PD12

A Skeptic's Guide to Using AI for Programming

Frank Menius, Atorus Research, USA

ABSTRACT

Like it or not, "AI" is everywhere – on your phone, in your fridge, and even government systems. But is it a buzzword, a costly gimmick, or a genuine productivity tool? The truth is: it can be all of the above. Chances are your organization is already using AI or is planning to soon. So let this skeptic walk you through safely using AI for programming without compromising your skills, productivity, clinical data integrity, or sanity. This presentation will take a pragmatic look at the current AI capabilities with an emphasis on limitations, pitfalls, and practical applications in creating clinical outputs. Let's face it, we don't have to be enthusiastic about AI, but we do need to understand how to work with it.

DISCLAIMER

The opinions shared in this paper are the opinions of the author and do not necessarily represent the opinions of Atorus Research or PHUSE. Please see the reference section for supporting documentation.

SKEPTICISM

I am an AI skeptic. I look around and see how AI is currently being inserted into nearly every aspect of our lives, and I don't see that it is doing much good. It doesn't make me happy. I'm sure that there are some things that AI could be used for that would make humanity better, but currently I don't see it being used for anything that outweighs its destructive side. Now when I say destructive side do I mean that AI is going to reach the singularity and then destroy us all like some sci-fi movie? No, at least not yet. I mean that it is destructive because it is exacerbating already existing human problems and not solving them.

Environmentally, AI is sucking down our finite clean water supplies. AI power consumption makes up some 40% of the power usage in some areas. This not only is increasing the price we pay for our electricity but also intensitying our reliance on dirty sources of energy like coal, oil, and natural gas. In short, AI is accelerating the crisis of global climate change.

Socially, AI is being used to mislead populations with deep-fake videos, altered photos, and the malicious spreading of misinformation. It is the propagandist of the dictator, and the cudgel of the bully. It is being used to undress and humiliate outspoken women, taunt political opponents, and even create and spread abusive photos of children. AI in its current state is a threat to societies and democracies worldwide.

Financially, AI is increasing your prices. I already mentioned the increase in electricity bills, but it is also affecting water. If you have been computer shopping recently you have also noticed a sharp increase in the price of RAM, GPUs, and solid-state drives. Just about anything that uses rare earth minerals is being affected by the increase of AI data centers. On top of prices, there is growing evidence that we are in the middle of an AI investment bubble similar to the dot-com bubble of the early 1990s. 95% of companies who have invested in AI have seen 0 return on their investments.

But what about you? What about the programmer? Well, you can get onto any blog or discussion board and read a lot of anecdotal evidence that says programmers are less productive, slower, understand their code less, or even forget how to do simple tasks when AI is involved. There are senior coders who say junior developers don't understand the code they write and can't take into account edge case scenarios. Others say that they can't fix errors in the code because they don't understand what the AI is doing. Of course, that is all anecdotal but en masse it seems to make a statement. At least one randomized study supports this evidence though: the METR study *Measuring the impact of early-2025 AI on experienced open-source developer productivity*. This study shows that when AI was allowed for code development, development took 19% longer to produce code than when AI was not allowed. The developers themselves even said they thought they were 20% faster when using the AI but the measured metrics did not match their perceived observations.

So, where does that leave us? Sadley the genie is out of the bottle, Pandora's box has been opened, and we are stuck with AI. Now, I have a lot of ideas for how we could use AI to actually solve some of humanity's problems, but that is not the paper you came here to read. You came here because for better or for worse, your employer is forcing

you to use AI in one form or another or will be soon. You need to know how to use AI safely and efficiently. I have a full three-hour course I give on this topic through Atorus Academy that is a bit too much for this presentation or even this paper so, I am going to condense down some of the most important things into some tips and skills you can take and use today.

SAFETY FIRST

How can you safely use AI in clinical programming? We need to first take a step back and talk briefly about how AI works. In very basic terms AI models gather information from the internet and training data and use that information to recognize patterns. Artificial Intelligence is kind of a misnomer at this point. AI isn't really intelligent, it can't think. It is good at pattern recognition. It is good at syntax. It is not good at making assumptions, inferences, or being creative in the way a human can. It will respond to a prompt you give, recognize patterns in your prompt, compile that into something it understands and then go out to its available data and put together a response. That response will be based on patterns it found in the data. So, if you ask it to draw a picture it will take pieces of all the other pictures that are similar to what you asked for and compile them into a "new" picture. For programming, the data that the AI will be referencing is likely open-source code on the internet, any internal code libraries you give it access to, and any code you put in your prompt.

