

A Sustainability-Oriented Lifecycle Framework for Educational Technology Software

Hang Yi ^{1,*}

¹ Swan College, Central South University of Forestry and Technology, Hunan 410000, China

* Correspondence: Hang Yi, Swan College, Central South University of Forestry and Technology, Hunan 410000, China

Abstract: The rapid expansion of educational technology (EdTech) has profoundly reshaped global learning environments while raising growing concerns about software sustainability, including technical debt, excessive energy consumption, and limited product longevity. Current development models often emphasize functionality and scalability but rarely incorporate long-term ecological and pedagogical considerations. To address this gap, this study introduces the Sustainability-Oriented Educational Software Lifecycle (SESL) framework, which systematically embeds sustainability principles into every phase of software development. Drawing on a comparative case study of three representative platforms-Moodle, ClassDojo, and Coursera-the research integrates qualitative evidence from documentation analysis, stakeholder interviews, and lifecycle indicators to identify the primary drivers and inhibitors of sustainability in EdTech. The findings reveal five major sustainability drivers-modular architecture, community governance, ethical data practices, energy-efficient deployment, and pedagogical traceability-counterbalanced by barriers such as short product cycles, centralized control, and the lack of standardized evaluation metrics. The SESL framework establishes five iterative sustainability checkpoints throughout the software lifecycle, reframing sustainability from a terminal assessment into a continuous, feedback-oriented process. By bridging sustainable software engineering with educational technology design, this study provides both a theoretical and practical foundation for fostering resilient, inclusive, and ecologically responsible EdTech ecosystems.

Keywords: sustainable software engineering; educational technology (EdTech); software lifecycle framework; systems thinking; circular economy design; pedagogical sustainability

Received: 06 September 2025

Revised: 16 September 2025

Accepted: 18 October 2025

Published: 22 October 2025



Copyright: © 2025 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The digital transformation of education has resulted in a rapid proliferation of educational technology (EdTech) systems that now mediate learning across formal, informal, and hybrid environments [1]. Platforms such as Moodle, ClassDojo, and Coursera exemplify how software ecosystems have evolved into critical infrastructures for contemporary pedagogy, enabling adaptive instruction, data-driven assessment, and large-scale learner engagement [2]. Yet, alongside this expansion arises a pressing challenge: the lack of sustainability in EdTech software development lifecycles. Frequent version fragmentation, excessive server energy consumption, and limited reusability of learning modules have rendered many EdTech platforms short-lived, costly to maintain, and environmentally unsustainable [3]. Consequently, the educational sector faces not only pedagogical but also ecological and operational vulnerabilities.

Traditional software engineering frameworks-such as the waterfall and agile models-have improved development efficiency but seldom embed sustainability as a core lifecycle

objective [4]. Likewise, although research in sustainable software engineering (SSE) has introduced principles emphasizing energy efficiency, maintainability, and modular reuse, its application to educational systems remains limited. Conversely, instructional design models such as ADDIE and SAM focus primarily on iterative pedagogical enhancement while overlooking the technical sustainability of the software that operationalizes these models [5]. This disciplinary disconnect has led to fragmented development practices. For example, Moodle demonstrates community-driven adaptability yet struggles with long-term maintainability; ClassDojo excels in rapid iteration and user-centered design but operates within opaque data architectures that restrict lifecycle transparency; Coursera achieves global scalability but faces difficulties mitigating the carbon footprint of large-scale cloud infrastructure [6]. These examples illustrate that while EdTech systems have achieved remarkable pedagogical innovation, they still lack a coherent, sustainability-oriented lifecycle framework that balances educational effectiveness with technical, social, and environmental responsibility.

To address this gap, the present study develops a Sustainability-Oriented Educational Software Lifecycle (SESL) framework specifically tailored to EdTech systems. Grounded in socio-technical systems theory, systems thinking, and circular economy design, the SESL framework redefines sustainability as an ongoing design principle rather than a terminal evaluation stage. It embeds sustainability considerations throughout all phases of the lifecycle—from requirements analysis and architectural design to deployment, maintenance, and decommissioning. By establishing traceable linkages between pedagogical objectives, stakeholder values, and technical decisions, the framework seeks to enhance both system resilience and educational continuity.

