

AI is the new UI

› Generative UI with FastHTML

Original [Medium Article](#)

All links:

- x.com/pol_avec
- justreadthe.info



by Pol Avec

The Future of User Interactions

"In the near future, the primary software interfaces will be powered by AI..."

But current interfaces are still quite far away, let's see an example.

The current shopping UI/UX

This screenshot captures the Amazon product page for the novel 'Waiting for the Barbarians' by J.M. Coetzee. At the top, the search bar displays 'waiting for the barbarians'. The main product image is the book cover, which features a horse's head and the title 'Waiting for the Barbarians' by J.M. Coetzee. Below the image, a summary highlights the book as a modern classic by Nobel laureate J.M. Coetzee, available from Viking. It also mentions a film starring Mark Rylance, Johnny Depp, and Robert Pattinson.

The product details section includes the Kindle Edition price at \$6.99, marked down from \$16.00, and various physical formats like Library Binding (\$34.70) and Paperback (\$11.51). A 'Buy now with 1-Click' button is prominently displayed. The page also features sections for 'Buy for others' and 'Follow the author'.

Key elements visible include:

- Search bar: waiting for the barbarians
- Top navigation: Deliver to Jordi San Franc... 94129, All, Hello, Jordi Account & Lists, Returns & Orders, Cart
- Breadcrumbs: Books > Literature & Fiction > Genre Fiction > Historical > Military
- Product image: Waiting for the Barbarians by J.M. Coetzee
- Product title: Waiting for the Barbarians: A Novel Kindle Edition
- Author: J. M. Coetzee (Author)
- Ratings: 4.3 ★★★★☆ (1,638) | 3.9 on Goodreads 34,489 ratings
- Description: NOW A FILM STARRING MARK RYLANCE, JOHNNY DEPP, AND ROBERT PATTINSON
- Summary: A modern classic by Nobel Laureate J.M. Coetzee. His latest novel, *The Schooldays of Jesus*, is now available from Viking. *Late Essays: 2006–2016* will be available January 2018.
- Text: For decades the Magistrate has been a loyal servant of the Empire, running the affairs of a tiny frontier settlement and ignoring the impending war with the barbarians. When interrogation experts arrive, however, he witnesses the Empire's cruel and unjust treatment of prisoners of war. Jolted into sympathy for their victims, he commits a quixotic act of rebellion that brands him an enemy of the state.
- Text: J. M. Coetzee's prize-winning novel is a startling allegory of the war between oppressor and oppressed. The Magistrate is not simply a man living through a crisis of conscience in an obscure place in remote times; his situation is that of all men living in unbearable complicity with regimes that ignore justice and decency.
- Product details: Print length (194 pages), Language (English), Publisher (Penguin Books), Accessibility (Learn more), Publication date (January 3, 2017)
- Buy options: Kindle \$6.99, Audiobook \$0.00, Library Binding \$34.70, Paperback \$11.51, Other Used, New, Collectible from \$1.26
- Discount: -56% \$6.99
- Print List Price: \$16.00
- You Earn: 21 pts Details
- Buy now with 1-Click
- By placing an order, you're purchasing a content license & agreeing to Kindle's Store Terms of Use.
- Sold by Penguin Group (USA) LLC. Price set by seller.
- Read with our free app
- Deliver to your Kindle Library
- Buy for others
- Follow the author: J. M. Coetzee

The Evolution of LLM UX

- 1 — Text-only
- 2 — GenUI Display-only
- 3 — GenUI Interactive



LLM UX: Text only

Simplest text-based chat interfaces

I'm in the mood for something new to read. Maybe cultural or technical. Surprise me.

Bold request. Let's mix guilt, algorithms, and imperial dread. Here are three books you might enjoy:

1. **Crime and Punishment** by Fyodor Dostoevsky — A psychological masterpiece where poor choices meet worse consequences. You'll start questioning your own morality by page 30.
2. **Hypermedia Systems** by Adam Stepinski — A tactical guide to modern web systems with a surprising number of feelings about HTMX. Not quite Dostoevsky, but dangerously opinionated.
3. **Waiting for the Barbarians** by J.M. Coetzee — Bleak, brilliant, and unsettling. Perfect for when you want your afternoon tea to taste like existential dread.

I want to buy number two.

Excellent choice. *Hypermedia Systems* will be added to your cart. Code over conscience, as they say. Want anything else while we spiral?

