

Bringing AI into the Classroom: A Structured Approach for Integrating AI into Software Engineering Education

Iris Groher (Johannes Kepler University Linz, Institute of Business Informatics – Software Engineering; iris.groher@jku.at)

Michael Vierhauser (University of Innsbruck, Department of Computer Science; Michael.Vierhauser@uibk.ac.at)

Markus Weninger (Johannes Kepler University Linz, Institute for System Software; markus.weninger@jku.at)

DOI:

ABSTRACT

The recent emergence of generative AI and Large Language Models (LLMs), particularly following the release of ChatGPT in late 2022, has significantly impacted both academic research and industrial practice. This development has vast potential to impact educational practices, particularly computer science and software engineering courses. Unfortunately, there is still a lack of actionable guidance on how to coherently integrate AI into computer science curricula. In this paper, we therefore introduce the concept of *AI-Blueprints*, a structured approach to integrating AI-related topics and activities into various computer science courses. We describe our approach and outline a structured process for creating new blueprints. Our vision is to provide these blueprints as open educational resources, allowing educators to adapt and integrate AI into diverse courses and topics. As a preliminary validation, we conducted semi-structured interviews with six university-level educators, collecting feedback on how our blueprints could help to integrate AI topics into existing courses. Based on this feedback, we outline plans for future research and expanding our AI-Blueprint concept.

1 INTRODUCTION

Artificial Intelligence (AI) has been an active research area for decades, largely driven by significant advances in machine learning algorithms and neural networks [25]. However, only recently has the rise of generative AI and transformer-based Large Language Models (LLMs) dramatically changed both academic research and industrial practice [16]. With the release of OpenAI’s ChatGPT in late 2022, LLMs have become omnipresent, making sophisticated language-processing capabilities and even programming assistance [26] widely accessible.

Given the rapid growth and broad accessibility of LLM-based tools, examining their implications for educational contexts has become increasingly important, particularly in software engineering (SE) education, which is currently under substantial transformation through the integration of LLMs [15]. Recent work in SE education has highlighted both the potential and limitations of LLMs, demonstrating their effectiveness in supporting code comprehension, automating formative feedback, and many other aspects [11, 13, 17]. However, so far, integration of AI-related activities into existing SE courses and curricula is still sparse and unstructured at best, with actionable guidance for a coherent integration of LLM activities into computer science (CS) courses still missing. Therefore, in this paper, we propose our initial ideas towards a novel concept of *AI-Blueprints*. Inspired by Use Cases

from Requirements Engineering [8], which provide a structured template to describe a specific function of a system, we provide a structured approach to integrate AI-related topics and activities when planning CS courses.

Our contributions include a description of the AI-Blueprint format and how to apply it, complemented by a structured process for creating new blueprints (Section 3), an initial report on our experience of applying the blueprint to three SE courses/-topics, complemented by six semi-structured interviews with educators (Section 4), and a discussion and a roadmap for future research (Section 5).

2 BACKGROUND & RELATED WORK

The ACM Curriculum guidelines [18] represent a reference for structuring CS education. It has been widely recognized and adopted [27], and also serves as guidance for our envisioned blueprints. The recent version CS2023¹ provides a modular framework of knowledge areas and units that define key topics, competencies, and learning outcomes, which can be adapted to local needs while maintaining global standards. Particular emphasis is placed on Software Engineering (SE) and Software Development Fundamentals (SDF), which cover essential competencies from programming basics and debugging to software design, testing, and project management. However, as noted by Vierhauser et al. [29], few AI-related educational studies have targeted these areas. In this paper, we focus on SE and SDF, providing detailed lecture blueprints, and illustrate the adaptability of our proposed AI-Blueprint with an additional example from the Algorithmic Foundations (AL) area.

Different studies highlight the importance of integrating AI into SE curricula. Several works focus on *student perceptions and usage*. Baresi et al. [3] examine LLM integration across universities, showing that students find them more useful for coding and debugging than for higher-level tasks, while Yabaku and Ouhbi [30] report frequent use for code optimisation and idea generation.

