Advanced Accounting Agent Skill Guide

Functional overview, subagent architecture, implementation notes, and sample test data

ADVANCED ACCOUNTING AGENT / WEST AFRICA

1. Overview

The Advanced Accounting Agent is an AI-assisted workflow for accounting procedures across West Africa, initially focused on Côte d'Ivoire. It helps convert receipts, invoices, bank statements and spreadsheets into structured, reviewable double-entry journal entries using SYSCOHADA accounting conventions and configurable jurisdiction-specific tax checks.

The goal is more than spreadsheet automation: extracted facts must be matched to appropriate accounts, tax treatment and supporting evidence. Final postings should be reviewed and approved by a qualified accountant.

2. Core functionality

- Document parsing: Ingest PDF, JPEG and XLSX files; extract issuer, dates, line items, amounts, tax fields and payment references.
- Data standardization: Normalize dates, currency (XOF), vendor identities, account mapping and cost centers against an approved chart of accounts.
- Compliance layer: Flag missing mandatory details, verify arithmetic, and apply configurable rules for SYSCOHADA and relevant Ivorian invoice/tax requirements, including FNE and VAT checks.
- Output: Produce balanced debit/credit journal entry candidates, traceable to original documents, for review and subsequent import.

3. Subagent capabilities and concurrency

For independent tasks such as processing Invoice A while reconciling Statement B, an application-level work queue is preferable to a strictly sequential subagent loop. The wrapper can enqueue tasks, assign bounded workers, capture job IDs and statuses, and aggregate results without blocking unrelated work.

- Suggested orchestration: Task queue + bounded worker pool + retry/backoff + idempotency keys + persistent job status.
- In-session parallelism: `Promise.allSettled` can collect independent asynchronous tool results in a JavaScript environment that supports them; avoid parallel writes to the same workbook or ledger.
- Memory optimization: Use appropriately quantized local models where compatible, reduce context size, cache parsed data, and put deterministic arithmetic and reconciliation in conventional code.
- Controls: Mark low-confidence extraction, require human approval for final postings, and preserve audit logs and supporting document references.

4. Technical comparison: Python vs. Rust

Criterion	Python	Rust
Development	Fast prototyping; broad OCR, data and AI libraries	Steeper learning curve; more implementation work
Runtime	Strong performance with optimized libraries	Excellent low-level speed and memory control
Operations	Straightforward integration with pandas, NumPy and pydantic	Useful for CPU-heavy components at scale
Recommendation	Default implementation for the agent	Consider later for a profiled bottleneck

Recommendation: Start with Python. Document processing, model inference and external I/O will commonly dominate total latency. Profile real workloads before considering any Rust rewrite.

5. Quick start

Prerequisites: Python runtime, installed dependencies, access to suitable document extraction / language-model capabilities (local models or an API), and an approved jurisdictional rule set.

```
pip install -r requirements.txt
python advanced_accounting_skill.py --receipt /path/to/photo.jpg --doc-type Receipt
```

Note: The command above is an illustrative CLI interface. It assumes an implementation of `advanced_accounting_skill.py` providing those arguments.

6. Sample test data: November 2026

Scenario: A manager submits expenses and supplier invoices for November 2026. The agent must identify duplicates, validate tax information, classify transactions and propose balanced entries in XOF. These figures are fictional and illustrative only; they do not establish legal tax treatment.

ID / type	Input data	Expected system behavior
INV-001 / invoice	Office supplies; net 100,000 XOF; VAT shown 18,000 XOF; gross 118,000 XOF	Validate totals; map expense and tax accounts; credit vendor payable for 118,000, subject to tax eligibility
REC-002 / receipt	Transport expense 12,500 XOF; cash paid; no VAT stated	Record gross cost and cash settlement; do not invent deductible VAT
BNK-003 / bank	Debit 118,000 XOF; reference INV-001	Match against supplier payment; avoid treating payment as a second expense
INV-004 / invoice	Same supplier, date, reference and 118,000 XOF as INV-001	Raise duplicate suspicion; require review before posting

Example candidate journal entries

Document	Debit	Credit
INV-001	Office supplies expense: 100,000; recoverable VAT: 18,000 (if eligible)	Supplier payable: 118,000
REC-002	Transport expense: 12,500	Cash: 12,500
BNK-003	Supplier payable: 118,000	Bank: 118,000

Test assertions: (1) Each entry balances; (2) invoice payment is reconciled without duplicate expense recognition; (3) INV-004 is flagged; (4) missing tax evidence is not fabricated; (5) source IDs and review status are retained.

7. Scope and compliance caution

SYSCOHADA is an accounting framework, while VAT and FNE obligations arise from applicable national tax and invoicing rules. The system should validate against maintained, versioned rule sets and require professional review of exceptions and final filings. It should not promise automatic legal compliance without jurisdictional validation.