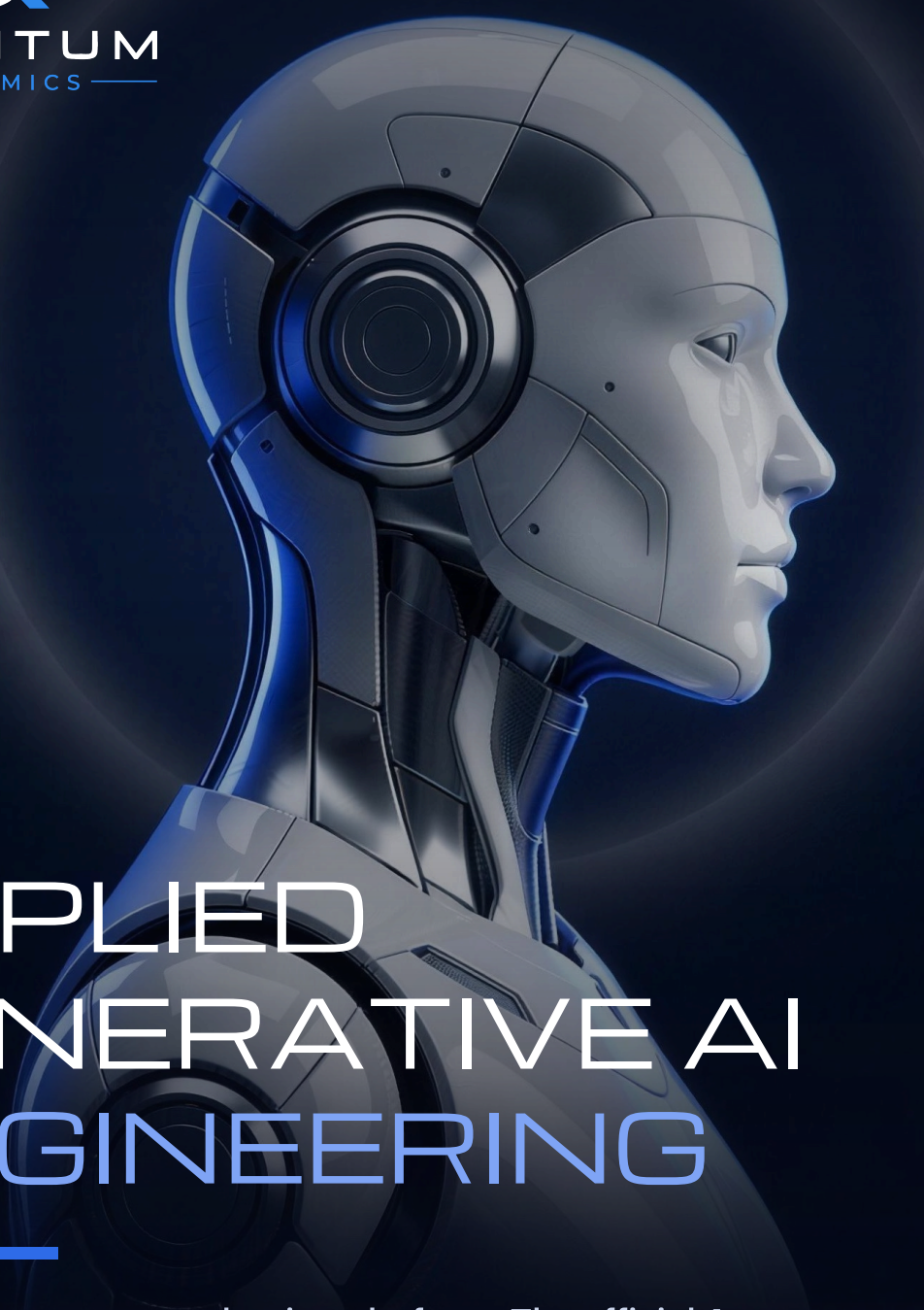




APPLIED GENERATIVE AI ENGINEERING

From first prompt to production platform. The official 1-year program.



