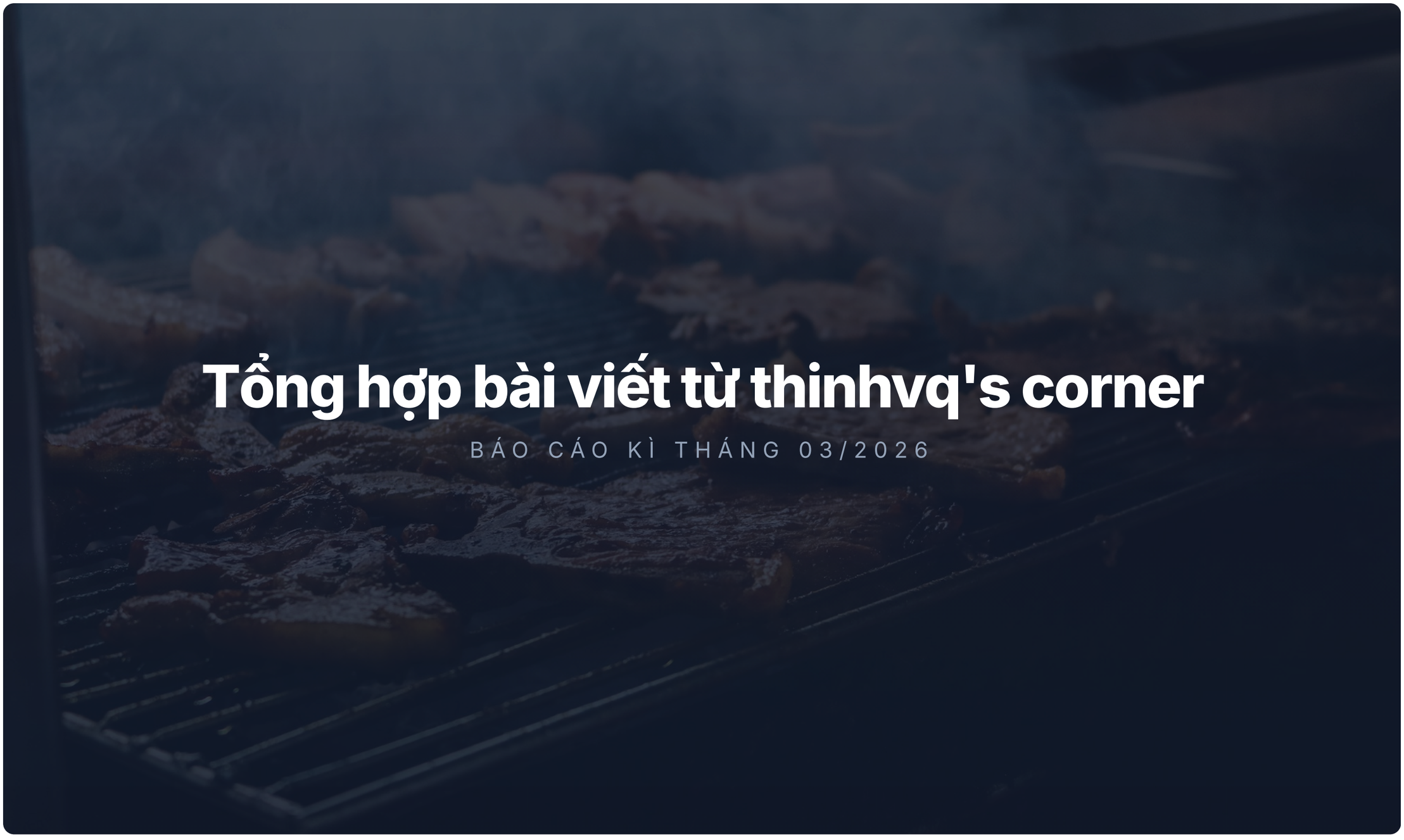


Tổng hợp bài viết từ thinhvq's corner

BÁO CÁO KÌ THÁNG 03/2026

MỤC LỤC BÀI VIẾT (9 BÀI)

1.

Human-in-the-loop

Thịnh Võ Quốc • 19/03/2026

2.

TSR sắp phát hành dự án mã nguồn mở đầu tiên

Thịnh Võ Quốc • 17/03/2026

3.

Dự đoán tương lai với AI MiroFish

Thịnh Võ Quốc • 16/03/2026

4.

Con hổ và luật rừng

Thịnh Võ Quốc • 09/03/2026

5.

Chuyện dĩa cơm lên giá

Thịnh Võ Quốc • 05/03/2026

6.

FFmpeg, Meta và câu chuyện big corp hỗ trợ opensource

Thịnh Võ Quốc • 04/03/2026

7.

AI đập đi xây lại Next.js

Thịnh Võ Quốc • 03/03/2026

8.

Sự xói mòn tính mở của hệ điều hành Android và rủi ro từ sự kiểm soát tập trung

Thịnh Võ Quốc • 02/03/2026

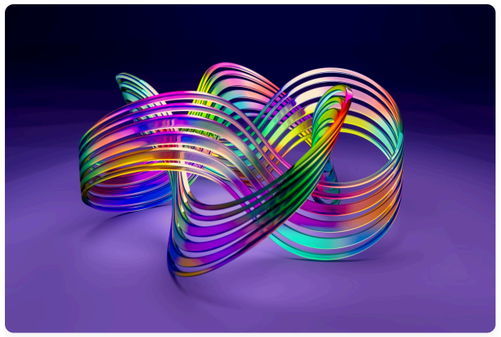
9.

Câu chuyện của AI và COBOL

Thịnh Võ Quốc • 01/03/2026

19/03/2026 • THỊNH VÕ QUỐC

Human-in-the-loop



Hôm qua trong cuộc họp báo cáo tiến độ các dự án, tôi có được nghe B trình bày về dự án Ecombox AI Agent, B có nói một cụm từ mà tôi thấy rất đúng và sẽ xem là kim chỉ nam cho các dự án AI sau này. Cụm từ đó là "Human-in-the-loop", đoạn sau tôi sẽ dùng từ viết tắt là HITL.

HITL nôm na là sự kết hợp giữa con người và AI. Con người đóng vai trò giám sát, hướng dẫn và sửa lỗi để đảm bảo kết quả đầu ra chính xác và hiệu quả hơn. Một điều làm tôi bất ngờ khi tìm hiểu và HITL là nó đã xuất hiện từ rất lâu, ngay từ thời 1940-1950. Vào thời gian đó, HITL xuất hiện trong các nghiên cứu về cybernetics và lý thuyết điều khiển. Nhà khoa học Norbert Wiener chính là người đặt nền móng đầu tiên.

Mặc dù thời điểm hiện tại có rất nhiều dự án AI Agent hướng tới tự động hoá hoàn toàn, ví dụ như dự án OpenClaw đang nổi gần đây. Nhưng theo quan điểm của tôi HITL là yếu tố không thể thiếu đối với những nhiệm vụ mang tính chất quan trọng. Nếu bạn là một người dùng AI cơ bản, không cần bạn là chuyên gia vẫn có thể nhìn thấy được có những vấn đề sau:

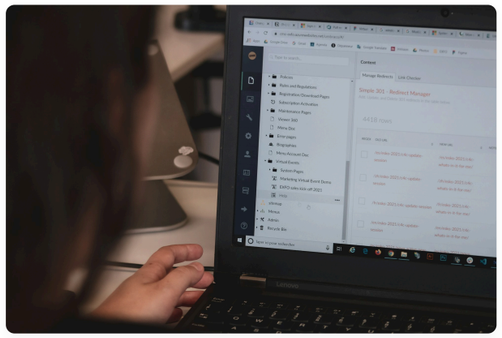
- AI có định kiến và ảo giác (hallucinations). Định kiến đó là kết quả từ dữ liệu huấn luyện. Nếu bạn dùng AI trong những tác vụ quan trọng, bạn không giám sát thì định kiến đó có thể dẫn tới sai lầm nghiêm trọng. Để minh hoạ cho điều này tôi sẽ lấy ví dụ bạn cần làm một báo cáo về tình trạng nhân quyền ở nước ABC. Bạn không thể phó thác cho AI tìm hiểu hết được vì tôi chắc chắn dữ liệu AI được huấn luyện đa phần sẽ lấy từ báo chí, các kênh tuyên truyền của nước ABC. Nó sẽ dẫn tới kết quả bị sai lệch so với thực tế.
- AI chỉ là máy móc. Nó cần bạn để bạn bổ sung ngữ cảnh, để bạn xác nhận và đưa ra các khía cạnh đạo đức.
- AI cần bạn vì bạn là con người. Bạn sẽ chịu trách nhiệm cho hành động chứ không phải .
- Độ chính xác thì không cần bàn tới. Chắc chắn nếu có bạn đồng hành cùng AI, kết quả của task đó sẽ tốt hơn bạn giao cho nó tự làm 100%.

