

SUMIT PARMAR

Software Engineer | Full-Stack & Backend (Java, Spring Boot, React) | Cloud & DevOps | Generative AI

Buffalo, NY | (716) 717-2580 | parmarsumit.k04@gmail.com | linkedin.com/in/sumit-parmar-07a544278 | github.com/sumitparmar19

SUMMARY

Software engineer and MS Computer Science candidate (December 2026, Systems Track) with 4 internships across full-stack web, mobile, and test automation, most recently at a US AI startup. Builds systems end to end: a healthcare drug-safety platform of Java 21/Spring Boot microservices with Kafka, JMS, SOAP/XML, Oracle, Jenkins, and Kubernetes; a live investing platform with an LLM/RAG AI service, a PostgreSQL ledger, and AWS infrastructure in Terraform; and a Unix-style kernel in C with virtual memory.

EDUCATION

Master of Science in Computer Science (Systems Track) | Jan 2025 – Dec 2026

University at Buffalo, State University of New York, Buffalo, NY | GPA: 3.5/4.0

Coursework: Algorithms, Operating Systems, Computer Architecture, Computer Networks, Computer Security, Data Intensive Computing, Machine Learning, Deep Learning

Bachelor of Technology in Computer Engineering | Jun 2021 – Dec 2024

Indus University, Ahmedabad, India | GPA: 9.08/10

TECHNICAL SKILLS

Languages: Java (Java 21), Python, TypeScript, JavaScript, C, C++, SQL, Bash

Frontend & Mobile: React.js (React 18/19), Next.js, React Native (Expo), Redux, Zustand, TanStack Query, React Router, Tailwind CSS, Vite, Recharts, HTML5, CSS3

Java & Spring: Spring Boot, Spring Security (OAuth2 Resource Server, JWT), Spring Data JPA, Hibernate, Spring Kafka, Spring JMS, Spring Web Services (SOAP, WS-Security), Flyway, Maven

Backend & Messaging: RESTful APIs, Microservices, Event-Driven Architecture, Apache Kafka, JMS (ActiveMQ Artemis), Transactional Outbox, Idempotent Consumers, Dead-Letter Queues, SOAP, XML/XSD, Node.js, Express.js, FastAPI, Pydantic, JWT Authentication, Rate Limiting

Databases: PostgreSQL, Oracle Database, Microsoft SQL Server, MySQL, MongoDB (NoSQL), Redis, Supabase (Row-Level Security), ChromaDB (Vector DB), Prisma ORM, Mongoose, Transactions, Row Locking, Indexing & Query Optimization, Data Modeling, Data Migration

AI & LLM: Generative AI, LLMs, Anthropic Claude API, LangChain, RAG, Embeddings, Prompt Engineering, Structured Output, Transformers (Self-Attention), Hugging Face, ONNX Runtime (Java), Human-in-the-Loop AI, AI-Assisted Coding Tools

Data & ML: Apache Spark (PySpark), Databricks, Delta Lake, Medallion Architecture, ETL Pipelines, Apache Airflow, MLflow, Spark MLlib, PyTorch, scikit-learn, LightGBM, LSTM Autoencoders, pandas, NumPy, Matplotlib, Jupyter

Cloud & Infrastructure: AWS (ECS Fargate, RDS, ALB, ECR, VPC, Lambda, Step Functions, EventBridge, Secrets Manager, CloudWatch, SNS, IAM, S3, EC2), Terraform (AWS, Azure), DigitalOcean, Neon, MongoDB Atlas

DevOps: Docker, Docker Compose, Kubernetes, Helm, kind, Strimzi, Jenkins (Configuration as Code), CI/CD (GitHub Actions), GitHub OIDC, gitleaks Secret Scanning, Checkov, Linux, Git, GitHub

Observability: OpenTelemetry, Prometheus, Grafana, Grafana Cloud (Tempo, Loki), Distributed Tracing, Alerting, Structured Logging (pino), Sentry

Testing & Quality: JUnit 5, Mockito, Testcontainers, Jest, Supertest, pytest, Vitest, Playwright, Selenium WebDriver, Selenium Grid, TestNG, BrowserStack, k6 Load Testing, Fuzz Testing, CodeQL, cppcheck; Unit, Integration, and End-to-End Testing

Systems: Operating Systems (processes, scheduling, virtual memory, paging), Concurrency and Synchronization, x86, QEMU, Data Structures & Algorithms, OOP, System Design

Healthcare Domain: Pharmacovigilance, Adverse Event Case Processing, ICH E2B(R3), MedDRA, FDA FAERS

