~/Akash Damle

// Backend Engineer

Badlapur, India · Open to remote roles

akashdamle07@gmail.com · +91 98333 58619

akashdamle.in · github.com/code-kasha · linkedin.com/in/akashdamle

$ summary

Backend engineer with 8+ years of total experience, building and operating systems in Django and Node/TypeScript. On client work I own the whole backend, sometimes as the only engineer and sometimes alongside a frontend developer, which means I have owned schema design, API surface, deployment and support rather than one slice of it.

Recent work has focused on making that ownership repeatable: typed API contracts generated from a single schema definition, integration test suites that run against a real database in CI, and infrastructure I can rebuild from scratch.

$ experience

Freelance Software Engineer — Independent · Remote

Mar 2021 – Present · 5 yr 7 mo

Django CRM and internal-tooling work for small organisations (schools, clinics and SMBs), delivered end to end and supported on an ongoing basis.

> Delivered 27+ client systems, each covering some combination of employee management, payroll calculation, attendance tracking, performance review and reporting.
> Designed the relational schema and REST API for each deployment, and built the admin and reporting surfaces on top.
> Provisioned and maintained self-managed Linux servers running PostgreSQL, including backups, TLS and upgrades. Hosting is part of the engagement rather than the client's problem.
> Worked directly with non-technical stakeholders to scope requirements, agree priorities and run acceptance.
> Also delivered custom websites with Django and Django CMS for clients whose needs stopped short of a full system.

stack: Django · Python · PostgreSQL · Django CMS · Linux · Nginx · Gunicorn

Junior Software Developer — Matalli Infotech · Dombivli

Aug 2018 – Feb 2021 · 2 yr 7 mo

First engineering role, on a small team maintaining internal web applications.

> Built and maintained internal applications on Django 2.2 LTS.
> Wrote unit and database-level tests to keep regressions out of releases.
> Implemented and worked with CI/CD pipelines to shorten the release cycle.
> Managed deployments across Linux servers, AWS and Heroku.
> Delivered production patches and bug fixes with minimal downtime.
> Learned project structure, code organisation and review practice in a team setting, and used Git throughout for collaboration.

stack: Django · Python · Git · Linux · AWS · Heroku · CI/CD

$ selected projects

Lead Management Platform

2026

github.com/code-kasha/lead-platform · Live demo · Write-up

A Django REST API that moves sales leads through a validated pipeline and records every change on a timeline, with a React and TypeScript app whose types come from the API's schema.

> The status pipeline (New, Contacted, Qualified, Proposal, Won or Lost) is enforced in one service function; a blocked change returns 400 with the reason.
> Every create, edit, status change, assignment and note is written to the timeline in the same transaction as the change.
> Admins manage every lead; members see and work on only the leads they created or are assigned to, and anything else returns 404.
> Access tokens last 15 minutes; refresh tokens rotate on every use and are blacklisted after use or on logout. The browser shares one refresh across every request waiting on it, because a rotated token works only once.
> 99 backend and 74 frontend tests. CI also runs flake8, a missing-migrations check, the stale-schema and stale-types check, and a Docker build with a smoke test of the running container.

stack: Python · Django · Django REST Framework · PostgreSQL · OpenAPI · React · TypeScript · TanStack Query · Docker · GitHub Actions · pytest · Vitest

Webhook Delivery

2026

github.com/code-kasha/webhook-delivery · Live demo · Write-up

A Node.js and TypeScript service for signed webhooks, durable retries and delivery history, backed by PostgreSQL.

> Transactional fan-out, fenced worker leases and bounded retries retain events and attempt history across failures; delivery is at least once, with no ordering guarantee.
> 85 tests run against PostgreSQL in CI on Node 22 and 24, including concurrent claims, worker crashes, database outages, HTTPS verification and SSRF boundaries.
> Timestamped HMAC signatures, overlapping secret rotation and DNS/IP validation with address pinning protect the receiver and outbound request boundary.
> A controlled Render restart retained credentials, signing secrets, queued deliveries and prior attempt history; the retained webhook then received HTTP 204.

stack: Node.js · TypeScript · Fastify · PostgreSQL · Zod · OpenAPI · Docker · GitHub Actions · Vitest

Operations API

2026

github.com/code-kasha/operations-api · Live demo · Write-up

A Django REST API for the staff side of a small organisation: staff and roles, shifts and attendance, leave, office timesheets and overtime, working-day payroll and reports, built from scratch with fictional data.

> Four business roles (operations admin, HR, manager and employee) follow a documented permission matrix. Managers see their department and employees see themselves; anything else returns 404, and nobody reviews their own leave, timesheet or overtime.
> Payroll pro-rates base pay by working days for joiners, leavers, absences and unpaid leave, rounds half-up to the paisa, pays each approved overtime request exactly once and locks finished pay runs. It calculates gross pay only and claims no statutory compliance.
> Monthly attendance summary, payroll register and headcount reports reuse each module's role-scoped querysets, so a report never shows what its reader could not open.
> One command loads a fictional office with a month of history and a locked pay run, all or nothing. The public demo resets to it on every start, and the reset refuses to run anywhere else.
> 346 tests, also run once against PostgreSQL 17. CI checks lint, formatting, missing migrations and a stale OpenAPI schema, smoke-tests the Docker image, publishes releases from tags and redeploys the demo after each push.

stack: Python · Django · Django REST Framework · PostgreSQL · OpenAPI · Docker · GitHub Actions · pytest

bharat-post-dir 2023–2026

github.com/code-kasha/bharat-post-dir · Live demo · Write-up

India's postal directory as a lookup page, a JSON API and a one-file download: 155,599 offices, with every page stating where the data came from.

> The lookup page costs one HTTP request and three database queries: inline CSS, no JavaScript, fonts or images. It works by keyboard and screen reader, in light and dark, and on phones.

> Imports are all-or-nothing. A short download is refused, identical rows merge, and offices listed more than once are kept and reported, never silently dropped.

> The whole-directory download is versioned by its SHA256 and sent with an ETag, so an unchanged directory is never downloaded twice.

> Anyone can load their own dataset. On a local clone an upload is temporary and private to one browser; in contributor mode it replaces the database and explains how to share it back. The hosted demo turns this off.

> CI deploys each push to main and waits until the demo's health check reports the new commit. 119 tests run against synthetic fixtures and never reach the network.

stack: Python · Django · Django REST Framework · SQLite · OpenAPI · Docker · GitHub Actions · pytest

CRM [In research and planning]

A CRM that arrives already set up for how your business works — sales, agency, real estate or field services — instead of making you configure it first.

$ skills

Backend	Django · DRF · Node.js · Express · REST	Languages	Python · TypeScript · JavaScript · SQL
Data	PostgreSQL · MongoDB · Redis · Query tuning	Platform	Docker · CI/CD · Linux · AWS · Vercel
Quality	Vitest · Pytest · Zod · OpenAPI	Frontend	React · Next.js · Tailwind · Redux
AI tools	Claude Code · OpenAI Codex · ChatGPT		

$ education

B.Sc, Computer Science P V G's College of Science 2014 – 2018

$ training and certifications

Full Stack Development Internshala Oct 2025 – Apr 2026

Microservices with Node.js and React Udemy Apr 2024 – Sep 2025