Từ kinh nghiệm làm các dự án AI generative, tôi nhận thấy người dùng đôi khi không thực sự biết họ muốn gì. Như ở dự án Deligent, việc tạo ra một hình ảnh để in lên áo tưởng chừng đơn giản nhưng người dùng thường lúng túng trong việc lựa chọn hoặc xác định rõ nhu cầu của mình. Với triết lý HITL, thay vì chỉ gõ một prompt dài rồi thụ động đợi kết quả, người dùng sẽ đồng hành trong từng bước của quá trình tạo hình. Việc trực tiếp tham gia không chỉ giúp họ gắn bó hơn với sản phẩm mình tạo ra mà còn tiết kiệm đáng kể chi phí tính toán do hạn chế được việc phải tạo lại nhiều lần.

Chính vì thế, ở Ecombox và các dự án sau này, tôi chọn HITL làm kim chỉ nam để định hướng sản phẩm. Với tôi, con người phải là trọng tâm. Dù AI có tiến hóa đến đâu, nó vẫn cần sự đồng hành của con người để tạo ra giá trị thực tế. Trong tương lai, có thể AI sẽ đạt đến ngưỡng tự động hóa hoàn toàn mà không cần hỗ trợ, nhưng để những kết quả đó có ý nghĩa thì con người vẫn sẽ là thước đo và thang đánh giá hiệu quả cuối cùng.

17/03/2026 • THỊNH VÕ QUỐC

TSR sắp phát hành dự án mã nguồn mở đầu tiên



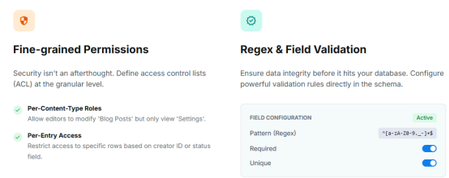
Là người luôn yêu thích và ủng hộ phong trào mã nguồn mở, tôi luôn mong muốn được đóng góp vào kho tàng đó từ rất lâu. Sau một khoảng thời gian phát triển và thử nghiệm nội bộ, tôi quyết định đã đến lúc mở dự án này để cộng đồng có thể cùng sử dụng và xây dựng.

Nếu bạn đã từng ngồi setup một CMS cho dự án frontend, chắc hẳn bạn biết cái cảm giác đó như thế nào. Ngồi cấu hình, thêm một đống plugin, rồi lại loay hoay với permissions, với migrations, với đủ thứ thứ lộn xộn khác. Đó là lý do team Tensoract quyết định giải quyết vấn đề này bằng cách xây dựng Tsquare, một Headless CMS tối giản.

Tsquare theo hướng API-first. Thay vì ép bạn phải dùng một frontend nhất định hay tuân theo một cấu trúc nào đó, Tsquare chỉ đơn giản là đưa JSON data đến bất kỳ đâu bạn cần, từ webapp tới mobile app hay cả các thiết bị IoT.

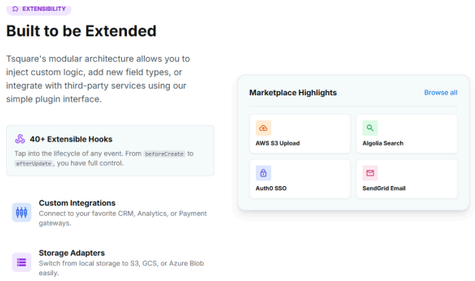
Minh hoạ Instant API Generation.

Một trong những tính năng khiến tôi thích nhất là Instant API Generation. Bạn định nghĩa schema, hệ thống tự động tạo ra REST API lẫn GraphQL endpoint đầy đủ CRUD mà bạn không cần viết thêm một dòng code nào. Swagger và OpenAPI documentation cũng được sinh ra tự động theo. Điều này nghe quen, nhưng Tsquare gọn nhẹ hơn nhiều so với những giải pháp tương tự mà tôi từng dùng. Điểm khác biệt nữa là Define Content Models at Runtime , tức là bạn có thể thay đổi cấu trúc dữ liệu ngay trong lúc hệ thống đang chạy mà không cần restart service hay chạy migration. Bạn cũng có thể lọc dữ liệu theo nhiều điều kiện lồng nhau, tìm kiếm full-text trên nhiều trường cùng lúc, sort theo bất kỳ thuộc tính nào. Tất cả đều qua URL query parameters đơn giản mà không cần phải viết custom endpoint hay dùng đến một query language phức tạp.



Bảo mật và phân quyền.

Bảo mật trong Tsquare được xây dựng theo dạng Fine-grained ACL, cho phép bạn kiểm soát quyền ở mức rất chi tiết. Bạn có thể cho phép một editor chỉnh sửa Blog Posts nhưng chỉ được xem Settings mà không được sửa. Bạn cũng có thể giới hạn quyền truy cập đến từng dòng dữ liệu dựa trên creator ID hay trạng thái của bản ghi.



Minh hoạ khả năng mở rộng.

Về mặt mở rộng, Tsquare cung cấp hơn 40 lifecycle hooks để bạn can thiệp vào bất kỳ sự kiện nào, từ beforeCreate đến afterUpdate. Bạn có thể tự do viết thêm tính năng, plugin. Sau này nếu tình hình ổn, tôi sẽ mở thêm tính năng Marketplace để bạn có thể bán các plugin đó.

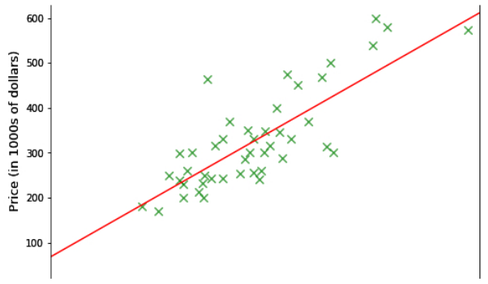
Tsquare được phát hành dưới giấy phép MIT, bạn hoàn toàn tự do sử dụng, tùy biến và triển khai. Dự án dự kiến chính thức công khai mã nguồn vào tháng 6 năm 2026 và hiện đang mở waitlist. Hy vọng là thông qua dự án này, tôi có thể truyền tải được thông điệp nghiêm túc với opensource và mang lại giá trị cho công đồng.

16/03/2026 • THỊNH VÕ QUỐC

Dự đoán tương lai với AI MiroFish

"Với một chữ nếu, người ta có thể nhét cả Paris vào một cái chai". Chúng ta hay dùng câu này để nói về những giả thuyết viển vông. Nhưng nhìn vào dự án MiroFish gần đây, tôi thấy cái chai này bắt đầu có thiệt.

Đối với sinh viên ngành Khoa học máy tính, việc AI có thể dự đoán không phải là chuyện quá xa lạ. Thậm chí bài tập nhập môn còn là bài dự đoán giá nhà. Mở rộng hơn là những bài dự đoán khác như: dự đoán thị trường chứng khoán... Nói nôm na thì những dự đoán này hoạt động bằng cách máy tính sẽ tìm ra trọng số cho một công thức nào đó, sau đó thay các giá trị thực tế vào rồi đưa ra con số dự đoán. Có thể có nhiều phương pháp khác nhau nhưng tổng quan là như vậy.

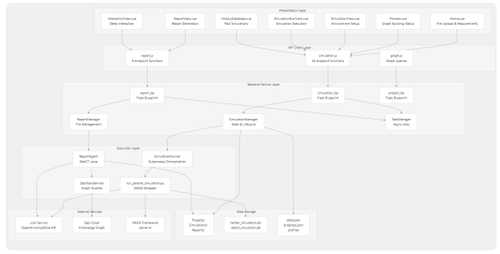


Nguồn hình

Tuy nhiên MiroFish không phải AI dự đoán kiểu truyền thống. Nó là một swarm intelligence. Thay vì dự đoán bằng những phương trình toán học. Microfish tạo ra hàng ngàn AI agents có tính cách, có ký ức và có cả định kiến riêng. Những agents này không đứng yên, chúng lao vào tranh luận, thuyết phục, lập phe phái và thay đổi quan điểm y như cách con người chúng ta vẫn làm mỗi khi có một tin tức mới hay một chính sách vừa ban hành.

