VOL. X | CURRICULUM VITÆ | 2026 EDITION

Raphael Bernardo

Senior AI / LLM Systems Engineer · *Agents, Orchestration & Local Inference*
Ex-Uber (contract) · ten-plus years of production engineering, now building the systems that run AI agents: multi-provider routing, agent tooling over MCP, retrieval and knowledge graphs, evaluation gates, and GPU inference on my own machine fleet.

MAIL raphaelbgr@gmail.com TEL +55 21 98885-6697 GIT github.com/raphaelbgr
IN linkedin.com/in/raphaelbgr

LOC Rio de Janeiro · GMT−3 CIT Brazilian + Portuguese (EU) dual citizen
LANG Portuguese (native) · English (fluent) · Spanish (intermediate)
ENG Offshore B2B contractor (own CNPJ) · full US Eastern overlap

§ 01 Summary

PROFILE · POSITIONING

Senior engineer who designs and operates AI systems, not only calls to them. I run a multi-provider LLM dispatch mesh across my own machine fleet (CUDA, Apple Metal, Linux, Windows, macOS), with cost-aware routing over 8 providers and 22 models, agent tooling exposed over MCP, and every external model's output reviewed before it is accepted. On the same fleet I serve local models behind an OpenAI-compatible coordinator that admits a request only if the model and its KV cache fit a node's free VRAM. I build retrieval and knowledge-graph pipelines, computer vision and speech services and LoRA fine-tuning for video, and keep agent work verifiable through written completion gates, execution-trace contracts and closed-loop evaluation against a running app. That AI work sits on ten-plus years of production software: Uber's Maps SDK in JNI/C++, OTT streaming platforms with 4K multi-DRM playback, and apps I publish myself on Google Play and the App Store.

§ 02 AI Engineering

BUILT AND RUN ON MY OWN MACHINES · NOT IN ANY CLIENT CODEBASE

Orchestration and model routing

A dispatch mesh with one HTTP daemon per machine, peer auto-discovery, WebSocket output streaming and full job telemetry, exposed to the orchestrating agent as **15 MCP tools**.

- Cost-aware routing over **8 providers and 22 models in 6 tiers**: local models (Ollama, LM Studio), OpenRouter, Z.AI (GLM), Moonshot (Kimi), Anthropic, OpenAI and xAI. Rate-limit backoff, per-provider cooldowns and capability matching apply in order; flat-rate providers are serial-only, so a parallel fan-out cannot exhaust a subscription's

concurrency limit.

- Every external model's output is reviewed against the brief before it is accepted (each changed line traced, tests actually run), with up to two retries and then escalation to a stronger tier.
- Encrypted job transport keyed off the existing SSH identity: **RSA-OAEP + AES-256-GCM**, fresh key and nonce per message.
- **Command Center:** a Node.js daemon on every fleet machine for signed REST and WebSocket remote execution, coding-agent CLI sessions run as jobs, token and cost analytics, peer discovery, self-update and a dashboard.

Agents, tools and control surfaces

A **skills system** for coding agents: 75 global skills in three scoping tiers (global, project, archived), an auto-creation gate, and a ranked search over the whole corpus before any capability is declared missing.

- Planner, executor, evaluator and monitor agents for multi-step work; a local LLM that supervises a headless coding agent, detects when it stops and decides the next action.
- Chat as a control surface: a **Slack** gateway (Bolt, Socket Mode) that turns messages into headless agent runs, with a serial queue per project, timeouts, a post-hoc reviewer and an SQLite session store. **Telegram** alerts for long-running jobs (crash, start, plateau).
- A fleet session manager that discovers and resumes agent sessions across five machines from one place (terminal UI, web UI and desktop window over REST and WebSocket).
- An agent-facing product: a live pay-per-call API that AI agents call and pay for per request (**x402**, over HTTP or as MCP tools), with a durable job store so a lost response is recovered without charging twice.

Inference and GPU operations

An **OpenAI-compatible coordinator** in front of a CUDA node (RTX 3080 Ti) and an Apple M4 node, with queueing, load-balancing, grouping by model for warm reuse and auto-pull of Hugging Face GGUF models.

- Admission control rejects a request with HTTP 507 when the model weights plus KV cache fit no node, instead of silently falling back to CPU; a per-node fitter picks the best quantization and largest context for each GPU.
- A llama.cpp model on a rented cloud GPU behind an SSH tunnel with a 131,072-token context; the agent runtime caps input at 114,688 tokens, summarises history at 98,304 and runs inside a Docker sandbox.
- **VRAM accounting:** every GPU allocation is registered before it happens (measured size, purpose, expected release); stale or overdue claims are detected and the exact holder is reported. Services lazy-load and unload after 30 idle minutes.

