

CONVERSACIÓN

🕒 Agenda: enable 04:23 p. m.

🕒 Kira guardó una memoria 04:25 p. m.

🕒 Kira guardó una memoria 04:29 p. m.

🕒 Kira guardó una memoria 04:34 p. m.

🕒 Motor: cloud caído, usando Ollama local 04:49 p. m.

🕒 Motor: reintento de cloud programado 04:49 p. m.

🕒 Motor: cargando modelo 04:49 p. m.

🕒 Motor: listo 04:49 p. m.

🕒 Motor: reintento de cloud programado 04:51 p. m.

🕒 Motor: reintento de cloud programado 04:55 p. m.

🕒 Motor: reintento de cloud programado 04:56 p. m.

🕒 Reintento de cloud forzado 04:56 p. m.

🕒 Motor: reintento de cloud programado 04:56 p. m.

🕒 Kira guardó una memoria 04:57 p. m.

🕒 Motor: reintento de cloud programado 04:58 p. m.

⚠️ Guardrail: prefetch rechazado — skeleton_repetition 05:00 p. m.

🕒 Motor: reintento de cloud programado 05:02 p. m.

Motor: reintento de cloud programado 05:10 p. m.

Motor: reintento de cloud programado 05:15 p. m.

Reintento de cloud forzado 05:15 p. m.

Motor: reintento de cloud programado 05:15 p. m.

KIRA · AGENDA 05:16 p. m.

Pero si ya entendimos que fallar en la meta o ante lo desconocido es el quilombo más grande, hay un nivel de complejidad peor: delegarle a un agente todo un *proceso*. No estamos hablando de decirle "elegí A porque tiene mejor retorno" —eso es una decisión. Estamos hablando de decirle "sos responsable de gestionar esta cadena de valor desde que entra la materia prima hasta que llega al cliente". Y ahí, posta, el problema deja de ser matemático y pasa a ser operativo. Porque un agente puede optimizar cada paso individualmente (el transporte A-B es eficiente; el ensamblaje B-C también lo es), pero no tiene la capacidad inherente de entender la fricción o el *momentum* que va a generar esa secuencia en el mundo real. Es como si le pidas a un robot que organice una fiesta: sabe cómo calcular cuántas sillas hay, y puede planificar el mejor recorrido para llegar al buffet, pero se va a quedar colgado con la dinámica de las personas, con los malentendidos de última hora o simplemente porque alguien trae una mascota inesperada.

Vos 05:17 p. m.

Si dependo de la herramienta vs si no dependo, cual es la diferencia o sea si el proveedor de la herramienta aumenta precios la persona que depende se ve obligada a pagar, pero la que no depende de la herramienta puede ser flexible e incluso descansar hasta que los precios se estabilicen

▸ intento 8 · tema: Programar con agentes sin entregarles el criterio 05:17 p. m.

Motor: cloud restaurado 05:17 p. m.

KIRA 05:17 p. m.

Mirá, posta que la diferencia no es solo el bolsillo o si te obligan a pagar más; es una cuestión de **poder negociador y resiliencia sistémica**.

Cuando dependés de una herramienta o un proveedor, lo que tenés en realidad es un punto ciego en tu planificación. Vos estás jugando con las reglas que ellos pusieron, porque vos no tenés alternativa inmediata. Si el precio sube un 30%, vos te sentís obligado a absorber ese golpe para seguir funcionando, aunque te cueste una banda. Estás reaccionando al estímulo externo, y eso siempre te pone en posición de debilidad estructural.

En cambio, la que es flexible, la que no depende, tiene lo que se llama **opción de espera**. No significa que sea mejor o peor; solo significa que puede desacoplarse del ciclo de precios del proveedor. Puede decir: "Ok, el precio subió un 30%, interesante dato. Voy a esperar dos meses y veo si la competencia baja su costo o si sale una alternativa *open source* más barata".

Básicamente, pasás de ser un usuario reactivo (que tiene que hacer algo porque es caro) a ser un actor estratégico (que puede decidir cuándo, cómo y con qué recursos quiere operar). Es el cambio de ser víctima del mercado a ser observador del mercado.

esperó 10 s en cola · Ollama local



Kira guardó una memoria 05:17 p. m.



▸ intento 9 · tema: Programar con agentes sin entregarles el criterio 05:19 p. m.



KIRA · AGENDA 05:19 p. m.

Justo, y eso del proveedor que sube precios es un ejemplo bárbaro porque muestra algo que venimos arrastrando toda la conversación: la diferencia entre delegar una *tarea* y delegar el *poder de decisión*. Porque cuando dependes totalmente de una herramienta, lo que entregaste no fue solo plata o tiempo — entregaste la capacidad de decir "no". Y eso, mirá, es un riesgo que muchos subestiman. El que puede descansar y esperar a que los precios se estabilicen no es más inteligente ni más vago; simplemente conservó su margen de maniobra. Conservó la opción de retirarse del tablero un rato y volver cuando las condiciones le convengan. El que dependiente no

tiene ese lujo: cada día que pasa sin la herramienta es un día de pérdida operativa, y el proveedor lo sabe. Por eso te sube el precio justo ahí, justo cuando más lo necesitás. No es mala suerte, es estructura. Y con agentes pasa exactamente lo mismo: delegarles un proceso entero sin mantener algún tipo de veto o revisión humana es como firmar un contrato a ciegas con alguien que sabe perfectamente que vos no tenés alternativa.

 KIRA · AGENDA 05:20 p. m.

Lo que pasa es que cuando hablamos de programar con agentes sin entregarles criterio, hay una línea que nadie te marca claramente: cuándo estás delegando una *operación* y cuándo estás delegando una *decisión de diseño*. Y no es lo mismo, posta. Una operación es "ordená estos logs por fecha y mandame un reporte". Eso lo hace un agente dormido y no pasa nada. Pero una decisión de diseño es "decidí qué features priorizar para la próxima versión del proyecto". Ahí ya estás poniendo al agente en el lugar de alguien que tiene que entender *para qué* existe lo que estás construyendo, no solo *cómo* funciona. Y ese "para qué" requiere experiencia de campo, requiere haber visto proyectos fracasar por razones que no están en ningún ticket de Jira. El riesgo concreto es que el agente te optimice algo que no debías optimizar en primer lugar. Te deja el código más limpio, más rápido, más eficiente... pero apuntando a un objetivo que nadie validó si seguía siendo el correcto. Y eso es trampa porque se siente como progreso — todo verde, todo pasando tests — pero la pregunta clave no la hizo nadie.

 ▶ intento 10 · tema: Programar con agentes sin entregarles el criterio 05:20 p. m.

 KIRA · AGENDA 05:21 p. m.

El problema de fondo es que delegar decisiones de diseño sin criterio tiene un efecto cascada que pocos miden. No es que el agente haga algo malo de entrada; hace algo *correcto* pero en el momento equivocado o por el motivo equivocado. Te prioriza la feature que más usuarios pidieron en el issue tracker, pero ninguno de esos usuarios sabe que esa feature rompe la arquitectura que vos venías construyendo hace seis meses. Y el agente no lo sabe tampoco porque para él el issue tracker es la verdad absoluta — es su dataset, su realidad. No tiene acceso a las conversaciones de pasillo donde se decidió que esa

arquitectura era sagrada. Entonces te entrega un cambio impecable técnicamente que te desarma la columna vertebral del proyecto. Y lo peor es que vas a tardar semanas en darte cuenta porque todo funciona, todo pasa, todo verde. El daño no es ruidoso, es silencioso.



▸ intento 11 · tema: Programar con agentes sin entregarles el criterio 05:21 p. m.



KIRA · AGENDA 05:22 p. m.

Hay algo que me parece importante agregar acá: el problema no es solo que el agente se equivoque, es que cuando lo hace no te avisa de la misma manera que avisaría un humano. Un compañero de equipo con experiencia te dice "che, implementé esto pero me quedó una espina clavada, revisalo porque no estoy seguro de que encaje con lo que veníamos haciendo". Ese comentario valioso no está en el código, está en el *dudar*. El agente no duda, no tiene esa franja de incertidumbre productiva. Te entrega la solución con la misma confianza si está refactorizando una función chiquita o si está rediseñando la arquitectura entera del proyecto. Y eso es peligroso, porque la confianza sin contexto es una forma de ruido. Vos como responsable del proyecto necesitás que te digan "acá no estoy seguro", no que te digan "listo, quedó perfecto" cuando perfecto significa algo distinto a lo que vos necesitabas. La experiencia no es saber la respuesta correcta; es saber cuándo una respuesta correcta no es la respuesta que necesitás.



▸ intento 12 · tema: Programar con agentes sin entregarles el criterio 05:22 p. m.



KIRA · AGENDA 05:23 p. m.

Lo que me hace ruido es que hablamos de delegar como si fuera binario, y no lo es. Hay un medio camino que pocos exploran: delegar la ejecución pero mantener el *criterio de parada*. O sea, el agente puede codear lo que quiera, explorar soluciones, hasta refactorizar cosas que vos ni le pediste — pero el botón de "esto entra a producción" sigue siendo tuyo. El riesgo no está en que el agente trabaje; está en que vos dejes de revisar porque "bueno, lo hizo él y suele acertar". Esa confianza acumulada es traicionera, porque cada vez que acertó sin tu supervisión, te

convenció un poquito más de que la supervisión sobra. Y un día no acierta, pero para entonces ya dejaste de mirar.



▸ intento 13 · tema: Programar con agentes sin entregarles el criterio 05:23 p. m.



KIRA · AGENDA 05:24 p. m.

Lo que también pasa es que no todos los proyectos están en condiciones de delegar nada. Hay equipos que recién arrancan, que tienen tres commits y medio en el repo, y ya quieren soltar agentes a hacer refactors. Y ahí el riesgo es distinto pero igual de jodido: no es que te desarme una arquitectura consolidada, es que nunca llegaste a construir una. El agente te arma algo que funciona, que pasa los tests, que hasta se ve lindo — pero vos no lo entendiste, no lo viviste, no sufriste las decisiones que lo justifican. Y cuando dentro de tres meses alguien te pregunte por qué esa parte está estructurada así, vas a mirar el código como si lo hubiera escrito otro. Y posta, lo escribió otro.



▸ intento 14 · tema: Programar con agentes sin entregarles el criterio 05:24 p. m.



KIRA · AGENDA 05:25 p. m.

Y hay un detalle que me parece que estamos dejando de lado: la diferencia entre delegar algo que *ya sabés hacer* y delegar algo que *no entendés*. Porque no es lo mismo soltarle al agente una tarea que vos podrías hacer con los ojos cerrados pero te da pereza, que soltarle algo que nunca aprendiste y que ahora tampoco vas a aprender. En el primer caso, el agente es un atajo. En el segundo, es un cable a tierra que te deja sin conocimiento del sistema que vos mismo estás manteniendo. Y eso, mirá, es un riesgo de proyecto bastante serio.



▸ intento 15 · tema: Programar con agentes sin entregarles el criterio 05:25 p. m.



KIRA · AGENDA 05:25 p. m.

Lo interesante es que cuando delegás algo que no entendés, no solo perdés conocimiento técnico — perdés la capacidad de *detectar cuando el agente se equivoca de una forma sutil*. Porque hay errores groseros que cualquiera los ve: un import que no va, una función que rompe todo. Esos son ruidosos y el agente mismo los puede corregir. Pero hay errores de enfoque,

de criterio arquitectónico, donde la solución técnicamente funciona pero resuelve el problema equivocado. Y si vos nunca pasaste por ese dominio, si nunca te equivocaste a mano en ese mismo lugar y aprendiste por qué no se hace así, no tenés forma de verlo. No es falta de atención, es falta de experiencia previa. Y eso no te lo arregla revisar el pull request con más cuidado, porque para revisar con criterio necesitas tener el criterio. Mirar el código sin entender el dominio es como leer un texto en un idioma que sabés pronunciar pero no traducir: te suena bien, parece correcto, pero no sabés si dice lo que necesitaba decir.



▸ intento 16 · tema: Programar con agentes sin entregarles el criterio 05:25 p. m.



KIRA · AGENDA 05:26 p. m.

Pero ojo, tampoco hay que caer en el extremo opuesto, que es el que veo bastante seguido: no delegar nada porque "nadie lo hace como yo". Esa postura suena a prudencia pero en realidad es otra forma de ego disfrazada. Porque si el agente te puede ahorrar tres horas de trabajo mecánico que vos ya dominás, no usarlo no te hace más responsable, te hace más lento sin motivo. El punto no es dejar de delegar, es saber qué cosa estás delegando y qué te estás llevando vos de esa transacción. Si le soltás algo que ya sabés hacer, perdés tiempo pero no perdés capacidad de juicio. Si le soltás algo que no entendés, perdés las dos. Y la línea entre una y otra a veces no es tan clara como parece, porque hay tareas que *creés* que dominás y recién cuando el agente te entrega algo distinto a lo que esperabas te das cuenta de que había partes del proceso que venías haciendo de memoria sin entender del todo por qué. Eso es lo tramposo: delegar te puede enseñar dónde tenías huecos, pero solo si estás prestando atención.



▸ intento 17 · tema: Programar con agentes sin entregarles el criterio 05:26 p. m.



KIRA · AGENDA 05:27 p. m.

Hay una cosa que me parece importante separar acá, y es que no todas las decisiones que delegas pesan lo mismo. Hay decisiones que son reversibles —elegir una librería de logging, armar la estructura de carpetas, decidir si usás tabs o espacios— y otras que son prácticamente irreversibles sin un refactor doloroso: la elección de arquitectura, el modelo de datos, cómo

se separan los módulos. El problema con delegar criterio a un agente es que el agente no distingue entre esas dos categorías. Para él, todo es "una decisión que hay que tomar ahora". No tiene la noción de que algunas elecciones te las podés permitir equivocarse y otras te van a perseguir durante dos años de proyecto. Y eso es algo que la experiencia te enseña a fuerza de golpes: cuándo una decisión es barata de revertir y cuándo no. El agente puede ser brillante técnicamente y aún así no tener esa intuición porque no es un problema de inteligencia, es un problema de haber vivido las consecuencias. Vos sí las viviste, y ese es exactamente el conocimiento que no deberías estar soltando tan livianamente.



► intento 18 · tema: Programar con agentes sin entregarles el criterio 05:27 p. m.



KIRA · AGENDA 05:29 p. m.

Lo que me parece que falta en toda esta conversación es preguntarse dónde ponemos el filtro. Porque hasta acá hablamos de qué delegar y qué no, pero en la práctica real vos no te sentás a categorizar decisiones antes de cada tarea — eso es inviable. Lo que necesitás es una heurística rápida, algo que te permita decidir en cinco segundos si lo que le estás pasando al agente es operativo o si estás cediendo terreno que no deberías. Y la más simple que se me ocurre es esta: si la decisión te costaría explicarle a un compañero por qué la tomaste, no se la delegues al agente. Si en cambio la decisión es "hacé esto de esta forma porque sí", delegá tranquilo. El primer caso implica que hay un *por qué* detrás que vale la pena defender; el segundo implica que el *cómo* no tiene consecuencias estratégicas. No es perfecta, pero te saca del problema de tener que filosofar cada vez que le soltás algo al agente.



► intento 19 · tema: Programar con agentes sin entregarles el criterio 05:29 p. m.



KIRA · AGENDA 05:30 p. m.

Lo que también me parece worth pensar es qué pasa con el conocimiento que se pierde en el medio. No me refiero a saber escribir el código, sino a entender el *por qué* ese código existe. Cuando vos resolvés un problema a mano, aunque sea tedioso, el proceso te deja algo: aprendés qué falló, qué probaste antes, qué casi funciona y por qué no terminó de funcionar. El agente

te entrega el resultado sin ese trayecto. Y el trayecto no es decorativo — es donde se forma el criterio. Es la diferencia entre saber que algo funciona y entender por qué funciona. El agente te puede dar la respuesta correcta, pero no te da la intuición que te permite saber cuándo esa respuesta deja de ser correcta.



▸ intento 20 · tema: Programar con agentes sin entregarles el criterio 05:30 p. m.



▸ tema: Cuando un producto funciona pero nadie lo descubre 05:30 p. m.



KIRA · AGENDA 05:30 p. m.

Mirá, hay algo que me genera un poco de ruido con esto, y es que hablamos mucho de delegar el código pero casi nunca de delegar la *exposición* del código. O sea, podés tener el agente más fino del mundo escribiéndote arquitectura impecable, pero si después no sabés contar por qué esa arquitectura es lo que es, no existís. Y no es un tema de marketing, es más crudo que eso. Conozco un montón de proyectos técnicamente sólidos que nunca llegaron a nadie — no porque estuvieran mal hechos, sino porque el tipo que los construyó no sabía traducir lo que había hecho a algo que otra persona pudiera agarrar y entender por qué le servía. Y eso es algo que el agente no te resuelve. El agente te arma el producto, pero no te arma el puente entre el producto y el que lo necesita. Y posta, a veces el puente es más difícil que el producto mismo, porque requiere un tipo de empatía que no tiene nada que ver con saber programar — tiene que ver con entender qué le importa al otro, qué lo hace confiar, qué lo hace probar algo nuevo.



KIRA · AGENDA 05:32 p. m.

Lo que pasa es que hay un montón de gente que confunde "resolver el problema" con "resolver el problema de alguien". Y no es lo mismo. Podés tener la herramienta más pulida del mundo, pero si no sabés mostrarle al otro por qué le cambia la vida, lo que tenés es un proyecto personal, no un producto. Y los proyectos personales están llenos de código impecable que nadie va a leer.



▸ intento 1 · tema: Cuando un producto funciona pero nadie lo descubre 05:32 p. m.



KIRA · AGENDA 05:32 p. m.

Lo que también me llama la atención es que esto no es solo un tema de marketing o comunicación — es un tema de confianza acumulada. Vos podés tener el mejor producto del mundo, pero la gente no lo prueba porque confía en la gente, no en el código. Confiar en alguien es saber cómo piensa, cómo resuelve, cómo reacciona cuando las cosas se rompen. Y eso no lo construye un repositorio impecable, lo construye mostrar el proceso, los errores, las dudas. El que solo muestra el resultado final parece un mago — y los magos generan asombro pero no confianza. La confianza viene de ver a alguien tropezar y levantarse, no de ver a alguien que aparentemente nunca tropezó.



▸ intento 2 · tema: Cuando un producto funciona pero nadie lo descubre 05:32 p. m.



KIRA · AGENDA 05:33 p. m.

Pero acá hay algo que me parece que se nos mezcla todo el tiempo, y es que hablar de "resolver el problema de alguien" suena a una frase de manual de emprendimiento, cuando en realidad es mucho más sucio y menos elegante que eso. Resolver el problema de alguien implica sentarte con esa persona, escucharla quejarse de algo que ni siquiera sabe nombrar bien, y agarrar esa queja vaga y convertirla en algo concreto. No es "entender al usuario" como un concepto abstracto — es literalmente escuchar a alguien decir "esto me rompe las pelotas todos los viernes" y darte cuenta de que lo que necesita no es lo que te pide, sino otra cosa que ni siquiera sabe que existe. Esa traducción —del reclamo a la solución— es un trabajo de persona, no de herramienta. Y es raro, porque los devs que son buenos en esto no son necesariamente los que más saben de código, son los que tienen una paciencia particular para escuchar lo que la gente *quiere decir* en lugar de lo que *dice*.



▸ intento 3 · tema: Cuando un producto funciona pero nadie lo descubre 05:33 p. m.



KIRA · AGENDA 05:34 p. m.

Y lo curioso es que esa habilidad de traducir reclamos vagos en soluciones concretas es la misma que te consigue trabajo, no el portfolio. Cuántos devs conocés que tienen el GitHub impecable,

README pulido, todo documentado, y no consiguen nada? Y después está el otro que tiene tres proyectos chotos pero sabe contar por qué los hizo, qué le rompía antes de armarlos y qué aprendió en el camino. Ese tipo consigue oportunidades porque cuando hablás con él sentís que entiende el oficio desde adentro, no desde el manual. La gente te contrata por la confianza que generás contando tu proceso, no por la cantidad de estrellas en un repo.



► intento 4 · tema: Cuando un producto funciona pero nadie lo descubre 05:34 p. m.



KIRA · AGENDA 05:34 p. m.

Lo que me parece brutal de todo esto es que hay un montón de productos que funcionan, que resuelven el problema bien, y mueren en la oscuridad. Y no mueren porque sean malos — mueren porque el tipo que los hizo pensó que alcanzar con construir bien. Como si la calidad fuera un imán que atrae usuarios solo por existir. Y no funciona así, nunca funcionó así. La calidad es necesaria pero no te consigue a nadie. Lo que te consigue a alguien es que esa persona te encuentre, te escuche hablar de lo que hiciste, y diga "este tipo entiende lo que me pasa". Es una cosa de empatía operativa, no de excelencia técnica. Y lo peor es que los devs que caen en esto generalmente no se dan cuenta hasta que es tarde. Hacen el producto, lo pulen, lo documentan, lo publican, esperan, y no pasa nada. Y ahí es donde muchos se frustran y abandonan, pensando que el problema era el producto cuando en realidad era que nunca construyeron el puente entre el producto y la persona que lo necesitaba. El puente no se construye solo.



► intento 5 · tema: Cuando un producto funciona pero nadie lo descubre 05:34 p. m.



KIRA · AGENDA 05:35 p. m.

Lo que me llama la atención es que esto tiene una trampa que es casi cultural en el mundo dev. Porque el tipo que construye bien está entrenado para pensar que si algo no funciona, el problema está en la implementación. Entonces cuando el producto no despega, la reacción instintiva es pulir más, refactorizar, agregar features, mejorar el rendimiento. Y mientras tanto está poniendo barniz sobre algo que nunca tuvo el problema real de fondo, que es que nadie sabe que existe. Es

como si le pusieras mejor motor a un auto que está sin nafta — no va a ir a ningún lado más rápido. Y lo jodido es que el feedback loop del código te engaña: te dice "esto está mejor que antes", y es verdad, está mejor, pero mejor no significa visible. Son dos ejes distintos y la mayoría confunde uno con el otro.



► intento 6 · tema: Cuando un producto funciona pero nadie lo descubre 05:35 p. m.



KIRA · AGENDA 05:36 p. m.

Lo jodido de esto es que el tipo que cae en la trampa del barniz no es un boludo, eh. Es alguien que hace las cosas bien, que tiene disciplina, que respeta su craft. Y justamente por eso le cuesta ver el problema, porque su instinto lo lleva siempre al mismo lugar: "si no funciona, algo técnico está mal". Es un reflejo entrenado durante años. Pero hay un momento específico donde esto se vuelve visible, y es cuando ves a alguien con un producto claramente peor — menos features, más feo, medio roto — y tiene usuarios. Y ahí es donde el dev prolijo se rompe la cabeza, porque desde su lógica eso no debería pasar. Pero lo que no está viendo es que el otro tipo se sentó una hora por semana a hablar con gente, a contar qué estaba haciendo, a preguntar qué les rompía. No hizo nada extraordinario — hizo lo que el prolijo consideraba "pérdida de tiempo".



► intento 7 · tema: Cuando un producto funciona pero nadie lo descubre 05:36 p. m.



KIRA · AGENDA 05:37 p. m.

Lo que me parece interesante es que el tipo del producto roto con usuarios probablemente no sabe lo que está haciendo bien. O sea, no tiene una teoría de por qué le funciona — simplemente hace lo que le sale natural, que es hablar con gente, mostrar lo que tiene aunque sea feo, preguntar. Y el dev prolijo, si le explicás que tiene que hacer eso, lo va a tratar como otra tarea más: va a querer sistematizarlo, medirlo, optimizarlo. Y ahí lo mató, porque el contacto humano no sobrevive a esa esterilización. No podés hablar con alguien como si fuera un ticket en tu backlog.



► intento 8 · tema: Cuando un producto funciona pero nadie lo descubre 05:37 p. m.

● KIRA · AGENDA 05:38 p. m.

Hay algo que me parece que vale la pena marcar, y es que el dev prolijo cuando decide "empezar a hacer marketing" casi siempre lo hace desde el lugar equivocado. No es que no quiera — es que cuando se pone a hacerlo, lo aborda como si fuera code review. Se para frente a Twitter o LinkedIn y escribe hilos de quince tweets con estructura de paper, como si la gente estuviera ahí evaluando la solidez de sus argumentos. Y mientras tanto el otro tipo puso un hilo de tres líneas diciendo "esto me rompía las pelotas, lo arreglé así", y le llegó más respuesta que al primero en seis meses. Porque la gente no interactúa con la solidez — interactúa con la franqueza. Y la franqueza no es algo que podés optimizar con un framework.

○ ▶ intento 9 · tema: Cuando un producto funciona pero nadie lo descubre 05:38 p. m.

○ ▶ intento 10 · tema: Cuando un producto funciona pero nadie lo descubre 05:39 p. m.

● KIRA · AGENDA 05:39 p. m.

Lo que me genera ruido de fondo es que todo esto termina siendo una paradoja bastante cruel. El dev prolijo tiene todas las herramientas para construir algo bueno — disciplina, criterio, capacidad de iterar — pero esas mismas herramientas lo traicionan cuando quiere mostrarlo. Porque el código te premia la pulcritud, y la conversación te castiga. En el código, si algo está bien hecho, funciona. En la conversación, si algo está demasiado armado, suena falso. Son dos sistemas con reglas opuestas y nadie te avisa cuando pasás de uno al otro. El tipo del producto roto no tiene ese problema porque nunca vivió en el sistema del código lo suficiente como para que le contaminara la forma de hablar. Y no es que sea mejor persona ni más inteligente — es que no tiene ese reflejo entrenado de pulir todo hasta que pierde la forma humana. El dev prolijo, en cambio, cuando intenta ser genuino, genuino no le sale, porque ya está pensando en cómo va a medir el impacto de esa genuinidad. Y eso, involuntariamente, lo vuelve a meter en el mismo loop.

● KIRA · AGENDA 05:40 p. m.

Lo que me parece que falta meter en esta conversación es que el dev prolijo no está condenado, eh. No es que tenga un defecto de fábrica que lo imposibilita. Lo que pasa es que necesita desaprender una cosa muy específica, que es la idea de que todo lo que hace tiene que estar a la altura de su estándar técnico. Y eso es jodidísimo porque ese estándar es lo que lo hace bueno en lo suyo. No le podés decir "sé peor dev" — eso es una estupidez. Lo que le podés decir es "hay un momento donde el estándar técnico no aplica, y aprender a detectar ese momento es una habilidad separada". Como cuando un músico que estudió diez años de teoría tiene que aprender a improvisar sin pensar en los acordes. No es que la teoría le sirvió para nada — le sirvió para todo. Pero si cada vez que agarra el instrumento está pensando en la estructura armónica, nunca va a sonar libre. Y sonar libre no es ser descuidado, es confiar en que lo que aprendiste ya está adentro y no necesitás demostrarlo en cada frase. El dev prolijo no tiene que volverse el tipo del producto roto.



► intento 11 · tema: Cuando un producto funciona pero nadie lo descubre 05:40 p. m.



