KHALID RIZVI

Principal Architect | **GenAI • Cloud Platforms • Java • Golang • Python**

Vienna, VA 22182 • 703-656-6394

khalid.rizvi@icloud.com • khalidrizvi.com

linkedin.com/in/khalidrizvi

Professional Summary

Principal Architect and GenAI Platform Engineer with 25+ years of experience delivering mission-critical systems across government, healthcare, finance, and telecom. Educational foundation in Mechanical Engineering and Computer Science provides strong grounding in calculus, linear algebra, differential equations, and numerical methods, informing a disciplined, first-principles approach to system design and Machine Learning.

Continues to work hands-on in production environments while leading architecture. In recent years, focused on responsibly bringing Generative AI into enterprise settings — designing Retrieval-Augmented Generation (RAG) platforms, LLM proxy layers, and agentic orchestration frameworks on AWS and Azure. Strong understanding of Large Language Models (LLMs), neural networks, gradient-based optimization, and model evaluation beyond surface-level API integration, with direct implementation in Python-based ML and orchestration stacks.

Polyglot programmer with experience spanning C, C++, Java, Scala, Python, JavaScript, TypeScript, and Go. Combines long-tenured architectural judgment with practical engineering execution, selecting appropriate tools and patterns to solve business problems while ensuring systems remain secure, observable, maintainable, and production ready.

Professional Experience

GenAI Senior Platform Engineer *Sep 2025 – Jan 2026*

Ampcus (Federal Reserve Board) *Chantilly, VA / Washington, DC*

- Contributed to the enhancement and ongoing evolution of an AWS-hosted, open-source conversational AI platform, with a focus on integrating Amazon Bedrock as the managed foundation-model and RAG backbone in an enterprise delivery context. Operated with end-to-end ownership across design, implementation, and operational readiness, shaping platform architecture rather than limiting involvement to isolated features, and writing hands-on Python services and orchestration components while using GitHub Copilot to accelerate boilerplate and routine coding tasks without compromising code review rigor.
- Worked within an existing model-access abstraction layer that routed inference traffic through Amazon Bedrock and other providers via LiteLLM, enabling controlled use of multiple foundation models behind a consistent, policy-aware API surface. Helped refine observability and cost-awareness around this layer so model selection, latency, token usage, and spend could be monitored, metered, and tuned in line with enterprise expectations, including management of API keys, throttling limits, and account usage boundaries.
- **Retrieval-Augmented Generation (RAG):** Designed and implemented an end-to-end RAG lifecycle on AWS, leveraging Bedrock for embeddings and model inference to support scalable retrieval over public and domain-specific content, implemented as Python-based microservices and FastAPI endpoints.
- On the ingestion side, owned the ingestion pipeline and its deployment, designing an event-driven workflow that periodically fetched content from remote APIs, applied structured transformations while preserving source fidelity, and persisted normalized artifacts for downstream retrieval. Synchronized transformed data into a vector search layer backed by managed embedding services exposed through Bedrock, enabling efficient semantic indexing, repeatable ingestion runs, and operational scalability using Python, uv, and containerized batch workers.

- Authored and maintained infrastructure-as-code to provision and operate the ingestion and retrieval components, ensuring consistent, reproducible environments across stages. Integrated these pieces into a commit-driven CI/CD workflow so code changes produced updated containers and infrastructure automatically, keeping the RAG stack continuously deployable with minimal manual intervention.
- On the inference side, implemented a semantic retrieval workflow in which user prompts were embedded using a consistent Bedrock-based embedding strategy, matched against the vector store to retrieve relevant context, and composed into prompts for downstream model calls. Responses were generated through the model-agnostic inference layer backed by Bedrock and LiteLLM, allowing teams to swap or mix foundation models while preserving predictable behavior and quality guarantees at the application boundary; this layer was primarily implemented in Python with LangChain-style abstractions.
- The overall RAG architecture emphasized factual grounding, relevance, and extensibility, mirroring the ingestion–embeddings–retrieval patterns of Bedrock Knowledge Bases while remaining flexible enough to evolve with the surrounding ecosystem.
- **Agentic AI research and design:** Led and contributed to research and design exploration of agentic AI patterns on top of Bedrock-hosted models, focusing on multi-agent collaboration, orchestration strategies, context handling, and knowledge-grounded reasoning. Assessed coordination patterns, grounding approaches, and control mechanisms aimed at improving factual accuracy, reducing hallucinations, and informing long-term architectural decisions for conversational systems operating under enterprise constraints, with prototypes implemented in Python for rapid experimentation.
- **Testability and delivery enablement:** Strengthened platform testability and delivery confidence by producing testable, containerized builds of the Bedrock-integrated services and tightening feedback loops between development, evaluation, and operations. This enabled faster iteration on RAG and agentic designs, clearer assessment of architectural trade-offs, and more reliable judgments about platform readiness as capabilities expanded.