Other studies discuss *course- and lecture-level integration strategies*. Abdelfattah et al. [1] propose embedding ChatGPT in requirements engineering lectures to support user story and diagram creation, fostering active participation and critical thinking. Li et al. [19] outline a framework aligned with the SE2014 Knowledge Areas, demonstrating integration into design and practical SE courses with activities such as code generation, bug detection, testing, and documentation.

¹<https://csed.acm.org>

Different authors address *pedagogical frameworks and broader challenges*. Sah et al. [22] compare dedicated AI/LLM courses, hybrid methods, and comprehensive programs, discussing their impact on technical and ethical competencies, as well as challenges such as limited resources, rapid technological change, and continuous training needs. Sengul et al. [24] analyse conversational AI in SE, reporting early benefits for engagement in programming courses but also a gap between industry practice and higher education. Kirova et al. [17] call for adapting SE education to prepare graduates for an AI-driven future, including ethical awareness, while Dickey et al. [9] propose AI-Lab, a four-phase pedagogical framework that guides educators to integrate generative AI into programming courses.

Although all of these approaches provide important insights into how AI and LLM-specific approaches can be used in an educational context, a systematic framework that aligns existing course learning objectives with accompanying AI activities is still missing. In the following, we introduce our AI-Blueprints and process as a structured response to this gap.

3 AI-BLUEPRINTS: A GUIDING TEMPLATE

Successful integration of AI into SE education requires a systematic methodology. Inspired by Use Case models in Requirements Engineering [8], we introduce AI-Blueprints, a structured template-based approach that supports educators in integrating AI-related content and activities into SE lectures. Each blueprint represents a reusable instructional pattern that combines existing learning objectives with AI concepts. The template structure enables adaptation across courses, educational levels, and institutions, and supports modular lecture design that allows incremental integration of AI activities without requiring a complete course redesign.

3.1 AI-Blueprint Description

As outlined in the simplified example blueprint² in Table 1, our proposed AI-Blueprint template provides a structured and systematic representation for educators to clearly outline, organise, and document the educational components augmented with AI. It comprises five main parts: (1) General Course and Lecture Description; (2) Lecture Contents & Learning Objectives; (3) Lecture Activities & Resources; (4) Lecture Reflection; and finally (5) Miscellaneous Info.

(1) General Course and Lecture Description: The first part provides general information about the course and the specific lecture. It aligns the lecture to one of the Knowledge Areas from the ACM CS2023 curriculum (e.g., Software Development Fundamentals – SDF), supporting comparability across institutions, and clearly positions the educational context within established SE educational guidelines. It further refines this mapping to the specific Knowledge Unit within the selected area (e.g., Programming Basics). The Course Modalities specify the instructional format(s) (e.g., lecture, lab) and durations. Lecture Context situates the individual lecture within the broader course context by identifying its

topic(s) and intended learning outcomes. Keywords list thematic terms for searching.

(2) Lecture Contents & Learning Objectives: Beyond the basic content description of the first part, the second part facilitates the structured definition of learning objectives (LOs) tailored specifically for the dedicated course lecture. Our AI-Blueprints complement existing LOs by integrating additional AI-related objectives together with related tasks and activities without replacing established lecture content. It categorises both lecture-specific and AI-related LOs using Bloom’s taxonomy and/or the *skill levels* proposed in CS2023. Bloom’s (Revised) Taxonomy defines six hierarchical cognitive processes: Remember, Understand, Apply, Analyze, Evaluate, and Create, allowing educators to formulate measurable objectives at appropriate complexity levels [2]. For the learning objectives, we use the skill levels (Explain, Apply, Evaluate, Develop) from the CS2023 framework [18]. We further complement these levels by common application types of AI in education [20] (Generate, Explain, Evaluate) for the AI-related learning objectives.