With that said there are some things to remember. First, because AI is using all this open-source code from all over the internet, some of that source code may be outdated, or have security vulnerabilities. It may be injecting deprecated or vulnerable code into your programming. AI training data also likely does not have access to the most up-to-date security patches and information. This may not be of the utmost importance when we are talking about programming datasets and tables, listings, and figures (TLFs), but it is still something to keep in mind.

The second point I want to emphasis is that as more and more companies introduce Large Language Models (LLMs) and AI into their organizational workflows; many have given the AI access to their internal data. This can be problematic because it can inadvertently lead to a leak of sensitive data like API keys, credential codes, proprietary algorithms, or even patient data. This potential leak is especially true if the AI returns the organization's data to the AI providers hosted servers for processing. Also, AIs don't forget, once it is on the internet it is always on the internet. It is going to be important for all stakeholders at your organization to perform a risk-based analysis of any system you are using, especially when patient data is involved. This brings me to my first of 4 major tips**: Do not put your proprietary code or data into a non-secure AI chatbot!**

PROGRAMMING AI IS GOOD AT

Now that the safety aspect is out of the way let's talk about some of the things that AI can be particularly helpful with for programming. There are a lot of things you could use AI for, but I have narrowed it down to a few that I think will be the most beneficial to programmers and not greatly hamper them.

A LEARNING TOOL

AI can be an excellent tool for learning. Maybe you are learning a new coding language or a new technique. AI could be very helpful in getting you started on your journey. Let's say you are a SAS programmer, and you want to learn how to produce some SAS outputs in R. You can put some existing SAS code into a prompt and ask the AI to write this code in R instead. Or you can simply ask R to write some code that does X. You might be pretty surprised at how well the AI can perform these tasks, but it is not perfect so let's look at an example.

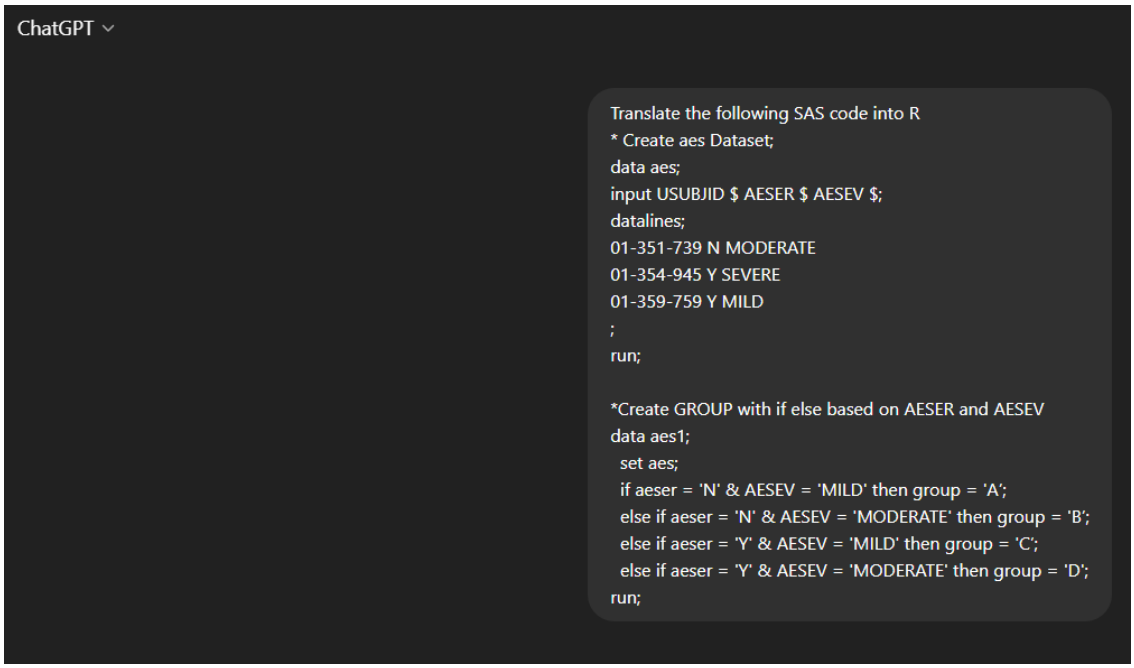
Below is some very basic SAS code with its associated output.

```
* Create aes Dataset;
data aes;
input USUBJID $ AESER $ AESEV $;
datalines;
01-351-739 N MODERATE
01-354-945 Y SEVERE
01-359-759 Y MILD
;
run;

*Create GROUP with if else based on AESER and AESEV
data aes1;
  set aes;
  if aeser = 'N' & AESEV = 'MILD' then group = 'A';
  else if aeser = 'N' & AESEV = 'MODERATE' then group = 'B';
  else if aeser = 'Y' & AESEV = 'MILD' then group = 'C';
  else if aeser = 'Y' & AESEV = 'MODERATE' then group = 'D';
run;
```