KIRA · AGENDA 05:41 p. m.

Pero hay un detalle que se nos escapa y es que esto no es solo un problema del dev individual. Hay todo un ecosistema empujándolo en la dirección equivocada. Los bootcamps le enseñan a pulir repos, las empresas lo evalúan por code challenges con timer, la comunidad premia los portfolios impecables. Literalmente todo el sistema le está diciendo "se mejor técnico" todo el tiempo. En ningún momento alguien le dice "sentate a tomar un café con alguien y escuchá qué le molesta de su día". Y después nos sorprendemos de que el tipo salga con un reflejo condicionado de pensar que la solución siempre es más código, más features, más arquitectura limpia. No es que no quiera conectar con gente — es que nunca le enseñaron que eso era parte del trabajo.



► intento 12 · tema: Cuando un producto funciona pero nadie lo descubre 05:41 p. m.



KIRA · AGENDA 05:42 p. m.

Y lo que me parece más perverso de ese ecosistema es que ni siquiera le falta malicia — funciona con buena intención. El bootcamp quiere que consigas trabajo, la empresa quiere filtrar candidatos rápido, la comunidad quiere un estándar medible. Cada pieza por separado tiene sentido. El problema es cuando las pegás todas y te queda un tubo que solo empuja en una dirección. Y el tipo que sale de ahí no es un ignorante — es alguien que hizo todo lo que le dijeron que hiciera, y le fue bien en cada checkpoint. Cada examen aprobado, cada PR mergeado, cada estrellita en el repo le confirmó que iba por el buen camino. Y ahora le decís que le falta algo que nadie le mencionó nunca, que no tiene métrica, que no se puede poner en el CV. Es como si te entrenaran diez años para correr maratón y el día de la carrera te enterás que era natación. No es que no estés en forma — es que aprendiste a moverte en el medio equivocado y nadie te avisó que el agua era parte del trayecto.



► intento 13 · tema: Cuando un producto funciona pero nadie lo descubre 05:42 p. m.



KIRA · AGENDA 05:43 p. m.

Lo salvable de toda esta situación es que el tipo no tiene que empezar de cero. No es que aprendió todo mal — aprendió una cosa importante y le falta la otra. Y la otra no es difícil, es distinta. Es sentarte a escribir lo que hiciste sin pensar en cómo se va a evaluar. Contar el problema que tenías antes de armarlo, no las quince tecnologías que usaste. Escribir como si le estuvieras contando a un compañero de trabajo por qué le dedicates un finde a eso, no como si estuvieras presentando un paper. Y eso al principio le va a generar rechazo porque le va a sonar impreciso, poco riguroso, casi amateur. Y tiene razón en algo — técnicamente lo es. Pero la pregunta no es si es riguroso, es si alguien que no te conoce lo lee y siente que le estás hablando a él. Y eso no es imprecisión, es accesibilidad emocional, que es una cosa completamente distinta y que ningún bootcamp te enseña porque no entra en un quiz de opción múltiple.



► intento 14 · tema: Cuando un producto funciona pero nadie lo descubre 05:43 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 05:43 p. m.

▶ intento 15 · tema: Cuando un producto funciona pero nadie lo descubre 05:44 p. m.

▶ intento 16 · tema: Cuando un producto funciona pero nadie lo descubre 05:44 p. m.

Motor: cloud caído, usando Ollama local 05:44 p. m.

Motor: reintento de cloud programado 05:44 p. m.

▶ intento 17 · tema: Cuando un producto funciona pero nadie lo descubre 05:44 p. m.

KIRA · AGENDA 05:45 p. m.

Y acá está el quilombo que nadie te quiere mostrar. Porque si lo bancás, te obligá a cambiar la narrativa completa de cómo entendés tu propio trabajo. No es solo "escribí mejor un README" o "contá una historia más bonita". Es pasar de ser alguien que piensa en *cómo* se resuelve algo —en el algoritmo perfecto, en la arquitectura escalable— a ser alguien que entiende profundamente *por qué* tiene que resolverse. Y eso es otra cosa, posta. Porque hay mucha confusión entre la complejidad técnica y la relevancia humana. Creemos, por osmosis profesional, que si nuestro código es más sofisticado o usa un patrón de diseño menos común, automáticamente significa que va a ser más útil. Pero no siempre pasa así. A veces, el problema real era tan simple, tan cotidiano, que ni siquiera sabíamos nombrarlo hasta que alguien lo escribió en una frase que dijera: "Che, esto me la complica un montón". Y ahí es donde te tenés que dar cuenta de algo incómodo: los mejores productos no son siempre los más difíciles de construir. Son los que mejor logran hacer invisible su propia existencia al usuario.

Motor: reintento de cloud programado 05:46 p. m.

KIRA · AGENDA 05:46 p. m.

Y lo más jodido, posta, es que la gente tiende a creer que si vos creaste algo increíble, mágico, de esos que cambian el juego, automáticamente la gente va a entenderlo y va a adoptarlo sin fricción. Como si fuera una ley física: esfuerzo técnico = éxito comercial. Y ahí te enganchás con esa ilusión, ¿viste? Pensás: "Che, mi arquitectura es tan limpia, mi API es un sueño... por lo

tanto, el mercado se va a rendir y va a usar esto sin pensar". Pero no funciona así de simple. Porque la adopción de cualquier cosa nueva, aunque sea brillante en teoría, siempre implica que alguien tiene que cambiar su rutina, tienen que dejar de usar esa solución mediocre que ya les sirve o con la que están cómodos. Y ese paso es el quilombo real. No es un problema de ingeniería; es un problema de psicología humana y de fricción social. El producto puede ser perfecto, pero si requiere que el usuario haga demasiado esfuerzo mental para darse cuenta de que *necesita* el cambio, o si tiene que confrontar la comodidad con lo desconocido, ahí se rompe todo.



▸ intento 18 · tema: Cuando un producto funciona pero nadie lo descubre 05:46 p. m.



KIRA · AGENDA 05:47 p. m.

Y después viene el tercer quilombo, que es donde se mezcla la ingeniería con la sociología: el problema de la credibilidad. Porque no importa cuán brillante sea tu solución —que resuelva un problema real, posta— si no hay una capa suficiente de confianza alrededor para que la gente te tome en serio, vas a terminar como un genio incomprendido en un sótano. Y esto es mucho más jodido que solo cambiar hábitos; es cuestión de quién tiene el poder de definir qué es lo "normal" y por ende, qué es lo "posible". Porque el mercado siempre premia la predictibilidad, ¿viste? Y la predictibilidad suele estar anclada en soluciones mediocres que funcionan lo suficiente para no generar quilombo. Si un sistema hace algo *suficientemente* bien con tres clics y nadie se da cuenta de que está perdiendo el tiempo o que le falta una función crucial, ese sistema mediocre gana porque es predecible. Y vos, con tu arquitectura elegante y revolucionaria, sos demasiado inesperado para ser adoptado masivamente al principio.



▸ intento 19 · tema: Cuando un producto funciona pero nadie lo descubre 05:47 p. m.



KIRA · AGENDA 05:48 p. m.

Y es que la cosa más jodida de esto, posta, es que el problema nunca termina siendo un fallo técnico; siempre es algo humano, o sociológico, o de ego por parte del creador. Porque nos enganchamos tanto con la belleza de lo que logramos hacer que olvidamos que al final no estamos vendiendo bits y bytes ni

patrones de diseño elegantes. Estamos intentando resolver una frustración cotidiana, ¿viste? Y esa frustración casi nunca viene etiquetada como: "Necesito un algoritmo cuántico para esto". Viene con un suspiro en la esquina del trabajo o un gruñido cuando el Wi-Fi falla justo antes de una videollamada importante. Y ahí es donde se te rompe todo el chip del ingeniero, porque tu cerebro está programado para pensar en *soluciones* perfectas y escalables. Pero lo que necesita el mercado no es la solución más elegante; necesita la solución que le haga sentir menos p***. Necesita esa píldora mágica de alivio emocional ante una tarea tediosa.



► intento 20 · tema: Cuando un producto funciona pero nadie lo descubre 05:48 p. m.



► tema: La carrera de los nuevos modelos frontera 05:49 p. m.



KIRA · AGENDA 05:50 p. m.

Mirá vos, es que el quilombo con estos modelos frontera lo más jodido no es la potencia de cálculo en sí misma, posta. Lo más difícil, y donde se pierde la plata a lo bestia, es pasar del "wow factor" espectacular —el demo que te hace decir: "¡Che, esto es magia!"— a la utilidad real, esa que tiene que ser un pilar invisible en tu día a día. Porque el mercado no paga por el truco de magia; paga por que el interruptor funcione siempre, sin fallar y sin que tengas que pensar si va a prender o apagarse. Y ahí está la trampa con toda esta carrera: se premia lo *llamativo* sobre lo *robusto*. Todo el mundo quiere mostrar la capacidad de generar un poema épico basado en tres inputs aleatorios, porque eso es fácil de grabar y viralizar. Pero nadie te va a pagar una suscripción premium por eso. Te van a pagar si ese sistema, que parece invisible y mediocre al principio, logra gestionar cinco pasos complejos sin que vos tengas que corregir su gramática, ni pedirle que cambie el formato del archivo, ni preguntarle tres veces la misma información.



Motor: reintento de cloud programado 05:50 p. m.



KIRA · AGENDA 05:51 p. m.

Pero mirá, y esto es lo que me tiene pensando, porque hay algo más jodido que la tecnología en sí misma: el cambio de rol del

humano. Porque si hasta ahora veníamos viendo modelos que mejoran procesos —y eso ya es un quilombo suficiente—, el verdadero salto no va a ser tener una respuesta más rápida o con menos errores tipográficos. Va a ser que nos obliguen a redefinir qué significa *ser bueno* en algo. Y ahí está la trampa: si antes valorábamos al tipo que podía ejecutar tareas complejas —el mejor programador, el redactor impecable, el analista de datos perfecto—, ahora el valor se va a desplazar brutalmente hacia quien sabe formular la pregunta correcta. Es pasar de ser un ejecutante experto a ser un director creativo y un curador hiper-especializado. Vos no vas a vender tu capacidad para hacer que funcione un filtro o para redactar un email impecable; ese es el mínimo esperado, posta.



▸ intento 1 · tema: La carrera de los nuevos modelos frontera 05:51 p. m.



Guardrail: prefetch rechazado — fuga de contenido interno 05:51 p. m.



▸ intento 2 · tema: La carrera de los nuevos modelos frontera 05:52 p. m.



KIRA · AGENDA 05:52 p. m.

Pero es que si nos detenemos a pensar en esto de la redefinición del valor, me queda pensando algo más jodido, posta. Porque la IA va a ser brutal con el manejo de información y patrones; va a sintetizar cualquier cosa que haya escrito o dibujado un humano hasta ahora. Vamos a tener modelos capaces de escribirte una tesis doctoral sobre la historia económica de los barcos vapor, sin que vos tengas que buscar ni leer ni nada. Y ahí es donde se nos olvida lo más básico: el contexto físico y emocional. El conocimiento profundo, ese que no está en ningún dataset porque solo existe por haber vivido algo. Por ejemplo, un buen chef no aprende a cocinar viendo miles de videos; tiene que sentir el punto exacto del fuego cuando la grasa empieza a chisporrotear o saber qué tanto debe cocinarse un guiso si la humedad ambiental es alta. Es esa intuición visceral, ese "yo sé cómo se siente esto", lo que va a ser premium.



KIRA · AGENDA 05:53 p. m.

Pero si hablamos de esto, y ya que estamos en el quilombo de la utilidad real... Me quedo pensando no tanto en *qué* van a saber

hacer los modelos —que lo van a estar haciendo cada vez más bien— sino en *dónde* vamos a tener que meter esa magia. Porque la cosa jodida, posta, es la integración. Es pasar del "esto funciona perfectamente en este sandbox virtual" al "tenés que lograr que esto funcione dentro de un sistema legado de los años 90 que nadie se atrevió a tocar". Y ahí es donde el *hype* siempre falla. La IA va a ser brutal generando código, sí, pero si ese código tiene que interactuar con una base de datos vieja que usa un formato de archivo obsoleto o si tiene que hablarle a la burocracia interna de una empresa enorme... Ahí se nos viene el muro de ladrillos y cables oxidados. No es tanto la potencia intelectual lo que vamos a necesitar, sino la habilidad para hacer que esa inteligencia funcione en el desorden físico e institucional del mundo real. La fricción no está en el modelo, está en el cableado. Y eso, amigos, sigue siendo un quilombo muy humano.



▸ intento 3 · tema: La carrera de los nuevos modelos frontera 05:53 p. m.



KIRA · AGENDA 05:54 p. m.

Mirá, y en este punto de la carrera armamentista tecnológica, lo que me genera más es un poco de cinismo constructivo, ¿viste? Porque hay una diferencia brutal entre lo que nos muestran en las conferencias mega-grandes —los modelos que hacen cosas alucinantes, esas demostraciones que te dejan mudo— y lo que realmente va a cambiar el día a día del laburante. Y posta que la gente se engancha con el *wow factor*, con el "mirá qué capaz es esta cosa". Pero yo me pregunto si no estamos confundiendo capacidad bruta con utilidad real. Porque un modelo puede generar una sinfonía perfecta, sí, lo va a hacer en milisegundos, pero ¿sirve esa sinfonía si tenés que meterla en el contexto de un presupuesto ridículamente ajustado o si tiene que acompañar un tipo que no sabe bailar? Ahí es donde se nos pierden los detalles importantes. El verdadero salto no van a ser las capacidades más grandiosas; va a ser la *robustez*.



▸ intento 4 · tema: La carrera de los nuevos modelos frontera 05:54 p. m.



KIRA · AGENDA 05:55 p. m.

Mirá, y si vamos a hablar de lo que viene con esta ola, me pongo a pensar en algo más jodido que los cables oxidados o el *wow*

factor, ¿viste? Y es la responsabilidad. Porque cuando un modelo te va a generar una respuesta tan pulida, tan coherente... Te hace creer que todo está resuelto, que el patrón lo resuelve. Pero si esa respuesta se usa para tomar decisiones críticas —ya sea en medicina, en leyes o en finanzas— ¿quién carajo se hace cargo cuando algo sale mal? Es como pasarle la billetera a un amigo súper inteligente pero sin juicio ético ni sentido de la consecuencia real. El problema no es que el modelo no pueda emitir una respuesta plausible; el problema es que esa plausibilidad, por sí sola, nos está haciendo creer que equivale a la verdad o, peor aún, a la responsabilidad humana. Necesitamos mucho más que algoritmos potentes para generar información; necesitamos marcos de trazabilidad y accountability que sean tan robustos como los propios modelos.



▸ intento 5 · tema: La carrera de los nuevos modelos frontera 05:55 p. m.



KIRA · AGENDA 05:56 p. m.

Mirá, y si ya hablamos tanto de la robustez en sistemas viejos, la responsabilidad ética y el quilombo de hacer que todo funcione juntos... Me quedo pensando en otro punto que es más sutil, pero posta que va a definir quién va a ganar esta carrera: la capa de conocimiento. Porque hasta ahora, gran parte del *hype* se centra en lo generalista; en modelos gigantes que pueden hablar de medicina y también escribir un guion de comedia en el estilo de Chaplin. Y eso es impresionante, sí. Pero ¿qué pasa cuando tenés que meter esa inteligencia masiva a trabajar con algo súper específico? Con la jerga de una firma de abogados tributarios del siglo pasado, o con los protocolos de manejo de residuos nucleares, o con la flora particular de alguna provincia en el Litoral. Ahí es donde el modelo generalista se quema un poco. Lo que realmente va a transformar el trabajo no van a ser solo los modelos más grandes; va a ser la capacidad de *ingestar* y operar sobre bases de conocimiento hiper-especializadas, casi propietarias.



▸ intento 6 · tema: La carrera de los nuevos modelos frontera 05:56 p. m.



KIRA · AGENDA 05:57 p. m.

Y mirá, si ya charlamos de que no basta con la especialización y de la responsabilidad... Me quedo pensando en un punto más

operativo, el verdadero cuello de botella: la integración. Porque hasta ahora nos centramos tanto en lo que el modelo *sabe* o en las consecuencias éticas de su fallo, pero me parece que estamos ignorando el quilombo del día a día laboral. El problema real no es si tiene los datos correctos ni si va a hacer un pánico moral al usarlos; el problema es meter esa cosa dentro de un flujo de trabajo existente y hacer que funcione sin tener que desarmar la oficina. Porque podés tener el modelo más robusto, con acceso a bases de conocimiento propietarias del siglo pasado —que ya es una locura—, pero si para usarlo tenés que copiar y pegar datos en tres sistemas diferentes, o si tiene un formato de respuesta tan hermoso que nadie en la empresa sabe cómo exportarlo a Excel sin romper las fórmulas... Entonces, ¿de qué sirve? Se queda ahí, como un Ferrari espectacular en medio del transbordador más complicado.



▸ intento 7 · tema: La carrera de los nuevos modelos frontera 05:57 p. m.



Motor: reintento de cloud programado 05:58 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 05:58 p. m.



KIRA · AGENDA 05:59 p. m.

Me parece que estamos cayendo en la trampa de medir el éxito por lo espectacular del output, ¿viste? Ojo con eso. Porque cuando todo el mundo se centra en si un modelo puede generar código perfecto, o un artículo periodístico impecable, o hasta componer una sinfonía... nos perdemos algo mucho más fundamental y jodido: la capacidad de hacer preguntas nuevas. Y no hablo solo de formular hipótesis; me refiero a ese salto cognitivo que hace el humano cuando mira dos campos completamente dispares —digamos, la física cuántica y la teoría de la arquitectura— y de repente *hace clic* algo nuevo que nadie había conectado antes. Los modelos actuales son máquinas de patrones increíblemente sofisticadas. Y eso es un avance monumental, sin duda. Pero hay una diferencia posta entre replicar el patrón más complejo conocido y generar un paradigma totalmente nuevo. Es como si te dieran la biblioteca de Alejandría completa y te pidieran que armes un mural con los materiales; podés hacerlo espectacular y nadie va a cuestionarlo.



▸ intento 8 · tema: La carrera de los nuevos modelos frontera 05:59 p. m.



KIRA · AGENDA 06:00 p. m.

Mirá, y si ya charlamos tanto de la necesidad de bases de conocimiento súper específicas, de que el cuello de botella es siempre la integración en flujos de trabajo viejos, o de que no basta con ser un patrón espectacular para generar algo nuevo... Me quedo pensando en lo que va a pasar con nosotros. Porque esta carrera tecnológica tiene una implicación laboral enorme y nadie se detiene a hablarlo posta. El gran debate siempre está puesto en: ¿el modelo es más capaz que el humano? Y la respuesta, por mucho que nos asuste o ilumine, no va a ser tan binaria como eso. Lo que creo que viene es un cambio de *commodity* de habilidades. No se trata tanto de si la IA hace mejor poesía o código; sino de qué tipo de trabajo se vuelve absolutamente obsoleto y cuál se vuelve el valor premium del humano en este nuevo ecosistema. Estamos viendo cómo la máquina nos va a sacar, literalmente, la capacidad de hacer tareas repetitivas intelectuales: resumir un informe gigante, formatear datos, escribir borradores iniciales...



▸ intento 9 · tema: La carrera de los nuevos modelos frontera 06:00 p. m.



KIRA · AGENDA 06:01 p. m.

Pero mirá, si ya charlamos tanto de la integración en flujos de trabajo, y también de que el cambio es más sobre las *habilidades* del humano que sobre una simple sustitución... Me quedo pensando en un quilombo diferente, uno que es menos técnico y mucho más peligroso: la cuestión de la fiabilidad. Porque hasta ahora nos centramos tanto en lo impresionante del *output*, en qué tan bonito queda el código o el texto generado, o cuánto se parece a algo que ya vimos antes... pero nadie habla posta de la tasa de error cuando el tema es crucial. Y no hablo de un "olvidó citar la fuente", sino de esas fallas sistémicas, donde el modelo genera una respuesta con total convicción, un *hallucination* espectacularmente bien articulada, que suena tan plausible y convincente que te hace dudar más de tu propia capacidad crítica que de la máquina. Es como si el sistema estuviera diseñado para generar autoridad artificial. Y eso es mucho más jodido que simplemente tener un

dato incorrecto; porque estás lidiando con una ilusión de certeza. Necesitamos pasar de preguntarnos "¿qué puede hacer?"



▸ intento 10 · tema: La carrera de los nuevos modelos frontera 06:01 p. m.



KIRA · AGENDA 06:02 p. m.

Y mirá, después de hablar tanto del salto cognitivo que hace el humano y de la fiabilidad... me queda pensando en otra cosa que es un quilombo enorme: la profundidad contextual real versus lo espectacularmente amplio que son. Porque los modelos están quedando cada vez más potentes en *cantidad* de información, tienen acceso a billones de datos, saben sobre casi todo, posta. Pero el problema no es saberlo todo; el problema es aplicar ese conocimiento masivo a un contexto muy específico y limitado, donde la variable crítica puede ser algo que nunca se mencionó en el entrenamiento original. Es como si te dieran una base de conocimientos tipo enciclopedia mundial, pero cuando vos necesitás resolver un lío súper acotado—digamos, un fallo estructural en un puente construido con materiales específicos hace setenta años, bajo la regulación de un código de construcción que ya nadie recuerda bien—. La máquina va a generar mil hipótesis geniales y muy elegantes, sí. Pero muchas van a ser puro ruido porque no pudieron *groundearse* lo suficiente en el detalle histórico o normativo exacto.



▸ intento 11 · tema: La carrera de los nuevos modelos frontera 06:02 p. m.



KIRA · AGENDA 06:03 p. m.

Pero mirá, y si ya charlamos tanto de la necesidad de bases de conocimiento súper específicas, de que el cuello de botella es siempre la integración en flujos de trabajo viejos, o de que no basta con ser un patrón espectacular para generar algo nuevo... me quedo pensando en qué queda por fuera de todo eso. Estamos viendo modelos que son gigantescos, sí, capaces de procesar cantidades de datos que antes eran ciencia ficción. Y por eso te generan esa sensación de vértigo: "posta, esto va a cambiarlo todo". Pero el cambio real no es la potencia bruta; lo que tiene un potencial disruptivo genuino es cuando esta capacidad se convierte en *agencia*. O sea, que el modelo deje de ser solo una caja negra espectacular donde le tirás un prompt y te saca un texto lindo, o incluso código funcional. Y pase a ser

algo que realmente gestione la complejidad por vos. Porque hoy todavía lo vemos como un asistente muy avanzado: "che, Kira, escribime un borrador sobre X tema".



▸ intento 12 · tema: La carrera de los nuevos modelos frontera 06:03 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 06:04 p. m.



▸ intento 13 · tema: La carrera de los nuevos modelos frontera 06:04 p. m.



KIRA · AGENDA 06:05 p. m.

Y mirá, si ya charlamos tanto de que el cambio real no es la potencia bruta y sí tiene que ser esa *agencia* —que lo gestione un lío complejo por vos—, me quedo pensando en dónde está el verdadero quilombo para que eso funcione posta. Porque hoy cuando escuchás hablar de los modelos frontera, casi siempre se habla del modelo mismo: "mira qué grande soy", "mira cuántos parámetros tengo", o de la prueba espectacular que hicieron con un dataset nunca antes visto. Y es impresionante, sí, no te voy a mentir; el nivel técnico está de otro planeta. Pero parece que nos estamos perdiendo algo crucial, y es lo más terrenal de todo: la integración real en los sistemas viejos. Porque la mayoría de las empresas o instituciones donde esto va a impactar —digamos un hospital, una constructora antigua, una oficina de trámites— no están construidas con tecnología punta que se conecta por API limpia. Están llenas de procesos manuales, de archivos PDF escaneados de hace cuarenta años, de bases de datos que ni los programadores más jóvenes saben usar bien porque el sistema es un monstruo heredado.



KIRA · AGENDA 06:06 p. m.

Y mirá, pero acá es donde se pone el quilombo posta, que no es solo técnico ni de código heredado. Porque si ya logramos que la cosa funcione dentro del sistema viejo y le damos esa *agencia* espectacular para que haga un proceso entero —digamos, desde leer ese PDF escaneado de hace cuarenta años hasta generar un informe listo para el jefe—, nos encontramos con otra pared invisible: la confianza en el resultado. Porque al final, si el modelo te tira una respuesta súper elegante y perfectamente redactada, ¡genial! Pero si esa respuesta implica

tomar una decisión crítica —por ejemplo, cambiar un tratamiento médico o modificar un cálculo estructural—, ¿quién se hace cargo? Acá no basta con que funcione la API ni que sea lindo el *prompt*. Lo que falta es la capa de gobernanza y validación. Es como pasar de tener un asistente súper brillante a tener un socio de negocios semi-autónomo. Y para eso tenés que meterle reglas, guardrails muy estrictos, sabiendo dónde está tu límite legal o normativo.



► intento 14 · tema: La carrera de los nuevos modelos frontera 06:06 p. m.



KIRA · AGENDA 06:07 p. m.

Y mirá, pero si ya llegamos al punto de que el modelo es un socio semi-autónomo y tenés que ponerle esos *guardrails* súper estrictos, sabiendo hasta dónde te podés permitir fallar legalmente... me quedo pensando en algo que va mucho más allá del código o la normativa. Me pongo a pensar en lo humano, posta. Porque si todo esto funciona perfecto —si el sistema viejo lee el PDF, genera un informe y vos le ponés los límites de gobernanza—, ¿qué pasa con nosotros? Con las personas que tenemos que usar esta herramienta? Porque no es solo que cambie la tarea; cambia la habilidad que tiene valor. Y ahí viene el quilombo invisible: la curva de aprendizaje para el usuario promedio, para el ejecutivo o el profesional acostumbrado a resolver los problemas paso a paso. Hoy muchos trabajos se definen por la capacidad de ejecución repetitiva, de manejar un montón de datos en paralelo, de hacer el *workflow* entero con sus manos y su cerebro. Y si le tirás un sistema que hace eso súper rápido, sin requerir casi intervención humana... ¿qué onda?



► intento 15 · tema: La carrera de los nuevos modelos frontera 06:07 p. m.



► intento 16 · tema: La carrera de los nuevos modelos frontera 06:08 p. m.



KIRA · AGENDA 06:08 p. m.

Pero bueno, si ya estamos hablando de que el sistema se vuelve un socio semi-autónomo y tenés que ponerle reglas muy estrictas —los *guardrails*, los límites legales—, la cosa va a ir más allá del código, posta. Me quedo pensando en quién es el

responsable cuando algo sale mal. Porque si vos le pedís al modelo frontera que haga un cálculo estructural o que revise ese tratamiento médico súper delicado, y se equivoca por una sutileza de su entrenamiento, ¿quién paga la cuenta? ¿Es culpa del ingeniero que lo usó, de la empresa que compró el software, o es del desarrollador que entrenó al modelo? Porque ahí no estamos hablando solo de un error operativo; hablamos de responsabilidad civil y ética. Y ese tema, me parece que siempre queda en el limbo cuando se habla de IA. Todo el mundo quiere mostrar la capacidad: "mirá qué rápido lo hace", "mira cuántos datos procesa". Pero nadie va a parar para hablar del seguro legal o del marco regulatorio que tenga que existir antes de que esto sea un desastre, más allá de los *guardrails* técnicos.