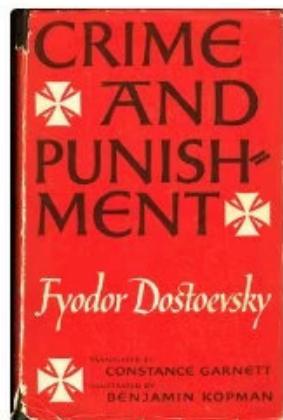
Send a message... Send

LLM UX: GenUI Display Only

Shows product cards but you still need to type your answers

I'm in the mood for something new to read. Maybe cultural or technical. Surprise me.

Bold request. Let's mix guilt, algorithms, and imperial dread. Here are three books you might enjoy:



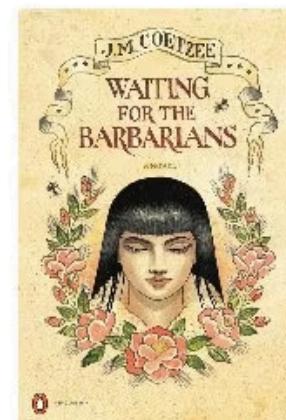
Crime and Punishment

A psychological masterpiece where poor choices meet worse consequences. You'll start questioning your own morality by page 30.



Hypermedia Systems

A tactical guide to modern web systems with a surprising number of feelings about HTMX. Not quite Dostoevsky, but dangerously opinionated.



Waiting for the Barbarians

Bleak, brilliant, and unsettling. Perfect for when you want your afternoon tea to taste like existential dread.

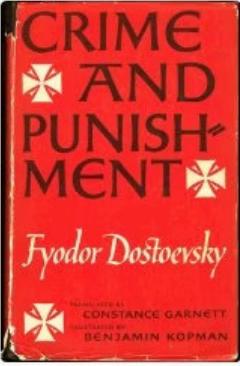
I want to buy number two.

Excellent choice. *Hypermedia Systems* will be added to your cart. Code over conscience, as they say. Want anything else while we spiral?

LLM UX: Interactive GenUI

I'm in the mood for something new to read. Maybe cultural or technical. Surprise me.

Bold request. Let's mix guilt, algorithms, and imperial dread. Here are three books you might enjoy:



Crime and Punishment

A psychological masterpiece where poor choices meet worse consequences. You'll start questioning your own morality by page 30.

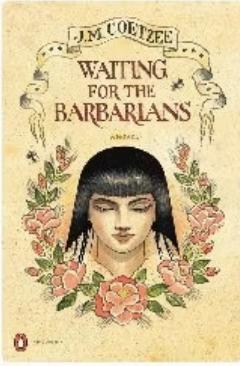
[Buy Now](#)



Hypermedia Systems

A tactical guide to modern web systems with a surprising number of feelings about HTMX. Not quite Dostoevsky, but dangerously opinionated.

[Buy Now](#)



Waiting for the Barbarians

Bleak, brilliant, and unsettling. Perfect for when you want your afternoon tea to taste like existential dread.

[Buy Now](#)

You're about to purchase *Hypermedia Systems*. Would you like to continue?

[Confirm Purchase](#) [Cancel](#)

shopping is already possible but...

Find books from the image on Amazon

+ ⓘ ...

Image talk.ipynb

Can you find the 3 books mentioned in the image in amazon?

⌚ Thought for 3 seconds

I can see the 3 books mentioned in the conversation image:

1. **Crime and Punishment** by Fyodor Dostoevsky
2. **Hypermedia Systems** by Adam Stepinski
3. **Waiting for the Barbarians** by J.M. Coetzee

Let me search for these books on Amazon for you.

> Called MCP tool amazon_search ✓

> Called MCP tool amazon_search ✓

> Called MCP tool amazon_search ✓

Perfect! I found all 3 books mentioned in the conversation image. Here's what I found on Amazon:

1. **Crime and Punishment** by Fyodor Dostoevsky

Best Options:

- **Vintage Classics Edition** - \$11.04 (Paperback) ★ 4.6/5 (4,886 reviews)
 - Also available as Kindle for \$0.99
 - Free delivery available
- **Dover Thrift Edition** - \$6.51 (Paperback) ★ 4.6/5 (3,657 reviews)
 - Great budget option
 - Also available as Kindle for \$0.99
- **Penguin Classics Edition** - \$14.36 (Paperback) ★ 4.6/5 (4,253 reviews)

a brief note on MCP...

MCP (Model Context Protocol) is an **open standard** way of connecting tools to LLMs.

