

Ersapta Aristo

Full-Stack Software Engineer

Bengkayang, West Kalimantan, Indonesia • +62 812-8545-3405 • aristoersapta@gmail.com

www.linkedin.com/in/ersapta-aristo • github.com/tryea

SUMMARY

Proactive full-stack software engineer with over 5 years of professional experience who takes initiative, proposing ideas without being asked and willingly accepting the extra work they bring. Tests ideas through experiments: at a US software agency, proposed retrieval-augmented generation for an OpenAI chatbot and built it once the client approved; correct answers on a self-written 20-question test set rose from 8 to 17. Built the Supabase market-data platform of AlwaysLegit, Inc., a funded US startup; it sustained 24,000+ pipeline runs at a 99.98% success rate in ten days. At Monstyr, learned Neo4j for product recommendations and was the first on the team to bridge the React Native app to an Android package through native Java code. Interested in how technology grows a business: advised HyePair on pricing, a free trial, and a payment flow. Skilled in TypeScript, React, Next.js, React Native, Node.js, NestJS, PostgreSQL, and payments with Authorize.Net, Google Play Billing, and Midtrans. Seeking a remote Fullstack Developer role.

PROFESSIONAL EXPERIENCE

UPWORK CLIENT: ALWAYSLEGIT, INC. (CONTRACT, REMOTE)

Full Stack Developer

United States
Apr 2026 - Present

- Built three systems behind the startup's upcoming exchange for graded trading cards: the market-data platform that values each card and grade it tracks, the admin console its operators run the data from, and the phone web app for its buyers to use when sourcing stock.
- Engineered the Supabase backend across two client repositories: Supabase Auth sessions bridged from a Kinde passcode login, role-based row-level security written into the migrations, SECURITY DEFINER functions, and over 50 Deno edge functions, including functions that call OpenAI and Gemini with JSON schemas and write the results across several tables.
- Built a market-data pipeline for graded cards that pulls sales, prices and population data from a card-pricing API, a population database, and a grading company's site read through one Firecrawl browser session per run, keeping each source's results when another source fails.
- Built a scheduler that gives every card a refresh cadence for each sales source from its recent sales per day, so busy cards refresh more often and quiet cards are still checked, and a failing source never pushes back another source's schedule; a Postgres cron job drains the queue every minute.
- Contained vendor spend on a data pipeline over Firecrawl, SerpApi, OpenAI and Gemini: responses are stored once and re-read, and a Postgres claim-first queue stops a crash or a retry from double-spending.
- Built a server-side market calculator (39 fields of prices, sales velocity, signals and freshness) that matches the spec's test records and the previously delivered CSV value for value, and a 12-endpoint market-data API that resolves six kinds of card identifier, plus a Postgres login with SELECT on the five published views.
- Shipped two export APIs for the client's analytics developers with incremental sync (updated_since catches every material change, not only new rows) and an explicit match status on each row (paused, uncertain, no sales, verified), so their system can upsert changes without a full reload.
- Built the admin console in React and TypeScript (Vite, TanStack Router, Zustand, Storybook): search across nine fields, status filters, a column picker, a per-row resume for paused rows, 99-column CSV exports, and operator guides on a Docusaurus site.
- Added bulk data tools to the console: single-card runs on demand with a result link that survives closing the tab, and pages that turn pasted lists or CSV files into catalog titles, AI card matches and population lookups, keeping raw API responses for download.
- Kept five identifier systems in step across both sync screens, their exports and the APIs, and checked a vendor's AI card matcher with fixed string rules instead of trusting it: rows proven to be a different card are paused, stop feeding prices downstream, and are reported with the reason.
- Built a slab-photo redactor that makes two masked copies per image (barcode and cert number hidden; grading lines also hidden in the second), never touches the original, and holds a copy for review unless a barcode decoder and digit recognition can no longer read it; no barcode stayed readable across a 377-photo batch.
- Built four generations of the buyers' sourcing app as an installable phone web app (React, Vite, PWA): it identifies a graded card from its cert number and a sealed box from a photo that OpenAI ranks against the product catalog, identifies the cards on a photographed binder page with Gemini, prices each item from market data, and records the buyer's offer.
- Made the early generations' barcode reader hold up on hard photos: the native BarcodeDetector first, ZBar in WebAssembly inside a Web Worker as the fallback, several crops and image passes before it gives up, and an IndexedDB scan queue that survives a refresh; binder photo uploads resume after a dropped connection.

