Nandan Ramesh

☎ +1 825-975-3030 | ✉ nandan@ualberta.ca | 📍 Edmonton, Alberta, Canada
linkedin.com/in/nandanramesh | github.com/Nandan-18

Education

University of Alberta | **Bachelor of Science, Specialization in Computing Science** **May 2026 (Expected)**
Dean's Honor Roll (**2x**), Undergraduate Academic Scholarship (**4x**), Undergraduate Leadership Award (**2x**) **GPA: 3.60/4.00**

Experience

Software Engineer | People Inc. **Sep 2024 – Present**

- Engineered a zero-config developer platform & **Deno CLI** ecosystem used by **500+** engineers, automating scaffolding, dependency updates, & provisioning; drove cross-team infra (secure dual-domain local dev, dedicated **LLM** DB clusters, automated API docs) to cut setup time by **90%**, reduce onboard time by **50%**, & eliminate environment drift. Authored onboarding course & mentored **3** new hires.
- Designed & implemented a managed database tooling layer for **Prisma ORM**, integrating secure credential vaults & supporting multi-tenant, multi-schema **PostgreSQL**. Cut DB connection errors by **95%** & accelerated production DB migrations by **10x**.
- Led development of an AI-powered survey & analytics platform (**Nuxt 3**, **Prisma**, **PostgreSQL**, **OpenAI**), delivering real-time analytics, dynamic question generation, & sentiment analysis. **3x** survey completion rates & adopted as the default by all product teams.
- Co-architected a modern design system in **Vue 3**, building **30+** reusable, accessible UI components & enforcing rigorous **QA/test automation**. Achieved **100%** adoption across new applications & a **3x** improvement in UI consistency.
- Modernized user management & access control by refactoring identity provider (**Keycloak**) integration, implementing robust group/role APIs, & delivering a full-featured admin UI. Enabled fine-grained permissions for **10,000+** users & increased platform uptime to **99.99%**.

Software Engineer Intern | People Inc. **Oct 2023 – Sep 2024**

- Engineered bulk data processing & core backend infrastructure (**NestJS**, **TypeORM**, **PostgreSQL**, **GraphQL**, **Redis**, **Kibana**, **Elasticsearch**), delivering **JSON**-driven job orchestration, self-healing service registry flows, robust job-runner services, & real-time job history analytics powering **hundreds** of large-scale data jobs; achieved **99.999%** uptime & reduced operational incidents by **80%**.
- Migrated **5+** mission-critical applications from legacy **Vue 2** to **Vue 3**, refactoring component architecture, state management (**Pinia**), & build tooling. Enhanced performance with **Composition API**, improved type safety with **TypeScript**, & modernized routing patterns.
- Overhauled the company-wide design system, introducing auto-imported components, prop hygiene, & comprehensive **Storybook** documentation. Enabled **100%** cross-app reusability & a **20%** reduction in bundle size.
- Engineered & optimized **Docker** image scripts & **CI/CD** automation (**AWS**, **Jenkins**, **Kubernetes**, **Playwright**), ensuring seamless deployments, robust **QA** integration, & a **50%** reduction in build failures.

Lead Teaching Assistant - Software Engineering | University of Alberta **Sep 2023 – May 2026**

- Design & own **15+** programming assignments across **C**, **Java**, **Bash**, **SQL**, **Android**, **Git** (specs, rubrics, autograders, & starter code) for **4000+** students; assignments mirror industry workflows (version control, testing, DB design, REST-style APIs).
- Built **Python**/**Shell** grading & plagiarism pipelines (Moss/JPlag-style checks, containerized runs); process **1000+** submissions per assignment, cut grading turnaround from **5** days to **<24hrs**, saving **1000+** staff hours per year.
- Lead **20+** TAs: run weekly syncs, assign lab/office-hour coverage, review grading consistency & escalate edge cases; created onboarding docs & run calibration sessions so rubric application stays uniform across sections.
- Deliver **8+** office hours per week & targeted debugging support; students report **25%** improvement in assignment completion rates & higher exam scores in post-course surveys.

Software Developer | NeurAlbertaTech Ltd. **Sep 2022 – Sep 2023**

- Owned end-to-end feature delivery for the org's flagship platform: built **25+** reusable **React**/**Next.js** & **Astro** components, designed **REST** & **WebSocket** integrations with internal APIs, & implemented server-side rendering pipelines that drove **5000+** monthly active users & **40%** improvement in time-on-page.
- Architected type-safe full-stack flows in **TypeScript**; introduced code-splitting & lazy-loaded routes that cut initial load by **35%**, reduced client-side bugs by **60%**, & improved **Lighthouse** performance score from **72** to **94**.
- Shipped CI/CD workflows (**GitHub Actions**) for preview deployments & regression tests; mentored **2** junior developers on component patterns & accessibility standards.

