PRACTICAL EXERCISE

EX-40H

USAREUR-AF

EX_40H — AI Engineer

Practical Exercise — SL 4H Proficiency

HEADQUARTERS
UNITED STATES ARMY EUROPE AND AFRICA
(USAREUR-AF)
Wiesbaden, Germany

26 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

EX_40H — AI ENGINEER

PRACTICAL EXERCISE — SL 4H PROFICIENCY

Version	1.0 — March 2026
Prerequisite	SL 3 REQUIRED; SL 4H (and SL 1 through SL 2)
Duration	3–4 hours
Environment	MSS with AIP Logic, Agent Studio, Python Transforms — see ENVIRONMENT_SETUP.md

COMPANION RESOURCES

Resource	Reference
Technical Manual	SL 4H — AI Engineer
Syllabus	SYLLABUS_TM40H
Pre-Exercise Exam	EXAM_TM40H_PRE
Post-Exercise Exam	EXAM_TM40H_POST
Continuation Track	SL 5H — Advanced AI Engineer

WFF AWARENESS

AI Engineer products built in this exercise (retrieval pipelines, classification workflows, agents) are consumed by WFF track personnel (TM-40A–F) as operational tools. Evaluators should assess whether the agent's natural-language interface and classification outputs are usable by non-technical WFF operators — a key AI Engineer delivery competency.

SCENARIO

The S2 section has a large corpus of synthetic intelligence summaries (free text). Build a retrieval pipeline over the corpus, create an AIP Logic workflow that classifies summaries by event type, and stand up a basic Agent Studio agent that answers natural-language queries about the corpus.

Training dataset: ~200 synthetic INTSUM documents (plain text, unclassified).

TASK LIST

Task 1 — Ingest and Chunk Documents (30 min)

- ☐ Ingest the synthetic INTSUM corpus into Foundry as a dataset
- ☐ Build a chunking transform: split each document into ~500-token chunks with metadata (`doc_id`, `date`, `event_type_label`)
- ☐ Verify chunk count and schema
- **Go:** Chunks dataset created; metadata columns present; no empty chunks
- **No-Go:** Chunking fails or metadata is missing

Task 2 — Build a Retrieval Pipeline (45 min)

- ☐ Generate embeddings for the chunk dataset using an available embedding model in AIP
- ☐ Store embeddings in a vector index
- ☐ Run a test query and confirm top-3 relevant chunks are returned
- **Go:** Vector index builds; test query returns plausible top-3 results
- **No-Go:** Index fails to build or query returns nonsensical results

Task 3 — AIP Logic Classification Workflow (45 min)

- ☐ Build an AIP Logic workflow that: takes a document as input, calls an LLM to classify it into one of 5 event types (define your taxonomy), outputs the classification label and confidence
- ☐ Test on 10 documents; report accuracy vs. the pre-labeled ground truth
- **Go:** Workflow runs; classification accuracy ≥ 70% on test set
- **No-Go:** Workflow errors or accuracy < 50%

Task 4 — Agent Studio Basic Agent (60 min)

- ☐ Create an Agent Studio agent with access to the vector index and document metadata
- ☐ Define at least 2 tools: (1) search corpus, (2) retrieve document by ID
- ☐ Test: ask the agent a question answerable from the corpus; verify it cites source documents
- **Go:** Agent answers the test question; citations reference real document IDs
- **No-Go:** Agent hallucinates citations or cannot use tools

Task 5 — Governance Documentation (20 min)

- ☐ Document: model/embedding used, retrieval parameters, known failure modes
- ☐ Add a data lineage note: source → chunks → embeddings → index
- **Go:** Documentation is present and accurate
- **No-Go:** Documentation absent or contains factual errors

EVALUATOR NOTES

Scoring: 5 tasks. Go on 4 of 5 = overall Go. No-Go on Task 2 or Task 4 = automatic No-Go.

Pre-exercise checklist: - Confirm AIP Logic is enabled and at least one LLM is available in the training tenant - Confirm Agent Studio is enabled and training accounts can create agents - Confirm embedding model is available (required for Task 2) - Pre-label 10 documents from the corpus with ground truth event types (see ENVIRONMENT_SETUP.md) - Verify vector index creation works in the training environment before exercise day

Common failure modes:

Task	Common Failure	Evaluator Guidance
Task 1	Metadata columns missing from chunk dataset	`doc_id` and `date` are typically forgotten; `event_type_label` often left as None — check schema
Task 2	Embeddings stored as text, not vector type	This prevents index creation; ask participant to show the column type — if wrong, No-Go
Task 2	Vector index builds but test query returns unrelated results	Likely embedding mismatch (wrong model or wrong column) — ask participant to walk through their approach
Task 3	Accuracy checked against wrong ground truth file	Confirm participant used the evaluator-provided ground truth, not self-labeled data
Task 4	Agent cites non-existent document IDs	Hallucinated citations are automatic No-Go; ask participant to click through one citation

Task	Common Failure	Evaluator Guidance
Task 5	Lineage note describes process but omits source dataset name	Coaching note only (not No-Go)

Timing notes: - Task 2 (embeddings + index) depends on model availability and index build time — budget 60 min - Task 4 (Agent Studio) is often the most time-consuming for first-time users — budget 75 min - Participants with prior RAG or vector DB experience will complete Tasks 1–2 in half the expected time

NEXT STEPS

Participants who receive an overall Go on EX_40H are eligible to enroll in **SL 5H — Advanced AI Engineer**. SL 5H extends into multi-agent architectures, adversarial robustness, and production deployment governance. SL 5 is G–O (advanced specialist tracks).

SUPPLEMENTAL — BUILD WITH AIP OFFICIAL TUTORIALS

The following Palantir-published tutorials can be installed directly into your Foundry environment via build.palantir.com. Each provides a complete working example.

Tutorial	Key Concepts	Relevance
Advanced Fault-Tolerant Document Extraction (Vision)	Vision + language models, PDF entity extraction, RAG	AIP Logic, Document Intelligence
Advanced Fault-Tolerant Document Extraction (Entity-Less)	Unstructured content processing at scale	Scalable extraction patterns
AIP Agent with Long-Term Memory	Agent Studio, cross-interaction persistent memory	Agent design, memory patterns
Register an LLM via Function Interfaces	Custom LLM integration, TypeScript functions	LLM integration
Cluster Text with Embeddings	Embedding-based text clustering	RAG, semantic search

NOTE

These tutorials use Palantir's standard datasets. Adapt ontology patterns to USAREUR-AF naming conventions before deploying to production.