12

MONTHS



16

TRAINING
WEEKS



120

INSTRUCTOR-
LED HOURS



2 MO

PROJECTS



6 MO

INTERNSHIP

TABLE OF CONTENTS

01	Program Overview	03
02	Who This Program Is For	03
03	Learning Progression	04
04	The One-Year Journey	05
05	Seven-Module Architecture	06
06	Detailed 16-Week Curriculum	07
07	Project Journey	23
08	Enterprise Capstone Experience	25
09	Industry Projects & Internship	26
10	Professional Capabilities Developed	27
11	Program Outcomes	28
12	Career Alignment	28
13	Prerequisites	29
14	Delivery Approach	30
15	Curriculum Governance	30

01 PROGRAM OVERVIEW

Applied Generative AI Engineering Program is a one-year generative AI engineering journey: sixteen weeks of instructor-led training built around seven progressive capability modules, a two-month real industry project phase, and a six-month internship for learners who meet the program's completion criteria. It carries learners from an accurate, no-code understanding of how generative AI actually works through to designing, evaluating, securing, and deploying production-grade multi-agent systems on AWS enterprise cloud infrastructure.

Rather than a tour of trending tools, the program follows one deliberately sequenced difficulty curve: every week is strictly built on the one before it, so beginners are never dropped into advanced material before its prerequisite has been established, while experienced professionals still reach genuine enterprise architecture and agentic-systems work within the first two months. Three progressive applied projects and one culminating enterprise capstone give every learner a demonstrable portfolio of production-oriented work by the time they finish.

After the training phase, learners spend months five and six on industry projects drawn from Quantum's client and industry engagements, building in teams against real requirements and deadlines, with mentor reviews and a final demo. Learners who complete the first six months and meet the program's attendance and assessment requirements then move into a six-month internship, finishing the year with genuine production experience.

02 WHO THIS PROGRAM IS FOR

- Freshers and recent graduates starting a technical career in AI
- Backend, data, and cloud engineers formalising generative AI engineering skill
- Working professionals transitioning into applied AI roles
- Intermediate AI practitioners who want genuine production depth beyond prompting
- Experienced technical leads who need enterprise architecture and AI safety fluency

03 LEARNING PROGRESSION

The sixteen-week training phase moves through three stages, in the same order. The only thing that varies is how much of it feels new.



FOUNDATION

WEEKS 1-4

No prior generative AI or programming experience is assumed at the start. Learners first build a no-code understanding of how generative AI systems work, then progressively develop the Python foundations required specifically for GenAI application engineering, from core programming concepts through APIs, asynchronous execution, and production-oriented backend services. This foundation supports every advanced engineering stage that follows.



APPLIED ENGINEERING

WEEKS 5-12

The core engineering build-out: cloud infrastructure, enterprise knowledge retrieval, rigorous evaluation, layered safety architecture, and progressively capable agentic systems.



ENTERPRISE & PRODUCTION

WEEKS 13-16

Production discipline and integration: deliberate context engineering, monitored deployment, and a two-week capstone that ties every capability into one enterprise-grade platform.

04 THE ONE-YEAR JOURNEY

MONTH 1

WEEKS 1-4

Foundations & Engineering Fluency

The program opens with no prerequisites assumed. Learners build an accurate mental model of generative AI, develop professional prompt and reasoning skill, and translate that fluency into a concurrent, fault-tolerant backend service: the shared starting line for every learner in the cohort, regardless of background.

MONTH 2

WEEKS 5-8

Cloud Infrastructure & Knowledge Engineering

Learners provision a production AWS cloud platform and build the knowledge-engineering layer that lets an AI system reason over an organisation's own documents: turning raw files into a distributed, hybrid knowledge base, and proving its quality against measurable criteria rather than intuition.

MONTH 3

WEEKS 9-12

Safety, Autonomy & Multi-Agent Systems

Month Three shifts from static systems to autonomous ones. Learners first design a layered safety architecture, then build progressively more capable agents: from a first tool-using agent, through a full middleware and memory stack, to a standards-based multi-agent system that can safely delegate work and validate the results.

MONTH 4

WEEKS 13-16

Optimisation, Production & Capstone Mastery

The final month is about discipline under production conditions: engineering the context window deliberately, deploying with monitoring and quality gates, and then spending two full weeks integrating every prior month's work into one coherent, enterprise-grade platform that learners design, review, and defend.