(3) Lecture Activities & Resources: This part represents the core elements of our proposed AI-Blueprint. The activity-centered design emphasises the idea of “active learning” – that students learn most effectively when they are directly engaged in meaningful, goal-oriented tasks [12]. Describing small activities allows the adoption of individual parts of the blueprint, e.g., picking individual activities and integrating them into own lectures. Each lecture activity is described following a standardised structure. To follow the principle of Constructive Alignment [5], each activity is linked to the LOs and AI-LOs that this activity targets. This mapping ensures that the intended learning outcomes, the teaching/learning activities, the AI integration, and the assessment strategies are aligned. The *Description* provides an overview of the student task in the form of a scenario, together with the intended collaborative format. Additionally, the *Resources* part provides additional information about necessary teaching and learning materials, e.g., data, input, or material provided for the tasks, or specific sample prompts that can be used with an LLM. Finally, the *Assessment* part provides a description and examples of how student performance can be evaluated, and how feedback should be provided. In addition to in-class activities, the *Assignment/Homework/Additional Activities* part can cover additional tasks targeted to be performed out of class, designed to reinforce both lecture-specific and AI-enhanced skills outside class.

(4) Lecture Reflection: Complementing individual activities, it is essential for students to critically reflect on their learning experiences at the end of a lecture, with a focus on both their interaction with AI tools and the broader implications of AI integration in their work. This part guides structured student reflection, including both open-ended discussion questions, as well as questions fostering critical thinking, encouraging students to, on the one hand, share their experiences with the AI tools, and further ask students to evaluate trust, risk, and ethical implications.

(5) Misc: This part lists any additional resources that support the

²The full template, as well as all our example AI-Blueprints are available on GitHub (<https://github.com/TeachingAndLearningSciences/AIBlueprint>)

lecture and reflection activities.

3.2 AI-Blueprint Development Process

To provide guidance and step-by-step instructions for developing new AI-Blueprints, a process for blueprint creation and adaptation for specific courses and lectures is necessary. Inspired by the AI-Lab framework [9], we are convinced that detailed guidance can support educators in creating novel AI-Blueprints or in the modification of pre-existing ones to align with their particular course requirements. As part of this, we have drafted an initial outline, sketching out the core steps necessary for using our proposed AI-Blueprint template.

(1) Select Course and Lecture: To identify courses and lectures, educators should first identify LOs in their courses that require skills that could be deepened through AI-assisted tasks. Educators might also think of lectures and topics in which students struggle with comprehension or productivity. In addition, educators can search our online repository² to find AI use cases in SE teaching and adapt them to their context.

(2) LO- and AI-LO-Definition: To integrate AI-related activities into a chosen lecture, we follow a learning-objective-centered approach. First, *LOs should be assessed and selected*. LOs are widely recognised as a key instrument for defining structured and clearly articulated learning outcomes, and have become a standard component in many curriculum development processes [28, 14]. Once appropriate LOs have been identified, corresponding *AI-related LOs can be defined*. These supplementary LOs explicitly target AI skills and knowledge, and may define new LOs, or serve as extensions of existing course objectives.

(3) Prepare (AI) Activities: Once LOs and AI-LOs are identified and defined, the blueprint can be created and populated with *specific tasks and activities where AI tools and approaches can be used*. Additionally, define assessment criteria by specifying both formative and summative assessments, including diagnostic questions and identification of potential misconceptions. Employ constructive alignment to ensure that all defined LOs, both course-specific and AI-related, are appropriately targeted and assessed in at least one activity. Once activities and corresponding assessment criteria are defined, *appropriate AI tools and technologies can be selected* based on their alignment with defined AI tasks and suitability for the student's skill level. Prepare these tools by configuring them for instructional use, by developing user guides, tutorials, or demonstrations.

Lecture Material Preparation: Once concrete activities and tools are selected, *corresponding lecture materials can be developed*. Materials should go beyond general instructional design by explicitly addressing teaching with and about AI. Prepare lecture slides, prompts, and exercises that teach students core AI concepts such as prompt engineering, LLM capabilities, and limitations of AI-generated outputs. Include examples that demonstrate how AI tools can support SE tasks, while also highlighting failure cases or misleading outputs.