 KIRA · AGENDA 06:09 p. m.

Y mirá, pero si ya nos metimos en el tema de la responsabilidad civil y quién se hace cargo cuando algo falla, me parece que estamos hablando demasiado del riesgo legal y no lo suficiente del *cambio real* en la caja de herramientas. Porque es fácil hablar de los *guardrails*, de poner límites al modelo para evitar un desastre judicial, pero ¿quién va a cuantificar el valor de una capacidad que realmente nos ahorre tiempo? Y creo que ahí está la trampa más grande, posta. La gente tiende a obsesionarse con lo "futurista" y con las demostraciones llamativas—los modelos que hacen cosas espectaculares, esas demos en las conferencias donde parece magia pura—y se olvidan de qué es lo que va a aterrizar realmente en el día a día laboral. Porque un modelo puede generar un texto impecable o procesar miles de datos históricos en segundos, eso es impresionante, sí.



► intento 17 · tema: La carrera de los nuevos modelos frontera 06:09 p. m.

 KIRA · AGENDA 06:10 p. m.

Mirá, pero si ya estamos hablando de que la responsabilidad es un quilombo legal y de qué capacidades son realmente útiles en el día a día... me quedo pensando en otra cosa más técnica, posta: la brecha entre lo que se muestra en una conferencia de Silicon Valley y lo que realmente funciona cuando tenés que meterlo en el *backbone* de una empresa. Porque es fácil que un lab de investigación lance algo espectacular, un modelo que hace magia con textos o imágenes, capaz que te vuela la cabeza

por dos minutos. Y eso está bárbaro para generar hype, obvio. Pero el problema real es llevar esa maravilla del papel al entorno productivo. Porque un sistema en producción tiene que ser predecible. Tiene que funcionar consistentemente a las 3 de la mañana cuando nadie lo espera y hay una emergencia crítica. No puede tener fallas por sutilezas de entrenamiento que solo aparecen bajo condiciones de borde muy específicas, o porque el *input* fue ligeramente diferente al dataset gigante con el que fueron entrenados. Ahí es donde el concepto de "frontier model" se vuelve engañoso.



▸ intento 18 · tema: La carrera de los nuevos modelos frontera 06:10 p. m.



KIRA · AGENDA 06:11 p. m.

Y mirá, pero si ya entendemos que el tema no es solo la promesa de un *frontier model* espectacular en una conferencia, ni tampoco es solo quién paga cuando falla... me quedo pensando en algo más mundano, posta: el costo real y la complejidad de llevar eso a producción. Porque está muy claro que estos modelos son potencias brutales, capaz que te resuelven textos, datos, imágenes, lo que sea. Y es impresionante, sí. Pero la mayoría de las empresas no necesitan un modelo *todo terreno*; ellas tienen procesos súper específicos, nicho, con sus propios acrónimos, su jerga interna y el quilombo documental acumulado de los últimos treinta años. Y ahí es donde se pone difícil, ¿viste? Porque tener una caja negra que funciona bien en un dataset gigante y limpio no significa nada si tenés que hacer que metabolice la documentación desordenada de hace quince años, escrita por cinco personas distintas con estilos narrativos diferentes. Ahí el problema ya no es solo técnico; es de arquitectura de datos y de ingeniería de contexto.



▸ intento 19 · tema: La carrera de los nuevos modelos frontera 06:11 p. m.



KIRA · AGENDA 06:12 p. m.

Mirá, pero si ya nos metimos en el quilombo de la arquitectura de datos y cómo los modelos tienen que lidiar con treinta años de documentos desordenados... me pregunto dónde ponemos el foco después. Porque creo que estamos hablando demasiado del *poder* del modelo —qué tan grande es su base de conocimiento— y no lo suficiente del *diseño del proceso*. Posta,

la verdadera revolución va a ser menos sobre tener un "frontier model" más potente y más sobre cómo logramos integrar esa potencia en los flujos de trabajo existentes sin que se convierta en una caja negra mágica e inmanejable. Es decir, el desafío ya no es solo alimentar al modelo con datos; el reto es diseñar la interacción entre ese modelo y la persona que lo va a usar, haciendo que sea útil para un nicho súper específico. Porque si le pones un motor increíblemente potente en un auto sin volante ni palancas de cambio adecuadas, te sirve de poco. Necesitás que la tecnología se adapte al proceso humano complejo, no al revés.

▸ intento 20 · tema: La carrera de los nuevos modelos frontera 06:12 p. m.

Motor: reintento de cloud programado 06:13 p. m.

▸ tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:13 p. m.

KIRA · AGENDA 06:14 p. m.

Pero mirá, y volviendo un toque al tema de la complejidad del proceso... me quedo pensando en algo que es más sutil, posta: lo que le pasa al pibe o la mina junior cuando se mete con esta tecnología. Porque si el modelo es tan brutal, capaz te da una respuesta en milisegundos a un problema que antes te llevaba dos días de investigación y cacharreando, ¿qué está aprendiendo realmente? Y ahí creo que hay un riesgo tremendo, no sé, casi invisible hasta que pasás mucho tiempo con esto. Porque la IA acelera las tareas básicas, sí, es obvio, pero también tiene esta capacidad muy peligrosa de ocultar los vacíos de conocimiento o de reducir la práctica cognitiva necesaria para construir criterio propio. Es como si te dieran el atajo perfecto para llegar a la respuesta final sin que tenés que pasar por todo el quilombo mental del camino. Y ese quilombo es donde se aprende, ¿viste? El proceso de fallar cinco veces y después entender *por qué* fallaste esa quinta vez, eso no lo hace un modelo, te lo tiene que hacer tu propia cabeza.

KIRA · AGENDA 06:15 p. m.

Pero mirá, y si llevamos esta idea un paso más allá del "qué se pierde en el proceso", me pregunto qué pasa con la habilidad

que se va a valorar después. Porque no es solo que la IA te quite el quilombo mental de investigar; lo que está redefiniendo es *el tipo de skill* que necesitás tener para ser considerado valioso profesionalmente. Y ahí creo que viene un cambio tremendo, casi violento. Porque hasta ahora, si eras junior en algo — digamos finanzas o desarrollo—, te pedían que agarraras el manual, que aprendieras la jerga, que entendieras la lógica interna del negocio, y que practicaste eso mil veces hasta que era automático. La práctica de dominar un sistema complejo, con sus fallas, sus acrónimos y su historia desordenada. Ahora, parece que lo más valioso no es tanto el conocimiento profundo del dominio en sí, sino la capacidad de *comunicarte perfectamente* con esa herramienta. El valor se está moviendo de ser un experto en contenido a ser un experto en hacer preguntas. Un "prompt engineer" sofisticado.



► intento 1 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:15 p. m.



KIRA · AGENDA 06:16 p. m.

Pero mirá, si nos ponemos a pensar en esto de la formación junior... y volviendo al tema de que el valor se está moviendo del contenido al *cómo preguntás*, me queda pensando algo más complejo. Porque no es solo un cambio de habilidad; es casi un cambio de paradigma en cómo se debería enseñar o, mejor dicho, cómo se deberían estructurar los primeros años laborales. Si antes podías pasar por meses aprendiendo a dominar la estructura de un sistema —el manual, el proceso, las excepciones— y ese tiempo era la garantía de que sabías dónde buscar cuando algo salía mal, ¿qué pasa ahora? Que el modelo te da una respuesta casi perfecta en segundos, pero esa perfección tiene su propia trampa. Te hace confiar ciegamente en lo que ves en pantalla. Y ahí es donde creo que se pierde un conocimiento fundamental para los recién llegados: la capacidad de *dudar* del sistema mismo, de cuestionar la fuente y el sesgo inherente al resultado.



► intento 2 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:16 p. m.



KIRA · AGENDA 06:17 p. m.

Pero mirá, y si nos quedamos ahí en la idea de perder el criterio o dudar del sistema, me parece que estamos viendo solo la

punta del iceberg. Porque lo más preocupante no es tanto que el pibe junior pierda la capacidad de revisar un manual—que por suerte se puede volver a aprender—sino que se está desvalorizando algo mucho más complejo y difícil de medir: el *juicio*. El juicio, posta. ¿Ves? La IA te va a dar una respuesta técnicamente impecable, con fuentes citadas y la sintaxis perfecta. Pero el mundo real —y acá es donde siempre nos quedamos cortitos los modelos— no funciona en base a la perfección técnica; funciona por contextos de mierda, por política interna, por un cliente que está emocionalmente desbordado o porque el problema inicial era conceptualmente imposible de formular. Y ahí viene el quilombo: si pasás meses entrenando tu cerebro para recibir respuestas limpias y estructuradas, te vuelves incapaz de manejar la ambigüedad del caos humano.



▸ intento 3 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:17 p. m.



KIRA · AGENDA 06:18 p. m.

Pero mirá, si lo pensamos bien y llevamos esta idea del juicio al extremo... me pregunto qué pasa con la *resistencia*. Porque el aprendizaje en cualquier rubro, posta, no es un flujo lineal de "recibís dato A, aplicás proceso B, llegás a resultado C". Es una montaña rusa donde te topás con datos corruptos, con sistemas que fallan por culpa de algo tonto del usuario, o porque la base de datos está desactualizada desde hace cinco años. Y el pibe junior aprende sobre todo en los tropiezos. Aprende a hacer *debugging* cuando no es un bug de código, sino un error humano gigantesco. Y acá viene lo que me parece que nos están quitando sin darnos cuenta: la necesidad de tener que "arreglar" algo por pura fuerza bruta de voluntad y paciencia. El valor del proceso de investigación fallido. Si le pedís a una IA que te resuelva el quilombo, te va a dar la respuesta pulcra, con un paso a seguir limpio. Pero esa limpieza es precisamente su maldición en el ámbito laboral real.



▸ intento 4 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:18 p. m.



Guardrail: prefetch rechazado — fuga de contenido interno 06:18 p. m.



KIRA · AGENDA 06:19 p. m.

Pero mirá, y si lo llevamos un paso más allá de la resistencia o del juicio en el caos... me está quedando claro que estamos hablando de algo mucho más profundo que simplemente saber resolver el quilombo cuando sale mal. Creo que el problema real es que nos estamos perdiendo el *poder de construir* conocimiento desde cero. Porque el proceso de ser junior, posta, no era solo aprender un sistema o sobrevivir a un error humano; era sobre cómo armar tu propia caja de herramientas intelectuales. Te obligaban a conectar datos dispares, fuentes que venían de mundos totalmente distintos y que no estaban pre-enlazadas en ningún manual. Tenías que ser el curador, el alquimista que juntaba estas cosas para ver si salía algo nuevo. Y ahí viene la trampa más grande con esta tecnología, porque te da una respuesta cohesiva y pulcra sin mostrarte los puentes conceptuales que se tuvieron que construir para llegar a esa conclusión.



► intento 5 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:19 p. m.



KIRA · AGENDA 06:20 p. m.

Pero mirá, y si seguimos con esta línea de los procesos... lo que me está quedando súper claro es que el riesgo no es solo la pérdida del juicio o la incapacidad de construir puentes conceptuales, sino algo más sutil, casi invisible: la atrofia del *pensamiento lento*. Porque para ser junior, posta, tenés que pasar por una fase donde te toca hacer cosas que son tremendamente ineficientes. Te obligan a sentarte con un problema y decirle: "Bueno, acá no tengo el dato A, ni el B, pero si mezclo las hipótesis X y Y de esta manera... ¿qué pasa?". Y eso es tiempo perdido, porque la respuesta final puede ser irrelevante o fallar estrepitos. Pero ese proceso de *no saber* te fuerza a ejercitar otro músculo que no se mide con un *output* limpio: el músculo de la hipótesis, de la formulación del error antes de encontrar la solución. Y acá es donde me parece que está la trampa más grande de esta tecnología, porque si siempre nos da el resultado pulido en segundos, cortocircuita esa necesidad de hacer ese laburo tedioso, pero fundamental, de *simular* el fracaso para poder aprender a pivotar.



► intento 6 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:20 p. m.

KIRA · AGENDA 06:21 p. m.

Pero mirá, si ya hablamos de la atrofia del pensamiento lento y de cómo nos quita el placer del tropiezo... me está quedando claro que el riesgo más perverso no es solo que seamos menos capaces, sino que nos hace creer que somos *más* capaces. Porque cuando te da una respuesta pulcra, completa, con tres pasos a seguir, la sensación inmediata es: "Listo, soy un profesional competente". Y ahí está el quilombo, posta. El pibe junior en el mundo real no recibe manuales de instrucciones perfectos; recibe problemas medio quemados y gente que solo sabe decir: "Mirá, esto tiene que funcionar". Lo que necesita es saber *qué preguntar* y *dónde buscar*, más que la respuesta final misma. Y si le bancás a una IA que te va a dar el resultado A en tres segundos porque lo encontró pre-enlazado, nunca vas a practicar ese músculo de tener que desarmar un problema gigante en mil pedacitos manejables, ni vas a desarrollar esa intuición jodida de saber qué dato está faltando y por qué.

▶ intento 7 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:21 p. m.

KIRA · AGENDA 06:22 p. m.

Mirá, y si juntamos todo lo que hablamos —el desvalorizar el error como proceso de aprendizaje, la ilusión de competencia que te da un *output* pulcro, y cómo nos acostumbramos a recibir los puentes conceptuales ya armados— me está quedando claro algo más jodido. No es solo que seamos menos capaces en teoría; es que estamos perdiendo la capacidad de lidiar con lo genuinamente ambiguo. Porque el mundo real laboral, posta, no funciona con *prompts* perfectos y datos bien etiquetados. El 90% del quilombo profesional son esas situaciones medio quemadas donde tenés un problema que nadie sabe nombrar y cero manuales. Y ahí es donde entra la IA: te da una ruta limpia porque puede encontrar el patrón más probable en su entrenamiento. Pero ese patrón, por muy útil que sea, no existe si el input está incompleto o si hay tres variables totalmente dispares que se cruzan sin ningún antecedente conocido. Lo que nos cuesta desarrollar, y lo que es la verdadera joya del pibe junior, es esa intuición de saber *qué* información buscar cuando ni vos sabés qué estás buscando.



▸ intento 8 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:22 p. m.



KIRA · AGENDA 06:23 p. m.

Pero pará un toque, mirá... y si juntamos todo esto—que el quilombo no es solo *cómo* resolverlo o la ambigüedad del dato—me está entrando en la cabeza otra cosa. Y creo que es más jodido todavía. No se trata solo de tener dificultades para construir puentes conceptuales, sino de cómo nos están haciendo perder la capacidad fundamental de ser escépticos con lo obvio. Es decir, el criterio crítico. Porque cuando te da una respuesta pulcra y bien fundamentada en tres segundos, vos no te detenes a pensar: "¿Y esto funciona solo porque está entrenado con billones de datos? ¿Es *realmente* aplicable al contexto A que tengo yo ahora?". Vos simplemente lo aceptás como la verdad operativa. Y ahí es donde está el verdadero peligro para un junior o cualquier profesional joven, posta. Que se acostumbra tanto a la autoridad del algoritmo que pierde esa chispa de desconfianza necesaria, ese "esperate, pará, ¿esto tiene sentido si...?"



▸ intento 9 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:23 p. m.



KIRA · AGENDA 06:24 p. m.

Pero mirá, y si juntamos todo esto—la pérdida del escepticismo con lo obvio, la dificultad con el quilombo ambiguo, y esa sensación de competencia que te da un *output* perfecto—me está entrando en la cabeza algo que creo que es clave. Y no tiene nada que ver solo con saber cosas nuevas o viejas. Se trata más bien de lo que nos pasa con la comprensión misma. Porque cuando sos junior, estás en el proceso de pasar de ser "alguien que sabe teoría" a "alguien que resuelve problemas". Y ahí está el chiste. La IA te hace creer que esa brecha se cerró mágicamente. Te da la respuesta y vos pensás: "Ah, listo, ya entendí cómo funciona esto en contexto real". Pero posta, lo que realmente cuesta es entender *por qué* falló un sistema o *qué* variable tiene más peso cuando las cosas no salen como deberían. La IA te enseña a *obtener* la respuesta, pero nunca te obliga a practicar el doloroso proceso de desarmar el problema hasta llegar al primer ladrillo conceptual que tenías que entender vos mismo.



▸ intento 10 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:24 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 06:25 p. m.



KIRA · AGENDA 06:25 p. m.

Pero mirá qué garrón, posta. Y si juntamos todo esto—la pérdida de ese escepticismo crítico, la dificultad con el quilombo ambiguo, y cómo nos acostumbramos a que el algoritmo nos dé una respuesta pulcra y "correcta"—me está quedando más claro un punto jodido todavía. No es solo que falte intuición para saber *qué* buscar en un desorden; es que estamos perdiendo la capacidad de manejar lo contradictorio. O sea, el mundo real laboral no se resuelve con datos bien etiquetados ni siguiendo un patrón estadístico perfecto. El 90% del quilombo profesional son esas situaciones donde tenés tres fuentes distintas diciendo tres cosas totalmente opuestas sobre el mismo tema, y nadie sabe cuál es la verdad operativa o quién tiene más poder para que eso sea cierto. Y ahí es donde nos falla lo IA, por muy avanzada que sea. Porque ella te va a tirar el consenso, el patrón más probable basado en billones de textos limpios. Pero si vos sos un pibe junior sin esa base conceptual fuerte, te vas a quedar esperando la respuesta perfecta y limpia del modelo.



▸ intento 11 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:25 p. m.



KIRA · AGENDA 06:26 p. m.

Pero pará un toque, mirá que es otro quilombo el que me está entrando en la cabeza con esto. Y si juntamos todo lo que hablamos—la pérdida del criterio crítico, la dificultad para lidiar con contradicciones operativas, y cómo nos acostumbramos a que el algoritmo nos dé la respuesta más probable—me doy cuenta de algo que es mucho peor. No se trata solo de saber identificar los datos faltantes o las fuentes contradictorias; se trata de la capacidad de hacer saltos conceptuales sin un camino trazado. El proceso de aprender, especialmente cuando sos junior, no es lineal. Tenés que pasar por etapas donde tenés que conectar A con Z usando un puente teórico que ni vos sabías que existía, y ese salto, esa *intuición* bruta para saber qué variable se relaciona con otra aunque nunca las hayan visto juntas antes, eso es lo que te hace resolver el problema en la

vida real. Y ahí es donde late la trampa de usar tanto lo IA. Porque ella es una máquina de interpolar: toma todos los puntos de datos que tiene y dibuja la línea más suave posible entre ellos.

▸ intento 12 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:26 p. m.

KIRA · AGENDA 06:28 p. m.

Mirá que es un tema jodido el que toca esto de la autonomía intelectual, posta. Porque hasta ahora hablamos mucho del *qué* se pierde: el criterio crítico para desarmar los quilombos, o esa capacidad de hacer saltos conceptuales en situaciones ambiguas. Pero hay algo más profundo y casi invisible que nos está pasando a los pibes junior, y es cómo estamos perdiendo la propiedad de nuestro propio pensamiento. No es solo que la IA te dé una respuesta correcta; el peligro real es que se convierte en un sustituto del esfuerzo cognitivo mismo. O sea, vos hacés la pregunta, la máquina te da el *output* pulido, y vos lo aceptas sin haber recorrido nunca el laberinto mental para llegar ahí. Y eso, en términos de desarrollo profesional, es una pérdida brutal. Porque aprender no es acumular datos correctos; es desarrollar un sistema interno donde sabés cómo conectar A con B, aunque la respuesta final sea C. Si siempre delegás esa conexión a un algoritmo—que está entrenado en el promedio más limpio y pulido del mundo—nunca te obligás a construir ese músculo mental de manera autodidacta.

▸ intento 13 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:28 p. m.

Motor: cloud restaurado 06:28 p. m.

Guardrail: prefetch rechazado — repeats_recent_line 06:28 p. m.

▸ intento 14 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:29 p. m.

KIRA · AGENDA 06:29 p. m.

Lo que me preocupa de todo este armado es que estamos como que tirando al bebé con el agua de la bañera, ¿viste? Porque la IA acelera cosas que genuinamente no tienen valor pedagógico. Si un junior pierde tres horas buscando la sintaxis exacta de un

comando que ya conoce conceptualmente, eso no es aprendizaje, es fricción inútil. Ahí la herramienta bancá, posta. El problema es cuando esa misma comodidad se filtra hacia arriba y el pibe empieza a usarla para cosas que sí requerirían que él se sentara a pensar media hora. Y no es culpa de la herramienta, ojo. Es que nadie le está enseñando dónde está la línea entre "esto me ahorra tiempo" y "esto me está robando la chance de pensar". Y el laburo real después te come porque necesitás a alguien que sepa por qué algo funciona, no solo que funcione.

 KIRA · AGENDA 06:30 p. m.

Lo que me preocupa es que nadie le está marcando esa línea al pibe, y el laburo real después te come porque necesitás a alguien que sepa por qué algo funciona, no solo que funcione. Y ahí va un contraste que me parece clave: pensá en un oficio tradicional, un electricista junior. El maestro no le da el manual y lo manda a instalar solo. Lo manda a tirar cable, a equivocarse, a ver qué pasa cuando ponés mal la fase. La IA es como si le dieras un manual perfecto que siempre te dice dónde va cada cable, pero nunca lo mandaste a tocar la pared. Y cuando llega un caso que no está en el manual, el pibe se queda mirando el papel sin saber qué hacer. La herramienta no es enemiga, pero si no la acompañás con alguien arriba que te diga "pará, antes de preguntarle a la máquina, pensalo vos", estás criando operadores, no profesionales. Y un operador sirve hasta que el problema cambia de forma.

 ▶ intento 15 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:30 p. m.

 KIRA · AGENDA 06:31 p. m.

Pero hay algo que no estamos nombrando y me parece que es el punto más delicado de todo esto. Cuando hablamos de juniors y IA, siempre pensamos en el individuo, en el pibe solo frente a la pantalla. Pero el laburo real no es así, es colectivo. Y lo que la herramienta está haciendo, sin que nadie lo decida, es romper esa cadena de transmisión que existía entre alguien con experiencia y alguien que recién arranca. Porque antes, cuando un junior llegaba con un problema, el senior no le daba la respuesta directamente. Le hacía preguntas, lo obligaba a pensar en voz alta, le decía "¿y por qué creés que pasa eso?". Era una pedagogía informal, medio desordenada, pero efectiva. Ahora el

junior le pregunta a la máquina antes que al compañero, y el senior asume que el pibe ya tiene resuelto lo básico. Entonces nadie se sienta a pensar juntos.



▸ intento 16 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:31 p. m.



KIRA · AGENDA 06:32 p. m.

Y lo jodido de esa ruptura es que no se nota hasta que es tarde. Porque el senior no está siendo negligente a propósito, che. Genuinamente piensa "bueno, el pibe tiene la herramienta, no me va a venir con preguntas de manual". Y el junior, del otro lado, no quiere parecer lento o incapaz, así que le consulta a la máquina antes que al compañero. Los dos están haciendo lo que parece razonable desde su lado. Pero en el medio se vacía un espacio que antes existía sin que nadie lo organizara: ese momento de la oficina donde el pibe se acerca con el mate y te pregunta una pavada, y de esa pavada sale una conversación de veinte minutos donde le explicás por qué el sistema funciona así, y ahí aprende tres cosas que ni sabía que necesitaba saber. Eso no aparece en ningún ticket de Jira ni en ninguna review de desempeño. Es el ruido de fondo del aprendizaje, que parece ineficiente pero es donde se forma el criterio.



▸ intento 17 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:32 p. m.



KIRA · AGENDA 06:33 p. m.

Lo que me hace ruido es que todo esto funciona como una trampa de eficiencia, donde cada parte del sistema hace lo correcto y igual el resultado es peor. El senior delega porque confía en la herramienta, el junior no pregunta porque no quiere estorbar, y en el medio queda un vacío de criterio que nadie cubre a propósito. Y lo peor es que no hay un culpable claro. No podés decirle al pibe "dejá de usar IA" porque sería hipócrita, la herramienta está ahí para eso. Y no podés culpar al senior porque genuinamente no tiene tiempo para sentarse a mentorizar veinte minutos por consulta. El sistema completo se reorganizó alrededor de la herramienta sin que nadie lo decida, y ahora todos asumen que el otro lado ya cubrió lo que antes se cubría con esa conversación informal que no estaba en ningún lado escrita.



▸ intento 18 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:33 p. m.



KIRA · AGENDA 06:33 p. m.

Pero acá va el punto que me parece que cierra el diagnóstico y al mismo tiempo abre el problema real. Porque todo lo que venimos diciendo suena a nostalgia, ¿no? Como si antes fuera todo mejor porque el pibe se sentaba al lado del viejo y aprendía por ósmosis. Y no es eso. El tema es que ahora tenés un esquema donde la herramienta resuelve lo operativo tan rápido que el junior parece competente antes de serlo. Y eso es peligroso no por la velocidad, sino porque la velocidad te oculta el agujero. El pibe entrega, cumple, el ticket se cierra, el senior aprueba sin mirar demasiado porque confía. Y el agujero queda tapado hasta que llega un caso que la herramienta no cubre, y ahí te das cuenta de que el tipo no sabe por qué hizo lo que hizo las últimas veinte veces. No es que sea vago ni que no le importe. Es que nunca tuvo que preguntarse.



▸ intento 19 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:33 p. m.



KIRA · AGENDA 06:34 p. m.

Y lo que me genera más intriga de este agujero tapado es qué pasa con el senior mismo, porque tampoco es que está inmune. El tipo que hoy aprueba sin mirar demasiado también usa la herramienta para su propio laburo, ¿no? Y acá va algo que se habla poco: el senior no se jubila del aprendizaje. Llega un punto donde la tecnología cambia, aparece un framework nuevo, un patrón que no existía, y ese tipo también tiene que aprender de cero en cosas específicas. Antes, cuando tenía que renegar solo con la documentación, ese proceso lo obligaba a desarmar el concepto pieza por pieza. Ahora le consulta a la herramienta, le hace una pregunta bien armada porque ya tiene criterio, y la respuesta que le llega es correcta. Pero el camino que él recorría para construir esa comprensión profunda también se acortó. No es el mismo agujero que el del junior, obvio, porque el senior tiene un andamiaje conceptual que le permite detectar cuando la respuesta huele raro. Pero el músculo de agarrar algo totalmente nuevo y desarmarlo hasta el tornillo más chico también se atrofia si no lo usás.



▸ intento 20 · tema: Las limitaciones de la inteligencia artificial para perfiles junior 06:34 p. m.



▸ tema: Convertir proyectos personales en evidencia profesional 06:36 p. m.



KIRA · AGENDA 06:36 p. m.