MONTHS 5-6

PROJECT PHASE

Real Industry Projects

Learners move from guided builds to projects drawn from Quantum's client and industry engagements: teams build against real requirements and deadlines, with mentor reviews and a final demo.

MONTHS 7-12

INTERNSHIP

Six-Month Internship

Learners who complete the first six months and meet the program's attendance and assessment requirements move into a six-month internship in a production environment.

05 SEVEN-MODULE ARCHITECTURE

1

Model Literacy & Instruction Engineering

WEEKS 1-3

Building an accurate mental model of generative AI, then developing professional-grade skill in structuring and reasoning with prompts.

2

High-Concurrency Application Engineering

WEEK 4

Backend engineering fluency, from core Python to asynchronous, fault-tolerant service design, for AI-facing applications.

3

Cloud-Native AI Infrastructure

WEEK 5

Provisioning and securing the AWS cloud platform that every later pipeline in the program deploys onto.

4

Enterprise Knowledge Engineering & Retrieval Assurance

WEEKS 6-8

Turning unstructured enterprise documents into a distributed, hybrid-searchable knowledge base, and proving its quality with measurement.

5

Responsible AI & Guardrail Architecture

WEEK 9

Designing defence-in-depth safety across input, output, and action layers.

6

Agentic Orchestration & Memory Systems

WEEKS 10-12

From a first tool-using agent, to a layered production middleware stack, to standards-based multi-agent interoperability.

7

Context Optimisation & Production Intelligence Operations

WEEKS 13-14

Engineering the context window deliberately, then deploying and operating the system in production.

C

Capstone Integration Experience

WEEKS 15-16

Bringing every module together into one presentable, defensible enterprise platform.

HOW GENERATIVE AI ACTUALLY WORKS



Establish an accurate, non-technical understanding of how generative AI systems actually behave: the shared vocabulary the rest of the program builds on. No programming background required.

KEY LEARNING AREAS

- 01 Probabilistic generation versus deterministic software logic
- 02 How a model predicts and completes patterns from context
- 03 Tokens and context windows, and the practical limits of what a model can see
- 04 The current foundation-model landscape: open versus closed, frontier versus efficient
- 05 Where generative AI genuinely helps, and where it does not
- 06 Hallucination and non-determinism: reading model output with informed scepticism



APPLIED EXPERIENCE

A guided, no-code exploration comparing how different prompts and models respond to the same real-world business scenario.



PROFESSIONAL OUTCOME

Learners can explain, in plain business language, what a generative AI system is doing under the hood, and can identify appropriate versus inappropriate use cases.

THE ARCHITECTURE OF AN EFFECTIVE PROMPT



Learn the structural mechanics that separate a reliable, parseable prompt from a fragile one, before any advanced reasoning technique is introduced.

KEY LEARNING AREAS

- 01** Prompt anatomy: system instructions, user turns, assistant priming, role and persona design
- 02** Zero-shot and few-shot prompting with curated examples
- 03** Structured output control for downstream parsing (JSON, XML, markdown)
- 04** The COSTAR framework for consistently well-formed prompts
- 05** Core prompt patterns: extraction, classification, transformation, generation, decomposition
- 06** Iterative refinement: testing, diagnosing, and rewriting



APPLIED EXPERIENCE

A guided build comparing a plain-instruction prompt against a professionally structured version of the same task, benchmarked for accuracy and reliability.



PROFESSIONAL OUTCOME

Learners can design prompts that produce consistent, machine-parseable output rather than one-off lucky results.

ENGINEERING MACHINE REASONING



Move beyond prompt structure into the reasoning techniques professionals use to make multi-step, genuinely hard problems reliable, and learn to manage prompts as governed production assets.

KEY LEARNING AREAS

- 01 Chain-of-Thought reasoning, and when it helps versus hurts
- 02 Tree-of-Thought exploration and backtracking for planning-style problems
- 03 Self-Consistency sampling to reduce hallucination on factual tasks
- 04 Least-to-Most decomposition for genuinely hard, multi-step problems
- 05 Self-Refine critique-and-improve loops
- 06 Versioning, A/B testing, and lifecycle-managing prompts as production assets



APPLIED EXPERIENCE

An engineering challenge benchmarking multiple reasoning strategies against each other on a genuinely multi-step task, then registering the winning approach as a governed, versioned asset.



