Issues in Information Systems
Volume 26, Issue 3, pp. 437-444, 2025

DOI: https://doi.org/10.48009/3_iis_2025_2025_135

A snapshot at the future of teaching computer programming courses from the lenses of artificial intelligence – A comparison with teaching web design courses

Azad Ali, *University of Fairfax, aali@ufairfax.edu*
Frederick Kohun, *Robert Morris University, kohun@rmu.edu*

Abstract

The introduction of Artificial Intelligence (AI) and its tools into education has upended many academic programs and led to numerous changes in curricula. It also led to questioning the viability and the content included in the teaching of some courses. Among the courses that face that question are computer programming courses. The AI tools can generate programming codes based on written instructions in human spoken languages, simply and without many complications. With this simplicity of producing programming code, the question arises about the feasibility of continuing to teach programming courses with the same methods as before, knowing the simplicity with which AI can produce programming code. Some are predicting that teaching programming the traditional methods will be obsolete and that technological advances dictate including AI tools in the teaching of computer programming courses.This comparative conceptual paper discusses the above concerns by comparing the changes that face the teaching of programming courses with similar significant changes that took place in teaching web design courses at the start of this century. At that time, similar calls were made to replace the teaching of web design courses. An explanation of both cases of teaching (programming and web design) are provided to draw parallels between the two. The purpose is to provide a snapshot of what teaching programming may look like in the future with respect to using AI tools

Keywords: teaching programming, artificial intelligence, artificial intelligence and programming

Introduction

The rapid development of Artificial Intelligence (AI) tools and models has led many academic programs to rethink their course offerings and the pedagogy they follow in teaching their courses. Among the courses that face this scrutiny are computer programming courses and similar calls to update the methods of teaching programming courses (Ali et al., 2023). Generative AI (GAI) models are capable of producing programming code to perform different tasks that are typically taught in programming courses by following rigid syntax and unfamiliar logic (Lytvynova et al., 2024). With the explosion of this AI technology, there is a need to question the current teaching of traditional logic and syntax, that generates the same code as does the AI tools. These AI tools can produce the code by simply writing a few descriptions into GAI about what they want from the program (Joshi, 2025). A similar debate took place in the early 2000's with the introduction of teaching web design courses. At first, web design courses were taught by teaching the coding of HTML tags that display web pages on web browsers. That was soon replaced with the introduction of web design editors. These are termed WYSIWYG (what you see is what you get) editors. With WYSIWYG editors, students can design web pages with simple drag-and-drop objects and then upload them to the browser (Mishra et al., 2021). A parallel story may exist between using WYSIWYG to design web pages and using GAI to write computer programs. The paper addresses the following research questions below which are sequentially addressed in the paper methodologically, through a comparative conceptual case study to explore paralells with respect to AI. :

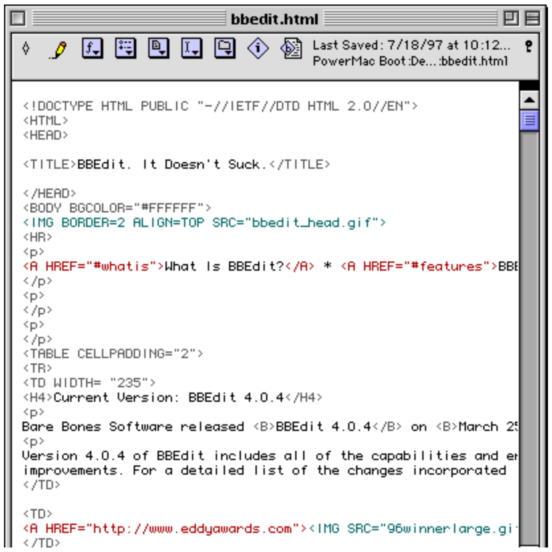
- *How were web design courses taught and how did they change with the introduction of WYSIWYG tools?*
- *How are programming courses taught and how introducing AI sparked the debate?*
- *Are the experiences of both cases (teaching web design and teaching programming) similar?*
- *What will teaching computer programming looks like in light of the increasing use of AI tools.*

