Mohammad Foroutan

Full-stack developer (4+ yrs) specializing in Python/Django/FastAPI backends and Next.js frontends — production systems spanning payments, media processing, and multi-role platforms, including a Django platform serving 20,000+ students.

LinkedIn
GitHub
foroutan.dev
+989394166207
foroutan.dev@gmail.com

EXPERIENCE

CS50xIran, Tehran — *Full Stack Developer – Tech Lead*

NOV. 2022 - JAN. 2026

- Owned course and user-management features on a Django platform that has been in continuous production for 4+ years, currently serving 20,000+ students
- Integrated external payment, registration, and video-hosting providers (Evand, IranHLS, Snapppay, Arvancloud VOD) to move enrollment from a manual process to a self-serve flow
- Set up and maintained the production deployment (Nginx, Gunicorn, Docker) and a multi-service Docker Compose stack (MySQL, Redis, Celery) supporting the platform's backend
- Refactored a legacy template-based codebase into multiple modular Django apps, models, and views to support continued feature growth without a rewrite
- Started a Next.js frontend to begin migrating the platform off server-rendered templates toward an API-based architecture
- Wrote internal Python scripts for bulk admin operations (e.g., batch enrollment/user updates), exposed through the admin panel for non-technical staff to use directly
- As Tech Lead: led a small team building course and campaign landing pages, onboarded incoming developers, and represented engineering in cross-functional planning

Confidential Client, Remote — *Full Stack Developer, ICT Enterprise Contractor Panel*

SEP. 2025 - FEB. 2026

- *Built a contractor-facing panel for a large ICT enterprise's tender management process, using a feature-driven Next.js architecture*
- *Worked directly against an existing Oracle database of 40+ tables and views, writing and calling 10+ stored procedures and packages via Knex and node-oracledb*
- *Used GitHub Copilot to speed up boilerplate code and documentation generation while working against an unfamiliar, large existing schema*

LANGUAGES

Python, SQL, JavaScript, TypeScript,

BACKEND

Django, DRF, FastAPI, Celery, Redis, PostgreSQL, Oracle DB

FRONTEND & INFRA

React, Next.js, Tailwind, Docker, Nginx, Git, Linux

PROJECTS

Notification and Chat Service — *Personal Project* — GitHub

- Built a FastAPI service simulating the real-time messaging layer of a VOD platform — livestream chat, push notifications, and support sessions — each with its own persisted model, service layer, and routes
- Implemented per-room WebSocket chat with sequenced message history and moderation actions, tracked through a custom in-memory connection manager for room/channel/session-scoped broadcast and direct delivery
- Built a mock event-ingress endpoint simulating upstream microservices publishing events, fanned out over per-user WebSocket connections with per-channel delivery tracking persisted to Postgres
- Designed a deterministic identity layer mapping free-text demo identifiers to stable integer/UUID ids, so reconnecting sessions resolve consistently without a real user service backing the sandbox
- Kept routers thin by pushing all DB access and business rules (get-or-create semantics, message sequencing, delivery status, moderation) into a service layer, with routers only wiring HTTP/WebSocket endpoints to services
- Managed schema exclusively through Alembic migrations, deliberately excluding auto-create_all from any code path to keep schema changes reviewable and reproducible

Video Processor Service — *Personal Project* — GitHub

- Built an async video-processing pipeline (FastAPI, Celery, Redis, PostgreSQL, FFmpeg) so heavy transcoding work runs off the request path instead of blocking API responses
- Built a direct-to-object-storage upload flow using presigned URLs, removing the API server as a bottleneck for large file uploads
- Modeled job lifecycle state (PENDING → QUEUED →PROCESSING → FINISHED/FAILED) in SQLModel/PostgreSQL to make stuck or failed jobs debuggable and recoverable, not just visible
- Built an FFmpeg transcoding workflow producing adaptive-bitrate outputs (DASH/HLS) from a single ingested source file
- Added ffprobe-based validation ahead of the processing queue to reject malformed uploads before they consume worker/compute time
- Wrote layered tests (API, worker, integration) with pytest, including tests that exercise real FFmpeg execution rather than mocking it entirely, to catch encoding regressions mocks would miss
- Containerized all service components with Docker Compose for reproducible local dev and deployment

Kabk Academy — *B.S. Thesis* — GitHub

- Built a commercial learning platform end-to-end (Django/DRF backend, Next.js/TypeScript frontend with Shadcn) supporting both subscription and per-course purchases, spanning 8 apps and 24 models across two core domains
- Built course enrollment with per-user progress tracking, a ticketing system with multiple file attachments per ticket, and async scheduled jobs (Celery, Redis) for billing and notifications
- Integrated NextAuth with the Django backend to support multiple auth methods (OTP, OAuth) from a single frontend

- Designed multi-role access (Teachers, Organizers, Authors, Admins, Users) to support a real content-marketplace workflow rather than a single-author blog
- Built with SEO in mind (metadata, routing structure) since discovery, not just conversion, was a stated goal of the thesis
- Deployed with Nginx and Docker Compose against PostgreSQL

EDUCATION

Shamsipour Technical and Vocational College, Tehran
B.S. in Information Technology

JAN. 2024 - JAN. 2026

Associate's in Software Engineering

JAN. 2021 - JAN. 2024