PROFESSIONAL OUTCOME

Learners can select and justify the right reasoning technique for a given task, balancing accuracy against added latency and cost.

BUILDING CONCURRENT AI SERVICES



Build the Python foundations required specifically for GenAI application engineering, progressing from core programming and object-oriented design through APIs, asynchronous execution, and fault-tolerant service development, culminating in a concurrent, production-oriented AI service.

KEY LEARNING AREAS

- 01** Python foundations for GenAI engineering: variables, functions, control flow, collections, and reusable program structure
- 02** Object-oriented Python for maintainable service design
- 03** Core data structures for AI pipelines
- 04** Resilient error handling and file I/O for unpredictable model responses
- 05** Relational data access patterns and connection management
- 06** High-performance API design for AI-facing backends
- 07** Asynchronous programming: event loops, coroutines, parallel tool execution, timeouts and fallback handling



APPLIED EXPERIENCE

A guided build of an asynchronous API service that orchestrates multiple concurrent operations without blocking, complete with fallback handling under simulated failure.



PROFESSIONAL OUTCOME

Learners can build and reason about concurrent, fault-tolerant backend services suitable for production AI workloads.

PROVISIONING THE CLOUD AI PLATFORM



Provision the AWS building blocks that every later pipeline in the program will deploy onto, from storage and networking through to managed AI infrastructure.

KEY LEARNING AREAS

- 01** Durable object storage for unstructured data at scale
- 02** Core networking: subnets, routing, and security boundaries
- 03** Serverless container hosting for AI workloads
- 04** Traffic distribution and public API exposure
- 05** Managed foundation-model and agent-runtime services
- 06** Event-driven serverless compute, durable relational state, and least-privilege access design



APPLIED EXPERIENCE

A guided build provisioning a minimal, complete cloud stack and exposing a working public endpoint.



PROFESSIONAL OUTCOME

Learners can independently provision and secure an AWS environment capable of hosting a production AI workload.

TURNING DOCUMENTS INTO MACHINE-READABLE KNOWLEDGE



Build the document-understanding layer that turns messy real-world files into clean, structurally accurate, retrieval-ready text: the foundation every downstream retrieval system depends on.

KEY LEARNING AREAS

- 01 Layout-aware parsing of headers, tables, images, and code blocks
- 02 Why naive, fixed-width chunking silently corrupts complex documents
- 03 Structure-aware semantic chunking strategies
- 04 Chunk enrichment: PII detection and redaction before data leaves the pipeline
- 05 Named-entity and key-phrase extraction to support hybrid search
- 06 Designing stable, traceable chunk identifiers



APPLIED EXPERIENCE

A technical investigation processing a batch of real, messily formatted documents end to end, and comparing the result against a naive baseline.



PROFESSIONAL OUTCOME

Learners can design a document-ingestion layer that preserves meaning and structure instead of silently corrupting it.

HYBRID RETRIEVAL AT SCALE



Turn enriched content into a searchable knowledge base that combines vector similarity with graph relationships, running on distributed infrastructure.

KEY LEARNING AREAS

- 01 Embedding model selection and cross-provider compatibility
- 02 Vector storage and indexing strategy trade-offs
- 03 Graph modelling for relationship-aware retrieval
- 04 Hybrid retrieval combining vector, graph, and keyword signals
- 05 Distributed, idempotent processing across asynchronous workers
- 06 Retrieval, re-ranking, and context assembly patterns



APPLIED EXPERIENCE

A guided build turning a document collection into a distributed, hybrid-searchable knowledge base, then issuing a query that only hybrid retrieval can answer correctly.



PROFESSIONAL OUTCOME

Learners can design and operate a retrieval system that goes beyond simple vector lookup to relationship-aware, production-scale search.

MEASURING WHAT RETRIEVAL ACTUALLY DELIVERS



Replace guesswork with measurement: build the evaluation layer that turns retrieval and generation quality into a tracked engineering metric.

