



Community Day Argentina 2025



TD SYNnex



CAYLENT



CRAFTECH
Your DevOps team



PAGOS360



CloudHesive



GoCloud



simplestore
merchandising



DinoCloud



Vibecodeando tu Data Lake

Lo que hice, lo que rompí, lo que arreglé
(y cuánto me costó)

Ramiro Savoie

Founder & Cloud Architect - deployr.ai

Tabla de contenido

- Por qué esta charla “desde la trinchera”
- ¿Vibecodear un Data Lake?
- El arranque: Marzo 2025, Sonnet 3.7 y Claude Code.
- La necesidad.
- Arquitectura y primera apuesta.



Tabla de contenido

- Lo que dolió.
- Validación y pruebas con “la realidad”.
- Refactors que cambiaron el juego.
- Aprendizajes para no repetir mis errores.



Desde la trinchera

Una historia de implementación, no otro pitch de “la magia de la IA”



La consigna que me puse

- “*Contalo como una historia*”: decisiones, costos, errores y arreglos.
- Sin atajos: qué hice, por qué y cómo me fue.
- Objetivo: que puedas evitar mis errores y acelerar lo que sí funcionó.



¿Saben lo que es vibecodear?



¿Se puede vibecodear un Data Lake?



Lo intenté.

No solo lo intenté, lo hice con las herramientas recién salidas.





Andrej Karpathy ✓

@karpathy



There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to



El arranque

Marzo 2025, vacaciones, Sonnet 3.7 y la fintech “verde”



Lo barato, sale caro

- *Squads que usan un DynamoDB como base relacional.*
- *Cuando tienen que extraer la data para hacer analitica.*
- *Queries NoSQL.*
- *DataDog es duro con los precios y la retención.*
- *Producto necesita tableros.*



La necesidad

- *¿Cómo sacamos la data?*
- *¡La necesitamos a la hora!*
- *Tenemos que “planchar los JSONs”*



La necesidad

- *Kinesis Firehose al rescate: cada Ítem nuevo o modificado llegaría como JSON al bucket.*
- *Luego seguiría el camino de los otros ETLs.*
- *¿Se imaginan donde vendría el problema después?*



Brownfield

- *El Lake ya existía con RDS, DMS, S3, Glue, Spark, DBT.*
- *Todas ingestas con Full Load y CDCs.*
- *Clásico Medallion: Bronze, Silver, Gold.*
- *Airflow para el orquestado, en otro repo.*
- *Infraestructura en Terraform con estado remoto dev/prod.*
- *CI/CD con Github Actions.*

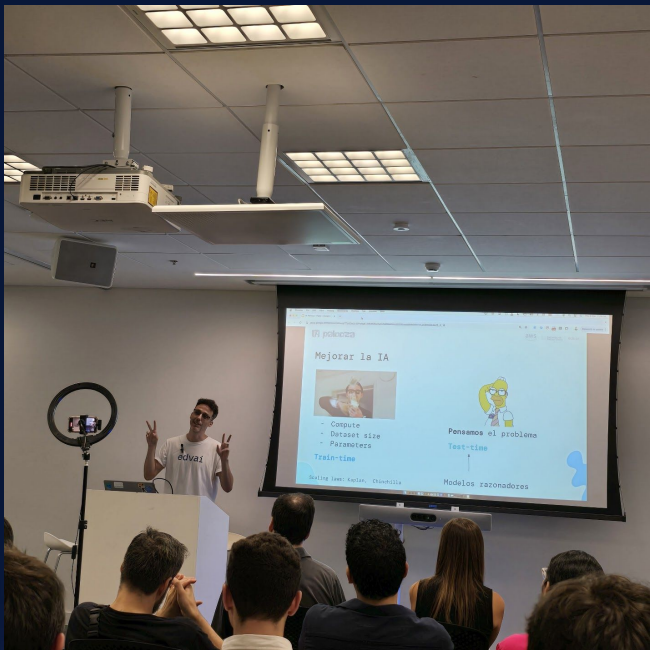


¿Ya llegamos a la parte de IA?