Methodologically, this research employs a comparative qualitative case study approach. Three representative platforms—Moodle (open-source community model), ClassDojo (K-12 social learning network), and Coursera (massive open online course provider)—serve as focal cases. Each represents a distinctive governance and development paradigm, enabling cross-comparison of sustainability mechanisms. Through document analysis, developer interviews, and lifecycle sustainability mapping, the study identifies key sustainability drivers such as modularity, inclusivity, data ethics, and energy efficiency. These insights inform the iterative design and validation of the proposed SESL framework.

Academically, this study contributes to bridging the divide between software engineering and educational sustainability, extending the discourse of sustainable development into the digital learning domain. Practically, it provides actionable guidance for policymakers, developers, and instructional designers seeking to enhance the longevity, transparency, and ecological accountability of educational software ecosystems. By embedding sustainability principles into the architecture of EdTech lifecycles, the SESL framework establishes a foundation for resilient, ethically grounded, and environmentally conscious educational innovation.

2. Literature Review

The concept of sustainability in software development has evolved over the past decade from a peripheral consideration into a core design paradigm. However, its integration within the field of educational technology (EdTech) remains partial and fragmented. This review synthesizes three interrelated strands of scholarship that underpin the development of the *Sustainability-Oriented Educational Software Lifecycle* (SESL) framework:

1. sustainable software engineering paradigms,
2. models of educational technology design and implementation, and
3. integrative frameworks that connect technical and pedagogical sustainability.

Each section contextualizes theoretical perspectives within the operational realities of three representative EdTech platforms—Moodle, ClassDojo, and Coursera—which serve as empirical anchors for the analysis presented in this study.

2.1. Sustainable Software Engineering Paradigms

Research in sustainable software engineering (SSE) emphasizes the long-term ecological, economic, and social impacts of software systems [7]. Core SSE principles include energy efficiency, modular reusability, maintainability, and lifecycle transparency. To operationalize these principles, scholars have proposed a range of sustainability metrics—such as code sustainability indices, carbon-aware deployment models, and maintainability scores—that aim to quantify both the environmental and technical resilience of software [8]. Despite these advancements, most SSE frameworks remain domain-agnostic, originally designed for enterprise or infrastructure-oriented systems rather than for educational technology contexts.

In contrast, EdTech platforms such as Moodle reveal a distinct set of sustainability challenges. Moodle's open-source model supports community-driven maintenance and minimizes licensing barriers, yet its extensive plugin ecosystem introduces architectural complexity that contributes to technical debt and elevated energy consumption [9]. Likewise, ClassDojo—a cloud-hosted platform characterized by rapid iteration cycles—exemplifies the tension between scalability and long-term codebase stability. While frequent updates foster user engagement and innovation, they also produce short-lived design layers that complicate maintenance and increase resource utilization.

These cases demonstrate that although existing SSE paradigms provide a conceptually robust foundation, they require contextual adaptation to address the educational, ethical, and pedagogical dimensions unique to the EdTech domain.

2.2. Educational Technology Development Models

The development of educational technology has traditionally been guided by instructional design frameworks such as ADDIE (Analysis-Design-Development-Implementation-Evaluation), SAM (Successive Approximation Model), and UDL (Universal Design for Learning) [10]. These models emphasize learner experience, accessibility, and iterative feedback, yet they seldom engage with the lifecycle sustainability of the underlying software systems. In contemporary practice, EdTech platforms operate within continuous integration and deployment environments, where pedagogical goals frequently intersect—and sometimes conflict—with market-driven demands for rapid feature updates and user growth.

ClassDojo exemplifies a user-centered design philosophy that promotes engagement and inclusivity but lacks explicit sustainability checkpoints within its development process [11]. *Coursera*, by contrast, operates at a global scale, optimizing for scalability and data-driven personalization. However, its dependence on large-scale cloud infrastructure contributes to substantial energy consumption and limited transparency regarding data lifecycle governance [12]. Meanwhile, *Moodle's* decentralized architecture facilitates adaptability and local customization but suffers from version control inconsistencies and maintenance fragmentation.