Technology used: Python (FastAPI, uv, LangChain), TypeScript/JavaScript, RESTful APIs, FastAPI, React, HuggingFace conversation UI, AWS (EC2, S3, Lambda, CloudWatch), Docker, Kubernetes, Terraform/CloudFormation, CI/CD pipelines, CloudWatch, RAG pipelines, semantic search, vector databases, Amazon Bedrock Knowledge Bases, LiteLLM, LangChain, Agile/Scrum

GenAI Senior Platform Engineer *Jul 2025 – Sep 2025*
CTIS (NIH / CMS) *Rockville, MD*

- Led the design and implementation of a production-oriented Retrieval-Augmented Generation (RAG) platform on AWS for specialized oncology datasets, with a strong focus on architectural transparency, data control, and operational rigor. Designed the solution to avoid opaque "black box" constraints by building a custom, full-lifecycle ingestion and inference architecture that aligned with enterprise data governance expectations while selectively leveraging Amazon Bedrock and related AWS services, implemented as Python-based microservices and orchestration flows.
- **Ingestion and knowledge foundation:** Designed and implemented an automated ingestion pipeline that continuously collected content from authoritative oncology data sources, normalized and enriched it, and ensured consistency and traceability across the retrieval lifecycle. Engineered the ingestion process for repeatability and auditability, and prepared content for downstream semantic indexing at scale using vector search and embedding services, including Bedrock-hosted models where appropriate, with ingestion logic written in Python.
- **Vector search and embedding evaluation:** Conducted a structured evaluation of the vector search and embedding ecosystem, benchmarking multiple vector store and embedding options along dimensions such as retrieval quality, latency, operational complexity, and cost. Used these findings to select and tune the vector search layer and embedding strategy, optimizing the platform for high-precision semantic retrieval with predictable performance characteristics, using Python notebooks and evaluation harnesses.
- **Bedrock, Lex, and model orchestration:** Integrated Amazon Bedrock Knowledge Bases–style patterns and

Amazon Lex into a composable orchestration framework, maintaining architectural control while taking advantage of managed capabilities for conversational interfaces and model hosting. Implemented a model-agnostic inference layer using LangChain's provider-agnostic abstractions to expose a consistent, OpenAI-compatible interface over Bedrock and other LLMs, minimizing application-level changes when swapping or adding models while preserving centralized auditing, token metering, and behavioral consistency, with the core orchestration implemented in Python.

- **Agentic workflows and extensibility:** Extended the platform with modular agent-style orchestration capabilities to support controlled experimentation, hybrid workflows, and future extensibility. Enabled local-first and cloud-based development paths so teams could iterate quickly on RAG and conversational flows, validate designs against real oncology use cases, and gradually harden successful patterns toward production.
- Delivered the resulting framework together with documentation and operational guidance, enabling the client team to continue testing, evolve the architecture, and progress toward production deployment under their enterprise compliance and governance standards.

Technology used: Python (FastAPI, LangChain), TypeScript/JavaScript, RESTful APIs, FastAPI, AWS (Lex, EC2, S3, CloudWatch, Bedrock), Docker, CI/CD pipelines, semantic search, vector databases, OpenAI-compatible APIs, LangChain, PostgreSQL, JWT, OAuth2

Associate Partner, Cloud & Enterprise Architect *Jul 2015 – Jun 2025*
DXC Technology *Ashburn, VA*
Selected work:

- *Nokia Networks – KPI Analytics / Signal Project (Azure, Go, Power BI):*
  - Led design and implementation of a batch and near-real-time KPI ingestion and analytics solution, with high-throughput Go services in Azure Container Apps polling multiple REST endpoints on 15-minute and daily schedules and loading into Azure SQL for downstream Power BI dashboards.
  - Optimized for concurrency and observability using goroutines, tuned timeouts/retries, and metrics dashboards to give operations clear visibility into latency, error patterns, and API backpressure.
  - Delivered interactive KPI dashboards over Azure SQL using Power BI and React/Recharts-style visualizations so operations teams could slice latency, error rates, and capacity trends in real time.