The Teaching of Web Page Development

The increasing use of the internet took place at the turn of this century, highlighting the importance of establishing a presence on the web. The steps for establishing this presence on the web can begin by creating or writing web pages. That is followed by deploying the web pages to the web so it can be viewed through web browsers (Jacksi & Abass, 2019). The creation of web pages has to start with writing code in HTML (Hypertext Markup Language). HTML (Hyper Text Markup Language), although is not considered a programming language, certain syntax has to be followed in order to correctly deploy the web page. Developing the web pages used to be done by entering the HTML code into a simple editor and following the HTML syntax.

Developing Web Pages with Coding

Web design courses started by requiring students to use HTML tags to organize the content they want to display on the web page. HTML is a language, although it is not officially considered a programming language, but it has content that needs to follow syntax for it to correctly display the content. This syntax typically follows certain rules, although minimal, but still, these rules are confusing for some. These rules could include how to start and close the tag, and it also continues that need to be additional attributes for each tag to control the appearance of the content within the tag (Peers, 2017). Figure 1 below shows an example of HTM code that displays a web page.


bbedit.html
Last Saved: 7/18/97 at 10:12...
PowerMac Boot:De...:bbedit.html

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<HTML>
<HEAD>

<TITLE>BBEdit. It Doesn't Suck.</TITLE>

</HEAD>
<BODY BGCOLOR="#FFFFFF">
<IMG BORDER=2 ALIGN=TOP SRC="bbedit_head.gif">
<HR>
<p>
<A HREF="#whatis">What Is BBEdit?</A> * <A HREF="#features">BBE
</p>
<p>
</p>
<p>
</p>
<TABLE CELLPADDING="2">
<TR>
<TD WIDTH= "235">
<H4>Current Version: BBEdit 4.0.4</H4>
<p>
Bare Bones Software released <B>BBEdit 4.0.4</B> on <B>March 25
<p>
Version 4.0.4 of BBEdit includes all of the capabilities and en
improvements. For a detailed list of the changes incorporated
</TD>