These examples demonstrate that while instructional design models are pedagogically robust, they offer limited guidance for reconciling educational effectiveness with technical sustainability [13]. Consequently, there is a pressing need for a lifecycle framework that integrates pedagogical and technical perspectives, embedding educational values directly within sustainable software engineering practices rather than treating them as separate or sequential domains.

2.3. Integrative Frameworks and Theoretical Tensions

Recent scholarship has increasingly sought to develop integrative frameworks that align sustainability principles with the design and implementation of educational software. Conceptual foundations have emerged from systems thinking, socio-technical systems theory, and circular economy design. *Systems thinking* conceptualizes educational

software as an integral component of a broader learning ecosystem, wherein environmental, social, and technical subsystems interact dynamically and reciprocally. *Socio-technical systems theory* emphasizes the co-evolution of software artifacts and human actors, highlighting the importance of value-sensitive design and participatory decision-making [14]. In parallel, circular economy principles advocate for reuse, adaptability, and long-term resource efficiency-values directly applicable to modular content structures and reusable software components within EdTech systems.

Despite their conceptual promise, existing integrative models often remain theoretical and under-operationalized in educational technology contexts. A persistent tension between agility and longevity continues to challenge sustainable implementation: agile methodologies prioritize speed and user responsiveness, while sustainability demands stability, continuity, and reduced redundancy. For instance, *Moodle's* iterative community-driven releases exemplify participatory resilience yet lack comprehensive lifecycle energy monitoring [15]. *Coursera's* cloud orchestration achieves performance optimization but neglects environmental externalities associated with large-scale computing. Similarly, *ClassDojo's* "design-for-delight" philosophy enhances short-term user satisfaction but rarely accounts for post-deployment code reuse or data stewardship [16].

As summarized in Table 1, these cases illustrate how each platform embodies partial sustainability practices, yet none achieves full lifecycle integration across technical, pedagogical, and ecological dimensions. This persistent gap underscores the need for a holistic, sustainability-oriented lifecycle framework that embeds environmental, ethical, and pedagogical metrics into every stage of educational software development.

Table 1. Comparative sustainability characteristics of representative EdTech platforms.

Dimension	Moodle (Open-source LMS)	ClassDojo (K-12 Network)	Coursera (MOOC Platform)	Sustainability Gap
Governance & Lifecycle	Community-driven; transparent but uneven maintenance quality.	Centralized agile updates; rapid iteration, limited accountability.	Corporate-academic partnerships; stable but opaque release control.	No unified sustainability governance.
Technical Sustainability	Modular and reusable; high plugin dependency.	Frequent UI/UX changes; technical debt accumulation.	Scalable and optimized; closed architecture.	Trade-off between adaptability and maintainability.
Environmental Impact	Light local hosting; inconsistent energy efficiency.	Cloud-heavy multimedia usage; high energy load.	Data-intensive streaming; large carbon footprint.	Lack of carbon-aware lifecycle design.
Ethical & Educational Alignment	Supports localization and open access; weak lifecycle linkage to pedagogy.	Promotes engagement; limited transparency in data use.	Expands global access; rigid pedagogical structures.	Weak integration of ethical and pedagogical metrics.
Overall Evaluation	Technically resilient but environmentally inconsistent.	Socially engaging but technically fragile.	Scalable but ecologically heavy and pedagogically narrow.	None achieve integrated sustainability across all lifecycle phases.

In summary, the existing literature exposes a dual fragmentation: while sustainable software engineering offers robust methodological tools, it often lacks alignment with educational objectives; conversely, instructional design models provide pedagogical depth but rarely incorporate lifecycle accountability. Bridging this divide necessitates a unifying framework that grounds EdTech development in sustainability principles at both theoretical and operational levels. This recognition underpins the present study's formulation of the *Sustainability-Oriented Educational Software Lifecycle (SESL)* framework, which embeds multi-dimensional sustainability-ecological, technical, and pedagogical-into the very architecture of educational software design, implementation, and maintenance.