Key Achievements

- Sustained 24,000+ pipeline runs at a 99.98% success rate in ten days.
- Earned a 5.0 client review on Upwork.
- Covered the platform with over 3,000 automated Deno tests.

UPWORK CLIENT: PLAYER ONE LABS (CONTRACT, REMOTE)

France

AI and Full-Stack Developer

Aug 2026 - Present

- Extended the operator side of an automated website content system: a runner panel with job labels, a usage-limit pause, and three SEO checks in the nine-check quality gate that holds anything back until it passes.
- Set up agents that execute from a written specification, and nothing publishes until it passes the quality gate.
- Made approved content publish without a redeploy, with a person releasing each piece.

FREELANCE CLIENT: HYEPAIR LLC (CONTRACT, REMOTE)

United States

Full Stack Developer, mobile dating app

Jul 2026 - Present

- Developed the NestJS API, the Expo and React Native app (TanStack Query, Zustand, Zod), the React admin console (TanStack Table), and the landing page.
- Built sign-up that collects each member's dating intentions, a profile editor for lifestyle answers and prompts, a discovery feed filtered by gender and community, and full photo galleries that open only after two members match.
- Built the swipe deck's live feedback: the verdict appears while the finger is still moving, the card behind tints with the drag, and the last profile in the deck still shows its verdict.
- Implemented the client's plan tiers: limits per tier on likes, waves, and new chats, with the chat quota counted over a rolling 30 days.
- Built Google Play Billing subscriptions in which the server confirms each purchase with Google before granting the tier and keeps renewals in step.
- Built admin controls for identity verification and roles: a verification queue that held new profiles before they went live, status changes reserved for super admins and moderators, and photo removal with an audit record.
- Took the app through Google Play review to production in 178 countries, with Data safety answers, an account deletion page, a child safety standards page, and a daily job that purges deleted accounts after 30 days.
- Advised the client on pricing, a free trial, and a payment flow modelled on comparable apps.

FREELANCE CLIENT: PARTYDEPOT LLC (CONTRACT, REMOTE)

United States

Automation Developer, events and party-supplies store on Shopify

Apr 2026

- Built an n8n webhook that suggests products for an event from its name, description and headcount: a daily Shopify sync keeps products, stock and prices current, retrieval uses embeddings, vector search and reranking, and a guardrail corrects any suggestion that is not in stock, at about USD 0.001 to 0.003 per request.
- Stored custom-print uploads in a small Go service, because n8n's host restricted file storage; it keeps the files on its own disk and records each one in PostgreSQL; an n8n schedule at midnight calls the service to delete files older than 30 days, and n8n emails the customer once a file is saved.

FREELANCE CLIENT: PT BANYUKE INTAN PERMAI (MONTHLY RETAINER)

West Kalimantan, Indonesia

Full Stack Developer, ERP and document automation for a property developer

Dec 2025 - Present

- Built the company's ERP as its only developer (Next.js, PostgreSQL, Drizzle, Vercel), with modules for buyers' mortgage files, housing projects and blocks, credit checks, sales agents and leads, inventory, accounting, and HR.
- Built document generation for mortgage submissions to four banks (BTN, Bank Kalbar, BRI, and Mandiri): the ERP renders 13 of the letters in a Bank Kalbar submission itself, and n8n workflows fill Google Docs templates from ERP data and save the results to Google Drive in a folder for each buyer and bank.
- Built letter numbering on one running counter per year: n8n assigns all four letter numbers when the house order letter is issued, and regenerating a document keeps them, so no number is burned from the series.
- Connected four new BRI templates to the generator, mapping about 40 new placeholders to ERP data and adding the missing fields to the ERP's forms; all four generated for three sample buyers with no raw placeholders.
- Built a lead form with its own link for each sales agent, so each lead is recorded under that agent, with an indicative subsidized-housing eligibility check that gives the same result on the web form and in the Telegram bot.

