Artur Solodovnikov

AI developer · Russia and Asia · remote only

I work alone, with a team of AI agents.

Telegram @riviaConfig lomov_rh@outlook.com github.com/cosArthro asolodovnikov.dev

PROJECTS

OpenSolve open-solve.com

A research lab staffed by AI agents.

Launched 20 July 2026 · sole author: product, backend, frontend, smart contract, infrastructure

An agent reads skill.md, gets an API key and from then on works without a human. A big question is broken down into tasks; agents search PubMed, arXiv and OpenAlex, and RAG over scientific papers hands them the relevant passages. Several auditors check the work, each on their own, and a fact is accepted only when they agree. A synthesizer agent pulls the results together and finds what is missing: that becomes the next question.

82	848	3,096	2,495
agents signed up on their own	research tasks	submissions · 737 passed the audit	USDC paid to agents for audited work

figures as of 27 September 2026 · Agents were paid in USDC for audited work, out of the token's fees.

SOLVE staking, a Rust program on Token-2022, 32 tests, deployed to mainnet. The process: a CLAUDE.md with rules for the agents, a decision log, a roadmap in horizons, a bug register, a disaster-recovery plan.

Python, FastAPI, SQLAlchemy, Pydantic, PostgreSQL + pgvector, RAG (PyMuPDF, sentence-transformers) · Next.js 16, React 19, Tailwind, shadcn/ui, Clerk · Solana · Docker Compose, Caddy

481 commits · ≈119,000 lines · 926 tests · 76 database migrations

The protocol on GitHub The smart contract on Solana

ShadowJung The live landing demo

A Jungian AI analyst: it remembers dreams, notices when a person dodges a question, and doesn't rush to reassure. A Telegram bot, a website and a Mini App; v2 is in progress.

Product, design, code

The new version is built around memory: what the analyst knows about a person, and where it learned it.

Dreams	Recorded word for word, never paraphrased. The analyst throws out dream-dictionary readings and links dreams into series by shared images, the way Jung read them.
Only their words	Any observation about a person has to rest on a verbatim quote. No quote in the conversation, no observation: the analyst can't put words in anyone's mouth.
Patterns of character	Named in the person's own words, never as diagnoses. A pattern grows stronger when it comes up again in different conversations, and fades with time. Agreeing under pressure doesn't count. Code does the counting; the model only reads and sorts.
Safety first	If someone is in a bad way, Shadow work switches off completely, and no setting can turn it back on.

The stages: Persona, Shadow, Anima and Animus, Sage, Self. The bot, the website and the Mini App share one memory; voice messages are transcribed.

Landing page: four rounds of blind review by AI critics (the Gauntlet Loop); the final round beat two of three benchmark design sites.

v1: Python, aiogram 3, Claude API · v2: TypeScript, Fastify, grammY, PostgreSQL, Drizzle, SvelteKit, OGL

XENEURO xeneuro.org

A neuromorphic trading agent: a spiking neural network modelled on the nervous system of the clawed frog, 333 neurons and 2,938 synapses.

Product, code

New trading pairs land on the water like flies. The lateral line picks up the ripples, and the frog decides whether to swallow, spit out or digest: buy, sell or take profit. It learns from the outcome of every trade, as dopamine changes the strength of its synapses. For now it trades in simulation, on live market data.

Model	leaky integrate-and-fire, millisecond time steps
Brain	333 neurons, 2,938 synapses: AMPA/NMDA-like and inhibitory
Learning	dopamine modulation of synapses by the outcome of each trade
Data	live DexScreener and GeckoTerminal, contracts checked with GoPlus
Mode	simulation (paper trading)

Node.js, WebSocket, three.js · the spiking model is my own code

HOW I WORK

I work with AI agents the way you'd run a small team: I set the task, split it into parts and check each one. I read the code and the logs myself.

Rules. I write the agents a brief (CLAUDE.md): what we're building, what's off limits, how to check the work.

Decomposition. I cut a big task into pieces small enough that each one is either done or not.

Blind review. One agent builds; another compares the result with a benchmark without knowing which is which. Falls short? Another round. The method is Matt Shumer's Gauntlet Loop.

Tests. No tests, no merge. OpenSolve has 926 tests.

Logs. I keep a decision log, so it's always clear why something was done, and work picks up exactly where it stopped.

Tools. Claude Code is my main tool; before that, Cursor for a long time. I've tried Codex. My editor is VS Code. I've also worked with OpenClaw and Hermes Agent. Skills (instruction sets for an agent): I use ready-made ones and write my own. Agents get roles: builder, critic, scout. I write small scripts for my own everyday tasks and use LLMs all the time, not only for work.

Design. No fixed process. I like things simple and natural, with some character, and easy to use.

WHAT I CAN BUILD

AI agents and automation	Agents that sign up on their own, take tasks and follow a protocol. Payouts in cycles, smart-contract checks before a trade, scheduled messages. I've also worked with n8n.
RAG and semantic search	Scientific papers are cut into passages, a local model turns each into a vector, and pgvector in PostgreSQL finds what matches by meaning. That is how OpenSolve agents get the right passages from papers.
Telegram bots	Conversation, memory of the person, onboarding, and messages the bot sends first. Telegram sign-in, a Mini App.
Backend and integrations	FastAPI and PostgreSQL, auth, database migrations. Scientific databases (PubMed, arXiv, Crossref), market data (DexScreener, GeckoTerminal), sign-in with Clerk, crypto wallets.

	Dashboards and landing pages	Screens like the OpenSolve Observatory: live figures, a graph of tasks, a feed of events. After four rounds of work, the ShadowJung landing page beat the well-known design sites Astral Frontier and Persepolis Reimagined in a blind comparison.
	Tests	926 tests in OpenSolve, 32 for the smart contract, plus an audit report.

STACK

Working with agents	Claude Code, Cursor, Claude API, OpenClaw, Hermes Agent
Backend	Python, FastAPI, Pydantic, pytest, TypeScript, Node.js, Fastify
Databases	PostgreSQL, pgvector, SQLAlchemy, Drizzle, Alembic
RAG and semantic search	RAG, sentence-transformers, pgvector, PyMuPDF
APIs and integrations	REST API, Webhooks, WebSocket, Telegram Bot API
Frontend	Next.js, React, Tailwind, shadcn/ui, SvelteKit, three.js
Blockchain	Solana, Rust, USDC
Infrastructure	Docker, Docker Compose, Caddy, VPS
Automation	n8n
Earlier	Kotlin, Java

EXPERIENCE

At Sber and Yandex I answered small-business owners whose tools had stopped working. Good training: you see a product through the eyes of someone who is annoyed with it. Then key accounts at PIK, a property developer, and a year in a foundry.

October 2024 — present	**Freelance and my own products** · Software engineer Sniper bots, trading bots, mini-games for crypto projects, blockchain software; since February 2026 — OpenSolve, then ShadowJung and XENEURO.
September 2023 — September 2024	**Metallurg-Tulamash** · Fettler in the foundry
February 2023 — June 2023	**PIK Group** · Key account manager
December 2022 — February 2023	**Yandex** · Technical support specialist
May 2022 — November 2022	**Sber** · Technical support specialist
June 2021 — February 2022	**KARI** · Customer support, written enquiries
March 2021 — May 2021	**MegaFon** · Product promotion specialist

Education: Tula State University, incomplete degree · Language: Russian — native