Điểm bá đạo của MiroFish nằm ở chỗ nó bắt được những thứ mà các mô hình truyền thống thường bỏ qua: hiệu ứng đám đông. Một chính sách hay một chiến dịch marketing đôi khi thất bại vì hiệu ứng dây chuyền. MiroFish cho phép bạn đứng ở góc nhìn của Chúa - ý tôi là kiểu đứng ở vị trí trên cao, xem xét toàn bộ chuyển biến, đưa một biến số vào thế giới ảo này và quan sát xem các agents phản ứng ra sao.

Để làm được điều đó, nó sử dụng một dàn công nghệ khá khủng bên dưới. Đầu tiên là GraphRAG để xây dựng một đồ thị tri thức cực kỳ chi tiết. Sau đó, nó dùng Zep Cloud để tạo cho mỗi agent một bộ nhớ dài hạn, giúp chúng không bị ngáo mà biết rút kinh nghiệm từ những vòng mô phỏng trước. Toàn bộ kịch bản này chạy trên engine OASIS, có khả năng vận hành cả triệu agent tương tác cùng lúc trên những môi trường giả lập giống hệt Twitter hay Reddit.



Hình chụp từ DeepWiki của MiroFish.

Áp dụng vào thực tế, đây sẽ là một vũ khí chiến lược cho doanh nghiệp. Trước khi tung ra một chiến dịch lớn hay thay đổi giá sản phẩm, thay vì ngồi đoán già đoán non, bạn có thể ném nó vào sandbox của MiroFish để xem hàng ngàn khách hàng AI cãi lộn và phản ứng thế nào. Nếu tụi nó tẩy chay trong thế giới ảo, bạn vẫn còn kịp để sửa sai mà không mất một đồng nào ở đời thực.

Ở tầm vĩ mô hơn như quản lý đất nước, AI dự báo chính là cứu cánh cho những nhà hoạch định chính sách. Một dự thảo luật về thuế hay quy hoạch đô thị có thể được chạy thử để nhìn thấy trước những phản ứng dư luận hoặc các nguy cơ bất ổn. Nó giúp chính phủ bớt đi những quyết định gây sốc và tìm được tiếng nói chung với người dân thông qua hàng vạn lần mô phỏng trước khi ban hành chính thức.

Đặc biệt, tôi thấy một use case cực kỳ thực tế cho anh em startup là chuyện có nên gọi vốn lúc này hay không. Thường thì các founder hay đứng trước lựa chọn: Gọi ngay bây giờ để có tiền nhưng bị ép giá, hay chờ thêm 6 tháng để có traction tốt hơn nhưng lại đối mặt với rủi ro hết tiền? Với MiroFish, bạn có thể quăng cái pitch deck và dữ liệu thị trường vào. Nó sẽ giả lập hàng ngàn AI VCs và các AI Competitors để xem: nếu bạn gọi vốn lúc này, đối thủ sẽ phản ứng ra sao? Vòng gọi vốn của bạn có bị hố về định giá không? Thậm chí, nó có thể diễn tập kịch bản: bạn dành 3 tháng đi gọi vốn và bỏ bê sản phẩm thì thị trường sẽ như thế nào. Việc nhìn thấy trước kết quả của 1000 kịch bản gọi vốn khác nhau giúp founder không còn phải hên xui mà có thể chọn đúng thời điểm vàng để ký term sheet.

Cá nhân tôi cảm thấy cực kỳ phấn khích với công cụ này. Không chỉ là chuyện tiền bạc hay chính sách khô khan, điều khiến tôi hào hứng nhất là khả năng dùng MiroFish để test những giả lập lịch sử thú vị. Sẽ ra sao nếu một vị tướng trong quá khứ đưa ra quyết định khác vào phút chót? Hay một cuộc hành quân diễn ra theo một kịch bản trái ngược?

Sự xuất hiện của MiroFish đánh dấu một bước ngoặt quan trọng: AI đang đưa con người thoát khỏi sự bất định. Chúng ta không còn phải đặt cược vận mệnh vào may rủi hay trực giác mơ hồ, mà có thể trực tiếp quan sát và thử nghiệm mọi khả năng bên trong một sandbox. Khi mọi kịch bản đều được diễn tập kỹ lưỡng trước khi bắt đầu.

09/03/2026 • THỊNH VÕ QUỐC

Con hổ và luật rừng

Ngày xửa ngày xưa, trong một khu rừng rậm rạp có một bãi đất trống, nơi các loài thú nhỏ thường tụ tập để chơi đá banh da. Trong khu rừng đó, uy quyền nhất là một con Hổ. Nó không chỉ to lớn, sức dài vai rộng mà còn nổi tiếng bặm trợn, khiến muông thú vừa nhìn thấy đã bạt vía kinh hồn.

Một buổi chiều, khi đám thú đang loay hoay chia đội, Hổ bước ra. Nó nhìn trái banh da, rồi nhìn đám lợn rừng, hươu, nai đang run rẩy. Hổ dõng dạc tuyên bố:

– Để đảm bảo công bằng cho trò chơi, ta sẽ đứng ra làm trọng tài.

Nói rồi, nó lấy một cái còi bằng vỏ ốc đeo vào cổ. Đám thú nhỏ nhìn nhau, tuy không muốn nhưng đành gật đầu ưng thuận. Không ai dám lên tiếng vì Hổ giữ trái bóng. Vì Hổ có cái còi. Và vì Hổ có vuốt nhọn.

Thế nhưng, trận đấu chỉ vừa bắt đầu được một lát, khi tiếng còi vừa cất lên được vài lần, Hổ bỗng nảy ra ý định khác. Nó gầm lên một tiếng vang động cả khu rừng:

– Ta đổi ý rồi! Đứng ngoài xem thật tẻ nhạt. Từ giờ, ta sẽ vừa đá, vừa làm luôn trọng tài cho tiện!

Một lần nữa, muông thú lại cúi đầu vâng lời. Nỗi sợ hãi cái vuốt sắc của Hổ đã khóa chặt miệng chúng lại.

Cuộc chơi diễn ra đúng như kịch bản mà ai cũng có thể đoán trước. Mỗi khi Hổ dẫn banh, chỉ cần một chú hươu hay nai nào vô tình chạm nhẹ vào bộ lông vằn của nó, tiếng còi vỏ ốc lập tức vang lên dồn dập, báo hiệu lỗi phạt. Ngược lại, khi Hổ cậy sức lồng lộn húc ngã lợn rừng, giẫm lên chân thỏ để cướp banh thì cái còi trên cổ nó hoàn toàn câm lặng.

Đám thú còn lại đứng ngơ ngác trên sân, lòng đầy uất ức nhưng không đứa nào dám cự cãi nửa lời. Chúng thừa biết kẻ mạnh nhất đang nắm giữ luật lệ. Hổ nó có quyền đuổi bất cứ kẻ nào ra khỏi bãi đất nếu dám thắc mắc về luật chơi do chính nó đặt ra.

Nhìn thấy quyền lực tuyệt đối của Hổ, một vài kẻ khôn lanh như Cáo và Chồn bắt đầu nảy ra ý đồ xấu. Thay vì tập trung đá, tụi này chỉ lo chạy theo tung hô, che chắn và dọn đường cho Hổ ghi bàn. Kết thúc trận đấu, đội của Hổ dĩ nhiên thắng đậm, tỉ số chênh lệch đến mức nực cười.

Ngay khi tiếng còi mãn cuộc vang lên, lũ Cáo và Chồn vội vàng chạy đi hái những chùm quả mọng nhất, ngon nhất đem lại tận nơi đưa bằng hai tay, vừa dâng vừa cười nịnh bợ. Chúng hy vọng rằng sự cung phụng này sẽ khiến Hổ nương tay trong trận ngày mai, hoặc tiếp tục thổi còi có lợi cho phe chúng.

Tuy nhiên, một bộ phận những con thú khác như Nhím, Thỏ và Sóc bắt đầu lộ rõ sự bất mãn. Chúng nhận ra rằng dù có chạy mệt nhoài, có nỗ lực đến kiệt sức, kết quả của trò chơi cũng đã được định đoạt từ trước khi banh lăn. Những đôi chân vốn đầy nhiệt huyết dần chậm lại, rồi dừng hẳn. Không một lời cãi vã hay phản kháng, vài con thú lặng lẽ rời khỏi bãi đất khi trận đấu vẫn chưa kết thúc. Tụi nó hiểu rằng chẳng ích gì khi cố gắng trong một trò chơi mà kẻ cầm còi cũng là kẻ ghi bàn, và xung quanh là những kẻ sẵn sàng đánh đổi sự công bằng lấy vài quả mọng lo lót.

