

Aryan Anand

+91 82981 58831 | heyaryan.xyz | aryananand.syn@gmail.com | linkedin.com/in/aryananand18 | github.com/aryann-anand

EXPERIENCE

Software Development Engineer - R&D

June 2025 - Present

Ninestars Information Technologies, Bangalore

- Architected and developed **AOTM Vault**, a multi-tenant secrets management platform supporting 50+ cryptographic algorithms (including post-quantum), multi-cloud storage abstraction (AWS, Azure, Google Cloud), automated Celery-based key rotation policies, real-time WebSocket file streaming, and immutable audit logging across vaults, resources, subscriptions, and billing, reducing enterprise key rotation overhead by 80%.
- Built an enterprise RAG system with hybrid vector search (dense+sparse+RRF) over Qdrant, integrated with self-hosted LLMs and Jina Embeddings, enabling sub-800ms semantic search across 100K+ files; and designed a private blockchain ledger with Proof-of-History consensus, immutable transaction chains, Merkle-root block validation, and on-chain document signing/verification for tamper-proof audit trails
- Engineered a distributed database migration engine with a long-running daemon process architecture (Unix socket IPC, background thread pools) for agent-side parallel batch extraction and direct-to-S3 upload of 60M+ row databases, with live progress reporting, dynamic thread resizing via backend config sync, and graceful stop - mirroring server-side Celery task behavior entirely on the client side (on-prem server, machines, VPS).
- Built a multi-tenant rate-limiting + usage-based billing layer with atomic Redis Lua admission (token bucket + per-minute/hour/day fixed-window counters), hierarchical user/project limits, two-phase credit reservation backed by Postgres-mirrored reserved columns + an orphan-reservation sweeper, Celery signal-driven cache invalidation, anniversary-based monthly refills, and partial-processing fallback that auto-trims document batches to the user's affordable cap sub-5ms hot path with zero Postgres queries on cache hit.
- Architected the backend of **AOTM FDI**, a multi-tenant credential-fraud detection platform for a global credential-evaluation organisation (11 Django apps, 119 REST routes, 43-table PostgreSQL schema): a 5-stage OCR-to-clustering analysis pipeline over RabbitMQ quorum queues, served by five queue-specialised Celery pools behind a uvicorn ASGI app.
- Made the pipeline safe under at-least-once delivery: a versioned request/reply envelope, per-template fan-out joined by a row-locked fan-in barrier that releases exactly once, idempotent replace-scoped writes, and presigned per-object MinIO URLs so workers never hold storage credentials.
- Built the rule engine: a depth-bounded boolean DSL (20 typed fields × 14 operators) validated at write time, three-valued evaluation that marks unverifiable inputs *unevaluable* instead of passing them, ~7 queries per docket regardless of rule count, and Gemini-rephrased reasons verified mechanically before display.
- Designed the verdict model: weighted rule failures and visual-anomaly severity scored 0–100, blended and banded on tenant-configurable thresholds, and re-derived transactionally as late evidence lands, so verdicts no longer depend on worker timing; separated lifecycle (`docket_status`) from judgement (`final_verdict`) via backfill migrations.
- Modelled versioned master-template *generations* per institution, credential and award year, matched on perceptual hashes (pHash/dHash) and embeddings of seals, logos and signatures; ranked candidate templates by per-document comparison verdicts and excluded byte-identical copies by SHA-256.
- Gated evaluation on a completeness check (required documents, fields and a matching master), and built bulk reprocessing as beat-fed, cancellable jobs with dry runs and row locks, where a restart from any stage first proves every earlier stage finished on each document's last run.
- Designed list APIs on the HTTP `QUERY` method with multi-value filters and computed-score ranges and sorts in bulk queries, plus WebSocket progress streams, optimistic-locked collaborative editing over Django Channels, and PDF/Excel report generation.
- Enforced fail-closed multi-tenancy across 33 tenant-scoped tables with ContextVar-scoped managers that raise rather than leak cross-tenant rows, plus RBAC rank guards, rotating JWT sessions and chunked AES-256-GCM export encryption; shipped 170+ versioned builds in seven weeks, backed by 70 scripted test suites.
- Fanned each sensor uplink out to an external ML worker pool across 10 RabbitMQ quorum queues (forecast, anomaly, threshold, LLM insight, photogrammetry, CFD), chained on a transactional reply-side fan-in barrier - `SELECT FOR UPDATE` + on-commit publishing + UUIDv5-derived keys for redelivery idempotency - and integrated CFD and photogrammetry as asynchronous job types with design snapshots frozen under row-level locks, worker-side S3 uploads registered by deterministic file IDs, floor-geometry-to-row placement sync enabling 3D proximity ranking of sensors to artifacts, and every stored asset link re-signed per request so URLs survive storage-endpoint migration without a data migration; executed 10,700+ inference jobs across 2,500+ completed multi-stage runs, generating 3,300 conservation risks and 3,900 LLM insights.