Marzo 2025

- *Asistí a un Meetup en las oficinas de AWS en Av. Libertador.*
- *@Sicarul me dijo “¿Probaste la nueva herramienta de Anthropic?”
“Tira comandos”*



Marzo 2025

- *Mi vida se pasa entre terraform destroy y terraform apply*
- *(En módulos, no se asusten)*

CLAUDE
CODE



Marzo 2025

- *Partía de una base de código con convenciones definidas.*
- *Tenía que respetar un estilo y sumar valor sin romper lo que ya funcionaba.*
- *Venía usando LLMs para correcciones puntuales.*
- *Acá me animé a un proyecto completo con Claude Code desde bash. Otra forma de interactuar con IA.*



```
: 1741796564:0;npm install -g @anthropic-ai/claude-code  
: 1741796685:0;npm config set os linux  
: 1741796695:0;npm install -g @anthropic-ai/claude-code --force --no-os-check  
: 1741796721:0;npm install -g @anthropic-ai/claude-code  
: 1741796726:0;npm install @anthropic-ai/claude-code
```



Diagnóstico de Claude Code

- *Primero que te hace es un archivo CLAUDE.md*
 - # What to do*
 - # Data Flow*
 - # Lambda and Glue Jobs*
- *(que no sabes si versionar o no)*



Aprendizaje 1

Cuidado con el Config Hell, deja todo en un solo lugar. Todavía faltaba para la fiebre de los MCPs.



RULES

- ✓  `.cursor`
 - ✓  `rules`
 -  `instructions.mdc`
- ✓  `.github`
 -  `copilot-instructions.md`
- ✓  `.kiro`
 - ✓  `steering`
 -  `instructions.md`
- ✓  `.windsurf`
 - ✓  `rules`
 -  `instructions.md`
 -  `.instructions.md`
 -  `.prompt`
 -  `.rules`
 -  `AGENT.md`
 -  `AGENTS.md`
 -  `CLAUDE.md`
 -  `CONVENTIONS.md`
 -  `global_rules.md`

MCP

- ✓  `.codeium`
 - ✓  `windsurf`
 -  `mcp_config.json`
- ✓  `.codex`
 -  `config.toml`
- ✓  `.cursor`
 -  `mcp.json`
- ✓  `.kiro`
 - ✓  `settings`
 -  `mcp.json`
- ✓  `.vscode`
 -  `mcp.json`
- ✓  `.zed`
 -  `settings.json`
 -  `.windsurfrules`
 -  `opencode.json`



Prompting en la CLI

- *I need to develop a new type of ingest...*
- *Por las dudas le escribi en ingles, y sigo haciéndolo.*
- *Tranqui, te entiende...*
- *Leete la cadena de pensamiento.*



Aprendizaje 2

Anda (sigue) mejor en inglés.



 Pinned



Andrej Karpathy  @karpathy · Jan 24, 2023



The hottest new programming language is English

 1.3K

 7.6K

 48K

 8M



¿Vas a seguir citando a Karpathy?



¿Qué salió bien al principio?

- *Velocidad: MVP que estimaba en 2 semanas lo resolví en 2 tardes.*
- *Claude Code me proponía pasos y me guiaba archivo por archivo.*
- *Ahora lo hace automático, pero primero pedí plan y después aprobá.*



Prompt Engineering

- *No se puede promptear lo que no conoces.*
- *Manejar conceptos: estadio, intermediate, lake.*
- *En esos momentos no había guías para promptear este tipo de agentes.*
- *No, No, No.*



Aprendizaje 3

Aprendé las bases conceptuales.



Sobreingeniería

- *Es eager. Es verbose.*
- *Al menos 20% del código, hay que tirarlo.*
- *Resuelve el 80%, el resto queda para mi.*



Aprendizaje 4

Si anda, refactorizá, con IA obvio.



Checkpoints

- *Todo puede desmadrarse en el próximo prompt.*



Aprendizaje 5

Versioná cada estadio bueno.



Aprendizaje 6

Versioná también tus prompts.