<TD>
<A HREF="http://www.eddyawards.com"><IMG SRC="96winnerlarge.gi
</TD>
```

Figure 1. Example of HTML Code. Adapted from (Hoffman, 2017)

The syntax of HTML, although minimal, but it is still a challenge to some. Added to the complexity is that during the early days of writing web pages, there were no specialized editors that pointed users to the errors that needed to be corrected before displaying the web page in a web browser (Ali & Wibowo, 2023). In other words, there is no life view of the page as the user enters the code. Instead, users have to save the code, display it separately in a browser, and if there are errors, come back to Notepad and make corrections. That cycle can continue until the desired page is completed. HTML was not the only thing that needed to be learned for web design at that time. Students taking web design courses had to learn about Cascading Style Sheets (or CSS) that control the appearance of the content of the page. This appearance could include color, font, text size, positioning of elements, background color/image, adjustment to screen size, and many other attributes. Using CSS requires learning the syntax of CSS as well (Flores, 2025).

Learning HTML and CSS enables the student to display static web pages. That is displaying pages that have no interactivity and no change in content. To learn to add interactivity or to make the webpage dynamic, students had to learn programming languages such as JavaScript (Dean, 2025). JavaScript is a programming language that uses rigid syntax, and it will not be easy to learn by non-technology majors (Kapoor, 2025). Figure 2 below shows the contrast between the three: html, CSS and JavaScript.

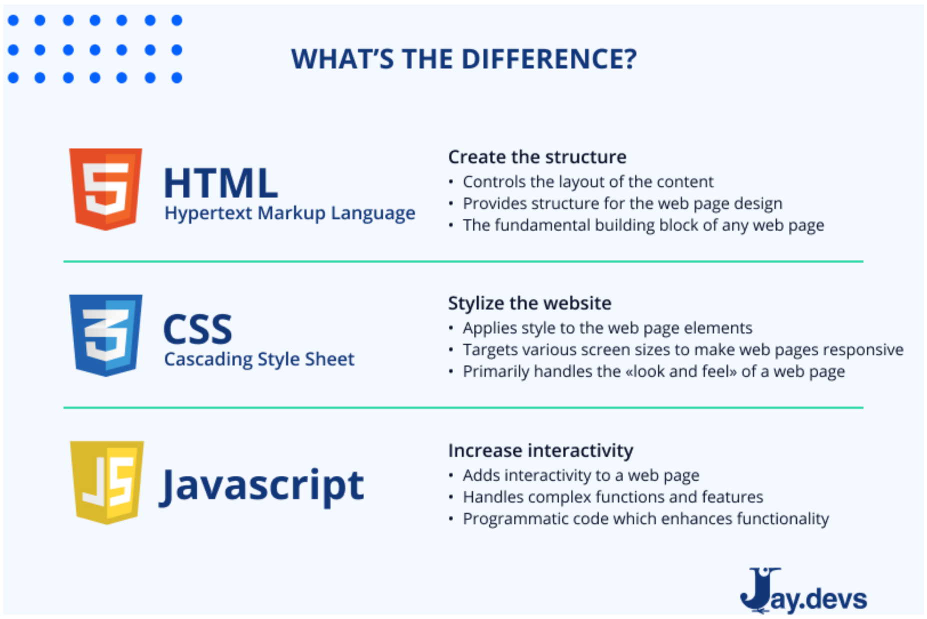

WHAT'S THE DIFFERENCE?
HTML
Hypertext Markup Language
Create the structure
• Controls the layout of the content
• Provides structure for the web page design
• The fundamental building block of any web page
CSS
Cascading Style Sheet
Stylize the website
• Applies style to the web page elements
• Targets various screen sizes to make web pages responsive
• Primarily handles the «look and feel» of a web page
Javascript
Increase interactivity
• Adds interactivity to a web page
• Handles complex functions and features
• Programmatic code which enhances functionality
Jay.devs

Figure 2. HTML, CSS & JavaScript Contrast – Adapted from (Valadzko & Talochka, 2023)

Learning about these technologies presented a challenge to students who were not Information Technology majors. This limited the development of web pages and the establishment and maintenance of web pages. Given the potential value that can be gained from having a presence on the web, it is necessary to look for ways that make it easier to develop web pages for those who find it a challenge to code with HTML, CSS, and JavaScript. This led to the development of WYSIWYG (What You See is What You Get) software.

Developing Web Pages with WYSIWYG Editors

The start of using WYSIWYG editors dates to the beginning of this century with the introduction of the Microsoft FrontPage web design tool (Crumlish & Chase, 2006). FrontPage represented the beginning of it, and that was followed by the development of many other tools that we see nowadays. We see these tools everywhere, including in MS Word and Excel. They all follow the same concept. That is, the web page can be developed with simple drag and drop objects (Maudet, 2025): Pointing with the mouse at an object and dragging objects from the toolbars and dropping them in the drawing areas of the page (similar to shaping it on a drawing canvas). After dropping in the canvas, modifications and adjustments can be made to give the object the desired look and to align it with other content on the page. Within the drag and drop process and other steps followed, in the background, FrontPage generates the HTML code, it places the CSS attributes, and changes the JavaScript code as the page is being developed. This goes on behind the scenes, and no need for the user to learn about them.