Software Engineer Intern | UniRely **Sep 2021 – Sep 2022**

- Built production **Python**/**Selenium** scraping pipeline ingesting **25K+** structured records from **2500+** sources into **PostgreSQL**; replaced **800+** hours/year of manual research & cut data-acquisition cost by **$12K**/year; pipeline still in use **2+** years later.
- Achieved **1200x** throughput via async workers, distributed rate limiting, & exponential-backoff retry; shipped **Pandas**/**Matplotlib**/**Plotly** dashboards for scholarship ranking & QA adopted by **20+** counselors; reduced time-to-match clients to programs by **50%**.
- Refactored **300+** schema fields, normalized **3** legacy ETL jobs into a single idempotent pipeline; added **Redis** caching & indexed views; query latency down **78%**, enabled real-time filtering for **500+** school/program combinations.

Projects

1. **Gazprea Language Compiler & Playground** | *C++, LLVM, MLIR, ANTLR4, TypeScript, Bun, Hono* | *https://gazlab.onrender.com*
   - Full-featured compiler for the **Gazprea** functional language with a **7-stage MLIR**-to-**LLVM IR** pipeline, **-O3** optimizations, advanced type inference with **8+** analysis passes, qualifier-based promotion, & comprehensive error reporting. Front-end (**ANTLR4**) parses to AST; middle-end lowers through **MLIR** dialects with constant folding & dead-code elimination; backend emits **LLVM IR** for native targets. Interactive web playground (**Bun**/**Hono**, **Vite**, **TypeScript**) supports real-time compile-and-run, syntax highlighting, & output streaming; validated against **2000+** test cases covering generators, matrices, vectors, structs, & functional purity.

2. **TrackIt Personal Finance App** | *Vue 3, TypeScript, ML, FastAPI, Docker, Tailwind, Google OAuth* | *https://trackit-sigma-taupe.vercel.app*
   - Personal finance platform with multi-account **CSV ingestion**, **duplicate detection**, & intelligent transaction categorization using an **ML sentence transformer** with **fuzzy matching** over **1000+** learned merchant patterns, powering dashboards & **Chart.js** analytics. Backend (**FastAPI**, **PostgreSQL**) exposes REST APIs with **JWT** auth & per-user data isolation; front-end (**Vue 3**, **TypeScript**, **Tailwind**) provides responsive dashboards, category rules, & export flows. Deployed via **Docker Compose**; **<100ms** API p95 & **<10ms** categorization latency; **Google OAuth** for sign-in & account linking.

3. **Distributed Multi-Player Poker System** | *Rust, Axum, WebSocket, PostgreSQL, Deno CLI* | *https://github.com/Nandan-18/poker-system*
   - Spec-driven distributed poker platform with a **Rust** casino "house" server managing player accounts, chip conversion, table lifecycle, & **persistent** statistics (rounds played, pots won, folds). Implemented complete game engines for **Five-Card Draw**, **Seven-Card Stud**, & **Texas Hold'em** with dealer rotation, blinds/antes, betting rounds, & configurable house rules (min/max bet, raise limits), supporting **2–5** players per table plus spectator clients over **HTTP/WebSocket** via **Deno CLI** & **egui**.

4. **Tartan Smart Home System** | *IoT, CI/CD, A/B Testing* | *https://cmput402-w25.github.io/tartan-evil-home*
   - Smart home automation system that manages security (sensors, alerts), climate control, & lighting based on occupancy & environmental conditions. Central controller aggregates device state & runs rule-based automation; dashboard for monitoring & manual overrides. Implemented **CI/CD** pipeline for firmware & app updates; **A/B** testing framework for feature flags & gradual rollouts so new logic can be validated before full deployment.

5. **PSRS High-Performance Distributed Sorting** | *C++, pthread, Parallel Algorithms* | *https://nandan-18.github.io/CMPUT-481*
   - Multi-threaded **C++** app using **pthread** to sort datasets up to **3B** elements by distributing work across CPU cores. Implements the **PSRS** (Parallel Sorting by Regular Sampling) four-phase algorithm: sample & sort, pivot selection, partition & local sort, & merge; uses barriers & shared memory for synchronization. Benchmarked against sequential baseline; includes a visualizer for runtime analysis & scaling behavior across core counts.