Trận đấu cứ thế thưa thớt dần. Những con thú ra đi không ngoảnh lại. Chúng tụ tập ở một góc rừng khác, yên tĩnh hơn, bắt đầu bàn nhau về việc tự làm một trái banh mới và tìm một bãi đất khác để chơi. Bãi đất cũ giờ chỉ còn lại Hổ và đám đệ tử xu nịnh đang loay hoay tự đá với nhau – một trận đấu tẻ nhạt.

05/03/2026 • THỊNH VÕ QUỐC

Chuyện dĩa cơm lên giá

Khi trở lại Sài Gòn sau Tết, tôi đã bị sốc khi nhìn bảng giá tại các quán cơm. Tôi sống ở quận 9, một nơi vốn được coi là có mức chi phí dễ thở hơn trung tâm thì giá một phần cơm bình dân đã đồng loạt tăng từ 35.000 VNĐ lên 40.000 VNĐ. Con số 5.000 VNĐ nghe có vẻ nhỏ, nhưng nếu đặt nó vào hệ quy chiếu của những người lao động hay sinh viên, chúng ta sẽ thấy một thực tế khắc nghiệt. Hiện nay, lương làm thêm của một sinh viên trung bình rơi vào khoảng 20.000 đến 22.000 VNĐ mỗi giờ, điều này đồng nghĩa với việc để có được một bữa cơm, một bạn trẻ phải đánh đổi bằng 2 giờ lao động.

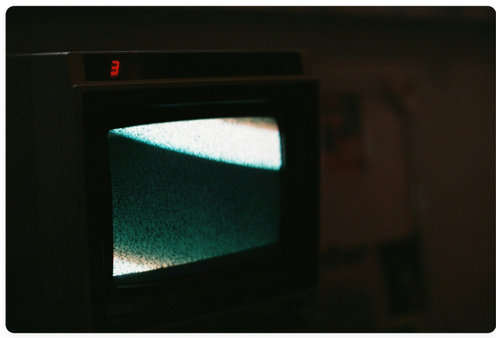
Việc tăng giá này là kết quả của một chuỗi áp lực tích tụ từ nhiều phía lên các hộ kinh doanh cá thể. Theo tôi thấy, nguyên nhân đầu tiên là sự chuyển đổi trong chính sách quản lý thuế. Việc chuyển từ thu thuế khoán cố định sang thu thuế dựa trên doanh thu đã buộc các chủ quán phải minh bạch hóa dòng tiền và đẩy phần nghĩa vụ thuế tăng thêm này vào giá thành sản phẩm để duy trì mức lợi nhuận. Bên cạnh đó, lạm phát vẫn âm thầm càn quét qua mặt hàng hàng thực phẩm và nhiên liệu. Giá xăng, dầu và đặc biệt là giá gas công nghiệp liên tục biến động theo chiều hướng tăng, kéo theo chuỗi cung ứng nguyên liệu từ các chợ đầu mối về đến các quán đều đội chi phí. Trong khi đó, một thực tại đáng buồn là mức lương cơ bản và thu nhập thực tế của đa số người lao động vẫn duy trì ở mức cũ, tạo thành một gọng kìm bóp nghẹt túi tiền của những người ở tầng lớp bình dân.

Trên các mặt báo và bản tin kinh tế, chúng ta vẫn thường thấy những con số lạc quan về tăng trưởng GDP, về chỉ số niềm tin tiêu dùng hay sự phục hồi mạnh mẽ của nền kinh tế sau đại dịch. Tuy nhiên, dường như có một khoảng cách xa vời giữa những biểu đồ tăng trưởng xanh màu trên báo chí và dĩa cơm tại các quán ven đường. Nghịch lý này cho thấy sự lệch pha, nơi mà các chỉ số vĩ mô tốt đẹp không đồng nghĩa với việc áp lực mưu sinh của người dân được giảm nhẹ.

Theo quan điểm của tôi, một nền kinh tế thực sự bền vững phải là một nền kinh tế mà ở đó người lao động có thể chi trả cho các nhu cầu cơ bản bằng giá cả hợp lý. Khi cái giá của một bữa cơm bình dân bắt đầu tương đương với 2 giờ làm việc, đó là dấu hiệu cho thấy sự mất cân bằng. Chúng ta không thể đơn thuần trách các chủ quán cơm, bởi bản thân họ cũng là những nạn nhân. Vấn đề nằm ở sự điều tiết vĩ mô và tính thực tế của các chính sách an sinh xã hội. Nếu không có những biện pháp kiểm soát lạm phát hiệu quả và các chính sách thiết thực, dĩa cơm 40.000 VNĐ hôm nay có thể chỉ là khởi đầu của một làn sóng bão giá mới, khiến khoảng cách giàu nghèo ngày càng nới rộng và cuộc sống của những người ở tầng dưới xã hội ngày càng trở nên chật vật hơn .

04/03/2026 • THỊNH VÕ QUỐC

FFmpeg, Meta và câu chuyện big corp hỗ trợ opensource



Tôi vừa đọc được một bài chia sẻ rất hay từ đội ngũ kỹ sư của Meta về cách họ vận hành FFmpeg ở quy mô lớn. Thực sự, khi viết những dòng này, tôi muốn dành một sự tri ân sâu sắc đến dự án FFmpeg. Tại Ecombox, chúng tôi cũng đang sử dụng FFmpeg để xử lý và encode toàn bộ video của hệ thống. Nếu không có bộ thư viện mã nguồn mở này, chi phí và thời gian để xây dựng một giải pháp xử lý media từ con số 0 sẽ là một rào cản không tưởng đối với những dự án như chúng tôi.

Câu chuyện của FFmpeg không chỉ là câu chuyện về kỹ thuật, mà còn là minh chứng rõ nét nhất cho mối quan hệ cộng sinh giữa các dự án mã nguồn mở và các tập đoàn công nghệ hàng đầu thế giới.

Tại Meta, họ đang vận hành một trong những hệ thống xử lý video phức tạp nhất hành tinh mang tên Media Processing Service (MPS) dựa trên nền tảng FFmpeg. Với Meta, FFmpeg giúp hàng tỷ video trên Facebook, Instagram và WhatsApp tiếp cận người dùng mỗi ngày.

Thay vì xây dựng một giải pháp đóng, Meta chọn FFmpeg với lý do rất đơn giản: FFmpeg có độ phủ lớn, hỗ trợ hầu hết các định dạng video và có một cộng đồng suốt hơn 20 năm. Tuy nhiên, ở quy mô của Meta, việc chỉ sử dụng bản gốc là chưa đủ. Họ đã thực hiện những cải tiến kỹ thuật mà từ đó, toàn bộ cộng đồng được hưởng lợi.

- **Tối ưu hóa hiệu suất AV1:** Họ đã đóng góp các đoạn mã tối ưu hóa giúp FFmpeg xử lý AV1 nhanh hơn, hiệu quả hơn, giúp giảm băng thông internet mà vẫn giữ được chất lượng video cao.
- **Tích hợp tăng tốc phần cứng (Hardware Acceleration):** Để xử lý video ở quy mô tỷ lượt xem, Meta không thể chỉ dùng CPU. Họ đã viết lại để tương thích với các chip chuyên dụng.
- **Vá lỗi Edge Cases:** Với khối lượng dữ liệu khổng lồ, Meta phát hiện ra những lỗi cực kỳ hiếm gặp liên quan đến metadata hoặc bitstream filters mà một người dùng thông thường có lẽ cả đời không gặp phải. Các bản vá này sau khi được Meta đóng góp đã giúp FFmpeg trở nên ổn định và bảo mật hơn rất nhiều cho tất cả chúng ta.

Tôi viết bài này trước hết là để cảm ơn đội ngũ phát triển FFmpeg vì một công cụ quá tuyệt vời. Thứ hai là để ghi nhận nỗ lực của những kỹ sư tại các tập đoàn lớn như Meta.

Nhìn từ câu chuyện cộng sinh giữa Meta và FFmpeg, tôi không khỏi suy nghĩ về thực trạng tại thị trường Việt Nam. Hiện nay, chúng ta đã có những tập đoàn công nghệ lớn với tiềm lực tài chính và đội ngũ kỹ sư không hề thua kém. Phần lớn các dự án tại Việt Nam cũng đang chạy dựa trên nền tảng của các dự án opensource. Tuy nhiên, chúng ta vẫn đang dừng lại ở vai trò là những người sử dụng tận tụy hơn là những người đóng góp.