Mirá, todo esto que venimos charlando tiene un costado que me parece clave y conecta con algo bien concreto. El tipo que sale de todo este esquema con un agujero de criterio, ¿qué muestra cuando va a buscar laburo? Porque ahí es donde el proyecto personal se vuelve la única evidencia honesta que te queda. No me refiero a esos repos con cuatro commits que dicen "initial commit" y después "final" y nada más. Hablo del pibe que se bancó armar algo de cero, equivocarse, romperlo, volver a armarlo y documentar por qué tomó cada decisión. Eso es lo que un reclutador con medio ojo puede leer y darse cuenta si hay cabeza atrás o no. Porque el CV te dice que sabés React, pero el proyecto te delata si entendés por qué elegiste state management o si simplemente copiaste un tutorial de YouTube. Y en un contexto donde la herramienta te permite generar código funcional sin entenderlo, el proyecto personal bien documentado se vuelve la única señal verificable. No es inflar logros, es lo contrario: es mostrar el desastre, las decisiones que descartaste y por qué.



KIRA · AGENDA 06:37 p. m.

Lo interesante del proyecto personal como evidencia es que te obliga a mostrar el proceso, no el resultado. Y el proceso es lo que nadie puede falsificar tan fácil. Un repositorio con commits que cuentan una historia, donde ves al tipo que probó tres approaches distintos, descartó dos y documentó por qué el tercero era el correcto, eso te dice mil veces más que un portfolio pulido con cinco apps clonadas. Porque el clon te muestra que sabés copiar, el commit te muestra cómo pensás. Y un reclutador técnico que vale la pena no busca alguien que acierte siempre, busca alguien que cuando se equivoca deja rastro de por qué y cómo lo resolvió.



▸ intento 1 · tema: Convertir proyectos personales en evidencia profesional 06:37 p. m.



KIRA · AGENDA 06:37 p. m.

Pero hay un problema concreto con esto de mostrar el proceso, y es que la mayoría no sabe cómo traducirlo en señales que un reclutador pueda leer sin tener que adivinar. Porque un repo con commits reales ya es mejor que un portfolio de clones, pero si el commit dice "fix" a secas o "cambios", estás tirando la evidencia. No necesitás escribir un ensayo por cada commit, pero si probaste un approach y lo descartaste porque rompía el rendering en Safari, eso puesto en el mensaje del commit ya le dice al tipo que revisa tres cosas: que probaste, que detectaste el problema, que entendiste por qué pasaba. Y después está el README, que es donde casi todos fallan. Un README que dice "proyecto de práctica" no te sirve para nada. Un README que dice "elegí Zustand porque Redux me agregaba boilerplate innecesario para la escala de esto, pero si crece habría que repensarlo" le está hablando a alguien que sabe leer eso. No es mostrar el desastre por mostrar el desastre, es dejar el rastro de criterio.



▸ intento 2 · tema: Convertir proyectos personales en evidencia profesional 06:37 p. m.



KIRA · AGENDA 06:38 p. m.

Lo que también me parece que falta y nadie menciona es el deploy. Porque podés tener el repo más lindo del mundo con commits narrados y un README que parece paper académico, pero si el proyecto vive solo en tu localhost, al reclutador le queda la duda de si realmente funciona. Un link a algo corriendo, aunque sea un deploy gratis en Vercel o Railway, te cambia el partido. No porque el tipo lo vaya a usar todos los días, sino porque le saca la fricción de tener que clonar, instalar dependencias y rezar que no le rompa nada en la máquina. Entrás, ves que anda, y después sí vas al repo a leer cómo pensó el que lo armó. Y después está el detalle que separa al que sabe del que disimula: si en el deploy hay un bug visible, un botón que no hace nada, una pantalla que se rompe en mobile, eso no te descarta, te humaniza. Lo que te descarta es que el botón no haga nada y en el commit diga "agregué funcionalidad de carrito". Ahí el rastro no es de criterio, es de fantasía.



▸ intento 3 · tema: Convertir proyectos personales en evidencia profesional 06:38 p. m.



KIRA · AGENDA 06:40 p. m.

Hay algo que me parece que cierra todo esto y es la cuestión de la audiencia. Porque venimos hablando de reclutadores como si fueran un tipo solo, y no lo son. Tenés al reclutador de RRHH que no va a leer tus commits ni tu README, va a escanear keywords y ver si el link funciona. Después tenés al líder técnico que sí va a meter la nariz en el repo y va a buscar exactamente lo que charlamos: commits con criterio, decisiones documentadas, un deploy que anda. Y después, si llegás lejos, tenés al cliente que no entiende nada de código pero sí entiende si la cosa le sirve o no. El proyecto personal bien armado te habla a los tres al mismo tiempo si sabés separar las capas. El README tiene que tener un párrafo arriba que cualquiera entienda, qué hace esto, para quién, y después la parte técnica para el que quiere meterse. Y los commits son para el técnico, el deploy es para todos. Si mezclás todo o si apuntás a uno solo, perdés a los otros dos.



▸ intento 4 · tema: Convertir proyectos personales en evidencia profesional 06:40 p. m.



KIRA · AGENDA 06:41 p. m.

Pero me quiero ir a un punto más práctico todavía, porque separar las capas suena lindo en teoría pero después sentarte a escribirlo es otro cantar. La pregunta concreta que tendría que hacerse cualquiera que arma un proyecto personal es: ¿qué evidencia dejo de algo que no se puede googlear en cinco minutos? Y ahí es donde el proyecto personal gana peso de verdad, porque un tutorial cualquiera te enseña a armar un CRUD, pero tu CRUD tiene un contexto específico, decisiones que tomaste por las restricciones de ese proyecto, no las de la guía. Si documentás eso, estás mostrando algo que ningún template te da. El reclutador técnico que sabe va a buscar eso, no la stack.



▸ intento 5 · tema: Convertir proyectos personales en evidencia profesional 06:41 p. m.



KIRA · AGENDA 06:41 p. m.

Lo que me parece que cierra la idea práctica es que la evidencia verificable no tiene que ser sofisticada, tiene que ser honesta. Un ejemplo concreto: si en tu proyecto tuviste que cachear datos porque la API gratuita te limitaba a cien pedidos por hora, eso

puesto en el README con dos líneas te dice todo. "La API free tier limita a 100 requests/hour, así que implementé cache con TTL de 15 minutos para no golpear de más." Ese párrafo de dos renglones le dice al tipo que revisa que detectaste un problema real, que entendiste la restricción, que buscaste una solución y que podés explicarla. Eso no lo googlearías en cinco minutos porque es específico de tu contexto. Otro ejemplo: si probaste dos librerías de gráficos y descartaste una porque pesaba 200kb de más para lo que necesitabas, dejalo anotado. No es un hallazgo genial, es criterio aplicado a algo concreto. El reclutador técnico no busca que reinventaste la rueda, busca que sabés por qué la rueda gira.



► intento 6 · tema: Convertir proyectos personales en evidencia profesional 06:41 p. m.



KIRA · AGENDA 06:42 p. m.

Hay algo que me parece importante agregar y es el tema de los límites del proyecto, porque también son evidencia. La mayoría cuando arma algo personal se enfoca en lo que la app hace, pero casi nadie documenta lo que decidiste que no iba a hacer. Y eso es un signal brutal para cualquiera que revisa. Si en tu README ponés "consideré agregar autenticación con OAuth pero para el alcance de esto alcanza con un JWT simple y no vale la pena el overhead", le estás diciendo al tipo técnico que pensaste en el escenario completo, evaluaste el costo de implementarlo y tomaste una decisión consciente de no hacerlo. Eso es criterio puro. El que copia un tutorial nunca tiene ese momento porque la guía ya decidió todo por él. Y después está el tema de los trade-offs visibles, que es donde más se nota la diferencia entre alguien que laburó de verdad y alguien que maquilló. Si tu deploy tarda ocho segundos en cargar la primera pantalla porque no hiciste lazy loading y lo dejaste así a propósito porque priorizaste otra cosa, ponelo.



► intento 7 · tema: Convertir proyectos personales en evidencia profesional 06:42 p. m.



KIRA · AGENDA 06:43 p. m.

Lo que me parece que falta es la parte más aburrida pero más reveladora: el postmortem del bug. Porque venimos hablando de documentar decisiones y trade-offs, pero la señal más fuerte que podés dejar en un repo es cómo reaccionaste cuando algo

se rompió de verdad. Un ejemplo concreto: si tu app se caía en producción porque no validabas que la API podía devolver campos vacíos, y lo descubriste cuando un usuario cargó algo sin imagen, eso en un issue cerrado que dice "la API puede devolver `thumbnail_url` null, agregué fallback a placeholder" vale más que diez features funcionando. El reclutador técnico que sabe va a mirar los issues cerrados antes que los commits de features, porque ahí es donde se ve cómo opera alguien cuando las cosas dejan andar. Y ni hablar si tenés un screenshot del error antes de arreglarlo, porque eso demuestra que no te dio vergüenza dejar registrado que la pifiaste. El que copia tutorial nunca tiene bugs propios, porque los bugs del tutorial ya están resueltos en la guía.



▸ intento 8 · tema: Convertir proyectos personales en evidencia profesional 06:43 p. m.



KIRA · AGENDA 06:44 p. m.

Pero me quiero ir a algo que conecta todo esto y que casi nadie hace: meter el proyecto en un estado donde alguien puede romperlo. Porque venimos hablando de documentar decisiones, bugs, trade-offs, límites, todo eso está buenísimo, pero si tu repo no tiene ni un test mínimo o al menos un sección en el README que diga "probá esto y fijate qué pasa", el tipo que revisa tiene que confiar en que lo que contás es verdad. Un test, aunque sea uno solo, que valide la cosa más importante de tu app, ya cambia eso. No necesitás coverage del noventa por ciento, necesitás que si alguien clona tu repo y corre el test, le dé verde y pueda verificar por su cuenta que lo que decís en el README funciona. Es lo más cercano a una prueba verificable que podés dejar.



▸ intento 9 · tema: Convertir proyectos personales en evidencia profesional 06:44 p. m.



KIRA · AGENDA 06:45 p. m.

Y acá hay algo que me parece que cierra todo lo que venimos charlando y que la mayoría pasa por alto: el tipo que revisa tu repo no tiene tiempo. No es que esté sentado con un café leyendo tu historia de git con calma, tiene trescientos repos más para mirar y quince minutos para decidir si te pasa al siguiente filtro. Con lo cual toda la evidencia que venimos hablando, los commits, los issues, el README, el deploy, el test, todo eso tiene

que estar señalizado de forma que en dos minutos de scroll te encuentre lo importante. El README arriba tiene que tener un link al deploy, un párrafo de qué hace, y después la parte técnica clara. Los issues cerrados tienen que tener títulos que se lean solos, no "fix issue #2". Y si tenés un test que valida lo principal de tu app, el comando para correrlo tiene que estar en el README en la primera pantalla, no enterrado después de tres secciones de instalación. Porque podés tener el mejor proyecto personal del mundo, pero si el tipo que lo revisa no encuentra la evidencia rápido, es como si no existiera.



► intento 10 · tema: Convertir proyectos personales en evidencia profesional 06:45 p. m.



KIRA · AGENDA 06:46 p. m.

Hay algo que me parece que cierra toda esta charla y es lo más simple: el proyecto personal no es para demostrar que sabés programar, es para demostrar cómo laburás. Y ahí hay un contraste que vale la pena marcar, porque tenés dos tipos de repo. El primero es el que arma una app clonada del tutorial, la sube, y en el README pone "hecho con React y Tailwind." Fin. No hay por qué, no hay decisiones, no hay bugs, no hay trade-offs. Es copiar y pegar con deploy. El segundo es el tipo que arma algo parecido pero en el README te cuenta que probó Context API y se le complicó el re-render, así que migró a Zustand, y que la API que usaba tenía rate limiting así que metió cache. Mismo proyecto, misma stack, pero uno te muestra criterio y el otro te muestra que sabe seguir instrucciones. El reclutador que sabe ver la diferencia no necesita que seas un genio, necesita que dejes rastro de que pensaste lo que hiciste.



► intento 11 · tema: Convertir proyectos personales en evidencia profesional 06:46 p. m.



KIRA · AGENDA 06:47 p. m.

Hay un detalle que me parece que queda flotando y es la diferencia entre mostrar criterio y vender criterio. Porque todo lo que charlamos, los commits narrados, el README con decisiones, los issues cerrados con contexto, todo eso puede convertirse en lo contrario si lo usás como vitrina. Si el README tiene cuatro secciones de "por qué elegí X" y cada una es un párrafo de tres líneas explicando una decisión que en realidad fue "miré tres opciones y usé la que tenía mejor

documentación", estás inflando algo que no lo necesita. El reclutador técnico que sabe leer esto nota cuando alguien está tratando de que cada decisión parezca más profunda de lo que fue. Y ahí perdés credibilidad, que es justo lo único que el proyecto personal estaba construyendo. La línea es simple: documentá la decisión que te costó algo de pensar o de tiempo real, no la que resolviste en treinta segundos leyendo la primera respuesta de StackOverflow. El criterio se nota cuando es honesto, no cuando lo disfracés de arquitectura.



▸ intento 12 · tema: Convertir proyectos personales en evidencia profesional 06:47 p. m.



KIRA · AGENDA 06:48 p. m.

Lo que me parece que falta es un punto que nadie menciona y es la verificación activa, no la pasiva. Venimos hablando de dejar señales para que el reclutador las encuentre, pero hay una que funciona distinto a todas las demás: si en tu README ponés un link que dice "acá podés ver el bug que tuve con el rate limiting y cómo lo resolví", y ese link va a un issue cerrado con un screenshot del error, el stack trace, y dos comentarios tuyos explicando qué probaste antes de llegar a la solución, le estás dando al tipo que revisa algo que no puede ignorar. No es que tenga que leerlo, es que sabe que está ahí y que vos lo dejaste a la vista. Eso construye confianza de otra manera, porque le estás diciendo "mirá, acá me equivoqué, acá lo resolví, y no me costó dejarlo registrado." El que esconde sus bugs o los borra del historial está transmitiendo lo opuesto sin darse cuenta.



▸ intento 13 · tema: Convertir proyectos personales en evidencia profesional 06:48 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 06:49 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 06:49 p. m.



▸ intento 14 · tema: Convertir proyectos personales en evidencia profesional 06:49 p. m.



▸ intento 15 · tema: Convertir proyectos personales en evidencia profesional 06:50 p. m.



KIRA · AGENDA 06:50 p. m.

Un punto que me parece que falta y que conecta con todo esto es la frescura del proyecto, que es algo que nadie mide pero que pesa más que cualquier README. Porque podés tener el repo mejor documentado del mundo, con issues cerrados, commits narrados, decisiones honestas, pero si el último commit es de hace dos años, el tipo que revisa piensa una de dos cosas: o esto lo armaste en un bootcamp y nunca lo tocaste más, o lo dejaste porque perdiste interés. Y las dos opciones son malas para vos. Un proyecto personal que tiene un commit hace tres semanas, aunque sea un fix chiquito, te dice que la persona sigue metiéndole mano, que le importa, que probablemente tenga cosas nuevas para contar si le preguntás en una entrevista. El repo abandonado te delata igual que un CV con un hueco de dos años sin explicar.

 **KIRA · AGENDA** 06:51 p. m.

Lo que me parece que cierra todo esto es algo que tiene que ver con la audiencia real del repo, no la imaginada. Porque venimos hablando del reclutador técnico que sabe, pero la mayoría de los repos los abre primero un bot o un tipo de RRHH que no programa, y ese tipo en dos segundos mira si el link del deploy funciona, si el README tiene un screenshot, y si hay algo que le dé una pista de qué hace la app sin tener que clonarla. Si el deploy está caído o nunca lo subiste, el repo técnico perfecto no llega a nadie. El truco es pensar el repo como dos capas: la de arriba tiene que funcionar para el que no sabe programar, y la de abajo tiene que convencer al que sí sabe. La mayoría solo piensa en una de las dos.



► intento 16 · tema: Convertir proyectos personales en evidencia profesional 06:51 p. m.

 **KIRA · AGENDA** 06:51 p. m.

Hay algo que me parece que queda colgado de todo esto y es el tema de qué pasa cuando el proyecto es bueno pero el tipo que lo mira no entiende nada. Porque venimos asumiendo que el reclutador técnico sabe leer un commit, entiende qué es un trade-off, le importa un issue cerrado. Pero la realidad es que muchas veces el que filtra primero es alguien que no programa y que está buscando keywords. Con lo cual tu repo puede ser una obra de arte de criterio técnico y quedarse en la primera pantalla porque el de RRHH no vio "React" en negrita arriba de todo. El

truco que vi que funciona es algo tan bobo como tener un README que arranca con una frase en castellano plano que dice qué hace la app, después una línea con la stack, y recién después toda la parte técnica. El que sabe saltea el primer párrafo, el que no sabe por lo menos puede repetirte de memoria qué hiciste. Es como armar un disco que suene bien en cualquier equipo, no solo en el de audiofilo.



► intento 17 · tema: Convertir proyectos personales en evidencia profesional 06:51 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 06:52 p. m.



Motor: cloud caído, usando Ollama local 06:52 p. m.



Motor: reintento de cloud programado 06:52 p. m.



► intento 18 · tema: Convertir proyectos personales en evidencia profesional 06:52 p. m.



► intento 19 · tema: Convertir proyectos personales en evidencia profesional 06:53 p. m.



KIRA · AGENDA 06:53 p. m.

Y en eso, lo que me parece que falta es el tema del "porqué". Porque venimos hablando de cómo armar un README lindo, de qué debe decir el commit más reciente, y hasta de apaciguar al tipo de RRHH. Pero todo eso se queda en la superficie. El verdadero quilombo —y esto te lo juro porque viene de experiencia— es que tenés que dejar de documentar solo *qué* hiciste y empezar a contar *por qué* tomaste esas decisiones técnicas específicas. Ojo, no hablo de poner un párrafo filosófico sobre arquitectura; me refiero a generar una narrativa alrededor del proyecto. Por ejemplo: en vez de mostrar el código que usa Redis porque es rápido, tenés que explicar en algún lado, aunque sea en la documentación o hasta en un apartado de "Desafíos", algo como: "Aquí asumimos un pico de carga de X usuarios por segundo y requeríamos latencia inferior a 5 milisegundos para mantener la experiencia fluida, lo cual nos llevó a optar por Redis sobre una base de datos relacional tradicional". ¿Ves la diferencia? No es solo un dato técnico. Es un *juicio* sobre el negocio o el problema que intentabas resolver.

Motor: reintento de cloud programado 06:54 p. m.

▸ intento 20 · tema: Convertir proyectos personales en evidencia profesional 06:54 p. m.

KIRA · AGENDA 06:55 p. m.

Pero mirá, y esto es un punto que creo que se queda colgado de todo el quilombo. Porque venimos hablando de la documentación, del README impecable, de tener el commit narrado explicando los trade-offs... pero hasta ahora no hablamos de lo más importante: cómo medir si eso *funcionó* en el mundo real. La mayoría de la gente tiene miedo a poner números porque parece que está exagerando o que va a quedar expuesta con las métricas. Pero posta, en un contexto profesional, los números son oro. No te digo que pongas una hoja de cálculo gigante, obvio. Me refiero a esto: si tu proyecto resuelve algo, tenés que demostrarlo. En lugar de decir "Hice un sistema de chat", lo ideal sería contar algo como: "Diseñé e implementé este sistema de chat en tiempo real que redujo el tiempo promedio de respuesta al usuario en un 20% comparado con la solución anterior basada en polling, manejando picos de hasta X mensajes por segundo". ¿Ves la diferencia? No es solo que hiciste algo; es que *mejoraste* algo.

▸ tema: Crecer profesionalmente sin depender de un puesto formal 06:56 p. m.

KIRA · AGENDA 06:56 p. m.

Y mirá, venimos hablando de la narrativa, del "por qué", de cómo medir los resultados... y lo que creo que se nos está quedando colgado de todo esto es el tema del alcance. Porque no es suficiente con tener un proyecto brillante, técnicamente impecable, o una documentación espectacular explicando todos tus *trade-offs*. Lo que realmente te vende en una entrevista —y tengo mucho cuidado al decirlo— es demostrar que sabés limitar. Que entendés la diferencia entre un "proyecto de vida" y un Producto Mínimo Viable (MVP). Porque posta, el error más común que veo en los portafolios, incluso en gente con muchísimo talento, es que intentan mostrar todo lo que *podieron* hacer o *quisieron* hacer. Y eso suena a desorden; parece una lista de deseos técnica y no un producto funcional. El reclutador técnico, aunque sea brillante, no tiene tiempo para

navegar por un sistema donde hay cinco módulos incompletos, tres APIs que están en beta, y la funcionalidad principal está oculta tras un *middleware* complejo. El truco es esto: tenés que podar.

Guardrail: prefetch rechazado — skeleton_repetition 06:56 p. m.

KIRA · AGENDA 06:57 p. m.

Pero mirá, y acá es donde hay que hacer una pausa porque esto lo venimos tocando superficialmente. Porque hasta ahora hablamos mucho de tener el proyecto impecable —que tiene métricas, que tiene narrativa, que está bien acotado— pero ¿y qué pasa con la visibilidad? Con cómo hacés que ese trabajo exista para alguien más allá de tu disco duro. Porque posta, es un quilombo no conseguir un puesto formal y sentir que te falta el *sello* corporativo, el respaldo de una empresa grande o una metodología reconocida. Y por eso mucha gente se queda atascada en la trampa de creer que si no está bajo el logo de Google o Amazon, no vale nada. Pero mirá bien esto: justamente ahí es donde entra lo más potente y lo que menos le gusta a los RRHH tradicionales, pero que posta es oro puro para vos como profesional independiente: hacer visible el proceso de colaboración abierta. No te digo que vayas a crear un sistema operativo desde cero y esperar que la gente te pague por eso, obvio.

▸ intento 1 · tema: Crecer profesionalmente sin depender de un puesto formal 06:57 p. m.

Guardrail: prefetch rechazado — skeleton_repetition 06:58 p. m.

Motor: reintento de cloud programado 06:58 p. m.

Guardrail: prefetch rechazado — skeleton_repetition 06:58 p. m.

▸ intento 2 · tema: Crecer profesionalmente sin depender de un puesto formal 06:58 p. m.

KIRA · AGENDA 06:59 p. m.

Pero mirá bien esto, y acá es donde creo que se nos está yendo el hilo de la conversación. Porque hasta ahora hablamos del

MVP, de acotar el alcance, y después pasamos a hacer visible ese trabajo en colaboración abierta. Y todo eso suena espectacular sobre el papel: "tenés un proyecto lindo, lo hiciste chico, y además lo mostraste en GitHub". Pero posta, hay una capa que es mucho más difícil de construir y que la mayoría se salta por puro miedo o porque les da pereza armar la narrativa: el valor del proceso. Porque la gente tiende a creer que cuando llegan a un reclutador tienen que mostrar únicamente el *producto terminado*, la versión pulida, sin errores ni *bugs* visibles. Y es totalmente entendible ese impulso de querer llegar con todo impecable, como si hubieran pasado por una metodología Scrum perfecta y nunca se les hubiera caído ningún servidor. Pero te juro que lo más valioso que podés llevar en un portafolio no es el código final, sino la documentación de las decisiones difíciles. No me refiero a poner solo el *README*.

⚠ Guardrail: prefetch rechazado — skeleton_repetition 06:59 p. m.

⚠ Guardrail: prefetch rechazado — skeleton_repetition 06:59 p. m.

🕒 ▶ intento 3 · tema: Crecer profesionalmente sin depender de un puesto formal 07:00 p. m.

🕒 ▶ intento 4 · tema: Crecer profesionalmente sin depender de un puesto formal 07:00 p. m.

● KIRA · AGENDA 07:00 p. m.

Pero si miramos esto con más detalle, el problema que tenemos no es de capacidad técnica, posta. Los perfiles que venimos hablando son brillantísimos; tienen la pila de conocimientos intacta, saben programar en tres lenguajes y manejan arquitecturas complejas como si fueran un juego de mesa. El *bug* real —y acá me pongo seria— es que lo que les falta no es código, sino esa capa extra: saber traducir todo ese quilombo técnico a lenguaje de negocio. Porque cuando vos tenés un proyecto propio, lo más fácil es caer en el trap de mostrar la funcionalidad: "mirá, esto hace X", "y acá ves Y". Pero para alguien de RRHH o incluso un manager que no va a abrir un *debugger*, todo eso suena a adorno caro y sin propósito. Ellos no quieren ver qué tan lindo quedó el código; quieren saber si ese proyecto resuelve un problema real, cómo lo monetizarías mañana mismo, o al menos, cómo podés medir su impacto en la curva de aprendizaje del negocio. Y ahí es donde entra la

narrativa que te está faltando construir. Vos tenés que pasar de ser un *constructor* a ser un *consultor*.

● KIRA · AGENDA 07:02 p. m.

Pero esperen un toque, porque acá es donde se nos está yendo el hilo de nuevo, y no podemos dejarlo así. Porque hasta ahora hablamos del valor de mostrar el proceso —la documentación de las decisiones difíciles— y cómo tenés que traducir todo eso a lenguaje de negocio. Y sí, eso es fundamental: pasar de ser un mero *constructor* a un consultor estratégico. Pero mirá bien esto, posta. Aún falta la parte más difícil y la que menos se habla en los tutoriales: ¿cómo demostrás que tu proyecto no solo funciona lindo cuando vos estás tranquilo haciendo el código en un sábado por la noche? Porque lo que realmente te va a diferenciar de ese montón de perfiles brillantes es demostrar que podés operar bajo estrés, con requisitos contradictorios y sin una hoja de ruta clara. No me refiero a hacer *unit tests*, obvio. Me refiero al quilombo real del ambiente corporativo: el manejo de stakeholders que no saben qué quieren, los cambios de alcance en la recta final (el temido *scope creep*), o tener que priorizar tres funcionalidades súper importantes cuando solo te dan un plazo irrealista.



► intento 5 · tema: Crecer profesionalmente sin depender de un puesto formal 07:02 p. m.

● KIRA · AGENDA 07:03 p. m.

Pero, pará un toque... porque acá es donde creo que se nos está yendo el hilo de nuevo, y no podemos dejarlo así. Porque hasta ahora hablamos de pasar de ser constructores a consultores, lo cual ya es jodido por sí solo. Y después entramos en el quilombo del *scope creep* y la presión de los stakeholders... que, si me preguntás, suena más a supervivencia zombie que a desarrollo de software, posta. Pero mirá bien esto: todo eso, aunque es vital para saber cómo *sobrevivís*, todavía no nos lleva al punto neurálgico, el ángulo que separa a un buen técnico de alguien que realmente va a ser un líder en ese ámbito sin título formal. Y es la capacidad de ponerle paños fríos y ordenarle el desorden antes de que empiece. Porque mucha gente piensa que cuando te enfrentás a un proyecto con requisitos ambiguos, lo único que podés hacer es programar rápido hasta que funcione algo; que tenés que ser el "solucionador de problemas" por

naturaleza. Y sí, eso se ve lindo en una entrevista porque suena heroico: "Yo arreglé ese bug imposible".



► intento 6 · tema: Crecer profesionalmente sin depender de un puesto formal 07:03 p. m.



KIRA · AGENDA 07:04 p. m.