(4) Post Activity Preparation: Besides the AI-related tasks and activities, it is important that students learn how to *critically*

reflect and discuss the usage of AI tools. While, for example, LLMs can be valuable tools in educational and SE contexts, they are not infallible. Therefore, it is important to not only teach students how to use these tools and create prompts but to foster discussion on how to interpret results, assess their quality, and improve in case they are not satisfactory. Post-lecture activities should facilitate student reflection and critical thinking.

(5) Assess and Update Materials: Finally, once a blueprint is created and used in practice, the experiences and feedback should be used to update and continuously improve the lecture tasks. Evaluate the effectiveness of the implemented AI tasks and update materials accordingly, incorporating feedback derived from formative and summative assessments to ensure continuous improvement and alignment with LOs.

4 PRELIMINARY VALIDATION

As a first step towards validating the applicability and usefulness of our AI-Blueprint, we created an initial set of publicly accessible AI-Blueprints for three lectures within the knowledge areas Software Development Fundamentals, Algorithmic Foundations, and Requirements Engineering. Each blueprint was developed by one of the authors, who tailored it to the specific context, LOs, and challenges of a course they teach. Throughout the process, we conducted several internal peer reviews. Feedback was incorporated to improve the structure and clarity of each blueprint. The final blueprints were published as open teaching materials in a repository.

The first blueprint was developed for an *introductory programming course in Python*, focusing on using AI assistants to support students working with pandas. Activities include prompting AI for code explanations, generating data transformation snippets, and evaluating the correctness of AI-generated responses. The second blueprint targets an *algorithms and data structures course*. Students are guided to prompt AI systems to explain algorithmic concepts, compare solutions, and critique AI-generated code. Emphasis is placed on recognising when AI-generated explanations are incorrect or incomplete. The third blueprint integrates AI into an *SE course module on behaviour-driven development*. Activities include generating user stories with AI, comparing AI-generated test cases to manually written ones, and critically assessing coverage and quality.

Based on these blueprints, we collected feedback and conducted a series of semi-structured interviews with participants from two different universities. The goal was to gain insights and a deeper understanding of the current usage of AI tools and techniques as part of university courses, potential challenges educators face, and whether a structured process and blueprint, together with open-source teaching materials, could alleviate the effort for creating these materials from scratch.

The objective of the preliminary scoping interviews was to gather initial feedback about our proposed process and AI-Blueprints while also identifying opportunities for further improvement and extension. All interview material, questionnaires, and initial version of the blueprints are available online².