The introduction of WYSIWYG editors was a simple breath of fresh air for many. It led to the increasing use of web development and the creation of more web pages (Mahmud et al., 2024). FrontPage was specifically built for amateurs and for those who do not want to bother with syntax, tags, styles, and many others. It resembled a document editor in word processing more than looking like a design tool (So, common sense can tell if a person can use Word processing software (and many people can), so they can use MS Frontpage to design web pages. Nevertheless, as the use of FrontPage increased, so did the problems that appeared from using it (Hyndman & Hyndman, 2005). One of the issues that faced using FrontPage was the inefficient production of tags, which led to errors. These errors can be corrected by fixing the underlying HTML tags. FrontPage included a view of the page and a view of the code. So experienced HTML users can edit and fix the code. However, for students who do not know HTML tags, it will be difficult to make these adjustments. Then came Macromedia DreamWeaver, which was better in its design and easier to use, but still had errors in its generated HTML code. It became easier to deal with images and media content because it came with the same suite. Dreamweaver offers both visual editors and a code editor, making it suitable for non-coders and for others who do not know coding and can use the design tools (Pizzi & Ruvalcaba, 2003).

What Was Learned from Web Design Courses

Lessons learned from teaching web design:

- Although the WYSIWYG makes designing web pages easier, in-depth knowledge is required for those who learn to code web pages.
- So many newer WYSIWG editors have been developed that are hard to count. Some are web-based while others require local installations.
- Technological development introduced another form of web development which is backend development. It emphasizes coding more than the use of HTML tags and JavaScript.

The Teaching of Computer Programming

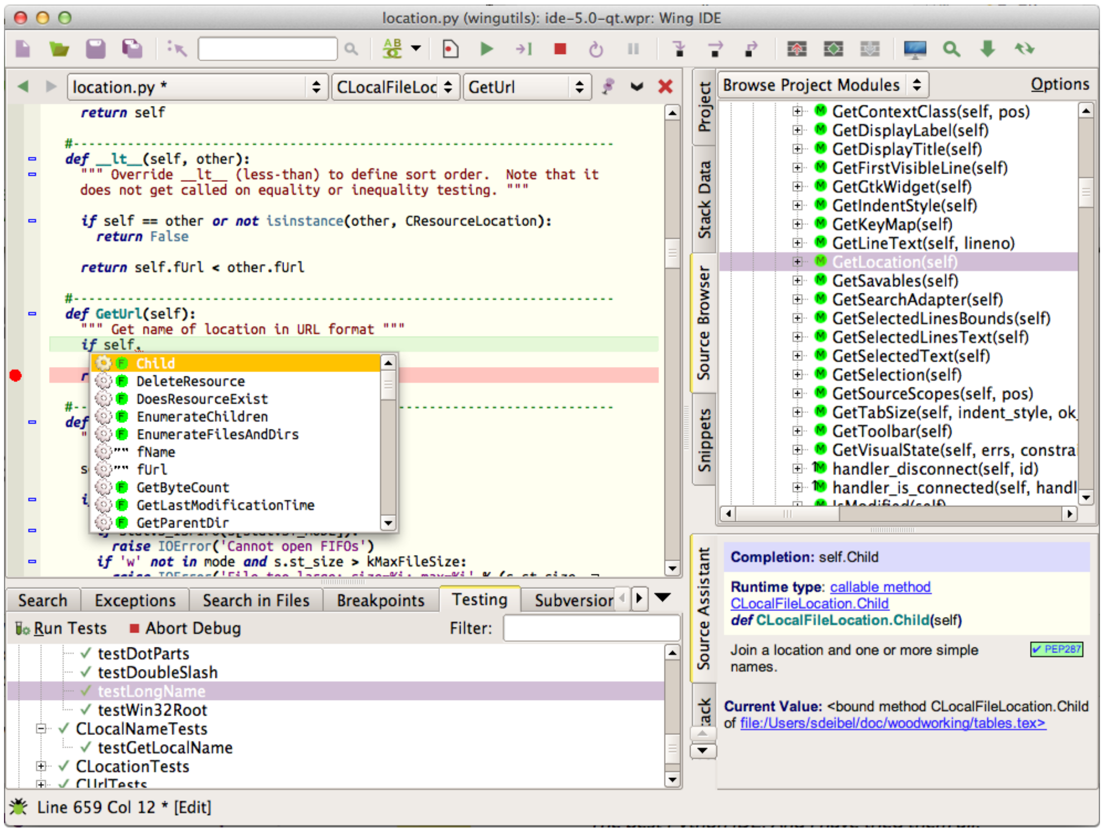
This section explains the teaching of computer programming, how it is done, and the challenges faced in teaching/learning programming. The writing of programs is first explained through learning the program syntax and structure. Then the development of program code through the use of Generative AI is presented.

Writing Programs with Code Editors

Writing programs with any programming language requires learning a few things to complete the required tasks. Initially, once a solution to solving a problem is developed, it then can be coded (programmed) in one of the many numerous programming languages. To successfully write computer programs, there needs understanding of the following points:

- *An appropriate programming language must be selected; It can be speculated that there are thousands of programming languages (Ali et al., 2023).*
- *Each programming language has its syntax.*
- *To write a program, there needs to be an editor that helps with coding the program*

Figure 3 below shows a screen capture of using the editor to write a program in the Python programming language.



