross-cutting concern in all software processes but specifically covered here as the main subject.	Implementing microlearning and gamification increased student engagement, performance, and knowledge retention. Students felt more motivated to participate in online discussions. The use of a dedicated platform led to a significant increase in course feedback rates (from ~3–14% to ~59–72%).
Correa Botero (2015)	To improve the learning of Web Application Frameworks (WAFs) for novice developers by creating a technique that guides their learning based on their specific concerns, using micro-learning and examples.	Novice WAF learners (evaluated with 15 undergraduate students in Systems and Informatics Engineering).	Micro-learning, Separation of Concerns, Example-based learning.	Learning to use Web Application Frameworks (specifically using Codeigniter in the case study).	How to use a WAF (Codeigniter) to develop web applications, focusing on components like Controllers, Models, Views, Route Manager, Error Handler, Database Class, etc.	**Architecture Design, Software Design, Coding (Back End)**: The work focuses on understanding and using a framework's architecture and components to design and code the back-end logic of web applications.	The new learning technique was effective. Experimental groups using the technique showed a significant time reduction (approx. 40% in the first task), completed more tasks (avg. 2.4–3.4 vs. 1.2), and demonstrated better knowledge acquisition (higher post-test scores) compared to the baseline group using only official documentation.
An (2022)	To develop a microlearning content design model for Software Engineering courses using an R&D (Research and Development) approach based on the ADDIE instructional design framework.	University students taking Software Engineering courses, with consideration for "Generation Z" learner characteristics.	**Microlearning** content development following the AD-DIE model (Analysis, Design, Development, Implementation, Evaluation). The approach focuses on creating bite-sized, self-contained learning units.	A microlearning-based course or module for a Software Engineering curriculum. The specific software engineering topics are not detailed in this proposal paper.	The syllabus/content is not specified, but the development process involves creating a content map, storyboards, and multimedia/video content based on established curriculum documents (Syllabus/SAP/GBPP).	**Software Engineering** (general). The model is designed for the entire software engineering domain, which would encompass **Conception/Analysis**, **Software Design**, **Coding**, **Testing**, and **Maintenance** depending on the specific topic chosen for content development.	This paper presents a theoretical model and development stages. No empirical findings or results from implementation are reported. The paper outlines the planned process for developing, validating, and testing the microlearning content through expert validation and limited/expanded trials.

Table 8. *Cont.*

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Skalka and Drlik (2020)	To analyze the relationship between students' use of automated assessment and microlearning units and their final achievements, and to evaluate these elements as predictors for identifying at-risk students in introductory programming courses.	First-year Computer Science students (aged 18–24) at Constantine the Philosopher University in Nitra, Slovakia, across four academic years (N > 300).	Automated Assessment (via Virtual Programming Lab), Microlearning, E-learning (LMS Moodle).	Introductory Programming (Java).	Introduction to programming (variables, loops, arrays, files, etc.), Object-Oriented Programming (classes, methods, inheritance, polymorphism), and GUI development.	**Coding (Back End), Testing, Debugging**: The course focuses on writing, debugging, and testing code. Automated assessment is used to evaluate the correctness of submitted source code, directly supporting the coding and testing phases.	Automated assessment significantly improved student outcomes in introductory topics (H1 accepted) but not in advanced topics (H2 rejected). Microlearning did not lead to significant differences in either part (H3, H4 rejected). A moderate to strong correlation was found between success in interactive activities (VPL, microlearning) and final course outcomes.
Skalka et al. (2020)	To analyze the relationship between students' use of automated assessment and microlearning units and their final achievements, and to evaluate these elements as predictors for identifying at-risk students in introductory programming courses.	First-year Computer Science students (aged 18–24) at Constantine the Philosopher University in Nitra, Slovakia, across four academic years (N > 300).	Automated Assessment (via Virtual Programming Lab), Microlearning, E-learning (LMS Moodle).	Introductory Programming (Java).	Introduction to programming (variables, loops, arrays, files, etc.), Object-Oriented Programming (classes, methods, inheritance, polymorphism), and GUI development.	**Coding (Back End), Testing, Debugging**: The course focuses on writing, debugging, and testing code. Automated assessment is used to evaluate the correctness of submitted source code, directly supporting the coding and testing phases.	Automated assessment significantly improved student outcomes in introductory topics (H1 accepted) but not in advanced topics (H2 rejected). Microlearning did not lead to significant differences in either part (H3, H4 rejected). A moderate to strong correlation was found between success in interactive activities (VPL, microlearning) and final course outcomes.
Correa et al. (2018)	To introduce and validate a new web application framework (WAF) learning technique called CME (Concerns, Micro-learning and Examples) that aims to reduce the learning curve for novice developers.	Novice web developers, specifically undergraduate students in a "System and Informatics Engineering" program.	A blended learning approach combining: **Concerns** (developer's needs), **Micro-learning** (small, focused documentation units), and **Example-based learning** (pre-coded examples).	Not a traditional course, but a structured learning path within a custom tool called the "driving-learning application".	The learning content is structured around a pre-defined list of WAF components and tasks, linked to 29 common web developer concerns. For each concern, learners access "micro-tasks" (specific documentation links) and "meso-tasks" (pre-coded examples). The final "macro-task" is to develop a small web application.	**Software Design** (specifically, designing with frameworks and architectural patterns like MVC), **Coding** (Back End, as the study uses CodeIgniter, a PHP framework), and **Conception/Analysis** (matching application requirements to developer concerns).	The quasi-experiment showed that novice developers using CME completed more development iterations and completed the first iteration significantly faster than those using a traditional cookbook. This provides preliminary evidence that CME makes developers more operational in a reduced period of time.

Table 8. *Cont.*

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Urem et al. (2025)	To explore the concept of micro-credentials and their role in modernizing computer science education by making it more flexible, relevant to the labor market, and focused on student needs.	Students in computer science and related fields, higher education institutions, and employers in the technology sector.	**Micro-credentials** integrated into formal study programs. The approach includes developing in-house micro-credentials, recognizing external industry certifications (e.g., from Oracle, Coursera), and collaborating with industry partners.	Not a specific course, but a model for integrating micro-credentials into computer science curricula. Examples include micro-credentials in SQL, Git, cloud computing, and machine learning developed through projects like CodeIn.	The content is modular and focused on specific, in-demand skills. Examples include Java programming, database design, PL/SQL, version control (Git), cloud computing, machine learning, and environmental data analytics.	The micro-credentials are designed to cover specific, practical skills across the software process, including **Coding** (front-end and back-end), **Database Management**, **Software Design**, **Testing**, and **Deployment/Cloud** (DevOps, cloud computing).	The paper finds that micro-credentials are a valuable tool for modernizing computer science education. They enhance student employability, enable personalized learning paths, allow for rapid integration of new technologies, and strengthen the link between education and industry. The case study of Sibenik University of Applied Sciences demonstrates the feasibility and benefits of this approach through long-term collaboration with industry and participation in European projects.
Wibisono et al. (2024)	To evaluate and compare the user experience (UX) of two mobile-based microlearning applications (Mimo and SoloLearn) for learning to code, using the HEART metrics and the Honeycomb framework.	College students majoring in programming who are learning Python basics.	**Microlearning** (delivering concise, digestible content for brief learning sessions). The apps also utilize **Gamification** to enhance user engagement.	Not a proposed course, but an evaluation of two existing "Learn to Code" (L2C) applications: **Mimo** and **SoloLearn**. Both offer courses in Python, SQL, JavaScript, HTML, and CSS. Mimo also offers career paths for Full-Stack and Front-End Developers.	The evaluated content focuses on learning Python basics. It includes concise lessons, interactive exercises, and practical application through integrated code editors.	**Coding** (specifically front-end and back-end basics through Python, HTML, CSS, and JavaScript), and **Software Design** (indirectly, through the structure and presentation of learning materials).	Mimo provided a more fulfilling user experience, particularly in user retention and perceived credibility. SoloLearn scored higher on the happiness metric but fell short in retention and credibility. Both apps scored high on task success and usability. The study highlights the importance of detailed, practical content for user-perceived value and credibility.
Skalka and Drlík (2018)	To propose and map an educational model for teaching programming, based on a conceptual microlearning framework, to the levels of established educational taxonomies like Bloom's taxonomy.	University students in IT study programs, particularly those at risk of learning failure in introductory and technology-focused programming courses.	**Micro-learning** framework combined with a two-level educational model (Introductory Courses and Technology Courses) mapped to Bloom's taxonomy and utilizing elements of PBL and gamification.	A two-level course structure: 1. Introductory Programming (IP) courses focused on fundamentals. 2. Technology courses focused on specific technologies (web, server, mobile, database, IoT).	The content is structured into micro-learning lessons with activities that progress through Bloom's taxonomy levels. It includes interactive assignments, automated source code assessment, discussions, and gamified elements (leaderboards, badges).	**Conception/Analysis** (problem-solving), **Software Design**, **Coding** (both front-end and back-end, depending on the technology course), and **Testing** (through automated assessment and debugging tasks).	The paper maps framework activities to Bloom's taxonomy phases (Remember to Create) and Fuller's matrix. It concludes that the microlearning model is suitable for modern IT education as it covers cognitive domain levels, supports different student learning paths, and prepares students for lifelong learning by involving them in content creation.

Table 8. *Cont.*

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Rusak (2017)	To develop and verify an algorithmic solution for personalizing learning content and learning paths by recommending micro-learning sessions to students based on their background knowledge, preferences, and performance.	Industrial design engineering students (bachelor level) at Delft University of Technology with diverse programming backgrounds (novice to expert).	**Micro-learning** combined with a hybrid **personalized learning path** recommender system (using collaborative and content-based filtering).	An elective "Software" course on Android app development. The course content was decomposed into 25 micro-learning sessions.	Android software development topics including UML, XML, and Java. Micro-learning sessions consist of short tutorials (video/text/slides) and online tests.	**Conception/Analysis** (designing app concepts), **Software Design** (using UML), and **Coding** (Back End, using Java and XML for Android).	Analysis of prior course data showed the algorithm could cluster students into four profiles (beginner to expert). Approximately 60% of the course content required personalization to suit different learner profiles, while 40% was suitable for all students without adjustment.
Luo and Li (2025)	To investigate the effectiveness of microlearning in developing key soft skills (teamwork, leadership, communication, time management, emotional intelligence) among university students across four academic disciplines.	358 university students from four disciplines in China: Humanities and Arts (HA), Business Studies (BS), Medical Sciences (MS), and Technical and Engineering (TE).	**Microlearning** intervention consisting of six structured, interactive modules delivered over six weeks. The approach included scenario-based learning, gamified elements, and self-reflection exercises.	Not a traditional software course, but a soft skills training program integrated into university curricula. The intervention included six microlearning modules designed to enhance the five target soft skills.	The syllabus focused on developing five soft skills: Teamwork Skills (TWS), Leadership Skills (LS), Communication Skills (CS), Time Management Skills (TMS), and Emotional Intelligence (EI). Content included interactive case studies, real-world scenarios, and reflective exercises.	The paper focuses on soft skills, which are essential for professionals in all fields, including software engineering. These skills are directly applicable to the **Conception/Analysis** (team collaboration), **Software Design** (collaborative design), **Coding** (team communication), and **Testing/Deployment/Maintenance** (project management and leadership) phases of the software process.	The study found that microlearning significantly improved all five soft skills. Leadership modules were most impactful for TE and MS students, while EI training was most valuable for BS students. Communication and Time Management skills improved notably in BS and TE groups. The intervention proved universally effective across all disciplines, with no single group showing significantly higher gains than others, confirming microlearning as a valuable tool for holistic skill development.
Gasca-Hurtado and Gómez-Álvarez (2021)	To present a gamified mobile application (Mr. Lottery) designed to promote microlearning for the assimilation of theoretical concepts in Software Engineering.	University students in Software Engineering-related courses (specifically tested in Business Processes and Project Planning courses).	**Gamification** (using game mechanics like points, leaderboards, and time constraints) integrated with a **Microlearning** strategy. The approach transforms the traditional "Lotería" (Bingo) game into an educational tool.	Not a full course, but a gamified microlearning activity to be used within courses like Business Processes and Project Planning. The application supports learning on any topic configurable by the teacher.	The content is configurable by the teacher and consists of "themes," "concepts," and their corresponding "definitions." The example used in the paper is related to Scrum and Software Engineering concepts.	The paper focuses on teaching foundational **Software Engineering concepts** that are relevant across the entire software process. This includes topics from **Conception/Analysis** (identifying needs) to **Software Design** and **Project Management**.	Functional tests confirmed that the application successfully allows teachers to plan and execute the strategy and allows students to interact and compete. A pilot study in Scrum training showed that using Mr. Lottery helped students assimilate and retain concepts, as evidenced by their performance in a subsequent Scrum Master certification test.
Jwo (2015)	To propose and evaluate a new approach for software engineering education that uses comics and micro-movie design projects to provide a realistic context for learning real-world software development challenges.	Freshmen students majoring in software engineering at Tunghai University, Taiwan.	A project-based learning approach combining **comics** (as case studies) and **micro-movie creation**. Students simulate real-world issues by interviewing industry mentors and scripting/creating short films based on software engineering challenges.	"Software Engineering Introduction," a required course for freshmen in a software engineering program.	The syllabus covers 30 software engineering topics based on real-world stories, including requirements, design, architecture, documentation, quality, lifecycle management, and project management. A comic series based on "The Mythical Man-Month" is used as the core teaching material.	**Conception/Analysis** (requirements, domain knowledge), **Software Design** (architecture, design decisions), **Project Management** (estimation, productivity), **Testing/Quality Assurance**, and **Software Maintenance** (lifecycle management, configuration management).	A survey of 43 students showed positive results. Students agreed that the comics and micro-movie projects helped them explore software engineering issues better (avg. 4.2/5) and that watching peers' movies helped them understand challenges easily (avg. 4.1/5). Students also reported that the course was more fun than others (avg. 3.9/5).