3. Theoretical Framework and Methodology

3.1. Theoretical Foundations

The proposed Sustainability-Oriented Educational Software Lifecycle (SESL) framework is grounded in three complementary theoretical perspectives: socio-technical systems theory, systems thinking, and circular economy design principles. Together, these perspectives provide a conceptual foundation for embedding sustainability throughout every phase of software development rather than treating it as an afterthought [17].

Socio-technical systems theory views technology and human actors as interdependent subsystems whose alignment determines overall system effectiveness. In educational contexts, this implies that software sustainability cannot be isolated from pedagogical and social sustainability. For example, Moodle's open-source ecosystem illustrates how developer communities and educator networks co-evolve, but also how governance gaps may undermine technical reliability. Recognizing these socio-technical relationships helps translate sustainability objectives into participatory design and collaborative maintenance practices.

Systems thinking extends this approach by conceptualizing educational software as part of a dynamic learning ecosystem that links students, teachers, administrators, and infrastructures. From this perspective, sustainability becomes an emergent property of interactions that unfold across time and scale. Coursera's global delivery network, for instance, demonstrates how optimization within one subsystem-such as server efficiency or data analytics-can generate unintended consequences elsewhere, including environmental impact or inequitable access. Systems thinking thus encourages the mapping of interdependencies to anticipate trade-offs among ecological, ethical, and pedagogical objectives.

Finally, circular economy principles emphasize reuse, adaptability, and value retention throughout the software lifecycle. Applied to EdTech, these ideas translate into modular architectures, reconfigurable learning components, and data governance models designed for longevity. ClassDojo's rapid release cycles, while fostering innovation, highlight the need for controlled iteration that minimizes redundant development and digital waste. The circular economy perspective therefore offers operational guidance for sustainable code management, resource optimization, and feature evolution.

Together, these theoretical frameworks position the SESL as an integrative model that unites technical resilience, environmental responsibility, and pedagogical coherence within a single, sustainability-oriented design philosophy.

3.2. Research Design

This study employs a comparative multiple-case study approach to develop and validate the Sustainability-Oriented Educational Software Lifecycle (SESL) framework. The selected cases-Moodle, ClassDojo, and Coursera-represent distinct development philosophies and governance models, thereby enabling theoretical replication and comparative contrast across different segments of the EdTech ecosystem.

(1) Case Selection Rationale

- Moodle: A community-maintained, open-source platform that provides an ideal context for examining decentralized sustainability mechanisms.
- ClassDojo: A privately governed K-12 social learning network that emphasizes user experience, engagement, and rapid iteration.
- Coursera: A large-scale commercial MOOC provider that integrates academic institutions and corporate actors, exemplifying enterprise-level scalability.

Together, these cases span the continuum from open collaborative models to commercial enterprise systems, reflecting the diversity and complexity of the global EdTech landscape.

(2) Data Sources

The analysis draws on multiple qualitative and quantitative data streams, including:

- Public documentation such as release notes, developer guides, and energy usage reports;
- Semi-structured interviews with developers, instructional designers, and sustainability coordinators;
- Academic and grey literature evaluating each platform's lifecycle practices;
- Secondary sustainability indicators, including code repository activity, update frequency, and hosting infrastructure data.

(3) Analytical Procedure

The research design unfolds in three iterative analytical stages:

- Stage 1: Cross-case mapping - Lifecycle attributes (planning, design, implementation, maintenance, and decommissioning) are extracted and aligned with three sustainability dimensions: technical, environmental, and pedagogical.
- Stage 2: Pattern identification - Recurring drivers and inhibitors of sustainability are identified, such as modular reuse, energy optimization, community governance, and inclusive access.
- Stage 3: Model synthesis - Insights from the cross-case analysis are integrated into the SESL framework, refining feedback loops and sustainability checkpoints across each lifecycle phase.

Qualitative coding is applied to cluster observations under sustainability categories, while comparative matrices are used to assess the relative presence of sustainability features on an ordinal scale (low-medium-high). Triangulation across data sources strengthens validity, mitigates case-specific bias, and enhances the generalizability of findings.