```
    return self

  #-----------------------------------------------------------------
  def __lt__(self, other):
    """ Override __lt__ (less-than) to define sort order.  Note that it
    does not get called on equality or inequality testing. """

    if self == other or not isinstance(other, CResourceLocation):
      return False

    return self.fUrl < other.fUrl

  #-----------------------------------------------------------------
  def GetUrl(self):
    """ Get name of location in URL format """
    if self.
```

```
Child
DeleteResource
DoesResourceExist
EnumerateChildren
EnumerateFilesAndDirs
fName
fUrl
GetByteCount
GetLastModificationTime
GetParentDir
```

```
      raise IOError('Cannot open FIFOs')
    if 'w' not in mode and s.st_size > kMaxFileSize:
```

```
GetContextClass(self, pos)
GetDisplayLabel(self)
GetDisplayTitle(self)
GetFirstVisibleLine(self)
GetGtkWidget(self)
GetIndentStyle(self)
GetKeyMap(self)
GetLineText(self, lineno)
GetLocation(self)
GetSavables(self)
GetSearchAdapter(self)
GetSelectedLinesBounds(self)
GetSelectedLinesText(self)
GetSelectedText(self)
GetSelection(self)
GetSourceScopes(self, pos)
GetTabSize(self, indent_style, ok
GetToolbar(self)
GetVisualState(self, errs, constra
handler_disconnect(self, id)
handler_is_connected(self, handl
```