Table 8. *Cont.*

Authors	Main Goal	Population	Approach	Course(s)	Content	Software Process	Findings
Jo et al. (2025)	To investigate the effectiveness of a microlearning-based coding education platform in the Philippines using a modified Information System Success Model (ISSM) and System Usability Scale (SUS).	Undergraduate students from three universities in the Mindanao region of the Philippines who participated in a virtual code camp.	**Microlearning** delivered through an online platform. Students engaged in self-directed, self-paced coding lessons. The study evaluated the platform using the ISSM framework.	A virtual code camp on coding fundamentals, delivered via a microlearning platform. The course was designed for self-paced learning over 2–3 weeks.	The content is focused on introductory coding and programming skills. The specific programming language or topics are not detailed, but the goal is to improve coding knowledge, programming skills, and digital literacy.	**Coding** (general programming fundamentals). The skills taught are foundational for all areas of software development, including **conception/analysis** (problem-solving), **software design**, and **implementation**.	The study found that Information Quality and System Usability significantly and positively influenced User Satisfaction and Net Benefits. User Satisfaction was also a strong predictor of Net Benefits. However, System Quality did not have a significant influence on either User Satisfaction or Net Benefits in this context. The findings validate the ISSM as an effective framework for evaluating microlearning platforms in resource-constrained settings.
Guzmán Rojas (2024)	To understand how universities can adapt their graduate programs through flexible curricula and alternative credentials to close the gap between higher education and labor market demands, using the Master's in Software Engineering (MISO) at Universidad de los Andes as a case study.	Professionals seeking continuous education (students), faculty who design and teach the programs, and employers who value the acquired credentials and skills.	**Microcredentials** (MasterTracks), **MOOCs**, and a **flexible, modular curriculum** delivered through a virtual, asynchronous format in partnership with the Coursera platform.	Master's in Software Engineering (MISO) at Universidad de los Andes. The program offers flexible entry paths, including individual courses, MasterTracks (macro-credentials), and the full master's degree.	The syllabus covers advanced software engineering topics aligned with Industry 4.0, including IoT, AI, cloud computing, multi-device applications, microservices, UX design, agile methodologies (DevOps), continuous integration, and software architecture. The curriculum is organized into four progressive semesters.	The program covers a wide range of the software process: **Conception/Analysis** (UX design, requirements), **Architecture Design**, **Software Design**, **Coding** (front-end and back-end, mobile, cloud), **Testing** (automated testing, quality metrics), and **Deployment/Maintenance** (DevOps, continuous integration/delivery).	The study found that curricular flexibility and alternative credentials are perceived as effective strategies to bridge the academia-industry gap. Key findings include the value of "early wins" through MasterTracks, the importance of practical, project-based learning, and the positive impact on student employability and professional development. Challenges identified include the need for greater recognition of alternative credentials and ensuring consistent pedagogical approaches across faculty.
Cambarieri et al. (2025)	To present a project for implementing a microcredential system for the Bachelor's Degree in Systems at UNRN, using W3C Verifiable Credentials (VC) to digitally and securely certify specific competencies acquired during training.	Students and graduates of the Bachelor's Degree in Systems (and the intermediate degree of University Technician in Programming) at the Universidad Nacional de Rio Negro (UNRN), Argentina.	**Microcredential** system based on **Verifiable Credentials (VC)** using the W3C standard. The project has two components: Organizational-Regulatory (establishing norms) and Technological (implementing VC infrastructure).	Not a specific course, but a system to issue microcredentials for specific competencies within the Bachelor's in Systems curriculum. Examples include credentials for Programming Languages, Database Management, Software Architecture, Cloud Computing, Data Science, AI, Cybersecurity, and even a digital "Regular Student Certificate.''	The content is based on competencies defined in the national accreditation standards (Ministry of Education Resolution 1558/2021) and the university's curriculum. It covers areas like Algorithms, Software Engineering, Databases, Architecture, Operating Systems, and Networks.	The microcredentials are designed to certify competencies across the entire software process, including **Conception/Analysis**, **Software Design**, **Coding** (front-end and back-end), **Testing**, and **Deployment/Maintenance**, as well as project management and social/professional aspects.	This is a project proposal. The paper presents the conceptual framework, the identified opportunity, the technological solution (W3C VCs), and the planned two-component design (organizational and technological). No empirical findings or results from an implemented system are reported. The intended benefits include enhanced graduate employability, reduced fraud, and alignment with international standards.

4. Results

The following section presents the results of this systematic review, synthesized from the 21 studies that met the inclusion criteria. The comprehensive overview of these studies, including their main goals, target populations, pedagogical approaches, course content, and key findings, serves as the foundation for the subsequent thematic analysis organized around the nine research questions.

5. Discussion

This section synthesizes the findings from the reviewed literature, organizing the discussion around the nine research questions that guided this systematic review. It critically examines the current state of microlearning in software engineering education, exploring the technological approaches, curriculum coverage, evidence of effectiveness, and the roles of microcredentials and artificial intelligence. The discussion also interrogates the design frameworks guiding implementation, the integration of professional competencies, and the contextual challenges revealed across diverse educational settings. Rather than merely summarizing positive findings, this section engages in critical analysis, identifying significant gaps, methodological fragilities, and unresolved tensions that temper premature enthusiasm while charting productive directions for future research and practice.

5.1. Pedagogical Approaches and Technological Integration (Addressing RQ1)

The microlearning initiatives surveyed reveal a pedagogical landscape characterized by considerable diversity but also notable fragmentation. The reviewed studies demonstrate that microlearning in software engineering education employs diverse pedagogical approaches—including project-based learning, gamification, concern-based frameworks, and automated assessment—implemented across mobile, web-based, and LMS platforms, with consistent evidence of student engagement and positive usability outcomes. While the literature demonstrates creative experimentation with various instructional strategies, it simultaneously exposes a field still grappling with fundamental questions about how best to design and deliver microlearning for software engineering education.

Project-based learning emerges as a prominent pedagogical approach, with Wira et al. (2026) demonstrating its integration with microlearning for object-oriented programming instruction. Their combination of videos, animations, interactive modules, and quizzes within a Moodle LMS environment represents a thoughtful attempt to balance content chunking with authentic, sustained projects. Similarly, Jwo (2015) pushes this further by incorporating comics and micro-movie creation projects, providing students with realistic contexts for understanding software engineering challenges. However, these implementations reveal a persistent tension: microlearning's inherent brevity sits uncomfortably alongside the sustained engagement required for meaningful project work, and neither study adequately addresses how learners integrate discrete microlearning episodes into coherent project understanding.

Gamification features prominently across multiple implementations, yet with varying degrees of pedagogical sophistication. Gasca-Hurtado et al. (2024) and Gasca-Hurtado and Gómez-Álvarez (2021) employ lottery-style mechanics in their "Mr. Lottery" application for teaching Scrum concepts. The motivational benefits are evident, but these approaches risk prioritizing engagement over depth. The finding by Gasca-Hurtado et al. (2024) that while motivation increased significantly, post-test score improvements were not statistically significant, raises uncomfortable questions about whether gamified microlearning genuinely enhances learning or merely makes students feel better about not learning. The field lacks critical examination of when gamification serves authentic pedagogical purposes versus when it functions as superficial engagement theater.

The separation-of-concerns approach developed by Correa Botero (2015) and extended by Correa et al. (2018) represents one of the more pedagogically sophisticated frameworks. By organizing learning around developer concerns and combining microlearning units with examples, this approach acknowledges that software engineering learning is fundamentally about addressing authentic problems. The demonstrated 40% time reduction for initial tasks is promising, yet the approach remains narrowly focused on framework learning rather than broader software engineering competencies. The question of whether such concern-based decomposition can scale to entire curricula remains unanswered.

Automated assessment integration, explored extensively by Skalka and Drlik (2020) and Skalka et al. (2020), represents a crucial technological bridge between learning and practice. The Virtual Programming Lab implementation within Moodle provides immediate feedback on code correctness, addressing one of microlearning's potential weaknesses: the gap between consuming content and applying it. However, the finding that automated assessment significantly improved outcomes for introductory topics but not advanced ones should give pause. It suggests that as programming tasks become more complex, the binary correctness feedback of automated assessment may be insufficient. The field has yet to develop sophisticated automated feedback mechanisms that address design quality, architectural decisions, or code maintainability.

Mobile learning implementations by Hadiyana et al. (2026) and Wibisono et al. (2024) demonstrate the technological affordances of contemporary platforms, including short-form video, learning analytics, and integrated code editors. The HEART metrics evaluation by Wibisono et al. revealing differential performance between Mimo and SoloLearn on retention and credibility highlights that technological features alone do not determine effectiveness—content quality and pedagogical design matter profoundly. Yet the field lacks systematic understanding of what makes microlearning content credible and retention-promoting versus superficial and forgettable.

The most striking absence across these implementations is meaningful personalization and adaptivity. Rusak (2017) provide a notable exception, demonstrating an algorithmic approach to personalizing learning paths based on student background knowledge and preferences. Their finding that approximately 60% of course content required personalization while 40% was universally suitable offers a valuable heuristic. However, this work remains isolated. Despite Hadiyana et al. (2026) incorporating AI-based rule systems and learning analytics, the personalization described is rudimentary—primarily content sequencing rather than genuine adaptation to learner cognition, misconceptions, or developing mental models. The field has not yet grappled with what intelligent personalization for software engineering learning might mean: adapting not just content difficulty but representational format, example domains, debugging strategies, or design approaches based on learner patterns.

Learning analytics integration remains similarly superficial. Multiple studies mention analytics instrumentation, but the actual use of analytics data for improving learning remains underdeveloped. Hadiyana et al. (2026) describe data collection for system performance monitoring rather than learning insight. Skalka and Drlik (2020) use interaction data for at-risk student prediction, representing one of the few examples of analytics informing pedagogical intervention. However, the field lacks closed-loop systems where analytics continuously inform content refinement, personalization algorithms, or instructional design improvements. The promise of learning analytics to illuminate how students actually engage with microcontent, where they struggle, and which formats prove most effective for different competencies remains largely unrealized.

The integration of technical and professional competencies through microlearning, explicitly addressed by Luo and Li (2025), reveals both possibilities and limitations. Their

six-module intervention targeting teamwork, leadership, communication, time management, and emotional intelligence demonstrated universal effectiveness across disciplines. However, the modules exist separately from technical learning rather than integrated within it. The more challenging pedagogical question—how to develop professional competencies through the very act of technical learning, how to embed communication practice within coding exercises, how to foster collaboration skills during individual microlearning—remains unaddressed. Hadiyana et al. (2026) similarly treat soft skills as standalone modules, perpetuating the problematic separation of technical and professional formation.

Perhaps most concerning is what remains unexamined: the cognitive load implications of fragmented learning across distributed microlearning episodes. While microlearning theoretically reduces cognitive load by chunking content, the literature provides no evidence about how learners construct coherent mental models from dispersed micro-episodes. The studies celebrate engagement metrics and immediate knowledge gains but ignore the critical question of whether microlearning supports or undermines the development of integrated understanding of software systems as interconnected wholes. The field has not investigated how learners connect discrete microlearning units into architectural understanding, design reasoning, or systematic debugging strategies.

Furthermore, the technological implementations largely assume individual, solitary learning. Despite software engineering being fundamentally collaborative, none of the platforms meaningfully support collaborative microlearning—paired programming within micro-episodes, collaborative debugging of shared code snippets, or social annotation of learning content. The social dimension of professional practice remains absent from the technological designs. The integration of industry certifications and microcredentials, explored by Guzmán Rojas (2024); Urem et al. (2025) and Cambarieri et al. (2025), raises yet-unanswered questions about the relationship between microlearning platforms and credentialing systems. While these works describe the organizational and technological infrastructure for issuing verifiable credentials, they do not examine whether microlearning actually prepares students for certification assessments or whether credential pursuit enhances or distorts learning motivation. The field lacks critical examination of when microcredentials serve as meaningful competency indicators versus when they function as credentialing theater.

In sum, the pedagogical and technological landscape reveals creativity within conservative boundaries. The field has successfully demonstrated that microlearning can engage students and convey discrete knowledge. It has not yet shown that microlearning can support the development of integrated software engineering understanding, that personalization and analytics can meaningfully transform learning, or that the technological platforms can move beyond digitized conventional instruction to enable genuinely new forms of software engineering pedagogy. The next generation of research must ask harder questions about learning coherence, personalization depth, collaborative possibilities, and the relationship between microlearning episodes and the development of expert software engineering practice.

5.2. Curriculum Coverage and Competency Development (Addressing RQ2)

The corpus of studies reveals a striking imbalance in how microlearning approaches have been applied across the software engineering curriculum, exposing both concentrated areas of activity and vast territories of neglect. The evidence confirms that microlearning has been successfully applied to teach introductory programming concepts, object-oriented principles, and declarative project management knowledge, with several studies demonstrating measurable learning gains in these foundational areas. This uneven coverage raises fundamental questions about whether microlearning is being deployed strategically to

address genuine educational needs or opportunistically in areas that align with convenience rather than curricular importance.

Programming and coding competencies dominate the literature to an extent that borders on monomaniacal. Wira et al. (2026) focus on object-oriented programming concepts including encapsulation, inheritance, and polymorphism delivered through videos and interactive modules. Skalka and Drlik (2020) and Skalka et al. (2020) concentrate intensively on introductory Java programming, tracking student interactions with automated assessment and microlearning units across variables, loops, arrays, and object-oriented constructs. Wibisono et al. (2024) evaluate the Mimo and SoloLearn applications, both fundamentally oriented toward teaching Python, SQL, JavaScript, and web development basics. Correa Botero (2015) and Correa et al. (2018) center their concern-based approach on learning web application frameworks, specifically CodeIgniter for PHP development. Even Jo et al. (2025), despite their broader evaluation framework, ground their work in coding fundamentals instruction. This concentration suggests that microlearning researchers have gravitated toward what is easiest to measure and implement—discrete syntax elements, isolated language features, and bounded programming tasks—rather than what might be most consequential for software engineering education.

The preponderance of programming-focused microlearning becomes particularly problematic when examined against the authentic demands of software engineering practice. Professional software development is fundamentally about design, architecture, trade-off analysis, and system thinking—competencies notably absent from this literature. Rusak (2017) make a partial contribution by including UML in their Android development microlearning sessions, acknowledging that modeling and design representation matter alongside coding. However, this remains an isolated exception. The field has not grappled with how to represent architectural patterns, design rationale, or quality attribute trade-offs in microlearning formats. These are not naturally bite-sized competencies; they require sustained reasoning about interconnected concerns that resist fragmentation. The absence of research on architectural thinking in microlearning contexts may indicate not that the problem is insoluble, but that researchers have avoided the harder pedagogical design work required.

Software project management represents a secondary but meaningful cluster of activity. Gasca-Hurtado et al. (2024) target Scrum framework concepts including roles, events, artifacts, and their interrelationships, using gamified microlearning to support comprehension of project management methodologies. Suciu (2022) similarly addresses software project management, encompassing both hard skills of project administration and soft skills of team coordination and facilitation. Jwo (2015) includes project management topics within his comics-based approach, addressing estimation, productivity, and lifecycle management. These contributions are valuable, yet they reveal a concerning pattern: project management is taught primarily through declarative knowledge about methodologies rather than through development of project management judgment. Students learn Scrum roles and artifacts but not how to navigate the ambiguous, politically complex situations that characterize actual project leadership. The microlearning format, with its emphasis on clear, bounded content, may inadvertently privilege codified methodology knowledge over the tacit, situation-specific judgment that distinguishes expert project managers.

Testing and quality assurance appear only peripherally, almost as afterthoughts to programming instruction. Skalka and Drlík (2018) and Skalka (2018) mention testing and debugging as activities supported by automated assessment, but testing is framed as verification of code correctness rather than as a systematic competency involving test design, coverage analysis, or defect prevention. No studies address test-driven development, behavior-driven development, or testing strategies through microlearning. Given that testing constitutes a substantial portion of professional software engineering effort and

represents a learnable competency with both conceptual and procedural dimensions, this absence is striking. The field has not asked whether microlearning could support the development of testing judgment—when to test, what to test, how deeply, with what techniques—or whether testing competencies inherently require the kind of sustained practice that microlearning cannot provide.

Requirements engineering and software architecture, foundational to software engineering practice, are virtually invisible. Jwo (2015) touches on requirements through his comics-based case studies, and the microcredential proposals of Urem et al. (2025) and Cambarieri et al. (2025) gesture toward architecture and design competencies in their credentialing frameworks. But these are mentions without substance—no studies investigate how microlearning might support the development of requirements elicitation skills, architectural decision-making, or design rationale documentation. These competencies involve managing uncertainty, negotiating with stakeholders, and making irreversible decisions with long-term consequences—precisely the kind of complex professional judgment that might seem resistant to microlearning. Yet the research community has not even attempted to explore whether microlearning could contribute, perhaps through case-based micro-episodes presenting architectural trade-offs, or through micro-simulations of stakeholder interactions. The absence represents not a demonstrated limitation but an unexplored possibility.

