

Yampier Medina

Forward Deployed Engineer · Applied AI on AWS

Indianapolis, IN · Remote or on-site · Permanent Resident

[Email](#) · [Portfolio](#)

[GitHub](#) · [LinkedIn](#)

SUMMARY

Forward deployed engineer: I work with the people who have the problem, turn it into a written spec, and ship the software that fixes it, from React + TypeScript interfaces to Node.js services and **GraphQL APIs running serverless on AWS Amplify**. My workflow is **agent-first and spec-driven**: every feature starts as a written spec (requirements, design, tasks), coding agents implement it, and I review, test and verify the result against the running app before it ships. I maintain the context agents work from: shared engineering standards, repository instruction files, and reusable skills used across all my projects. I've built React apps on AWS since 2020, and before that I spent six years in formal software quality, which is why **automated tests are my safety net for agent-written code**.

69

features shipped spec-first with coding agents

17

Lambda functions incl. event-driven job pipelines

200+

automated test suites incl. 28 Playwright end-to-end specs

WCAG

axe-core checks, keyboard and screen-reader support

SKILLS

Agentic development

Claude Code and Kiro as the main implementation surface; spec-driven development (EARS acceptance criteria > design > tasks); context engineering with instruction files, standards and skill libraries; MCP servers for browser and AWS access.

Back end & APIs

Node.js / TypeScript Lambda handlers, GraphQL (AppSync) with typed contracts, REST route handlers, webhooks, Zod validation, event-driven dispatcher/worker pipelines.

AI product features

Claude on Amazon Bedrock, OpenAI Realtime (voice over WebRTC), streaming chat, structured extraction, model catalogs with per-route configuration, Anthropic and OpenAI SDKs.

Front end

React 19, TypeScript, Next.js (App Router, server components, server actions), Tailwind CSS, Headless UI, React Aria, next-intl (14 locales), accessibility (WCAG 2.x AA) and performance.

AWS & infrastructure as code

Lambda (+ layers), DynamoDB, Cognito, AppSync, S3, IAM, CloudWatch, Bedrock, Polly, Amplify Gen 2. AWS CDK (TypeScript) deployed as CloudFormation; per-branch environments.

Testing & delivery

Playwright end-to-end, Jest + React Testing Library, property-based tests (fast-check), pre-merge quality gates, staging-first deploys, code review, Git/GitHub, GitHub Actions automation.

TypeScript

React

Next.js

Node.js

AWS Serverless

AWS Amplify

Claude Code

Spec-driven development

GraphQL

AWS CDK

DynamoDB

MCP

Playwright

WCAG

CURRENTLY BUILDING

Back to Fluency

[Website](#)

AI English-fluency platform

Voice tutor, streaming chat, 17 Lambdas, Bedrock, Stripe billing. 31 features shipped spec-first.

20 Playwright specs · 100+ unit/integration suites

Speak Level

[project page](#)

Speaking-assessment API and learner dashboard

Async audio-scoring jobs, API-key platform, dashboards. 38 features shipped spec-first.

8 Playwright specs · 80 unit/integration suites

EXPERIENCE

Independent Full Stack Engineer — AI-Native Products

2015 – Present

Self-employed · Indianapolis, IN (remote)

- **Agent-first, spec-driven delivery:** every GitHub issue becomes a reviewed spec (EARS acceptance criteria, design, task list) before any code; coding agents execute the tasks and I review, test and verify each change against the running app. Nothing ships on a green build alone.
- **Context engineering:** wrote and maintain the engineering standards, repository instruction files and reusable agent skills (issue > spec, spec > implementation, manual QA suites) that every project installs, so humans and agents work to the same bar.
- **Features end to end:** React/TypeScript UI through Node.js Lambda functions and GraphQL APIs to production on AWS, across learner, admin and marketing surfaces in 14 locales.
- **Event-driven services:** dispatcher/worker Lambda pipelines for long-running text-to-speech, translation and audio jobs, and async jobs that split long audio so no function hits its time limit.
- **AI product features:** real-time voice tutor (WebRTC), streaming chat, Claude on Bedrock for content generation and structured extraction, with a curated model catalog and per-route settings.
- **Contracts and infrastructure:** typed API and data contracts, including a cross-product handoff tested on both sides; the whole backend is CDK (CloudFormation) with a staging stack per branch.
- **Tests as the safety net:** 28 Playwright end-to-end specs and 180+ unit/integration suites that test behavior, not implementation; property-based tests caught billing rounding and boundary bugs that hand-written tests had missed.
- **Security:** Cognito group-based authorization, secrets kept in Lambda rather than build variables, API keys with per-key quotas, owner-scoped data, SHA-pinned binaries that fail closed.

Software Engineer · Analyst, Planner & Head of Quality

2009 – 2015

GEySED Software Development Center, University of Computer Science (UCI) · Cuba

- Gathered and specified requirements, produced UML analysis and design artifacts, and planned iterations for the center's software projects.
- Led quality for teams of up to ~50 engineers: test strategy, code and document reviews, audits, and status reporting to management; taught software engineering as university faculty.
- Planning & Control Advisor during the center's preparation for international **CMMI Level 2** certification, which it obtained.

EDUCATION

B.S., Engineer in Computer Science — University of Computer Science (UCI), Cuba, 2009

Coursework: Algorithms & data structures (C++) · OOP (C++, C#, PHP, JavaScript) · Database Systems (SQL Server, PostgreSQL, MySQL) · Software Engineering (UML, RUP, XP, Scrum) · Quality Management · CMMI

Continuing education: Epic React (Kent C. Dodds, 2020–21)

HOW I WORK

- Work where the problem is, with the people who have it, until it runs in production.
- Agents draft; I own architecture, security and the decision to ship.
- Specs before code, tests before trust.
- Improve the shared assets that make the whole team faster.