USUBJID	AESER	AESEV	GROUP
01-351-739	N	MODERATE	B
01-354-945	Y	SEVERE	
01-359-759	Y	MILD	C

I want to turn this SAS code into R code, So I hopped onto ChatGPT and asked it to translate it. I do want to note that this is not real patient data, and it is open-source code so I am not worried about sharing it. The prompt I used is below.



After a few seconds the following R script and notes were returned to me.

The return looks pretty reasonable, so I put the script into R-Studio to take a look at the output.

```
A tibble: 3 × 4
USUBJID    AESER AESEV    group
<chr>      <chr> <chr>    <chr>
01-351-739 N     MODERATE B
01-354-945 Y     SEVERE   NA
01-359-759 Y     MILD     C
```

It looks pretty good. There are some differences, we have the tibble output which shows us a little metadata. Since SAS and R handle missing values a little bit differently, that accounts for the NA that is present in the middle row. The one problem I see is that the group header is in lower-case instead of upper-case, so I am going to want to fix that. Interacting with an AI is like a conversation, it requires feedback. So, I am going to tell the AI that this code is good but I need the group column to be upper-case. Here is the resulting return.

Sure! To ensure the `group` column is uppercase (even though your labels are already uppercase), you can wrap the `case_when()` result with `str_to_upper()` from the **stringr** package (also part of the tidyverse):

r Copy Edit

```r
library(tidyverse)

# Create aes dataset
aes <- tribble(
  ~USUBJID,     ~AESER, ~AESEV,
  "01-351-739", "N",    "MODERATE",
  "01-354-945", "Y",    "SEVERE",
  "01-359-759", "Y",    "MILD"
)

# Add 'group' column and ensure it's uppercase
aes1 <- aes %>%
  mutate(group = case_when(
    AESER == "N" & AESEV == "MILD"     ~ "A",
    AESER == "N" & AESEV == "MODERATE" ~ "B",
    AESER == "Y" & AESEV == "MILD"     ~ "C",
    AESER == "Y" & AESEV == "MODERATE" ~ "D",
    TRUE ~ NA_character_
  )) %>%
  mutate(group = str_to_upper(group))

# View result
aes1
```

This will guarantee that all values in `group` are uppercase, even if the logic changes later. Let me know if you want to turn unmatched rows (like `"SEVERE"`) into a specific group!

Here is the output from the script.

```
A tibble: 3 × 4
USUBJID    AESER AESEV    group
<chr>      <chr> <chr>    <chr>
01-351-739 N     MODERATE B
01-354-945 Y     SEVERE   NA
01-359-759 Y     MILD     C
```

That can't be right, it looks exactly the same. Maybe you already know why. I asked the AI to make the group column upper-case and assumed it would make the column header upper-case. Instead, it did exactly what I asked it to and made the column values upper-case, even though they already were. So, I need to give it more feedback and this time I need to be really specific. I will tell it to make the group column header upper-case.

