Nehmat Farooq

nehmat429@gmail.com | linkedin.com/in/nehmat-farooq | github.com/neh2332 | https://personal-portfolio-5v8.pages.dev/

Canadian Citizen

Software Engineer graduating June 2026. I build complex, high-performance systems. From DAG-based video rendering pipelines to containerized code execution environments. Looking to join a team where I can handle difficult technical problems from day one.

Education

Ontario Tech University — Oshawa, ON
Bachelor of Engineering in Software Engineering (Hons.) — *Sept. 2021 – June 2026*

Technical Skills

Languages: TypeScript, JavaScript, Python, Java, C++, SQL (PostgreSQL, MySQL, H2), Dart, HTML/CSS
Frameworks/Runtimes: React, Next.js, Spring Boot, Redux, Vite, Node.js, Bun, Hono, Turbo, Zustand, ReactFlow, Recharts
Cloud & Infrastructure: AWS (LocalStack, CloudFormation, S3, DynamoDB, SQS, IAM), Google Cloud Storage (GCS), CloudFlare Pages, Nginx, Redis, Linux, Server-Sent Events (SSE)
Developer Tools: Git, AWS CLI, JIRA, Confluence, Postman, Jest, Zod, RESTful APIs, CI/CD, SDLC, Framer Motion, Shadcn/ui
Systems: Linux Namespaces, cgroups, Docker, FFmpeg, DAG execution, Topological Sort

Experience

Mobile Software Engineer (Contract) — October 2025 – January 2026
My Peng Life — *Toronto, ON*

- Built and shipped a Gemini-powered AI app from zero to production as the sole engineer, handling everything from the React frontend to native mobile deployment.
- Architected a React/TypeScript frontend from scratch, optimizing for real-time AI state streaming without latency drops
- Deployed native mobile distributions by bridging web technologies with native device APIs via Capacitor.
- Designed the NoSQL database schema in Firestore to handle real-time chat persistence and complex travel itineraries with minimal read latency.

Software Engineer Intern — August 2024 – March 2025
Savi Finance — *Toronto, ON*

- Drove total users from 110 to 200+ and scaled paid subscribers 7x (2 to 15+) by architecting and shipping a full-stack financial goals feature and other features during my time.
- Redesigned the expenses and income dashboard end-to-end, establishing a new UI architecture the rest of the engineering team adopted.
- Eliminated manual data entry for users by engineering an automated receipt-scanning pipeline that parses and categorizes transactions with zero human intervention.
- Owned tickets across the full stack, React frontend, Node.js backend, and PostgreSQL through the complete SDLC from technical design documents to production.

Projects

Daily Finance: Multi-Source Financial Tracking Platform | *Java, Spring Boot, React, Python, AWS LocalStack, Docker*

- Architected a full-stack financial dashboard with a Spring Boot REST API and React/Redux SPA, using Recharts to render asset allocation and cost-basis visualizations.
- Engineered a real-time market data ingestion pipeline with Python AWS Lambda and yfinance to fetch and process live stock and crypto prices.
- Set up a zero-cost local development environment with Docker Compose and AWS LocalStack to emulate S3, DynamoDB, and SQS.
- Automated production-ready AWS resources using CloudFormation to deploy free-tier compliant infrastructure, strictly mapping IAM roles to required execution limits.
- **GitHub**

StoryWeaver: Node-Based AI Video Generation Platform | *React, TypeScript, Bun, Hono, PostgreSQL, GCS, FFmpeg*

- Built a node-based video editing interface supporting non-linear workflows, structured within a Bun/Turbo monorepo to manage complex rendering dependencies.
- Architected a Directed Acyclic Graph (DAG) execution engine using Topological Sort to resolve video scene dependencies and enforce generation order.
- Developed a real-time progress system using SSE and Zustand to keep AI generation feedback latency under 100ms.
- Automated a backend media worker using FFmpeg to handle video stitching, enabling seamless concatenation of AI-generated clips into high-fidelity outputs.
- Reduced storage query overhead by streaming graph state directly to GCS using JSONB, avoiding repeated database round-trips during generation.
- **GitHub**