```
✓ testDotParts
✓ testDoubleSlash
✓ testLongName
✓ testWin32Root
✓ CLocalNameTests
  ✓ testGetLocalName
✓ CLocationTests
```

Completion: self.Child

Runtime type: callable method CLocalFileLocation.Child
def CLocalFileLocation.Child(self)

Join a location and one or more simple names. ✓ PEP287

Current Value: <bound method CLocalFileLocation.Child of file:/Users/sdeibel/doc/woodworking/tables.tex>

Line 659 Col 12 * [Edit]

Figure 3. Editing Python Programming. Adapted from (Wing Python IDE, 2015.

The syntax of a programming language is the rules and grammar of the language. These rules must be followed to make a statement that is executed in the programming language. Not following the rules gives syntax errors, which must be corrected to continue the work of the program. The structure of the program is the way the program is organized. Programmers often use the term "Modules" to describe the sections that the program is divided into. For programs and in order to avoid long programs, it is often necessary to divide them into multiple smaller modules. Understanding the structure means understanding how these modules are connected or linked together. Algorithms are the steps that need to be followed to programmatically solve a problem or complete an assigned task. It is known that computer programs are written to solve a problem(s) or complete a task(s). The steps that need to be followed in order to use the programming language to complete the task are called following the algorithm.

Learning to program has proven challenging to many students. Many studies attribute the first course in programming as the reason for the attrition of students who enroll in Information Technology programs. This relates mainly to two reasons: rigid syntax and unfamiliar structure, and then, as a result, following the algorithm. There have been calls for a long time to find a way to develop programming codes that do not require learning the three factors: syntax, structure, and algorithm. These calls are finally answered through the development of Generative Artificial Intelligence (GAI) tools that are capable of coding programs without dealing with Syntax, Structure, and Algorithm of programs.

Developing Program Code with GAI
Students can request to write a program with GAI. This can be done anytime and without requirements to understand coding. However, to be precise and to get the correct responses from GAI, students may need to learn how to use Prompt Engineering. A prompt is the text that individuals use to enter a dialogue with AI. A prompt can be a question to ask, or in the case of programming, it can specify the output to be generated, based on the constraints. This can specify the steps to follow (Algorithm), the section to code (Structure of the program), and the language to write it in (to follow the syntax of the language). The user enters the description of the problem for which to generate the code, presses Enter, and the GAI goes to work and generates the code. There is no need to follow a syntax, nor is there a need to learn the structure and plan the structure of the program. However, not every command to AI produces the correct code; instead, some rules need to be followed for a better chance of correctly producing the desired output (Nambisan, 2024).

The popular term (GIGO) or garbage in and garbage out applies here. If AI receives incomplete or incomprehensible questions or information, then the output from AI could be the same, incomplete, or incomprehensible. Yet if the questions are entered clearly and follow correct protocols with AI, chances are better that the answers could be consistent with the questions asked. It is important to stress that questions need to be crafted according to the code asked for. Additionally, it should be emphasized that prompt engineering could include more than one question and answer. If the answer that is received is not satisfactory, the user can follow up with another question. AI can remember the information from the earlier prompt and build on it. Users can enter more follow-up questions and can enter related data or information. This is termed an "iterative prompt" (Wang et al., 2022). A prompt starts by first initiating a question. The initial results generated from AI can be analyzed to check if the generated code answers the question and the requirements. If the generated code is not sufficient, the question can be refined and supported with more data to make it understandable and then asked again. This kind of back and forth could go multiple times until the user gets the responses being looked for, or they could look another way.

Limitations

This study has numerous limitations including but not limited to:

- Assumes widespread adoption of AI without addressing social, economic, and pedagogical barriers.
- While this study was limited to the teaching of programming courses, there was no consideration of subject matter, complexity, and discipline specific needs or constraints.

Conclusions

Based on the limitations and scope of the previous discussion including the learning about the teaching of web design courses, we estimate that the following may characterize the future of the teaching of computer programming courses:

