Principles of DevOps Education: A Framework for Modern Teaching and Learning

DevOps education must reflect the realities of modern software engineering. To prepare students for professional practice, teaching must be authentic, iterative, collaborative, toolchain-aware, reflective, accessible, and adaptable. These principles form the foundation of a scalable and future-ready DevOps curriculum. This document presents a research-grounded framework outlining the essential dimensions of effective DevOps education and their implications for designing learning tools. It is intended as a practical reference for educators, curriculum designers, and developers committed to delivering meaningful, industry-aligned DevOps learning experiences.

Core Principles

1. **Authenticity:** Learning must involve realistic DevOps workflows — Git, CI/CD, IaC, Docker, monitoring, incidents — not oversimplified lab tasks. Students should experience end-to-end delivery, continuous integration, and real dependency between code, pipelines, and infrastructure.

2. **Iterative Learning and Feedback:** DevOps is defined by continuous improvement. Learning environments should mirror this through build–fail–debug–retry cycles supported by clear, immediate feedback and observable system behavior.

3. **Collaboration and Roles:** DevOps is a team sport. Students benefit from role rotation (Dev, Ops, QA), peer reviews, shared ownership, and communication practice that reflects real professional workflows.

4. **Toolchain Exposure and IaC:** Confidence with Git workflows, Docker, pipelines, IaC, Kubernetes, and DevSecOps practices is critical. Graduates must be able to navigate authentic tooling ecosystems used in industry.

5. **Balanced Mentorship:** Students need guidance that reduces cognitive overload without removing autonomy. Scaffolded hints, checkpoints, example tasks, and micro-tutorials help learners progress while keeping instructors' workload manageable.

6. **Reflective Practice:** DevOps culture relies on retrospectives, postmortems, and shared learning from failure. Reflection builds metacognition and deep conceptual understanding.

7. **Accessibility and Inclusiveness:** Browser-based, low-friction environments avoid the typical barriers of complex local setup, making them more accessible and scalable.

8. **Modularity and Adaptability:** DevOps evolves quickly and so must the teaching. Modular content, extensible tools, and progressive learning pathways ensure long-term relevance.

These principles serve as the foundation for the DevOps PBL Framework and the tool design guidelines presented in this handout.

Contents

1 Purpose of This Handout **1**

2 DevOps PBL Framework **1**

2.1 Authenticity and Real-World Context 2
2.2 Iterative Learning and Continuous Feedback 2
2.3 Collaboration, Roles, and Soft Skills 2
2.4 Toolchain Exposure and Infrastructure-as-Code 2
2.5 Mentorship and Balanced Autonomy 2
2.6 Reflective and Metacognitive Practice 2
2.7 Accessibility and Inclusiveness 3
2.8 Modularity and Adaptability 3

3 Design Implications for DevOps Education Tools **3**

4 Conceptual Tool Design: Operation DevOps **3**

5 Mapping Model **4**

1 Purpose of This Handout

This handout presents a research-based framework for teaching DevOps through Project-Based Learning (PBL), along with design implications for developing web-based or hybrid learning tools. The material is intended to serve as a practical reference for educators, curriculum designers, and developers creating DevOps learning environments.
The information has been collected through a literature review that includes work published between 2019 and 2025 on DevOps education, software engineering education, Git pedagogy, cloud and IaC learning, monitoring and incident-based instruction, reflective learning design, PBL/Challenge-Based-Learning (CBL)/System-Based-Learning (SBL) pedagogical models.
The goal is to give educators and learning-tool designers a clear, actionable blueprint for:

- Structuring DevOps courses
- Designing labs and learning journeys
- Creating web-based/simulation tools
- Closing skill gaps identified in academia and industry
- Supporting scalable, authentic, and feedback-rich DevOps learning

For this purpose, the document includes:

- A complete DevOps PBL Framework with eight pedagogical dimensions.
- The research justification for each dimension.
- Design implications derived from the literature.
- A tool concept developed following implications of the framework.
- A mapping model linking challenges and gaps to framework dimensions, design implications, and tool features.

2 DevOps PBL Framework

The DevOps PBL Framework consists of eight core pedagogical dimensions. Each is motivated by recurring findings in the DevOps education literature.

2.1 Authenticity and Real-World Context

Students learn through realistic workflows that replicate modern DevOps practices: Git branching, pull requests, CI/CD pipelines, Infrastructure-as-Code (IaC), monitoring, containerization, and incident response. Research emphasizes that academic exercises are often too abstract and lack realism (Alves & Rocha, 2021; Bobrov et al., 2020; Sánchez-Cifo et al., 2023; Sarmiento-Calisaya et al., 2024). Authenticity improves student confidence and job readiness.