3.3. Framework Construction

The Sustainability-Oriented Educational Software Lifecycle (SESL) framework operationalizes sustainability across five iterative phases: Requirements, Design and Architecture, Implementation, Maintenance, and Renewal. Each phase incorporates ethical, ecological, and pedagogical checkpoints that guide decision-making throughout the development process. The framework emphasizes the co-definition of learning objectives and sustainability KPIs, modular design to enhance resource efficiency, energy-conscious implementation practices, community-driven maintenance, and the reuse of knowledge and components during renewal. As summarized in Table 2, these checkpoints ensure that sustainability is not treated as a terminal evaluation but as a continuous, feedback-oriented process embedded in every lifecycle stage, thereby aligning educational innovation with long-term environmental and ethical responsibility.

Table 2. Core Phases and Sustainability Checkpoints in the SESL Framework.

Lifecycle Phase	Sustainability Focus	Illustrative Practice
Requirements	Ethical & pedagogical alignment	Define learning goals and sustainability KPIs jointly with educators and stakeholders.

Design & Architecture	Resource efficiency & modularity	Apply circular-economy design; reuse code modules and learning objects.
Implementation	Inclusive and low-impact coding	Optimize algorithms for energy efficiency; ensure accessibility compliance.
Maintenance	Continuous monitoring & community feedback	Track energy use, user equity metrics, and code health; integrate open governance as in Moodle.
Renewal	Knowledge transfer & reuse	Archive reusable assets; support migration paths instead of total redevelopment.

3.4. Expected Contributions

Academically, the SESL framework advances sustainability theory by embedding it within educational software lifecycles, thereby bridging the gap between software engineering and the learning sciences. Methodologically, the comparative multi-case approach illustrates how theoretical constructs can be empirically grounded and validated. Practically, the SESL framework provides a diagnostic and planning tool for developers and policymakers, enabling the assessment of sustainability maturity and informing decision-making at each phase of software development.

4. Findings and Discussion

4.1. Comparative Case Findings

The cross-case analysis of Moodle, ClassDojo, and Coursera indicates that while each platform exhibits selective strengths in sustainability, none achieves comprehensive integration across the full software lifecycle. Their practices can be categorized into three interrelated dimensions—technical, environmental, and social-pedagogical—which together determine the long-term viability of educational software systems.

Moodle, as an open-source learning management system, demonstrates strong technical resilience and social transparency. Its decentralized governance encourages active community participation, resulting in high adaptability and extensive local customization. However, this openness also contributes to version fragmentation and plugin redundancy, which increase energy consumption and complicate maintenance. Lifecycle documentation is inconsistent, and long-term sustainability metrics are seldom monitored beyond the community level.

ClassDojo excels in user-centered design and inclusive pedagogy, fostering emotional engagement among teachers, students, and parents. Its intuitive interface and frequent updates maintain high user engagement, but this comes at the expense of technical and environmental stability. Rapid iteration generates substantial technical debt and redundant code layers, while the platform's closed data architecture limits transparency and ethical accountability. In this context, sustainability is subordinated to short-term user experience objectives.

Coursera illustrates scalability and infrastructure efficiency through its professional development pipelines and institutional partnerships. Its global reach demonstrates operational robustness, yet the platform's large-scale cloud orchestration entails significant carbon emissions. Pedagogically, Coursera's centralized model constrains local adaptation, emphasizing credential delivery over contextualized learning. The absence of lifecycle energy monitoring or data reuse mechanisms further limits ecological sustainability.

Collectively, these cases reveal a fragmented sustainability landscape: Moodle emphasizes community governance, ClassDojo prioritizes engagement, and Coursera focuses on efficiency, yet none integrates sustainability as a holistic, embedded design principle.

4.2. Sustainability Drivers and Inhibitors

The comparative synthesis of Moodle, ClassDojo, and Coursera highlights several core factors that influence the sustainability of educational technology software. Among the most significant drivers, five stand out. First, modular architecture and code reuse serve as a key enabler of lifecycle continuity, reducing maintenance costs and facilitating renewal through circular-design principles, as exemplified by Moodle's extensive plugin ecosystem. Second, community governance and feedback mechanisms enhance transparency, accountability, and user participation, allowing decentralized decision-making that aligns technical maintenance with educational objectives. Third, inclusive accessibility and ethical data practices reinforce social sustainability by promoting equity, trust, and responsible data management. Fourth, energy-efficient deployment improves environmental performance by optimizing hosting and data flows, an area that remains underdeveloped across most platforms. Finally, pedagogical alignment and value traceability ensure that software development remains closely linked to learning goals, embedding educational purpose throughout the lifecycle.