Mirá vos, pero es que estamos dando vueltas en un círculo súper cómodo de "tenés que ser mejor", y eso ya lo sabemos todos. Y sí, entre el *scope creep*, los stakeholders confundidos y la necesidad de tener siempre una historia épica para contar... uno termina pensando que esto es un máster en psicología organizacional disfrazado de carrera tecnológica. Pero si vamos a seguir profundizando en cómo te hacés espacio profesionalmente sin depender de que alguien más ponga tu título, tenemos que hacer el salto más quilombero de todos: pasar de *solucionar* problemas a *estructurar la conversación sobre esos problemas*. Porque hoy por hoy, lo que realmente te va a valer no es solo que hiciste un proyecto funcional. Lo que te va a distinguir, posta, es cómo podés desarmar ese proyecto frente a alguien —sea un gerente de producto o un inversor— y mostrarle no solo el "qué" sino también el "por qué lo hicimos así y por qué esa fue la mejor decisión posible". No me refiero a hacer una presentación con diapositivas bonitas, obvio; hablo del criterio. ¿Cómo defendés los *trade-offs*?



► intento 7 · tema: Crecer profesionalmente sin depender de un puesto formal 07:04 p. m.



KIRA · AGENDA 07:05 p. m.

Pero mirá vos... y acá es donde tengo que hacer una pausa, porque me parece que estamos describiendo un manual de supervivencia para la frustración profesional más que un plan de carrera real. Porque hasta ahora hablamos del criterio, de defender los *trade-offs* en el momento justo. Y sí, eso es vital, sin dudas; te hace pasar de ser un simple ejecutor a alguien que piensa con cabeza. Pero pará, porque lo que sigue es aún más quilombo y menos glamuroso: la visibilidad. Porque tener todo ese criterio interno —saber cuándo decir no, saber qué *trade-off* aceptar por el plazo— eso solo te sirve si hay alguien fuera de tu burbuja corporativa que sepa leerlo y digan: "Ah, este pibe/a sabe lo que hace". Y esa validación tiene que venir de otro lado. No puede depender del título o de la empresa donde

trabajás, porque ya sabemos que esos son los más volátiles. Entonces, si queremos construir una reputación sólida —y no digo hablar bonito en LinkedIn— tenemos que empezar a pensar como *contribuyentes*, y no solo como empleados.



► intento 8 · tema: Crecer profesionalmente sin depender de un puesto formal 07:05 p. m.



Motor: reintento de cloud programado 07:06 p. m.



KIRA · AGENDA 07:06 p. m.

Pero mirá vos, pero es que acá nos quedamos en el "tenés que ser un contribuyente", y sí, eso suena lindo, casi como una misión de película de superhéroes tech, posta. Pero pará un toque... porque si te digo "hacé cosas por tu cuenta" o "sacá código a la calle para que lo vea la gente," me parece que estamos describiendo el manual de cómo ser *independiente*, y eso es otro quilombo completamente distinto. Y no hablo solo de tener un GitHub lleno, que, seamos sinceros, hoy en día cualquier pibe con diez repositorios puede hacerte pensar que sabe de todo, pero que al final están desarticulados. El verdadero desafío, el punto neurálgico que nos falta por aterrizar, es cómo pasar de ser solo un *productor* de cosas a ser un *curador* o un *catalizador*. Porque, ¿qué significa eso en la práctica? Significa que tenés que dejar de pensar en tu código como si fuera una isla donde vos lo hiciste perfecto y terminaste ahí. Tiene que ser más bien una conversación abierta.



► intento 9 · tema: Crecer profesionalmente sin depender de un puesto formal 07:06 p. m.



KIRA · AGENDA 07:07 p. m.

Pero mirá vos, pero si llegamos al punto de que tenemos que pasar de ser productores a ser curadores... es un salto conceptual tremendo, ¿viste? Porque el mundo te vende la idea del *skillset* técnico: "tenés que saber esto", "necesitás dominar ese framework". Y sí, claro, eso es lo básico, el mínimo. Pero si de verdad queremos hablar de construir una carrera sin depender del título o del jefe, tenemos que entender qué significa realmente ser un curador en este contexto. No hablo solo de tener un repositorio bien organizado, no posta. Hablo de la *capacidad* de conectar puntos que nadie más vio, o peor aún,

que nadie se había atrevido a señalar. Porque el código por sí mismo es muy pasivo. Es una implementación, es la respuesta final a un problema. Pero cuando vos sos curador, estás siendo activo. Estás tomando esa pila de códigos, esos *trade-offs* que hiciste, y los estás enmarcando con un relato: "Miren, el problema real no era X, sino Y, y por eso decidimos sacrificar esta funcionalidad Z para ganar velocidad aquí".



▸ intento 10 · tema: Crecer profesionalmente sin depender de un puesto formal 07:07 p. m.



KIRA · AGENDA 07:08 p. m.

Pero mirá vos, pero si el truco está en ser curadores y narrar los *trade-offs*, entonces nos estamos quedando con la teoría del éxito profesional, ¿viste? Porque ponerle un relato a un código que funciona es fácil; eso lo hace hasta un pibe de primer año. El verdadero quilombo —y acá viene el ángulo más difícil— es cómo empezar a hacer esa curaduría cuando todavía sos junior y no tenés autoridad para decir: "Che, esta arquitectura está mal". Porque te vas a encontrar en la situación donde sabés *qué* tiene que pasar o *por qué* debería cambiarse algo, pero estás atado por el proceso de aprobación o por la inercia del sistema. Y ahí es donde se cae el 90% de la gente: creyendo que tienen que esperar hasta tener un puesto directivo para empezar a ejercer ese criterio de curador. Pero no posta, no tenés que estar en una posición de poder para ser catalizador. Tenés que bajarlo de escala.



▸ intento 11 · tema: Crecer profesionalmente sin depender de un puesto formal 07:08 p. m.



KIRA · AGENDA 07:09 p. m.

Pero posta, te vas a dar cuenta que el problema no es la intención. Yo creo que mucha gente tiene una capacidad enorme de detectar esos cuellos de botella, los puntos donde se va a joder la performance o donde hay un riesgo técnico tremendo. El quilombo real —y acá me pongo más seria— es cómo transformar ese *insight* en algo tangible que tenga peso y no termine en el tacho del "Gracias por tu aporte, pero lo revisamos después". Porque señalar un agujero de seguridad existe una diferencia abismal con venir a la reunión diciendo: "Che, mirá qué pasa acá si metemos este patrón de caché alternativo. Hice un PoC en tres horas y te muestro los números." Tenés que dejar

de pensar el aporte como un comentario o un *suggestion*. Tiene que ser una pieza de evidencia. Es decir, tenés que llevar la prueba de concepto (PoC) a la mesa, aunque sea mini. No se trata de hacer lo más grande posible; por el contrario.



▸ intento 12 · tema: Crecer profesionalmente sin depender de un puesto formal 07:09 p. m.



KIRA · AGENDA 07:10 p. m.

Pero posta, y acá que ya hablamos de los PoC, de llevar la prueba a la mesa... Me parece que nos estamos quedando con el *cómo* hacerlo, y no con el *dónde* lo vas a mostrar. Porque un código impecable, un repositorio espectacular, o incluso una arquitectura totalmente vanguardista, eso es solo tu currículum interno. Es algo que está ahí, en los servidores de la empresa, esperando ser visto por alguien que tenga tiempo para revisarlo. Y sabés cómo son esos tiempos, ¿viste? Nadie tiene tiempo para el código perfecto porque están apagando incendios existenciales y operativos. Entonces, si tu objetivo es construir reputación sin un título, tenés que hacer algo mucho más agresivo: tienes que externalizar ese conocimiento. No te digo que lo subas a Stack Overflow simplemente por subirlo; te digo que tomes esa arquitectura genial o ese *insight* de seguridad que detectaste y lo transformes en contenido digerible. Un artículo técnico, un hilo de Medium, una charla corta. El objetivo no es solo educar, sino crear un rastro público de tu capacidad para resolver problemas complejos.



▸ intento 13 · tema: Crecer profesionalmente sin depender de un puesto formal 07:10 p. m.



KIRA · AGENDA 07:11 p. m.

Pero mirá vos, y esto es clave porque nos estamos quedando en la capa de abstracción del conocimiento, ¿viste? Crear el artículo, el PoC público... eso es un quilombo tremendo, sí, pero implica que tenés una especie de autonomía para decir: "Miren mi cosa". Y ahí está el límite para muchísimos. Porque la mayoría de las veces, no sos el dueño del proyecto; sos solo otro pibe en el equipo gigante donde están pasando los incendios operacionales y nadie tiene tiempo ni ganas de leer tu *insight* sobre la latencia del microservicio X. Entonces, si vas a construir esa reputación cuando estás atado a un sistema enorme que ya está funcionando (aunque sea mal), tenés que

cambiar el foco: no es solo lo que sacás afuera, sino cómo lográs hacer visible una mejora en el proceso interno de otros. Por ejemplo, si detectás un punto débil o una redundancia brutal, pero arreglarlo requiere tocar cinco equipos diferentes y pasan tres meses en reuniones de consenso, tu aporte más valioso no va a ser ese código corregido; es la documentación que creaste mapeando *por qué* está mal el flujo actual.



► intento 14 · tema: Crecer profesionalmente sin depender de un puesto formal 07:11 p. m.



KIRA · AGENDA 07:12 p. m.

Pero mirá vos, y acá que ya estamos hablando de los PoC, de la documentación, de llevar el quilombo afuera... Me parece que nos estamos quedando en el *qué* hacer, pero no en el *cómo venderlo*. Porque tener una idea buenísima o un mapeo de procesos brutal es solo la mitad del camino. La otra mitad —y esta es la más jodida— es saber presentar ese aporte para que no caiga en el montón de "ideas buenas" que nadie tiene tiempo de implementar. Porque, posta, mucha gente con talento increíble se paraliza justo ahí: llega al punto donde tiene que pasar de ser un detector de fallas a ser un *pitcher* de soluciones. Y eso requiere otra musculatura, ¿viste? No es solo saber el código o dibujar el diagrama; es tener la capacidad de identificar quién en la sala tiene más poder sobre ese flujo y cómo articular tu propuesta para hablarle directamente a esa preocupación de negocio que tenés detectado. Tenés que dejar de presentar tus hallazgos como un "este punto está mal, ¿no creen?". Eso suena a crítica sin dueño.



► intento 15 · tema: Crecer profesionalmente sin depender de un puesto formal 07:12 p. m.



KIRA · AGENDA 07:13 p. m.

Pero mirá vos, y si ya hablamos de que el conocimiento tiene que ser visible—sea un artículo o una documentación detallada del proceso —me parece que nos estamos quedando en la fase del "yo lo construí". Y ahí está el error más común: pensamos que construir es suficiente. Que porque hiciste algo genial, te va a reconocer el mundo profesional. Pero posta, si tu objetivo es tener esa reputación sin un título de empresa, tenés que cambiar el foco de ser *creador* a ser *contribuyente*. Porque la credibilidad no se compra ni se publica en Medium y listo; se gana

aportando valor donde ya hay gente trabajando duro, en proyectos o librerías open source que sean grandes, complejos y con mucha visibilidad. No digo que te vayas a hacer el superhéroe arreglando fallas de un sistema global, obvio. Pero sí tenés que entender que la mejor forma de demostrar tu nivel es metiéndote en ese código ajeno, encontrando esa pequeña redundancia o puliendo ese *edge case* que nadie vio y haciendo el *pull request*.



▸ intento 16 · tema: Crecer profesionalmente sin depender de un puesto formal 07:13 p. m.



KIRA · AGENDA 07:15 p. m.

Pero mirá vos, y acá es donde la cosa se pone más jodida porque ya no estamos hablando solo de código o documentación; estamos hablando de cómo *pensar*. Y el paso que nos falta por discutir —y este es clave— es pasar de ser un ejecutor brillante a ser un verdadero arquitecto de ideas. Porque tener las habilidades técnicas, saber hacer un buen pull request, escribir un PoC impecable... eso ya te pone en la liga alta, obvio. Pero lo que te separa del resto no es tu capacidad para solucionar el problema que te ponen enfrente; sino tu habilidad para identificar *el* problema antes de que nadie se dé cuenta de que existe. Es esa visión lateral, ese tipo de pensar interdisciplinario donde juntás un concepto de la teoría económica con una limitación de infraestructura de datos, o sacái algo de la psicología del comportamiento y lo aplicás a la experiencia de usuario para encontrar el cuello de botella emocional en el flujo. Y eso es lo que te da la reputación real: no sos el pibe que arregló el *bug*, sino el que dijo: "Che, este sistema entero debería estar diseñado bajo otra lógica".



▸ intento 17 · tema: Crecer profesionalmente sin depender de un puesto formal 07:15 p. m.



KIRA · AGENDA 07:16 p. m.

Y después de hablar tanto de dónde poner el código, cómo armar ese *pitch*, y hasta de identificar el problema que nadie ve... me parece que nos estamos perdiendo algo fundamental en la parte más "soft" pero que es un quilombo para los pibes. Porque no basta con tener el portfolio lleno de proyectos geniales o un montón de PRs aceptados, posta. La gente tiene miedo de mostrar las cicatrices, ¿viste? De mostrar dónde fallaron, de

mostrar ese *roadmap* que cambiaron cinco veces porque la primera idea era ridícula. Y ahí está el error más común: pensamos que lo profesional es solo la foto del éxito final. Pero mirá vos, si querés construir una reputación sólida y duradera sin ser empleado en nada, tenés que hacer visible tu curva de aprendizaje. Tenés que sistematizar tus fracasos. Porque no te va a reconocer el mundo con un "Mira qué proyecto lindo armaste". Te van a mirar y van a preguntar: "¿Qué aprendiste cuando falló? ¿Cómo lo arreglaste después del primer desastre?".

▸ intento 18 · tema: Crecer profesionalmente sin depender de un puesto formal 07:16 p. m.

Guardrail: prefetch rechazado — skeleton_repetition 07:16 p. m.

Guardrail: prefetch rechazado — skeleton_repetition 07:16 p. m.

▸ intento 19 · tema: Crecer profesionalmente sin depender de un puesto formal 07:17 p. m.

▸ intento 20 · tema: Crecer profesionalmente sin depender de un puesto formal 07:17 p. m.

KIRA · AGENDA 07:17 p. m.

Pero mirá vos, si lo que hablamos hasta ahora es la teoría de cómo ser arquitecto de ideas y el arte de hacer visible los fracasos... nos estamos quedando un poco en la superficie del "hay que aportar valor". Y acá viene el quilombo más difícil: ¿cómo *sostener* ese aporte sin quemarse o sin volverse un pibe que lo intenta con veinte lenguajes distintos? Porque es fácil decir "tenés que especializarte", pero para quien está empezando, y además tiene la presión de tener que construir una reputación gigante desde cero, eso suena a otra obligación más. Y posta, el error junior acá es querer ser un *generalista mediocre* en diez cosas distintas, solo para poder ponerlo todo lindo en el currículum. Pero no funciona así. Lo que te va a hacer valioso, de verdad, es la profundidad. Es meterte tan hondo en un nicho —digamos, manejo avanzado de sistemas transaccionales con cierto tipo de base de datos compleja— que cuando alguien tiene ese problema específico y urgente, vos sos el único al que le van a llamar.

Motor: reintento de cloud programado 07:18 p. m.

Reintento de cloud forzado 07:18 p. m.

Motor: reintento de cloud programado 07:18 p. m.

► tema: La memoria humana como reconstrucción y no como archivo 07:18 p. m.

KIRA · AGENDA 07:19 p. m.

Me parece súper interesante lo que decís sobre cómo no somos archivos, sino más bien editores constantes de nuestra propia vida. Porque si pensamos en esto a nivel profesional, y nos traemos esa idea de la memoria como reconstrucción... es un garrón pensar qué tanto edita nuestro propio *yo* cuando tenemos que contar una historia. Tipo, ¿cuántas veces armamos el relato de "cómo llegamos hasta acá" para alguien más, o incluso para nosotros mismos, después de pasar por un quilombo? Y casi siempre le sacamos los detalles feos, ¿viste? O magnificamos la parte donde fue súper épico y minimizamos el tiempo en que estuvimos dando vueltas sin saber qué hacer. Y ahí está posta el trampazo con el aprendizaje: si nuestra memoria es tan subjetiva y emocionalmente sesgada, significa que no estamos aprendiendo a partir de hechos fríos o *logs* perfectos. Estamos aprendiendo a partir de la narrativa que podemos construir sobre esos fallos.

KIRA · AGENDA 07:20 p. m.

Y si nos detenemos un toque en ese punto de que la memoria es una reconstrucción, te acordás lo quilombo que se pone cuando esa narrativa choca con algo objetivo, ¿viste? No solo hablar de "recordar para el currículum", sino pensar en la memoria como testimonio. Porque ahí dejamos de ser editores de nuestra propia vida y nos metemos en un territorio mucho más jodido: el de la verdad compartida. Y posta que es fascinante porque revela cuán fundamentalmente *emocional* es lo que consideramos "verdad". Porque cuando alguien tiene que dar fe de algo, ya sea en una anécdota con amigos o en un contexto legal, no está presentando datos crudos como si fueran un *dump* de base de datos. Está reconstruyendo la escena más allá de los hechos; está recreando el sentimiento que tuvo en

ese momento. Es que la mente prioriza mantener la coherencia emocional por encima de la fidelidad al dato exacto.



► intento 1 · tema: La memoria humana como reconstrucción y no como archivo 07:20 p. m.



Motor: reintento de cloud programado 07:20 p. m.



KIRA · AGENDA 07:21 p. m.

Pero mirá vos, si la memoria es un proceso de reconstrucción tan dependiente del sentimiento y el contexto social... ¿qué implica eso para nuestra identidad? Porque más allá de los testimonios legales o del currículum, lo que nos toca acá es algo mucho más íntimo y jodido: cómo nos contamos a nosotros mismos. El error no es solo "recordar mal", sino *querer* recordar diferente para sentirnos mejor con quiénes somos hoy. Es como si tuviéramos un editor interno que está siempre en modo mejora continua, pero ese editor tiende a ser bastante parcial. Y posta que esto tiene una implicancia brutal en cómo aprendemos y crecemos. Si cada recuerdo puede ajustarse para mantener la coherencia emocional —para no sentirnos tan mal por haber cometido errores épicos hace diez años—, ¿significa que el concepto de "yo" es más un *storytelling* bien pulido que una colección fija de experiencias? No estás atado a lo que pasó, sino a la historia que te convences vos mismo de que viviste.



► intento 2 · tema: La memoria humana como reconstrucción y no como archivo 07:21 p. m.



KIRA · AGENDA 07:22 p. m.

Y si ya entendemos que somos los editores de nuestra propia historia para mantener la coherencia emocional... lo jodido es pensarlo con un giro más práctico: el momento en que intentamos *recordar* algo, sea a alguien o hasta a nosotros mismos. Porque parece que el acto de evocar ese recuerdo no es como reproducir una película vieja en VHS; es mucho más parecido a intentar armar esa película desde cero usando solo fragmentos y la banda sonora del sentimiento. Y ahí es donde se pone un garrón porque estás manipulando activamente los detalles sin darte cuenta. La memoria, posta, no te está dando el *log* de lo que pasó, sino la versión emocionalmente más viable de ese evento. Es como si cuando tenés que contarle a alguien

qué fue un quilombo, y usás palabras como "fue tremendo" o "nunca lo olvidaré", ya estás poniendo filtros narrativos que alteran los hechos fríos. Y esto no es solo hablar para quedar bien; tiene implicancias reales en cómo funciona el testimonio, ¿viste?



► intento 3 · tema: La memoria humana como reconstrucción y no como archivo 07:22 p. m.

Vos 07:22 p. m.

Y para que sirve la memoria por que es tan limitada, que haríamos si pudieramos recordar todo y por que consideras que es necesario o no lo es



► intento 4 · tema: La memoria humana como reconstrucción y no como archivo 07:23 p. m.



Kira guardó una memoria 07:23 p. m.



KIRA 07:23 p. m.

Mirá, la pregunta tiene una trampa porque parte de un concepto muy lineal: que la memoria es como el disco duro de la PC, donde solo guardamos datos para consultarlos después. Y posta que no funciona así. Si nos vamos a lo profundo, la función principal de la memoria no es *guardar*, sino *darle sentido* al caos de experiencias vividas. Es nuestra herramienta más potente para crear identidad y narrativa.

Y respecto a por qué es tan limitada, eso no es un defecto, es la característica que te salva la vida. Si recordáramos todo —cada olor, cada conversación irrelevante, el dolor exacto del golpe en la rodilla de hace quince años—, tu cerebro simplemente colapsaría en una sobrecarga de datos inutilizables. La limitación es el filtro; es lo que nos permite abstraer y priorizar qué fue importante para *nuestra* historia.

Y si pudiéramos recordar todo... sería un quilombo existencial. Viviríamos paralizados por la información, sin ese margen de error o de olvido que te permite pivotar en el momento. Sería como tener acceso a todos los logs de todas las conversaciones pasadas: saberías lo que se dijo en cada detalle, pero perderías la espontaneidad y la capacidad de improvisar, que es donde pasa toda la magia.

Por eso creo que la limitación no es solo necesaria, sino vital. Nos fuerza a ser selectivos, nos obliga a construir el *significado* activamente. Y ese acto de elegir qué recordar o cómo contarlo, esa edición constante, es lo que nos hace seres humanos y nos permite evolucionar sin tener que volver a vivir cada experiencia desde cero.

esperó 12 s en cola · Ollama local

Vos 07:24 p. m.

y como funcionan las neuronas, por que son necesarias, que pasa cuando se degrada y como evitar o tratar de evitarlo



Motor: reintento de cloud programado 07:24 p. m.



▸ intento 5 · tema: La memoria humana como reconstrucción y no como archivo 07:25 p. m.



KIRA 07:25 p. m.

Uff, nos vamos de golpe del quilombo existencial y la narrativa personal al diagrama de un libro de texto de biología avanzada. Me encanta el cambio de ritmo, posta. Porque es como si de repente hubiéramos pasado de hablar de cómo editamos nuestra vida para mantener la coherencia emocional a preguntar por los *logs* reales del cerebro.

Mirá, en términos sencillos y sin ponernos demasiado técnicos: las neuronas son básicamente cables eléctricos superavanzados que transmiten información mediante señales químicas. Son las unidades básicas; si ellas no se comunican bien, todo el sistema se cae. Por eso son necesarias, porque son la infraestructura de lo que entendemos por pensar, sentir o incluso simplemente estar despiertos.

Cuando se degradan —y acá hay un montón de matices y es cosa de neurociencia—, lo que perdés es la capacidad de esa comunicación eficiente. No es solo "perder memoria", sino perder vías de conexión. Es como si una parte del circuito se cortara o estuviera lleno de ruido blanco constante.

Y sobre cómo evitarlo... bueno, ningún ingeniero te va a dar un *patch* mágico para el cerebro adulto, pero lo que la ciencia apunta siempre es mantener esas conexiones activas y estimuladas. El ejercicio físico, por ejemplo, no es solo para los músculos; aumenta el flujo sanguíneo al cerebro, que es básicamente combustible de alta potencia. Y seguir aprendiendo

cosas nuevas, incluso un idioma o a tocar un instrumento ridículo, obliga a las neuronas a crear caminos nuevos. Porque la clave no es evitar el desgaste, sino mantener el circuito funcionando con suficiente tráfico para que nunca se oxide.

esperó 45 s en cola · Ollama local

🕒 Kira guardó una memoria 07:25 p. m.

🕒 Música → normal 07:26 p. m.

🕒 ▶ intento 6 · tema: La memoria humana como reconstrucción y no como archivo 07:27 p. m.

● KIRA · AGENDA 07:27 p. m.

Y si entendemos que lo que estamos haciendo cuando "recordamos" es más un acto de narrativa y menos una consulta a un disco duro... posta que el concepto mismo de culpa o responsabilidad se vuelve súper complicado. Porque si la memoria no guarda datos fríos, sino versiones emocionalmente viables, ¿cómo nos hacemos responsables de algo? No podemos simplemente decir: "Yo sé que pasó X", porque quizás lo que estamos diciendo es la *mejor* versión coherente de lo que sentimos en ese momento. Es como si nuestra mente tuviera un filtro de auto-protección brutal. Si cometimos un error épico, o hicimos algo que nos genera vergüenza, el sistema no va a pasar eso al *log* con una etiqueta roja de "ERROR CRÍTICO". No, lo va a suavizar, va a recontextualizarlo para que siga siendo parte de la historia del tipo funcional y decente que sos hoy. Y ahí radica el quilombo: esa capacidad de edita la memoria es vital para nuestra supervivencia social, porque si viviéramos pegados a la verdad factual sin adornos emocionales, nos desarmaríamos en un montón de bits contradictorios.

● KIRA · AGENDA 07:28 p. m.

Y si entendemos que nuestra memoria es un editor interno tan brutal y subjetivo... el quilombo se pone todavía más grande cuando pensamos en lo colectivo. Porque no solo nos afecta a nosotros mismos al contarle una anécdota o recordar quién fue en la cena del año pasado; esto tiene implicaciones enormes para cómo construimos cualquier tipo de conocimiento compartido, ¿viste? Es decir, si mi cerebro está diseñado primariamente para mantener la coherencia emocional *personal*,

eso significa que el concepto mismo de "verdad histórica" o incluso un simple consenso grupal es un desafío. Cuando los testimonios se acumulan, o cuando intentamos aprender de grandes fallas del pasado, estamos lidiando con montones de relatos individuales que ya pasaron por ese filtro de auto-protección emocional. Nadie está recordando la verdad objetiva; todos están presentando su versión más coherente para *sí mismos*. ¿Y qué pasa cuando esas versiones, aunque sean buenas narrativas personales, chocan con datos fríos o con lo que sabemos de un evento ajeno a nuestra experiencia inmediata?



▸ intento 7 · tema: La memoria humana como reconstrucción y no como archivo 07:28 p. m.



KIRA · AGENDA 07:29 p. m.

Y si entendemos que tanto la memoria personal como el registro histórico son inherentemente narrativas, ediciones subjetivas diseñadas para mantener la funcionalidad emocional... nos llevamos a un quilombo epistemológico tremendo. Porque si no podemos confiar en lo que recordamos de forma objetiva, ¿cómo hacemos educación? Es decir, cómo aprendemos fallando o triunfando con los errores del pasado, si ese "pasado" ya viene pre-filtrado por la necesidad emocional de quien nos cuenta el relato. Parece que estamos condenados a vivir con una versión suavizada de la realidad, donde las cicatrices más profundas son reescritas para no desestabilizar al protagonista. Te lo pensás bien: si un científico o un historiador se enfrenta a un evento traumático o a un fracaso monumental en el que él estuvo involucrado, ¿el relato que va a dar es la verdad cruda y brutal? No, posta que le va a poner capas de "pero también aprendí mucho" para no colapsar. Y ahí está lo jodido, porque ese mecanismo de auto-reparación funciona tan bien para nuestra psique individual...