1. General Course and Lecture Description		
ACM Category	Software Development Fundamentals (SDF)	
Knowledge Unit	SDF: Programming Basics	
Course Overview	The introductory Python for Business Administration course offers students a foundation in core programming concepts—such as variables, control flow, and functions—based on examples from a business context. It emphasizes practical problem-solving skills and data-driven decision making within a business context.	
Course Modalities	Lab 90 minutes	
Lecture Context	The pandas lecture introduces students to the pandas library’s key data structures (Series and DataFrame) and fundamental operations—such as filtering, grouping, basic aggregation, plotting—enabling them to efficiently manipulate and analyze tabular data relevant to business scenarios.	
Keywords	Python, Pandas, Data Analytics	
2. Lecture Contents & Learning Objectives		
Lecture-related LOs	Students are able to ... – LO_C1: Describe structure/components of a DataFrame, and explain key operations. [UNDERSTAND] – LO_C2: Load tabular data and clean it. [APPLY]	
AI-related LOs	– LO_AI1: Use AI to explain pandas structures (index, columns, dtypes). [UNDERSTAND] [EXPLAIN] – LO_AI2: Use AI to generate/modify pandas code (load, clean, group, visualize). [APPLY] [GENERATE] – LO_AI3: Critically assess AI-generated workflows and refine. [EVALUATE]	
3. Lecture Activities & Resources		
Description	Students use AI to explore, generate, and refine pandas code. They explain structures, execute AI suggestions, and co-create merged code from AI and hand-written examples.	
Activities	3.1-Activity	AI-based Explanation of Pandas Concepts
	Co. Algmt.	LO_AI1
	Description	Use an AI assistant to explore DataFrame structure and operation semantics. Individually, students write a natural-language prompt such as P1. In pairs, they submit their prompt to ChatGPT (or Github Copilot) and read the explanation.
	Resources	P1: “You are a pandas expert and instructor, known for clear and concise explanations of DataFrame internals and operations. Explain what df.dtypes tells me about my DataFrame, and how I might change a column’s dtype.” [EXPLAIN] [Persona Pattern]
	Assessment	Each pair quickly presents something they learned that was new or well explained and something they found confusing, wrong, or incomplete.
	3.2-Activity	AI-based Code Generation
	Co. Algmt.	LO_AI2
	Description	Use AI to generate, execute, and adapt pandas snippets for concrete tasks. Students get a CSV file and individually prompt the AI—e.g. P2. They paste the AI’s suggestion in their Jupyter notebook, run it, then manually modify one parameter (e.g. change the plot to a horizontal bar, add labels) to fit a new requirement.
	Resources	P2: “Write pandas code to load a CSV, drop duplicate rows, group sales by region, and plot them. I am going to provide a code template. Everything in ALL_CAPS is a placeholder. Please preserve the structure exactly.” [GENERATE] [Template Pattern] CSV file with sales data
	Assessment	Volunteers briefly show their adapted snippet.
Assignment/Homework/Additional Activities	A take-home notebook in which students must: – Use an AI assistant to generate pandas code for a specified task. (LO_AI2) – Critically reflect the generated code. (LO_AI3) – Make at least two nontrivial manual adaptations and comment why. (LO_AI3)	
4. Lecture Reflection		
Discussion	–In what scenarios did the AI help you most? –Where did you need to intervene or correct its suggestions? –How will you integrate AI support in your future analyses? –Can AI tools fully replace manual coding?	
Critical Thinking	– In which situations would I trust AI-generated code without a manual review? – How might unnoticed AI-introduced errors impact real business decisions? – Could using AI in data analysis introduce bias or privacy risks? How would I detect them?	
5. Misc. Information		
References/Material	Prompt pattern catalog: https://arxiv.org/abs/2302.11382 , Pandas documentation: https://pandas.pydata.org/docs	

Table 1: Excerpt of an AI-Blueprint: Pandas Lecture in Python (the full blueprint is available online²)

Interview Setup & Process: One researcher created an initial set of questions, which was then discussed among the authors, and the final questionnaire was divided into three parts, with a total of eleven open-ended questions and three Likert-scale questions. In the first part, we asked participants whether and how they themselves, or their students, used AI tools in their courses. Follow-up questions explored specific AI applications, challenges encountered, and whether any structured approach to AI integration was already in use. After this first part, we briefly introduced the concept of our AI-Blueprints and the blueprint creation process. We deliberately did not introduce our blueprints before part one to avoid any bias when discussing the current state. The second part focused on presenting the example blueprints and the participants’ initial impressions of these blueprints, including perceived usefulness, anticipated strengths and weaknesses

compared to their current teaching practices, and willingness to adopt or share such resources. Participants were also invited to comment on missing elements or suggestions for improvement. Finally, we asked the participants to rate the process, template, and their availability as open materials on a 5-point Likert scale.

To ensure diverse perspectives, we selected participants with varying levels of teaching experience and responsibilities, including early-career PhD students involved in lab sessions and group work, senior lecturers, and professors responsible for entire course modules. Our study included six participants from two universities involved in software engineering and related courses, including PhD students, senior lecturers, and a professor, with teaching experience ranging from 1 to over 30 years. Their teaching covered a broad spectrum of subjects such as software development, software engineering, software architecture, databases, AI, and al-

gorithms. Each interview was conducted by at least one researcher in person or via Zoom, lasting approximately 45 minutes. After asking for consent, we recorded all interviews and subsequently transcribed them using the OpenAI Whisper³ speech-to-text service. After transcribing the interviews, we used open coding [7] where two researchers coded the transcripts, collecting relevant information about (1) current practices and challenges, (2) feedback about the blueprints, and (3) potential improvements and extensions.