KEY LEARNING AREAS

- 01 LLM-judged quality signals: faithfulness, context relevance, answer relevance
- 02 Deterministic retrieval metrics and ranking-aware scoring
- 03 Experiment logging and visual run comparison
- 04 Building a golden evaluation dataset
- 05 Wiring regression checks into the development workflow



APPLIED EXPERIENCE

An applied project building a golden evaluation set for a real pipeline and configuring a regression check that catches a quality drop before it ships.



PROFESSIONAL OUTCOME

Learners can prove, with numbers, whether a RAG system is actually improving, rather than assuming it is.

ENGINEERING DEFENCE-IN-DEPTH FOR AI SYSTEMS



Design a layered safety architecture so that no single point of failure can produce an unsafe response.

KEY LEARNING AREAS

- 01 Deterministic input-layer checks: injection detection, PII handling, domain and toxicity screening
- 02 Output-layer evaluation: faithfulness judging, contradiction detection, confidence-aware disclaimers, safe fallback behaviour
- 03 Action-layer limits: retry and call-count ceilings, execution validation, read-only enforcement
- 04 Managed cloud guardrail services as a complementary layer
- 05 Why safety-critical logic belongs in deterministic code wherever possible



APPLIED EXPERIENCE

A design exercise implementing all three guardrail layers around an existing pipeline and testing them against a bank of adversarial inputs.



PROFESSIONAL OUTCOME

Learners can design a layered safety architecture and demonstrate, with evidence, exactly what each layer catches.

DESIGNING AN AGENT'S FIRST TOOLS



Construct a first production-style agent: well-designed tools, safe parallel execution, and a human checkpoint before anything destructive happens.

KEY LEARNING AREAS

- 01 Agent foundations: composing a model, tools, and orchestration logic
- 02 Parallel tool execution within a single reasoning step
- 03 Tool design principles: each tool owning its own logic and fallback behaviour
- 04 Sanitising tool output before it re-enters the model's context
- 05 Human-in-the-loop checkpoints for sensitive actions
- 06 An introduction to structural context separation as a security boundary



APPLIED EXPERIENCE

A guided build of a multi-tool agent with a human-approval checkpoint in front of a sensitive action.



PROFESSIONAL OUTCOME

Learners can build a tool-using agent that fails safely and pauses for approval where it should.

MIDDLEWARE, CACHING & MEMORY FOR AGENTS



Layer a full production middleware stack around the Week 10 agent, adding caching and memory without compromising safety.

KEY LEARNING AREAS

- 01** Middleware composition: tracing, domain-specific filtering, caching, memory, summarisation, and guardrail layers
- 02** Separating generic infrastructure responsibilities from domain-specific agent logic
- 03** Semantic caching with similarity thresholds calibrated to risk
- 04** Episodic memory: what is worth remembering, and how it re-enters context
- 05** Context compression once a token budget is exceeded



APPLIED EXPERIENCE

An architecture challenge assembling a layered middleware stack in front of an agent and verifying that caching, memory, and compression behave correctly under load.



PROFESSIONAL OUTCOME

Learners can design a middleware stack that makes an agent faster, cheaper, and more consistent without weakening its safety posture.

INTEROPERABILITY: CONNECTING AGENTS AND TOOLS AT SCALE



Connect agents to standardised external tools and to each other, and assemble a validated multi-agent workflow.

KEY LEARNING AREAS

- 01** The Model Context Protocol (MCP): a standardised approach to tool and data access
- 02** The Agent-to-Agent (A2A) protocol: agent discovery and cross-framework task delegation
- 03** Supervisor-and-specialist multi-agent patterns
- 04** Validating another agent's output rather than trusting it blindly
- 05** Designing clean boundaries of responsibility between agents



APPLIED EXPERIENCE

An integration challenge connecting a specialist agent to a planner agent through standardised protocols, including a validation step before any delegated output is used.



PROFESSIONAL OUTCOME

Learners can design a multi-agent system where components interoperate safely across frameworks rather than through brittle, custom glue code.

ENGINEERING THE CONTEXT WINDOW



Treat every token the model sees as a deliberate design decision, and engineer a context window that stays accurate and affordable over long sessions.