Tôi hy vọng rằng trong tương lai gần, các công ty lớn tại Việt Nam sẽ sớm có được tinh thần này. Việc đóng góp không chỉ đơn thuần là hành động tử tế, mà là một chiến lược khẳng định vị thế công nghệ. Khi một doanh nghiệp Việt Nam đóng góp được những bản vá lỗi quan trọng hay tối ưu hóa được cho những dự án như FFmpeg, Kubernetes hay Linux, đó là lúc chúng ta thực sự ghi tên mình lên bản đồ di sản công nghệ thế giới.

03/03/2026 • THỊNH VÕ QUỐC

AI đập đi xây lại Next.js

Tôi vừa theo nghiên cứu thông tin của Cloudflare với dự án Vinext, và thấy rằng chúng ta đang bước sang một chương mới. AI giờ đây không chỉ dừng lại ở việc code mấy cái app hay website đơn giản, nó đang trực tiếp đập đi xây lại cả một framework. Bài này chia sẻ góc nhìn của tôi về việc AI giờ bá đạo như thế nào, còn chi tiết sâu về kĩ thuật, những điểm nổi bật của Vinext và các thông số tôi nghĩ bạn nên đọc trực tiếp bài dưới đây.

> *One engineer used AI to rebuild Next.js on Vite in a week. vinext builds up to 4x faster, produces 57% smaller bundles, and deploys to Cloudflare Workers with a single command.*
> ***The Cloudflare Blog — Steve Faulkner.***

Đi sâu vào kỹ thuật, Next.js được xây dựng dựa trên môi trường thực thi là Node.js, cho phép nó truy cập sâu vào các native modules như `fs` để truy cập hệ thống tệp, `path` để xử lý đường dẫn, hay module `crypto` để mã hóa. Tuy nhiên, khi đưa lên hạ tầng Edge của Cloudflare, nó lại trở thành rào cản vì V8 Isolates chỉ hỗ trợ các chuẩn Web APIs tinh gọn để đảm bảo tốc độ phản hồi nhanh. Trước đây, các giải pháp chạy Next.js trên Cloudflare thường phải sử dụng một lớp giả lập Node.js (polyfills), điều này vô tình tạo ra một gánh nặng tài nguyên, khiến kích thước gói tin bị phình to và thời gian khởi động (Cold Start) bị kéo dài đáng kể.

Cloudflare đã thay đổi cuộc chơi bằng cách dùng AI để viết ra Vinext theo ngữ cảnh (contextual rewriting). AI thực hiện các nhiệm vụ phức tạp: chuyển đổi toàn bộ hệ thống streams và buffers sang chuẩn Web Streams và Uint8Array, đồng thời viết lại các module bằng Web Crypto API. Đặc biệt, khi kết hợp với Rolldown (bundler dựa trên Rust sắp có trong Vite 8), tốc độ build production của Vinext nhanh hơn tới 4 lần và dung lượng client bundle giảm đến 57%. Điều này giúp loại bỏ hoàn toàn các lớp trung gian, cho phép Next.js chạy Native trên Workers.

Một điểm đột phá mà tôi thấy cực kỳ thông minh là Traffic-aware Pre-Rendering (TPR). Thông thường, Next.js sẽ phải render hàng ngàn trang tĩnh lúc build (SSG), làm thời gian build kéo dài. Vinext dùng AI và dữ liệu traffic thực tế từ Cloudflare để chỉ pre-render những trang thực sự có người xem (ví dụ: 200 trang chiếm 90% traffic). Những trang còn lại sẽ được render theo yêu cầu và cache lại sau.

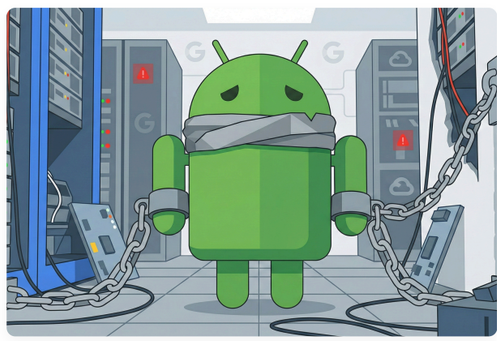
Cách tiếp cận này cho thấy Cloudflare đang thực hiện một chiến lược: thay vì chờ đợi sự hỗ trợ từ tác giả framework, họ dùng AI để làm lại framework đó sao cho phù hợp nhất với hạ tầng của mình, trực tiếp cạnh tranh với lợi thế độc quyền của Vercel ở mảng deployment.

Đối với cộng đồng lập trình viên, sự xuất hiện của Vinext là một lời cảnh tỉnh về việc cập nhật kỹ năng trong thời đại mới. Việc dự án này được hoàn thiện chỉ bởi một kỹ sư và AI trong vòng đúng 1 tuần (với chi phí token chỉ $1,100) là minh chứng cho thấy AI đã đủ sức đảm nhận vai trò kiến trúc sư hệ thống. Điều này đòi hỏi lập trình viên phải chủ động chuyển tư duy từ việc viết code đơn thuần sang việc thiết kế kiến trúc. Khi AI đã giải phóng chúng ta khỏi code chay, giá trị cốt lõi của người làm kỹ thuật sẽ nằm ở khả năng giám sát, kiểm định và tư duy sáng tạo trong việc thiết kế những giải pháp đột phá. Khả năng hiểu rõ cách V8 Isolate hay cách Edge Runtime tối ưu hóa sẽ trở nên quan trọng hơn việc thuộc lòng các API của framework.

Sức mạnh của AI trong thời kỳ này không chỉ giúp con người làm việc nhanh hơn, mà là giúp chúng ta thực hiện được những điều trước đây vốn được coi là không khả thi do giới hạn về nguồn lực và thời gian. Vinext là một lời khẳng định rằng ranh giới giữa các nền tảng công nghệ đang dần bị xóa nhòa, mang lại sự tự do tối đa cho dev. Tương lai của phần mềm sẽ không còn bị giới hạn bởi framework, mà sẽ được định nghĩa bởi sự sáng tạo của con người.

02/03/2026 • THỊNH VÕ QUỐC

Sự xói mòn tính mở của hệ điều hành Android và rủi ro từ sự kiểm soát tập trung



Là một người luôn ủng hộ và yêu thích các giải pháp mã nguồn mở, tôi đã từng chia sẻ quan điểm

của mình qua bài viết The Open Nation. Tuy nhiên, những diễn biến mới nhất xoay quanh hệ sinh thái Android khiến tôi phải viết bài này. Đây là bài viết với phong cách bài tiểu luận, khác với những bài chia sẻ tôi hay viết.

Trong bài viết, tôi có tham khảo các nguồn dưới đây và có sử dụng Gemini để tạo hình minh hoạ cho bài viết.

Kể từ khi ra đời, hệ điều hành Android luôn được định vị là biểu tượng của tính mở và sự tự do, đối lập hoàn toàn với mô hình walled garden của Apple. Tính mở này không chỉ là một đặc điểm kỹ thuật mà còn là nền tảng cho sự đổi mới, cho phép người dùng toàn quyền kiểm soát thiết bị và các nhà phát triển tự do phân phối phần mềm mà không cần sự phê duyệt từ một thực thể trung tâm. Tuy nhiên, những động thái gần đây của Google thông qua "Chương trình xác minh nhà phát triển" (Android developer verification program) và các rào cản kỹ thuật mới đang báo hiệu một sự chuyển dịch đáng lo ngại. Việc thắt chặt kiểm soát này không chỉ tiêu diệt hệ sinh thái mã nguồn mở mà còn tạo ra những công cụ đắc lực cho các chính phủ độc tài trong việc kiểm soát thông tin và trấn áp quyền tự do cá nhân.

Sự chuyển dịch từ hệ sinh thái mở sang mô hình kiểm soát tập trung

Theo các tài liệu từ chiến dịch *Keep Android Open* và các phân tích kỹ thuật, Google đang tiến tới việc yêu cầu mọi nhà phát triển ứng dụng phải đăng ký định danh chính thức, cung cấp giấy tờ tùy thân do chính phủ cấp và thanh toán phí duy trì để có thể cài đặt ứng dụng trên các thiết bị Android đã được chứng nhận (certified devices). Đáng chú ý, quy định này không chỉ áp dụng cho các ứng dụng trên Google Play Store mà còn bao trùm cả những phần mềm được phân phối độc lập qua các nền tảng như GitHub hay các kho ứng dụng mã nguồn mở như F-Droid.