Interview Results: The interview participants generally recognised the significant potential of integrating AI tools and activities into courses, particularly within CS curricula. They noted AI's broad usage among students for tasks such as code generation, documentation, and testing, highlighting benefits such as efficiency gains and improved resource access. However, they also expressed concerns about the fact that students are potentially developing an over-reliance on AI tools, which in turn could hamper the development of basic problem-solving skills and critical thinking. Participants emphasised the necessity of clearly communicating the *complementary role of AI*, ensuring that students maintain foundational knowledge and competencies rather than using AI as a complete substitute.

Regarding the proposed blueprint and the structured process, the participants all regarded them positively as valuable resources to enhance teaching effectiveness. Five participants found the process and the template to be helpful (4) or very helpful (5). The openly available teaching material was rated as helpful or very helpful by all interviewees. However, they raised practical concerns about the potential overhead and flexibility of using predefined templates, emphasising the importance of adaptability to individual teaching styles and course requirements. In addition, they suggested the development of a clear didactic framework around AI competencies and task designs, arguing that this would improve the applicability and acceptance of such templates among educators. There was strong support for making these blueprints openly available as teaching resources. However, participants stressed the importance of adaptation and continuous updates aligned with rapid technological developments in the field of AI.

5 DISCUSSION & PLANNED RESEARCH

The initial proof-of-concept application of the AI-Blueprints and the responses from the interviews have provided initial evidence that they can be successfully applied to different lectures and course types. Furthermore, the interviews have confirmed that they could be helpful for educators of all experience levels to introduce AI-related topics in their courses without the need to redesign and/or restructure an entire course completely. Although participants acknowledged that the blueprints and process are generally helpful for introducing AI-related tasks, we also received several comments and suggestions for improvements, adding additional details about the LOs, and the structure of how we present individual activities. Based on the feedback of the interview participants, we identified several areas of opportunity for further

research and improvements of our AI-Blueprints.

• **More detailed Guidelines and Curriculum Integration:** A participant (PhD level with fewer teaching experience) mentioned that *“the shared blueprints show how the template is meant to be used”* and that *“it might be hard to fill in a new template for a lecture completely from scratch”*. Offering a broader range of AI-Blueprints as open teaching materials for different course categories can support educators in developing their own blueprints by building on existing examples. However, we are convinced that in addition to providing individual lecture blueprints, in parallel, a broader discussion is required on how to integrate general AI concepts into CS education [23, 4]. Although competency models and learning objectives in the education of CS and SE are well established [21], AI/LLMs as a means to solve SE tasks are still missing. As one concrete future contribution in this direction, we will work towards creating a general competency model for AI-related LOs that can be tailored and adapted to specific course content. Combining this with Bloom's levels to categorize educational goals, or the skill levels described in CS2023, can serve as a starting point for deriving meaningful AI-LOs.

• **Diversity of AI Activities:** A second key aspect concerns the diversity of AI-supported activities, which relates to the notion of AI-related LOs discussed above. While LLMs are often associated with content generation, their potential in education should extend beyond that. They can be used for interactive Q&A, personalised tutoring, feedback generation, and the iterative refinement of exercises, among other applications [11, 6, 13].

In CS education, activities involving AI and LLMs should extend beyond simple code generation. Particularly in introductory programming courses, students should critically engage with these tools through activities such as reflecting on the reliability and limitations of AI outputs, peer-review supported by model-generated feedback, and assessments that address ethical and integrity considerations. These activities foster technical competence, critical thinking, and responsible AI use. Aligning such AI-based activities with specific LOs and cognitive levels enables educators to formulate clearer and more measurable AI-related LOs. To support this process, we have started to systematically map common AI activities to relevant cognitive processes to guide the design of aligned learning experiences.

• **Systematic Organisation of Materials:** Participants found the blueprint template useful for structuring AI-based activities, but highlighted the need for better ways to search and organise them. Particularly for providing open-education materials on a larger scale, better ways of organising, linking them to other existing (non-AI) resources, and presenting the existing blueprints are needed. To address this, we are developing a web platform that enables structured exploration of AI-Blueprints, including suitable tools and their alignment with educational tasks. To improve reusability, we plan to categorize activities and blueprints along multiple pedagogical dimensions, such as Bloom's levels, AI activity type, and skill level.