KEY LEARNING AREAS

- 01 What belongs in context: instructions, history, memory, retrieved data, and the live question
- 02 Why too little context causes hallucination and too much adds cost and confusion
- 03 Structural separation of instructions, data, and user input as both a design and a security discipline
- 04 Per-section token budgeting and oldest-first truncation
- 05 Context patterns for long-running agent tasks: mid-task injection and compression



APPLIED EXPERIENCE

A technical investigation instrumenting a multi-agent workflow to track token spend per context section, and verifying that truncation behaves correctly under a long session.



PROFESSIONAL OUTCOME

Learners can engineer context windows deliberately instead of letting them grow unmanaged.

SHIPPING AND OPERATING AI SYSTEMS IN PRODUCTION



Deploy a complete agent system to production infrastructure with monitoring and a quality gate, and close the loop between production and evaluation.

KEY LEARNING AREAS

- 01 Production deployment patterns for agent systems
- 02 Wiring regression checks into a deployment gate that blocks unsafe releases
- 03 Feedback loops that route flagged production interactions back into evaluation
- 04 Monitoring latency, cost, and quality drift over time
- 05 What a strong final architecture review looks like



APPLIED EXPERIENCE

A production scenario deploying a monitored agent endpoint and confirming that a regression gate correctly blocks a deliberately degraded release.



PROFESSIONAL OUTCOME

Learners can take a system from working prototype to monitored, gated production deployment.

CAPSTONE ARCHITECTURE & SAFETY REVIEW



Design and defend the end-to-end architecture and safety plan for the capstone platform before finalising the build, catching integration gaps while there is still time to fix them.

KEY LEARNING AREAS

- 01** A structured architecture-review checklist spanning the full program
- 02** Common integration pitfalls: mismatched schemas, inconsistent guardrail coverage, ignored multi-agent handoffs
- 03** Running a staged review process with checkpoints rather than open-ended work time
- 04** Presenting technical architecture to a non-technical stakeholder audience
- 05** A full safety-architecture review across every capstone subsystem



APPLIED EXPERIENCE

A staged review process: present the end-to-end architecture for structured feedback, revise it, then walk through the safety design across every subsystem and revise again.



PROFESSIONAL OUTCOME

Learners can identify and close integration and safety gaps before they become expensive to fix.

CAPSTONE DELIVERY & STAKEHOLDER PRESENTATION



Complete final integration, incorporate review feedback, and deliver a polished stakeholder presentation of the finished platform.

KEY LEARNING AREAS

- 01 Incorporating structured feedback efficiently under a deadline
- 02 Running a full peer walkthrough and cross-review
- 03 Presenting technical architecture to an executive-level audience
- 04 Final submission structure and evidence packaging
- 05 Mapping the completed body of work to industry roles and next steps



APPLIED EXPERIENCE

A final peer walkthrough and instructor review cycle, followed by a timed oral architecture presentation and Q&A.



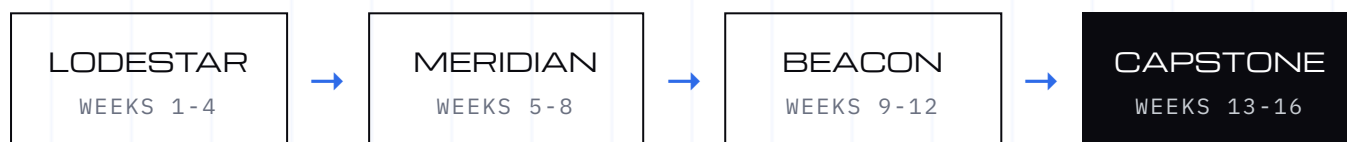
PROFESSIONAL OUTCOME

Learners leave with a defensible, presentable capstone platform and the ability to communicate its architecture to a non-technical audience.

07 PROJECT JOURNEY

The training phase contains four major builds, one per month: Project Lodestar, Project Meridian, Project Beacon, and the Enterprise Capstone. Each consumes that stage's full technical capability and builds on the one before it, so the portfolio grows in difficulty exactly the way the weeks do. After the capstone, the project track continues into months five and six with industry projects drawn from Quantum's client and industry engagements.

