Certified AI Administrator and Engineer (CAIAE) Exam Objectives

Introduction

When you pass the CAIAE-101 exam, you earn the CAIAE certification and validate your knowledge of AI administration and engineering. The CAIAE-101 exam tests your knowledge of the topics covered in this document. There are no prerequisite certifications required to sit for the CAIAE exam.

The CAIAE has the knowledge and skill set required to select, implement, manage, and decommission AI solutions in modern systems and networks. This professional has sufficient knowledge of AI concepts, architectures, tools, and ecosystems to plan, implement, monitor, and control AI solutions across the full AI system lifecycle. The individual understands the feature sets and capabilities of AI technologies, frameworks, and services, including how they are integrated, secured, and operated, at the depth needed to make sound engineering and administrative decisions. The CAIAE is not primarily a model developer or data scientist; rather, this professional focuses on implementing, configuring, and managing AI solutions built on existing models and platforms.

The exam is taken in the CWNP Learning Management System (LMS), and the purchase of the certification kit includes the e-learning material (covering all required knowledge), practice test, and final exam. The exam consists of 40 questions that must be answered within 100 minutes, requiring a score of 70% to earn the certification. Currently, the exam is offered only in English. While the exam has no verified prerequisites, it is recommended that the candidate have 1–3 years of experience in IT, networking, or systems administration.

The following table provides the breakdown of the exam as to the distribution of questions within each knowledge domain.

Knowledge Domain	Percentage
AI Concepts, Types, and Applications	15%
Planning AI Solutions	25%
Implementing AI Solutions	35%
Securing AI Solutions	25%

1.0 AI Concepts, Types, and Applications – 15%

1.1 Understand the definition, history, and taxonomy of artificial intelligence

- General definition of artificial intelligence; narrow AI, general AI, superintelligence
- Symbolic AI and classical approaches
    - Expert systems
    - Case-based reasoning (CBR)
    - Fuzzy logic
    - General/formal logic
    - Rules-based systems
- Algorithms
- Differences among paradigms and appropriate use cases for each

1.2 Explain machine learning, deep learning, and neural network architectures

- Machine learning paradigms: supervised, unsupervised, semi-supervised, self-supervised, reinforcement learning
- Common ML methods: clustering, dimensionality reduction, ensemble methods
- Neural network fundamentals
- Neural network architectures and their primary applications
    - Feedforward neural networks (FFN)
    - Convolutional neural networks (CNN)
    - Recurrent neural networks (RNN)
    - Transformers
    - Graph neural networks (GNN)
    - Modern variants: Mixture of Experts (MoE), state space models (Mamba), diffusion architectures
- Embeddings

1.3 Describe generative AI, large language models, and related model types

- Generative AI

- Large language models (LLMs)
- Small language models (SLMs)
- Foundation models and frontier models
- Fine-tuned models
- Multimodal models
- Embedding models
- Model formats: safetensors, GGUF, ONNX, PyTorch, TensorFlow; quantization (INT8, INT4, GPTQ, AWQ, GGUF/GGML) and distillation
- Context windows, tokenizers, and token-counting implications
- Open-source AI: Hugging Face Hub, Ollama model library, GitHub repositories; model licenses (Apache 2.0, MIT, OpenRAIL, Meta Llama Community License, GPL/AGPL, Creative Commons) and their operational implications

1.4 Explain agentic AI architectures and patterns

- Agentic AI: autonomous agents that perceive, reason, plan, and act using tools and memory over extended task horizons
- Core patterns
    - ReAct (Reason + Act)
    - Plan-and-execute
    - Multi-agent orchestration
- Tool/function calling
- Model Context Protocol (MCP)
- Agent-to-Agent (A2A) protocol
- Agent frameworks: LangChain Agents, LangGraph, LlamaIndex Agents, AutoGen, CrewAI, OpenAI Assistants API, etc.
- Memory types: in-context (conversation history), external (vector store, database), episodic, semantic

1.5 Identify AI applications, explainable AI, and trustworthy AI principles

- AI application domains: natural language processing (NLP), computer vision (CV), speech recognition/synthesis, recommendation systems, anomaly/fraud detection, predictive analytics, robotic process automation, agentic automation
- Explainable AI (XAI): LIME, SHAP (TreeSHAP, DeepSHAP, KernelSHAP), attention visualization, Grad-CAM, counterfactual explanations
- Trustworthy AI principles: fairness, accountability, transparency, explainability, robustness, privacy, safety
- Bias and fairness concepts
- Python and Python AI libraries used across AI roles
    - Numerical and data: NumPy, pandas, SciPy
    - Classical ML: scikit-learn
    - Deep learning: TensorFlow, Keras, PyTorch, JAX
    - LLM/NLP ecosystem: Hugging Face Transformers, Hugging Face Datasets, Hugging Face PEFT, LangChain, LlamaIndex, spaCy, NLTK
    - Local inference and serving: Ollama, vLLM, ONNX/ONNX Runtime, BentoML
    - Computer vision: OpenCV
    - UI and API serving: FastAPI, Gradio, Streamlit
    - MLOps: MLflow, Weights & Biases (W&B), DVC