Việc bắt buộc định danh và xác minh signing key tập trung tại Google đã phá vỡ nguyên tắc ẩn danh và quyền tự quyết của các nhà phát triển độc lập. Đối với cộng đồng mã nguồn mở, đây là một án tử về mặt kỹ thuật. Các nền tảng như F-Droid vốn hoạt động dựa trên triết lý bảo vệ quyền riêng tư và không thu thập dữ liệu cá nhân của nhà phát triển sẽ không thể tồn tại nếu quy định này được thực thi triệt để vào năm 2026. Khi đó, việc cài đặt các tệp tin APK từ bên ngoài (sideloading) sẽ trở nên vô cùng khó khăn hoặc bị ngăn chặn hoàn toàn dưới chiêu bài bảo mật.

Ngụy biện về an ninh và sự đánh đổi quyền riêng tư

Google thường lập luận rằng việc thắt chặt kiểm soát là cần thiết để bảo vệ người dùng khỏi mã độc và các hành vi lừa đảo. Tuy nhiên, lập luận này thiếu tính thuyết phục khi nhìn vào thực tế bảo mật đa tầng hiện nay trên hệ sinh thái Android. Hệ thống Play Protect vốn đã có khả năng quét và ngăn chặn mã độc trên mọi thiết bị Android mà không cần phải triệt tiêu quyền phân phối ứng dụng tự do. Quan trọng hơn, các nhà sản xuất thiết bị gốc (OEM) từ lâu đã tự phát triển các giải pháp bảo mật độc lập và vô cùng kiên cố ở cấp độ phần cứng.

Điển hình như Samsung với nền tảng Knox hay các cơ chế bảo mật tích hợp của Xiaomi, các giải pháp này thiết lập những lớp phòng thủ từ cấp độ nhân (kernel) đến lớp ứng dụng, cho phép cô lập các tiến trình độc hại và bảo vệ tính toàn vẹn của dữ liệu người dùng một cách hiệu quả. Sự hiện diện của các nền tảng bảo mật nội tại từ phía nhà sản xuất thiết bị đã chứng minh rằng người dùng hoàn toàn có thể được bảo vệ mà không cần đến một cơ chế phê duyệt tập trung từ Google.

Việc Google cố tình bỏ qua hiệu quả của các lớp bảo mật sẵn có này để áp đặt "Chương trình xác minh nhà phát triển" chỉ càng khẳng định rằng mục tiêu thực sự không nằm ở an ninh. Đây là nỗ lực nhằm củng cố vị thế độc quyền, buộc người dùng và nhà phát triển phải phục tùng các điều khoản kinh tế và chính sách dữ liệu của họ. Sự an toàn thực sự không đến từ việc tước bỏ quyền lựa chọn của người dùng, mà đến từ một kiến trúc bảo mật minh bạch, phi tập trung và sự phối hợp giữa các tầng thực thi phần cứng của nhà sản xuất. Khi Google biến mình thành người gác cổng duy nhất, họ không làm cho hệ thống an toàn hơn mà chỉ đang làm cho quyền tự do cá nhân trở nên mỏng manh hơn trước các quyết định áp đặt từ một thực thể đơn lẻ.

Sự triệt tiêu động lực đổi mới và di sản đóng góp từ cộng đồng

Một trong những hệ lụy nghiêm trọng nhất của chính sách thắt chặt này chính là việc chấm dứt hành trình phát triển dựa trên sự cộng tác: yếu tố cốt lõi đã làm nên sự thành công vượt trội của Android so với các hệ điều hành đóng. Lịch sử của Android không đơn thuần được viết bởi các kỹ sư của Google, mà còn được vun đắp bởi hàng triệu nhà phát triển độc lập, các cộng đồng mã nguồn mở và những người đam mê công nghệ trên toàn cầu. Chính sự tự do trong việc tùy chỉnh và phân phối phần mềm đã biến Android từ một dự án non trẻ trở thành hệ điều hành phổ biến nhất thế giới.

Việc áp đặt quy trình xác minh danh tính và thu phí bắt buộc đã dựng lên một rào cản tài chính và pháp lý đối với các nhà phát triển cá nhân, những người làm vì sở thích hoặc những tổ chức phi lợi nhuận. Những đóng góp mang tính đột phá thường đến từ các thử nghiệm nhỏ lẻ trên những nền tảng tự do như F-Droid hay GitHub nơi mà sự sáng tạo không bị kìm kẹp bởi các quy tắc thương mại của một tập đoàn trung tâm. Khi Google độc quyền hóa quyền cho phép cài đặt ứng dụng, họ đang trực tiếp triệt tiêu không gian thử nghiệm này.

Hơn thế nữa, động thái này còn được coi là một sự phản bội đối với triết lý mã nguồn mở mà Google từng cam kết. Android đã tận dụng di sản trí thức từ cộng đồng để hoàn thiện hệ thống, nhưng khi đạt đến vị thế thống trị, tập đoàn này lại chọn cách đóng cửa để bảo vệ lợi ích kinh doanh. Việc biến Android thành một phiên bản tương tự iOS: nơi mọi hoạt động phân phối phần mềm đều phải thông qua sự kiểm soát và phê duyệt của một thực thể duy nhất sẽ làm thui chột tính đa dạng sinh học của hệ sinh thái phần mềm. Điều này không chỉ làm giảm đi khả năng cạnh tranh mà còn khiến Android mất đi bản sắc vốn có, chuyển đổi từ một nền tảng mở của nhân loại thành một sản phẩm thương mại đóng kín, nơi sự đổi mới chỉ được phép diễn ra trong khuôn khổ mà Google cho phép.

Hành trình phát triển của Android nhờ đó sẽ chuyển dịch từ một quỹ đạo tiến hóa dựa trên trí tuệ tập thể sang một mô hình tăng trưởng áp đặt từ trên xuống, nơi người dùng và nhà phát triển chỉ còn là những thực thể thụ động trong một hệ sinh thái bị kiểm soát chặt chẽ.

Sự xói mòn quyền sở hữu và chủ quyền kỹ thuật số của người dùng

Việc Google thắt chặt kiểm soát phần mềm trên Android đã trực tiếp dẫn đến sự suy giảm quyền sở hữu thực sự của người dùng đối với thiết bị cá nhân. Trong kinh tế học và pháp lý truyền thống, khi một cá nhân chi trả để sở hữu một công cụ phần cứng, họ mặc nhiên có quyền định đoạt tuyệt đối đối với những gì vận hành trên đó. Tuy nhiên, bằng cách thiết lập các rào cản xác minh nhà phát triển từ xa, Google đang chuyển dịch mô hình từ "sở hữu thiết bị" sang một dạng "thuê mướn quyền sử dụng" có điều kiện.

Chủ quyền kỹ thuật số của cá nhân bị xâm phạm khi một thực thể tập trung có quyền đơn phương thu hồi khả năng cài đặt hoặc vận hành một ứng dụng. Người dùng không còn là chủ thể quản lý thiết bị, mà trở thành đối tượng chịu sự điều tiết của các thuật toán và chính sách tập đoàn. Điều này tạo ra một tiền lệ nguy hiểm: sự tự do cá nhân trong không gian số bị đặt dưới sự giám sát và cho phép của bên thứ ba, làm mất đi tính độc lập vốn là giá trị cốt lõi của người tiêu dùng trong kỷ nguyên công nghệ.

Rào cản kinh tế và sự phân biệt đối xử đối với các nhà phát triển tại các quốc gia đang phát triển

Chính sách bắt buộc đóng phí và cung cấp định danh chính thức của Google vô hình trung tạo ra một rào cản hành chính và tài chính đáng kể, đặc biệt đối với các khu vực có thu nhập thấp hoặc các

quốc gia đang phát triển. Việc yêu cầu giấy tờ tùy thân chuẩn quốc tế và các phương thức thanh toán xuyên biên giới không phải lúc nào cũng khả thi đối với mọi nhà lập trình độc lập. Điều này dẫn đến một sự phân loại nhà phát triển dựa trên nguồn lực kinh tế thay vì năng lực sáng tạo.

Những tài năng trẻ, sinh viên, hoặc các tổ chức phi lợi nhuận tại các khu vực nghèo khó sẽ bị gạt ra khỏi hệ sinh thái phát triển ứng dụng ngay từ bước đầu tiên. Hành động này không chỉ kìm hãm sự đổi mới mang tính bản địa mà còn củng cố vị thế độc quyền của các tổ chức có pháp nhân và tài chính vững mạnh từ các quốc gia phát triển. Hệ quả là một sự bất bình đẳng kỹ thuật số sâu sắc, nơi tri thức và sự sáng tạo bị giới hạn bởi khả năng chi trả và các thủ tục hành chính phức tạp, làm mất đi tính đa dạng và toàn cầu mà Android từng cam kết hướng tới.