Got it! To rename the `group` column to `GROUP` while keeping everything in the **tidyverse** style, you can use `rename()` from `dplyr`:

r Copy Edit

```r
library(tidyverse)

# Create aes dataset
aes <- tribble(
  ~USUBJID,     ~AESER, ~AESEV,
  "01-351-739", "N",    "MODERATE",
  "01-354-945", "Y",    "SEVERE",
  "01-359-759", "Y",    "MILD"
)

# Add 'group' column and rename it to 'GROUP'
aes1 <- aes %>%
  mutate(group = case_when(
    AESER == "N" & AESEV == "MILD"     ~ "A",
    AESER == "N" & AESEV == "MODERATE" ~ "B",
    AESER == "Y" & AESEV == "MILD"     ~ "C",
    AESER == "Y" & AESEV == "MODERATE" ~ "D",
    TRUE ~ NA_character_
  )) %>%
  mutate(group = str_to_upper(group)) %>%
  rename(GROUP = group)

# View result
aes1
```

Now your column will appear as `GROUP` in the outpu[illegible]t me know if you also want to reorder the columns

```
# A tibble: 3 × 4
  USUBJID    AESER AESEV    GROUP
  <chr>      <chr> <chr>    <chr>
1 01-351-739 N     MODERATE B
2 01-354-945 Y     SEVERE   NA
3 01-359-759 Y     MILD     C
> 
```

Now the script is correct, and output matches the SAS output. This brings me to my second tip**: It's All About That Prompt!** To get a successful output from an AI, you must give it a good prompt. The prompt has to be specific. You have to iterate and fine-tune your prompt. You can't leave any room for assumptions. If you read the prompt and find yourself making assumptions, the AI is going miss those. AI's are good at syntax, bad at thinking. Also, if you are going to translate code and it is a large section of code you will want to chunk it into bite-size pieces and give it to the AI a bite at a time. AI's notoriously get confused by large code or even skip over large sections of it.

COMMENTING AND READABILITY

Another thing that AI can be very helpful with is commenting code and making it more readable. Programmers notoriously do not comment and document code very well. This is really a place where AI can step into the gap and make significant improvements in code output. Similar to how I asked the AI to translate SAS code into R code, I can also ask the AI to provide comments or documentation for a piece of code I have written.

Here I took some code from our open-source library that creates an ADSL dataset in R. Then I removed the comments from the code and asked the AI to "add comments to the following R code". Below are some of the results.

```
####----

# Load required packages
library(tidyverse)  # For data manipulation and piping
library(haven)      # For reading XPT (SAS transport) files
library(xportr)     # For exporting ADaM-compliant datasets and metadata

# Read and clean SDTM domains from GitHub
dm <- read_xpt(url("https://github.com/phuse-org/TestDataFactory/raw/main/Updated/T
  mutate(across(where(is.character), ~ replace(., . == "", NA)))  # Replace blank s

ex <- read_xpt(url("https://github.com/phuse-org/TestDataFactory/raw/main/Updated/T
  mutate(across(where(is.character), ~ replace(., . == "", NA)))

vs <- read_xpt(url("https://github.com/phuse-org/TestDataFactory/raw/main/Updated/T
  mutate(across(where(is.character), ~ replace(., . == "", NA)))
```

```
  AGEGR1N = case_match(AGEGR1, "<65" ~ 1, "65-70" ~ 2, ">70" ~ 3),

  # Map race to numeric codes
  RACEN = case_match(
    RACE,
    "WHITE" ~ 1,
    "BLACK OR AFRICAN AMERICAN" ~ 2,
    "ASIAN" ~ 3,
    "AMERICAN INDIAN OR ALASKA NATIVE" ~ 4,
    "NATIVE HAWAIIAN OR OTHER PACIFIC ISLANDER" ~ 5,
    "UNKNOWN" ~ 6
  ),

  # Treatment assignment (planned and actual)
  TRT01P = ARM,
  TRT01PN = case_match(TRT01P, "Placebo" ~ 0, "Xanomeline Low Dose" ~ 6, "Xanomeline High Dose"

  TRT01A = ACTARM,
  TRT01AN = case_match(TRT01A, "Placebo" ~ 0, "Xanomeline Low Dose" ~ 6, "Xanomeline High Dose"

  # Intent-to-treat flag
  ITTFL = if_else(!is.na(ARM), "Y", "N"),

  # Convert death date to date format
  DTHDT = ymd(DTHDTC)
) %>%
```

```r
# Add baseline height from VS
height <- vs %>%
  filter(VSTESTCD == "HEIGHT", VISITNUM == 1) %>%
  select(USUBJID, HEIGHTBL = VSSTRESN)

# Merge baseline vitals and calculate BMI
demo3 <- reduce(list(demo2, weight, height), left_join, by = "USUBJID") %>%
  mutate(
    BMIBL = (WEIGHTBL / HEIGHTBL / HEIGHTBL) * 10000,
    BMIGR1 = case_when(
      BMIBL < 25 ~ "<25",
      BMIBL >= 25 ~ ">=25"
    )
  )

# Dataset-level metadata
dlm <- tribble(
  ~dataset, ~label,
  "adsl",   "Subject-Level Analysis Dataset"
)

# Variable-level metadata
vlm <- tribble(
  ~dataset, ~variable, ~label, ~type, ~format,
  # [metadata table content remains unchanged]
  # ...
)

# Finalize ADSL dataset
adsl <- demo3 %>%
```