Computer vision, speech and fine-tuning

GPU inference for vision and audio in my own products, from face recognition to karaoke preparation and video LoRAs.

- Face and person recognition with **InsightFace on ONNX Runtime (CUDA)** and FAISS embeddings, distributed across GPU workers; local vision-language captions (Qwen2.5-VL) for photos and video keyframes.
- Vision grounding: a local server that maps an image plus a natural-language query to boxes or points (NVIDIA LocateAnything-3B), with a pinned model revision and model output treated as data, never as instructions.
- Hand-gesture control: **MediaPipe Gesture Recognizer (TensorFlow Lite)** in the phone's browser on WebAssembly and GPU, streaming 21 hand landmarks at about 30 Hz over WebSocket to a tvOS Swift package, so a phone becomes a multiplayer motion controller with no install.
- Speech recognition and source separation: **faster-whisper** (large-v3-turbo) and **Demucs** vocal and instrument separation in a queued service with leases, heartbeats and idempotent revisions.
- **LoRA fine-tuning for video:** dataset preparation and validation, still-image LoRA training (Flux) and image-to-video LoRA training (Wan 2.2), run locally through ComfyUI workflows.

Retrieval and knowledge graphs

A fleet **knowledge graph** of 3,489 nodes and 83 communities extracted from 2,018 files across six machines, with a reproducible pipeline (harvest, merge, chunk, extract, build) and graph queries for agents.

- A **RAG** research agent (Agno) with parallel web search and a context limited to the latest refinements plus a knowledge base, so it runs on small local models.
- Embedding search: FAISS face-embedding search over a personal media archive; embedding-based ranking of CVE descriptions to prioritise fuzzing targets.

Evaluation and completion gates

Substantial agent work counts as done only when a written ledger of runnable checks (a command plus the output pattern it must match) passes, with a lint step and re-verification.

- Closed-loop evaluation on my own TV app: an on-device HTTP diagnostic surface with **149 named test routines** returning structured pass/fail; an agent drives the app and evaluates what the screen and the event stream actually show.
- Benchmark-gated model adoption for generative video: a fixed four-scenario benchmark, scored before any model is adopted, with a manifest per run so results are reproducible.

Observability and fleet operations

A language-neutral **trace-contract** standard (JSON Schemas, a cross-language conformance oracle, golden cases) and an agent plugin whose hook blocks completion while the reference graph is stale.

- Process hygiene for agents: every spawned process is tagged and registered with an owner and an expiry before it runs; long jobs write heartbeats and checkpoint each unit, so they can pause, crash and resume where they stopped.
- Manifest-driven deployment across five machines under launchd, systemd and Task Scheduler: each app declares install, launch and health, and deployment is a git push.

§ 03 Experience

ENGAGEMENTS · 2014 → PRESENT

Nexstar Media Group, Inc.

Aug 2025 — Present · current

Senior Fire OS / Android TV Engineer · full remote CONTRACT

- Run a **client-approved agent-driven delivery workflow** end to end: one command starts it, an agent picks up the tracker issue with its acceptance criteria, works the change on an isolated branch, runs the test suite alongside agent-driven end-to-end checks against a running build, and opens a **GitHub** pull request carrying the issue context and test evidence. **GitHub Actions** runs the pipeline; a second model audits the diff against the original request before anything is reported done.
- Contributed to the architecture of an enterprise **OTT streaming framework** (MVVM + Clean Architecture, plugin-based integrations); five specialised Media3 / ExoPlayer implementations; DRM-protected playback (Widevine, PlayReady) with server- and client-side ad insertion.

FreeCast, Inc.

Mar 2021 — Jul 2025 · 4 yrs 5 mo

Android / Android TV / Fire OS Engineer · part-time · EST hours · full remote CONTRACT

- Led the Java-to-Kotlin migration with Clean Architecture; owned DRM, EPG, tuner orchestration, transcoding and multi-device streaming end to end; AI-assisted development on the migration and on test coverage, later with coding agents as a primary development method and automated verification in CI before review.

Claro TV+

Apr 2024 — Dec 2024 · 8 mo

Android TV / Fire OS Engineer · part-time · full remote · via Global Hitss FREELANCE

- Greenfield Android TV / Fire OS app for one of Brazil's largest pay-TV operators: 4K DASH/HLS with multi-DRM and a shared Kotlin Multiplatform data layer for Android, webOS and Tizen; AI-assisted code generation and review across the build.