6. **ScanPal Event Management App** | *Android, Java, Firebase, GCloud, MUI, JUnit* | *https://github.com/CMPUT301W24T30/ScanPal*
   - Production-ready **Android** event management platform for creating, browsing, & joining events via **QR code scanning**, geo-location, & social follow feeds. **Java**/**Android** client with **Material Design** UI; backend on **Firebase** (**Firestore**, Auth, Storage) & **Google Cloud** for real-time concurrent sync, image uploads, & offline-capable caching. Implemented profile management, event CRUD, check-in flows, & notifications; **95% JUnit** test coverage with repository & use-case patterns; built for **4000+**-student course cohort with strict reliability & accessibility requirements.

7. **BearPath** | *Django, Python, JavaScript, Bootstrap, Pandas, Google Directions API* | *https://bearpath.up.railway.app*
   - Live responsive campus map that generates indoor routes through pedways & walkways for efficient class-to-class navigation, especially during Edmonton winters. Backend (**Django**, **Python**, **Pandas**) preprocesses building graphs & waypoints; front-end (**JavaScript**, **Bootstrap**) integrates **Google Maps API** with dark-mode tiles & custom overlays. Implemented Hierarchical Path-finding A* (**HPA***) layered over **Google Directions API** for multi-building, indoor-outdoor routing with minimal detours; gathered **100+** early testers & unit-test coverage for path correctness & edge cases.

8. **MapReduce Library** | *C, pthread, Concurrent Programming* | *https://nandan-18.github.io/mapreduce*
   - **C** implementation of the MapReduce paradigm: thread-safe partitioned key-value stores protected by mutexes, configurable thread-pool & **POSIX** threads. Mapper/reducer interface; intermediate keys sorted in ascending order within partitions. Word-count application (**distwc**) processes multiple input files; validated against **21** sample files (**105K** words total) with correctness checks & **Valgrind** leak detection; **Makefile** build & test harness.

9. **DragonShell** | *C, UNIX, Systems Programming* | *https://nandan-18.github.io/dragonshell*
   - Built a Unix shell in **C** from scratch: full command interpreter with **pipes**, **I/O redirection** (**<**, **>**), **background** execution (**&**), & **signal** handling (**SIGINT**, **SIGTSTP**). Implemented tokenizer, **fork**/**exec**, **dup2** for fd redirection, **pipe()** for IPC; job control & foreground tracking with **waitpid**. Modular design across 6 source files; builtins include cd, pwd, exit with resource-usage reporting.

10. **UNIX File System Simulator** | *C, Systems Programming* | *https://nandan-18.github.io/fs-sim*
    - File system simulator in **C** for a simulated **128KB** disk: **mount**/**unmount**, create/delete/resize files & directories, block & inode management, bitmaps for allocation. Modular layout: core ops, block/inode helpers, bit-manipulation utils, **CLI** driver. Implements directory navigation (**cd**, **ls**), **defragmentation**, & low-level **read**/**write** via system calls; test suite for correctness & disk-invariant checks.

11. Interactive Game of Life Simulator | *RISC-V Assembly, MMIO, Interrupts, CRISCV* | *https://github.com/Nandan-18/CMPUT-229*

- Real-time **Game of Life** cellular automaton in **RISC-V assembly** (**650+** lines) with **timer** & **keyboard interrupts** for paced simulation & interactive input. **MMIO**-driven display with a movable cursor; full keyboard controls (wasd navigation, j/k toggle cells, t pause/run, space single-step, q quit). Toroidal grid with wrapping boundaries; dual-buffer simulation (**updateGrid**, **liveNeighbors**, **getCell**/**setCell**); interrupt handler uses **uscratch**, **ucause** & **uret** for trap routing. Demonstrates low-level systems programming without high-level runtimes.

12. onPhone: Adaptive StarCraft II AI Zerg Bot | *C++, StarCraft II API* | *https://github.com/onPhone/onPhone*

- Adaptive **Zerg** bot for StarCraft II using the game's API: **role-based unit control** with threat targeting and retreat-on-damage micro, supply-triggered **build orders** with unit replacement on loss, and scouting logic to infer enemy location. Modular Scout/Attack/Worker controllers, resource & supply tracking, & real-time decisions in the game's step loop for low-latency reactions.