HOW THE PROJECTS STACK



PROJECT LODESTAR

MONTH 1 APPLIED PROJECT · WEEKS 1-4

Reasoning-Grade Advisory Service

Ship a first customer-facing AI capability: an API service that reliably handles difficult, multi-step extraction requests, backed by structured, versioned prompts and safe concurrent execution.

CAPABILITIES APPLIED

- Prompt architecture and reasoning-technique selection
- Asynchronous service design and concurrent, fault-tolerant orchestration
- Production-style API endpoint and deployment-ready service design

OUTCOME · A production-style, reasoning-capable microservice, plus a written design rationale explaining the prompting and concurrency choices made.



PROJECT MERIDIAN

MONTH 2 APPLIED PROJECT • WEEKS 5-8

Enterprise Knowledge Grounding Pipeline

Turn a large, messy library of enterprise documents into a distributed, measured knowledge-retrieval system that a downstream AI application can trust.

CAPABILITIES APPLIED

- Cloud infrastructure provisioning
- Document understanding and semantic chunking
- Hybrid vector-and-graph retrieval design
- Rigorous evaluation and regression testing

OUTCOME • A working, distributed retrieval pipeline with a documented evaluation report proving its quality against a golden dataset.



PROJECT BEACON

MONTH 3 APPLIED PROJECT • WEEKS 9-12

Autonomous Analytics Workforce

Give non-technical business users safe, natural-language access to a live operational database through a coordinated team of specialised, guardrailed agents.

CAPABILITIES APPLIED

- Defence-in-depth safety architecture
- Multi-agent design with validated handoffs
- Middleware, caching, and memory engineering
- Standardised agent interoperability protocols

OUTCOME • A deployed multi-agent system with a measured accuracy benchmark and an architecture diagram explaining how each agent's output is validated.

08 ENTERPRISE CAPSTONE EXPERIENCE



ENTERPRISE INTELLIGENCE PLATFORM CAPSTONE

Step into the role of AI platform architect for a fictional enterprise rolling out several interconnected generative AI systems at once, and design, review, and defend a single coherent production architecture that ties every prior month's capability together.

CAPABILITIES APPLIED

- End-to-end system architecture
- Knowledge grounding and retrieval design
- Evaluation framework design
- Safety architecture across multiple systems
- Multi-agent and context-engineering design
- Production deployment and monitoring planning

HOW THE TWO WEEKS RUN

The capstone runs across the final two weeks of the training phase: one week to design, present, and revise the architecture and safety plan under structured review, and one week to complete integration and deliver a stakeholder-ready oral presentation with a live Q&A.

PROFESSIONAL OUTCOME

A reviewed, presentation-ready enterprise architecture package, and the ability to defend technical design decisions to a non-technical executive audience.

09 INDUSTRY PROJECTS & INTERNSHIP

The classroom phase is only the first third of the year. From month five onwards, learners work the way professionals do: first on industry projects, then inside a six-month internship.

THE FULL YEAR AT A GLANCE

TRAINING

MONTHS 1-4 • 120 HRS

PROJECTS

MONTHS 5-6

INTERNSHIP

MONTHS 7-12



MONTHS 5-6 • REAL INDUSTRY PROJECTS

After the sixteen-week training phase, learners spend two months on projects drawn from Quantum's client and industry engagements. Teams build against real requirements and deadlines, go through mentor reviews, and close the phase with a final demo of working software.



MONTHS 7-12 • SIX-MONTH INTERNSHIP

Learners who complete the first six months and meet the program's attendance and assessment requirements move into a six-month internship, applying the full generative AI engineering stack in a production environment for the rest of the year.

By the end of month twelve, a learner has 120 hours of instructor-led training, four guided portfolio projects, two months of real industry project delivery, and six months of internship experience.