- GAI is going to stay here and is going to reshape the teaching of many courses.
- Similar to the proliferation of WYSIWYG tools, there will be an increasing number of tools that generate program code without knowledge of programming.
- Learning GAI and prompt engineering is becoming a "clear must" for different IT majors.
- There will be continuous demand for programming skills and knowledge, so the teaching of these courses will continue.

- IT majors will be better equipped to handle the details and issues of programming if taught programming languages the traditional way – teaching syntax, structure, and algorithm.
- There are widespread calls to include the teaching of GAI along with programming in one of the following formats.
    - Teach programming first, teaching AI tools second, then use the knowledge gained from traditional programming to critique the code generated from AI.
    - Teach GAI first and then teach programming second. This may not encourage the student to learn the details of programming, but students can have knowledge of what programming does and then get involved in the details.
    - Critiquing the code generated from GAI.
    - Code maintenance can be done by submitting the code to GAI and asking it to explain the code.

It is expected that AI will reshape the roles of programming but will not replace the job of programming. There will be a need for programmers as the AI changes revolve into more use and more technology development. The above are general ideas about the directions of teaching programming, taking into consideration the continuous development of GAI. This technology is moving so fast that many other studies will emerge to debate the content and the teaching of courses.

References

Ali, A., Chaudhary, P., & Wibowo, K. (2023). Considerations for updating programming courses. *Issues in Information Systems, 24*(2). https://doi.org/10.48009/2_iis_2023_112

Ali, A., & Wibowo, K. (2023). Assessment of ChatGPT-generated programming code based on exercises in an introductory programming course. *Issues in Information Systems, 24*(2). https://doi.org/10.48009/2_iis_2023_117

Crumlish, C., & Chase, K. J. (2006). *Microsoft FrontPage 2003: Savvy*. John Wiley & Sons.

Dean, J. (2025). *Web Programming with HTML, CSS, and JavaScript*. Jones & Bartlett Learning.

Flores, L. (2025). *What Does CSS stand for?* https://www.cssacademy.com/blog/what-does-css-stand-for

Hoffman, J. (2017, June 5). *The History of the Web. A Moment in Time With Editors*. https://thehistoryoftheweb.com/moment-time-editors/

Hyndman, S. M., & Hyndman, J. (2005). Creating an Eportfolio with MS FrontPage: It Doesn't Get Any Easier!. *Essays in Education*, *14*(1), 9.

Jacksi, K., & Abass, S. M. (2019). Development history of the world wide web. *Int. J. Sci. Technol. Res*, *8*(9), 75-79.

Joshi, S. (2025). Introduction to Generative AI: Its Impact on Jobs, Education, Work and Policy Making.

Kapoor, S. (2025). Control Flow in JavaScript. In *Beginning JavaScript Syntax* (pp. 209-260). Apress, Berkeley, CA.

Lytvynova, S. H., Rashevska, N. V., & Proskura, S. L. (2024). The use of artificial intelligence in teaching students programming languages. *In Proceedings of the IX International Workshop on Professional Retraining and Life-Long Learning using ICT: Person-oriented Approach (3L-Person 2024) co-located with 19th International Conference on ICT in Education, Research, and Industrial Applications* (No. 3781, pp. 10-29).

Maudet, N. (2025). Design Templates: Between Empowerment and Control of Amateur Graphic Designers. *Design Issues*, *41*(1), 38-50.

Mishra, D. P., Rout, K. K., & Salkuti, S. R. (2021). Modern tools and current trends in web-development. *Indonesian Journal of Electrical Engineering and Computer Science*, *24*(2), 978-985.

Mahmud, M. F., Yee, M. H., Mohamad, M. M., Azid, N., & Putra, A. B. N. R. (2024). E-Learning Web Elements for Brickwork Course in the Construction Technology for Vocational Training. *Journal of Technical Education and Training, 16*(2), 153-164.

Nambisan, R. (2024). *Types of Prompts*. https://www.linkedin.com/pulse/types-prompts-ranjit-nambisan-2engf/

Peers, N. (2017). *WYSIWYG Web Builder 12 makes building websites easier than ever with new Blocks feature.* https://betanews.com/2017/04/04/wysiwyg-web-builder-12/

Pizzi, M., & Ruvalcaba, Z. (2003). *Macromedia Dreamweaver MX Unleashed*. Sams Publishing.

Singh, R., Kim, J. Y., Glassy, E. F., Dash, R. C., Brodsky, V., Seheult, J., ... & Pritt, B. S. (2025). Introduction to Generative Artificial Intelligence: Contextualizing the Future. *Archives of pathology & laboratory medicine*, 149(2), 112-122.

Phillips, T. (2025). AI Code Tools: *The Ultimate Guide in 2025*. https://codesubmit.io/blog/ai-code-tools/

Valadzko, A., & Talochka, A. (2023). What is JavaScript Used for: 11 App Ideas to Consider Using JavaScript. https://jaydevs.com/what-is-javascript-used-for/

Wang, B., Deng, X., & Sun, H. (2022). Iteratively prompt pre-trained language models for chain of thought. *arXiv preprint arXiv:2203.08383*.

Wing Python IDE. (2015). *The Intelligent Development Environment for Python.* https://wingware.com/news/2015-01-22