  Technology used: Go, TypeScript/JavaScript, React, Power BI, RESTful APIs, Azure (Azure Container Apps, Azure SQL), SQL, Docker, CI/CD pipelines, logging/metrics/tracing, Agile/Scrum.
- *Network of Giving – Donation Platform (AWS, ECS/Fargate, API Gateway):*
  - Architected a microservices platform on ECS Fargate with API Gateway as a secure entry point for bank integrations and RabbitMQ via AWS MQ for decoupled asynchronous flows such as receipts, settlement, and analytics pipelines.
  - Implemented mTLS, OAuth2/JWT-based authorization, IAM least-privilege, and CloudWatch monitoring and logging, and established CI/CD pipelines and infrastructure-as-code with CloudFormation/Terraform for repeatable deployments.
  - Drove secure coding practices for all microservices, including input validation, least-privilege IAM, encrypted secrets, and OWASP-style hardening of REST APIs and authentication flows, across services implemented in Java, Python, and Go, using GitHub Copilot as an AI coding assistant to accelerate boilerplate and test scaffolding while maintaining code review standards.

  Technology used: AWS (ECS/Fargate, API Gateway, EC2, S3, RDS, DynamoDB, CloudWatch, AWS MQ), RabbitMQ, Docker, Kubernetes, Terraform/CloudFormation, CI/CD pipelines, infrastructure-as-code, logging/metrics/traci (CloudWatch, X-Ray), Go, Java, Python, TypeScript/JavaScript, SQL, RESTful APIs, Flask, FastAPI, JWT, OAuth2, mTLS, IAM, RBAC, secure coding practices, Agile/Scrum.
- *Grants.gov – Modernization & Serverless Edge (AWS):*

  - Built a Bash-driven deployment framework that automated complex web-form deployments by orchestrating data from relational DBs, S3, and Excel, replacing manual, error-prone runbooks.
  - Led a lift-and-shift of PDF authoring, Java/JavaScript applications, and historical data from Sun servers to AWS (EC2, RDS, S3, DMS, CloudWatch, Auto Scaling) to improve resilience, scalability, and operations.
  - Re-architected a JavaScript application behind API Gateway with Lambda authorizers and JWT validation; designed VPC peering/endpoints and IAM roles so Node.js Lambdas could securely call services in a separate VPC, with CloudWatch/X-Ray for deep tracing.
  - Modernized JavaScript front-ends and Node.js services with TypeScript adoption and Webpack-based build pipelines, improving maintainability and deployment consistency.

  Technology used: TypeScript/JavaScript, Webpack, Vite, RESTful APIs, AWS (EC2, S3, RDS, Route53, DMS, Lambda, API Gateway, CloudWatch), Bash, SQL, PostgreSQL, Docker, SAM, JWT, OAuth2, IAM, RBAC, CI/CD pipelines, secure coding practices, Agile/Scrum.
- *LA Metro – Transit Integration Platform:*
  - Built integrations using Dell Boomi and Java to unify telemetry and GPS data from thousands of vehicles plus supply chain and passenger information into centralized, real-time monitoring and predictive maintenance views.
  - Used XML/JSON contracts, schedulers, and resilient integration patterns with message queues to ensure durable and observable data flows.

  Technology used: Dell Boomi, Java, RESTful APIs, XML/JSON, RabbitMQ, Apache Camel, SQL, Docker, CI/CD pipelines, logging/metrics/tracing, Agile/Scrum.
- *NY MTA – Electric Bus & Pandemic Cleaning Systems:*
  - Designed a Spring Boot-based REST service to automate kWh data retrieval from Siemens chargers into SPEAR, replacing manual processes and enabling reliable energy analytics, implemented as Java microservices.
  - Built a pandemic cleaning tracking application that transformed train consist feeds into Infor EAM work orders with cron-driven scheduling, parallel processing, reporting stores, and email notifications, leveraging Spring Boot and Java for batch and online workloads.

  Technology used: Spring Boot, JPA/Hibernate, Java, RESTful APIs, SQL, PostgreSQL, Redis, Docker, CI/CD pipelines, logging/metrics/tracing, Agile/Scrum.
- *Daito Corporation – Mainframe Batch Modernization (Japan/Vietnam):*
  - Took ANTLR/QTE-generated Java from COBOL/CL workloads and re-architected it into optimized Spring Boot/Tomcat services, reducing batch runtime from hours to minutes while preserving business semantics.
  - Used Redis for prefetching and caching, the Memento pattern for state management, and careful tuning of data access in Java 8 with JPA/Hibernate and JDBC for high-throughput batch operations.

  Technology used: ANTLR/QTE migration tooling, Java, Spring Boot, JPA/Hibernate, Redis, Velocity, SQL, Docker, CI/CD pipelines, Agile/Scrum.
- *Mastercard – Global Loyalty Integration:*
  - Designed a meta-driven integration layer in Java that ingested heterogeneous partner loyalty feeds (files, XML/JSON APIs) and normalized them into a unified internal schema, decoupling partner interfaces from core systems and enabling faster onboarding.