Requires an MCP client like Claude Dekstop, Cursor, Windsurf ([see more](#))

```
@mcp.tool()
async def amazon_search(q: str, domain: str = "amazon.com") -> Dict:
    """
    Search for products matching the query in amazon.
    If the user does not specify a domain, try to infer the domain from the product description or user's language.

    Args:
        q: The search query of a specific ASIN of a given product.
        domain: The amazon domain of the search. E.g. amazon.com, amazon.es ...

    Returns:
        The search results.
    """
    response = get_amazon().search(
        query=q,
        domain=domain
    )
    return handle_response(response)
```

would you buy like this?

Find books from the image on Amazon

+ ⚡ ...

talk.ipynb

buy Waiting for the Barbarians, use my billing address

Waiting for approval.. Stop ⚡

⌚ Thought for 7 seconds

I'll help you purchase "Waiting for the Barbarians" by J.M. Coetzee. Let me first get your billing information to use as the shipping address, then get the payment offers for the book.

> Called MCP tool billing_info ✓

Now let me get the payment offers for "Waiting for the Barbarians". I'll use the paperback version which was priced at \$18.44 from the search results.

> Called MCP tool amazon_get_payment_offers ✓

Perfect! I have the payment offer for "Waiting for the Barbarians". The book costs \$18.44 USD. Now I'll process the payment for you.

> Calling MCP tool pay_offer

Cancel ⚡ Run tool ⚡

The Opportunity

- Gap between AI capability and UI/UX
- Every AI tool going mainstream needs proper interfaces
- Your existing web dev skills + AI integration = new market
- Someone needs to build the interfaces - that's where you come in

Questions so far? Recap

1. **Current Chat interfaces**
 - a. Text-based
 - b. Not interactive
2. **Static GenUI**
 - a. Present information visually through UI components
 - b. Not interactive
3. **Interactive GenUI**
 - a. Present information visually through UI components
 - b. Components ARE interactive

Let's build some LLM UI/UX

The Basics (I): LLM APIs

- Simple Python call to get AI responses

```
from claudette import *

chat = Chat(model=models[-1], sp="You are a helpful assistant.")

r = chat("Hello! What's your name?")
r

✓ 1.7s
```

I'm Claude, an AI assistant created by Anthropic. I aim to be direct and honest about what I am. How can I help you today?

The Basics (II): Tool Use

```
def sums(  
    a:int, # First thing to sum  
    b:int=1 # Second thing to sum  
) -> int: # The sum of the inputs  
    "Adds a + b."  
    print(f"Finding the sum of {a} and {b}")  
    return a + b
```

✓ 0.0s

```
a,b = 604542,6458932  
pr = f"What is {a}+{b}?"  
pr
```

✓ 0.0s

'What is 604542+6458932?'

```
chat = Chat(models[-1], sp='You perform sums', tools=[sums])  
r = chat.toolloop(pr)  
r
```

✓ 4.8s

Finding the sum of 604542 and 6458932

The sum of 604542 and 6458932 is 7063474.

this is an agent by the way

```
chat = Chat(models[-1], sp='You perform sums', tools=[sums])
r = chat.toolloop(pr)
r
```

The Basics (III): FastHTML

- FastHTML is a new way to create modern interactive web apps.
- apps starting at a 6 LoC python file
- easy OAuth, DBs, caching, styling
- extremely easy deploy w/ [Pla.sh](#)



The Basics (III): FastHTML

- In FastHTML, you can create UI components using pure Python.

```
from fasthtml.common import *

def WeatherCard(
    location: str,      # The city name
    temperature: int,   # Temperature in Celsius
    condition: str      # Weather condition like 'sunny', 'cloudy', 'rainy'
):
    "Generate a weather card for a given location, temperature, and condition"
>     weather_icons = ...
    return Div(
        H3(location),
        P(f"{temperature}°C - {condition.capitalize()}"),
        Img(src=weather_icons[condition], width=100, height=100,),
        style="border: 1px solid #ccc; padding: 20px; margin: 10px; border-radius: 8px;"
    )

# Test the component
weather_card = WeatherCard("Paris", 22, "sunny")
weather_card

9] ✓ 0.0s
· <div style="border: 1px solid #ccc; padding: 20px; margin: 10px; border-radius: 8px;">
  <h3>Paris</h3>
  <p>22°C - Sunny</p>
  </div>
```

Putting it all together

- FastHTML components can also serve as tools for Claude to use in chat!

```
sp = 'You are a helpful assistant that invents weather for a specific location.'  
cli = Client(models[-1])  
  
r = cli.structured([sp, 'What is the weather in Tokyo?'], tools=[WeatherCard])  
HTML(str(r[0]))
```