13. Dr. DrEEG: Realtime Emotion-Powered Karaoke | *Python, EEG, Muse S* | *https://github.com/Neural-Natties/Dr.DrEEG*

- Karaoke app that matches songs to the user's emotional state using real-time **EEG** from **Muse S** headbands. Ingests raw brain-wave streams, extracts features (e.g. band powers), & classifies emotional state; recommendation pipeline selects from a music library based on current & target mood. Built with **Python**; integrates Muse SDK & a simple playback **UI** for live demos.

14. AI Eye | *PyGame, Stable Diffusion, Python* | *https://devpost.com/software/ai-eye-dz5tyh*

- Trivia game where players identify AI-generated images; won **1st place** (**$1,250**) among **50+** teams at Upper Bound AI Game Jam. Full **PyGame** game loop with state machine, scoring, timers, & **10+** asset types; integrated **Stable Diffusion** API with multi-threaded request batching, connection pooling, & cached generations to cut per-round latency by **6x** and inference cost by **$500**/event. Dynamic prompt pipeline ensures unique images per round; supported **100+** concurrent players at live demo with stable frame rates & minimal API throttling.

15. TicCatToe | *TypeScript, PhaserJS* | *https://bytemydust.github.io/TicCatToe*

- Cat-themed **Tic-Tac-Toe** game in **TypeScript**/**PhaserJS**: sprite-based board, win detection, & a playful twist on classic rules. Won **1st place** overall & "horrible hacker" category at DevelopEd 2.0. Demonstrates game loop, input handling, & scene management in **Phaser**; built as a timed hackathon submission.

16. Speed Typing Coach | *Python, Tkinter* | *https://github.com/Nandan-18/Speed-Typing-Coach*

- Typing drill desktop app (**Python**/**Tkinter**) with live WPM/accuracy stats, persistent score history, genre-based word lists (tech, literature, general), & configurable difficulty & session length. Timer pipeline & high-precision key event handling yield sub-**50ms** latency for accurate measurement; local JSON storage for progress & trends.

17. Open Source | *Git, GitHub* | *https://github.com/Nandan-18*

- **Hacktoberfest 3** consecutive years; merged **25+** PRs across **8** repos (tooling, docs, bug fixes). Contributed to community projects across languages & stacks; maintained consistent review & merge cadence.

Skills

Languages: Python, JavaScript, TypeScript, Java, C, C++, Rust, RISC-V Assembly, SQL, HTML, CSS, Bash/Shell, LaTeX, Markdown.

Frameworks & Libraries: React, Vue 2/3, Nuxt, Next.js, Astro, Django, FastAPI, NestJS, Axum, Prisma, TypeORM, Pinia, Tailwind, Bootstrap, Bulma, MUI, Android, Vite, Bun, Hono, egui, PyTorch, Pandas, NumPy, Matplotlib, Plotly, Chart.js, Selenium, PyGame, PhaserJS, Tkinter, JUnit.

Databases & Backend: PostgreSQL, MongoDB, Firebase (Firestore, Auth, Storage), Google Cloud, Redis, Elasticsearch, Kibana, GraphQL, REST, WebSocket, JWT, Google OAuth, Google Maps/Directions API.

Systems & Compilers: LLVM, MLIR, ANTLR4, pthread/POSIX, UNIX, Valgrind, Makefile; RISC-V, MMIO, interrupt handling, CRISCV; parallel algorithms, concurrent programming.

DevOps & Tools: Docker, Docker Compose, Kubernetes, AWS, Jenkins, GitHub Actions, CI/CD, Playwright, Storybook, Keycloak, Deno, Git, VSCode, Jira, OpenAI/APIs, Lighthouse; AI coding tools (Cursor, Claude Code, GitHub Copilot, ChatGPT, Windsurf, Tabnine, Gemini, MCP).

Other: Material Design, ETL, IoT, A/B testing, accessibility (a11y), QA/test automation, Stable Diffusion API, EEG/Muse SDK.

Relevant Coursework: Intro to CS I, Intro to CS II, Practical C Programming, Algorithms I, Computer Architecture, Artificial Intelligence I, Discrete Mathematics, Database Management, Software Engineering I, Practical Algorithmics, Non-Procedural Programming, Advanced C++, Operating Systems Design, Software Quality, Compiler Design, Distributed Systems, C++ Programming, Information Systems, Java Programming, UNIX Systems, Distributed Rust Systems, Calculus I–IV, Linear Algebra I & II.