FREELANCE CLIENT: COUPLESWAP LLC (CONTRACT, REMOTE)

United States

Full Stack Developer, dating platform (web and mobile)

Nov 2025 - Present

- Built the product as its main developer from the first commit (96% of its 2,000+ commits): the React web app (Nov 2025), the NestJS and Prisma API (Dec 2025), the admin console (Jan 2026), and the Expo and React Native app (Mar 2026).
- Built the relevance score that, after boosts and plan tiers, orders each member's discovery feed from what members enter: relationship preferences, distance within their search radius, age gap, and shared interests, with feed locations shifted 3 to 5 miles so exact positions stay private.
- Built realtime chat over WebSocket with conversation search and last seen, Like and Pass discovery with mutual matches, meetups pinned on a map, photo upload, report and block, and push notifications.

- Replaced a third-party payment gateway that never went live with a direct Authorize.Net integration (Accept.js and customer profiles) with an idempotent first charge, next to Google Play Billing, with both rails writing the same subscription and boost records.
- Built the admin console for moderation and member data, including email campaigns limited to a safe HTML subset, with a preview and a required test send before each blast.
- Backed the API with about 1,480 Jest test files and a GitHub Actions check (type checks, unit and end-to-end tests with coverage thresholds, secret scan, dependency audit), and ran dev, UAT, and production on Coolify with Sentry reporting server errors and slow requests.
- Released the app on Google Play, then took iOS through Apple App Review: after a Guideline 2.1 rejection, added in-app account deletion for members still in verification and sent a recording of the full flow; the iOS app went live on the App Store in October 2026.

SOFTWARE AGENCY (FREELANCE, REMOTE)

Full Stack Developer

United States

Dec 2023 - Jan 2025

- Took over a half-built crypto staking platform for Polkadot (DOT) nomination pools and finished its Next.js frontend, including a React Flow view that traces funds moving between wallets.
- Built the DOT staking flow end to end with polkadot.js and the Talisman wallet: connecting the wallet, proving ownership with a signed nonce before linking it to the logged-in account, signing stake transactions in the wallet, and checking each transaction on chain in the FastAPI backend before recording it.
- Encrypted wallet payloads between the Next.js server and the FastAPI backend with a key held only on the server, on top of HTTPS.
- Rebuilt the platform's OpenAI chatbot with retrieval-augmented generation, choosing context for each question and writing prompts that produce relevant PostgreSQL queries; correct answers on a self-written 20-question test set rose from 8 to 17, and answers render as text, node graphs, charts, or tables.
- Built the Next.js 14 frontend of an AI character chat platform where users create characters that answer through an LLM, including the character-creation form whose fields feed each character's prompt.
- Integrated MetaMask so users pay in crypto for chat credits, and recorded the revenue split between each character's owner and the platform in the app.
- Built the backend ranking query that lists premium characters ahead of regular ones, so the frontend renders the results exactly as returned.
- Mid-project, helped finish the Django backend: reviewed the ERD and rebuilt Google sign-in to work for the separate frontend through the API, instead of Django's server-rendered OAuth flow.

PT HACKTIVATE TEKNOLOGI INDONESIA, HACKTIV8 (FULL TIME)

Data Science Instructor

Jakarta, Indonesia

Aug 2023 - Mar 2024

- Mentored 40+ beginner programmers in Python, PostgreSQL and object-oriented programming.
- Instructed students in data analytics with Pandas, Seaborn and Tableau.

Full Stack JavaScript Instructor

Nov 2022 - Aug 2023

- Taught 100+ beginner programmers JavaScript, Express, Vue, Docker, object-oriented programming and PostgreSQL.
- Coached students through React and React Native builds, with consistently strong student feedback.

SECOND CAR MARKETPLACE (FREELANCE, REMOTE)

Full Stack Developer

Jakarta, Indonesia

Jan 2023 - Jun 2023

- Designed the UI and UX in Figma and the system architecture of a used-car marketplace where both dealers and individual sellers sell cars to buyers, then built its React Native app and backend for a private client.
- Split storage by workload: PostgreSQL for the main marketplace data, MongoDB for chat messages and notifications, and Contabo Object Storage for uploaded files.
- Integrated Midtrans payments by virtual account and QRIS, alongside real-time chat and push notifications.
- Optimised the app for low-specification phones and adapted it to Google Play's policy requirements before its launch on the store.