Pruebas locales

- *O combinadas con Staging, si tenes ese ambiente, deberías.*
- *Unit tests, que difícil que es probar infraestructura, mocks.*
- *Data sintética o un trozo de la data real para pruebas.*



Aprendizaje 7

Probá local, probá seguido.



Contexto y Costo

- *25 U\$S por tarde*
- *Dale al LLM lo mínimo que necesita para trabajar.*

```
You've spent $5 on the Anthropic API this session.  
Learn more about how to monitor your spending:  
https://docs.anthropic.com/s/claude-code-cost
```

```
> Got it, thanks!
```



Aprendizaje 8

Cuidá el contexto, cuidá los tokens, cuidá la billetera





Validación y pruebas “a lo real”



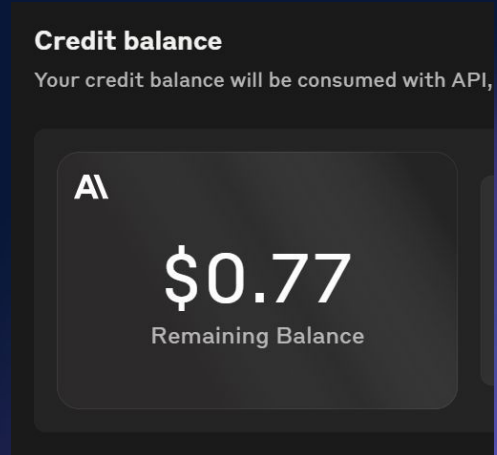
Pruebas en Staging

- *Funcionó la ingesta local, vamos a Staging.*
- *Llega Airflow, en otro repo.*
- *Tuve que hacer enlaces simbólicos de otro repo.*
- *Con una fracción del volumen de producción.*



Costos y small files

- *Costo en Staging rondando USD ~1.000/mes con poco volumen.*
- *Granularidad horaria → demasiados objetos en S3.*
- *JSON inconsistente en algunos lotes.*
- *Frecuencia de los crawlers muy alta.*



Aprendizaje 9

No tenía nociones de volumen, caro, faltaba SWE.



Refactors que cambiaron el juego



Refactors de Ingeniería

- *Bajamos la cantidad de archivos chicos. Hora a dia.*
- *Removimos el estadio silver porque había pocas transformaciones intermedias.*
- *Compactamos en archivos más grandes con parquet.*
- *Arreglamos, anduvo mejor, a producción.*



¿Que pasó en producción?

- *Incidente.*
- *Sobreescribió los triggers del bucket de Landing, no arrancaban los ETLs.*
- *Bajé 12 horas toda la infra del equipo de Analítica.*
- *40 años, no es la primera vez que bajo producción.*



Aprendizaje 10

Si vibecodeas, bajás producción.



Documentando

- *Una vez resuelto el incidente, terminamos el proyecto!*
- *Dejar que sugiera mensajes de commit/PR, pero no pushear nada que no pueda explicar.*
- *Aprendí sobre módulos de Terraform y su gestión de estados.*



Aprendizaje 11

Hacé pero explicame, quiero aprender.



Reglas del Vibecoding

- *¿En qué punto le pones tantas reglas a Vibecodear que deja de ser “Vibe”?*



Aprendizajes para no repetir mis errores



Aprendizajes

- *Dejá toda la “configuración para IA” todo en un solo lugar.*
- *Promptea en inglés.*
- *Aprendé los conceptos de base.*
- *Refactorizá el exceso de código.*
- *Versioná cada estadio bueno de tu aplicación.*



Aprendizajes

- *Versioná también tus prompts.*
- *Probá local, probá seguido.*
- *Cuidá el contexto, cuidá los tokens, cuidá la billetera.*
- *Aplicá buenas prácticas de SWE.*
- *Más cuidado que nunca en Producción.*



Aprendizajes

- *Pedí explicaciones, usalo para aprender.*



Q&A





aws Community Day
== ARGENTINA ==

¡Gracias!

Ramiro Savoie - deployr.ai
Contacto ramiro@deployr.ai