2.0 Planning AI Solutions – 25%

2.1 Apply use case discovery and problem framing methods

- Stakeholder workshops for identifying and prioritizing candidate AI use cases by business value and feasibility
- Framing the AI problem: defining inputs, desired outputs, decision types, and required human oversight level (human-in-the-loop, human-on-the-loop, fully automated)
- Assessing whether a problem is AI-addressable vs. solvable by simpler rule-based automation

- Build vs. buy vs. subscribe decision framework: scoring capability fit, data control, customizability, vendor risk, time-to-value, and cost
- Concept of Operations (ConOps) for AI systems: operational environment, user roles, system boundaries, use scenarios
- PoC/pilot planning: scope definition, measurable success criteria, exit thresholds, feedback collection mechanisms
- AI project phase structure: ideation → PoC → pilot → production → steady-state → sunset

2.2 Define requirements for AI systems

- Functional requirements: supported use cases, input/output formats, inference behavior, API contracts
- Non-functional requirements
    - Performance: latency (p50/p95/p99 targets), throughput (requests/sec, tokens/sec), availability (uptime SLA), scalability, RTO/RPO
    - Quality: precision, recall, hallucination rate ceiling, calibration error bound
    - Explainability: which decisions require explanation, format
    - Cost: cost-per-inference ceiling, monthly GPU budget, token budget
    - Energy usage
- Privacy and security requirements: PII data types processed, retention limits, input sanitization, adversarial robustness
- SLA/SLO specification: availability, latency, accuracy, error budget policy, RTO/RPO
- Acceptance criteria: quantitative model performance thresholds, definition of "done" at each project gate

2.3 Plan data requirements and data architecture

- Data requirements specification: volume, variety, velocity, veracity, value (5 Vs); acceptable sources, formats, labeling standards
- Data readiness assessment: completeness, quality, licensing, access controls, PII/consent review

- Data labeling and annotation: annotation schemas, inter-annotator agreement, quality thresholds, active learning
- Data lineage and provenance: tracking data origins and transformations
- Train/validation/test split methodology: stratification, leakage prevention, temporal splits, group-based splits
- Synthetic data generation: LLM-based, GAN/VAE-based, rule-based synthesis
- Data governance planning: lineage documentation, retention schedules, classification (public/internal/confidential/regulated), data minimization
- Privacy-enhancing technologies: differential privacy, federated learning, secure enclaves

2.4 Select AI models and architecture patterns

- Model selection criteria: performance benchmarks, cost, licensing, context window, latency, hardware requirements
- Approach selection: foundation model + RAG, fine-tuning, in-context learning, build-from-scratch
- Fine-tuning approaches: supervised fine-tuning, RLHF, DPO, LoRA/QLoRA
- Architecture pattern selection: inference-as-a-service, RAG pipeline, agentic pipeline, streaming inference, edge AI
- Open vs. closed model trade-offs; hosted API vs. self-hosted; cloud managed services vs. on premises
- Model card review: intended use, limitations, training data, evaluation results, ethical considerations
- Vendor evaluation: security posture, data handling, compliance certifications, incident response capability, SLAs
- CRISP-DM methodology and mapping to enterprise project phases

2.5 Plan infrastructure, compute, storage, and network

- Compute sizing: GPU; VRAM estimation; tokens/sec benchmarks; concurrent user projections
- Inference optimization concepts: quantization, speculative decoding, continuous batching, KV-cache management, prefix caching, Flash Attention, tensor and pipeline parallelism
- Storage planning: object storage for model artifacts and training data; vector databases for embeddings; feature stores for ML features; model registries for versioned artifacts; data lakes/lakehouses
- Storage tiering: hot, warm, cold, archive; lifecycle policies
- Network planning: bandwidth budgets for model downloads and training data ingestion; latency budgets; egress cost estimation; private connectivity for cloud AI services
- GPU cluster interconnect awareness
- Cloud AI infrastructure options