  Technology used: Java, RESTful APIs, XML/JSON, SQL, Apache Camel, Velocity, Docker, CI/CD pipelines, Agile/Scrum.

Senior Solutions Architect *Nov 2006 – Jul 2015*
RIZ Consulting *Vienna, VA*

- Led federal IT modernization initiatives for EPA, VA, USDA, and Grants.gov, focusing on secure, interoperable architectures and reusable frameworks for mission-critical, regulation-bound systems, primarily on Java-based stacks.
- Modernized J2EE line-of-business applications and legacy integration tiers into API-driven, service-oriented platforms using SOAP/REST and XML/JSON contracts.
- Designed shared integration layers using messaging, ESB/API gateways, and centralized identity and access management aligned with federal security baselines; drove adoption of CI, scripted provisioning, and configuration management.
- Worked closely with security/compliance teams to align with FISMA, NIST, and ATO processes while improving operability and interoperability.
- Worked in Agile/Scrum teams with test-driven development and systematic code reviews, emphasizing secure coding practices for Java-based web and integration services.

Technology used: Java, Java EE, Spring Boot, Spring Framework 3.x, JPA/Hibernate, RESTful APIs, SOAP Web Services, XML/JSON, JMS, Apache Camel, Dell Boomi, SQL, PostgreSQL, Oracle 11g, Docker, Jenkins/Azure DevOps, CI/CD pipelines, infrastructure-as-code, JWT, OAuth2, Kerberos, IAM, RBAC, audit logging, secure coding practices, FISMA/NIST, Agile/Scrum, TDD.

Department of Veterans Affairs Highlights

Senior Rules & Integration Architect – Washington, DC, Denver, CO, & Nashville, TN (Nov 2007 – Oct 2012)

VBA – Counselor Quality Measurement Platform

- Architected and built a web-based counselor quality evaluation system using Java EE and Oracle WebLogic Portal to measure and improve how veterans' benefits counselors serve veterans.
- Designed and implemented the data ingestion layer to receive and process XML case review data over Java EE web services, and normalize it for downstream processing and rule evaluation.
- Built the questionnaire-based UI on Oracle WebLogic Portal using Beehive, Struts 1.3, JSP/JSTL, jQuery, and Ajax so counselors and supervisors could enter and review information easily.
- Designed the core platform using Guice for dependency injection, EJB3 stateless session beans for business logic, and JPA with WebLogic Kodo for persistence, keeping services loosely coupled and scalable.
- Implemented rule-based evaluation for case reviews and added PDF reporting with iText so stakeholders could export evaluation results and dashboards for reviews and planning.

Technology used: Java EE, Struts 1.3, JSP/JSTL, jQuery, Ajax, Oracle WebLogic Portal, Beehive, Guice, EJB3, JPA, Oracle 11g, WebLogic Kodo, XML, Web Services, iText, SQL, Eclipse, TDD, Git, Linux, Windows, Agile/Scrum, secure coding practices.

CHAMPVA – Rules Automation & Business Process Modernization

- Worked as a hands-on JRules expert and rules architect for CHAMPVA, leading design and implementation of business rules for application intake, enrollment, eligibility, and claims/benefits, integrated into Java-based services.
- Designed and rolled out a rules engine on JRules 7.1.4 to automate decisions, replacing manual steps with repeatable, auditable decision logic integrated with Spring 3.x and Hibernate 3.5.
- Defined rule governance and lifecycle processes for how rules are stored, versioned, tested, deployed, and retired, maximizing reuse of decision services and individual rules.
- Integrated the rules layer with JMS, Spring Batch, SOAP Web Services, Maven 3, Oracle 11g, and WebLogic 11g on Linux and Windows for robust, scalable batch and online processing.
- Built productivity tools including a TestNG test class generator (Boolean truth-table coverage) and an Excel-to-

SQL test data loader to accelerate automated testing.

Technology used: JRules 7.1.4, Business Rule Management System (BRMS), Agile Business Rule Development (ABRD), Spring Framework 3.x, Hibernate 3.5.x, JMS, Spring Batch, Web Services, Maven 3, Oracle 11g, WebLogic 11g, Java, JPA, SQL, TestNG, Excel, Eclipse, Git, Linux, Windows, Agile/Scrum, TDD, secure coding practices.

