Project 2: Personalized Learning Application

Robert Foreman

Central Michigan University

EDU 814 - Dr. Raymond Francis

July 6, 2025

Prompt-to-Product: Building AI Tools Without Knowing How to Code

Introduction

As artificial intelligence continues to reshape the landscape of education, instructional designers and educators face a growing imperative: move beyond passive consumption of AI outputs and begin building tools that integrate AI into instructional workflows. While platforms like ChatGPT have made generative AI widely accessible, many Educational Technology professionals still use them only for surface-level tasks such as brainstorming or drafting content. Very few are building with AI, creating tools, manipulating APIs, or customizing outputs in meaningful ways. This gap is particularly evident when coding is involved; As RAND (2024) notes, many districts are prioritizing professional development that helps educators overcome fear or discomfort with AI technologies, rather than just teaching how to use the tools themselves.

This instructional module, *Prompt-to-Product: Building AI Tools Without Knowing How to Code*, addresses that challenge head-on. It is designed for graduate students in Educational Technology who may have no coding experience but are eager to understand and apply generative AI more effectively. By using large language models (LLMs) like ChatGPT not just for answers but for actual code generation, students will learn to build a working quiz generator that uses the OpenAI API. The emphasis is not on learning Python syntax, but on crafting structured prompts, interpreting responses, and collaborating with AI to refine their output. This shift (from syntax memorization to AI-enhanced design) empowers learners to cross the threshold from tech consumers to tech creators.

1. Identification of the Learner Group

The target learners for this instructional module are graduate students enrolled in Educational Technology programs. These students are typically motivated to explore emerging tools like generative AI but often have limited or no prior coding experience. Many are adult learners managing professional responsibilities, academic coursework, and instructional design roles in K–12 or higher education settings. Although they are familiar with digital platforms such as Canva, Google Classroom, or ChatGPT, they have rarely worked with code or APIs firsthand. Their comfort with digital tools does not necessarily translate into technical fluency.

This group represents a vital population within the field of education: professionals with strong instructional knowledge but little formal training in programming. Their interest in using AI to improve assessment, content creation, or learner feedback is steadily increasing. However, the idea of programming, even at a basic level, often presents a psychological and technical barrier to entry. The fear of failure or misunderstanding can prevent these learners from engaging with

tools they are fully capable of using. This module is designed to remove that fear by making code generation accessible through conversational interaction with AI.

The instructional approach reframes AI as a collaborator that can guide students through the coding process using natural language prompts. While prior Python experience is useful, it is not required to succeed. The module scaffolds technical tasks by teaching learners how to prompt ChatGPT to write, explain, and revise functional Python code. This structure enables students to build practical tools through dialogue rather than memorization. As a result, learners are empowered to move from passive AI users to active creators who can develop custom tools for instructional contexts.

2. SMART Learning Objectives

Clear, measurable objectives are essential to guiding learners through a structured instructional experience. This module follows the SMART framework, which emphasizes goals that are Specific, Measurable, Achievable, Relevant, and Time-bound (Doran, 1981). Each objective is crafted to support both the technical steps involved in building a tool and the reflective process of working alongside AI. The focus is on learning through creation, not just acquiring knowledge. These outcomes ensure that students finish the module with both a working product and a deeper understanding of prompt-based development.

At the end of this module, students will be able to:

a. Set up and configure a basic Python development environment using Visual Studio Code and the OpenAI Python package within the first session.
b. Use ChatGPT to generate Python code that sends a text input to the OpenAI API and returns multiple-choice and short-answer quiz questions.
c. Apply prompt engineering techniques to refine the quality, length, and format of AI-generated quiz questions.
d. Collaboratively debug Python scripts using LLM assistance, correcting functional or syntax errors through dialogue-based exploration.
e. Submit a final quiz generation tool and a reflection on the experience of building code through prompt-driven collaboration with AI.

3. Needs Analysis

Many educators express interest in learning how to integrate programming into instructional design but stop short due to fear of failure, lack of exposure, or technical complexity (Grover & Pea, 2013; Sentance & Waite, 2017). While Educational Technology programs often include coursework on digital tools, few provide instruction in coding. As a result, students may view programming as inaccessible or irrelevant to their instructional goals. Even basic scripting in

Python is perceived as a steep and intimidating learning curve. This gap leaves many professionals dependent on pre-built platforms rather than empowered to design their own.

Generative AI, particularly through tools like ChatGPT, creates a unique opportunity to challenge that assumption. These large language models can function as real-time scaffolds, translating natural language instructions into working code. This turns the programming process into a dialogue, where students develop tools through guided exploration. Rather than memorizing syntax, they learn by prompting, refining, and debugging with AI support. The shift transforms coding from an individual technical skill into a collaborative design experience.