2.6 Plan risk, cost, and governance

- Risk identification and classification: technical, ethical, legal, reputational
- Risk assessment methods: FMEA adapted for AI, risk matrix, risk register; risk treatment
- Standards and regulatory framework awareness
    - NIST AI Risk Management Framework
    - ISO/IEC 42001; ISO/IEC 23894
    - EU AI Act: risk tiers; conformity assessment obligations
    - Sector regulations: HIPAA for healthcare AI, FINRA/SEC for financial AI, GDPR automated decision-making
- Total Cost of Ownership (TCO) modeling: infrastructure, licensing, labor, data, energy; GPU cost models; token cost analysis; build vs. buy

- FinOps planning: chargeback/showback models; budget alerting; cost-per-inference targets
- AI ethics and responsible AI planning: ethics review triggers, human oversight requirements by risk tier, dual-use risk assessment

3.0 Implementing AI Solutions – 35%

3.1 Provision and configure AI compute infrastructure

- Provisioning GPU instances and clusters: cloud GPU instances, on-premises GPU servers, containerized GPU workloads
- Kubernetes for AI: GPU node pools, device plugins (NVIDIA device plugin), KEDA autoscaling, model deployment manifests
- Container fundamentals: Docker images for AI workloads, Dockerfiles for ML applications, dependency bundling, GPU-enabled containers
- Infrastructure as Code (IaC): Terraform, Helm charts for reproducible AI infrastructure
- Environment separation: dev, test, staging, production
- High-performance interconnects

3.2 Install and configure AI inference runtimes and serving frameworks

- Local inference servers: Ollama, llama.cpp
- Production inference servers: vLLM, NVIDIA Triton Inference Server (multi-framework serving, dynamic batching, ensemble pipelines, model repository layout), TorchServe, TGI (Text Generation Inference), KServe
- Inference optimization configuration: quantization (GPTQ, AWQ, GGUF), continuous batching parameters, KV-cache size, batch size, concurrency settings, prefix caching
- Model serving with BentoML and Ray Serve
- OpenAI-compatible API conventions
- API/application server configuration: FastAPI, Flask, gRPC services

3.3 Deploy and manage models throughout their lifecycle

- Model registries
- Downloading, validating, and loading models
- Model artifact integrity
- Fine-tuning deployment
- CI/CD for models
- Model lifecycle stage management: promote, demote, archive; deprecation headers and sunset procedures; notifying downstream consumers

3.4 Integrate data stores and databases for AI workloads

- Relational databases (PostgreSQL, MySQL, SQL Server)
- NoSQL document stores (MongoDB, Couchbase, Firestore)
- Key-value stores (Redis, Memcached)
- Wide-column stores (Apache Cassandra, ScyllaDB)
- Graph databases (Neo4j, Amazon Neptune, ArangoDB)
- Time-series databases (InfluxDB, TimescaleDB, Prometheus, VictoriaMetrics)
- Object storage (Amazon S3, Azure Blob, GCS, MinIO)
- Feature stores
- Data formats

3.5 Build and operate RAG pipelines and vector databases

- RAG architecture: retriever + generator; naive RAG vs. advanced RAG vs. modular RAG
- Document ingestion: loaders, parsers, metadata extraction, deduplication
- Chunking strategies
- Embedding generation

- Vector database administration: Pinecone, Weaviate, Milvus, Qdrant, ChromaDB, FAISS, pgvector
- Vector indexing: Flat, IVF, HNSW, PQ/IVF+PQ, DiskANN; recall vs. latency vs. memory trade-offs; capacity planning
- Hybrid search: BM25/SPLADE (sparse) + dense ANN + Reciprocal Rank Fusion (RRF)
- RAG evaluation: faithfulness, answer relevancy, context precision, context recall

3.6 Configure agentic AI systems

- Tool/function calling
- MCP server setup
- Multi-agent pipelines
- Guardrails for agents
- Agent memory configuration
- Agentic observability

3.7 Configure network protocols and API infrastructure for AI

- Protocol selection for AI services: REST/HTTP, gRPC, WebSockets, SSE, MCP
- HTTP/2 multiplexing, HTTP/3/QUIC (0-RTT, UDP-based, low-latency), chunked transfer encoding
- TLS 1.3 configuration; mTLS for AI microservice-to-microservice authentication; certificate management
- API gateway and reverse proxy configuration
- Load balancing: Layer 4 (TCP) vs. Layer 7 (HTTP)
- Service mesh: Istio/Linkerd deployment; mTLS enforcement
- Message queues and streaming
- DNS configuration

- Private connectivity: AWS PrivateLink, GCP Private Service Connect, Azure Private Endpoint for cloud-managed AI services; VPN (IPSec/IKEv2), Direct Connect, ExpressRoute for hybrid connectivity