Uber Technologies

May 2021 — May 2022 · 1 yr

Android Engineer, Maps SDK team · PST hours · full remote · via Matchpoint Solutions CONTRACT

- Maps SDK module in **JNI / C++** inside Uber's RIBs architecture and internal monorepo, across a 10K+ engineer codebase; shipped to Uber Rider, Uber Eats and Uber Driver.

Earlier

2014 — 2021

- Super Revendedores, Android Engineer (2018–2021; AWS SQS/SNS order processing) · Concrete / Accenture, Jr. Android Engineer (2016–2018; Minha CVC, 1M+ downloads) · PraPagar, Jr. Android Engineer (2015–2016) · IBM Brasil, intern (2014).

§ 04 Products & Publications

BOOK · APP STORE · GOOGLE PLAY · WEB · API



Automate Your Work Once and for All

BOOK · APPLE BOOKS · GOOGLE PLAY BOOKS

A practical ebook about AI-assisted workflows, reusable skills, connected tools and evidence-based checks.

Apple Books Google Play

Archive — Home Video Player

ANDROID TV · APPLE TV (TVOS)

TV client for my self-hosted media library, built natively for each platform: HLS playback, server-side transcoding, library browsing and resume across devices.

Kotlin, Media3, Compose for TV, Room · Swift, SwiftUI, AVFoundation, HLS

Google Play App Store

IPTV Player

APPLE TV (TVOS) · IPHONE & IPAD (IOS)

Player for the user's own Xtream accounts and M3U playlists: live TV with an EPG guide, movies and series with resume, HLS, MP4 and MPEG-TS playback, an on-device library that loads in small batches so large lists stay responsive, and private iCloud sync.

Swift, SwiftUI, AVKit, GRDB (SQLite), CloudKit, StoreKit 2

App Store · tvOS & iOS

Beer Price Calculator

ANDROID · IPHONE & IPAD (IOS)

Compares beer prices by volume to find the best value per millilitre, built natively for each platform. Offline-first with local persistence and in-app billing on Android.

Kotlin, Jetpack Compose, Room, Play Billing · Swift, SwiftUI

Google Play App Store

ADAMA Studio

WEB APP FOR ARCHITECTS

Turns a floor plan into an engineering model in the browser: upload the plan, let the AI draft it in 3D, review it in conversation and in 2D, isometric and room views, then export to DXF, IFC and Revit. Each account connects its own AI provider.

Next.js, React, TypeScript, Three.js, Better Auth

vilanova.fastsoftware.uk

RugShield

PAY-PER-CALL API FOR AI AGENTS · PERSONAL AGENTIC PROJECT

A live API that AI agents call, and pay for per call, to check a Solana token or an unsigned transaction before signing, or to ask several independent risk advisors in parallel. Agents pay in USDC on Base or Solana through x402 (HTTP 402 payments, no account or API key), over plain HTTP or as MCP tools. A durable job store saves each paid result before the payment settles, so a lost response can be recovered without charging twice.

Python, FastAPI, x402, MCP, SQLite, Fly.io

getrugshield.com

§ 05 Education

FORMAL · CONTINUING

Instituto Infnet · Rio de Janeiro 2013 — 2018

B.S. Computer Engineering — software development, mathematical modelling, OS & hardware architecture.

§ 06 Technical Stack

CORE COMPETENCIES

AI & LLM	Multi-provider routing · Agent architectures · MCP servers & tools · Coding-agent skills · RAG · embeddings · Knowledge graphs · Evaluation & completion gates · LoRA fine-tuning
INFERENCE · GPU	Ollama · LM Studio · llama.cpp · LiteLLM · GGUF quantization · KV-cache-aware VRAM admission · CUDA · Apple Metal
VISION · SPEECH	InsightFace · FAISS · ONNX Runtime · MediaPipe · TF Lite · Qwen2.5-VL · faster-whisper · CTranslate2 · Demucs · ComfyUI
LANGUAGES	Python · Kotlin · Java · Swift · C++ · TypeScript · SQL
BACKEND · INFRA	FastAPI · asyncio · Node.js · REST · WebSocket · SQLite · PostgreSQL · Docker · GitHub Actions · Fly.io · Cloudflare · Tailscale
MOBILE · TV	Android SDK · Jetpack Compose · Media3 / ExoPlayer · Kotlin Multiplatform · NDK · JNI · iOS · tvOS · SwiftUI · Widevine · PlayReady