This module is structured to meet the needs of learners who are interested in AI but hesitant to write code. First, it lowers the psychological barrier by offering a low-stakes, conversational entry into Python development. Second, it provides immediate feedback through ChatGPT, which allows students to diagnose and correct errors through natural language interaction. Finally, it builds transferable skills that students can carry forward into other projects, increasing their confidence and self-efficacy as instructional designers. These outcomes are foundational to developing long-term fluency with emerging technologies.

4. Module Framework

Module Title:
Prompt-to-Product: Building AI Tools Without Knowing How to Code

Delivery Format:
The module is designed for online or hybrid delivery and can be completed asynchronously, with optional synchronous support through office hours or virtual peer check-ins. Instruction is scaffolded across four sequential units, each building upon the last to support learning through guided exploration. Students engage with content through written walkthroughs, AI-generated code, and peer discussion. Each unit includes embedded opportunities for immediate practice, AI interaction, and reflection.

Unit 1: Rethinking “Knowing Code”

This unit introduces students to the fundamental tools and concepts they will use throughout the module. Learners install Python, Visual Studio Code, and the required OpenAI packages with support from a walkthrough and video demonstration. The emphasis is on seeing code as something students can generate, read, and modify with help from AI. ChatGPT is introduced as a “coding tutor” that can guide the student through each step of setup and experimentation.

Unit 2: Prompting as Programming

Students are introduced to prompt engineering as a foundational skill for tool development. This includes crafting prompts with role instructions, few-shot examples, and format requests to produce structured outputs. Learners work with ChatGPT to write a Python script that connects to the OpenAI API and generates multiple-choice and short-answer questions from a user-provided passage. AI responses are discussed, evaluated, and refined collaboratively.

This unit follows a bootcamp-style approach: brief lecture, live demo, and immediate application. Students are encouraged to test code, revise prompts, and use ChatGPT as a tutor and debugger in real time. To support this process, they receive a student-facing assignment sheet titled *Prompting and Programming*, which includes installation steps, starter prompts, and common error scenarios. The worksheet helps scaffold prompt construction and troubleshooting in a low-stakes, trial-and-error format and is submitted as part of the module's documentation.

A complete demonstration of this process is documented in a shared ChatGPT session, which also serves as a live example of the Prompt Design Journal:

🔗 ChatGPT Prompting Session – Module 2 Demo

Unit 3: Debugging with an LLM

In this unit, students begin testing and refining their quiz generator scripts. They intentionally or unintentionally encounter common issues such as authentication errors, JSON formatting bugs, or API misconfigurations. ChatGPT is positioned as the learner's primary debugger. Students copy their error messages into the prompt window and ask ChatGPT to explain the issue and provide fixes, then verify the results in their own IDE.

Unit 4: Product Showcase and Reflection

Students complete the module by finalizing their quiz generator, applying any optional enhancements such as styling, storage, or additional formatting. They record a brief screencast (2–3 minutes) demonstrating how the tool works and describing how they used AI to help build it. Finally, they reflect on the learning process and the role AI played in helping them succeed without formal coding instruction.

The final deliverables of this activity include:

• A Python script created through AI-supported prompting

• A formatted quiz generated by the script

• A screencast reflecting on their process, learning, and collaboration with AI

5. Technology Integration

The tools selected for this module were chosen for their accessibility, industry relevance, and support for learner autonomy. Each platform allows students to engage in coding activities without needing to memorize syntax or deeply understand system architecture. Together, these tools create a scaffolded learning environment that supports rapid experimentation, immediate feedback, and confidence-building. Their combined use helps demystify AI development while aligning with professional workflows. This integration also models real-world problem-solving with AI-enhanced tools.

a. OpenAI API

The OpenAI API is the engine behind the quiz generator tool and provides access to GPT-4 for educational use. Students use structured prompts to send requests to the API and receive AI-generated responses formatted as quiz questions. The API is accessed through Python using a simple authentication key, which students either receive through a free trial or supply themselves. This step introduces basic principles of API use and secure key management. The OpenAI API is publicly documented and widely used in EdTech and development settings (OpenAI, 2024).

b. ChatGPT

ChatGPT serves as a conversational coding assistant, tutor, and partner in problem-solving. Students rely on ChatGPT to generate new code, fix bugs, explain complex logic, and brainstorm alternate ways to structure their quiz tools. This is the core enabler that makes programming accessible to non-coders. Conversational AI tools like ChatGPT support learner engagement, coding fluency, and confidence through interactive problem-solving and on-demand feedback. In this module, ChatGPT bridges the gap between learner questions and real-time solutions.

c. Visual Studio Code (VSC)

VSC is a free, lightweight integrated development environment (IDE) used by developers around the world (Microsoft, 2024). It allows students to run and modify Python scripts while seeing outputs in real time. Unlike browser-based editors, VSC gives learners exposure to file systems, extensions, and terminal commands they may encounter in professional settings. It also helps make the learning experience feel authentic and transferable. By using VSC, learners develop habits and technical familiarity that carry over into future work.

