

Abraham José Kazerian Blengio

Desarrollador backend senior. Construyo y acelero sistemas serverless en AWS con Node.js y TypeScript: más de 8 años haciendo software, desde 2023 para un cliente financiero de EE.UU. Montevideo, Uruguay · hello@abrahamkazerian.dev · abrahamkazerian.dev · linkedin.com/in/abraham-kazerian-3ba470ba

SOBRE MÍ

Soy desarrollador backend de Montevideo, Uruguay. Desde 2017 construyo productos para fintech, crypto, retail y salud; empecé como full stack con Angular, React, React Native y Node.js.

Desde 2023 trabajo a través de DEPT para un cliente financiero de EE.UU., construyendo backends serverless en AWS con Node.js y TypeScript. Me importan los sistemas rápidos y fáciles de mantener: armé el middleware compartido sobre el que corren nuestras Lambdas y reduje un 70% el tiempo de ejecución de las más pesadas.

Trabajo en inglés todos los días y estoy abierto a puestos remotos con equipos de EE.UU. y Europa, y a proyectos freelance.

EXPERIENCIA

DEPT

Senior Software Developer · Ene 2023 - Actualidad

- Contractor para un cliente financiero de EE.UU.: backends serverless en AWS Lambda con Node.js y TypeScript, y React en el frontend.
- Armé un middleware compartido para las Lambdas (validación, logs y manejo de errores) que se reutiliza entre funciones y hace el código más fácil de mantener y leer.
- Reduje un 70% (2,5 s → 800 ms) el tiempo de ejecución de las Lambdas más pesadas sobre millones de registros: ajuste de queries, joins e índices, y reescritura completa de la lógica del backend (ver el caso destacado).
- Planifico y entrego funcionalidades de punta a punta, y hago code review en el equipo.

Globant

Senior Software Developer · Dic 2021 - Dic 2022

- NestWealth (National Bank of Canada): nuevas funcionalidades para un portal de apertura de cuentas bancarias online, con KYC, datos personales y firma de documentos. Angular, NestJS y PostgreSQL.
- Investigué e introduje NgRx como fuente única de verdad para el estado compartido entre componentes, con lo que se terminaron las inconsistencias de datos entre ellos, y lo dejé como un patrón que cualquiera del equipo podía seguir para sumar estado nuevo.

Switch Software Solutions

Semi Senior Software Developer · Feb 2021 - Dic 2021

- Tata (cadena de supermercados de Uruguay): app en React Native para hacer el pedido y retirarlo en el local, con backend en Node.js y MySQL. Mejoré un backend cuyas requests tardaban demasiado.

SpaceDev

Software Developer · Sep 2018 - Feb 2021

- Ndax: exchange de criptomonedas (compra, venta, alertas de precio). Angular, Node.js, MongoDB y AlphaPoint.
- Flexio: plataforma de links de pago para que los comercios vendan en cuotas, integrada con SR.Pago. Angular, Node.js y MongoDB.
- CP&A: app offline-first en Ionic para que los vendedores tomen pedidos en supermercados.
- Somos.UY: plataforma para contratar enfermeros que cuiden a familiares. AngularJS, Node.js y MongoDB.

Dedicado a estudiar

Sep 2017 - Sep 2018

Altimetrik

Software Developer · May 2017 - Sep 2017

- Migré componentes legacy de Salesforce de un cliente de EE.UU. a componentes Lightning.

CASO DESTACADO

-70% de tiempo de ejecución en una Lambda crítica

Trabajo actual en DEPT · para un cliente financiero de EE.UU.

PROBLEMA	La Lambda más pesada corría una query compleja que traía mucha información sobre millones de registros, y gastaba buena parte del tiempo ordenándolos en la base de datos.
QUÉ HICE	Lo atacué por varios frentes: simplifiqué las queries, mejoré los joins y los índices, y reescribí desde cero la lógica del backend detrás. La reescritura también sacó el ordenamiento de la base: los resultados se agrupan en un objeto cuyas claves son un ID numérico chico (como mucho ~2.000 valores), que JavaScript recorre en orden ascendente. En vez de un ORDER BY sobre millones de filas, quedan ordenados solo un par de miles de grupos.
RESULTADO	De ~2,5 s a ~800 ms (600 ms en el mejor caso): alrededor de 70% menos de tiempo de ejecución, y menos uso de CPU y memoria, medido con CloudWatch y Splunk.

Node.js · TypeScript · AWS Lambda · SQL · CloudWatch · Splunk

PROYECTO PERSONAL

tialuk

SIN LANZAR

LA IDEA	Las pymes responden los DMs de venta desde el celular, en varias cuentas a la vez. tialuk los junta en una bandeja en tiempo real, marca la intención y la urgencia, resume cada conversación y ayuda a redactar respuestas, enmascarando los datos personales antes de que lleguen al modelo de IA.
QUÉ CONSTRUI	Webhooks de Meta verificados con HMAC y encolados en un outbox transaccional en PostgreSQL (FOR UPDATE SKIP LOCKED): respuesta inmediata, procesamiento idempotente y actualizaciones en tiempo real por SSE. IA con Vertex AI (Gemini) para intención, urgencia, resúmenes y reescritura de borradores, con enmascarado de datos personales, registro de consumo por empresa, topes diarios y créditos prepagos para controlar el costo. Seguridad multi-tenant: tokens de Meta cifrados con AES-256-GCM, JWT en cookies HttpOnly con refresh rotativo, Argon2id y el state de OAuth atado a la sesión. Infraestructura como código con Terraform sobre GCP (Cloud Run, Cloud SQL privado, Secret Manager, CI/CD sin claves con Workload Identity Federation) y Cloudflare Workers. Cada deploy migra primero, hace un smoke test y vuelve atrás solo si falla.
DÓNDE QUEDÓ	Llegó a un entorno de desarrollo funcionando de punta a punta, pero no salió a producción: quedan pendientes la App Review de Meta y armar el entorno de producción.
CÓDIGO	Repositorio privado. Con gusto lo recorro en una entrevista o doy acceso temporal de solo lectura si me lo piden.
CÓMO	Lo construí solo entre mayo y julio de 2026 con un flujo de desarrollo asistido por IA (Claude Code), respaldado por documentos de diseño y decisiones de arquitectura registradas para cada elección clave.

~22k líneas de TypeScript en el backend ~965 tests automatizados 50+ endpoints de API

Bun · ElysiaJS · TypeScript · PostgreSQL · MikroORM · SvelteKit · Vertex AI · Terraform · GCP · Cloudflare Workers · GitHub Actions

SKILLS

SERVERLESS EN AWS	Lambda · SQS · SNS · API Gateway · DynamoDB · ECS · CloudWatch
BACKEND	Node.js · TypeScript · JavaScript · NestJS · Jest · Bun · Vertex AI · Gemini
INFRAESTRUCTURA Y ENTREGA	Terraform · Google Cloud · AWS CDK · Serverless Framework · Docker · GitHub Actions · Datadog
BASES DE DATOS	PostgreSQL · MySQL · MongoDB
FRONTEND Y MOBILE	React · React Native · Angular · Ionic · SvelteKit