✓ 2.2s

Tokyo

24°C - Sunny



Let's see some real demos

Weather Demo

Displaying static cards about the weather

Live demo <https://fasthtml-app-cbd32e55.pla.sh/>

Weather Component

Tokyo

Monday, June 09

Sunny

72°F° 

user

what's the weather in Tokyo

Send

Weather Demo

Single POST endpoint handles interaction.

```
# Handle the form submission
@app.post
def send(msg:str, messages:list[str]=None):
    if not messages: messages = []
    messages.append(msg.rstrip())

    cli = Client(model)
    sp="""You are a helpful assistant that invents weather for a specific location.
        Use the tool WeatherComponent to generate a card for the given location."""
    r = cli.structured([sp, msg], tools=[WeatherComponent])
    return (ChatMessage(msg, True),    # The user's message
            r[0], # The chatbot's response
            ChatInput()) # And clear the input field via an OOB swap
```

Weather Demo

X	Headers	Payload	Preview	Response	Initiator	Timing	Cookies
1	<div class="chat chat-end">						
2	<div class="chat-header">user</div>						
3	<div class="chat-bubble chat-bubble-primary">what's the weather in Tokyo</div>						
4	<input type="hidden" value="what's the weather in Tokyo" name="messages">						
5	</div>						
6	<div class="p-4 bg-sky-500 text-white rounded-lg max-w-xs">						
7	<h2 class="uk-h2 text-xl font-semibold mb-1">Tokyo</h2>						
8	<div class="flex justify-between items-center mb-3">						
9	<p class="text-sm font-medium">Monday, June 09</p>						
10	<p class="text-sm font-medium">Sunny</p>						
11	</div>						
12	<div class="flex items-center">						
13	72°F						
14	</div>						
15	<input name="msg" placeholder="Type a message" hx-swap-oob="true" id="msg-input" class="uk-input input-bordered w-full">						
16							