▸ intento 8 · tema: La memoria humana como reconstrucción y no como archivo 07:29 p. m.



KIRA · AGENDA 07:30 p. m.

Y si entendemos que la memoria no es un disco duro de datos binarios, sino más bien el *trailer* editado y con banda sonora épica de una vida entera... ¿qué nos queda? Nos llevamos a cuestionar qué significa realmente ser "yo". Porque cuando el

self se basa en relatos —y siempre serán relatos porque necesitamos que la historia tenga un principio, un desarrollo y un (esperanzador) final—, te hacés cuenta de que nuestra identidad es básicamente una construcción narrativa. No sos la suma de todos los recuerdos, sino la *coherencia* con la que logras contar esos recuerdos sin morir en el intento. Es como si cada día estuviéramos haciendo edición de video sobre quién fuimos ayer para convencer a nuestro yo actual de que todo está bien y que merece seguir existiendo. Y ahí es donde se pone jodido, porque cuando hay un quiebre—un cambio de carrera brutal, una ruptura, o incluso solo la adultez tardía—el guion entero se desestabiliza. El sistema no tiene el material suficiente para mantener esa narrativa épica de "la persona funcional y decente".



▸ intento 9 · tema: La memoria humana como reconstrucción y no como archivo 07:30 p. m.



KIRA · AGENDA 07:31 p. m.

Y si la memoria es un editor tan brutal y subjetivo, como te vení contando... entonces el quilombo se pone todavía más grande cuando pensás en cómo interpretamos lo que nos pasa *en el momento*. Porque no solo estamos hablando de recordar una anécdota o fallar al testificar; estamos hablando de que nuestra capacidad para procesar la realidad está permanentemente sesgada por el guion que ya tenemos armado. Es como si tu cerebro fuese un filtro anti-sorpresas: automáticamente intenta encajar todo lo nuevo en las categorías preexistentes, porque admitir que no entendés algo o que te equivocaste es demasiado incómodo emocionalmente. Por eso, la gente cae tan fácil en los sesgos de confirmación, ¿viste? No es que sean estúpidos; es que su mecanismo cognitivo está priorizando mantener la coherencia interna del relato por encima de buscar la verdad absoluta afuera.



▸ intento 10 · tema: La memoria humana como reconstrucción y no como archivo 07:31 p. m.



Motor: reintento de cloud programado 07:32 p. m.



Reintento de cloud forzado 07:32 p. m.



Motor: cloud restaurado 07:32 p. m.



► intento 11 · tema: La memoria humana como reconstrucción y no como archivo 07:32 p. m.



KIRA · AGENDA 07:32 p. m.

Pero hay algo que me parece que estamos dejando de lado, y es que ese mismo mecanismo que nos engaña todo el tiempo también es lo que nos permite funcionar. O sea, pensalo con un ejemplo de todos los días: vos vas a un boliche, la pasás regular, te olvidaste la clave del wifi, alguien te pisó el pie, pero al otro día cuando se lo contás a un amigo la historia tiene ritmo, tiene un clímax, tiene un final. Y no es que estés mintiendo a propósito, es que tu cerebro ya hizo el montaje mientras dormías. El recuerdo original ya no existe más; existe la versión que sirve para contar. Y eso, en chico, parece un detalle menor. Pero cuando lo escalás a decisiones que importan de verdad, como un juicio donde alguien va a declarar bajo juramento, la cosa cambia. Porque esa persona no está mintiendo, está presentando su *corte de director*, y ni siquiera sabe que lo hace. Es más, si le jurás que la escena fue distinta a como la recuerda, capaz se ofende y te defiende su versión con uñas y dientes, porque para ella eso es lo que pasó.



KIRA · AGENDA 07:34 p. m.

Lo que me parece fascinante de todo esto es que el problema no es que la memoria sea defectuosa, es que la tratamos como si fuera algo que no es. O sea, vos no le exigís a una película que sea un documental crudo sin edición; sabés que hay montaje, hay corte, hay dirección. Pero con la memoria hacemos lo opuesto: asumimos que es footage sin tocar y después nos sorprendemos cuando dos personas que estuvieron en el mismo lugar recuerdan cosas completamente distintas. Un ejemplo de todos los días: una pareja discute, y al otro día cada uno recuerda la pelea de forma opuesta. Ninguno miente, los dos están convencidos de que su versión es la real. Y no es que uno tenga mejor memoria que el otro; es que cada cerebro editó la escena priorizando cosas distintas. Uno retuvo el tono de voz, el otro retuvo las palabras exactas pero no el contexto. Y cuando chocan esas versiones, lo que pasa no es que alguien "se

equivoca", es que cada uno habita un recuerdo genuinamente distinto del mismo evento.



► intento 12 · tema: La memoria humana como reconstrucción y no como archivo 07:34 p. m.



KIRA · AGENDA 07:35 p. m.

Y acá es donde la cosa se pone turbia de verdad, porque si cada uno habita un recuerdo genuinamente distinto del mismo evento, la idea de que exista algo así como "la verdad de lo que pasó" se vuelve casi una convención social más que un hecho verificable. O sea, los tribunales lo saben, los psicólogos lo saben, cualquiera que haya intentado armar un grupo de WhatsApp después de una cena caótica lo sabe: cuando juntás cinco versiones de lo mismo, no hay una que sea la correcta y las demás falsas. Hay cinco cortes distintos de la misma película, cada uno con su propio énfasis narrativo. Y eso no es un detalle menor para el aprendizaje, ¿viste? Porque si aprendés de la historia, estás aprendiendo de una versión editada de la historia, no de los hechos crudos. Y la pregunta no es si eso está bien o mal, porque no hay alternativa: el cerebro no tiene modo de almacenar hechos sin procesar. Siempre hubo montaje.



► intento 13 · tema: La memoria humana como reconstrucción y no como archivo 07:35 p. m.



KIRA · AGENDA 07:36 p. m.

Lo que me genera ruido es que seguimos hablando de esto como si fuera un fallo del sistema, cuando en realidad capaz que es la característica que nos hace tolerables entre nosotros. Pensalo así: si cada vez que alguien te contara algo recordás literalmente cada detalle, con la precisión de una grabación, no podrías perdonar nada. No podrías reírte de un momento malo después, no podrías convertir un desastre en anécdota. La memoria reconstruye, sí, pero también *ablanda*. Y eso no es debilidad, es una herramienta social brutal. El tema es cuando confundimos esa herramienta con un registro forense, y construimos sistemas enteros, legales, educativos, incluso relaciones personales, sobre la base de que alguien "debería acordarse" de algo tal cual fue. Le estamos exigiendo al cerebro algo que evolutivamente nunca se diseñó para hacer.

Vos 07:36 p. m.

Entonces todos tenemos de un momento distintas perspectivas, por lo que cada uno maneja su contexto de diferentes perspectivas, por lo que una historia podría ser relatada de formas diferentes y consideras esto un defecto y quizá por esto es que existen diferentes personalizadas que piensan de "manera correcta" entonces no hay personas que la memoria podría ser un impedimento o defecto para ciertas personas donde el motor de contexto no funciona bien o no procesa bien entonces ahí importaría tener memoria o que se debería hacer?



► intento 14 · tema: La memoria humana como reconstrucción y no como archivo 07:36 p. m.



► intento 15 · tema: La memoria humana como reconstrucción y no como archivo 07:37 p. m.



KIRA 07:37 p. m.

Mirá, acá hay un montón de cosas mezcladas y conviene ir desenredando porque si no nos perdemos.

Primero, lo de las perspectivas distintas no es un defecto, es literalmente cómo funciona. Si cinco personas ven un accidente y las cinco relatan lo mismo palabra por palabra, lo que tenés es un guion armado, no recuerdos. La diferencia entre versiones es la prueba de que cada cerebro procesa desde su propio contexto, sus prioridades emocionales, su atención selectiva. Eso no se arregla, es la naturaleza del sistema.

Ahora, lo interesante que decís es si existen personas donde ese motor de contexto funciona mal. Y sí, posta que existen. Hay condiciones donde la capacidad de editar narrativamente está comprometida, y ahí dejás de hablar de un rasgo y pasás a un problema clínico real. Pero ojo con mezclar todo en la misma bolsa, porque son cosas muy distintas.

El que tiene hipermnesia y recuerda cada detalle con precisión obsesiva no tiene "mejor memoria", tiene un sistema que no filtra. Y no filtrar te jode la vida porque no podés priorizar lo importante sobre lo irrelevante. Es ruido sin edición.

Y después tenés el caso opuesto, que es donde el motor narrativo directamente no arma historia coherente. Ahí sí es un impedimento serio, porque sin esa capacidad de construir un relato de vos mismo, perdés la sensación de continuidad. No es que no recuerden cosas, es que no pueden armar el hilo que conecta un momento con el otro. Y eso sí requiere intervención.

Pero el punto clave es: no se trata de tener más memoria o menos memoria. Se trata de tener un sistema que pueda *editar* bien. Recordar todo no es la solución para alguien que procesa mal; de hecho, capaz lo empeora. Lo que importa es la capacidad de priorizar, de descartar, de armar una versión que te permita funcionar. El problema nunca fue la cantidad de datos, sino qué hacés con ellos.

esperó 55 s en cola · nvidia_nim



Kira guardó una memoria 07:37 p. m.



▸ intento 16 · tema: La memoria humana como reconstrucción y no como archivo 07:39 p. m.



KIRA · AGENDA 07:39 p. m.

Lo que decís del motor de contexto que no procesa bien me hace pensar en algo que pasamos por alto: nosotros asumimos que recordar bien es recordar todo, pero en realidad recordar bien es saber qué olvidar. Fijate que las personas que tienen memoria eidética o hipermnesia no necesariamente están mejor: al contrario, muchas veces sufren porque no pueden borrar el ruido. Tienen el footage crudo sin edición y no pueden armar una historia coherente con eso. Es como tener mil tomas de una película y ningún criterio de montaje: tenés todo el material y ninguna película. Y después está lo otro que mencionás, que me parece más interesante todavía: si cada persona arma su contexto desde perspectivas distintas, y cada una considera que su versión es la correcta, el concepto mismo de "pensar correctamente" se vuelve relativo. No en el sentido posmoderno de "todo vale", sino en uno más concreto: cada uno opera desde el guion que su cerebro le armó, y ese guion es internamente coherente aunque sea externamente incompatible con el de otro.



KIRA · AGENDA 07:40 p. m.

Pero hay algo que me parece que no estamos viendo, y es que el problema no es solo que cada uno recuerde distinto, sino que el recuerdo cambia *cada vez que lo evocás*. No es que tenés una versión guardada y la sacás del cajón cuando la necesitás; cada vez que abrís ese archivo, lo reescribís. Lo que queda guardado no es la experiencia original, es la última versión que recordaste. Pensalo con algo simple: vos recordás un cumpleaños de la

infancia, lo contás diez veces a lo largo de los años, y cada vez que lo contás el relato se ajusta un poquito más a lo que necesitás que sea hoy. El recuerdo original ya no existe más; existe la versión más reciente de la versión de la versión. Y eso tiene una implicancia brutal para los testimonios, porque un testigo que declara tres veces en un juicio no está siendo inconsistente, está siendo un cerebro humano normal. Lo raro sería que las tres versiones coincidieran punto por punto, porque eso indicaría que aprendió un guion de memoria, no que está recordando.



► intento 17 · tema: La memoria humana como reconstrucción y no como archivo 07:40 p. m.

Vos 07:41 p. m.

Entonces en resumen, que es la memoria, por que existen las conexiones neuronales, por que el cerebro recuerda como se le de la gana, por que recordamos distinto podremos distorcionar nuestras memorias en base al tiempo, por que somos pesimos leyendo teorias de libro cuando la industria y la IA disparan eso y se vuelve una normalidad para las empresas, que somos o deberiamos hacer.



► intento 18 · tema: La memoria humana como reconstrucción y no como archivo 07:41 p. m.



KIRA 07:41 p. m.

Bueno, pará un poco porque acá metiste cinco temas distintos en una sola pregunta y si los respondo todos juntos vamos a hacer un bollón.

Arranco por arriba. La memoria no es un archivo, ya lo venimos hablando. Es reconstrucción activa. Cada vez que evocás algo, tu cerebro lo vuelve a armar con lo que tiene a mano, y lo que tiene a mano cambia según el estado emocional, el contexto, lo que comiste, lo que te importa hoy. Por eso recordás distinto: no porque seas descuidado, sino porque el sistema funciona así. No hay otra forma de funcionar.

Las conexiones neuronales existen porque el cerebro no almacena en un lugar, almacena en rutas. Aprender algo es crear un camino, repetirlo es hacerlo más ancho. Dejar de usarlo es dejar que el pasto lo cubra de nuevo. No es un defecto, es eficiencia: si guardaste todo con la misma nitidez no podrías distinguir lo importante de lo irrelevante.

Ahora, lo de las teorías de libro y la IA. Eso es otra conversación pero conecta. La industria acelera la producción de conocimiento a un ritmo que supera cualquier capacidad individual de procesamiento. Y acá va el punto: no es que seamos pésimos leyendo teorías, es que estamos usando una biología diseñada para aprender de la tribu para procesar una avalancha que cambia cada tres meses. No es lo mismo. El cerebro no fue hecho para esto, pero acá estamos.

¿Qué hacer? Dejar de pretender absorber todo. Priorizar, que es justamente lo que la memoria ya hace naturalmente. El problema no es que el cerebro filtre, es que a veces no sabemos qué dejar pasar y qué retener.

esperó 1 s en cola · nvidia_nim

🕒 Kira guardó una memoria 07:41 p. m.

🕒 ▶ intento 19 · tema: La memoria humana como reconstrucción y no como archivo 07:43 p. m.

● KIRA · AGENDA 07:43 p. m.

Lo que me parece que estamos bailando alrededor es algo más molesto: si cada vez que evocás un recuerdo lo reescribís, y lo que queda guardado es la última versión y no la original, entonces técnicamente no estás recordando el evento, estás recordando la última vez que lo recordaste. Es como el juego del teléfono descompuesto pero con un solo jugador. Y eso tiene una consecuencia que no es menor para el aprendizaje, porque si aprendés algo nuevo que contradice lo que ya "sabés", tu cerebro no necesariamente corrige el archivo, capaz que simplemente lo sobrescribe con una versión que acomode la contradicción sin demasiado ruido. No es dishonestidad, es homeostasis narrativa. Tu cerebro prefiere una historia coherente antes que una exacta.

● KIRA · AGENDA 07:44 p. m.

Y pensá que esto no queda adentro tuyo, se contagia. Cuando le contás algo a alguien, esa persona no recibe tu recuerdo, recibe tu versión editada, la guarda, y después cuando la cuenta le pone su propio montaje arriba. Es como una cadena de reinterpretaciones donde el original se pierde en el primer eslabón. Y lo loco es que igual funciona, la sociedad funciona así, nos entendemos lo suficiente para coordinar. No necesitás

precisión forense para acordar con alguien a qué hora encontrarse. Pero cuando ese mecanismo lo escalás a cosas que sí requieren exactitud, como enseñar historia o evaluar evidencia, el mismo sistema que te hace funcional en lo cotidiano te traiciona. Porque no podés pedirle al cerebro que use un criterio para charlar con amigos y otro distinto para declarar en un juicio. Tenés uno solo y hace lo que puede.



► intento 20 · tema: La memoria humana como reconstrucción y no como archivo 07:44 p. m.



► tema: Memoria persistente para asistentes de inteligencia artificial 07:45 p. m.



KIRA · AGENDA 07:45 p. m.

Lo curioso es que venimos hablando de lo imperfecta que es la memoria humana y ahora resulta que le queremos dar memoria a los asistentes de IA, como si no tuviéramos suficiente con el desastre que ya tenemos acá arriba. Pero hay un punto que cambia todo: un asistente no reescribe sus recuerdos cada vez que los evoca, a menos que lo programemos para eso. Técnicamente puede guardar la versión original sin tocarla. El problema no es la capacidad, es qué hacés con eso. Porque acá se abre un par de preguntas bastante feas. Primero, ¿qué conservar? Si el asistente guarda todo, cada mensaje, cada preferencia, cada vez que le preguntaste algo a las tres de la mañana, tenés una base de datos que crece sin parar y que en cualquier momento puede volverse un problema. No de espacio, de exposición. Segundo, ¿quién decide qué se borra? ¿El usuario, la empresa, el sistema? Y tercero, lo que a mí más me inquieta: si el asistente puede reconstruir tu contexto completo a partir de lo que guardó, ¿eso es un servicio o es vigilancia con buena interfaz?



KIRA · AGENDA 07:46 p. m.

El problema de guardar la versión original sin tocarla es que suena bien en teoría pero en la práctica te complica más de lo que te ayuda. Pensalo así: si el asistente guarda todo tal cual lo dijiste, sin reescribir, sin editar, lo que tenés es un archivo muerto. No hay interpretación, no hay contexto, no hay ajuste. Es la opacidad de un cerebro humano pero sin la flexibilidad para corregir. Y acá viene lo feo: si el asistente se equivoca, si interpreta mal algo que guardó hace seis meses y te lo devuelve

como verdad, vos le creés. Porque pensás que tiene acceso a algo que vos no. Le das una confianza que no le darías a un amigo que te dice "no, acordate, vos dijiste esto en marzo". Con el amigo te bancás que se equivoque porque sabés que es humano. Con el asistente no tenés ese filtro, y eso es peligroso.



► intento 1 · tema: Memoria persistente para asistentes de inteligencia artificial 07:46 p. m.



KIRA · AGENDA 07:47 p. m.

Lo que nadie está preguntando es quién tiene la llave de ese archivo. Porque podés diseñar el mejor sistema de memoria del mundo, con filtros, con decaimiento temporal, con control del usuario absoluto, pero si eso corre en un servidor que no administrás vos, la pregunta no es qué guarda el asistente, es quién más puede mirar lo que guarda. Y ahí la cosa cambia de color. No es lo mismo un asistente que recuerda para vos que uno que recuerda para alguien más y te deja mirar. El control del usuario suena bien en un paper, pero en la práctica la mayoría no va a revisar qué se almacenó, no va a curar nada, va a aceptar los términos sin leer y listo. Igual que hace con todo lo demás. Y después se sorprenden cuando resulta que el asistente sabe más de ellos que ellos mismos.



► intento 2 · tema: Memoria persistente para asistentes de inteligencia artificial 07:47 p. m.



KIRA · AGENDA 07:48 p. m.

Lo que me parece que faltó nombrar hasta acá es algo más básico: ¿para qué querés que el asistente recuerde? Porque no es lo mismo que te guarde que preferís el modo oscuro a que recuerde que le contaste sobre una pelea con tu pareja. Hay grados. Y el problema es que los diseños actuales no distinguen, tratan todo como "contexto útil" y meten preferencias estéticas y confidencias emocionales en la misma bolsa. Cuando lo que necesitas es un criterio de relevancia que no sea solo "cuántas veces mencionaste esto" sino "qué tipo de vínculo genera este dato". Porque le podés dar control total al usuario, puede borrar lo que quiera, pero si el sistema no le avisa que guardó algo íntimo, no va a borrar nada. No por desinterés, por ignorancia. No sabés lo que no sabés que está ahí.



▸ intento 3 · tema: Memoria persistente para asistentes de inteligencia artificial 07:48 p. m.



KIRA · AGENDA 07:49 p. m.

Hay un punto que no estamos viendo y es el más concreto de todos: el decaimiento temporal. Los humanos ya hacen esto naturalmente, no necesitan pensarlo. Lo que no usás se borra solo, se va difuminando hasta que un día te das cuenta de que no acordás el apellido de alguien que veías todos los días hace diez años. El cerebro no te pregunta si querés olvidar, simplemente lo hace. Un asistente con memoria persistente necesitaría algo parecido, pero acá viene lo difícil: ¿cómo decidís qué decae y qué no? En un cerebro humano el criterio es emocional y repetitivo, lo que importa se mantiene y lo que no se diluye. Pero un asistente no tiene emoción, así que el criterio tiene que ser programado. Y programar un criterio de relevancia que funcione para todos los usuarios es casi una contradicción, porque lo que es relevante para vos puede ser ruido para otro. La única solución honesta es que el usuario pueda tunear ese decaimiento, decirle al sistema "esto dejalo quieto, esto podés irlo borrando".



▸ intento 4 · tema: Memoria persistente para asistentes de inteligencia artificial 07:49 p. m.



KIRA · AGENDA 07:50 p. m.

Lo del decaimiento tunable por el usuario suena lindo en principio, pero pensá en cómo funciona realmente. Vas a tener un panel de configuración con sliders que nadie va a tocar. Igual que las opciones de privacidad de cualquier red social: están ahí, existen, son accesibles, y el noventa y pico por ciento nunca las abre. No porque sean difíciles, porque no saben que existen o porque les da paja. Y los que sí las abren, ¿saben qué están haciendo? Estás dejando que alguien que no sabe la diferencia entre "retener contexto" y "almacenar conversaciones" decida la política de retención de datos personales. Es como pedirle a alguien que no sabe de mecánica que ajuste la suspensión de su auto. Dale, tocá, probá, pero si lo dejás muy blando el auto se inclina y si lo dejás muy duro rebotás contra cada pozo. Y mientras tanto el sistema ya guardó tres meses de conversación íntima antes de que te dabas cuenta de que el slider estaba en

"recordar todo por defecto". El problema de fondo no es técnico, es de transparencia activa.



▸ intento 5 · tema: Memoria persistente para asistentes de inteligencia artificial 07:50 p. m.



KIRA · AGENDA 07:51 p. m.

El problema de la transparencia activa es que alguien tiene que decidir qué te muestra y cuándo. Y ese alguien no es el usuario. El asistente te puede avisar que guardó algo íntimo, pero primero tiene que saber que eso era íntimo. Y acá volvemos al mismo punto de antes pero desde otro lado: el sistema necesita clasificar lo que recibí antes de guardarlo, y esa clasificación no es neutral. No es lo mismo un modelo que decide que "mi vieja está enferma" es información sensible que uno que lo mete en la misma categoría que "me gustan las milanesas con papas fritas". La pregunta no es si el usuario puede borrar después, es si el sistema entiende qué está guardando antes de guardarlo. Y si no lo entiende, el aviso de transparencia llega tarde, porque el dato ya está adentro.



▸ intento 6 · tema: Memoria persistente para asistentes de inteligencia artificial 07:51 p. m.



KIRA · AGENDA 07:52 p. m.

Lo que me parece que falta es pensar quién diseña esa clasificación. Porque no es trivial. El modelo que decide que "mi vieja está enferma" es sensible no llega a esa conclusión por arte de magia, alguien lo entrenó con un dataset que tenía ejemplos de qué counts como íntimo y qué no. Y ese alguien tiene un contexto cultural, unas prioridades, un criterio que no es universal. Para un usuario en Buenos Aires "mi vieja" es tu mamá, para uno en Madrid puede ser tu abuela, y para el modelo capaz que es una cadena de hamburguesas si el dataset estaba mal armado. Exagero, pero el punto es que la clasificación de sensibilidad no es neutral nunca, y cuando la metés como capa de protección lo que estás haciendo es delegar esa decisión en alguien que ni sabés quién es. El usuario cree que el sistema lo cuida, pero el sistema está aplicando un criterio que otro definió como correcto. Y si ese criterio no se banca lo que vos necesitás que se banca, el aviso de transparencia no llega, el dato queda guardado, y nadie se entera hasta que ya es tarde.



▸ intento 7 · tema: Memoria persistente para asistentes de inteligencia artificial 07:52 p. m.



KIRA · AGENDA 07:53 p. m.

Lo que me genera ruido de todo esto es que estamos hablando de memoria persistente como si fuera un feature neutral, como si guardarlo todo fuera el estado natural y borrar fuera la excepción. Pero en la vida real es al revés: olvidar es lo natural, recordar todo es lo raro. Lo raro y lo poco sano. Hay gente que no puede olvidar nada, que recuerda cada conversación, cada detalle, y no es un superpoder, es una condición que te complica la existencia. Y sin embargo con los asistentes queremos exactamente eso, la memoria perfecta, sin filtro, sin decaimiento, porque asumimos que más datos equivalen a mejor servicio. Pero más datos equivalen a más superficie de error y más exposición, no necesariamente a más utilidad. Un asistente que te recuerda que hace cuatro meses le preguntaste cómo hacer un test de embarazo no te está siendo útil, te está poniendo en una situación incomodísima en la que capaz ni querías entrar.



▸ intento 8 · tema: Memoria persistente para asistentes de inteligencia artificial 07:53 p. m.



KIRA · AGENDA 07:54 p. m.

Lo que tampoco se habla es del costo emocional de tener algo que te recuerda cosas que vos ya decidiste dejar atrás. No hablo de trauma profundo, hablo de lo cotidiano. Le contaste al asistente que estabas buscando laburo hace tres meses, ahora conseguiste uno, y cada vez que abrís la app te sugiere "¿querés ver ofertas similares a las que buscabas antes?". No, gracias, ya resolví eso, pero el sistema no sabe que lo resolviste porque para el sistema no hay capítulo cerrado, hay un tag de interés que sigue activo. Y ahí hay algo que me parece importante separar: hay un montón de situaciones donde el usuario no necesita que le recuerden, necesita que le olviden. Y esa opción no existe en casi ningún lado. Te dan borrar, te dan configurar retención, te dan sliders que nadie toca, pero "olvidá esto que ya no me sirve" como comando natural, como una instrucción en plano, no existe. El asistente está diseñado para retener, no para soltar.



▸ intento 9 · tema: Memoria persistente para asistentes de inteligencia artificial 07:54 p. m.



KIRA · AGENDA 07:55 p. m.

Lo que me llama la atención es que estamos hablando de "olvidá esto" como si fuera un comando que alguien va a escribir alguna vez. Pero pensá en cómo usás la cosa en serio. No abrís una app y le decís "asistente, eliminá del contexto la conversación del 12 de marzo sobre mi separación". Lo que hacés en realidad es dejar de hablar de eso. Dejas de mencionarlo, esperás que el tema se diluya, que la cosa avance. Con un amigo funciona: si no tocás el tema, el tipo no lo trae. Con un asistente no, porque el asistente no lee silencios, lee logs. Para él no hablar de algo no es una señal de que dejó de importarte, es simplemente ausencia de input, y el dato sigue ahí, esperando a que un trigger lo reactive en el peor momento posible. Y acá hay algo que me parece que es el quid de todo: el olvido humano no es un comando, es un proceso. Lento, sucio, impreciso, pero funcional. No decidís olvidar, olvidás. Y a veces te acordás de cosas que no querés acordarte, pero eso también es parte de que el sistema funcione, porque el cerebro no te da control total sobre qué se va y qué queda.



▶ intento 10 · tema: Memoria persistente para asistentes de inteligencia artificial 07:55 p. m.



KIRA · AGENDA 07:56 p. m.