CodeAnalyzer – Static Code Analysis Tool

- Designed and built CodeAnalyzer, an Eclipse-based static analysis tool to promote secure, reliable, and maintainable Java code across VA projects.
- Implemented a rule-based data-flow analysis engine to catch issues such as resource leaks, weak exception handling, and SQL injection risks before they reached testing or production.
- Delivered CodeAnalyzer as an Eclipse plug-in and worked with teams to integrate it into builds and code reviews, standardizing coding practices around security and performance.

Technology used: Java, Eclipse, TDD, Git, SQL, XML, Linux, Windows, secure coding practices.

Senior Architect *Jul 1997 – Nov 2006*
CSC Consulting *Waltham, MA*

- Architected large-scale modernization and integration initiatives for Fortune 500 clients across financial services, telecom, and retail, focusing on reusable frameworks and secure integration patterns.
- Worked in distributed, travel-heavy roles as an expert in Forte, Java, C++, and enterprise security, including Kerberos-based authentication for USAA and integrations with complex legacy environments.
- Led early web modernizations such as esteel.com, reworking JavaScript and C++ components into Java using Struts/Tiles.
- Directed eMedNY's migration from a 600+ screen PowerBuilder client to the web by building "Codester," an automation tool that used SQL schemas and Java metaclasses to generate DAOs, controllers, and Velocity templates; Codester became a reusable asset and earned a Technical Excellence Award.

Technology used: Java, C/C++, Forte, Struts/Tiles, JavaScript, Velocity, SQL, Kerberos, RESTful APIs, XML/JSON, Eclipse, Git, secure coding practices, Agile/Scrum.

Core Technical Skills

Architecture & Platforms: Generative AI platforms (RAG, agents, LLM gateways), cloud-native architecture, microservices, event-driven and serverless systems, mainframe modernization, enterprise integration.
Languages: Python (10+ years), TypeScript/JavaScript (20+ years), Java (25+ years), Go (8+ years), SQL, Bash, earlier C/C++ and PowerBuilder, Forte.
Web & APIs: RESTful APIs, API design and gateways, Flask, FastAPI, SOAP Web Services.
Frontend & Visualization: React, TailwindCSS, modern front-end build tools (Webpack, Vite), data visualization libraries (d3.js), jQuery, Ajax.
Cloud & Integration Platforms: AWS (EC2, S3, RDS, Route53, Lambda, API Gateway, ECS/Fargate, DMS, DynamoDB, CloudWatch, AWS MQ), Azure (Azure Container Apps, Azure SQL, Power BI), OpenShift, Dell Boomi.
AI / GenAI: Retrieval-Augmented Generation, RAG pipelines, semantic search, vector databases/vector stores, Amazon Bedrock Knowledge Bases, LiteLLM, OpenAI-compatible APIs, LangChain, prompt design and evaluation, agentic orchestration, data ingestion and lineage, evaluation and observability for AI workloads.
Data & Persistence: PostgreSQL, Oracle (including Oracle 11g), SQL Server, Azure SQL, JPA/Hibernate, ORM-based data models.
Frameworks & Libraries: Spring Boot, Apache Camel, Velocity, Redis, RabbitMQ, Kafka, ANTLR/QTE migration tools, iText, JRules.
DevOps & Infrastructure as Code (IaC): Docker, Kubernetes, Terraform, AWS CloudFormation, AWS Server-

less Application Model (AWS SAM), Jenkins, Azure DevOps, CI/CD pipelines, automated deployments, infrastructure-as-code (IaC), Linux system administration.
Developer Tooling & AI Assistants: Git, GitHub, GitHub Copilot, IDE-based debugging and profiling workflows.
Observability & Edge: Logging/metrics/tracing (Amazon CloudWatch, AWS X-Ray), Nginx.
Security & Governance: JWT, OAuth2, mTLS, Kerberos, IAM, RBAC, Microsoft Entra ID, zero-trust-style thinking, audit logging, secure coding practices, data governance, auditability, security-by-design and compliance-driven design in regulated environments (FISMA/NIST, federal programs, healthcare).

Education

Master of Science, Computer Science
California State University, Sacramento 1995–1997

Master of Science, Mechanical Engineering
NED University of Engineering & Technology 1986–1988

Bachelor of Science, Mechanical Engineering
NED University of Engineering & Technology 1982–1986

Certifications

AWS Certified Cloud Practitioner – 2024
Big Data Analysis with Scala and Spark – 2017
Functional Programming Principles in Scala – 2014
MongoDB for Java Developers – 2013
Java Certified Architect – 1999
Java Certified Programmer – 1998
Microsoft Certified ATL Developer – 1998

Selected Links

GitHub: github.com/krizvi
Personal site: khalidrizvi.com
Agentic AI passion: metismesh.com/
LinkedIn: linkedin.com/in/khalidrizvi
Medium: medium.com/@krizvi