Soft skills and professional competencies receive attention primarily through standalone interventions rather than integrated development. Luo and Li (2025) provide the most comprehensive treatment, targeting teamwork, leadership, communication, time management, and emotional intelligence through structured microlearning modules. Hadiyana et al. (2026) similarly focus on communication, collaboration, problem-solving, and time management through AI-enabled mobile microlearning. These studies demonstrate that microlearning can improve professional competencies, yet they perpetuate a problematic separation: soft skills are taught in dedicated modules rather than developed through the very activities of technical work. The integration challenge—how communication skill develops through code review microlearning, how leadership emerges through project management micro-episodes—remains unaddressed. Furthermore, these studies measure immediate self-reported improvements rather than transfer to authentic professional situations or longitudinal development.

The coverage of contemporary and emerging technology areas reveals both responsiveness and superficiality. Guzmán Rojas (2024) includes IoT, AI, cloud computing, microservices, and DevOps in the MISO master's curriculum, demonstrating that microcredential frameworks can encompass modern topics. Urem et al. (2025) similarly incorporate cloud computing, machine learning, and environmental data analytics. However, these appear in curriculum descriptions rather than in studies of actual microlearning implementation. The research has not examined whether microlearning effectively supports understanding of distributed systems concepts, machine learning pipeline construction, or cloud architecture patterns. These domains involve emergent behavior, complex dependencies, and conceptual understanding that may or may not align with microlearning's affordances. The field lacks evidence.

Perhaps most concerning is the absence of attention to software maintenance and evolution, which consume the majority of professional software effort. Jwo (2015) mentions lifecycle management and configuration management in passing, but no studies address how microlearning might support comprehension of legacy code, refactoring strategies, or evolution planning. This neglect reflects a broader educational bias toward greenfield development, but it is particularly problematic for microlearning, which could theoretically support just-in-time learning about specific maintenance tasks, legacy system comprehension strategies, or refactoring techniques. The field has not explored these possibilities.

The competency frameworks underlying these studies reveal additional limitations. Where competencies are specified—as in Wira et al. (2026) focusing on encapsulation, inheritance, and polymorphism, or Gasca-Hurtado et al. (2024) targeting Scrum roles and artifacts—they tend toward discrete, declarative knowledge rather than integrated, procedural capability. The field lacks articulation of what competency progression might look like across microlearning episodes. How does mastery of isolated syntax elements aggregate into programming fluency? How do discrete Scrum concept quizzes translate into project management capability? These questions remain not just unanswered but unasked.

Furthermore, the coverage exhibits systematic neglect of higher-order competencies according to established taxonomies. Skalka and Drlík (2018) explicitly map their microlearning framework to Bloom's taxonomy, demonstrating aspiration toward create-level activities. Yet the implemented studies predominantly address remember, understand, and apply levels. The creation of original software artifacts, evaluation of design alternatives, and synthesis of multiple concerns into coherent systems—the competencies that distinguish software engineers from programmers—are absent from the empirical literature. The field has not demonstrated that microlearning can support development of these highest-priority professional competencies.

The vocational and professional education contexts represented—vocational high school students in Wira et al. (2026) and Wigati et al. (2024), undergraduate computer science students across multiple studies, and professionals in Guzmán Rojas (2024)—suggest that microlearning is being explored across the educational spectrum. Yet comparative questions remain unexamined: whether vocational students benefit from different microlearning designs than undergraduates, whether professionals need different competency representations than students, or whether the same microlearning approaches transfer across contexts. The studies treat populations as settings rather than as variables worthy of systematic investigation.

Finally, the relationship between microlearning coverage and industry demand receives only superficial attention. Garcia and Stefany (2023) analyzes industry job data to identify in-demand competencies, and the microcredential proposals claim alignment with labor market needs. Yet no studies examine whether microlearning-developed competencies actually transfer to workplace performance or whether employers recognize and value microcredentialed competencies. The field has not closed the loop between educational design and professional outcomes, leaving claims of industry relevance largely aspirational.

In sum, the curriculum coverage reveals a field that has concentrated on what is easiest rather than what is most needed. Programming fundamentals dominate while architecture, requirements, testing, and maintenance languish. Declarative knowledge receives attention while professional judgment remains unexplored. The next generation of research must ask harder questions about whether microlearning can address the full scope of software engineering competency, or whether its appropriate role is more focused and strategic—supporting discrete skill development within broader educational designs that address complex competencies through other means. The current literature cannot answer this question because it has not seriously attempted to develop microlearning for software engineering's most challenging competencies.

5.3. Evidence of Effectiveness: Learning Outcomes and Engagement (Addressing RQ3)

The reviewed studies provide evidence that microlearning positively influences student engagement, motivation, and course participation rates, with quasi-experimental designs demonstrating improvements in task completion efficiency and immediate knowledge acquisition on certification-aligned assessments. The evidence base for microlearning's effectiveness in software engineering education presents a paradox: studies consistently

report positive outcomes, yet closer examination reveals methodological fragilities, measurement superficiality, and unanswered questions that should temper enthusiasm. The field has accumulated abundant claims but comparatively little robust evidence about what microlearning actually achieves, for whom, and under what conditions.

Learning outcome measurements across the corpus rely heavily on immediate knowledge gains and course performance metrics. Wira et al. (2026) report improved learning outcomes with an N-Gain of 0.52 (moderate category) and achievement of minimum mastery criteria, yet this tells us little about what students actually learned beyond passing threshold assessments. The N-Gain measure, comparing pre-test and post-test scores, captures short-term knowledge acquisition but provides no evidence about retention, transfer, or application.

The handful of studies examining longer-term outcomes reveal more complicated pictures. Gasca-Hurtado et al. (2024) found that while their gamified microlearning strategy significantly improved motivation measures, the increase in Scrum certification exam scores was not statistically significant. This dissociation between affective response and learning outcome should provoke reflection across the field. Students felt more motivated and stimulated, yet this heightened engagement did not translate into measurable knowledge gains. The finding raises uncomfortable questions about whether microlearning's celebrated engagement benefits might sometimes function as affective placebo—making learners feel better about learning without actually improving learning. The field has not systematically investigated when engagement correlates with learning outcomes and when it operates independently.

Skalka and Drlik (2020) large-scale study across four academic years with over 300 students provides some of the most methodologically rigorous evidence, yet their findings are notably mixed. Automated assessment significantly improved student outcomes in introductory programming topics but not in advanced topics. More striking, microlearning units themselves did not lead to significant differences in either introductory or advanced course components. This null finding, buried within a study primarily cited for its automated assessment contributions, should unsettle microlearning advocates. If microlearning units fail to produce measurable effects even when implemented within supportive institutional contexts, what does this imply about the hundreds of studies reporting positive results without rigorous controls? The authors do find moderate to strong correlations between success in interactive activities (including microlearning) and final course outcomes, but correlation is not causation—students who engage more with any learning resource tend to perform better, regardless of the resource's intrinsic effectiveness.

Correa Botero (2015) and Correa et al. (2018) provide evidence with greater ecological validity, measuring time reduction (approximately 40% in initial tasks) and task completion rates rather than test scores alone. Their quasi-experimental design comparing the CME approach against traditional documentation use represents a meaningful attempt to measure what matters for practitioners: getting work done effectively. The finding that experimental groups completed more tasks and completed the first iteration faster suggests transfer to authentic activity. Yet even here, the outcome measures remain proximal—task completion in controlled settings rather than sustained performance in professional contexts. The field lacks longitudinal studies tracking whether microlearning-experienced students demonstrate different career trajectories, adaptation to new technologies, or problem-solving approaches years after intervention.

Motivation and engagement metrics, while consistently positive, suffer from measurement approaches that may not capture what they claim. Gasca-Hurtado et al. (2024) found mean scores for "Motivation to learn in the specific context" and "Stimulus for learning" higher than theoretical averages, yet these are self-report measures collected immediately after gamified experiences. Suciu (2022) reports increased student engagement, participation

in online discussions, and dramatic increases in course feedback rates (from approximately 3–14% to 59–72%). These behavioral indicators are more objective than self-reports, yet they measure engagement with the learning platform rather than engagement with software engineering as a discipline. Students may complete more microlearning modules while remaining superficially engaged with the underlying professional practice. The conflation of platform engagement with learning engagement pervades the literature.

Wibisono et al. (2024) introduce greater methodological sophistication through their HEART metrics and Honeycomb framework evaluation of Mimo and SoloLearn. Their finding that Mimo provided more fulfilling user experience, particularly in retention and credibility, while SoloLearn scored higher on happiness but lower on retention, suggests that user experience dimensions are not unitary and may trade off against each other. However, this study evaluates commercial applications rather than educational interventions, and the relationship between user experience ratings and learning outcomes remains unexplored. Students may prefer platforms that feel good without learning more from them—a possibility the field has not seriously investigated.

The pilot study by Hadiyana et al. (2026) reports module completion rates of 66–78% and average quiz scores of 72–81/100, suggesting reasonable engagement and immediate comprehension. Yet these metrics, collected during prototype testing with ten modules, provide no evidence about sustained engagement over longer periods or about whether quiz performance predicts applied capability. The system simulations indicating architecture readiness for 300 concurrent users address technical scalability rather than educational effectiveness—an conflation of technological with pedagogical validation that appears across multiple studies.

Luo and Li (2025) provide the most comprehensive soft skills evidence, demonstrating significant improvements across all five targeted competencies with differential effects by discipline. Leadership modules proved most impactful for technical and engineering students while emotional intelligence training was most valuable for business students. These differential findings suggest that microlearning effectiveness may vary meaningfully by learner population and competency type—a complexity the field has only begun to acknowledge. Yet even here, measurement relies on self-reported skill development rather than behavioral assessment or third-party observation. Students believe they have improved their communication and leadership abilities; whether colleagues or supervisors would agree remains unknown.

The absence of transfer studies represents perhaps the most significant gap in the evidence base. No studies examine whether microlearning-developed competencies transfer to authentic software engineering tasks, whether students apply learning from micro-episodes in subsequent project work, or whether microlearning produces different transfer patterns than traditional instruction. Software engineering education ultimately aims to produce professionals who can design, build, and maintain systems—not students who can pass microlearning quizzes. The field has not established the relationship between microlearning performance and professional capability.

Longitudinal evidence is similarly absent. The studies capture learning at single time points or over single semesters, providing no evidence about retention over months or years. Given that microlearning's theoretical advantage lies in alignment with how people naturally learn and forget, the absence of retention studies is particularly problematic. Do students retain microlearned content better than traditionally learned content after six months? After two years? After encountering subsequent learning that builds on foundational concepts? The literature cannot answer these questions.

Comparative effectiveness research remains underdeveloped. While individual studies report positive outcomes, few directly compare microlearning against well-designed

alternatives. Skalka and Drlik (2020) include comparison conditions, but most studies lack control groups receiving equivalent instructional time through different modalities. The field cannot claim that microlearning is more effective than alternatives because it has not conducted the comparative studies that would support such claims. The positive findings may reflect that any well-designed instructional intervention, regardless of modality, improves outcomes compared to no intervention.

The measurement instruments themselves warrant critical examination. Quiz scores, certification exam performance, and N-Gain calculations all assess declarative knowledge and basic comprehension—the lower levels of cognitive taxonomies. No studies assess whether microlearning supports development of evaluation, synthesis, or creation capabilities in software engineering. Students may accurately recall Scrum roles while being unable to apply Scrum in ambiguous project situations. They may correctly answer questions about encapsulation while writing monolithic, tightly coupled code. The field's measurement choices systematically exclude the higher-order competencies that distinguish expert software engineers.

Affective measurement raises additional concerns. Engagement, motivation, and satisfaction are typically measured through researcher-developed instruments with limited validation evidence. Students may report satisfaction because microlearning respects their time and fits into busy schedules, but satisfaction with learning format differs from satisfaction with learning itself. The field has not disentangled whether students appreciate microlearning because it makes learning easier or because it makes learning better.

The heterogeneity of outcome measures across studies prevents meta-analytic synthesis and accumulation of knowledge. Each study operationalizes effectiveness differently, using varied instruments, timeframes, and competency definitions. This methodological fragmentation means that even consistently positive findings cannot be aggregated into robust estimates of effect size. The field lacks common outcome measures or validated instruments for assessing microlearning outcomes in software engineering.

Finally, the publication bias question remains unaddressed. The corpus contains no studies reporting negative or null findings for well-designed microlearning interventions. Skalka and Drlik (2020) report null findings for microlearning units, but these are embedded within a study primarily framed around automated assessment's positive effects. The field may be systematically underreporting interventions that failed, overrepresenting positive cases, and creating an artificially optimistic evidence base. Without registered reports, replication studies, or journals committed to publishing null results, the true effectiveness landscape remains unknowable.

In sum, the evidence base for microlearning effectiveness in software engineering education is weaker than the volume of positive findings suggests. The field has accumulated abundant claims supported by methodologically limited studies measuring proximal outcomes with instruments that may not capture what matters. The next generation of research must prioritize longitudinal designs, transfer assessments, comparative studies with appropriate controls, and measurement of higher-order competencies. It must confront the possibility that engagement and learning outcomes dissociate, that positive affective responses may not predict professional capability, and that microlearning's role in software engineering education may be more limited and strategic than its advocates suggest. The current evidence cannot rule out these uncomfortable possibilities because it has not been designed to test them.

5.4. Microcredentials and Industry Alignment (Addressing RQ4)

The intersection of microlearning with microcredentials and industry certification represents one of the most enthusiastically promoted yet empirically underdeveloped areas

in the corpus. Institutional case studies and curriculum proposals demonstrate that microcredentials can be feasibly integrated into formal software engineering programs through in-house development, industry partnership models, and technological infrastructure such as W3C Verifiable Credentials, with stakeholders perceiving value for curricular flexibility and graduate employability. While multiple studies advocate for microcredentials as bridges between education and employment, the evidence supporting these claims remains largely aspirational, and critical examination reveals fundamental questions about value, recognition, and educational integrity that the field has barely begun to address.

Urem et al. (2025) provide the most comprehensive treatment, presenting a model for integrating microcredentials into formal computer science education through three mechanisms: developing in-house microcredentials, recognizing external industry certifications from providers like Oracle and Coursera, and collaborating with industry partners. Their case study of Sibenik University of Applied Sciences demonstrates the feasibility of this approach through long-term industry collaboration and European project participation. Yet the paper's findings remain descriptive rather than evaluative—it establishes that microcredentials can be implemented within university structures but provides no evidence about whether this implementation achieves its intended outcomes. Do students who earn microcredentials demonstrate superior workplace performance? Do employers preferentially hire or promote microcredentialed graduates? Does microcredential integration enhance or distract from foundational degree education? These questions go not just unanswered but unasked.