Practices: Agile/Scrum, Jira, Code Review, Architecture Decision Records, Technical Writing, SDLC

WORK EXPERIENCE

Full Stack Development Intern | Uplifty AI | Texas, USA (Remote) | May 2026 – Aug 2026

- Developed cross-platform mobile features (user profiles, social connections, and the challenge flow) for a consumer social application in **React Native (Expo)** and TypeScript, converting Figma designs into reusable UI components.
- Designed **PostgreSQL** schemas on Supabase with row-level security (RLS) policies that enforce per-user access inside the database, and built serverless backend logic with Supabase Edge Functions in TypeScript.
- Integrated an **LLM-powered feature** into the product and debugged rendering, navigation, and client-to-database integration issues on Android emulators and physical devices.
- Shipped each feature end to end through Git feature branches, peer-reviewed pull requests, and Jira tickets in a distributed **Agile** team, using daily stand-ups and mob programming sessions to unblock integration issues quickly.

Software Engineer Intern | Prodigy InfoTech | Mumbai, India (Remote) | Jan 2025 – May 2025

- Designed and developed full-stack applications with **React, Next.js, TypeScript, Node.js, and PostgreSQL**, implementing RESTful APIs, authentication, and modular backend services.
- Architected asynchronous data processing workflows with Python, **Apache Kafka**, and **Apache Airflow**, covering data ingestion, validation, transformation, and analytics reporting.
- Built and maintained a regression suite of **450+ automated test cases** (Jest, Supertest, Playwright), cutting regression execution time from **8 hours to 25 minutes**.
- Developed an end-to-end UI **test automation** framework in Java with Selenium WebDriver and TestNG (Page Object Model) for authentication and checkout flows, running in parallel across 4 browsers on Selenium Grid and BrowserStack.
- Built **CI/CD** pipelines with GitHub Actions, Docker, and AWS with code-quality checks, automated tests, and deployment validation, and optimized PostgreSQL queries with indexing, query analysis, and schema improvements to cut response times.

- Built a **Java** contact management application with CRUD operations and persistent storage, and deployed it on AWS (EC2, S3).

Full Stack Development Intern | Prodigy InfoTech | Mumbai, India (Remote) | Apr 2024 – May 2024

- Built responsive web applications in **JavaScript**, HTML5, and CSS3: a landing page with light/dark mode and sign-in, a stopwatch, a tic-tac-toe game, and a weather app backed by a REST weather API.
- Refactored sequential API calls to async/await with parallel request batching, removing redundant round-trips on first render.

Web Development Intern | Green Apex Solutions Limited | Ahmedabad, India | Jun 2023 – Jul 2023

- Built reusable **React.js** components in a component-driven architecture and introduced a Redux store to replace prop drilling, giving related components a single source of truth.

PROJECTS

PharmaVigil: AI Drug-Safety Platform | Java, Spring Boot, Kafka, Oracle | GitHub | Live Demo | 2026

- Architected a healthcare (pharmacovigilance) platform as five **Java 21 Spring Boot** microservices with a React 19/TypeScript UI: each adverse drug event report is de-duplicated, coded, triaged, human-reviewed, and filed as ICH E2B(R3) XML.
- Built an **event-driven Kafka pipeline** with a transactional outbox, idempotent consumers, three retries, and dead-letter topics, plus a **JMS** (ActiveMQ Artemis) hand-off to a mock regulator that validates each file and returns ACK or NACK.
- Processed **12.9M real FDA adverse-event reports** (2019–2026) into 10.8M de-duplicated cases with Spark and Delta Lake (98.5% of drug names mapped to RxNorm); best signal-detection AUC 0.82 on public reference sets.
- Load tested with k6 and Prometheus, fixed two bottlenecks (a query per duplicate candidate, a serial outbox relay), and cut p95 report-to-coded latency from **103 s to 7.5 s** as throughput rose 3.7x to 2,300 reports/min with 0 failed requests.
- Secured REST and SOAP APIs with **Spring Security** (OAuth2 resource server, JWT) and WS-Security; persisted data with JPA/Hibernate and Flyway on **Oracle** and SQL Server; every XML export is validated against the official ICH schemas.
- Integrated ML models (MedDRA coding, LightGBM seriousness triage, duplicate detection) into Java with **ONNX Runtime**, tested for parity with Python; each AI suggestion keeps its model version beside the reviewer's final decision.
- Automated CI/CD with **Jenkins** and GitHub Actions (106 JUnit 5, Mockito, and Testcontainers tests, plus pytest, Vitest, Playwright, CodeQL); ran the stack with **Helm on Kubernetes** and 3 Grafana dashboards.