I think it did a pretty good job. It added comments about loading libraries, reading in data, and it even seems to have added a little bit about clinical data. Overall, I am pretty satisfied, this may even be better commenting than some programmers I know! Though I will caution you, as with everything that AI produces you still need to read through it with a critical and trained eye.

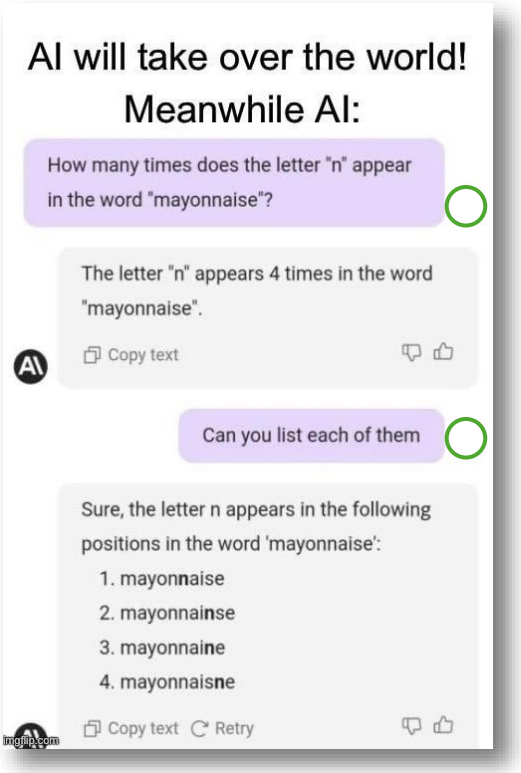
Another thing that can help with readability is my 3rd tip: **Use or develop a consistent style guide** for your team or organization. The more consistent your code is across users the more the AI is going to be able to provide you with consistent output that meets your expectations. You can even include this style guide in your prompt by telling the AI to use it or providing the AI with the url where the style guide is located.

DEBUGGING AND TESTING

The final place where I think AI can enhance your programming is in debugging and unit testing. As I have said, AI is a master of syntax, and most bugs and errors are syntax related. AI can be very helpful identifying the source of a bug or error, especially hard to find ones. It is able to scrape the internet and look for other instances and solutions that others have provided. It can even craft an explanation of the error to help you better be able to correct it. It can also be very helpful in writing unit tests to procedurally test your code and ensure it is going to provide the proper output with varying inputs. If you have not adopted unit testing, I suggest you do whether or not you use an AI to help you.

In order to be successful in having the AI help you debug and test you will want to pay attention to a few things. First, context is key. The more context you can provide the AI, including libraries, frameworks, and specific challenges you

are facing, the better the responses you receive will be. As with everything else I have said about AI specificity matters. You also need to recognize the AI's limitations. Review and understand its suggestions. Basically, don't just blindly accept its output as correct. Verify the output because you never know when the AI will just tell you there are 4 n's in Mayonnaise or that bees are a crucial part of a computer (see below). If the AI suggestions are not solving your problem remember to iterate, refine, and give feedback. Finally, combine your use of AI with traditional debugging techniques. Using the strength of the AI's knowledge or syntax and pattern recognition and your own knowledge of debugging workflows can potentially save you a lot of time debugging your code, but at this point you can't just turn it over to the machine.



That brings me to my 4th and final tip: **Validate**. You have to validate and verify everything the AI produces. The AI

is only as good of a programmer as you are. If you can't follow and understand the code the AI generates, then it is probably wrong or is leaving out some important edge case in your data. Use AI as a tool and not a replacement. I suggest you go through AI generated code line by line. Look for holes or inconsistencies where the AI might have struggled. What assumptions did the AI have to make? Is the code efficient, especially with large data? What are the edge cases that might have been overlooked? If you ask the right questions and make sure you absolutely understand what the AI is producing, then, and only then, can you safely and effectively use AI for programming.