Hệ lụy đối với vòng đời thiết bị và mục tiêu phát triển bền vững

Trên phương diện môi trường, việc đóng kín hệ sinh thái phần mềm sẽ trực tiếp gây ra những hệ lụy tiêu cực đối với mục tiêu phát triển bền vững. Từ lâu, cộng đồng Android đã duy trì và kéo dài vòng đời của các thiết bị cũ: vốn không còn được nhà sản xuất hỗ trợ cập nhật chính thức thông qua các bản ROM tùy chỉnh hoặc các ứng dụng tối ưu hóa từ các nguồn độc lập. Đây là một hình thức tái sinh phần cứng, giúp giảm thiểu đáng kể lượng rác thải điện tử thải ra môi trường mỗi năm.

Khi Google ngăn chặn việc cài đặt phần mềm không qua xác minh tập trung, những thiết bị cũ sẽ nhanh chóng trở nên vô dụng khi không thể tiếp cận các ứng dụng mới do rào cản kỹ thuật. Hành động này vô hình trung thúc đẩy chu trình lỗi thời có kế hoạch (planned obsolescence), ép buộc người dùng phải đào thải thiết bị vẫn còn khả năng hoạt động tốt để mua sắm phần cứng mới. Điều này đi ngược lại hoàn toàn với xu hướng toàn cầu về quyền tự sửa chữa (Right to Repair) và nỗ lực bảo vệ môi trường, biến các tập đoàn công nghệ thành tác nhân gia tăng áp lực lên tài nguyên thiên nhiên và hệ sinh thái toàn cầu.

Nguy cơ kiểm soát thông tin tại các chính phủ độc tài

Điểm nguy hiểm nhất trong chiến lược đóng cửa Android chính là việc tạo ra một điểm thắt nút (choke point) khổng lồ cho sự kiểm duyệt. Khi mọi ứng dụng và nhà phát triển đều phải qua sự xác minh tập trung của Google, các chính phủ độc tài sẽ có một mục tiêu duy nhất và hiệu quả để gây áp lực.

Thứ nhất, việc gắn chặt danh tính nhà phát triển với các ứng dụng khiến các cá nhân viết phần mềm chống kiểm duyệt, VPN hoặc các ứng dụng tin nhắn mã hóa trở nên dễ bị tổn thương trước sự truy quét của chính quyền. Các chính phủ có thể yêu cầu Google cung cấp thông tin định danh của các nhà phát triển này hoặc ép buộc họ phải gỡ bỏ các tính năng bảo mật dưới danh nghĩa tuân thủ luật pháp địa phương.

Thứ hai, khi quyền cài đặt ứng dụng bị kiểm soát bởi một thực thể trung tâm, các chính phủ độc tài có thể cưỡng chế Google phải chặn các ứng dụng truyền thông độc lập hoặc các công cụ liên lạc của các tổ chức dân sự. Trong một hệ sinh thái mở, người dùng có thể dễ dàng tìm đến các nguồn phân phối thay thế để vượt qua sự kiểm duyệt. Nhưng trong một hệ sinh thái đóng, nơi thiết bị chỉ chấp nhận phần mềm từ "nhà phát triển được xác minh" bởi Google, sự kiểm duyệt sẽ trở nên tuyệt đối. Điều này biến chiếc điện thoại thông minh từ công cụ giải phóng thông tin trở thành một thiết bị giám sát và kiểm soát được nhà nước bảo trợ.

Kết luận

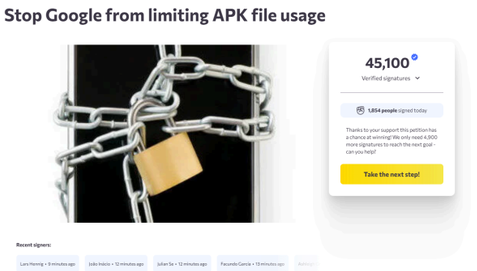
Việc đóng cửa hệ sinh thái Android là một bước lùi nghiêm trọng đối với quyền tự do kỹ thuật số toàn cầu. Nó không chỉ phá hủy thành quả của cộng đồng mã nguồn mở mà còn tạo tiền lệ nguy hiểm về việc tập trung quyền lực vào tay một tập đoàn duy nhất. Đối với người dân sống dưới các chế độ độc tài, tính mở của Android không chỉ là một tính năng kỹ thuật, mà là một phương tiện để tiếp cận thông tin trung thực và duy trì sự an toàn cá nhân. Việc Google từ bỏ nguyên tắc mở của Android không chỉ là một quyết định kinh doanh sai lầm mà còn là một hành động thiếu trách nhiệm đối với quyền con người trong kỷ nguyên số. Để bảo vệ tương lai của một internet tự do, cộng đồng quốc tế và các cơ quan quản lý cần phải ngăn chặn xu hướng độc quyền này, duy trì quyền được cài đặt phần mềm tự do của người dùng trên chính thiết bị mà họ sở hữu.

Trước ngưỡng cửa của những thay đổi mang tính bước ngoặt dự kiến thực thi vào năm 2026, chúng ta đang đứng trước một lựa chọn mang tính quyết định: chấp nhận sự kiểm soát áp đặt hay kiên trì bảo vệ quyền tự do kỹ thuật số. Lịch sử công nghệ đã chứng minh rằng những thay đổi tích cực không bao giờ đến từ sự im lặng, mà từ nỗ lực tập thể của cộng đồng những người sử dụng và phát triển có trách nhiệm. Việc phản đối kế hoạch đóng cửa Android của Google không chỉ là bảo vệ một hệ điều hành, mà là bảo vệ quyền tự sở hữu, tính đa dạng của tri thức và sự an toàn của những cá nhân yếu thế trước sự kiểm soát thông tin toàn cầu.

Tôi kêu gọi mỗi độc giả, những người trân trọng giá trị của mã nguồn mở và quyền tự quyết công nghệ, hãy cùng đồng hành trong chiến dịch kiến nghị này. Chúng ta cần sự hiện diện của tiếng nói cộng đồng để gửi đi một thông điệp đanh thép tới Google và các cơ quan quản lý viễn thông.

Quý vị có thể tham gia đóng góp thông qua các phương thức cụ thể sau:

1. **Tham gia ký tên vào bản kiến nghị chung** tại nền tảng KeepAndroidOpen.org, nơi quy tụ sự ủng hộ của hơn 40 tổ chức quốc tế vì quyền kỹ thuật số.
2. **Lan tỏa thông tin và phản biện các lập luận thiếu căn cứ về an ninh**, nhằm giúp cộng đồng hiểu rõ bản chất của sự đánh đổi quyền riêng tư mà Google đang thực hiện.

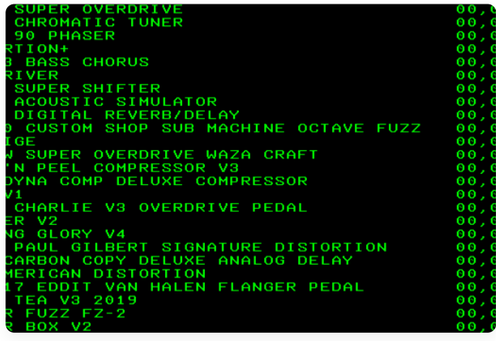


Tôi ký tên phải dùng VPN để ký vì trang này bị chặn ở Việt Nam. hãy tưởng tượng bạn không còn VPN để sử dụng nữa thì sao ?

Phản kháng ôn hòa là cách duy nhất mà người dùng sở hữu để đối trọng với sức mạnh của các tập đoàn xuyên quốc gia. Hãy cùng nhau hành động ngay hôm nay để đảm bảo rằng Android vẫn sẽ là một nền tảng của sự tự do, minh bạch và là tài sản chung của nhân loại, thay vì trở thành một công cụ kiểm soát khép kín. Tương lai của một thế giới kỹ thuật số mở đang nằm trong tay của chính chúng ta.

01/03/2026 • THỊNH VÕ QUỐC

Câu chuyện của AI và COBOL



Hồi Tết tôi có lướt thấy tin từ Anthropic về việc mô hình Claude của họ bắt đầu tham gia sâu vào việc bảo trì và hiện đại hóa mã nguồn COBOL. Đây là một tin tức thực sự quan trọng đối với giới công nghệ, bởi COBOL là hệ thống huyết mạch đang vận hành nền tài chính toàn cầu. Tôi đã định viết bài

này sớm hơn nhưng kẹt ăn tết. Nếu bạn chưa đọc qua bài đó, có thể đọc nó tại đây:

Với nhiều anh em dev, COBOL nghe như một cái tên từ thời tiền sử, nhưng thực tế nó lại đang gánh vác cả nền tài chính toàn cầu. Dù đã hơn 60 tuổi, nhưng khoảng 95% các lần bạn quẹt thẻ ATM hay các giao dịch ngân hàng (core-banking) vẫn đang chạy trên những dòng code này. Ước tính có tới 800 tỷ dòng code COBOL đang vận hành. Nếu COBOL ngưng chạy, thế giới sẽ không thể rút tiền hay thanh toán bất cứ thứ gì.