- Built the telemetry layer on TimescaleDB hypertables (composite time+UUID keys) with a GraphQL API unifying historical/forecast/anomaly reads across sensor, zone, floor and building scopes - sustaining 3.6M rows over 119 days of continuous ingest and serving 50K-row, 30-day analytical windows in ~250 ms - plus a fleet-health engine deriving battery depletion (regr_slope 30-day projection), persistent weak links (p10 RSSI, not min) and per-series dropout gaps in raw SQL, surfacing failing sensors the static config columns reported as 97% healthy.
- Owned the delivery stack: 5-process Docker Compose topology (ASGI + beat + three queue-specialised workers), Jenkins CI, a tmux dev orchestrator, and an idempotent setup script that health-checks Postgres/TimescaleDB/Redis/RabbitMQ/S3 and provisions vhosts and quorum queues from the application's own queue definitions, so broker topology cannot drift from code.

Machine Learning Intern

Sept 2023 - Dec 2023

TIH IoT, IIT Bombay, Mumbai

- Built a U2-Net pipeline for automated image masking and applied CNNs on NDVI and hyperspectral imagery for vegetation segmentation (11% gain), preprocessing, augmentation, and hyperparameter tuning for deployment.

EDUCATION

Vellore Institute of Technology

CGPA: 8.24/10

Bachelor of Technology in Computer Science and Engineering

2021 - 2025

TECHNICAL SKILLS

Languages: Python, JavaScript, TypeScript, Java, C, C++, SQL, PHP, Bash/Shell, HTML, CSS

Frameworks: Django, Django REST Framework, Django Channels, FastAPI, Flask, Node.js, Express.js, Next.js, React, GraphQL

Distributed Systems & Messaging: Celery, RabbitMQ, Apache Kafka, Redis Streams, Pub/Sub

DevOps & Database: Git, Docker, Kubernetes, Jenkins, GitHub Actions, GitLab CI/CD, Terraform, Ansible, MySQL, PostgreSQL, MongoDB, Redis, TimescaleDB, Qdrant

Cloud: AWS (S3, EC2, Lambda), Google Cloud, Azure, Google Cloud Storage, Azure Blob Storage, MinIO, RESTful APIs, Celery, WebSocket, Qdrant, Jina Embeddings

Libraries: OpenCV, LangChain, Apache Ray, Scikit-learn, NLTK, Numpy, Pandas, Pillow, React

AI/ML: PyTorch, TensorFlow, HuggingFace Transformers, Llama, RAG

Other Technologies: WebSockets, ASGI, Uvicorn, Git, GitHub, GitLab, Linux, Ubuntu, VMware

PROJECTS

MedAI Vision | *Python, PyTorch, TensorFlow, Llama, React.js, Node.js, MongoDB, Git* 2025

- Built a super-resolution GAN (TensorFlow) for MRI/CT clarity achieving 92% accuracy on 54,000 samples, and a CycleGAN (PyTorch) for bidirectional MRI-to-CT translation; designed a hybrid Vision Transformer-CNN for multi-label disease classification across brain, breast, and lung scans with 88.4% accuracy.
- Fine-tuned Llama 3.2 11B on medical literature to deliver explainable AI insights with diagnosis, severity levels, and treatment recommendations, achieving 74% user retention.

Gesture Echo | *Python, OpenCV, TensorFlow, Google Cloud, Git* 2024

- Developed a real-time hand sign language-to-speech system using ResNet-18 and OpenCV achieving 97% accuracy and 200ms latency on low-end devices, with Google TTS API integration and optimized scalability for high user accessibility.

CERTIFICATIONS AND ACHIEVEMENTS

Neural Networks and Deep Learning, Generative AI with LLMs, Supervised Machine Learning, React Basics, Python, JAVA, PHP and MySQL, AWS Cloud Practitioner, Introduction to Front-End Development, SIH 2022