6. Personalized Strategies

This module offers personalization by allowing students to work at their own pace, apply AI assistance when needed, and choose content relevant to their interests. Each student selects a passage or topic from a field they care about, such as education, history, or science, to use as the

basis for quiz generation. This content freedom increases engagement and makes the assignment more meaningful.

Rather than splitting learners into rigid beginner or advanced tracks, this course supports all students through prompt-based scaffolding. ChatGPT functions as a tutor, debugger, and co-creator, responding in real time to student questions and errors. Learners can simply ask for help using natural language, such as “Explain this line of code” or “Why did this error happen?” These interactions allow students to adjust to the difficulty based on their own needs.

The structure follows principles of Universal Design for Learning (UDL), supporting variable readiness, motivation, and strategy use (CAST, 2018). Whether students rely on step-by-step prompting or explore independently, all are guided toward building the same final product. The personalization lies not in different assignments, but in how each student works with the AI to complete a shared challenge.

7. Evaluation and Reflection

Assessment includes both formative and summative strategies to support student progress and measure mastery.

Grading Breakdown:

- Quiz Generator Script – 40%
- Screencast Walkthrough – 20%
- Prompt Design Journal – 20%
- Final Reflection – 20%

Sample Rubric Criteria:

- Code executes without errors and produces usable output
- Prompts are clearly structured and intentionally crafted
- Student demonstrates thoughtful and iterative use of AI for learning
- Reflection shows metacognitive insight and growth over time

Reflection Questions:

- How did the AI support you during the process of building your quiz generator?
- What challenges did you encounter, and how did you solve them?
- Would you use this workflow again in your instructional practice? Why or why not?

Note: The shared ChatGPT transcript from Module 2 serves as a live record of the student's prompting, error handling, and iterative development.

8. Presentation and Documentation

Final deliverables include instructional artifacts and presentation materials for peer and instructor review.

Required Components:

- This APA-formatted project paper
- Hand-drawn or digital concept map outlining instructional design and learner pathways
- Slide deck summarizing the module's structure, tools, and strategies
- 10–15 minute video presentation (module overview and screencast demonstration)

The concept map centers on the theme of "Learning to Code by Prompting AI" and visually organizes instructional phases, learner decision points, and feedback mechanisms. It also illustrates how AI scaffolding replaces traditional programming instruction, highlighting the shift from syntax-based teaching to prompt-based tool creation. The map is generated using Python and is included in this paper as **Exhibit A**.

Conclusion

This module redefines what it means to "learn to code" in an era shaped by artificial intelligence. Rather than asking learners to memorize syntax or understand complex software architecture, it teaches them to approach coding as a design process. By collaborating with large language models like ChatGPT, students learn to structure clear instructions, analyze output, and refine results through iterative feedback. This shift in mindset not only makes tool creation more accessible, but it also reframes AI as a partner in the learning process. Through prompt-based development, learners develop agency, creativity, and technical fluency without needing a traditional programming background.

The goal of this module is not only to help students build a working quiz generator, but to help them build confidence in learning innovative technologies. When students see that they can use natural language to create real tools, it transforms their identity from passive users to active creators. This transformation has implications far beyond a single project; it reshapes how educators think about their role in digital innovation. Generative AI is no longer a speculative future, it is a practical, present-day resource that instructional designers can harness today. This module equips them to do exactly that.

References

CAST. (2018). *Universal Design for Learning Guidelines version 2.2*. http://udlguidelines.cast.org

Doran, G. T. (1981). There's a S.M.A.R.T. way to write management's goals and objectives. *Management Review*, 70(11), 35–36.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher*, 42(1), 38–43.

Johnston, W. R., Diliberti, M., Grant, D., & Schwartz, H. L. (2024). Artificial intelligence in K–12: What district leaders need to know and do. RAND Corporation. https://www.rand.org/pubs/research_reports/RRA956-31.html

Microsoft. (2024). *Visual Studio Code*. https://code.visualstudio.com/

OpenAI. (2024). *OpenAI API documentation*. https://platform.openai.com/docs/

Sentance, S., Waite, J., & Kallia, M. (2019). Teaching computer programming with PRIMM: A sociocultural perspective. *Computer Science Education, 29*(2–3), 136–176. https://doi.org/10.1080/08993408.2019.1608781

Exhibit A: Concept Map