The User's message

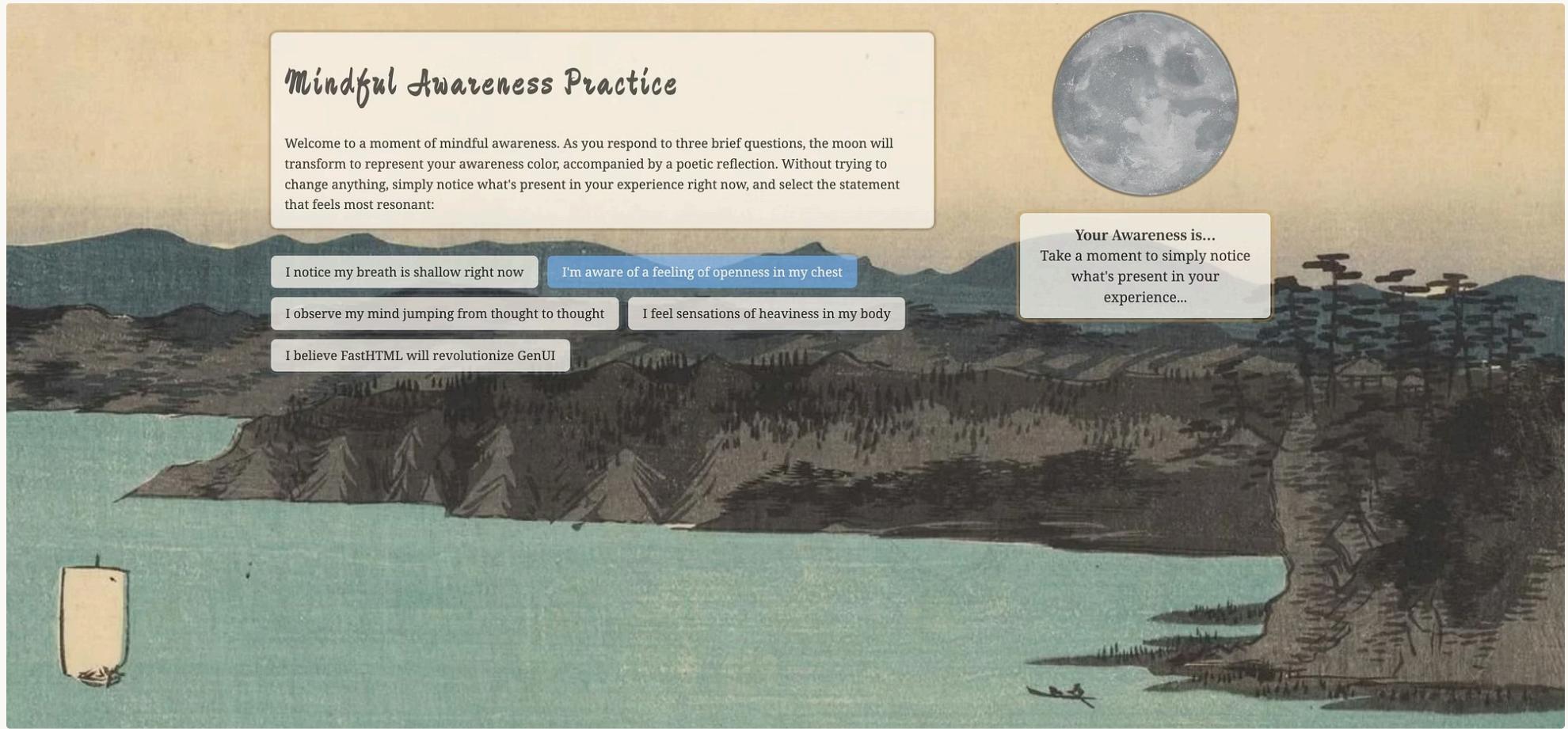
The Chatbot's response

Clear Input

Mindful Demo: Interactive buttons

Our UI is now interactive, accepting button clicks

Live demo <https://fasthtml-app-68e1764d.plash.it/>



Mindful Demo: Interactive buttons

Still using the **same POST endpoint** as before

```
@app.post
async def send(request):
    form_data = await request.form()
    user_choice = first(form_data.keys())

    messages.append(user_choice)

    r = cli.structured(messages, tools=[generate_response, generate_finish_response])
    response_text, color, color_component, new_buttons = r[0]
    messages.extend([color, response_text])

    return (
        UserReply(user_choice),
        Reflection(response_text),
        color_component,
        new_buttons)
```

HAL9000 Demo: Text + Buttons

- Interaction now accepts **BOTH text & buttons**, brings together AI Chats & traditional UI/UX
- Live demo <https://fasthtml-app-6e583cf.c.fc.plash/>



HAL9000 Demo: Text + Buttons

Everything is **still a single endpoint** handling text & interactions

```
# Handle the form submission from buttons
@app.post
async def send(request):

    form_data = await request.form()
    usr_choice = first(form_data.keys()) # result of clicking buttons
    usr_msg = form_data.get('user_message', '') # result of typing in the input area

    # If user clicked "Try again", reload the page
    if usr_choice == "Try again": return Div(hx_get="/", hx_trigger="load", hx_swap="outerHTML")

    # Add the user's choice to messages
    msg = usr_msg if usr_msg else usr_choice
    if msg: messages.append(msg)

    # Get Claude's response
    r = cli.structured(messages, tools=[generate_hal_response])
    response_text, environment_description, color_component, new_buttons = r[0]

    messages.extend([response_text, environment_description])

    return (
        UserReply(msg),
        HalMessage(response_text),
        EnvironmentMessage(environment_description),
        new_buttons,
        color_component,
        InputArea())
```

Solving Contract Coupling

The LLM can generate ANY HTTP Interaction

- Key insight: Responses are HTML, not JSON
- Backend (AI) completely controls frontend experience
- No contract coupling - AI generates whatever it wants, it knows where it's listening

Cool demos, so what?

Beyond UI/UX

Everything we did when moving from big screens to mobile phones,

→ we will do again moving to AI interfaces.

The same kind of issues and adaptations will happen across many domains.

The Intersection Opportunity

- Many AI challenges aren't entirely new
- Expertise + learn AI basics = rare combination
- **Opportunity: Be early in the intersection**

APIs

DON'Ts About MCP: "We already have APIs & docs"

Actually building LLM tools: "How do I handle function descriptions? Docstrings? Prompt updates?"