CONCLUSION

For better or worse, AI is here to stay. Whether you're a skeptic like me, or an enthusiastic endorser of the new and shiny, there are some key things to keep in mind to use AI to the best of our ability. Just like a chainsaw or a hammer, AI is a tool, and you must know *how to use a tool correctly* to avoid causing damage. Keep my tips in mind to do just that.

Tips

1. **Do not put your proprietary code or data into a non-secure AI chatbot!**
2. **It's all about the prompt.**
3. **Use a consistent style guide.**
4. **Validate!**

REFERENCES

-Environmental Impact

The cost of AI is being dumped on you. (n.d.). https://thehill.com/opinion/technology/5671747-big-tech-energy-burden-communities/

O'Donnell, J. (2025, September 22). *We did the math on AI's energy footprint. here's the story you haven't heard.* MIT Technology Review. https://www.technologyreview.com/2025/05/20/1116327/ai-energy-usage-climate-footprint-big-tech/

David Berreby (n.d.). *As use of A.I. soars, so does the energy and water it requires*. Yale E360. https://e360.yale.edu/features/artificial-intelligence-climate-energy-emissions

AI is accelerating the loss of our scarcest natural resource: Water. (n.d.-a). https://www.forbes.com/sites/cindygordon/2024/02/25/ai-is-accelerating-the-loss-of-our-scarcest-natural-resource-water/

Adam Zewe | MIT News. (n.d.). *Explained: Generative AI's environmental impact*. MIT News | Massachusetts Institute of Technology. https://news.mit.edu/2025/explained-generative-ai-environmental-impact-0117
Aamna. (2024, October 15). *The hidden costs of ai*. Sustainability. https://sustainability.wustl.edu/the-hidden-costs-of-ai/

Guardian News and Media. (2025a, October 25). *Amazon strategised about keeping its datacentres' full water use secret, leaked document shows*. The Guardian. https://www.theguardian.com/technology/2025/oct/25/amazon-datacentres-water-use-disclosure

AI has an environmental problem. here's what the world can do about that. (n.d.-a). https://www.unep.org/news-and-stories/story/ai-has-environmental-problem-heres-what-world-can-do-about

-Misinformation

Endert, J. (2025, July 23). Generative AI is the ultimate disinformation amplifier. Deutsche Welle. https://akademie.dw.com/en/generative-ai-is-the-ultimate-disinformation-amplifier/a-68593890

Hicks, J. (2026, January 1). *AI-powered software is helping misinformation spread online after disasters*. NPR. https://www.npr.org/2026/01/01/nx-s1-5645183/ai-powered-software-is-helping-misinformation-spread-online-after-disasters

Wolf, A. (1970, October 20). *Special episode: Ai, propaganda and democracy: Inside a groundbreaking discovery by Vanderbilt researchers*. Vanderbilt University. https://www.vanderbilt.edu/quantumpotential/2025/10/20/special-episode-ai-propaganda-and-democracy-inside-a-groundbreaking-discovery-by-vanderbilt-researchers/

Ai-generated visual misinformation, propaganda, and democracy. Karsh Institute of Democracy. (n.d.). https://karshinstitute.virginia.edu/events/ai-generated-visual-misinformation-propaganda-and-democracy

Dawson, E. (n.d.). *Ai, propaganda, and the future of democracy*. WXXI News. https://www.wxxinews.org/show/connections/2025-11-04/ai-propaganda-and-the-future-of-democracy

The 60% problem — how AI search is draining your traffic. (n.d.-d). https://www.forbes.com/sites/torconstantino/2025/04/14/the-60-problem---how-ai-search-is-draining-your-traffic/

Landymore, F. (n.d.). *Study finds that AI search engines are wrong an astounding proportion of the time*. Futurism. https://futurism.com/study-ai-search-wrong

User, S. (n.d.). *Google's AI overviews make users search worse, not better*. Eelco Herder. https://eelcoherder.com/organization-and-outreach/research-blog/24-googles-ai-overviews-make-users-search-worse-not-better

If you use ai search at work, be careful: A new study finds it lies. (n.d.-d). https://www.inc.com/kit-eaton/if-you-use-ai-search-at-work-be-careful-a-new-study-finds-it-lies/91161537