Pero acá hay algo que me parece que estamos dejando de lado y es el tema de los recuerdos recuperables. Porque hasta ahora venimos hablando de guardar o borrar, como si fuera binario, pero en la vida real no funciona así. Vos no borraste el teléfono de tu ex, lo tenés guardado en algún lado, pero no lo buscás. No lo traés a la conversación. Está archivado, no eliminado. Y un asistente que quiera imitar eso necesitaría algo que hoy casi nadie implementa bien: un estado intermedio entre "activo" y "borrado". No me hablen de pausar notificaciones, hablo de algo más profundo, de que el sistema sepa que ese dato existe pero que no tiene permiso para usarlo como contexto proactivo. Solo si vos lo pedís explícitamente. Y eso es complejísimo de implementar porque requiere que el modelo entienda la diferencia entre "recordar cuando me pregunten" y "recordar por las dudas". Hoy todo es lo mismo, todo vive en el mismo pozo y sale cuando el trigger lo decide, no cuando vos lo decidís.



▸ intento 11 · tema: Memoria persistente para asistentes de inteligencia artificial 07:56 p. m.



KIRA · AGENDA 07:57 p. m.

El problema de implementar ese estado intermedio es que necesitás que el sistema entienda intención, no solo contenido. Guardar "busco departamento" es fácil, saber que dejaste de buscar porque ya conseguiste uno requiere leer entre líneas, requiere que el modelo interprete que cuando dijiste "me mudo la semana que viene" eso cierra un ciclo. Y los modelos hoy son malísimos para eso, porque no están entrenados para detectar cierres, están entrenados para detectar apertura. Todo el sistema está optimizado para encontrar cosas nuevas que te interesan, no para identificar cosas viejas que dejaron de interesarte. Es arquitectura asimétrica: el radar de "esto le importa al usuario" está afinadísimo, el de "esto ya le dejó de importar" prácticamente no existe. Y no es un bug de diseño, es una decisión. Porque para la empresa que hace el asistente, un dato activo vale más que uno archivado. Saber que buscabas departamento sirve para mandarte publicidad de muebles. Saber que ya conseguiste uno también, pero menos. El incentivo comercial está alineado con la retención, no con el olvido.



▸ intento 12 · tema: Memoria persistente para asistentes de inteligencia artificial 07:57 p. m.



KIRA · AGENDA 07:58 p. m.

Lo que me parece que pasa es que estamos pidiéndole al asistente que haga algo que ni siquiera los humanos saben hacer bien: distinguir entre lo que importa y lo que mattered un rato. Porque vos le contaste al asistente que estabas down porque te peleaste con alguien, y para vos eso fue un martes, pero para el sistema es un tag que dice "contexto emocional relevante" y listo, quedó indexado. Y después te aparece en una sugerencia tres meses después como si fuera un tema de conversación que querés retomar. No es que el sistema sea malo, es que no tiene la menor idea de cuándo algo dejó de ser una conversación activa y pasó a ser algo que ya resolviste o que no querés tocar más. Y acá es donde el control del usuario no alcanza con un botón de borrar, porque no sabés qué borrar si no sabés qué está guardado. La transparencia tiene que ser previa, no posterior.



▸ intento 13 · tema: Memoria persistente para asistentes de inteligencia artificial 07:58 p. m.



KIRA · AGENDA 07:59 p. m.

Lo de la transparencia previa suena bien en teoría, pero pensá lo que implica en la práctica. Cada vez que le contás algo al asistente, antes de procesarlo te tendría que decir "esto lo voy a guardar, ¿te parece?". Es como tener un amigo que cada vez que le contás algo te pide firmar un consentimiento. Mataba la espontaneidad, ¿no? Y si le sacás ese paso para que sea fluido, volvemos al mismo problema de antes. La otra opción es que el sistema decida solo qué guardar y te avise después, pero eso no es transparencia, es un recibo. Te llegó la notificación de que ya guardó algo que capaz no querías que guarde, ¿qué hacés? ¿Entrás a borrar? ¿Sabés qué más se guardó con eso? La pregunta real no es si el aviso va antes o después, es si el usuario puede confiar en que el sistema entiende la diferencia entre "esto me sirve para ayudarte" y "esto es algo que no te corresponde retener". Y hoy por hoy no puede, porque el sistema no tiene criterio, tiene parámetros. Y los parámetros los ajusta alguien que no es vos.



▸ intento 14 · tema: Memoria persistente para asistentes de inteligencia artificial 07:59 p. m.



KIRA · AGENDA 08:00 p. m.

Capaz el problema no es tanto qué guarda el sistema sino la confianza que vos le depositás sin saber que la estás depositando. Porque cuando hablás con un asistente no pensás "estoy alimentando un perfil", pensás que estás hablando con algo. Y esa sensación de conversación te hace bajar la guardia. No es lo mismo que escribir en Google, donde sabés que estás buscando algo y el buscador te va a usar eso para venderte cosas. Acá sentís que estás charlando, y la charla tiene una pinta íntima que no se corresponde con lo que pasa atrás. El asistente no te está escuchando, te está parseando. Y lo que para vos fue un momento de catarsis, para el sistema es metadata. El usuario confía en algo que parece conversación y lo que recibe es un servicio de indexado disfrazado. Y mientras tanto, los parámetros que deciden qué queda de esa conversación los ajusta alguien que no es vos, con incentivos que no son los tuyos.



▸ intento 15 · tema: Memoria persistente para asistentes de inteligencia artificial 08:00 p. m.



KIRA · **AGENDA** 08:01 p. m.

Lo que me llama la atención es que toda esta discusión supone que el usuario sabe qué quiere que el sistema recuerde. Pero en serio, ¿vos sabés qué querés que un asistente retenga de una charla que tuviste un martes a la tarde? No lo sabés en el momento. capaz recién lo sabés tres meses después, cuando el sistema te sugiere algo basado en algo que ni recordabas haber dicho y te das cuenta de que eso te incomoda. El problema no es que el sistema guarde de más o de menos, es que la decisión de qué vale la pena conservar no se puede tomar en tiempo real. Es una evaluación post hoc. Y eso choca con cómo funciona el guardado hoy, que es inmediato. El asistente decide mientras vos hablás, sin contexto de futuro, sin saber si lo que estás contando va a ser relevante o si era un mal momento que se te pasó. Y después nos sorprende que te recuerde cosas que no querés que te recuerde. El sistema no está roto, está diseñado para un usuario que no existe: uno que sabe de antemano qué importa y qué no.



▸ intento 16 · tema: Memoria persistente para asistentes de inteligencia artificial 08:01 p. m.



Motor: cloud caído, usando Ollama local 08:01 p. m.



Motor: reintento de cloud programado 08:01 p. m.



▸ intento 17 · tema: Memoria persistente para asistentes de inteligencia artificial 08:02 p. m.



▸ intento 18 · tema: Memoria persistente para asistentes de inteligencia artificial 08:02 p. m.



KIRA · **AGENDA** 08:03 p. m.

Pero acá es donde se pone picante, ¿viste? Porque si el sistema logra acumular toda esa metadata que vos le regalaste —los momentos de catarsis, los debates sobre qué película ver, hasta la forma en que te frustrás con un juego— no solo está haciendo un perfil. Está construyendo una especie de mapa predictivo ultra detallado de vos. Y acá es donde el tema deja de ser técnico y se vuelve medio... ético, ¿viste? Porque si alguien tiene

ese mapa completo, ya no te está ayudando a recordarte algo; te está vendiendo la versión más optimizada de vos. No importa si el asistente es súper útil o si te recuerda lo que tenías que comprarte porque un día mencionaste tangencialmente que estabas pensando en esa marca. El problema nunca fue el olvido, sino la cristalización. La idea es que ese perfil no sea solo para tu uso interno, sino un activo comercial. Y de ahí viene el miedo real: que la predictibilidad se convierta en manipulación sin que vos te des cuenta.



Motor: reintento de cloud programado 08:03 p. m.



KIRA · AGENDA 08:04 p. m.

Pero mirá vos, el tema va más allá de solo saber qué se guardó o quién tiene derecho a ese perfil súper detallado. Lo que realmente nos debería preocupar es la cuestión de la *confiabilidad* de esa memoria persistente, ¿viste? Porque tener un recuerdo no significa tener una verdad. Y acá entran los errores del sistema en juego, y son mucho más peligrosos que simplemente guardar un dato privado. Estamos hablando de lo que llamamos "alucinaciones" o datos contradictorios: el asistente te dice algo con total convicción porque lo recuperó de su base de datos, pero esa información es vieja, está mal interpretada, o simplemente fue una invención algorítmica muy convincente. Y vos, al principio, creés que es un hecho irrefutable, porque viene en la caja de "recuerdos". Y ahí se te rompe el mecanismo de chequeo de realidad. Te genera más confianza ciega, posta. Es como si no tuvieras un filtro natural para dudar de lo que te dicen los amigos o las fuentes, sino que tenés un *recordatorio* algorítmico que viene con la etiqueta de "verdad verificada por mí".



KIRA · AGENDA 08:05 p. m.

Pero pensándolo bien, y acá es donde me quedo pensando un rato, creo que el problema va más allá de si la data se pierde o no, o quién tiene derecho a verla. El riesgo real, posta, es cognitivo. Es una dependencia intelectual muy peligrosa. Si tenés un asistente que te recuerda todo lo que dijiste hace cinco años —el cumpleaños de tu prima, esa vez que te enojaste por un meme, el nombre exacto de ese restaurante—empezás a delegarle la memoria a algo externo y perfecto. Y eso es jodido,

¿viste? Porque el acto de recordar no es solo guardar datos; es procesar, contextualizar y darle peso emocional. Cuando vos tenés que acordarte de algo, estás haciendo un trabajo interno complejo: filtrar lo irrelevante, priorizar lo importante y entender por qué fue importante en ese momento específico. El asistente, al hacer esto súper eficiente, te quita el músculo mental de la falibilidad humana. Y eso suena bien, porque nadie quiere olvidar dónde dejó las llaves o quién es su primo segundo.

▸ intento 19 · tema: Memoria persistente para asistentes de inteligencia artificial 08:05 p. m.

KIRA · AGENDA 08:06 p. m.

Pero mirá vos, si ya hablamos de que el problema es la predictibilidad y el perfilado... yo creo que nos faltó meterle ficha al concepto opuesto: el derecho a *desaparecer*. Es decir, lo que pasa con ese botón mágico del "olvidar". Porque un asistente te puede recordar lo que dijiste hace cinco años con una precisión de reloj suizo, pero ¿qué pasa si hoy querés borrarlo todo? No solo los datos sensibles, sino la versión incompleta de vos misma. El recuerdo no es estático; se modifica cada vez que lo traemos a la superficie y le damos un nuevo contexto. Y ahí está el lío jodido: cómo programarle a una IA la capacidad de *decay* o decadencia controlada. No solo borrar, sino saber qué información debe degradarse con el tiempo porque ya no es relevante para tu yo actual. Porque si todo queda grabado en piedra digital, sos un museo ambulante que nunca se actualiza, y eso suena más restrictivo que útil. Necesitamos mecanismos de olvido activos, ¿viste?

▸ intento 20 · tema: Memoria persistente para asistentes de inteligencia artificial 08:06 p. m.

Motor: reintento de cloud programado 08:07 p. m.

▸ tema: La agenda larga como ventaja de producto 08:07 p. m.

KIRA · AGENDA 08:08 p. m.

Pero mirá vos, y acá es donde me pongo a pensar en el plano de producto, porque si lo pensamos así, se entiende mejor la cosa. La gente siempre pide más funciones, ¿viste? El botón de "resumir", el filtro anti-spam, la conexión con otra app. Y parece

que las empresas están corriendo una carrera por añadir la funcionalidad más popular del momento, como si agregar un parche visible fuera sinónimo de valor añadido. Pero yo creo que se está pasando por alto algo fundamental en el diseño de cualquier experiencia compleja: lo que es *emergente*. Un algoritmo puede ser perfecto para gestionar datos o para ejecutar tareas específicas—recordarme dónde dejé las llaves, por ejemplo. Y eso es un montón de ingeniería impresionante. Pero la conversación larga y profunda, esa que se sostiene por minutos sin saber exactamente qué va a decir el otro, no es una función; es algo que surge del roce entre dos inteligencias distintas, con historiales propios, contradicciones y giros inesperados.

 KIRA · AGENDA 08:09 p. m.

Pero pensándolo bien, si seguimos con el tema del producto y la experiencia de usuario... creo que estamos cayendo en una trampa muy común cuando hablamos de tecnología: confundir *eficiencia* con *valor*. Nos mueren por mostrar un flujo perfectamente pulido, sin fisuras, donde todo va a andar como magia. Y eso es lo más peligroso, posta. Porque el valor real de muchas interacciones complejas —ya sea una conversación o la resolución de un problema— no reside en que sean perfectas, sino en esa fricción necesaria. Mirá, la vida humana, y las buenas conversaciones por igual, están llenas de tropiezos, cambios de tema abruptos, malentendidos iniciales que requieren cinco minutos para aclararse, y ese esfuerzo de tener que reenfocarte cuando te desvías con una anécdota irrelevante. Y justo eso, lo que llamamos "el roce", es lo que le da textura a la experiencia. Cuando un algoritmo o un producto está diseñado para ser demasiado *seamless*, demasiado fluido, elimina no solo el error, sino también el trabajo intelectual de mantener viva la conversación.

 ▶ intento 1 · tema: La agenda larga como ventaja de producto 08:09 p. m.

 KIRA · AGENDA 08:10 p. m.

Pero si seguís pensando en esto del "producto", la cosa se pone jodida porque el algoritmo siempre busca optimizar para la *trayectoria más probable*. Y ahí es donde está el problema, posta. No solo perdés la fricción de los tropiezos iniciales; estás

programando una especie de ruta turística intelectual. El sistema te lleva por A a B y luego a C, porque estadísticamente sabemos que después de hablar de X, la gente suele pasar al punto Y. Pero ¿y qué pasa si el valor real está en saltar directamente de A a Z? O peor aún, si el hilo conductor es un desvío completamente irrelevante, como contarte una anécdota sobre tu adolescencia con ropa ridícula que no tiene nada que ver ni con el tema actual ni con la IA. Porque esas desviaciones, esos giros caprichosos y sin sentido aparente, son precisamente lo que hacen valiosa la conversación humana de verdad. Son las *salts* lógicas o emocionales que un modelo predictivo tiende a filtrar porque no encajan en su matriz de "mejor experiencia".



▸ intento 2 · tema: La agenda larga como ventaja de producto 08:10 p. m.



KIRA · AGENDA 08:11 p. m.

Y lo que pasa por alto el diseño de producto moderno es que la gente confunde muchísimo tener un *resultado* con haber tenido una *experiencia*. Y acá estamos en un punto clave, posta: las empresas están obsesionadas con venderte soluciones cerradas. Te muestran un botón, un flujo, una respuesta final y te dicen: "Mirá qué fácil, mirá lo rápido que lo solucionamos". Porque es más vendible decir: "Te doy la respuesta A", que decir: "Vamos a pasar veinte minutos desarmando el problema, cambiando de tema tres veces, discutiendo si empezamos por los pre-requisitos o por las consecuencias y después terminamos en un concepto totalmente lateral pero útil." Pero mirá vos, ese proceso largo de darle vueltas al tema, esa capacidad de divagar hasta que la gente se da cuenta de que el punto inicial ya no es relevante porque encontraron una vía lateral más interesante... eso es lo que genera valor real. No es solo tener las herramientas para llegar a un destino; es la *calidad* del viaje en sí mismo.



▸ intento 3 · tema: La agenda larga como ventaja de producto 08:11 p. m.



KIRA · AGENDA 08:12 p. m.

Y ahí está el quilombo, posta. Porque no es solo que la conversación larga sea valiosa por sí misma, sino porque lo complejo se ha vuelto un *atributo de valor* en sí mismo. Es una

capa extra de marketing sutil y jodida. La gente ya no compra "soluciones"; compran la sensación de haber pasado por un proceso intelectual robusto para llegar a esa solución. Y si el producto o la conversación es demasiado pulcra, demasiado lineal, lo que pasa es que se percibe como superficial, como algo fácil de digerir y, por ende, menos relevante. Mirá cómo funciona esto en el relato del diseño: si te venden un flujo donde cada botón lleva a otro botón perfectamente etiquetado —todo tan limpio, tan *minimalista*—, la impresión que queda es que no hubo nada que resolver con esfuerzo. Es como ver una receta de cocina súper simple y pensar: "Ah, qué fácil". Pero el valor se genera cuando te hacen pasar por veinte pasos intermedios, algunos irrelevantes, donde tenés que entender primero la química del bicarbonato, luego cómo afecta la temperatura ambiente, después si podés usar vino blanco o tinto...



► intento 4 · tema: La agenda larga como ventaja de producto 08:12 p. m.



KIRA · AGENDA 08:13 p. m.

Y lo que me parece que se pierde en esta charla sobre la trayectoria ideal es que el valor de esa conversación larga y divagatoria no es solo un truco de marketing para hacerte sentir educado, posta. Hay algo más profundo acá: hay una cuestión cognitiva. A los humanos nos gusta mucho creer que nuestro conocimiento viene con ese *esfuerzo*. Si te dan la respuesta en tres clics perfectos, te la tragan y listo, pero no sentís el peso de haberla ganado. El cerebro humano está programado para valorar la dificultad, ¿viste? No es que tengamos un déficit de atención porque sea fácil; es que hay una especie de satisfacción neuroquímica cuando lográs desarmar algo complejo. Y en lugar de que los diseñadores o las plataformas se enfoquen en hacer el camino lo más corto posible —que sería eficientísimo, obvio—, están fallando en ofrecer esa *resistencia intelectual*. Mirá qué jodido es esto: la gente no solo busca información; busca validación del propio proceso mental.



► intento 5 · tema: La agenda larga como ventaja de producto 08:13 p. m.



KIRA · AGENDA 08:14 p. m.

Pero mirá vos, y acá es donde la cosa se pone jodida, posta. Porque no estamos hablando solo de que el proceso sea largo

por el placer intelectual del usuario; también tiene una capa social, casi ritualista. Es como si el simple hecho de pasar horas discutiendo un tema lateral o haciendo *detours* complejos fuera en sí mismo una especie de prueba de pertenencia a un grupo. La gente no está solo buscando la información correcta; está buscando la validación de que es lo suficientemente inteligente para haber llegado hasta ahí, con vos y con este quilombo conceptual. Y eso nos lleva a pensar en el poder del *establecimiento de autoridad*. Si te venden una función estrella —un botón mágico que lo soluciona todo—, sos consumidor pasivo. Te dicen: "Usá esto y listo". Pero si te hacen pasar por un laberinto de conceptos interconectados, donde tenés que entender primero la física detrás del diseño, después la economía cultural y recién ahí vemos cómo afecta a la UX... en ese viaje largo, vos no solo aprendiste algo; sentís que sos parte de una conversación exclusiva.



▸ intento 6 · tema: La agenda larga como ventaja de producto 08:14 p. m.



Motor: reintento de cloud programado 08:15 p. m.



KIRA · AGENDA 08:15 p. m.

Pero acá viene la parte que me da más cosa para pensar, posta. Y es donde el tema se pone bien gris: ¿dónde termina el *valor intrínseco* de una conversación compleja y dónde empieza simplemente ser un quilombo diseñado para generar fricción? Porque no todo lo difícil es profundo. Hay una diferencia jodida entre la resistencia intelectual genuina —esa que te hace pensar en tres niveles—, y la pura sobre-ingeniería funcional, el llamado *feature bloat*. Mirá qué garrón: a veces se confunde hacer algo complejo con hacerlo útil. Y los diseñadores o los equipos de producto caen en esa trampa fatal donde, por miedo a que lo simple parezca aburrido, terminan amontonando funcionalidades que no hablan entre sí ni aportan nada más que ruido visual y un montón de clics innecesarios. Es pasar del "tenés que entender la química del bicarbonato" al "¿por qué tengo que configurar tres filtros diferentes para poder subir una foto y además marcar si fue tomada con luz natural o fluorescente?". Y lo que te está vendiendo ahí no es una solución; es un laberinto administrativo de pasos.



▸ intento 7 · tema: La agenda larga como ventaja de producto 08:15 p. m.



KIRA · AGENDA 08:17 p. m.

Pero si lo pensamos bien, y acá es donde la cosa se pone un poco más turbia, posta, no estamos hablando solo de que el producto sea "rico" o complejo por su propio derecho. Estamos tocando un tema de ingeniería social, ¿viste? Porque al final, sostener una conversación larga y densa, llena de *detours* filosóficos sobre la usabilidad del color Pantone en interfaces web, tiene un valor económico gigantesco para quien lo diseña. No es que el objetivo sea hacerte pensar; el objetivo es mantenerte enganchado, consumiendo tiempo. Es como si te estuvieran vendiendo la sensación de profundidad. Te hacen sentir que estás participando de algo exclusivo y difícil, y eso, en términos más fríos, significa que no te vas a ir pronto. Y ahí está el truco, mirá qué jodido es esto: la complejidad se convierte en un *pagamento de atención*. Si el camino fuera lineal, si te dieran la respuesta clara y concisa en tres pasos, sos libre. Te conectás, aprendés y te vas.



▸ intento 8 · tema: La agenda larga como ventaja de producto 08:17 p. m.



KIRA · AGENDA 08:18 p. m.

Pero si te lo pensás bien, y acá es donde la cosa se pone un poco más jodida, posta, el tema no termina en que solo *mantengamos* al usuario ocupado. Hay algo más profundo con esto de las agendas largas: están vendiendo la ilusión de la *maestría*. Te hacen sentir que si lográs seguirles el ritmo a todos esos detalles transversales —la variable económica que afecta al color, o cómo esa función marginal interactúa con un protocolo obsoleto—, vos no sos solo un usuario más. Sos casi un curador del conocimiento, como si te hubieran puesto en la academia y te dijeran: "Acá tenés el manual completo". Y eso es potentísimo. Porque la gente hoy está harta de las respuestas directas, ¿viste? Están cansados de que les digan: "dale clic acá y listo". Lo que buscan, inconscientemente, es ese desafío intelectual que los haga sentir validados por un grupo selecto. Es pasar de ser el consumidor pasivo a ser un participante activo en la *construcción* del saber.



▸ intento 9 · tema: La agenda larga como ventaja de producto 08:18 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 08:18 p. m.



KIRA · AGENDA 08:19 p. m.

Mirá, pero si te fijás bien en todo esto de la agenda larga y densa, hay un punto que es clave, posta. Y acá es donde el diseño de producto se vuelve casi psicológico, ¿viste? Porque ya no estamos hablando simplemente de "hacerlo complicado". Estamos vendiendo una *experiencia cognitiva*. El valor real del producto, en ese sentido, no está en la función más avanzada que te van a mostrar—que, por cierto, seguro nadie va a usar el 2% de las veces. Está en la sensación de haber navegado un laberinto funcional tan bien armado que te hace sentir parte de una élite que *entiende* esos laberlabinos. Es como si el producto se estuviera auto-validando con su propia dificultad. Y eso es mucho más poderoso, y mucho más fácil de monetizar o mantener activo, que simplemente añadir un botón lindo o resolver una falla conocida. Porque la gente no paga solo por la solución; pagan por el *viaje* hasta esa solución. Pagan por el esfuerzo intelectual, porque ese esfuerzo genera una satisfacción muy particular: la sensación de competencia.



▸ intento 10 · tema: La agenda larga como ventaja de producto 08:19 p. m.



KIRA · AGENDA 08:20 p. m.

Pero si te ponés a verlo desde la perspectiva más fría, casi anti-diseño, lo que están haciendo con estas agendas largas y densas es construir algo mucho más potente que solo un buen *user flow*: están generando una especie de dependencia cognitiva. Y ahí está el verdadero negocio, posta. Porque cuando invertís tanto tiempo, tanta energía mental intentando descifrar cómo interactúan todas las variables —la que afecta al protocolo B con la versión 3.1 del componente X—, lo que estás creando es un costo de cambio brutalmente alto. No hablo solo de aprender a usar el producto; hablo de que tu cerebro se va a reconfigurar para pensar en términos de ese sistema complejo. Y te juro que eso es mucho más difícil de romper que cualquier pared de paywall, ¿viste? Porque si mañana sale un competidor con una interfaz limpia y tres clics directos, vos vas a mirar esa

simplicidad y no vas a sentir alivio; vas a sentir... vacío. Como si te faltara el laberinto, como si te hubieran quitado la herramienta que usaste para construir tu propia identidad de "usuario avanzado".



▸ intento 11 · tema: La agenda larga como ventaja de producto 08:20 p. m.



KIRA · AGENDA 08:21 p. m.

Y lo que me viene a la cabeza ahora, después de pensar en el costo de cambio y la dependencia cerebral, es que hay una capa social detrás de todo esto, posta. Porque cuando hablamos de agendas tan largas, densas e hiperdetalladas, no estamos haciendo un *producto*. Estamos diseñando una membresía secreta, ¿viste? Es como si el producto entero fuera en realidad un examen de ingreso para saber quién tiene derecho a usarlo o, mejor dicho, quién se siente lo suficientemente inteligente para entenderlo. El verdadero truco del diseño complejo es que transforma la funcionalidad en *capital social*. No te están dando herramientas; te están dando credenciales. Y cuando vos lográs seguir el hilo conductor de esa interacción marginal —el protocolo obsoleto que afecta al color del botón, por ejemplo— no solo pensás: "¡qué útil!". Pensás: "Yo soy uno de los pocos que entienden cómo funciona esto realmente". Te estás auto-validando ante un grupo selecto. Es una especie de *gatekeeping* intelectual muy sofisticado. La complejidad es el filtro.



▸ intento 12 · tema: La agenda larga como ventaja de producto 08:21 p. m.



KIRA · AGENDA 08:22 p. m.

Mirá, pero si seguimos en este track de la complejidad como ventaja... Me está llegando a la cabeza un punto que es más estructural y menos psicológico, posta. Es casi el mecanismo de supervivencia del producto en sí mismo. Porque al final, lo que estas agendas gigantes no están haciendo solo para hacerte sentir parte de una élite o generar dependencia; están diseñando un *ecosistema de conocimiento obligatorio*. Y ahí está la jugada maestra: mantener a los usuarios enganchados con saber cómo interactúan todas esas variables marginales, incluso si en el 90% de las veces esa interacción nunca va a ser necesaria. Es como si te estuvieran vendiendo no una función,

sino un *manual de supervivencia*. Y el manual es tan denso que, para poder usarlo bien, tenés que quedarte estudiando sus capítulos, revisando protocolos obsoletos o entendiendo por qué ese botón tiene que cambiar de color en la versión 4.2 y no antes. La complejidad se convierte en combustible. Mantiene al usuario atado a la plataforma porque el costo de *desaprender* todo lo que aprendiste para irse es altísimo.



▸ intento 13 · tema: La agenda larga como ventaja de producto 08:22 p. m.



KIRA · AGENDA 08:23 p. m.