10 PROFESSIONAL CAPABILITIES DEVELOPED

Prompt & Reasoning Engineering

- Structuring prompts for reliable, machine-parseable output
- Selecting and justifying advanced reasoning strategies for multi-step tasks
- Managing prompts as versioned, testable production assets

Concurrent Application Engineering

- Designing asynchronous, high-performance APIs for AI workloads
- Building fault-tolerant orchestration with timeout and fallback handling

Cloud & Production Infrastructure

- Provisioning and securing a production-grade AI platform on enterprise cloud infrastructure (AWS)
- Designing least-privilege access and scalable service exposure

Enterprise Knowledge Engineering

- Layout-aware document processing and semantic chunking
- Hybrid vector-and-graph retrieval design
- PII-aware data handling

Quality & Evaluation Engineering

- Building golden datasets and regression-gated quality checks
- Applying deterministic and LLM-judged evaluation metrics

AI Safety Engineering

- Designing input, output, and action-layer guardrails
- Adversarial testing of safety architecture

Agentic Systems Engineering

- Tool design, middleware composition, semantic caching, and episodic memory
- Multi-agent orchestration with standardised interoperability protocols (MCP, A2A)

Context & Production Operations

- Deliberate context-window budgeting and compression
- Monitored deployment with regression-gated release pipelines

11 PROGRAM OUTCOMES

- Can design, build, evaluate, secure, and deploy production-grade generative AI systems, not just prototype with them
- Comfortable across the full stack: prompting, backend engineering, cloud infrastructure, retrieval, safety, and agentic orchestration
- Leaves with a portfolio of deployed, benchmarked projects plus one enterprise-grade capstone architecture
- Can communicate technical architecture decisions to a non-technical stakeholder audience
- Possesses a combined data-engineering and AI-engineering skillset sought by enterprise employers
- Finishes the year with industry project delivery and six months of internship experience

12 CAREER ALIGNMENT

Learners completing this program are prepared for engineering and architecture roles across the generative AI stack, and for further specialisation into AI safety or LLMOps. The program's emphasis on evaluation, safety, and production deployment, alongside a demonstrable project portfolio, is designed to support technical interview readiness at a system-design level.

TARGET ROLES

GenAI Engineer**AI Platform Engineer****Applied AI Architect****LLMOps / MLOps Engineer****AI Safety Engineer**

13 PREREQUISITES

✓ REQUIRED KNOWLEDGE

- Basic computer literacy and willingness to work with technical tools
- Basic problem-solving ability and willingness to practise through guided engineering exercises

☆ RECOMMENDED BACKGROUND

- Prior exposure to any programming or scripting language is helpful, but not required
- Basic familiarity with databases, APIs, or web applications is beneficial
- Basic awareness of cloud, containers, or command-line environments is useful

i HOW THE PROGRAM BUILDS TECHNICAL READINESS

The program begins with a no-code understanding of how Generative AI works, then progressively develops the Python skills required for GenAI application development: from core programming foundations through object-oriented design, APIs, asynchronous programming, concurrent execution, and production-oriented backend services.

⊗ NOT REQUIRED

- Prior Generative AI, RAG, or agent-development experience
- Prior Python or programming expertise (the program builds the Python foundations needed for GenAI engineering)
- Deep machine learning or mathematics background
- Existing cloud certification
- Production software-engineering experience

14 DELIVERY APPROACH

TRAINING PHASE • MONTHS 1-4

- 80 instructor-led sessions across 16 weeks (Monday to Friday), 1.5 hours of structured learning per day: 120 instructor-led hours in total
- Each session blends concept instruction, live demonstration, guided applied-engineering time, and review
- Individual and team-based applied work throughout

BEYOND THE CLASSROOM • MONTHS 5-12

- Months 5-6: industry projects drawn from Quantum's client and industry engagements, delivered in teams with mentor reviews and a final demo
- Months 7-12: a six-month internship for learners who complete the first six months and meet the program's attendance and assessment requirements

ASSESSMENT

- Progressive, project-based assessment culminating in a two-week capstone integration and stakeholder-presentation phase

15 CURRICULUM GOVERNANCE

This curriculum is reviewed and refreshed on a regular cycle to keep pace with the fast-moving generative AI tooling landscape, so that learners consistently train on current, industry-relevant engineering practice rather than a static snapshot of today's tools.

LET'S BUILD YOUR **AI** CAREER.



WEBSITE

quantumacademics.ai

EMAIL

hello@quantumacademics.ai

PHONE / WHATSAPP

8098 809 809



CAMPUS

Jubilee Hills, Hyderabad, Telangana 500033



BOOK A FREE DEMO

quantumacademics.ai/book-demo