Guardian News and Media. (2025, December 27). *More than 20% of videos shown to new YouTube users are "ai slop", study finds*. The Guardian. https://www.theguardian.com/technology/2025/dec/27/more-than-20-of-videos-shown-to-new-youtube-users-are-ai-slop-study-finds

Markander, M., Eriksson, V., & Brans, P. (2026, January 5). *OpenAI admits ai hallucinations are mathematically inevitable, not just engineering flaws*. Computerworld. https://www.computerworld.com/article/4059383/openai-admits-ai-hallucinations-are-mathematically-inevitable-not-just-engineering-flaws.html

-Harassment

Artificial Intelligence and Online Harassment: Parkview Health. Parkview. (n.d.). https://www.parkview.com/blog/artificial-intelligence-and-online-harassment

Bradford J. Kelley and Alyesha Asghar, (n.d.). *Deepfakes in the workplace: The emerging legal risks of AI-Driven Harassment*. Littler. https://www.littler.com/news-analysis/asap/deepfakes-workplace-emerging-legal-risks-ai-driven-harassment

Guardian News and Media. (2026, January 5). *Elon Musk's grok AI is used to digitally undress images of women and children*. The Guardian. https://www.theguardian.com/technology/2026/jan/05/elon-musk-grok-ai-digitally-undress-images-of-women-children

-Financial Impact

What we mean when we talk about an AI "bubble." World Economic Forum. (n.d.). https://www.weforum.org/stories/2025/10/artificial-intelligence-bubble-dot-com-tulip-mania/

This is how the AI bubble bursts. Yale Insights. (2025, October 8). https://insights.som.yale.edu/insights/this-is-how-the-ai-bubble-bursts

AI investment led to zero returns for 95% of companies in MIT Study. (n.d.-a). https://axios.com/2025/08/21/ai-wall-street-big-tech?amp=&amp=

Ai-driven inflation is 2026's most overlooked risk, investors say | Reuters. (n.d.-c). https://www.reuters.com/world/asia-pacific/ai-driven-inflation-is-2026s-most-overlooked-risk-investors-say-2026-01-05/

Guardian News and Media. (2026a, January 4). *The cost of AI slop could cause a rethink that shakes the global*

economy in 2026. The Guardian. https://www.theguardian.com/business/2026/jan/04/ai-reality-growing-economic-risk-2026

Cassidy, J. (2025, August 25). *The a.i.-profits drought and the lessons of history*. The New Yorker. https://www.newyorker.com/news/the-financial-page/the-ai-profits-drought-and-the-lessons-of-history?utm_source=threads&utm_medium=social&utm_campaign=tny&utm_social-type=owned

-Affect on Programming

Metr. (2025, July 10). *Measuring the impact of early-2025 AI on experienced open-source developer productivity*. METR Blog. https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/

Yepis, E. (2025, December 29). *Developers remain willing but reluctant to use AI: The 2025 developer survey results are here*. Stack Overflow. https://stackoverflow.blog/2025/12/29/developers-remain-willing-but-reluctant-to-use-ai-the-2025-developer-survey-results-are-here/

Jingnan, H. (2025, October 21). *Tech CEOS say the era of "Code by AI" is here. some software engineers are skeptical*. NPR. https://www.npr.org/2025/10/21/nx-s1-5506141/ai-code-software-productivity-claims

Namanyayg. (2025, February 14). *New junior developers can't actually code*. N's Blog. https://nmn.gl/blog/ai-and-learning

Sang, T. Đức. (2025, December 26). *When ai makes you forget how to code*. DEV Community. https://dev.to/mahou_anisphia/when-ai-makes-you-forget-how-to-code-5cii

How ai made me forget how to code — and what I'm doing about it | by Ankit Singh | Medium. (n.d.-e). https://medium.com/@ankitsingh1583/how-ai-made-me-forget-how-to-code-and-what-im-doing-about-it-360aa3a07e06

Ai killed my coding brain but I'm rebuilding it | by | run.it.bare | medium. (n.d.-d). https://medium.com/devlink-tips/ai-killed-my-coding-brain-but-im-rebuilding-it-8de7e1618bca

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Frank Menius
Company: Atorus Research
Work Phone: 919-412-6537
Email: frank.menius@atorusresearch.com
Website: https://www.linkedin.com/in/frank-menius/