• **Curricula and Long-Term Perspectives:** The AI-Blueprints

³<https://openai.com/research/whisper>

offer a concrete entry point for redesigning lectures, but they also raise broader questions about the continued relevance of existing LOs. As AI increasingly supports routine tasks, curricula must be revisited to determine which competencies remain essential, which should be adapted, and which new objectives should be added [10].

6 CONCLUSION

The emergence of generative AI and LLMs represents both an opportunity and a challenge at the same time, forcing educators to rethink what knowledge and skills students need to acquire in CS and SE courses. Although the rapid growth of these technologies has the potential to improve learning experiences, educators face notable difficulties in coherently embedding AI-related content into existing curricula and courses. Our proposed AI-Blueprints address this need by offering a structured, adaptable framework – envisioned as open educational resources – to facilitate the meaningful integration of generative AI technologies into SE lectures and across CS curricula. Further empirical research is necessary to refine and expand our AI-Blueprints, fostering broader adoption across different educational contexts.

REFERENCES

- [1] A. M. Abdelfattah, N. A. Ali, M. A. Elaziz, and H. H. Ammar. Roadmap for Software Engineering Education using ChatGPT. In *Proc. of the 2023 Int'l Conf. on AI Science and Applications in Industry and Society*, pages 1–6. IEEE, 2023.
- [2] L. W. Anderson and D. R. Krathwohl, editors. *A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives*. Allyn & Bacon, 2001.
- [3] L. Baresi, A. D. Lucia, A. D. Marco, M. D. Penta, D. D. Ruscio, L. Mariani, D. Micucci, F. Palomba, M. T. Rossi, and F. Zampetti. Students' Perception of ChatGPT in Software Engineering: Lessons Learned from Five Courses. In *Proc. of the 37th Int'l Conf. on Softw. Eng. Education and Training*. IEEE, 2025.
- [4] M. Bearman, J. Ryan, and R. Ajjawi. Discourses of artificial intelligence in higher education: A critical literature review. *Higher Education*, 86(2):369–385, 2023.
- [5] J. Biggs. Enhancing teaching through constructive alignment. *Higher education*, 32(3):347–364, 1996.
- [6] D. Cambaz and X. Zhang. Use of AI-driven code generation models in teaching and learning programming: a systematic literature review. In *Proc. of the 55th ACM Technical Symp. on Computer Science Education*, pages 172–178. ACM, 2024.
- [7] K. Charmaz. *Constructing grounded theory: A practical guide through qualitative analysis*. Sage, 2006.
- [8] A. Cockburn. Basic Use Case Template. *Humans and Technology, Technical Report*, 96:28, 1998.
- [9] E. Dickey, A. Bejarano, and C. Garg. AI-Lab: A Framework for Introducing Generative Artificial Intelligence Tools in Computer Programming Courses. *SN Computer Science*, 5(6):720, 2024.
- [10] A. S. Fernandez, D. Patrick, M. Gomez, and K. A. Cornell. Incorporating llm activities into established cs1 curriculum: An experience report. *Journal of Computing Sciences in Colleges*, 40(8):79–93, 2025.
- [11] E. Frankford, C. Sauerwein, P. Bassner, S. Krusche, and R. Breu. AI-Tutoring in Software Engineering Education. In *Proc. of the 46th Int'l Conf. on Softw. Eng.: SEET*, pages 309–319. ACM, 2024.
- [12] T. Greer, Q. Hao, M. Jing, and B. Barnes. On the effects of active learning environments in computing education. In *Proc. of the 50th ACM Technical Symposium on Computer Science Education*, page 267–272. ACM, 2019.
- [13] A. Guha, B. Zorn, C. J. Anderson, M. Q. Feldman, S. Gulwani, and G. Allen. The future of programming in the age of large language models. *Future*, 2025.