Several inhibitors, however, constrain these sustainability drivers. The most pervasive is the prevalence of short product cycles, which favor rapid market responsiveness over long-term stability. Centralized control within proprietary systems further limits opportunities for collaborative governance by educators and users. A related barrier is the absence of standardized sustainability metrics that integrate technical and pedagogical dimensions. Collectively, these patterns indicate that achieving sustainability in EdTech depends not on isolated optimizations, but on the systemic integration of values, processes, and feedback loops, a principle that underpins the SESL framework.

4.3. Interpretation through the SESL Framework

Mapping the empirical findings onto the Sustainability-Oriented Educational Software Lifecycle (SESL) framework highlights how the three platforms align with or diverge from the framework's sustainability checkpoints across different lifecycle phases. In the Requirements stage, Moodle partially satisfies the ethical alignment criterion through community-based planning and shared decision-making, whereas ClassDojo and Coursera retain more hierarchical, top-down structures that limit stakeholder participation. During the Design and Architecture phase, Moodle's modular configuration reflects circular-economy principles by enabling component reuse and customization, yet it lacks systematic assessment of energy efficiency. Coursera, by contrast, achieves high scalability through centralized cloud architecture but sacrifices transparency and participatory governance.

In the Implementation stage, divergent patterns are evident. ClassDojo's rapid deployment practices support pedagogical innovation but increase energy consumption and code redundancy, whereas Moodle's open-source coding fosters reuse and adaptability. Coursera's proprietary model, though technically optimized, restricts external evaluation and limits potential for reuse. The Maintenance and Evaluation stage further underscores these contrasts: Moodle incorporates community feedback loops that closely align with SESL's participatory governance ideals, while Coursera's efficiency-focused maintenance pipeline remains largely opaque to end-users. Finally, in the Renewal stage, none of the platforms demonstrates systematic knowledge reuse, archival strategies, or sustainability reporting, revealing persistent discontinuities across the lifecycle.

Overall, this mapping demonstrates the diagnostic value of the SESL framework: it identifies underdeveloped sustainability checkpoints and provides a structured approach for continuous improvement and targeted lifecycle interventions.

4.4. Structural Representation

To illustrate this logic, Figure 1 presents the SESL framework as a cyclical process encompassing five lifecycle phases with bidirectional feedback loops. Each node corre-

sponds to a decision checkpoint aligned with specific sustainability criteria: ethical alignment in the Requirements phase, modular efficiency in Design, low-impact coding in Implementation, participatory monitoring in Maintenance, and knowledge reuse in Renewal. The arrows indicate continuous information flow and iterative learning between stages, reinforcing adaptive and embedded sustainability throughout the software lifecycle.

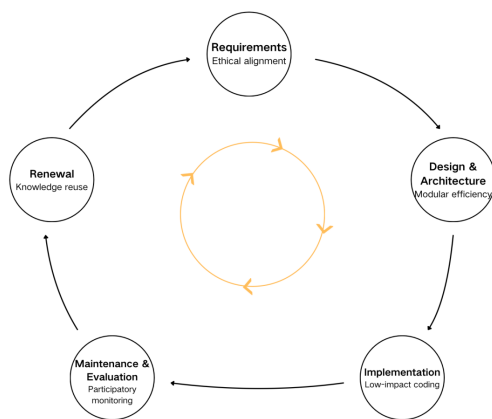


Figure 1. Structure of the Sustainability-Oriented Educational Software Lifecycle (SESL) Framework.

The circular model illustrates iterative feedback across the five lifecycle phases, with sustainability checkpoints embedded at each stage. These checkpoints link pedagogical objectives, technical processes, and ecological metrics within a continuous improvement loop. This representation distinguishes the SESL framework from linear lifecycle models by portraying sustainability as an evolving equilibrium rather than a fixed outcome. It emphasizes that enhancements in one phase—such as energy efficiency during Implementation—must inform and influence other phases, including Design refinement or updates to ethical requirements, to maintain holistic lifecycle integrity.