Nivesh-Path: AI-Powered Investing Platform | React, Node.js, PostgreSQL, AWS | GitHub | Live Site | 2025 – Present

- Architected and deployed a live paper-trading platform (nivesh-path-vzeak.ondigitalocean.app) as **microservices**: a React 18/TypeScript app, a Node.js/Express REST API (40+ endpoints, JWT on every private route), and a Python/FastAPI AI service.
- Built a generative AI portfolio advisor with **LangChain** and the Anthropic Claude API using RAG over per-user ChromaDB embeddings, with schema-validated output and a rule-based fallback; numbers are computed in code, never by the LLM.
- Built a **Behavioral Mirror** that detects panic selling, FOMO buying, and overconcentration after every trade with a versioned rule engine, shows them in the trade panel within 1.5 seconds, and uses an AI coach to explain each pattern.
- Built a **PostgreSQL** ledger with Prisma (one transaction per order, SELECT FOR UPDATE, CHECK constraints, idempotency keys; 20 parallel orders never overdraw) and migrated live data from MongoDB with a dry-run, copy, and verify tool, matched to the cent.
- Served live market data for 50+ US stocks and ETFs (Finnhub) through a **Redis** cache with per-resource TTLs; k6 load tests (42,000 requests, 0 errors) show it cuts read p95 latency 2.3x (22.3 to 9.6 ms).
- Automated **CI/CD** with GitHub Actions and Docker: 300+ unit, integration, and end-to-end tests (Jest, Supertest, pytest, Vitest, Playwright), gitleaks secret scanning of the full history, and a live smoke test that signs up, trades, and deletes a real account.
- Added **observability** with OpenTelemetry and Grafana Cloud: traces that follow a request across the Node and Python services, structured JSON logs with trace IDs and redacted secrets, a 15-panel dashboard, and tested alerts.
- Deployed the stack to **AWS** with **Terraform** and GitHub Actions over OIDC (no stored keys): ECS Fargate behind an ALB, RDS PostgreSQL, and CloudWatch alarms; it autoscaled from 1 to 3 tasks under a 96,000-request k6 test.
- Built a **serverless** nightly job (EventBridge Scheduler, Step Functions, Lambda) that saves daily closing prices to PostgreSQL and re-checks every user's concentration, with retries, backoff, SNS failure emails, and IAM limited by a permissions boundary.

Unix-Style OS Kernel: System Calls, Concurrency, and Virtual Memory in C | C, x86 | GitHub | 2025 – 2026

- Implemented a POSIX-style **system-call layer** in C for the Pintos x86 kernel (process lifecycle, file I/O, IPC) with race-free synchronization, plus a **priority scheduler** with nested donation and a 4.4BSD MLFQS (27/27 tests).
- Built demand-paged **virtual memory** (supplemental page table, frame table, swap, mmap, stack growth, frame pinning) and measured eviction policies across a benchmark matrix published to GitHub Pages: clock eviction needed 100x fewer swap-ins than FIFO.
- Turned it into a **sandbox for untrusted programs**: Unix pipes and dup2, plus cgroups-style hierarchical limits on processes, memory pages, open files, and CPU time that contain fork bombs and CPU hogs.
- Gated every change in CI (Docker, GitHub Actions, CodeQL, cppcheck) on 245 tests; a system-call **fuzzer** ran 116,000 calls with zero kernel panics, and code review plus fuzzing found and fixed 5 kernel bugs, including a sandbox escape and a use-after-free race.

Cloud Financial Analytics Pipeline | PySpark, Databricks, Delta Lake, MLflow, PyTorch | Data Intensive Computing | 2026

- Built a Bronze/Silver/Gold medallion **ETL pipeline** on Databricks with PySpark and Delta Lake, applying schema enforcement, deduplication, and feature engineering, with incremental loads that merge new data without reprocessing history.
- Trained Spark MLlib and scikit-learn models tracked in **MLflow**, and a **PyTorch** LSTM autoencoder that flags abnormal market volatility by reconstruction error.

Transformer Language Model with PPO Fine-Tuning (team of 3) | PyTorch | Deep Learning | Sep 2026 – Present

- Co-developing an **encoder-decoder Transformer** in PyTorch with no pretrained weights that writes stories using required keywords; next, pretraining on TinyStories and **PPO** (RLHF-style) fine-tuning on a keyword-inclusion reward with a KL penalty.
- Validated the data pipeline, embeddings, and causal multi-head **self-attention** with shape, parameter-count, and causality tests (0.0 change in earlier outputs when future tokens change), and found an end-of-text truncation bug in the data pipeline.