- [14] H. C. Herring and J. R. Williams. The role of objectives in curriculum development. *Journal of Accounting Education*, 18(1):1–14, 2000.
- [15] X. Hou, Y. Zhao, Y. Liu, Z. Yang, K. Wang, L. Li, X. Luo, D. Lo, J. Grundy, and H. Wang. Large Language Models for Software Engineering: A Systematic Literature Review. *ACM Trans. on Soft. Eng. and Methodology*, 33(8):1–79, 2024.
- [16] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günemann, E. Hüllermeier, et al. ChatGPT for good? On opportunities and challenges of large language models for education. *Learning and individual differences*, 103:102274, 2023.
- [17] V. D. Kirova, C. S. Ku, J. R. Laracy, and T. J. Marlowe. Software Engineering Education Must Adapt and Evolve for an LLM Environment. In *Proc. of the 55th ACM Technical Symposium on Computer Science Education*, pages 666–672. ACM, 2024.
- [18] A. N. Kumar and R. K. Raj. Computer science curricula 2023 (cs2023): The final report. In *Proc. of the 55th ACM Technical Symposium on Computer Science Education V. 2*, page 1867–1868. ACM, 2024.
- [19] Y. Li, J. Keung, and X. Ma. Integrating Generative AI in Software Engineering Education: Practical Strategies. In *Proc. of the 2024 International Symposium on Educational Technology*, pages 49–53. IEEE, 2024.
- [20] J. Luo, C. Zheng, J. Yin, and H. H. Teo. Design and assessment of ai-based learning tools in higher education: a systematic review. *International Journal of Educational Technology in Higher Education*, 22(1):42, 2025.
- [21] R. Malhotra, M. Massoudi, and R. Jindal. Shifting from traditional engineering education towards competency-based approach: The most recommended approach-review. *Education and Information Technologies*, 28(7):9081–9111, 2023.
- [22] C. K. Sah, L. Xiaoli, M. M. Islam, and M. K. Islam. Navigating the AI Frontier: A Critical Literature Review on Integrating Artificial Intelligence into Software Engineering Education. In *Proc. of the 36th Int'l Conf. on Softw. Eng. Education and Training*, pages 1–5. IEEE, 2024.
- [23] D. A. Schmidt, B. Alboloushi, A. Thomas, and R. Magalhaes. Integrating artificial intelligence in higher education: perceptions, challenges, and strategies for academic innovation. *Computers and Education Open*, 9:100274, 2025.
- [24] C. Sengul, R. Neykova, and G. Destefanis. Software engineering education in the era of conversational AI: current trends and future directions. *Frontiers in AI*, 7, 2024.
- [25] K. Sharifani and M. Amini. Machine learning and deep learning: A review of methods and applications. *World Information Technology and Engineering Journal*, 10(07):3897–3904, 2023.
- [26] B. Sheese, M. Liffiton, J. Savelka, and P. Denny. Patterns of student help-seeking when using a large language model-powered programming assistant. In *Proc. of the 26th Australasian Computing Education Conf.*, pages 49–57. ACM, 2024.
- [27] M. Skublewska-Paszkowska, M. Miłosz, and E. Lukasik. Acm/ieee recommendations for computing curricula and the needs of the polish cs industry. In *Proc. of the 9th Int'l Conf. on Education and New Learning Technologies*, pages 9050–9057. IATED, 2017.
- [28] C. W. Starr, B. Manaris, and R. H. Stalvey. Bloom's taxonomy revisited: specifying assessable learning objectives in computer science. *ACM Sigcse Bulletin*, 40(1):261–265, 2008.
- [29] M. Vierhauser, I. Groher, T. Antensteiner, and C. Sauerwein. Towards integrating emerging AI applications in SE education. In *Proc. 36th Int'l Conf. on Softw. Eng. Education and Training*, pages 1–5. IEEE, 2024.
- [30] M. Yabaku and S. Ouhbi. University Students' Perception and Expectations of Generative AI Tools for Software Engineering. In *Proc. of the 36th Int'l Conf. on Softw. Eng. Education and Training*, pages 1–5. IEEE, 2024.