2.2 Iterative Learning and Continuous Feedback

DevOps itself is defined by continuous improvement. Learning environments should mirror this through build–fail–debug–retry cycles and automated feedback. Studies highlight the lack of immediate, actionable feedback in traditional courses (Ferino et al., 2021; Hobeck et al., 2021; Lanubile et al., 2023).

2.3 Collaboration, Roles, and Soft Skills

DevOps is inherently collaborative. Students benefit from role rotation (Dev, Ops, QA), peer review structures, and communication training. Several studies identify weak teamwork preparation as a key gap (Cardoso et al., 2021; Kuusinen & Albertsen, 2019; Zarour et al., 2024).

2.4 Toolchain Exposure and Infrastructure-as-Code

Exposure to Git, pipelines, Docker, Kubernetes, and IaC is essential. Research shows graduates often lack confidence working with real DevOps toolchains, especially IaC and Kubernetes (Demchenko et al., 2019; Rahman et al., 2024; Sarmiento-Calisaya et al., 2024).

2.5 Mentorship and Balanced Autonomy

Students need guidance that reduces cognitive overload without limiting autonomy (Capozucca et al., 2019; Vicente & Cunha, 2022). Scaffolded hints, checkpoints, and micro-tutorials support learning while maintaining agency.

2.6 Reflective and Metacognitive Practice

Reflection, journaling, postmortems, and retrospectives are central to DevOps culture but underrepresented in education. Reflection improves retention and conceptual understanding (Bonetti et al., 2025; Capozucca et al., 2019; Lanubile et al., 2023).

2.7 Accessibility and Inclusiveness

Browser-based or low-friction environments enable inclusive participation by removing technical barriers. Research notes that complex local setups hinder learning and motivation (Iyer et al., 2024; Wang et al., 2021).

2.8 Modularity and Adaptability

DevOps evolves quickly. Educational tools must be modular, flexible, and updateable. Modular progression from beginner to advanced is widely recommended (Capozucca et al., 2019; Demchenko et al., 2019; Wang et al., 2021).

3 Design Implications for DevOps Education Tools

Based on the framework and literature synthesis, the following design implications guide tool development:

- Progressive Git/branching pedagogy and visual commit graphs
- Automated CI/CD feedback with realistic logs
- Incident-driven monitoring and traceability
- IaC "plan" previews with linting and rule checks
- Tiered, contextual hints and micro-tutorials
- Guided scenarios using example-based learning
- Structured reflection prompts integrated into workflow
- Browser-based accessibility to avoid environment complexity
- Plugin-based architecture to enable future technology updates

4 Conceptual Tool Design: Operation DevOps

The learning tool concept operationalizes the PBL framework dimensions through the following components:

- Git workflow simulation (branches, commits, merges, PRs)
- CI/CD pipeline simulation with build, lint, test, and IaC checks
- Feedback engine generating contextual, multi-level hints
- IaC editor with rule engine and plan preview

- Dockerfile correctness checking
- Monitoring dashboard with simulated incidents
- Reflection journal with structured prompts and export options
- Modular progression system (beginner to advanced)

5 Mapping Model

Figure 1 shows a high-level conceptual mapping from known educational gaps to the framework, design implications, and tool features.

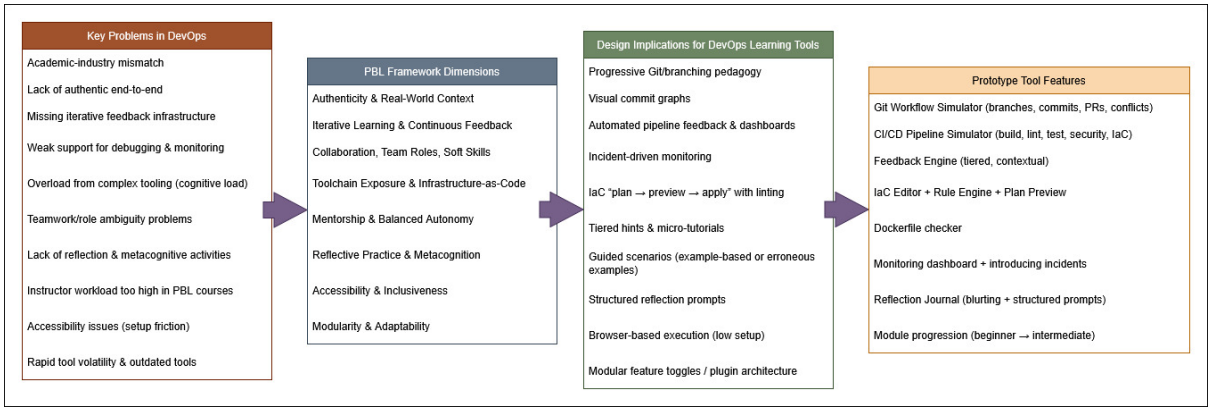

Figure 1: Mapping Model: Linking Literature Gaps, Framework Dimensions, Design Implications, and Tool Features

