Certified AI Administrator and Engineer (CAIAE-101): Exam Preparation

Course Introduction:

This course prepares professionals for the CWNP CAIAE-101 certification exam. It is organized around the four official knowledge domains and their exam weightings: AI Concepts, Types, and Applications (15%); Planning AI Solutions (25%); Implementing AI Solutions (35%); and Securing AI Solutions (25%). The Certified AI Administrator and Engineer has the knowledge and skill set required to select, implement, manage, and decommission AI solutions in modern systems and networks. This professional focuses on implementing, configuring, and managing AI solutions built on existing models and platforms — not on model development or data science. The exam consists of 40 multiple-choice questions to be answered in 100 minutes, with a passing score of 70%. No prerequisites are required; 1–3 years of IT, networking, or systems administration experience is recommended.

Exam Details:

- **Exam Code:** CAIAE-101
- **Format:** 40 multiple-choice questions (single correct answer)
- **Duration:** 100 minutes
- **Passing Score:** 70%
- **Language:** English
- **Certification Validity:** 5 years
- **Prerequisites:** None required (1–3 years IT / networking / systems admin recommended)
- **Delivery:** Web browser within CWNP Learning Management System (LMS)

Module 1: AI Concepts, Types, and Applications (15%)

1.1 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 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: Feedforward (FFN), Convolutional (CNN), Recurrent (RNN), Transformers, Graph Neural Networks (GNN)
- Modern variants: Mixture of Experts (MoE), state space models (Mamba), diffusion architectures
- Embeddings

1.3 Generative AI, Large Language Models, and Related Model Types

- Generative AI
- Large language models (LLMs) and Small language models (SLMs)
- Foundation models, 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 ecosystem: 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 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
- Memory types: in-context (conversation history), external (vector store, database), episodic, semantic

1.5 AI Applications, Explainable AI, and Trustworthy AI Principles

- AI application domains: 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 — Numerical/data (NumPy, pandas, SciPy), Classical ML (scikit-learn), Deep learning (TensorFlow, Keras, PyTorch, JAX), LLM/NLP (Hugging Face Transformers/Datasets/PEFT, LangChain, LlamaIndex, spaCy, NLTK), Inference/serving

(Ollama, vLLM, ONNX Runtime, BentoML), Computer vision (OpenCV), UI/API (FastAPI, Gradio, Streamlit), MLOps (MLflow, Weights & Biases, DVC)

Module 2: Planning AI Solutions (25%)

2.1 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, 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: capability fit, data control, customizability, vendor risk, time-to-value, cost
- Concept of Operations (ConOps) for AI systems: operational environment, user roles, system boundaries, use scenarios
- PoC/pilot planning: scope, 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 — Performance: latency (p50/p95/p99), throughput (req/sec, tokens/sec), availability (uptime SLA), scalability, RTO/RPO
- Non-functional — Quality: precision, recall, hallucination rate ceiling, calibration error bound
- Non-functional — Explainability, Cost (cost-per-inference ceiling, GPU/token budgets), Energy usage
- Privacy and security requirements: PII types processed, retention limits, input sanitization, adversarial robustness
- SLA/SLO specification: availability, latency, accuracy, error budget policy, RTO/RPO
- Acceptance criteria and 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); sources, formats, labeling standards
- Data readiness assessment: completeness, quality, licensing, access controls, PII/consent review
- Data labeling and 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: 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/pipeline parallelism
- Storage planning: object storage for artifacts/training data; vector databases for embeddings; feature stores; model registries; data lakes/lakehouses
- Storage tiering: hot, warm, cold, archive; lifecycle policies
- Network planning: bandwidth budgets, latency budgets, egress cost estimation, private connectivity
- 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), sector regulations (HIPAA, FINRA/SEC, 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

Module 3: 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), 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 and 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; 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, Structured output, Retrieval-augmented, Prompt chaining
- Prompt engineering components: system prompts vs. user prompts vs. assistant turns; templates (Jinja2, f-strings, LangChain PromptTemplate); parameterized reusable prompts; negative prompting
- Sampling parameter effects: temperature, top-p, top-k, repetition penalty
- Prompt versioning: tracking prompt changes alongside model versions; regression testing on prompt changes
- Prompt injection awareness: direct and indirect injection; mitigation via input sanitization and separation of data/instructions
- Coding with AI: AI code assistants; generating and modifying scripts; AI-assisted code review; 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
- Drift detection and monitoring
- Capacity management and scaling
- FinOps for AI
- A/B testing and deployment strategies
- Retraining and maintenance
- Decommissioning AI systems

Module 4: 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), VPC/private endpoint configuration, 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 and 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: cataloguing all AI models, APIs, and applications; inventory attributes; continuous maintenance
- Algorithmic impact assessments (AIAs): scope, triggers, methodology, documentation, disclosure requirements
- Bias and fairness audits
- Copyright and IP risk
- AI logging and auditability
- AI incident response
- Secure AI decommissioning

Conclusion and Certification

Course Review and Assessment: Recap of all four exam domains with weighted emphasis matching the exam breakdown. Certification Process: complete the CWNP CAIAE-101 exam within the LMS to earn the Certified AI Administrator and Engineer credential.