Guzmán Rojas (2024) similarly presents the MISO master's program at Universidad de los Andes as a case study of curricular flexibility through MasterTracks and MOOC integration with Coursera. The findings emphasize perceived effectiveness: stakeholders view curricular flexibility and alternative credentials as valuable strategies for bridging academia-industry gaps. The importance of "early wins" through MasterTracks, the value of practical project-based learning, and positive impacts on student employability and professional development are all reported as stakeholder perceptions rather than demonstrated outcomes. The study identifies challenges including need for greater recognition of alternative credentials and ensuring consistent faculty access—important implementation considerations, yet still operating at the level of program logistics rather than educational or employment impact. The field lacks evidence that microcredentialled students actually secure better positions, command higher salaries, or perform more effectively than traditionally credentialed peers.

Cambarieri et al. (2025) approach microcredentials from a technological perspective, proposing W3C Verifiable Credentials infrastructure for the Bachelor's Degree in Systems at UNRN. Their two-component design addresses organizational-regulatory norms and technological implementation, with examples spanning programming languages, database management, software architecture, cloud computing, and cybersecurity. The intended benefits—enhanced graduate employability, reduced fraud, alignment with international standards—are clearly articulated. Yet this remains a project proposal without implementation evidence. The paper establishes technical feasibility and conceptual soundness but provides no data about whether employers recognize or trust W3C Verifiable Credentials, whether graduates find the credentialing process valuable, or whether the credentialled competencies actually predict professional success. The technical infrastructure precedes the social infrastructure of employer recognition and trust, and the paper does not address how this recognition gap might be bridged.

The microcredential literature reveals a troubling pattern: enthusiastic adoption driven by institutional and technological factors rather than demonstrated educational or employment value. Universities see microcredentials as mechanisms for attracting new student

populations, responding to employer complaints about graduate readiness, and generating additional revenue streams. Technology providers see opportunities for platform sales and integration services. Employers appreciate signals that may reduce hiring uncertainty. Yet the interests of learners—the individuals whose time, money, and effort are invested in credential pursuit—receive remarkably little empirical attention. LaPlante (2024), cited in the references, begins to address graduate student motivations for microcredential adoption, but this perspective remains marginal in the corpus.

The relationship between microcredentials and the competencies they claim to certify receives insufficient critical examination. Garcia and Stefany (2023) analyzes industry job data using big data and artificial intelligence to identify in-demand competencies in IT-computer software, arguing that microcredentials can respond to these demands. Yet the mapping between job posting analysis and credential design involves numerous inferential leaps. Job postings list desired skills, but they do not reveal whether those skills are best developed through microlearning, whether credentials accurately assess them, or whether credential holders actually possess transferable capability rather than test-passing ability. The field has not established validity evidence for microcredentials as competency indicators.

Oliver (2022) works toward a common definition of microcredentials at the UNESCO level, addressing conceptual clarity. Yet definitional work, while valuable, cannot substitute for evidentiary work. The field can agree on what microcredentials are while remaining ignorant of whether they work. The proliferation of definitions and frameworks—Keeley (2025) Scope and Depth Micro-Credential Framework, Catenazzi et al. (2025) comprehensive methodology for curriculum development and certification—creates an appearance of scholarly activity that may outpace actual understanding of microcredential effectiveness.

The industry alignment claims merit particular scrutiny. Microcredentials are promoted as responses to employer demands for graduates with specific, job-ready skills. Yet the studies provide no evidence that employers actually prefer microcredentialled graduates, that microcredentials influence hiring decisions, or that microcredential holders require less onboarding or training. The employer perspective is almost entirely absent from the research. Studies cite employer engagement in curriculum design or advisory boards, but they do not systematically investigate employer recognition, trust, or valuation of microcredentials. This absence is particularly striking given that industry alignment is the primary justification for microcredential investment.

Furthermore, the relationship between microcredentials and traditional degrees remains undertheorized. Orman et al. (2023) reflect on microcredentials and higher education, noting both opportunities and tensions. Kohler et al. (2021) situate microlearning and microcredentials within higher education contexts. Yet the studies do not grapple with fundamental questions about whether microcredentials complement or undermine degree education. If students can earn credentials for specific competencies through microlearning, what incentive remains for sustained engagement with foundational, integrative, or breadth-oriented degree curricula? If universities prioritize microcredential development, what happens to faculty expertise in areas not easily credentialized? The field has not examined whether microcredential proliferation creates perverse incentives that degrade comprehensive education.

The competency coverage in microcredential proposals reveals additional concerns. Urem et al. (2025) include Java programming, database design, PL/SQL, version control, cloud computing, and machine learning—all valuable competencies, yet all representing current industry preferences rather than enduring disciplinary foundations. The microcredential model privileges what is currently in demand over what will remain important across careers, potentially incentivizing shallow learning of transient technologies rather

than deep learning of durable principles. Students may earn cloud computing microcredentials while lacking understanding of distributed systems fundamentals that would enable adaptation to next-generation platforms. The field has not investigated whether microcredential seekers develop adaptable foundational knowledge or merely acquire current-tool familiarity.

The assessment validity question remains largely unaddressed. Microcredentials typically certify competency through some assessment mechanism—quizzes, projects, or examinations. Yet the studies provide no evidence about whether these assessments predict real-world performance, whether they measure what they claim to measure, or whether they are vulnerable to coaching, cramming, or cheating. The correlation between assessment performance and professional capability is assumed rather than demonstrated. Given that microcredentials are intended to signal competence to employers, this validity gap is particularly problematic.

The stackability and progressive development aspects of microcredentials receive rhetorical attention but empirical neglect. Proponents argue that microcredentials can stack into larger credentials or degrees, enabling flexible learning pathways. Guzmán Rojas (2024) describes MasterTracks as macro-credentials that can lead to full master's degrees. Yet no studies track whether learners actually progress through stacked credentials, whether stacked credentials provide coherent educational experiences, or whether employers recognize stacked credentials differently from individual ones. The stackability claim remains theoretical.

The international and cross-cultural dimensions of microcredential recognition receive no attention. If microcredentials are to enhance graduate mobility and employability across borders, they must be recognized by employers and educational institutions in multiple countries. The corpus contains no studies examining international recognition, cross-border transfer, or employer attitudes in different cultural contexts. The assumption that microcredentials will function globally ignores the highly localized nature of credential recognition and the variability in employer trust across different labor markets.

Finally, the equity implications of microcredential proliferation remain unexplored. If microcredentials become important signals for employment, learners must bear the costs of credential pursuit—fees, time, technology access. Learners with greater resources can accumulate more credentials, potentially exacerbating rather than reducing employment inequality. Learners from underresourced backgrounds may be priced out of credential markets. The studies do not examine who earns microcredentials, who benefits from them, or whether they widen or narrow opportunity gaps. The enthusiastic promotion of microcredentials as democratizing forces proceeds without evidence about their actual distributional effects.

In sum, the microcredential literature reveals a field heavy with promise and light on proof. Institutional case studies demonstrate feasibility but not effectiveness. Technological proposals establish infrastructure without examining social recognition. Competency frameworks proliferate while validity evidence remains absent. The enthusiasm for microcredentials as bridges between education and industry has outpaced the evidence that they actually function as bridges. The next generation of research must examine employer recognition systematically, track graduate employment outcomes rigorously, investigate assessment validity thoroughly, and confront the equity implications honestly. Without such evidence, microcredentials risk becoming solutions in search of problems, adopted for institutional convenience rather than demonstrated educational or employment value.

5.5. AI and Personalization in Microlearning (Addressing RQ5)

The integration of artificial intelligence and personalization technologies into microlearning platforms represents one of the most enthusiastically proclaimed yet empirically underdeveloped areas in the corpus. Pilot implementations have established the technical feasibility of rule-based recommendation systems for content personalization, learning analytics dashboards for at-risk student prediction, and system architectures capable of supporting medium-scale concurrent users in AI-enabled microlearning environments. While multiple studies invoke AI, machine learning, and learning analytics as transformative capabilities, critical examination reveals a striking gap between rhetorical ambition and implemented reality. The field has produced abundant speculation about what AI could do but remarkably little evidence about what AI actually does in software engineering microlearning.

Hadiyana et al. (2026) provide the most explicit engagement with AI integration, describing an "AI-capable, mobile-based microlearning application" that incorporates "learning analytics, and a rule-based recommendation system" for vocational soft skills development. The language suggests sophisticated intelligent capabilities, yet the implemented system reveals considerably more modest functionality. The AI component consists of rule-based recommendations—essentially conditional logic that suggests content based on predetermined rules rather than genuine machine learning that adapts to individual learner patterns. The learning analytics instrumentation collects data on module completion and quiz performance but does not appear to use this data for personalization, adaptive sequencing, or intelligent feedback. The gap between the "AI-enabled" framing and the rule-based implementation exemplifies a pattern across the literature: AI functions as rhetorical enhancement rather than substantive capability.

Rusak (2017) represent a more technically sophisticated approach, developing and verifying an algorithmic solution for personalizing learning content and paths by recommending micro-learning sessions based on student background knowledge, preferences, and performance. Their hybrid recommender system combining collaborative and content-based filtering demonstrates genuine intelligence, clustering students into four profiles (beginner to expert) and identifying that approximately 60% of course content required personalization while 40% was universally suitable. This finding provides valuable design guidance for the field. Yet the study remains isolated—a proof of concept rather than an implemented system with demonstrated learning impact. The verification demonstrates algorithmic functionality, not educational effectiveness. Whether personally recommended microlearning sessions actually improve learning outcomes, engagement, or retention remains unexamined.

Skalka and Drlik (2020) and Skalka et al. (2020) incorporate learning analytics more substantively, using student interaction data from automated assessment and microlearning units as predictors for identifying at-risk students in introductory programming courses. Their finding of moderate to strong correlation between success in interactive activities and final course outcomes demonstrates that learning analytics can serve diagnostic functions. However, prediction is not personalization. The studies identify which students are at risk but do not implement adaptive interventions based on this identification. The analytics capability exists in isolation from instructional adaptation, representing intelligence without action. The field has not closed the loop between prediction and personalization.

Skalka (2018) proposes even more ambitious AI integration, describing data processing methods for automated content development through text summarization. The conceptual framework suggests using natural language processing to automatically generate microlearning content from existing materials, potentially addressing the scalability challenge of microcontent creation. Yet this remains entirely conceptual—no implementa-

tion, no evaluation, no evidence that automatically summarized content supports learning as effectively as human-designed materials. The proposal raises fundamental questions about whether learning content can be algorithmically generated without loss of pedagogical quality, coherence, or engagement. The field has not begun to investigate these questions empirically.

The learning analytics implementations across the corpus reveal a pattern of data-rich, insight-poor environments. Multiple studies mention instrumentation for collecting interaction data, yet the actual use of this data for improving learning remains superficial. Hadiyana et al. (2026) describe data collection primarily for system performance monitoring rather than learning insight. Wibisono et al. (2024) evaluate user experience through HEART metrics but do not connect these metrics to personalization algorithms. The field has accumulated considerable data about what learners do but has not developed the analytical frameworks to understand what these behaviors mean for learning or how to respond intelligently.

The absence of adaptive content sequencing is particularly striking. Rusak (2017) provide the sole example of algorithmic learning path personalization. No other studies examine how microlearning sequences might adapt to learner performance, misconceptions, or developing mental models. The field has not investigated whether adaptive sequencing improves learning outcomes compared to fixed sequences, whether different adaptation strategies suit different competency types, or how to balance learner control with algorithmic guidance in software engineering domains where prerequisite relationships may be complex and non-linear.

Intelligent feedback represents another underdeveloped area. Automated assessment systems like those in Skalka and Drlik (2020) provide correctness feedback on code submissions, but this feedback is binary and generic rather than adaptive and explanatory. The systems do not diagnose specific misconceptions, suggest alternative approaches, or adapt feedback based on learner history. The gap between automated assessment and intelligent tutoring reflects a broader limitation: the field has implemented automation without intelligence, feedback without adaptation.

The AI and personalization literature reveals a troubling pattern of technological solutionism—assuming that AI capabilities will automatically translate into educational benefits without careful design, implementation, and evaluation. Studies invoke AI as a marker of innovation while implementing rule-based systems or data collection without adaptive action. The rhetorical appeal of AI may be outpacing substantive integration, creating expectations that implemented systems cannot meet.

Furthermore, the absence of rigorous evaluation of AI-enhanced microlearning is nearly complete. No studies compare adaptive personalization against non-adaptive alternatives using experimental designs. No studies examine whether AI-generated content supports learning as effectively as human-designed materials. No studies investigate whether learning analytics dashboards help instructors provide better support or whether students benefit from seeing their own analytics data. The field has not established the value proposition for AI integration because it has not conducted the studies that would demonstrate value.

The ethical and privacy implications of AI-enabled learning analytics receive no attention whatsoever. Studies describe collecting detailed interaction data without discussing consent, data governance, algorithmic bias, or student autonomy. The field has not grappled with questions about who owns learning data, how algorithms might disadvantage certain learner populations, or whether predictive analytics might label students in ways that shape instructor expectations and student self-perceptions. The enthusiastic pursuit of AI integration proceeds without the ethical framework that should accompany it.

The personalization research also reveals limited conceptualization of what personalization might mean for software engineering learning. Rusak (2017) personalize based on background knowledge, but personalization could address many dimensions: learning style preferences, interest areas, career goals, common misconceptions, debugging strategies, or design approaches. The field has not developed a taxonomy of personalization dimensions or investigated which dimensions matter most for different learning objectives. Personalization remains an undifferentiated concept rather than a design space with meaningful choices.

The integration of AI for content recommendation, as in Hadiyana et al. (2026) and Rusak (2017), raises questions about whether recommendation algorithms optimized for engagement might inadvertently narrow learner exposure. If algorithms recommend content similar to what learners have already engaged with, they may create filter bubbles that prevent exploration of diverse topics or approaches. The field has not investigated whether personalized recommendation supports breadth of learning or inadvertently constrains it.

Finally, the relationship between AI personalization and learner autonomy remains unexamined. Software engineering education aims to develop professionals who can self-direct their learning, identify knowledge gaps, and seek appropriate resources. If AI systems increasingly handle these metacognitive functions, do learners develop the self-regulation capabilities they will need throughout their careers? Or does intelligent personalization scaffold metacognitive development by modeling effective learning decisions? The field has not investigated whether AI personalization supports or undermines the development of lifelong learning capability.

In sum, the AI and personalization literature reveals a field rich with aspiration and poor on evidence. The implemented systems demonstrate rule-based logic rather than machine learning, data collection without adaptive action, and prediction without intervention. The conceptual proposals outline ambitious capabilities without implementation or evaluation. The ethical dimensions remain entirely unaddressed. The next generation of research must move beyond rhetorical AI to implemented intelligence, beyond prediction to adaptation, beyond data collection to learning insight, and beyond technological solutionism to rigorous evaluation. Without such progress, the promise of AI-enabled personalized microlearning will remain exactly that—promise, unfulfilled.

5.6. Design Frameworks and Instructional Models (Addressing RQ6)

The design frameworks and instructional models guiding microlearning development in software engineering education reveal a field that has borrowed eclectically from established instructional design traditions while making limited progress toward frameworks tailored to microlearning's distinctive affordances and challenges. Existing research has successfully mapped microlearning activities to established educational taxonomies including Bloom's taxonomy and Fuller's matrix, validated domain-specific frameworks such as the CME (Concerns, Micro-learning, and Examples) approach for framework learning, and demonstrated the applicability of systematic design models like ADDIE to microlearning content development. The corpus demonstrates awareness of design principles but exposes significant gaps in how these principles are operationalized, validated, and adapted to software engineering's unique characteristics.