3.8 Apply prompting strategies and code AI solutions with AI assistance

- Prompting strategies
    - Zero-shot
    - Few-shot
    - Chain-of-thought (CoT)
    - Tree-of-thought (ToT)
    - ReAct
    - Self-consistency
    - Role/persona prompting
    - Structured output prompting
    - Retrieval-augmented prompting
    - Prompt chaining
- Prompt engineering components: system prompts vs. user prompts vs. assistant turns; prompt templates (Jinja2, f-strings, LangChain `PromptTemplate`); parameterized reusable prompts; negative prompting
- Sampling parameter effects: temperature, top-p, top-k, repetition penalty; impact on output diversity and quality
- Prompt versioning: tracking prompt changes alongside model versions; regression testing on prompt changes
- Prompt injection awareness: direct and indirect injection; mitigation through input sanitization and separation of data and instructions
- Coding with AI: AI code assistants; generating and modifying scripts using LLM prompts; AI-assisted code review for bugs, refactors, and security issues; verifying AI-generated code; limitations
- Python fundamentals for AI administrators: reading/modifying existing scripts; virtual environments (‘venv’, ‘conda’), pip, requirements.txt, pyproject.toml; async Python;

Jupyter notebooks; environment variables and '.env' files; exception handling and logging; argument parsing

3.9 Monitor, manage, and verify AI solutions

- Observability: metrics (Prometheus/Grafana), distributed tracing (OpenTelemetry, Jaeger, Tempo), structured logging (ELK stack, Grafana Loki); LLM-specific tracing (LangSmith, Arize, WhyLogs)
- AI-specific KPIs: token usage, inference latency, TTFT (time-to-first-token), inter-token latency, throughput, error rate, cache hit rate, queue depth
- Output verification and evaluation metrics
    - Text generation
    - Benchmarks
    - LLM-as-judge
    - Human-in-the-loop review
    - Hallucination detection
    - Regression testing for outputs
- Drift detection and monitoring
- Capacity management and scaling
- FinOps for AI
- A/B testing and deployment strategies
- Retraining and maintenance
- Decommissioning AI systems

4.0 Securing AI Solutions – 25%

4.1 Identify AI threats, attacks, and vulnerabilities

- OWASP Top 10 for LLMs
- OWASP Top 10 for Machine Learning
- OWASP Top 10 for Agentic Applications

- MITRE ATLAS
- Specific attack techniques
    - Prompt injection variants
    - Jailbreak taxonomies
    - Adversarial examples
    - Training data attacks
    - Model inversion and verbatim memorization
    - Membership inference
    - Model extraction/stealing
    - Multi-modal injection
    - Insecure deserialization
    - Side-channel attacks
    - Vector database attacks
    - Hallucinations as security risk

4.2 Implement AI security controls and guardrails

- Input validation and sanitization
- Output filtering and moderation
- Guardrails frameworks
- Prompt firewalls
- Rate limiting and quota management
- Sandboxing AI tool and code execution
- Red-teaming AI systems

4.3 Secure the AI model supply chain and infrastructure

- Model supply chain security
- Model scanning
- Safetensors preference over pickle-based formats; ONNX operator allow-listing
- AI BOM (AI Bill of Materials)

- Private model registries with access control and scanning policies; Hugging Face gated model access
- Identity and access management for AI services: OAuth 2.0, JWT validation, API key management (rotation, scoping, revocation), RBAC/ABAC for model access tiers
- Secrets management
- Network security for AI: egress filtering, WAF rules for LLM API endpoints, DDoS protection, Zero Trust Network Access (ZTNA) for AI API endpoints, VPC/private endpoint configuration for cloud AI services, segmentation of training/inference/data/management planes
- mTLS between AI microservices
- Confidential computing for AI
- Watermarking and content provenance

4.4 Apply AI privacy and data protection

- PII detection and redaction
- Data minimization
- Differential privacy
- Federated learning
- Machine unlearning and right to erasure
- Privacy-preserving inference
- Data Protection Impact Assessments (DPIAs)
- Consent management
- Data geo-residency
- Encryption for AI data stores

4.5 Implement AI governance, compliance, and incident response

- NIST AI RMF (AI RMF 1.0) governance functions
    - GOVERN
    - MAP
    - MEASURE
    - MANAGE
- ISO/IEC 42001
- ISO/IEC 23894
- EU AI Act
- Sector-specific compliance
- AI governance structures and policies: AI acceptable use policy; model use policy; employee use of public AI tools policy; shadow AI detection and management; vendor AI usage requirements
- AI inventory and asset register: cataloging all AI models, APIs, and applications; inventory attributes; continuous inventory maintenance
- Algorithmic impact assessments (AIAs): scope; triggers; assessment methodology; documentation and disclosure requirements
- Bias and fairness audits
- Copyright and IP risk
- AI logging and auditability
- AI incident response
- Secure AI decommissioning