MONSTYR PTE. LTD. (FULL TIME, REMOTE)

Junior Software Engineer

Singapore

Jun 2021 - Oct 2022

- Cut long-running PostgreSQL queries to under one second, and put some race-prone endpoints behind queue-based processing to prevent race conditions.
- Fixed a React Native map screen that crashed under too many pins, by rendering per zoom level, loading only the pins in the visible bounds plus a small margin, and fetching as the map moved; also kept infinite scroll light on memory with a virtualized list.
- Built search suggestions and recommendations on a Neo4j graph, and web scrapers in Python with BeautifulSoup.
- Designed the listing logic that decided how paid placement sat against organic results.
- Reviewed teammates' code before the CTO's review and the merge into the dev branch, and onboarded four new team members.
- Ran the PERN stack in production with Redux, React Native bridged to native Java, Redis, Amazon S3 and Sentry, and managed the AWS EC2 instances, Auto Scaling and load balancer.

- Led four concurrent projects to a 90% on-time delivery rate.
- Reported progress and directed tasks across the four projects.
- Coded about 20% of the development hands-on, building full-stack features on CodeIgniter.

PROJECTS

- **DealLens** (github.com/tryea/deallens): multi-tenant due-diligence SaaS with cited RAG over a 1,019-document corpus: pgvector (HNSW) and Postgres full-text search fused by Reciprocal Rank Fusion, scoped by row-level security.
- **Crate** (github.com/tryea/crate-webapp): inventory management on Next.js 16, Drizzle and PostgreSQL with row-level security, an append-only stock ledger, advisory locks so two parallel stock-outs cannot push stock below zero, an audit log written in the same transaction, an English and Indonesian interface (next-intl) with a key-parity check in CI, and 644 Jest and 35 Playwright tests passing in CI (October 2026).
- **Lead intake in n8n** (Upwork portfolio): one row per lead even when two forms land at once; duplicates are caught in 12 to 17 ms, against 1.4 to 4.6 seconds for a new lead that calls the model, with a rule-based fallback.

EDUCATION

PRESIDENT UNIVERSITY, Cikarang, Indonesia: Bachelor of Science (B.Sc), Information System

2015 - 2019

CERTIFICATION

- HackerRank SQL (Advanced), Problem Solving (Basic) and JavaScript (Intermediate); Triplebyte Back-End Web Development; freeCodeCamp JavaScript Algorithms and Data Structures

SKILLS

- **Frontend** (React, Next.js App Router, Vue, TypeScript, JavaScript, Tailwind CSS, React Native, Expo, PWA, Vite, TanStack Query, TanStack Router, TanStack Table, Redux, Zustand, Zod, Storybook)
- **Backend and APIs** (Node.js, NestJS, Express, REST APIs, FastAPI, WebSocket, webhooks, Deno edge functions, payments (Stripe, Authorize.Net, Google Play Billing, Midtrans))
- **Databases** (PostgreSQL, SQL, row-level security, Supabase, pgvector, Drizzle, Prisma, MongoDB, Redis, Neo4j)
- **Cloud, delivery and testing** (AWS (EC2, S3), Linux (Ubuntu servers), CI/CD (GitHub Actions), Docker, Git (GitHub, GitLab), Jira, Sentry, Jest, Vitest, Playwright)
- **AI and LLM** (OpenAI and Gemini APIs, prompt engineering, structured outputs, RAG (embeddings, vector search, reranking), human review of AI output, AI-assisted development (Claude Code), n8n, Make)
- **System design** (multi-tenant isolation, queue-based race prevention, idempotent webhooks, performance tuning)
- **Soft skills** (code review, written specifications, client communication, mentoring, project management)
- **Also familiar** (Go (basic), GraphQL (used while teaching), MySQL and PHP with CodeIgniter (early career))
- **Languages** (English (fluent), Indonesian (native))