An (2022) provides the most explicit engagement with formal instructional design, employing the ADDIE model (Analysis, Design, Development, Implementation, Evaluation) as the framework for developing microlearning content for software engineering courses. This represents a theoretically sound approach—ADDIE is among the most established instructional design frameworks—yet the paper remains at the proposal stage, describing

planned development, validation, and testing rather than reporting implemented results. The field thus has a model for how microlearning could be systematically designed but no evidence about whether ADDIE-guided design produces more effective microlearning than alternative approaches. Furthermore, the paper does not address whether ADDIE, originally developed for traditional instructional contexts, requires adaptation for microlearning's fragmented, just-in-time, mobile-oriented delivery. The assumption that established frameworks transfer unmodified may be unwarranted.

Skalka and Drlík (2018) offer one of the more theoretically grounded contributions, mapping their conceptual microlearning framework to Bloom's taxonomy and Fuller's matrix. Their two-level educational model (Introductory Courses and Technology Courses) explicitly addresses how microlearning activities can progress through cognitive levels from Remembering to Creating. This represents genuine design thinking—using established educational taxonomies to structure learning progression. The mapping demonstrates that microlearning can theoretically support higher-order thinking, not merely declarative knowledge acquisition. Yet the framework remains conceptual, and the mapping to Bloom's taxonomy describes intentions rather than demonstrated outcomes. The critical question—whether learners actually achieve creation-level capabilities through microlearning sequences—remains unanswered because the framework has not been implemented and evaluated for higher-order outcomes.

Correa Botero (2015) and Correa et al. (2018) develop and validate a framework more specific to software engineering's distinctive needs. The CME approach (Concerns, Micro-learning, and Examples) addresses a fundamental challenge in software engineering learning: how to help developers learn frameworks efficiently by organizing content around authentic developer concerns rather than framework documentation structure. The framework demonstrates design thinking grounded in software engineering's unique characteristics—the separation of concerns principle from software design is applied to learning design. The quasi-experimental evaluation showing significant time reduction (approximately 40%) and task completion improvements provides evidence that this domain-specific framework outperforms traditional approaches. Yet the framework remains narrowly focused on framework learning rather than broader software engineering competencies. The question of whether concern-based decomposition can scale to entire software engineering curricula remains unexplored.

The design frameworks reveal a field struggling with the fundamental tension between coherence and fragmentation. Microlearning by definition fragments content into small units, yet software engineering understanding requires integrating these fragments into coherent mental models of systems, processes, and practices. The frameworks provide limited guidance on how to maintain coherence across fragmented episodes. Skalka and Drlík (2018) address this partially through their two-level model distinguishing introductory and technology courses, but they do not specify mechanisms for connecting microlearning units into integrated understanding. The field lacks design principles for what might be called "macro-structures for micro-content"—the scaffolding that helps learners assemble discrete learning episodes into cohesive knowledge architectures.

The absence of attention to learning progression across microlearning episodes represents a significant design gap. The frameworks specify how to design individual microlearning units but provide limited guidance on how units should sequence, how difficulty should progress, how concepts should spiral and revisit, or how learners should develop increasingly sophisticated understanding over time. Skalka and Drlík (2018) Bloom's taxonomy mapping addresses cognitive progression but does not specify content progression—which concepts should precede which, how prerequisite relationships should be handled, or how

learners should integrate related concepts across episodes. The field has not developed design models for microlearning curricula, only for microlearning units.

Gasca-Hurtado et al. (2024) and Gasca-Hurtado and Gómez-Álvarez (2021) embed microlearning within gamification frameworks, demonstrating attention to motivational design. Their Mr. Lottery application transforms traditional lottery mechanics into educational experiences, showing how game elements can structure microlearning engagement. Yet the framework focuses primarily on engagement rather than learning progression. The finding that motivation increased significantly while learning gains did not suggests that motivational design frameworks may need better integration with cognitive design frameworks. Engagement without learning is insufficient, yet the field lacks frameworks that systematically address both dimensions.

The instructional models reveal limited attention to the social dimensions of learning. Most frameworks treat microlearning as individual activity—learners alone with content. Jwo (2015) provides a notable exception through comics and micro-movie creation projects that involve interviewing industry mentors, scripting, and collaborative film production. This project-based framework incorporates social learning, authentic context, and collaborative creation. Yet it represents microlearning only in the sense of micro-movie production—the learning activities are sustained and complex rather than bite-sized. The field has not developed frameworks that integrate social learning into genuinely micro-sized episodes, such as collaborative debugging of shared code snippets, paired programming within brief sessions, or social annotation of learning content.

The frameworks also reveal limited attention to mobile-specific design considerations. Hadiyana et al. (2026) and Wibisono et al. (2024) evaluate mobile implementations but do not articulate mobile-specific design frameworks. The field lacks principles for designing microlearning that leverages mobile affordances—context awareness, location sensitivity, camera integration, push notifications, offline access, or cross-device synchronization. The frameworks treat mobile as delivery channel rather than design opportunity, missing possibilities for context-aware learning that responds to where learners are and what they are doing.

Assessment design within frameworks receives surprisingly limited attention. The frameworks describe content design, sequencing, and engagement but provide minimal guidance on how to assess learning within microlearning architectures. Skalka and Drlik (2020) incorporate automated assessment, but as a separate component rather than integrated within the instructional model. The field lacks frameworks that specify how assessment should be distributed across microlearning episodes, how formative feedback should function, how mastery should be determined, or how assessment data should inform subsequent learning design.

The validation of design frameworks is almost entirely absent. While individual studies evaluate the products of framework-guided design (specific courses, platforms, or interventions), none validate the frameworks themselves. The field cannot answer whether following Framework A produces better outcomes than Framework B, whether certain frameworks suit certain competency types better than others, or whether framework adherence predicts learning effectiveness. The frameworks function as design heuristics rather than evidence-based models.

The frameworks also reveal limited attention to instructor roles and practices. How should instructors design microlearning sequences? How should they monitor learner progress across fragmented episodes? How should they intervene when learners struggle? How should they integrate microlearning with other instructional activities? The frameworks describe learner experiences but not instructor practices, assuming that microlearning either replaces or marginalizes instructor roles rather than transforming them.

The field has not developed frameworks for instructor professional development or instructional practice in microlearning contexts.

Finally, the cultural and contextual assumptions embedded in design frameworks receive no critical examination. Frameworks developed in European and Latin American contexts (Gasca-Hurtado et al., 2024; Skalka & Drlík, 2018) may embed assumptions about learner autonomy, instructor authority, or educational values that do not transfer across cultural contexts. The frameworks are presented as universal but have not been tested for cross-cultural validity. The field has not investigated whether microlearning design principles require adaptation for different educational cultures, learner populations, or institutional contexts.

In sum, the design frameworks literature reveals a field with sophisticated design thinking but limited framework validation. Established instructional models are borrowed and combined creatively, domain-specific frameworks like CME show promise, and taxonomies provide theoretical grounding. Yet the field lacks comparative framework evaluation, attention to coherence across episodes, integration of social learning, mobile-specific principles, assessment design guidance, instructor role specification, and cultural validation. The next generation of research must move from framework proposal to framework validation, from individual unit design to curricular architecture, and from implicit assumptions to explicit testing. Without such progress, microlearning design will remain an art guided by heuristics rather than a science informed by evidence.

5.7. Integrating Technical and Professional Competencies (Addressing RQ7)

The integration of technical and professional competencies through microlearning represents one of the most rhetorically acknowledged yet practically underdeveloped areas in the corpus. Studies confirm that standalone microlearning modules can improve self-reported soft skills including teamwork, leadership, communication, and emotional intelligence across diverse student populations, while project management courses demonstrate that microlearning can simultaneously address both methodological knowledge and team coordination competencies within a single domain. While multiple studies acknowledge the importance of soft skills for software engineering professionals, critical examination reveals that integration remains largely aspirational—professional competencies are typically addressed in standalone modules rather than woven into technical learning, and the relationship between technical and professional development remains undertheorized and unevaluated.

Luo and Li (2025) provide the most comprehensive treatment of professional competencies, targeting teamwork, leadership, communication, time management, and emotional intelligence through six structured microlearning modules delivered over six weeks. Their finding that microlearning significantly improved all five soft skills, with differential effects across disciplines (leadership modules most impactful for technical and engineering students, emotional intelligence most valuable for business students), demonstrates that professional competencies can indeed be developed through microlearning formats. Yet the fundamental integration question remains unaddressed: these modules exist separately from technical learning rather than integrated within it. Students develop communication skills in dedicated modules while presumably developing coding skills elsewhere, with no mechanism for connecting these developments. The study cannot speak to whether communication skills developed in isolation transfer to technical communication contexts—code reviews, technical documentation, requirements discussions—because it does not examine transfer.

Hadiyana et al. (2026) similarly treat soft skills as standalone modules, developing AI-enabled mobile microlearning for communication, collaboration, problem-solving, and

time management. Their prototype with ten modules demonstrates feasibility and shows promising completion rates (66–78%) and quiz scores (72–81/100). Yet again, the soft skills content exists in parallel to rather than integrated with technical content. The study does not address how learners might apply collaboration skills during pair programming or how problem-solving strategies might transfer from general modules to specific debugging challenges. The integration challenge—designing learning experiences where technical and professional competencies develop simultaneously through the same activities—remains unaddressed.

Suciu (2022) makes partial progress toward integration by including both hard skills (project administration) and soft skills (team coordination, facilitation, communication) within software project management courses. This represents a domain where technical and professional competencies inherently intertwine—project management requires both methodological knowledge and interpersonal capability. The finding that microlearning and gamification increased student engagement, performance, and knowledge retention suggests that integrated approaches can be effective. Yet the study does not examine the integration itself—whether students who develop both competency types simultaneously demonstrate different outcomes than those who develop them separately, or whether the instructional design explicitly connects technical and professional learning. Integration is assumed by domain rather than designed and evaluated.

Gasca-Hurtado et al. (2024) and Gasca-Hurtado and Gómez-Álvarez (2021) address Scrum framework learning, which similarly inhabits the intersection of technical and professional competencies. Scrum involves roles, events, and artifacts (technical/procedural knowledge) while fundamentally depending on team collaboration, communication, and facilitation (professional competencies). Their gamified microlearning approach successfully teaches Scrum concepts, as evidenced by motivation increases and certification exam performance. Yet the studies focus on conceptual knowledge rather than the collaborative capabilities that Scrum actually requires in practice. Students learn what Scrum is without developing capability to do Scrum—facilitate events, navigate team dynamics, resolve conflicts, or collaborate effectively. The technical knowledge is assessed; the professional capability is not.

Jwo (2015) offers one of the more innovative approaches to integration through comics and micro-movie creation projects. Students interview industry mentors and create short films based on software engineering challenges, simultaneously engaging with technical content (requirements, design, architecture, quality) and developing professional capabilities (communication, collaboration, project management). The survey findings—students agreed that projects helped explore software engineering issues (avg. 4.2/5) and that watching peers' movies helped understanding (avg. 4.1/5)—suggest that project-based integration can be engaging and effective. Yet the study relies entirely on self-report measures and does not assess either technical or professional competency development objectively. Whether students actually developed deeper technical understanding or enhanced professional capabilities remains unknown.

The microcredential literature addresses integration aspirations without providing integration evidence. Urem et al. (2025) describe microcredentials covering both technical competencies (Java programming, database design, cloud computing) and presumably professional competencies, though the latter are not explicitly detailed. Guzmán Rojas (2024) mentions that the MISO curriculum addresses both technical and professional dimensions, but the study focuses on curricular flexibility and alternative credentials rather than integration effectiveness. Cambarieri et al. (2025) include "social/professional aspects" in their microcredential categories but provide no detail on what these encompass or how they are assessed. The credentialing frameworks acknowledge that professional

competencies matter; they do not demonstrate that microcredentials effectively develop or certify them.

The conceptual frameworks for integration are notably absent from the literature. No studies propose or validate design principles for creating microlearning experiences where technical and professional competencies develop synergistically. Questions that should guide design remain unanswered: How can debugging exercises incorporate communication skill development? How can architecture design activities foster collaboration capabilities? How can testing tasks develop attention to user needs and perspectives? How can code review microlearning build both technical quality and interpersonal feedback skills? The field lacks frameworks that would enable intentional integration rather than accidental co-occurrence.

The assessment of integrated competencies presents particular challenges that the literature does not address. If technical and professional competencies develop together, assessment should presumably capture both dimensions and their interaction. Yet the studies assess either technical knowledge (Gasca-Hurtado et al. (2024), using Scrum certification exams) or self-reported professional skill development (Luo & Li, 2025) but never both in relation. The field has not developed assessment instruments that capture integrated capability—the quality of technical communication, the technical soundness of collaborative decisions, the process awareness demonstrated during problem-solving. Without such instruments, claims about integration remain unsubstantiated.

The temporal relationship between technical and professional development receives no attention. Should professional competencies precede technical learning, follow it, or develop concurrently? Does communication skill development before technical training produce different outcomes than integrated development? Do professional competencies decay faster than technical knowledge without reinforcement? The studies treat professional development as a discrete intervention rather than a longitudinal process, providing no evidence about sequencing or sustainability.

The transfer question—whether professional competencies developed in isolation transfer to technical contexts—remains entirely unexamined. Students who complete communication modules may communicate more effectively during code reviews, or they may not. The modules teach general communication principles; technical communication has domain-specific characteristics—precision about technical concepts, clarity about system behavior, diplomacy about code quality—that may not be addressed. The field has not investigated whether general soft skills training transfers to technical contexts or whether domain-specific professional development is necessary.

The collaboration between technical and professional competency development in team-based learning receives minimal attention. Most microlearning implementations assume individual learners working alone, yet professional competencies inherently involve interaction with others. Jwo (2015) micro-movie projects involve collaboration, but the microlearning format here is stretched—projects are sustained rather than bite-sized. The field has not developed microlearning formats that support collaborative development of professional competencies within genuinely micro-sized episodes—paired debugging sprints, collaborative design exercises, or brief team retrospectives.

The industry perspective on integrated competencies is notably absent. Employers consistently report that graduates lack professional competencies—communication, teamwork, problem-solving—yet possess adequate technical foundations. The studies do not examine whether microlearning-developed professional competencies address these employer concerns, whether employers perceive differences between integrated and standalone development, or whether microcredentialled professional competencies influence hiring decisions. The integration rhetoric claims industry relevance without industry evidence.

Finally, the critical question of whether professional competencies can be developed through microlearning remains unanswered by the current evidence. Luo and Li (2025) demonstrate self-reported improvement, but self-report is not behavior change. Professional competencies involve dispositional and behavioral dimensions—actually communicating effectively, actually collaborating productively—that may require sustained practice, feedback, and reflection beyond what microlearning episodes can provide. The field has not established whether microlearning can produce genuine professional capability development or merely knowledge about professional capabilities. The distinction between learning that communication matters and learning how to communicate effectively is precisely the integration challenge, and the current literature cannot speak to it.

In sum, the integration of technical and professional competencies through microlearning remains more aspiration than achievement. Studies address either technical or professional development but rarely both in relation. Standalone modules dominate while integrated designs remain exceptional. Assessment captures either dimension but not their interaction. Frameworks for intentional integration are absent. Transfer remains unexamined. The next generation of research must move beyond acknowledging that both competency types matter to designing, implementing, and evaluating microlearning that develops them synergistically. It must develop assessment instruments that capture integrated capability, investigate transfer from isolated modules to technical contexts, and engage industry perspectives on what integrated competence actually means. Without such progress, the integration rhetoric will remain disconnected from educational reality.