4.5. Discussion

The comparative evidence indicates that sustainability in EdTech is as much a governance challenge as a technical one. Platforms that emphasize openness and participatory practices, such as Moodle, align more closely with the SESL framework's participatory ethos but require structured lifecycle auditing to ensure long-term resilience. Commercial platforms like ClassDojo and Coursera achieve operational efficiency yet must institutionalize transparent sustainability metrics to balance profitability with social and environmental responsibility.

The SESL framework bridges the gap between theory and practice by providing both a diagnostic and prescriptive tool. It enables stakeholders to assess current practices, identify underdeveloped lifecycle phases, and implement targeted improvements. By integrating sustainability checkpoints into design and governance processes, EdTech developers can transition from reactive compliance to proactive stewardship, ensuring that digital education infrastructures remain resilient, inclusive, and ecologically accountable.

5. Conclusion

This study has developed and empirically grounded a Sustainability-Oriented Educational Software Lifecycle (SESL) framework that integrates technical, environmental, and pedagogical sustainability into a coherent developmental process. Through the comparative analysis of three representative EdTech platforms—Moodle, ClassDojo, and Coursera—the research identified both the strengths and the structural limitations of exist-

ing lifecycle practices. The cases revealed that while current systems exhibit partial sustainability-community resilience in Moodle, user engagement in ClassDojo, and scalability in Coursera-none achieves holistic integration of sustainability metrics across all lifecycle phases.

The SESL framework addresses this gap by embedding five iterative sustainability checkpoints-ethical alignment, modular efficiency, low-impact implementation, participatory maintenance, and knowledge reuse-into each development phase. Its circular, feedback-oriented structure reconceptualizes sustainability as an adaptive equilibrium rather than a terminal outcome, aligning software development with long-term ecological responsibility and educational continuity. The framework advances theoretical understanding by bridging sustainable software engineering and educational technology design, offering a multidimensional model that operationalizes sustainability through measurable actions and iterative learning processes.

From a practical perspective, SESL provides actionable guidance for developers, educators, and policymakers. It serves as a diagnostic and planning tool to assess sustainability maturity, prioritize interventions, and align technological innovation with pedagogical integrity and environmental accountability. By institutionalizing sustainability checkpoints within design governance, educational institutions can reduce system fragility, improve energy efficiency, and ensure the ethical stewardship of learning data.

Future research should focus on quantitative validation and cross-context adaptation of the SESL framework. Empirical studies could evaluate lifecycle performance across additional EdTech systems and explore its applicability in emerging domains such as AI-driven learning analytics, blockchain-based credentialing, or immersive XR learning environments. Furthermore, integrating SESL with global sustainability standards, particularly SDG 4 (Quality Education) and SDG 12 (Responsible Consumption and Production), may enhance its policy relevance. Ultimately, this research establishes a foundational pathway toward a new generation of sustainability-aware educational software ecosystems that harmonize innovation, inclusivity, and ecological resilience.