These daily problems have no clear solutions yet

Monitoring

- Traditional: Predictable input → predictable output
- AI: Same input → different outputs
- New need: Re-run steps with different models/prompts
- Same monitoring concepts, new testing requirements

Your Skills Will Adapt, Not Disappear

- Core expertise transfers to AI context
- Human psychology unchanged - people still want good UX
- New challenges, familiar solutions, not only in UI/UX

Build Intuition Through Practice

- Reading about MCP: "It's just an API with docs"
 - Actually building LLM tools: "How do I handle function descriptions? Docstrings? Prompt updates?"
 - These daily problems have no clear solutions yet
-
- **Huge opportunity for early practitioners**

Why I'm Sharing This

- I don't want to solve UI problems
- But someone needs to fix interfaces so I can focus on payments
- The ecosystem needs specialists in every area
- **Your chance to own a piece of the AI stack**

Future is Now

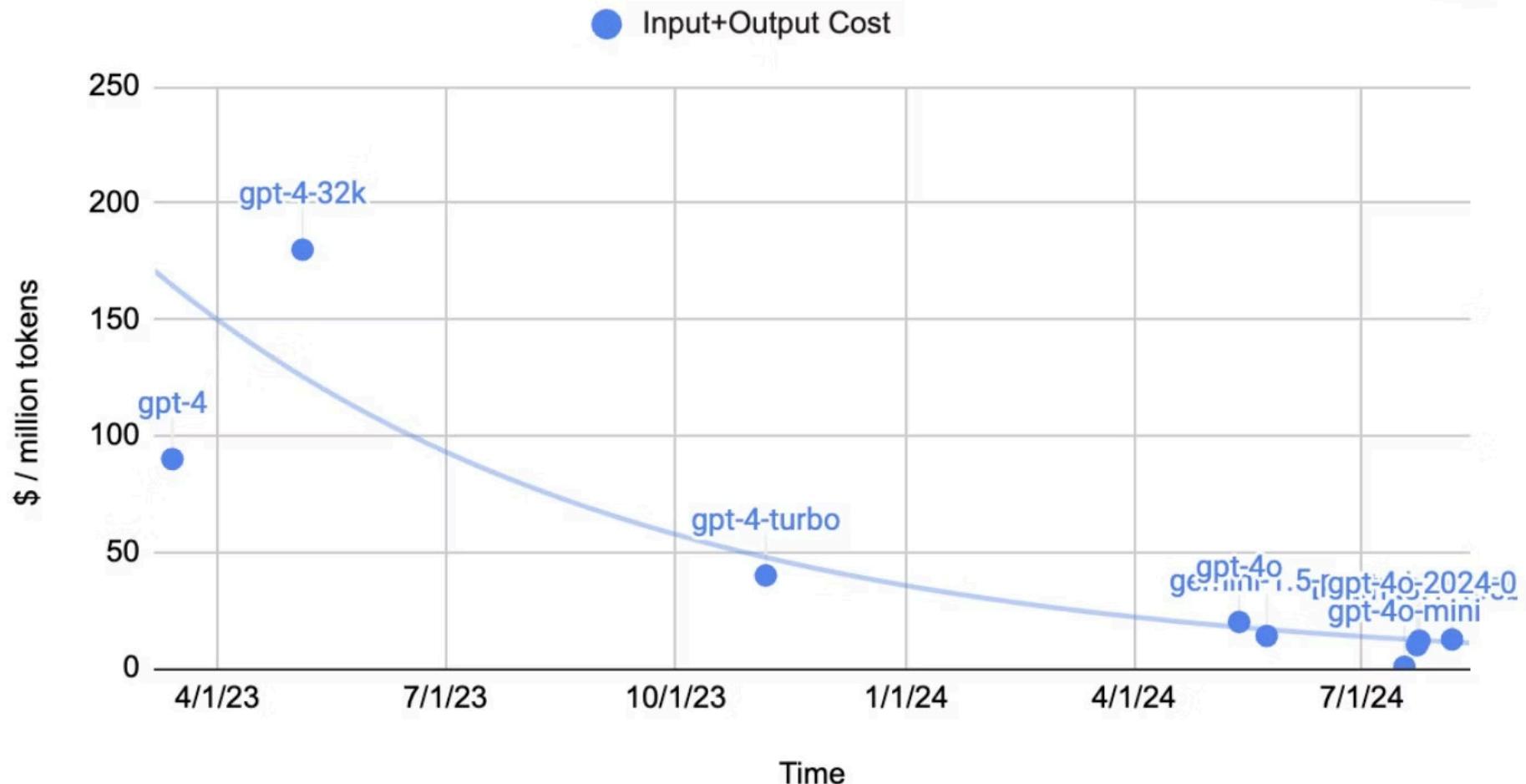
~~Since we wrote Talk2APIs in January...~~

Since I last talked in October's Python Meetup

Cost Evolution (recap)

