VIBE CODING

SCENARIO

At a Ghanaian university, Professor Harry Barton Essel is teaching a final-year entrepreneurship seminar focused on graduate employability. Many of his students have strong ideas for solving local problems—transport coordination, student housing, digital retail, agribusiness logistics, and informal market payments—but only a few can code well enough to turn those ideas into working prototypes. To address this gap, Professor Essel introduces them to vibe coding: a workflow in which students describe an application in natural language and use AI tools to generate, refine, and test code. Rather than beginning with syntax, they begin with a problem, a user, and a value proposition. In class, student teams define a campus or community challenge, sketch a solution, and prompt an AI coding assistant to produce an initial version. Professor Essel does not present vibe coding as a shortcut to deep technical mastery. Instead, he frames it as an entrepreneurship enabler: a way for students to move more quickly from concept to prototype, gather feedback, and improve their employment prospects by demonstrating initiative, digital fluency, and innovation. Some students build simple minimum viable products; others discover that the first AI-generated version is flawed and requires testing, revision, and clearer specifications. By the end of the semester, students are not only discussing what they want to build but also showing working proof-of-concept applications to peers, faculty, and potential partners.

1 What is it?

Vibe coding is an AI-assisted approach to software development in which a person describes what an application should do, and an AI system generates and refines the code through conversational prompts. The term was coined in 2025 and is commonly associated with a workflow that shifts attention away from line-by-line coding and toward intent, iteration, and rapid prototyping. In educational settings, vibe coding can be understood as a bridge between idea formation and technical implementation. Students who are not yet expert programmers can still participate in software creation by articulating requirements, testing outputs, and refining prompts. This does not eliminate the importance of programming knowledge, but it changes the entry point: students begin with problem framing and design logic rather than syntax alone.

2 How does it work?

A typical vibe coding workflow begins with a clear description of a task, user need, or desired feature. The AI generates an initial codebase or component, after which the user runs it, evaluates the results, identifies errors or gaps, and issues follow-up prompts to improve functionality. In practice, the process is iterative: describe, generate, test, refine, and repeat. For Professor Essel's students, this means translating entrepreneurial ideas into structured prompts such as target users, core features, interface expectations, and success criteria. The strongest results usually come when students define the problem carefully, test outputs critically, and document revisions. In this sense, vibe coding is not simply "asking AI to build something"; it is a form of guided problem solving that depends on clarity, evaluation, and iteration.

3 Who's doing it?

Vibe coding is being used by professional developers, startup founders, students, and nontraditional creators who want to move quickly from concept to prototype. Major technology platforms and developer ecosystems are now openly discussing vibe coding as part of the broader transformation of software development enabled by AI-assisted tools. Its appeal in higher education is especially strong in disciplines where students need to demonstrate innovation but may not have extensive formal training in software engineering. Entrepreneurship, business, design, digital media, and interdisciplinary capstone courses are natural sites for adoption. In such contexts, vibe coding lowers the threshold for experimentation and may allow a wider range of students to participate in digital product development.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them. [Project number: 10112831

4 Why is it significant?

Vibe coding is significant because it reduces the time and expertise traditionally required to build a first working prototype. For entrepreneurship education, this matters: students can validate ideas earlier, test assumptions with users, and develop demonstrable artifacts that strengthen portfolios and employability. Instead of stopping at concept notes or slide decks, students can produce interactive solutions that communicate initiative and applied problem-solving ability. For universities in Ghana and similar contexts, the approach also has strategic relevance. It can support innovation ecosystems by enabling students to experiment with locally relevant tools and services even when advanced software development capacity is limited. Used thoughtfully, vibe coding can widen participation in digital creation and help students connect academic learning to enterprise development, freelancing, and self-employment opportunities.

5 What are the downsides?

Vibe coding can create the illusion that software development is easy, when in fact reliable systems still require sound architecture, testing, security checks, and human judgment. Current guidance from major technology organisations consistently notes that this approach is powerful for prototypes but less dependable for complex, high-stakes, or mission-critical systems when users rely on AI output without adequate review. In education, this means students may overestimate the quality of AI-generated code, misunderstand how their applications work, or struggle to debug hidden errors. There are also concerns about academic integrity, overreliance on automation, and the risk of producing applications that function superficially but fail under real-world conditions. Faculty, therefore, need to position Vibe coding as an aid to learning and innovation, not as a substitute for computational thinking, ethics, or technical accountability.

6 Where is it going?

Vibe coding is likely to become more structured, not less. Industry discussions already point toward workflows that combine conversational code generation with clearer specifications, validation steps, and human oversight. The direction of travel appears to be from informal experimentation toward more disciplined AI-assisted development. For higher education, this suggests that vibe coding may evolve into a recognised pedagogical practice within innovation, computing, and entrepreneurship programs. Universities may increasingly teach students how to write effective prompts, evaluate AI-generated code, test prototypes, and manage product development responsibly. In that future, employability may depend not only on whether graduates can code manually but also on whether they can direct AI tools intelligently and critically.

7 What are the implications for teaching and learning?

The central implication is that teaching may need to move beyond a narrow focus on syntax and include prompting, verification, prototyping, user-centred design, and ethical evaluation. Vibe coding can support active learning because students can build, inspect, revise, and present artefacts more quickly than in traditional development pipelines. When well integrated, it can strengthen entrepreneurial learning by linking opportunity recognition to prototype development and feedback. At the same time, the model changes what educators must assess. If AI can generate substantial portions of code, then academic value shifts toward problem definition, design reasoning, testing rigour, reflection, and responsible use. In Professor Essel's classroom, the most important learning outcome is not merely that students produce an app. It is that they learn how to convert ideas into evidence-based digital solutions, critique AI output, and use emerging tools to expand their employability and entrepreneurial capacity.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them. [Project number: 10112831