5.8. Implementation Challenges and Contextual Factors (Addressing RQ8 and RQ9)

The literature on implementation challenges and contextual factors reveals a field that has documented practical obstacles while remaining largely silent on deeper structural, cultural, and equity dimensions that may determine microlearning's long-term viability. Studies across vocational, undergraduate, and professional contexts have identified common implementation challenges including credential recognition barriers, faculty access issues, student preferences for concise mobile-accessible materials, and the need for technical infrastructure capable of supporting institutional-scale deployment, with preliminary evidence suggesting that information quality and usability may be more critical than system sophistication in resource-constrained settings. The studies acknowledge barriers at the surface level—technical limitations, resource constraints, institutional resistance—yet fail to examine how these challenges interact, how they vary across contexts, and whether they can be overcome through design or require fundamental rethinking of microlearning's role.

Guzmán Rojas (2024) identifies several implementation challenges from the MISO master's program experience, including the need for greater recognition of alternative credentials and ensuring consistent access for faculty. These are real and significant barriers, yet they are presented as logistical problems requiring administrative solutions rather than as symptoms of deeper tensions between microlearning and traditional educational structures. The credential recognition challenge reflects not merely employer unfamiliarity but fundamental questions about what kinds of learning should be credentialed, who should do the credentialing, and how traditional degrees and microcredentials should relate. The faculty access issue similarly reflects not just scheduling problems but questions about faculty roles, expertise, and autonomy in increasingly modularized curricula. The study identifies challenges without examining their structural roots. The credential recognition challenge reflects not merely employer unfamiliarity but fundamental questions about which learners' credentials gain recognition. Guzmán Rojas (2024) notes the need for greater recognition of alternative credentials, yet the study does not examine whether recognition varies by credential issuer (elite university versus vocational provider), by

learner demographic characteristics, or by the economic sectors into which different learner populations seek entry. The equity dimension of credential recognition—who gets their credentials recognized, under what conditions, and at what labor market value—remains entirely unexplored.

Urem et al. (2025) describe the implementation of microcredentials at Sibenik University, noting the importance of long-term industry collaboration and participation in European projects. The implication is that successful implementation requires extensive institutional infrastructure, external partnerships, and project funding—resources that may not be available to less privileged institutions. The study does not address what implementation looks like for institutions without European project access, without established industry partnerships, or without the organizational capacity to develop in-house microcredentials. The implementation model described may be feasible for well-resourced European universities while remaining inaccessible to institutions in developing contexts, underfunded systems, or regions without robust industry presence.

Wigati et al. (2024) conduct a needs analysis at a vocational high school in Indonesia, finding that students need concise, engaging, and easy-to-understand teaching materials accessible via mobile devices, and that current materials are perceived as "too many and complicated." This represents one of the few studies attending to learner perspectives on implementation challenges. Yet the analysis identifies needs without examining systemic barriers to meeting them. Why are current materials too numerous and complicated? What institutional, curricular, or resource constraints produce this situation? How might microlearning implementation address these root causes rather than merely treating the symptom of student dissatisfaction? The analysis stays at the surface, failing to examine how students' socioeconomic backgrounds, device access, or home internet connectivity might shape their ability to benefit from mobile microlearning. The finding that students prefer mobile-accessible materials assumes access; it does not interrogate whether access is equitably distributed.

The studies reveal striking variation in implementation contexts—vocational high schools in Indonesia (Wigati et al., 2024; Wira et al., 2026), undergraduate programs in Latin America (Gasca-Hurtado et al., 2024), European universities (Skalka & Drlik, 2020; Suciu, 2022), and professional education in Colombia (Guzmán Rojas, 2024). Yet no studies systematically compare implementation across contexts. The field cannot answer whether implementation challenges identified in Indonesian vocational schools generalize to European universities, whether strategies successful in Latin American undergraduate contexts transfer to professional education, or whether contextual factors like class size, technological infrastructure, or educational culture moderate implementation effectiveness. The studies treat context as setting rather than variable.

The technological infrastructure challenges receive attention primarily in studies from developing contexts. Hadiyana et al. (2026) conduct system simulations indicating their architecture is ready for medium-scale classes ($\leq$300 concurrent users), implicitly acknowledging scalability as a concern. Jo et al. (2025) explicitly position their study in the Philippines as examining microlearning effectiveness "in resource-constrained settings". Their finding that Information Quality and System Usability significantly influenced User Satisfaction while System Quality did not, provides insights relevant to contexts where sophisticated technical infrastructure may be less important than content and usability. Yet these contextual insights remain disconnected from systematic equity analysis. Jo et al. (2025) do not examine whether students with the most constrained access—those without reliable electricity, those sharing devices with family members, those balancing paid employment with study—experienced different outcomes than their more privileged peers. The study identifies that usability matters more than sophistication in resource-

constrained settings but does not investigate whether the intervention reaches or benefits the most constrained learners.

The student population comparisons that RQ9 calls for are almost entirely absent from the literature. Luo and Li (2025) provide the sole example comparing responses across different student populations, examining microlearning effectiveness for soft skills development across Humanities and Arts, Business Studies, Medical Sciences, and Technical and Engineering students. Their finding that leadership modules were most impactful for technical and engineering students while emotional intelligence training was most valuable for business students, and that no single group showed significantly higher gains than others, suggests that microlearning can be universally effective while producing differential patterns of impact. Yet this study compares disciplinary backgrounds rather than the educational levels, prior experiences, or professional aspirations that might matter more for software engineering education specifically. The field cannot answer whether vocational students respond differently to microlearning than undergraduates, whether professionals benefit from different designs than students, or whether novice and experienced learners require different microlearning approaches.

Skalka and Drlik (2020) and Skalka et al. (2020) examine at-risk students specifically, using automated assessment and microlearning engagement as predictors for identifying students likely to struggle in introductory programming courses. This represents attention to student subgroup differences, but the focus is prediction rather than differential response to intervention. The studies identify which students are at risk but do not examine whether at-risk students respond differently to microlearning than their peers, whether microlearning interventions are equally effective across ability levels, or whether certain microlearning designs might particularly benefit struggling learners. The prediction capability exists without corresponding instructional adaptation.

The implementation challenges related to curriculum integration receive minimal attention. Studies describe microlearning implementations within specific courses but do not address how multiple microlearning experiences across a curriculum might cohere, how microlearning might articulate with traditional instruction, or how institutions might manage the transition from conventional to microlearning-enhanced curricula. An (2022) proposes a development model but does not address integration with existing courses. The field lacks guidance on whether microlearning should supplement, replace, or transform traditional software engineering education, and what implementation challenges each approach entails.

Faculty development and acceptance represent another underexamined challenge. Guzmán Rojas (2024) mentions faculty access issues but does not explore faculty attitudes, capabilities, or professional development needs. Implementing microlearning requires faculty to design new materials, adopt new pedagogical approaches, and potentially relinquish some control over learning sequences. The studies do not examine whether faculty possess the design skills needed, whether they resist modularization of their courses, or how institutions can support faculty through implementation transitions. The faculty perspective is almost entirely absent from the literature.

The cost and sustainability challenges remain unaddressed. Developing high-quality microlearning content with video, interactive elements, and assessment requires significant investment. Maintaining and updating content as technologies evolve requires ongoing resources. The studies do not examine development costs, maintenance requirements, or sustainability models. Institutions considering microlearning adoption cannot estimate whether benefits justify investments or whether they can sustain implementations over time. The economic dimension of implementation is invisible.

The equity implications of microlearning implementation receive no attention. If microlearning requires mobile devices and reliable internet access, learners without such resources may be excluded. If microlearning assumes self-directed learning capability, learners from educational cultures emphasizing direction may struggle. If microlearning content reflects cultural assumptions about communication, collaboration, or learning, learners from different cultural backgrounds may find it less accessible. The studies do not examine who microlearning implementations include and exclude, whether they widen or narrow educational opportunity gaps, or how contextual factors like socioeconomic status, cultural background, or prior educational experience moderate implementation effectiveness.

The scalability challenges beyond technical infrastructure—scaling instructional support, scaling assessment, scaling quality assurance—are not addressed. Hadiyana et al. (2026) simulate technical scalability for 300 concurrent users but do not address how instructors would support 300 learners progressing through personalized microlearning paths, how assessment quality would be maintained at scale, or how content would be updated across large implementations. The pedagogical scalability challenges may be more significant than technical ones, yet they remain unexamined.

Finally, the temporal dimension of implementation—how microlearning implementations evolve over time, how they respond to changing conditions, how they survive faculty turnover or leadership changes—receives no attention. The studies capture implementations at single time points or over single semesters. The field has no longitudinal implementation research examining whether microlearning initiatives persist, adapt, or fade; what factors support sustainability; or how implementations mature as institutions gain experience. Implementation is treated as event rather than process.

In sum, the implementation challenges and contextual factors literature reveals a field that has identified practical barriers while remaining blind to structural, cultural, and equity dimensions. The studies document infrastructure needs, credential recognition problems, and student preferences but do not examine how these challenges interconnect, how context shapes them, or how they might be systematically addressed. The student population comparisons that could inform differentiated design are almost entirely absent. The faculty perspective, economic sustainability, equity implications, and longitudinal implementation dynamics remain unexplored. The next generation of research must move from documenting isolated challenges to understanding implementation as complex system, from treating context as setting to investigating context as variable, and from assuming universal applicability to examining differential effectiveness across populations. Without such understanding, implementation guidance will remain superficial and microlearning's potential will be realized unevenly across the contexts that need it most.

5.9. Theoretical Alignment: Microlearning and Adult Learning Principles

The microlearning interventions reviewed in this corpus invite systematic examination through the lens of adult learning theories introduced earlier—specifically Knowles' andragogy, Lave and Wenger's situated learning, and Zimmerman's self-regulated learning. While these frameworks were acknowledged in the Introduction, their explanatory power for understanding microlearning's affordances and limitations in software engineering education warrants explicit analysis against the empirical findings.

5.9.1. Andragogical Alignment

Knowles' (1984) assumptions about adult learners—that they are self-directed, draw on prior experience, are ready to learn what they need to know, and prefer problem-centered over content-oriented learning—find partial expression in the reviewed implementations (Knowles, 1984). The just-in-time, task-focused character of microlearning aligns

with adults' orientation toward immediate application: Correa Botero (2015) concern-based framework explicitly organizes learning around developer problems rather than abstract content sequences, demonstrating recognition that professional learners seek knowledge applicable to authentic challenges. Similarly, the self-paced, on-demand access characterizing platforms like Mimo and SoloLearn (Wibisono et al., 2024) supports learner autonomy. However, the corpus reveals limited attention to learners' prior experience as a design resource. Rusak (2017) provide a notable exception, clustering students into profiles based on background knowledge to personalize learning paths, but most interventions treat learners as uniform novices, neglecting andragogy's emphasis on leveraging existing expertise. The field has yet to systematically investigate how microlearning might differentiate instruction based on professional experience—a significant gap given software engineering's diverse learner populations spanning vocational students through experienced practitioners.

5.9.2. Situated Learning and Legitimate Peripheral Participation

Lave and Wenger's (1991) framework posits that learning is inseparable from the social contexts and practices of professional communities (Lave & Wenger, 1991). This lens reveals a fundamental tension in microlearning implementations. The format's emphasis on individual, decontextualized skill acquisition runs counter to situated learning's core insight that competence develops through participation in communities of practice. Jwo (2015) comics and micro-movie projects represent the corpus's most explicit engagement with situated learning, having students interview industry mentors and create artifacts reflecting authentic software engineering challenges. Yet even here, the microlearning format stretches to accommodate what is essentially sustained, collaborative project work. The studies conspicuously lack attention to how microlearning might support legitimate peripheral participation—perhaps through micro-episodes that analyze authentic workplace artifacts, brief case studies of professional practice, or social features connecting learners with practitioners. The individualistic framing dominating the literature suggests that microlearning, as currently implemented, may better support discrete skill acquisition than the socialization into professional practice that situated learning theorizes as central to expertise development.

5.9.3. Self-Regulated Learning

Zimmerman's (2002) cyclical model of forethought, performance, and self-reflection provides a framework for examining whether microlearning supports metacognitive development (Zimmerman, 2002). The ideal microlearner would set goals, monitor comprehension, select appropriate strategies, and reflect on outcomes. However, the reviewed studies largely treat learners as consumers of content rather than architects of their own learning. Personalization, where present, is algorithmically directed (Rusak, 2017) rather than supporting learners' self-regulatory capabilities. The platforms collect extensive analytics but rarely make these visible to learners for self-monitoring purposes. No studies examine whether microlearning experiences transfer to improved self-regulation in subsequent learning or professional practice. This absence is particularly problematic given software engineering's requirement for continuous learning across a career—if microlearning does not develop learners' capacity to direct their own learning, its long-term value may be limited to immediate skill acquisition rather than fostering the lifelong learning capability the profession demands.

Implications for design and research. This theoretical analysis suggests that microlearning's current implementation patterns align unevenly with established adult learning principles. The alignment with andragogical preferences for immediacy and self-direction is strong; the alignment with situated learning's emphasis on social practice and community

participation is weak; the alignment with self-regulated learning's developmental aims is almost entirely absent. Future design should explore: (1) how microlearning might incorporate learners' prior experience as a design variable rather than treating all learners as novices; (2) whether social features can connect individual microlearning episodes to professional communities without undermining brevity; and (3) how analytics and feedback might be repurposed to support metacognitive development rather than merely tracking consumption. The theoretical frameworks introduced in this review's opening should not remain decorative—they should actively inform design choices and research questions, ensuring that microlearning serves adult professionals in ways consistent with how adults actually learn.

6. Conclusions

This systematic review reveals a field in a state of creative ferment yet marked by significant fragmentation and blind spots. The evidence confirms that microlearning can engage students and convey discrete programming knowledge, with project-based and gamified approaches showing particular promise for motivation. However, the concentration of research on introductory coding and declarative project management concepts exposes a critical gap: the core competencies of software engineering—architectural thinking, requirements negotiation, testing strategies, and system evolution—remain largely unexplored. The field has gravitated toward what is easiest to measure rather than what is most consequential for professional practice, leaving unanswered whether microlearning can support the integrated, higher-order thinking that distinguishes expert software engineers from novice programmers.

The enthusiasm surrounding microcredentials and AI-personalized learning considerably outstrips the evidence supporting them. Institutional case studies demonstrate that microcredentials can be implemented, but their actual value for employability, employer recognition, and competency validation remains unsubstantiated. Similarly, AI integration largely consists of rule-based recommendations and analytics dashboards rather than adaptive intelligence, with no rigorous evidence that personalization improves learning outcomes over well-designed fixed sequences. This evidence gap presents a significant downside to premature adoption. Institutions risk investing substantial resources in developing and marketing microcredential programs or AI-driven platforms based on promise rather than proof, potentially diverting attention and funding from evidence-based pedagogical improvements. Furthermore, an overemphasis on easily-credentialed technical skills could inadvertently devalue the foundational, durable knowledge and integrated professional judgment that traditional degree programs are designed to cultivate. A critical, cautious approach is therefore warranted: institutions should pause before committing to microcredential-only pathways or AI-driven personalization at scale, and instead prioritize rigorous, comparative research to validate whether these innovations deliver on their educational and employment promises.

