

Gonzalo Argañaraz

Senior Backend Engineer (.NET/C#) | AI-Native Delivery with Multi-Agent Workflows

Córdoba, Argentina (UTC-3) | Open to remote, full-time (employee or contractor) | English C1

arggondev@gmail.com | linkedin.com/in/arggon | github.com/Arggon | arggon.github.io/arggon-cv

SUMMARY

Senior .NET/C# backend engineer who ships features end to end with multi-agent AI workflows (Claude Code, GitHub Copilot, OpenCode). Creator of ArggonManager, a spec-driven methodology for AI coding agents, and designer of a multi-agent SDLC pipeline at Coalfire (proofs of concept). Strong in APIs, SQL Server performance (cut client processing time 80% with a batch endpoint) and verifying AI output with tests and review. Remote from Córdoba, Argentina (UTC-3), available to overlap with US working hours.

EXPERIENCE

Software Engineer to Senior AI Software Engineer | GlobalLogic | Apr 2022 – Aug 2026

Remote. Engineering services company; assigned to US client projects.

Coalfire (cybersecurity and compliance) | Senior AI Software Engineer | May 2026

- Owned agent design and the delivery methodology in a three-person AI team, alongside a project manager who presented the proposal to the client and an engineer who tested the agents against the methodology.
- Designed a multi-agent SDLC pipeline: a transcript agent extracts specs from meeting transcripts, a backlog agent turns them into work items, a developer agent implements and opens a pull request, a QA agent validates it with the Playwright CLI, and a tech-lead agent reviews the PR. Any stage hands work back when a check fails.
- Built the pipeline on a Spec-Driven Development (SDD) methodology, drafted with Claude Sonnet and validated through proofs of concept in GitHub Copilot (GPT-4o).
- Extended the multi-agent SDLC methodology with test-driven development: one behavior at a time, failing test first, then implementation, then refactor.
- Refactored the agents' skills to focus on requirement gathering and clear feature scope before any code is generated.

Coalfire | Senior Software Engineer | Jul 2025 – Dec 2025

- Migrated critical endpoints from a legacy monolithic API to a .NET 9 microservice, reducing response times by 30% under high load.
- Designed and implemented a new access role with its permissions and restrictions, enabling monetization through AWS Marketplace default offerings.

Lytix (video telematics and fleet safety) | Software Engineer II | Apr 2022 – May 2025

- Designed and built a bulk data endpoint that replaced clients' one-by-one calls with a single batch request, backed by a new SQL Server stored procedure that processes the whole set in the database, cutting client processing time by 80%.
- Ran independent async operations in parallel with Task.WhenAll and integrated backend services with existing AWS Lambda functions.
- Mentored a mid-level developer through pair programming on business features and C# backend problems.
- Built advanced filtering so users could find the data they needed quickly instead of scanning large result sets.
- Diagnosed and fixed issues that spanned frontend, backend and database, improving system stability.

Backend Developer (contract) | Green Dot (fintech, payments) | Feb 2026

- Implemented CQRS with the Mediator pattern in a payments microservice, separating command and query handling.

Software Developer | e.tres (e-commerce) | 2021 – 2022

- Redesigned core transaction logic to fix payment gateway integration issues, improving reliability and reducing integration failures.
- Refactored critical parts of the codebase to match business requirements, improving readability and maintainability.

PROJECTS

dotnet-batchyard | Portfolio project, open source | .NET 10, SQL Server, YARP, k6, docker compose, GitHub Actions | github.com/Arggon/dotnet-batchyard

- Built an open-source .NET 10 showcase (docker compose, GitHub Actions) that moves batch ingestion out of a legacy monolith using the strangler pattern behind a YARP gateway; in k6 tests on a cloud VM at 500 items per request, median latency fell 93% (about 300 ms to 20 ms) and throughput rose from about 7.7k to 95k items/s. Built with my ArggonManager methodology.
- Built Docker setups and CI/CD pipelines with the help of AI agents in proof-of-concept work (this repo is the public example).

ArggonManager | Current focus | TypeScript, open source | github.com/Arggon/ArggonManager

- A methodology for delivering software with AI agents and humans in git, carried by a CLI, an MCP server and agent plugins. Runs on OpenCode, ZCode and Claude Code through one shared rules kernel, so the rules stay shared across tools.
- Roles are defined by what they decide: delivery lead, practice and standards, maker and verifier agents, plus a human product owner. Paperwork is sized by risk (task only, spec and plan, or exploration with a spec-analysis gate), and reviews are recorded on the work item as verdicts with evidence: commands run, expected versus observed.
- Done gate: an item can't close while any acceptance criterion is unticked. Waiving that gate, taking over a claim, reopening finished work and releasing are human-only, enforced by the tool rather than by prompt wording. Built and tracked with itself: 27 ADRs, 25 specs, published to npm with release-please.

SKILLS

- **Backend:** C#, .NET 9 / .NET Core, ASP.NET Core Web API, Entity Framework, REST API design, microservices, CQRS, Mediator pattern, Clean Architecture, SOLID
- **Data and messaging:** SQL Server (T-SQL, stored procedures), Redis, RabbitMQ, Kafka
- **Cloud and DevOps:** AWS (API Gateway, Lambda, S3, AWS Marketplace), Docker, CI/CD, Git
- **AI-assisted engineering:** agentic SDLC workflows, multi-agent orchestration, ArggonManager methodology, spec-driven development, TDD for AI-generated code, MCP, agent skills, OpenCode, ZCode, Claude Code, GitHub Copilot, Claude Sonnet, GPT-4o
- **Testing and observability:** xUnit, NUnit, Playwright, Vitest, Jasmine, Datadog, New Relic
- **Security and auth:** OAuth, JWT, IdentityServer, role-based access control
- **Frontend:** Angular 19, TypeScript, JavaScript, Angular Material, PrimeNG, HTML, CSS
- **Also:** Rust (learning), Go, Python (side projects)

EDUCATION

Técnico Programador Universitario (university degree in programming) | Universidad Tecnológica Nacional, Facultad Regional Tucumán (UTN FRT), Argentina | 2021

LANGUAGES AND COURSES

- Spanish (native), English C1 (EF SET certified)
- Courses (Udemy): Getting Started with .NET Core Clean Architecture; Enterprise Application Architecture with .NET Core