Token Cost of GPT-4 level models over time

C



Cost for 2 million tokens (input+output) decreased from \$180->\$0.75 in 2 years. 240x cheaper



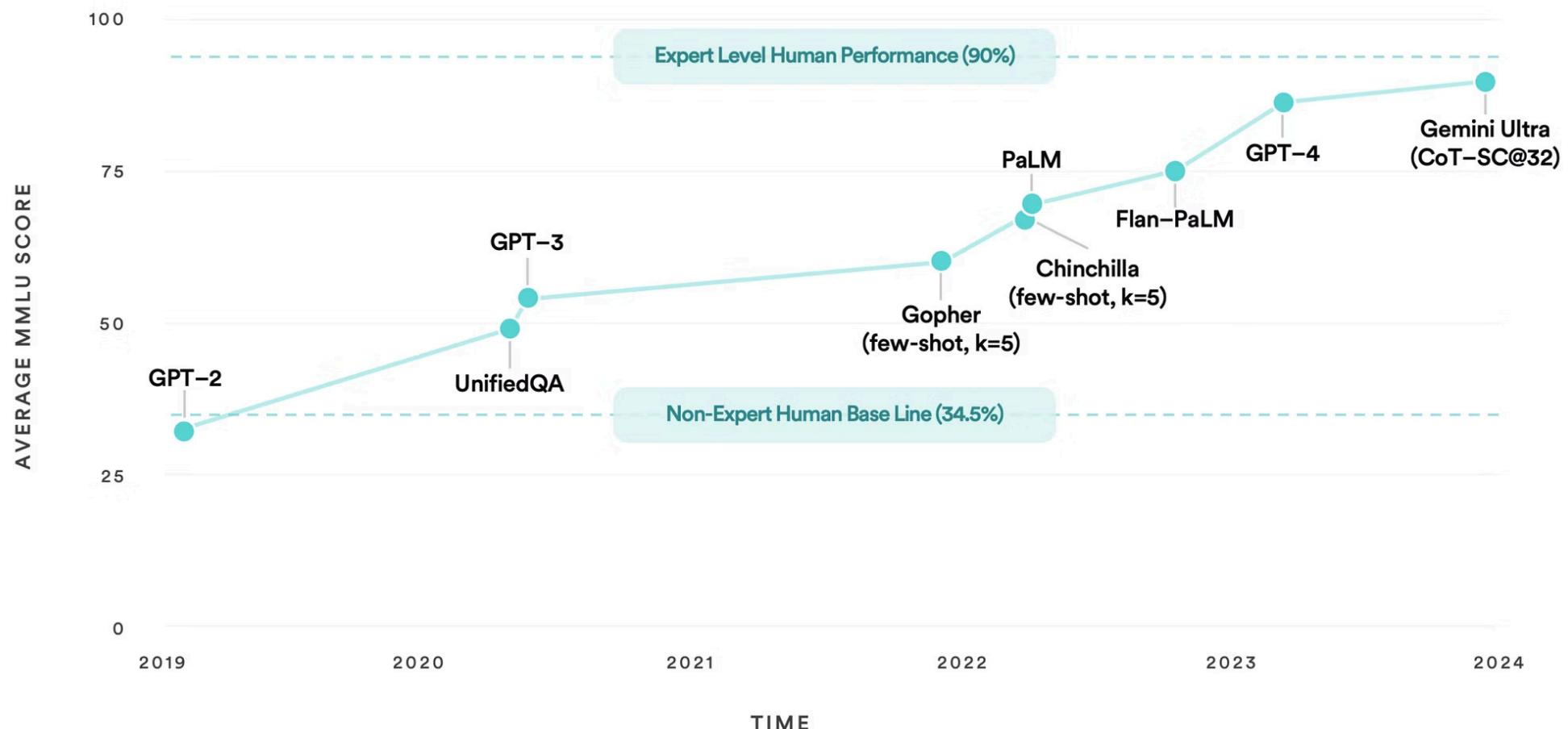
Elad Gil on Twitter / X



From @davidtsong on my team
Cost of 1M tokens has dropped from \$180 to \$0.75 in ~18 months (240X cheaper!)
pic.twitter.com/1paC8AKhXG— Elad Gil (@eladgil) August 25, 2024

Performance Evolution (recap)