References

Alves, I., & Rocha, C. (2021). Qualifying software engineers undergraduates in devops - challenges of introducing technical and non-technical concepts in a project-oriented course. *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 144–153. https://doi.org/10.1109/ICSE-SEET52601.2021.00024

Bobrov, E., Bucchiarone, A., Capozucca, A., Guelfi, N., Mazzara, M., & Masyagin, S. (2020). Teaching devops in academia and industry: Reflections and vision. In *Software engineering aspects of continuous development and new paradigms of software production and deployment* (Vol. 12055). Springer. https://doi.org/10.1007/978-3-030-39306-9_1

Bonetti, T. P., Silva, W., & Colanzi, T. E. (2025). Example-based learning in software engineering education: A systematic mapping study. https://arxiv.org/abs/2503.18080

Capozucca, A., Guelfi, N., & Ries, B. (2019). Design of a (yet another?) devops course. In *Software engineering aspects of continuous development and new paradigms* (Vol. 11350). Springer. https://doi.org/10.1007/978-3-030-06019-0_1

Cardoso, T., Chanin, R., Santos, A., et al. (2021). Combining agile and devops to improve students' tech and non-tech skills. *Proceedings of the 13th International Conference on Computer Supported Education*.

Demchenko, Y., Zhao, Z., Surbiryala, J., et al. (2019). Teaching devops and cloud based software engineering in university curricula. *2019 15th International Conference on eScience*, 548–552. https://doi.org/10.1109/eScience.2019.00075

Ferino, S., Fernandes, M., Fernandes, A., Kulesza, U., Aranha, E., & Treude, C. (2021). Analyzing devops teaching strategies: An initial study. *Proceedings of the 35th Brazilian Symposium on Software Engineering (SBES '21)*, 180–185. https://doi.org/10.1145/3474624.3477071

Hobeck, R., Weber, I., Bass, L., & Yasar, H. (2021). Teaching devops: A tale of two universities. *SPLASH-E 2021*, 26–31. https://doi.org/10.1145/3484272.3484962

Iyer, G. N., Yisheng, A. G., Er Metilda, C. H., Choong, W. X., & Koh, S. W. (2024). A web-based ide for devops learning in software engineering higher education. *2024 IEEE TALE*, 1–8. https://doi.org/10.1109/TALE62452.2024.10834361

Kuusinen, K., & Albertsen, S. (2019). Industry-academy collaboration in teaching devops and continuous delivery. *2019 IEEE/ACM ICSE-SEET*, 23–27. https://doi.org/10.1109/ICSE-SEET.2019.00011

Lanubile, F., Martínez-Fernández, S., & Quaranta, L. (2023). Teaching mlops in higher education through project-based learning. *2023 IEEE/ACM ICSE-SEET*, 95–100. https://doi.org/10.1109/ICSE-SEET58685.2023.00015

Rahman, M. M., Barek, M. A., Akter, M. S., et al. (2024). Authentic learning on devops security with labware: Git hooks to facilitate automated security static analysis.

2024 IEEE COMPSAC, 2418–2423. https://doi.org/10.1109/COMPSAC61105.2024.00388

Sánchez-Cifo, M. Á., Bermejo, P., & Navarro, E. (2023). Devops: Is there a gap between education and industry? *Journal of Software: Evolution and Process*, *35*(12), e2534. https://doi.org/10.1002/smr.2534

Sarmiento-Calisaya, E., Mamani-Aliaga, A., & Leite, J. C. S. D. P. (2024). Introducing computer science undergraduate students to devops technologies from software engineering fundamentals. *46th International Conference on Software Engineering: Software Engineering Education and Training*, 348–358. https://doi.org/10.1145/3639474.3640071

Vicente, A., & Cunha, J. (2022). Applying the devops methodology for a more efficient process of teaching-learning computer programming. *2022 EAEEIE Conference*, 1–6. https://doi.org/10.1109/EAEEIE54893.2022.9820399

Wang, S., Neupane, R., Pandey, A., Cheng, X., & Calyam, P. (2021). Online learning platform for application-inspired cloud and devops curriculum. *2021 IEEE HiPCW*, 35–42. https://doi.org/10.1109/HiPCW54834.2021.00012

Zarour, M., Akour, M., & Alenezi, M. (2024). Enhancing devops engineering education through system-based learning approach. *Open Education Studies*, *6*(1), 20240012. https://doi.org/10.1515/edu-2024-0012