The field's methodological fragilities compound these concerns: overreliance on self-reports, absence of longitudinal and transfer studies, and lack of comparative effectiveness research mean that positive findings must be interpreted with caution. The null results for microlearning units in controlled studies should unsettle advocates, suggesting that engagement does not guarantee learning. Moving forward, the research agenda must shift from asking whether microlearning can work to specifying for which competencies, under which conditions, and for which learners it is most valuable. Designers must confront the coherence challenge: how to assemble fragmented micro-episodes into integrated professional understanding. Educators should treat microlearning as a strategic supplement for just-in-time skill acquisition rather than a comprehensive solution for software

engineering formation. Without rigorous longitudinal designs, validated assessment of higher-order competencies, and critical examination of credentialing and AI claims, the field risks building a future on promise rather than proof. The potential of microlearning to support lifelong learning in a rapidly evolving discipline remains genuine, but realizing that potential requires moving beyond enthusiasm to evidence.

Funding: This research was supported in part by "Universidad de Guayaquil" under Project FCI-021-2024, and in part by "Universidad Bolivariana del Ecuador" under PROY-UBE-2024-019.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ADDIE	Analysis, Design, Development, Implementation, Evaluation
AI	Artificial Intelligence
CME	Concerns, Micro-learning and Examples
GOAL	Gamification-Oriented Agile Learning (framework)
HEART	Happiness, Engagement, Adoption, Retention, Task Success
ICT	Information and Communications Technology
IoT	Internet of Things
ISSM	Information System Success Model
LMS	Learning Management System
MISO	Maestría en Ingeniería de Software (Master's in Software Engineering)
MOOC	Massive Open Online Course
MVC	Model-View-Controller
N-Gain	Normalized Gain
PBL	Project-Based Learning
R&D	Research and Development
RQ	Research Question
SUS	System Usability Scale
UX	User Experience
VC	Verifiable Credentials
VPL	Virtual Programming Lab
W3C	World Wide Web Consortium
WAF	Web Application Framework

Appendix A. Adapted MERSQI for Microlearning in Software Engineering Education (MERSQI-MSE)

It is a quality appraisal tool modified from the original Medical Education Research Study Quality Instrument (Jaros & Beck Dallaghan, 2024) to address the unique characteristics of microlearning interventions and software engineering competencies.

Appendix A.1. Rationale for Adaptation

The original Medical Education Research Study Quality Instrument (MERSQI) was designed for medical education research. This adaptation modifies items to address:

- The unique characteristics of microlearning interventions (bite-sized, technology-mediated, often self-paced)

- Software engineering competencies (programming, architecture, testing, project management)
- The technological platforms common in this domain (LMS, mobile apps, automated assessment tools)
- The blend of technical and professional skills required in software engineering

Appendix A.2. Scoring Instructions

Rate each study on the following items. Calculate total score by summing all domain scores (maximum = 18). Two independent reviewers should score each study, with discrepancies resolved through consensus discussion.

DOMAIN 1: STUDY DESIGN (Maximum 3 points)

Score	Criterion	Description for Microlearning in SE Education
3	Randomized controlled trial	Participants randomly assigned to microlearning intervention vs. control (traditional instruction, alternative microlearning design, or no intervention)
2	Quasi-experimental with comparison	Non-randomized comparison group (e.g., cohort comparison, concurrent non-randomized control)
1	Single-group pre-post	No comparison group; pre-test and post-test only
0	Single-group post only	No pre-test, no comparison group

Notes:

- For microlearning platform comparisons (e.g., Mimo vs. SoloLearn), if participants are randomly assigned to platforms, score as 3
- For studies comparing microlearning to traditional lectures with non-random assignment, score as 2

DOMAIN 2: SAMPLING (Maximum 3 points)

2a: Sampling—Institutions (Maximum 1.5 points)

Score	Criterion	Description
1.5	Multi-institution	Participants from two or more institutions (universities, vocational schools, companies)
1	Single institution	Participants from one institution
0.5	Not specified	Sampling method unclear

2b: Sampling—Response Rate (Maximum 1.5 points)

Score	Criterion	Description
1.5	Response rate ≥75% or sample size justification with power analysis	≥75% of eligible participants enrolled, OR authors provide power analysis demonstrating adequate sample size
1	Response rate 50–74%	50–74% of eligible participants enrolled
0.5	Response rate <50% or not reported	Less than 50% response rate, or response rate not reported

Notes for microlearning context:

- For online/mobile microlearning studies, *response rate* refers to completion of baseline measures among those who initiated the intervention
- For platform analytics studies, report the percentage of users who consented to data collection

DOMAIN 3: TYPE OF DATA (Maximum 3 points)

Score	Criterion	Description
3	Objective performance measure	Direct assessment of software engineering competency (e.g., code quality metrics, automated assessment scores, project evaluations, certification exam performance, task completion time/accuracy)
2	Performance-based self-report	Self-reported competency (e.g., "I can now write Java classes"), or study-specific knowledge tests without validation evidence
1	Subjective	Satisfaction, attitudes, perceptions, or engagement metrics without performance data

Notes for microlearning context:

- Learning analytics data (completion rates, time-on-task) without performance assessment = Score 1
- Automated assessment of code correctness = Score 3
- Scores on validated instruments (e.g., Scrum certification exams) = Score 3
- Platform analytics showing module completion but not learning = Score 1

DOMAIN 4: VALIDITY EVIDENCE FOR EVALUATION INSTRUMENTS (Maximum 3 points)

Score the highest level achieved

Score	Criterion	Description
3	All three types of validity evidence	Internal structure (e.g., factor analysis, reliability $\geq$0.70), content validity (expert review), and relationships with other variables (convergent/discriminant validity, correlation with external measures)
2	Two types of validity evidence	Any combination of two validity types
1	One type of validity evidence	Internal structure OR content validity OR relationships with other variables
0	No validity evidence	No reported validity evidence for instruments

Notes for microlearning context:

- Internal structure: Cronbach's $\alpha \geq 0.70$ for survey instruments, inter-rater reliability for code grading
- Content validity: Expert review of microlearning content by software engineering educators or industry practitioners
- Relationships: Correlation with course grades, prior experience, or external certifications

DOMAIN 5: DATA ANALYSIS (Maximum 3 points)

5a: Data Analysis—Appropriateness (Maximum 2 points)

Score	Criterion	Description
2	Appropriate for study design and complexity	Statistical analyses appropriate to research questions, with consideration of clustering, repeated measures, or covariates as needed
1	Appropriate for study design but basic	Basic analyses appropriate to design (t-tests, chi-square, ANOVA) without addressing complexity
0	Inappropriate for study design	Analyses inappropriate for design or not described

5b: Data Analysis—Sophistication (Maximum 1 point)

Score	Criterion	Description
1	Analysis beyond descriptive	Inferential statistics, effect sizes, confidence intervals, or predictive modeling
0	Descriptive statistics only	Means, percentages, frequencies only

Notes for microlearning context:

- Learning analytics studies using predictive modeling = Score 2 + 1 = 3
- Studies reporting only completion rates and satisfaction = Score appropriate design (1) + descriptive only (0) = 1
- N-Gain calculations with pre-post designs = Score appropriate design (2) + inferential (1) = 3

DOMAIN 6: OUTCOMES (Maximum 3 points)

Score	Criterion	Description
3	Transfer/Application	Assessment in authentic context (e.g., project work, internship performance, workplace application, subsequent course performance)
2	Higher-order learning	Assessment of analysis, synthesis, evaluation, or creation (e.g., software design, architectural decisions, debugging complex systems)
1.5	Knowledge/Skills	Assessment of recall, comprehension, or application of discrete concepts (e.g., syntax quizzes, definition recall)
1	Satisfaction/Attitudes/ Engagement	Self-reported satisfaction, motivation, engagement, or perceptions
0	Reactions only	No assessment of learning or behavioral outcomes

Notes for microlearning context:

- Students completing a full project using microlearned content = Score 3
- Students creating original software artifacts = Score 2
- Students passing certification exams = Score 1.5 (knowledge) or 2 (if exam requires analysis/synthesis)
- Platform analytics showing high engagement = Score 1

Appendix A.3. MERSQI-MSE Scoring Sheet

Table A1. MERSQI-MSE Quality Appraisal Scoring Sheet.

Domain	Item	Score	Max
1	Study Design		3
2	Sampling: Institutions		1.5
	Sampling: Response rate		1.5
3	Type of Data		3
4	Validity Evidence		3
5	Data Analysis: Appropriateness		2
	Data Analysis: Sophistication		1
6	Outcomes		3
	TOTAL		**18**

Appendix A.4. Interpretation Guidelines

Table A2. Quality Rating Interpretation

Total Score	Quality Rating	Description
15–18	High Quality	Strong evidence; methodologically rigorous studies with appropriate design, sampling, objective measures, and higher-order outcomes
11–14	Moderate Quality	Suggestive evidence; studies with some methodological strengths but notable limitations
7–10	Low Quality	Preliminary evidence; significant methodological concerns limit confidence in findings
<7	Very Low Quality	Limited confidence; major methodological flaws prevent reliable interpretation

Appendix B. Standardized Data Extraction Form

It is a comprehensive template capturing bibliographic information, study context, intervention characteristics, software engineering competencies, outcome measures, key findings, and methodological limitations across seven domains.

Reviewer ID: _____
Extraction Date: _____
Study ID: _____

Appendix B.1. Section A: Bibliographic Information

Field	Entry
Authors	
Year	
Title	
Journal/Conference	
Publication Type	☐ Journal article ☐ Conference proceedings ☐ Book chapter ☐ Technical report ☐ Dissertation ☐ Other: _____
Country(ies)	
Language	☐ English ☐ Spanish ☐ Other: _____
Funding Source	☐ Government/Public ☐ Industry/Private ☐ Foundation/NGO ☐ Not reported ☐ None declared

Appendix B.2. Section B: Study Context

Field	Entry
Educational Level	☐ Vocational/High school ☐ Undergraduate ☐ Graduate/Master's ☐ Professional development ☐ Mixed ☐ Other: _____
Institution Type	☐ University ☐ Vocational school ☐ Industry/Corporate training ☐ Online platform ☐ Multiple types ☐ Other: _____
Country/Region	
Population Description	
Sample Size	Total N: _____ Intervention group n: _____ Comparison group n: _____ (if applicable)
Gender Distribution	Female: _____% Male: _____% Not reported: _____% Other: _____%
Age Range/Mean	Range: _____ Mean (SD): _____
Prior Experience	☐ Novice (no prior programming) ☐ Some experience (1-2 courses) ☐ Experienced (multiple courses/projects) ☐ Professional ☐ Mixed ☐ Not reported
Recruitment Method	☐ Convenience ☐ Voluntary response ☐ Required course enrollment ☐ Purposive sampling ☐ Not specified ☐ Other: _____

Appendix B.3. Section C: Intervention Characteristics

Field	Entry
Intervention Name	
Course Name	
Primary Pedagogical Approach	☐ Project-based learning ☐ Gamification ☐ Mobile learning ☐ Automated assessment ☐ Video-based microlearning ☐ Concern-based learning ☐ Collaborative learning ☐ Spaced repetition ☐ Mixed: _____ ☐ Other: _____
Secondary Pedagogical Approaches	☐ Self-directed ☐ Instructor-led ☐ Peer learning ☐ Scenario-based ☐ Example-based ☐ Just-in-time learning ☐ Other: _____
Technological Platform	☐ Mobile app (native) ☐ Web-based platform ☐ LMS (e.g., Moodle) ☐ Custom-built tool ☐ Commercial platform: _____ ☐ No technology (e.g., paper-based) ☐ Other: _____
Platform Name	
Delivery Mode	☐ Fully online ☐ Blended/hybrid ☐ In-person with technology ☐ Mobile-only ☐ Desktop-only ☐ Cross-platform
Microlearning Design Features	☐ Videos (<10 min) ☐ Interactive modules ☐ Quizzes/assessments ☐ Code exercises ☐ Gamification elements ☐ Leaderboards ☐ Badges/achievements ☐ Discussion forums ☐ Push notifications ☐ Progress tracking ☐ Adaptive content ☐ Spaced repetition ☐ Other: _____
Micro-unit Duration	☐ <3 min ☐ 3–5 min ☐ 5–10 min ☐ 10–15 min ☐ Variable ☐ Not specified
Total Intervention Duration	☐ Single session ☐ 1 week ☐ 2–4 weeks ☐ 1–3 months ☐ Full semester ☐ Variable ☐ Not specified
Number of Micro-units	_____ modules/lessons (if specified)
Scheduling	☐ Self-paced ☐ Fixed schedule ☐ Spaced (e.g., daily reminders) ☐ Just-in-time (on-demand) ☐ Other: _____
Instructor Role	☐ Content designer ☐ Facilitator/monitor ☐ Feedback provider ☐ Not involved ☐ Not specified ☐ Other: _____

Appendix B.4. Section D: Software Engineering Competencies

Appendix B.4.1. Mapping to Software Development Lifecycle (SDLC) Phases

SDLC Phase	Competencies Addressed	Present	Description/Evidence
Conception/Analysis	Requirements engineering, Feasibility analysis, Stakeholder negotiation, Problem definition	☐	

Architecture Design	System architecture, Design patterns, Quality attribute trade-offs, Technology selection	☐	
Software Design	Detailed design, UML, Database design, Interface design, Component design	☐	
Coding (Front End)	UI/UX implementation, Client-side logic, Responsive design, Front-end frameworks	☐	
Coding (Back End)	Server-side logic, API development, Business logic, Database interactions	☐	
Testing/QA	Unit testing, Integration testing, Test-driven development, Debugging, Test automation	☐	
Deployment/DevOps	Continuous integration, Continuous delivery, Cloud deployment, Containerization	☐	
Maintenance/Evolution	Refactoring, Legacy code comprehension, Configuration management, Version control	☐	
Project Management	Agile/Scrum, Estimation, Risk management, Team coordination, Resource planning	☐	
Professional/Soft Skills	Communication, Teamwork, Leadership, Time management, Emotional intelligence	☐	

Appendix B.4.2. Specific Technical Competencies Taught

Competency Category	**Specific Topics (e.g., "inheritance in Java," "Scrum roles," "REST APIs")**
Programming Languages	
Frameworks/Libraries	
Tools/Environments	
Methodologies	
Concepts/Principles	

Appendix B.5. Section E: Outcome Measures

Appendix B.5.1. Learning Outcomes

Measure Type	**Instrument/Tool**	**Timing**	**Results**	**Effect Size**
☐ Knowledge test (researcher-developed)		☐ Pre ☐ Post ☐ Follow-up		
☐ Knowledge test (standardized/certification)		☐ Pre ☐ Post ☐ Follow-up		
☐ Code quality assessment		☐ Pre ☐ Post ☐ Follow-up		
☐ Project evaluation		☐ Pre ☐ Post ☐ Follow-up		
☐ Task completion (time/accuracy)		☐ Pre ☐ Post ☐ Follow-up		
☐ Course grades		☐ End of course		
☐ Other: _____				

Appendix B.5.2. Engagement Metrics

Metric	**Measurement Method**	**Results**
☐ Completion rate	Platform analytics	
☐ Time-on-task	Platform analytics	
☐ Module/lesson completion	Platform analytics	
☐ Voluntary participation	Attendance logs	
☐ Discussion participation	Forum posts/comments	
☐ Retention (return rate)	Platform analytics	