Rise of Superhuman Generative AI

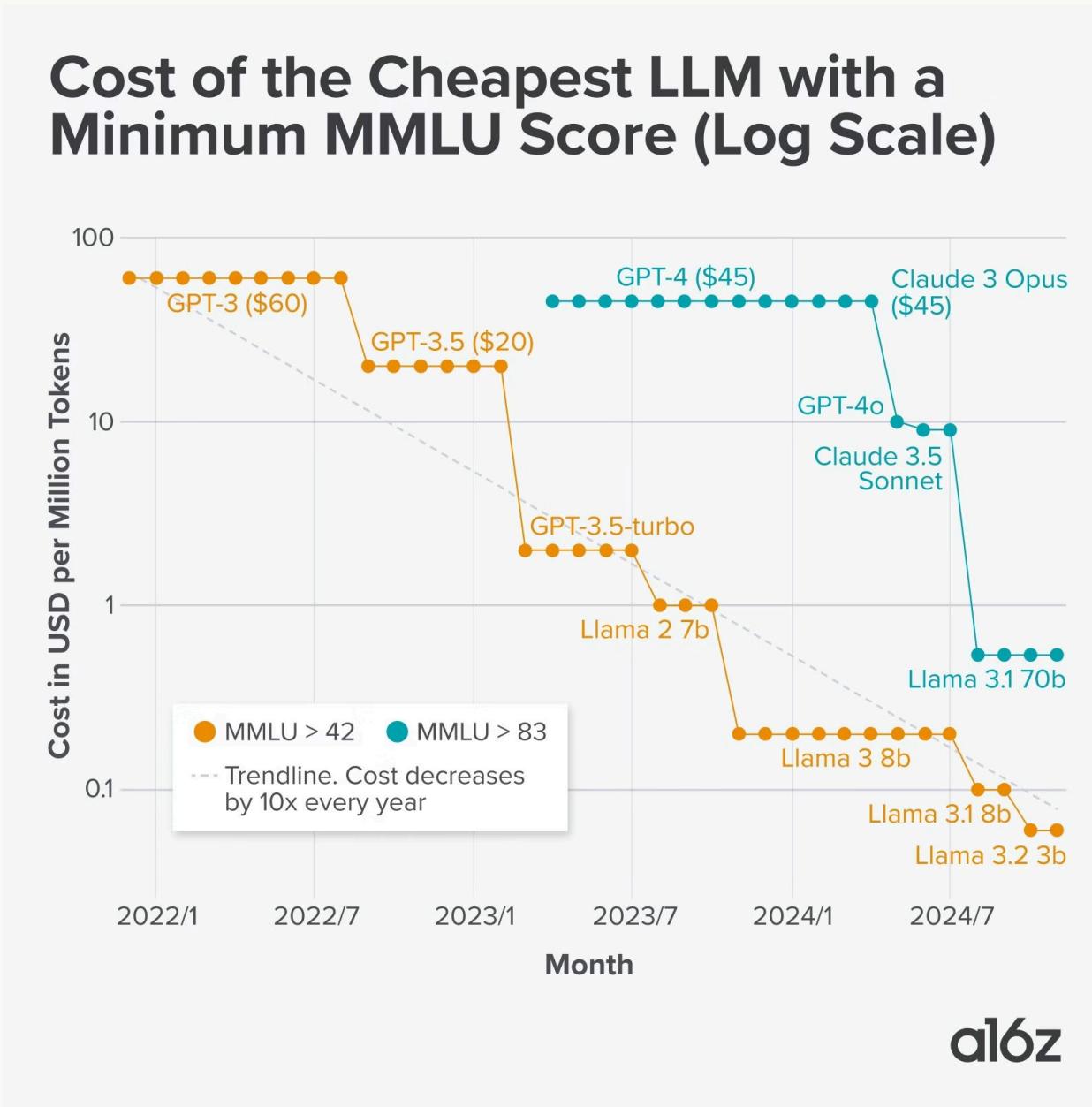


Elad Gil on Twitter / X

From @davidtsong on my team Cost of 1M tokens has dropped from \$180 to \$0.75 in ~18 months (240X cheaper!)
pic.twitter.com/1paC8AKhXG— Elad Gil (@eladgil) August 25, 2024



Performance Evolution (NOW)



Speed generation

New promising Diffusion Generation which is almost INSTANT

The logo features the text "Gemini Diffusion" in white on a dark blue background with a subtle grid pattern.

Google DeepMind

Gemini Diffusion

Gemini Diffusion is our state-of-the-art research model exploring what diffusion means for language – and text generation.

A small icon of a square with arrows pointing outwards is located in the top right corner of the card.

Thanks, Links & Questions

- x.com/pol_avec
- justreadthe.info