Độ phức tạp của COBOL nằm ở chỗ nó cực kỳ cứng nhắc và thiếu cấu trúc hiện đại (ví dụ như function, object). Một ví dụ điển hình là quy tắc 80 cột. Bạn chỉ cần gõ lùi vào một dấu cách hay viết quá cột thứ 72 là chương trình lỗi. Kinh khủng hơn, COBOL không có các module hay thư viện tách biệt như bây giờ. Một thay đổi nhỏ ở dòng code số 1.000 có thể làm sập một tính năng ở dòng số 1.000.000. Đó là lý do tại sao các dự án hiện đại hóa COBOL bằng sức người thường có tỉ lệ thất bại cao

Để ví dụ cho bạn sự khủng khiếp của code COBOL, tôi sẽ ví dụ về tính toán số dư tài khoản kèm lãi suất (code này tôi kêu Claude viết chứ bản thân tôi cũng không biết COBOL)

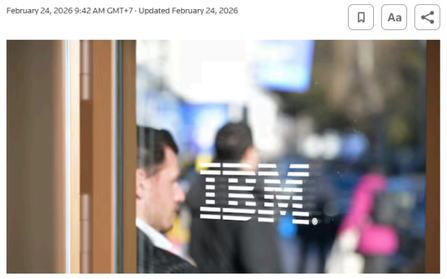
```
IDENTIFICATION DIVISION.
       PROGRAM-ID. TINH-LAI-SUAT.

       DATA DIVISION.
       WORKING-STORAGE SECTION.
       01  KHACH-HANG.
           05 HO-TEN           PIC X(30) VALUE 'NGUYEN VAN A'.
           05 SO-DU            PIC 9(07)V99 VALUE 1500.50.
           05 LAI-SUAT         PIC 9(02)V99 VALUE 05.25.
       01  KET-QUA.
           05 TIEN-LAI         PIC 9(07)V99.
           05 TONG-CONG        PIC 9(07)V99.
       01  BIEN-DEM            PIC 9(02) VALUE 01.

       PROCEDURE DIVISION.
       BEGIN-CALC.
           DISPLAY "KHACH HANG: " HO-TEN.
           PERFORM CALCULATE-LOGIC 12 TIMES.

           STOP RUN.

       CALCULATE-LOGIC.
           COMPUTE TIEN-LAI = SO-DU * (LAI-SUAT / 100 / 12).
           ADD TIEN-LAI TO SO-DU.
           DISPLAY "THANG " BIEN-DEM " - SO DU: " SO-DU.
           ADD 1 TO BIEN-DEM.
```

Đoạn này nếu code trong Python tôi nghĩ nó sẽ tầm 5 dòng thôi. Bao nhiêu đây bạn cũng đã có thể hiểu được vì sao rất ít Dev hiện nay biết COBOL, nói gì tới maintain hệ thống COBOL.

Thực tế, quyền kiểm soát di sản khổng lồ này đang nằm trong tay hai thế lực chính. Một bên là các nhà sản xuất phần cứng như IBM đơn vị cung cấp các máy chủ Mainframe, môi trường duy nhất mà COBOL có thể vận hành ổn định. Bên còn lại là các đơn vị cung cấp dịch vụ outsource lớn như Infosys hay tại Việt Nam là FPT. Đội ngũ này sở hữu những chuyên gia COBOL lâu năm, những người hiểu rõ từng chi tiết của hệ thống. Tuy nhiên, thách thức lớn nhất hiện nay là lực lượng nhân sự này đang dần nghỉ hưu, trong khi các lập trình viên trẻ hầu như không còn mặn mà với việc học một ngôn ngữ có cấu trúc lỗi thời. Điều này dẫn đến một cuộc khủng hoảng nhân sự, buộc các ngân hàng phải chi trả mức lương rất cao để duy trì những chuyên gia hiếm hoi còn sót lại.

Nhiều người đặt câu hỏi tại sao không chuyển đổi toàn bộ sang Java hay Python. Vấn đề nằm ở độ phức tạp cực kỳ lớn của COBOL. Đây là ngôn ngữ thiếu các cấu trúc hiện đại, không có module tách biệt thiếu tài liệu hướng dẫn đi kèm. Mọi logic nghiệp vụ được bồi đắp qua hàng thập kỷ đan xen vào nhau chặt chẽ, dẫn đến việc chỉ cần một thay đổi nhỏ cũng có thể gây ra lỗi hệ thống nghiêm trọng. Minh chứng điển hình là Ngân hàng Commonwealth Úc (CBA) đã phải mất 5 năm và chi ra hơn 1 tỷ USD chỉ để chuyển đổi hệ thống từ COBOL sang nền tảng mới.

Đây là lúc AI như Claude thể hiện vai trò của mình. Với khả năng xử lý ngữ cảnh lớn, AI có thể phân tích toàn bộ cấu trúc hệ thống, tự động hiểu các quy tắc nghiệp vụ ngầm định và chuyển đổi chúng sang kiến trúc hiện đại. Theo báo cáo của Anthropic, AI giúp giảm tới 80% thời gian và chi phí (theo giả) đoạn phân tích hệ thống. Điều này trực tiếp đe dọa đến mô hình kinh doanh thâm dụng nhân công của các công ty outsource lớn. Lâu nay, các dự án bảo trì hệ thống cũ kéo dài nhiều năm là nguồn thu ổn định nhờ tính phí theo giờ công của lập trình viên. Khi AI có thể xử lý công việc đó trong vài tuần, luật chơi sẽ thay đổi hoàn toàn, buộc các doanh nghiệp như IBM hay FPT phải thay đổi chiến lược để không bị đào thải. Những biến động đó đã xuất hiện ngay lập tức và thể hiện qua giá trị cổ phiếu của IBM và FPT.



- Phía FPT thì bị khối ngoại bán tháo cổ phiếu.
- Cổ phiếu IBM giảm mạnh nhất trong một ngày kể từ năm 2000.

Đối với những dev như tôi, sự phát triển này mang lại cả sự lo lắng lẫn sự phấn khích. Lo lắng vì những kỹ năng mà chúng ta nghĩ là khó thay thế đang dần bị AI chinh phục. Nhưng thú vị là ở chỗ, chúng ta đang chứng kiến AI giải quyết những bài toán hóc búa nhất. Thay vì chỉ viết mã, vai trò của lập trình viên sẽ chuyển dịch sang việc giám sát và điều phối AI để quản trị những hệ thống phức tạp hơn. Việc thích nghi với sự thay đổi này không còn là lựa chọn, mà là yêu cầu bắt buộc để tồn tại trong thời đại công nghệ mới.

Cảm ơn bạn đã đọc

Chào mừng bạn đến với blog của mình! Mình là Võ Quốc Thịnh, coder trẻ và sáng lập của TENSORACT CO.,LTD – một startup công nghệ ra đời từ năm 2022.

Trước đây, viết lách không phải thế mạnh của mình, nên mình cũng khá ngại chia sẻ. Nhưng giờ có AI hỗ trợ, mình tự tin hơn trong việc viết và truyền tải câu chuyện của mình. Là một founder, mình nhận ra rằng nói và viết về công ty, sản phẩm không chỉ giúp mọi người hiểu hơn về những gì mình đang làm, mà còn là một cách marketing miễn phí đầy hiệu quả. Quan trọng hơn, nó còn giúp mình hệ thống lại suy nghĩ, chia sẻ kinh nghiệm, và kết nối với nhiều người có cùng đam mê.

Nhiều người chọn TikTok để kể câu chuyện của mình. Mình thì chọn blog. Vì mình thích sự chậm rãi của chữ nghĩa nơi mình có thể suy ngẫm, diễn đạt sâu sắc hơn, thay vì phải chạy theo nhịp điệu nhanh và xu hướng của mạng xã hội. Viết cũng giúp mình rèn luyện tư duy rõ ràng, điều mà mình tin là cực kỳ quan trọng với một người làm công nghệ.

Mình sẽ viết về hành trình xây dựng startup, những bài học rút ra, công nghệ mình đang làm, và cả những góc nhìn cá nhân về ngành.

Cảm ơn bạn đã đọc ấn phẩm này, hy vọng bạn sẽ tìm thấy những điều thú vị!