☐ Other: _____		

Appendix B.5.3. Affective Outcomes

Construct	Instrument	Validity Evidence	Results (Mean, SD)
☐ Motivation		☐ Yes ☐ No	
☐ Satisfaction		☐ Yes ☐ No	
☐ Engagement (self-report)		☐ Yes ☐ No	
☐ Self-efficacy		☐ Yes ☐ No	
☐ Attitudes toward subject		☐ Yes ☐ No	
☐ Perceived learning		☐ Yes ☐ No	
☐ Other: _____			

Appendix B.5.4. Retention/Transfer

Measure	Timing	Results
☐ Delayed post-test	_____ weeks/months	
☐ Subsequent course performance		
☐ Workplace application		
☐ Project transfer		
☐ Other: _____		

Appendix B.6. Section F: Key Findings

Appendix B.6.1. Primary Findings

1. _____
   _____
2. _____
   _____
3. _____
   _____

Appendix B.6.2. Statistical Significance

Finding	*p*-Value	Effect Size (95% CI)

Appendix B.7. Section G: Methodological Limitations

Appendix B.7.1. Limitations Noted by Study Authors

1. _____
   _____
2. _____
   _____
3. _____
   _____

Appendix B.7.2. Limitations Identified During Quality Appraisal (MERSQI-MSE)

Domain	Limitation	Score Deduction
Study Design		
Sampling		
Type of Data		
Validity Evidence		
Data Analysis		
Outcomes		

MERSQI-MSE Total Score: _____ /18

Quality Rating: ☐ High (15–18) ☐ Moderate (11–14) ☐ Low (7–10) ☐ Very low (<7)

Appendix B.8. Section H: Additional Notes

Appendix B.8.1. Contextual Factors

_____

_____

_____

Appendix B.8.2. Equity Considerations

_____

_____

_____

Appendix B.8.3. Quotable Quotes

_____

_____

_____

Appendix B.8.4. Questions for Study Authors

_____

_____

_____

Appendix B.9. Section I: Verification

☐	Data entered accurately matches source
☐	All extraction fields completed (or marked N/A)
☐	Page numbers noted for key findings
☐	MERSQI-MSE scoring completed

Second Reviewer Verification: _____ (initials) **Date:** _____

References

Al-Bhloly, L. A., Almutawkkil, A. Y., & Zahary, A. T. (2024). Micro-learning (ML): A comprehensive survey. In *2024 1st international conference on emerging technologies for dependable internet of things (ICETI)* (pp. 1–15). IEEE.

Alias, N. F., & Razak, R. A. (2025). Revolutionizing learning in the digital age: A systematic literature review of microlearning strategies. *Interactive Learning Environments*, *33*(1), 1–21. [CrossRef]

Alshawi, A. (2023). *An investigation into the rise of boot camp against educational offering of traditional educational institutions: Building a personality and preference–driven model for digital marketing and coding programs offered in university, community college, and bootcamp educational institutions* [Doctoral dissertation, Alliant International University].

An, R. (2022). Development of a microlearning content model for software engineering courses. In *Conference on electrical engineering, informatics, industrial technology, and creative media* (Vol. 1, pp. 170–177). CAPE Eoruma.

Álvarez, P. M. (2025). *Microaprendizajes: La revolución del conocimiento en pequeñas dosis* (Vol. 20). Narcea Ediciones.

Boumalek, K., Bakki, A., El Mezouary, A., Hmedna, B., & Eddahibi, M. (2025). Micro-learning design and micro-course structuring: A systematic literature review. *Interactive Learning Environments*, 1–27. [CrossRef]

Cambarieri, M., Viadana, C. A., Vivas, L., Rached, S., Jauge, M., Rizzo, A., Leder, J., Gonzalez, F., & Farra, C. (2025). Microcredenciales en la Licenciatura en Sistemas, una carrera regulada por el Estado y el Rol del estándar W3C en el desarrollo de las Credenciales Verificables. *JAIIO, Jornadas Argentinas de Informática*, *11*(13), 145–160.

CASP. (2018). *Critical appraisal skills programme (CASP). Qualitative checklist*. CASP Oxford.

Catenazzi, N., Sommaruga, L., De Angelis, K., & Caboni, S. (2025). A comprehensive methodology for curriculum development, training delivery and certification using learning outcomes and digital badges. *Computers and Education Open*, *8*, 100248. [CrossRef]

Correa, D., Arango Isaza, F., Mazo, R., & Giraldo, G. L. (2018). CME–A web application framework learning technique based on concerns, micro-learning and examples. In *International conference on web engineering* (pp. 17–32). Springer.

Correa Botero, D. (2015). *Learning of web application frameworks based on concerns, micro-learning and examples*. Universidad Nacional de Colombia.

Dixit, R., Yalagi, P. S., & Nirgude, M. (2021). Breaking the walls of classroom through Micro learning: Short burst of learning. *Journal of Physics: Conference Series*, *1854*, 012018. [CrossRef]

Garcia, M., & Stefany, S. (2023). Micro-credentials for Indonesia: Responding to industry demand for jobs in IT-computer software and in the metaverse (an analysis using big data and artificial intelligence). In *Education technology in the new normal: Now and beyond: Proceedings of the international symposium on open, distance, and e-learning (ISODEL 2021), Jakarta, Indonesia, 1–3 December 2021* (pp. 298–306). Routledge.

Gasca-Hurtado, G. P., & Gómez-Álvarez, M. C. (2021). Mobile application based on gamification to promote microlearning in Software Engineering. In *2021 16th iberian conference on information systems and technologies (cisti)* (pp. 1–8). IEEE.

Gasca-Hurtado, G. P., Morillo-Puente, S., & Gómez-Álvarez, M. C. (2024). Microlearning strategy in the promotion of motivation and learning outcomes in software project management. *Computer Applications in Engineering Education*, *32*(3), e22717. [CrossRef]

Guzmán Rojas, A. L. (2024). *Flexibilidad curricular y credenciales alternativas en el posgrado: Estrategias para cerrar la brecha de habilidades entre la educación superior y el mercado laboral* [Maestría en Ingeniería de Software, Universidad de los Andes].

Hadiyana, R. W., Putra, F. A., & Khair, R. (2026). Design and engineering of an AI-enabled mobile microlearning application integrating short-form video and learning analytics for vocational soft skills development. *Journal of Informatics and Telecommunication Engineering*, *9*(2), 403–417.

Herath, I. (2024). *Cross-platform development with full-stack frameworks: Bridging the gap for seamless integration* [Bachelor's thesis, Jamk University of Applied Sciences]. Available online: https://www.theseus.fi/handle/10024/867532 (accessed on 19 January 2026).

Hridoy, S. M. A. S., Rahman, S., Rishad, S. M. S. I., Bhuiyan, M. S., Islam, S., & Raihan, M. J. (2023, December). *A comprehensive framework for evaluating software engineering technologies* (Tech. Rep.). SSRN. Available online: https://ssrn.com/abstract=4650826 (accessed on 19 January 2026).

Jaros, S., & Beck Dallaghan, G. (2024). Medical education research study quality instrument: An objective instrument susceptible to subjectivity. *Medical Education Online*, *29*(1), 2308359. [CrossRef]

Jo, S., Bernadas, I. T., Kim, Y., & Kim, D. (2025). The effectiveness of microlearning in coding education using information system success model: Evidence from the Philippines. *International Journal of Advanced Culture Technology*, *13*(2), 102–114.

Jwo, J.-S. (2015). Teaching software engineering through comics and micro-movie design projects. In *2015 10th international conference on computer science & education (iccse)* (pp. 328–331). IEEE.

Keeley, K. L. (2025). The scope and depth micro-credential framework: Aligning education with evolving industry demands. In *Innovative approaches in vocational and regional education* (pp. 49–86). IGI Global Scientific Publishing.

Kesavan, E. (2025). The impact of cloud computing on software development: A review. *International Journal of Innovations in Science, Engineering and Management*, *4*, 269–274. [CrossRef]

Knowles, M. S. (1984). *The adult learner: A neglected species* (3rd ed.). Gulf Publishing.

Kohler, M., Gamrat, C., Raish, V., & Gross, E. (2021). Microlearning and micro-credentials in higher education. In *Microlearning in the digital age* (pp. 109–128). Routledge.

LaPlante, K. (2024). *Exploring the motivations of graduate students: A phenomenological study of micro-credential adoption through student perspective* [Unpublished doctoral dissertation]. Southern New Hampshire University.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. Cambridge University Press.

Lee, Y.-M. (2023). Mobile microlearning: A systematic literature review and its implications. *Interactive Learning Environments*, *31*(7), 4636–4651. [CrossRef]

Leong, K., Sung, A., Au, D., & Blanchard, C. (2021). A review of the trend of microlearning. *Journal of Work-Applied Management*, *13*(1), 88–102. [CrossRef]

Luo, H., & Li, W. (2025). Impact of microlearning on developing soft skills of university students across disciplines. *Frontiers in Psychology*, *16*, 1491265. [CrossRef]

Λυγούρα, A. (2024). *Defining and comparing digital (micro-) credentials for learning. A systematic literature review* [Master's thesis, University of Piraeus]. Available online: https://dione.lib.unipi.gr/xmlui/handle/unipi/17100 (accessed on 19 January 2026).

Mangena, A., Ncube, N., & Ncube, C. (n.d.). Transforming Computer science education through the use microcredentials in computer networking: Teachers and lecturers perspectives. *International Journal of Research and Innovation in Social Science*, *IX*(IV), 205–215. [CrossRef]

Monib, W. K., Qazi, A., & Apong, R. A. (2025). Microlearning beyond boundaries: A systematic review and a novel framework for improving learning outcomes. *Heliyon*, *11*(2), e41413. [CrossRef]

Moore, R. L., Hwang, W., & Moses, J. D. (2024). A systematic review of mobile-based microlearning in adult learner contexts. *Educational Technology & Society*, *27*(1), 137–146.

Murphy-Hill, E., & Grossman, D. (2014). How programming languages will co-evolve with software engineering: A bright decade ahead. In *Proceedings of the 36th international conference on software engineering (icse 2014)* (pp. 145–154). ACM. [CrossRef]

Oliver, B. (2022). *Towards a common definition of micro-credentials* (Report). UNESCO. Available online: https://unesdoc.unesco.org/ark:/48223/pf0000381668 (accessed on 19 January 2026).

Orman, R., Şimşek, E., & Kozak Çakır, M. A. (2023). Micro-credentials and reflections on higher education. *Higher Education Evaluation and Development*, *17*(2), 96–112. [CrossRef]

Rahutomo, R., Hiererra, S. E., Arifin, Y., Suprayogi, M. N., & Pardamean, B. (2023). Fundamental components of microlearning for sustainable quality education: A systematic literature review. In *2023 international conference on information management and technology (icimtech)* (pp. 684–689). IEEE.

Rusak, Z. (2017). Exploitation of micro-learning for generating personalized learning paths. In *Ds 87-9 proceedings of the 21st international conference on engineering design (iced 17) vol 9: Design education, vancouver, canada, 21–25 August 2017* (pp. 129–138). Delft University of Technology.

Schultheiss, T., & Backes-Gellner, U. (2023). Different degrees of skill obsolescence across hard and soft skills and the role of lifelong learning for labor market outcomes. *Industrial Relations: A Journal of Economy and Society*, *62*(3), 257–287. [CrossRef]

Shatte, A. B., & Teague, S. (2020). *Microlearning for improved student outcomes in higher education: A scoping review*. OSF Preprints. Available online: https://osf.io/download/fhu8n (accessed on 19 January 2026).

Silva, E. S., Costa, W. P. d., Lima, J. C. d., & Ferreira, J. C. (2025). Contribution of microlearning in basic education: A systematic review. *Education Sciences*, *15*(3), 302. [CrossRef]

Sinhabahu, N. (2020). *Visualizing security vulnerability evolution of software systems* [Master of Computer Science (MCS) thesis, University of Colombo School of Computing]. Available online: http://dl.ucsc.cmb.ac.lk/jspui/handle/123456789/4515 (accessed on 17 February 2026).

Skalka, J. (2018, May 2–4). *Data processing methods in the development of the microlearning-based framework for teaching programming languages*. Divai 2018: 12th International Scientific Conference on Distance Learning in Applied Informatics, Štúrovo, Slovakia.

Skalka, J., & Drlik, M. (2020). Automated assessment and microlearning units as predictors of at-risk students and students' outcomes in the introductory programming courses. *Applied Sciences*, *10*(13), 4566. [CrossRef]

Skalka, J., & Drlík, M. (2018). Educational model for improving programming skills based on conceptual microlearning framework. In *International conference on interactive collaborative learning* (pp. 923–934). Springer.

Skalka, J., Drlík, M., Obonya, J., & Cápay, M. (2020). Architecture proposal for micro-learning application for learning and teaching programming courses. In *2020 IEEE global engineering education conference (EDUCON)* (pp. 980–987). IEEE.

Stankovski, V. (2025, March 8–10). *Microcredentials for microcompetences in education*. International Conferences on Mobile Learning 2025 and Educational Technologies 2025 (pp. 259–264), Porto, Portugal. Available online: https://files.eric.ed.gov/fulltext/ED674523.pdf#page=280 (accessed on 19 January 2026).

Suciu, D. M. (2022). Implementing microlearning and gamification techniques in teaching software project management concepts. In *Proceedings of the 4th international workshop on education through advanced software engineering and artificial intelligence* (pp. 37–44). ACM.

Tricco, A. C., Lillie, E., Zarin, W., O'Brien, K. K., Colquhoun, H., Levac, D., Moher, D., Peters, M. D., Horsley, T., Weeks, L., Hempel, S., Akl, E. A., Chang, C., McGowan, J., Stewart, L., Hartling, L., Aldcroft, A., Wilson, M. G., Garritty, C., … Straus, S. E. (2018). PRISMA extension for scoping reviews (PRISMA-ScR): Checklist and explanation. *Annals of Internal Medicine*, *169*(7), 467–473. [CrossRef] [PubMed]

Tunguz, B. (2025). *Transforming education and training in the age of advanced ai reasoning tools*. JSTOR.

Urem, F., Goleš, I. K., & Pavelić, M. (2025). Modernizing computer science education with micro-credentials. In *2025 15th international conference on advanced computer information technologies (ACIT)* (pp. 1026–1029). IEEE.

Wibisono, I., Maringga, A. P. I. B., & Sunardi. (2024). Comparing the user experience of mobile-based microlearning application using heart metrics and honeycomb framework. In *2024 international conference on information management and technology (ICIMTech)* (pp. 450–455). IEEE.

Wigati, W., Iriani, T., & Khaerudin, K. (2024). Analysis and exploration of mobile microlearning for vocational high school students. *Journal of Science and Education (JSE)*, *5*(1), 246–253. [CrossRef]

Wira, I. B. G., Sudatha, I. G. W., Warpala, I. W. S., Parwati, N. N., & Suartama, I. K. (2026). Project-based e-learning development with a microlearning approach to improve programming learning outcomes of vocational high school students. *JKTP: Jurnal Kajian Teknologi Pendidikan*, *9*(1), 010–021. [CrossRef]

Zimmerman, B. J. (2002). Becoming a self-regulated learner: An overview. *Theory into practice*, *41*(2), 64–70.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.