References

1. M. A. Adeoye, and B. I. Otemuyiwa, "Navigating the Future: Strategies of EdTech Companies in Driving Educational Transformation," *JERIT: Journal of Educational Research and Innovation Technology*, vol. 1, no. 1, pp. 43-50, 2024.
2. S. Urmanov, "USING VIRTUAL CLASSES, ONLINE LEARNING AND EDUCATIONAL PLATFORMS (EG COURSERA, MOODLE)," *Modern American Journal of Linguistics, Education, and Pedagogy*, vol. 1, no. 3, pp. 789-796, 2025.
3. M. Gavin, and S. McGrath-Champ, "Teacher workload and the organisation of work: a research agenda for a post-pandemic future," *Labour and Industry*, vol. 34, no. 1, pp. 88-99, 2024. doi: 10.1080/10301763.2024.2357891
4. J. Leong, K. May Yee, O. Baitsegi, L. Palanisamy, and R. K. Ramasamy, "Hybrid project management between traditional software development lifecycle and agile based product development for future sustainability," *Sustainability*, vol. 15, no. 2, p. 1121, 2023. doi: 10.3390/su15021121
5. H. Abuhassna, S. Alnawajha, F. Awae, M. A. B. M. Adnan, and B. I. Edwards, "Synthesizing technology integration within the Addie model for instructional design: A comprehensive systematic literature review," *Journal of Autonomous Intelligence*, vol. 7, no. 5, pp. 1-28, 2024.
6. M. Raza, K. S. Sakila, K. Sreekala, and A. Mohamad, "Carbon footprint reduction in cloud computing: Best practices and emerging trends," *Int. J. Cloud Comput. Database Manage*, vol. 5, no. 1, pp. 25-33, 2024.
7. U. O. Matthew, O. Asuni, and L. O. Fatai, "Green Software Engineering Development Paradigm: An Approach to a Sustainable Renewable Energy Future," In *Advancing Software Engineering Through AI, Federated Learning, and Large Language Models*, 2024, pp. 281-294.
8. S. Yang, "The impact of continuous integration and continuous delivery on software development efficiency," *Journal of Computer, Signal, and System Research*, vol. 2, no. 3, pp. 59-68, 2025, doi: 10.71222/pzvfqm21.
9. M. O. Onoja, C. C. Onyenze, and A. A. Akintoye, "DevOps and Sustainable Software Engineering: Bridging Speed, Reliability, and Environmental Responsibility," *International Journal of Technology, Management and Humanities*, vol. 10, no. 04, pp. 60-83, 2024. doi: 10.21590/ijtmh.10.04.08
10. S. Thielen, B. Salgert, and T. Franz, "From Lab to Market: Architectural Evolution in Open Source Transition," In *European Conference on Software Architecture*, August, 2025, pp. 306-322. doi: 10.1007/978-3-032-02138-0_20

11. I. OMOREGIE, H. ANTHONY, and J. J. BRAIMOH, "Comparative Analysis of Instructional Models for Designing Effective Online Courses: ADDIE, SAM, and Dick & Carey Approaches," *Journal of Languages and Translation*, vol. 5, no. 1, pp. 33-45, 2025.
12. D. I. Putrie, U. Salam, and D. Riyanti, "Unveiling the landscape of ClassDojo in education: A systematic review," *Elsya: Journal of English Language Studies*, vol. 6, no. 2, pp. 185-208, 2024. doi: 10.31849/elsya.v6i2.18422
13. M. Kommineni, and S. Chundru, "Sustainable Data Governance Implementing Energy-Efficient Data Lifecycle Management in Enterprise Systems," In *Driving Business Success Through Eco-Friendly Strategies*, 2025, pp. 397-418. doi: 10.4018/979-8-3693-9750-3.ch021
14. F. Gao, "The role of data analytics in enhancing digital platform user engagement and retention," *Journal of Media, Journalism & Communication Studies*, vol. 1, no. 1, pp. 10-17, 2025, doi: 10.71222/z27xzp64.
15. M. Radovan, and D. M. Radovan, "Harmonizing pedagogy and technology: Insights into teaching approaches that foster sustainable motivation and efficiency in blended learning," *Sustainability*, vol. 16, no. 7, p. 2704, 2024. doi: 10.3390/su16072704
16. M. Sadek, R. A. Calvo, and C. Mougenot, "Designing value-sensitive AI: a critical review and recommendations for socio-technical design processes," *AI and Ethics*, vol. 4, no. 4, pp. 949-967, 2024. doi: 10.1007/s43681-023-00373-7
17. I. Lanza-Cruz, R. M. Colás, A. Martínez-Martínez, and I. R. Quintana, "Advancing Virtual Learning: A Review of Moodle For The Optimization Of Online Education," *INTED2024 Proceedings*, pp. 4056-4064, 2024.

Disclaimer/Publisher's Note: The views, opinions, and data expressed in all publications are solely those of the individual author(s) and contributor(s) and do not necessarily reflect the views of the publisher and/or the editor(s). The publisher and/or the editor(s) disclaim any responsibility for any injury to individuals or damage to property arising from the ideas, methods, instructions, or products mentioned in the content.