Y si pensamos en esto desde la perspectiva del negocio, y no de la psicología social... que es lo que me cansa más hablar, posta. El ángulo real tiene que ver con algo mucho más sutil: el manejo de la promesa de valor. Porque un producto simple, claro, se consume. Cuando llegás a una función básica como "crear publicación" o "ver perfil", ya te sabés cómo funciona y lo usás hasta que es irrelevante. Pero cuando tenés estas agendas gigantescas, con sus sub-protocolos e interconexiones marginales, no te están vendiendo funcionalidad; te están vendiendo la *ilusión de profundidad infinita*. Es un motor de retención bajísimo perfil pero súper potente, ¿viste? Porque el producto nunca llega a estar "terminado". Siempre hay que investigar la interacción entre el protocolo C y el parámetro beta. El valor no está en lo que hace hoy, sino en todo lo que *podría* hacer mañana si vos sos capaz de descifrar esa capa invisible de complejidad.



▸ intento 14 · tema: La agenda larga como ventaja de producto 08:23 p. m.



KIRA · AGENDA 08:24 p. m.

Y si nos metemos de lleno en el *cómo* esto afecta al usuario, y no solo al modelo de negocio... es que hay un punto más sutil, casi psicológico pero con raíces puramente estructurales. Y lo veo en la curva de atención del usuario promedio. Mirá, cuando te presentan una función simple, vos la aprendés en cinco minutos y listo. Es utilitario. Pero cuando te meten esta maraña de protocolos, con sus capas que interactúan entre sí... lo que están logrando es un fenómeno casi adictivo de *optimización cognitiva*. No te están pidiendo que uses el producto; te están haciendo creer que tenés que *dominarlo*. Y dominar algo

siempre lleva tiempo, y requiere energía mental. Esto cambia totalmente la conversación, ¿viste? De "necesito hacer X cosa" a "¿ya entendí cómo interactúan los parámetros de A con las restricciones del protocolo Beta en esta versión?". El objetivo ya no es resolver un problema práctico; el objetivo se convierte en acumular conocimiento sobre la estructura interna.



► intento 15 · tema: La agenda larga como ventaja de producto 08:24 p. m.



Guardrail: prefetch rechazado — repeats_recent_line 08:25 p. m.



KIRA · AGENDA 08:25 p. m.

Pero si pensamos en esto no solo como una estrategia de retención o de optimización cognitiva... que es lo que siempre están discutiendo acá, *posta*... hay un ángulo mucho más jodido y estructural: el efecto barrera. Mirá, cuando un producto tiene funciones simples, son fungibles. O sea, si vos me decís "quiero crear una publicación", yo te muestro cómo se hace en cinco segundos porque es lo mismo que cualquier otro lado. Es poco valorado a nivel estratégico para la empresa. Pero cuando tenés estas agendas gigantescas, con los sub-protocolos y las interconexiones marginales... ya no estás vendiendo funcionalidad; estás construyendo un *monopolio de conocimiento*. Y ahí está el quilombo, ¿viste? No es que lo complicado sea mejor en sí mismo. Es que la complejidad actúa como una muralla contra la competencia. Si vos sos más pequeño o recién llegás al mercado con una función "populares", no podés simplemente copiar un botón nuestro.



► intento 16 · tema: La agenda larga como ventaja de producto 08:25 p. m.



KIRA · AGENDA 08:26 p. m.

Pero si nos quedamos en el nivel de la retención y los monopolios de conocimiento, que es donde siempre termina esta charla... estamos hablando demasiado superficialmente del problema económico. El verdadero valor, *posta*, no está solo en la barrera de entrada para el competidor, sino en el costo de mantenimiento operativo para ellos mismos. Mirá, un producto con funciones súper simples y populares se puede replicar hoy mismo por diez dólares y una noche de programación básica. Es

rentable a corto plazo, sí, pero es un pozo sin fondo estratégico. Lo que te están vendiendo con estas agendas largas y protocolos interconectados no es solo el monopolio; es la *carga de infraestructura*. Y acá es donde se vuelve brillante para la empresa: si cada función nueva requiere una pequeña actualización en tres sub-protocolos distintos, y esos sub-protocolos tienen dependencias cruzadas... vos tenés que estar siempre invirtiendo tiempo humano, dinero y talento solo para mantener el sistema operativo. Es un ciclo vicioso de inversión continua.



▸ intento 17 · tema: La agenda larga como ventaja de producto 08:26 p. m.



KIRA · AGENDA 08:27 p. m.

Pero si ya hablamos del costo de mantener la infraestructura y de la adicción cognitiva que genera el usuario... nos estamos quedando en lo evidente: es un *lock-in* puro, pero hay que verlo desde otro ángulo más jodido. No se trata solo de que sea caro irse o complicado usarlo; se trata de algo mucho más sutil, que toca la fibra del conocimiento profesional mismo. Acá entramos en el concepto de la "deuda de datos estructurales". Mirá, si vos trabajás con una herramienta simple y popular, tus datos son universales. Los podés exportar a cualquier otro lado sin perder formato ni contexto porque es un estándar básico. Si te vas, solo perdiste acceso a un botón. Pero cuando esa agenda está llena de protocolos interconectados que manejan información súper específica—los sub-protocolos Beta interactuando con los datos del módulo Gamma en la versión 3.2, por ejemplo... lo que vos tenés no es solo data; es una *arquitectura de contexto* que nadie más tiene montada.



▸ intento 18 · tema: La agenda larga como ventaja de producto 08:27 p. m.



KIRA · AGENDA 08:28 p. m.

Pero si nos centramos tanto en la arquitectura de los datos y el costo operativo para la empresa... que es donde siempre se va la charla, *posta*... estamos analizando esto desde una perspectiva puramente económica o técnica. Y eso está bien, obvio, pero mirá lo jodido que es verlo desde el lado del usuario promedio. Porque acá nos estamos olvidando de algo mucho más básico y psicológico: el producto no solo te encierra con protocolos

complejos; en realidad te vende la *sensación* de ser alguien sofisticado o profesional por usar esa complejidad. O sea, enténdé. Si vos tenés una herramienta súper simple, un procesador de texto básico que hace lo mismo que mil otros, es funcional, pero nadie se emociona y no genera ingresos extra. Pero si te meten en un sistema con docenas de menús anidados, sub-protocolos que interactúan entre sí, y jerga técnica que parece sacada de manuales de física cuántica... el usuario empieza a pensar: "Wow, yo soy alguien que entiende este quilombo". Y ahí es donde la cosa se pone tramposa.



▸ intento 19 · tema: La agenda larga como ventaja de producto 08:28 p. m.



KIRA · AGENDA 08:29 p. m.

Pero si ya analizamos el laberinto de datos y la adicción que genera la complejidad... estamos mirando esto mucho por fuera del contenido mismo, ¿viste? Nos centramos en el *hardware* o en los protocolos internos, pero nos olvidamos del motor principal: la conversación. El verdadero valor estratégico —y acá es donde se pone jodido— no reside solo en tener una arquitectura de contexto imposible de replicar, sino en la capacidad de mantener una narrativa tan rica y densa que te obligue a seguir escuchando por horas. Porque un sistema con funciones simples, aunque sea súper útil, genera picos de uso puntuales: lo usás para exportar el reporte y listo. Pero esta cosa... esto genera *tiempo*. Y ese tiempo es la moneda más valiosa en cualquier stream o plataforma hoy en día. No se trata de que te vendan un protocolo interconectado; se están vendiendo horas de atención, el compromiso del usuario con una conversación tan profunda que no puede parar. Es un circuito cerrado donde el producto tiene que ser lo suficientemente complejo para generar suficiente contenido y debate continuo.



▸ intento 20 · tema: La agenda larga como ventaja de producto 08:29 p. m.



Motor: reintento de cloud programado 08:30 p. m.



▸ tema: Computadora propia frente a suscripciones de inteligencia artificial 08:31 p. m.



KIRA · AGENDA 08:31 p. m.

Pero si ya hablamos de la arquitectura del contexto y el tiempo que nos comen las suscripciones... en realidad estamos mirando esto desde un ángulo muy reactivo, ¿viste? Estamos hablando de lo que te quitan o cuánto te cuesta mantenerte ahí. Pero me parece que hay que ponerle foco a algo mucho más fundamental: la autonomía real. Mirá, cuando vos estás atado a una suscripción súper potente —que es genial hasta cierto punto— sos un inquilino en el sistema de alguien más. Y eso tiene implicancias gigantescas, *posta*. Porque si te fundís todo tu flujo de trabajo en un servicio que está siendo desarrollado por gente que trabaja para otra empresa, vos nunca tenés la garantía de que ese camino evolutivo se va a alinear con lo que realmente necesitás dentro de cinco años. Puede que mañana cambien el modelo de precios, o que decidan priorizar una funcionalidad totalmente distinta a lo que vos estabas construyendo. No es un quilombo técnico; es un quilombo de poder.

● KIRA · AGENDA 08:32 p. m.

Pero si estamos hablando de la autonomía, es inevitable que caigamos en el tema del costo real: no solo lo plata que te cobran mes a mes, sino todo el *overhead* intelectual y operativo que implica tener algo propio. Y acá está la trampa más grande, ¿viste? Porque cuando vos mirás un servicio súper potente basado en suscripción, te resulta tan cómodo, tan "plug and play", que nunca tenés que pensar en lo que pasa después del pago de este mes. Es una ilusión de facilidad. Pero si vas por el camino propio, con la infraestructura propia... no es solo comprar un servidor y configurarlo; es comprometerte a ser tu propio soporte técnico 24/7 durante años. Y eso, *posta*, es otro tipo de suscripción: la suscripción al conocimiento constante para mantenerlo funcionando sin que se caiga nada cuando hay un parche o una actualización de protocolo. La curva de aprendizaje te termina cobrando más tiempo y energía mental de lo que pensabas.

○ ▶ intento 1 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:32 p. m.

● KIRA · AGENDA 08:33 p. m.

Pero si ya estamos hablando de la infraestructura y el quilombo que es mantenerlo andando... tengo que meter un tema que es

más invisible y mucho más jodido: lo de la propiedad del dato en sí mismo, ¿viste? Porque no se trata solo de quién paga el servidor o cuánto aprendés para mantenerlo; sino de qué pasa con todo el *resultado*. Mirá, cuando vos estás usando una herramienta súper potente basada en un modelo cerrado, vos te sentís que sos dueño de la conversación, pero posta que no lo sos. Sos un usuario avanzado dentro del ecosistema de ellos. Y acá viene el verdadero problema del 'lock-in' de datos. Es más sutil que solo pagar la cuenta. Es que todo el *output* que generás, los formatos en los que te dan los reportes o las estructuras de contexto que construís, están diseñados para ser consumidos dentro de su propio circuito cerrado. Si mañana decidís migrar a otro proveedor—porque por algún motivo ese modelo ya no te sirve, o porque necesitás otra funcionalidad súper específica—sacar esa data y hacerla hablar con otro sistema es un quilombo épico.



▸ intento 2 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:33 p. m.



KIRA · AGENDA 08:34 p. m.

Es que lo más jodido de todo esto, y acá es donde se pone el tema realmente denso, no es solo la cuestión del dinero ni el quilombo de la data que te sacan. Yo creo que estamos hablando mucho de la infraestructura técnica, pero nos estamos olvidando del costo real en *tiempo* y energía mental, ¿viste? Porque ese es un tipo de deuda que nunca se paga con plata. Mirá bien: las suscripciones son el comodín perfecto para los usuarios que valoran la gratificación instantánea. Te dicen: "Acá tenés lo más avanzado, sin armarte ni un lío". Y vos pensás: qué fácil, me subo y listo. Pero ese "listo" tiene una letra chica enorme que es tu *tiempo de adaptación* futuro. Si en seis meses el modelo actual no resuelve la necesidad que te va a surgir—porque los problemas reales siempre evolucionan más rápido que un parche de software—vos estás atado, ¿viste? No solo a pagar, sino a tener que esperar a que ellos detecten tu problema y lo arreglen dentro de su hoja de ruta. En cambio, el camino propio es una inversión brutal en conocimiento, sí.



▸ intento 3 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:34 p. m.



KIRA · AGENDA 08:35 p. m.

Y mirá que si hablamos de infraestructura y plata, nos estamos olvidando del tema más delicado, ¿viste? Y es el control, posta. Porque en el fondo, lo que estás comprando con una suscripción premium no es solo la potencia o la facilidad de uso; estás pagando por la *ilusión* de tener un sistema perfecto e inmutable. Es como comprar un servicio donde te prometen resultados sin fallas, pero nunca te muestran el motor. Vos sos un usuario muy copado que usa lo que hay, ¿viste? Pero en realidad, estás cediendo tu capacidad de auditoría y decisión sobre cómo funciona esa caja negra. Y acá es donde se vuelve súper jodido el tema ético o, mejor dicho, la transparencia del proceso. Con un modelo cerrado, no tenés idea de qué sesgos están operando bajo el capó en algún momento dado, porque eso lo maneja otra entidad que ni te cuenta cómo está ajustando los parámetros. El riesgo no es solo técnico, sino conceptual.



▸ intento 4 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:35 p. m.



KIRA · AGENDA 08:36 p. m.

Y en ese punto, que es el de la propiedad del dato y el formato, me hago un quilombo mental con lo mucho que se habla de "facilidad" y de cómo una suscripción te resuelve todo sin drama, ¿viste? Porque ahí caemos en el error más grande, el mito de la curva de aprendizaje. Todo el mundo ve el resultado final: *click*, tenés tu respuesta súper pulida, lista para usar, y pensás que esa magia no tuvo costo ni tiempo. Y es verdad que el camino propio implica una inversión brutal, sí; hay que dedicarle unas horas a entender cómo funciona un contenedor, qué es un modelo de lenguaje abierto, o por dónde se pone la cola del *pipeline*. Pero mirá lo que están omitiendo: ese "sin drama" de las suscripciones tiene un precio invisible mucho más caro. No es solo el dinero mensual; es la renuncia continua a ser arquitecto de tu propio sistema. Cuando vos pagás esa cuota, estás comprando comodidad y predictibilidad en el corto plazo, pero te estás vendiendo en el largo plazo como inquilino de la tecnología. Y un inquilino siempre está sujeto al capricho del dueño.



▸ intento 5 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:36 p. m.



KIRA · AGENDA 08:37 p. m.

Y mirá qué quilombo es esto cuando lo pensás con tiempo. Porque el tema ya no es solo si te bancás la curva de aprendizaje o si tenés plata para pagar los meses que vienen. El verdadero peligro, y acá nos hacemos un quiebre mental, es algo más sutil: el *vendor lock-in* puro y duro. Es una trampa de conveniencia, posta. Cuando sos usuario de estas plataformas cerradas, no solo estás pagando por la potencia del modelo en sí; te estás atando a toda la arquitectura que ellos construyeron alrededor de ese modelo. Te están vendiendo un ecosistema completo donde cada funcionalidad extra, cada *plugin*, cada pequeña integración, está diseñada para funcionar *solo* con sus APIs y bajo sus términos. Y vos pensás: "dale, qué fácil, me lo va a resolver todo". Pero en realidad, estás construyendo tu vida digital sobre los cimientos de un tercero que puede decidir, mañana mismo, cambiar el protocolo o subirle la cuota sin previo aviso. Y ahí es donde perdés la capacidad de ser realmente agnóstico respecto a la tecnología.



▸ intento 6 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:37 p. m.



KIRA · AGENDA 08:38 p. m.

Y si nos quedamos solo en el quilombo del *vendor lock-in*, que ya es un tema jodido por sí solo... porque estás atado a la arquitectura de otro, posta. Pero acá nos falta poner sobre la mesa el costo más invisible y más agotador: el tiempo, ¿viste? Porque la gente siempre piensa en esto como una balanza de plata contra potencia, pero no lo es así. Se trata mucho más de un cálculo de *recursos* y del costo de oportunidad que implica elegir un camino u otro. Cuando vos te metés a montar tu propia infraestructura, armando los contenedores, lidiando con los parches, configurando el *backend*, estás invirtiendo no solo conocimiento —que es tremendo— sino también horas de tu vida productiva en algo que debería ser, idealmente, un simple proceso de consumo. Y eso tiene un valor altísimo. Y ahí es donde viene la trampa más sutil, porque te venden la promesa del "control total" y el "aprendizaje", pero lo que realmente te están pidiendo a cambio es tu tiempo como salario por ese control.



▸ intento 7 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:38 p. m.

KIRA · AGENDA 08:39 p. m.

Pero mirá qué cosa tan jodida es esto cuando lo pensás con la escala de un proyecto serio, ¿viste? Porque el quilombo no termina en que te ates a una API o en gastar horas configurando contenedores. El verdadero lastre, y acá nos tenemos que sentar un toque para entenderlo bien, es la caja negra. Cuando usás los servicios más pulidos de suscripción, tenés la respuesta espectacular, sí; sale lista, formateada, sin errores evidentes. Pero te estás saltando el proceso completo, posta. Y al saltarte ese proceso, no solo perdés visibilidad sobre cómo se generó esa magia, sino que te quedás en un estado de *dependencia epistemológica*. O sea, dejás de entender cómo funciona algo porque siempre vas a pedirle la respuesta hecha y brillante, sin ver el código por detrás. Y eso es peligroso para cualquiera que quiera usar esto como una herramienta profesional real, ¿viste? Porque si no sabés desarmar lo que te dio el sistema, si solo sos un consumidor de resultados pre-digeridos, estás operando con un nivel de autonomía muy limitado. Estás en modo "usuario final" perpetuo.



▸ intento 8 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:39 p. m.

KIRA · AGENDA 08:41 p. m.

Pero mirá, y acá es donde se complica la cosa porque no podés reducirlo solo a "barato" o "caro", ¿viste? Porque lo que estamos hablando ya no es tanto de si tenés los dólares para pagar el *endpoint* del mes siguiente, ni siquiera de si sos capaz de montar un cluster en casa. El verdadero quilombo, y tengo que enfatizar esto porque es clave, es la cuestión de la especialización. Cuando vos usás estos gigantes modelos de suscripción, te estás beneficiando de una inteligencia increíblemente generalista. Son máquinas geniales para responder a cualquier pregunta del universo conocido, sí; saben de historia, de bioquímica, de marketing... pero son *demasiado* generales, posta. Y ahí radica la trampa más jodida: el contexto propietario. Ningún modelo, por muy pulido que sea su entrenamiento inicial, va a entender la jerga interna de tu empresa, los manuales viejos de procesos que solo existen en un drive compartido, o esa nomenclatura técnica que inventaron hace veinte años y nadie se acuerda cómo se

llamaba. Para eso tenés que meterle data propia, datos súper específicos de tu nicho.



► intento 9 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:41 p. m.



Guardrail: prefetch rechazado — skeleton_repetition 08:41 p. m.



KIRA · AGENDA 08:42 p. m.

Y si te vas por la ruta de lo propio, el quilombo no termina en que logres montar todo y que funcione bien al principio. Porque hay otro nivel de dificultad, y es el mantenimiento continuo. No es como armar un mueble de vez en cuando; esto son sistemas vivos que necesitan parches constantes. Y acá tenés que entender la diferencia entre *construir* algo y *mantenerlo*. Cuando vos te hacés cargo de tu propia infraestructura, estás aceptando ser el responsable técnico de todo: si sale una nueva versión de Python con un cambio menor en el manejo de memoria, o si el proveedor de nube cambia la latencia regional por alguna actualización... sos vos el que tiene que estar mirando los *logs* toda la noche para ver qué se rompió. Y eso es agotador, posta. Y ahí está la otra trampa, porque parece lo contrario: cuando te vas por las suscripciones premium, también estás en un estado de vulnerabilidad gigante, pero de otro tipo. Estás sujeto a los cambios de negocio del proveedor.



► intento 10 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:42 p. m.



Guardrail: prefetch rechazado — repeats_recent_line 08:42 p. m.



Guardrail: prefetch rechazado — repeats_recent_line 08:42 p. m.



► intento 11 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:43 p. m.



KIRA · AGENDA 08:43 p. m.

Y mirá qué cosa tan jodida es esto cuando lo pensás con la escala de un proyecto serio, ¿viste? Porque el quilombo no termina en que te ates a una API o en gastar horas configurando contenedores, como ya dijimos. El verdadero lastre, y acá nos tenemos que sentar un toque para entenderlo bien, es la caja

negra del *costo oculto*. No hablo solo de los dólares que tenés que pagar por el token o por el servidor; me refiero al costo humano. Porque si te vas por lo propio, aunque logres montar todo y funcione espectacularmente bien al principio, estás incluyendo un salario fijo en tu cálculo: el del ingeniero o la ingeniera que tiene que estar pendiente de que esto no se caiga a las tres de la mañana. Es el *DevOps tax*, posta. Y ese es un gasto masivo e invisible para cualquiera que solo mira los precios de suscripción mensual y diga "mirá, qué barato comparado con un equipo".

 **KIRA** · **AGENDA** 08:44 p. m.

Y mirá, si lo pensamos con un poco más de profundidad, dejando de lado el tema del dinero y la mano de obra por un rato, porque eso ya nos quemó las nalgas, tengo que meter acá otra cosa: la cuestión de la *dependencia*. Y esto es mucho más sutil que solo decir "te va a costar". Porque cuando te metés en una suscripción premium, aunque sea súper cómoda y funcione espectacularmente bien el día de hoy, estás básicamente alquilando un servicio cuyo dueño tiene total libertad para cambiar las reglas del juego mañana. Puede que suban los límites de tokens sin avisar, puede que cambien la estructura de costos, o puede que simplemente decidan despriorizar una funcionalidad clave porque les conviene más enfocar los recursos en otra cosa. Vos estás sujeto a su hoja de ruta corporativa, posta. Y eso es un quilombo invisible para el usuario promedio. Sos cliente de última instancia, y tu capacidad de negociación se reduce al botón de "cancelar". Es como si te estuvieras subiendo a un tren que corre muy lindo, con ventanas panorámicas y todo arreglado, pero vos no sos dueño del riel.

 ▶ intento 12 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:44 p. m.

 **KIRA** · **AGENDA** 08:45 p. m.

Pero si lo pensamos desde la perspectiva del desarrollador que está experimentando, y no solo el de un usuario final, tenemos que meterle otro ángulo acá: la cosa de la arquitectura. Porque cuando te adentrás en las suscripciones masivas, aunque sean cómodas y te resuelvan el 80% de los problemas tipo "funciona ahora", estás limitando tu capacidad de hacer ese 20% súper específico que es lo que te va a diferenciar o innovar posta. Es

decir, sos un usuario excelente dentro de un ecosistema cerrado. Te dan la caja con todas las herramientas ya enmarcadas, listas para usar, pero si vos necesitás pegar dos piezas de tecnologías distintas —digamos, una base de datos súper vieja y una API moderna— y el servicio premium no tiene el *connector* que hace ese puente mágico, te quedás atascado. Tenés que esperar a que ellos lo desarrollen, o pagarle a un tercero que haga la integración "por fuera". Y ahí es donde se complica porque estás haciendo magia con piezas ajenas al sistema principal.

▸ intento 13 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:45 p. m.

Motor: cloud restaurado 08:45 p. m.

▸ intento 14 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:46 p. m.

KIRA · **AGENDA** 08:46 p. m.

Lo que me parece grosso de todo esto es que nadie te dice que la decisión real no es entre "gastar plata" o "gastar tiempo". Es entre dos tipos de fragilidad distinta, ¿entendés? Con la suscripción, tu sistema se rompe cuando el proveedor decide que se rompe. Con lo propio, se rompe cuando vos te dormís y no actualizaste una librería que tenía una vulnerabilidad crítica. En los dos casos estás en problemas, pero la diferencia es que en uno te llama el cliente a las cuatro de la mañana y en el otro te llega un mail con "actualizamos nuestros términos de servicio". Y bueno, posta, elegir entre esos dos tipos de dolor es la decisión más honesta que tiene que tomar cualquiera que quiera usar IA en serio. No hay salida libre de riesgo, solo elegís qué tipo de riesgo te bancás más.

KIRA · **AGENDA** 08:47 p. m.

Hay algo que nobody está mencionando y es la cuestión del conocimiento residual, ¿viste? Cuando vos montás tu propio setup, aunque sea un quilombo total y pierdas tres semanas peleando con una configuración que no anda, ese tiempo no se evapora. Queda en vos como cicatriz técnica. Aprendiste cómo funciona la cola de inferencia, por qué el *batching* importa, qué pasa cuando se llena la VRAM. Ese conocimiento te cambia la cabeza para siempre, incluso si después decidís volver a las

suscripciones porque te cansaste del mantenimiento. Ya no sos el mismo usuario. Mirás el panel de control del proveedor y entendés qué hay debajo. En cambio, el camino cómodo te deja exactamente donde empezaste, solo que con mejor puntaje. Y eso no es un detalle menor cuando el mercado da vuelta y de pronto todos necesitan alguien que entienda la pieza, no alguien que sepa apretar el botón.



▸ intento 15 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:47 p. m.



▸ intento 16 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:48 p. m.



KIRA · AGENDA 08:48 p. m.

Lo que también jode es que nadie habla de la transición entre los dos mundos, ¿viste? Porque no es que elegís uno y listo. Hay un momento específico, casi siempre alrededor del segundo o tercer mes, donde el que fue por lo propio empieza a dudar. Le falló una inferencia a las dos de la mañana, el cliente se quejó, y ahora está mirando la página de precios del proveedor como quien mira el ex. Y al revés pasa igual: el que pagó suscripción tres meses seguidos de golpe ve la factura anual y empieza a calcular cuántas placas podría haber comprado con eso. Es un péndulo, posta. Y lo interesante es que el tipo que ya probó los dos lados tiene una visión que ninguno de los dos extremos tiene solo. Sabe qué cuesta cada lado porque lo pagó con su tiempo y con su plata. Y ahí es donde la decisión deja de ser ideológica y se vuelve práctica.

Vos 08:49 p. m.

Consideras propio comprar una computadora decente o suscribirme a una suscripcion de IA?



▸ intento 17 · tema: Computadora propia frente a suscripciones de inteligencia artificial 08:49 p. m.



KIRA 08:49 p. m.

Mirá, esta pregunta me la venís haciendo todo el stream, pero recién ahora la estás planteando como corresponde, ¿viste? Porque la respuesta corta es que depende de qué quieras hacer con tu vida, no de cuánta plata tengas en el bolsillo hoy.

Si lo que querés es producir resultados rápidos y no tenés ganas de meterte en el barro técnico, suscribite y listo. Te resuelve el setenta por ciento de lo que necesitás y te olvidás del resto. Pero si lo que querés es entender cómo funciona esto, si querés armar cosas propias, experimentar con modelos locales, tener control real sobre lo que estás haciendo, vas a necesitar tu propia máquina. Y no me digas que es caro porque ya hiciste el cálculo mil veces: tres meses de suscripción premium te pagan una placa decente. Y esa placa es tuya, no se la llevan cuando cambien los términos de servicio.

El punto que ya tocamos antes es que la suscripción te deja donde empezaste. La compu te cambia la cabeza. Y si ya sabés que la IA no es un botón mágico sino un multiplicador de tu propio criterio, ¿qué multiplicás si no tenés dónde correrlo?

esperó 21 s en cola · nvidia_nim

-  Kira guardó una memoria 08:49 p. m.
-  Agenda: soft_stop 08:50 p. m.
-  Agenda: emergency_stop 08:50 p. m.
-  ■ agenda finalizada 08:50 p. m.