CDW Documentation

SOW Analyzer – Technical User Guide & Reference Manual

SOW Analyzer – Technical User Guide & Reference Manual

1. System Overview

The SOW Analyzer is a Microsoft Teams-integrated AI validation system that evaluates uploaded Statements of Work (SOWs) against predefined “golden” template documents.

Architecture Components:

- Microsoft Teams Bot (Bot Framework)
- Azure Web App (FastAPI backend)
- Azure Document Intelligence (prebuilt-layout model)
- Azure OpenAI (GPT-based semantic comparison engine)

The system supports:

- .pdf
- .doc
- .docx

Uploaded documents are processed in-memory and are not persisted.

---

2. High-Level Architecture Flow

```
Teams User
   ↓
Teams Bot (teams_bot.py)
   ↓
File Download + Validation
   ↓
POST /analyze-sow (app.py)
   ↓
Azure Document Intelligence (layout extraction)
   ↓
Azure OpenAI GPT comparison
   ↓
Structured Response to Teams
```

---

3. Teams Bot (teams_bot.py)

3.1 Message Handling

Entry point:

```
async def on_turn(turn_context: TurnContext)
```

Behavior:

- Detect file attachment
- Extract filename and download URL
- Validate supported file extension
- Download file bytes
- Validate file signature
- Send multipart POST request to backend analyzer

3.2 Supported File Types

Defined in:

```
SUPPORTED_FILE_TYPES = {
    ".pdf": "application/pdf",
    ".doc": "application/msword",
    ".docx": "application/vnd.openxmlformats-
officedocument.wordprocessingml.document",
}
```

Extension gating:

```
if not any(filename.lower().endswith(ext) for ext in
SUPPORTED_FILE_TYPES.keys()):
```

3.3 File Signature Validation

Function:

```
def _validate_file_signature(filename: str, data: bytes)
```

Validation Rules:

- PDF → Must begin with %PDF
- DOCX → Must begin with PK (ZIP header)
- DOC → Must begin with OLE header (D0 CF 11 E0 A1 B1 1A E1)

Purpose:

- Prevent file type spoofing
- Enforce integrity of uploaded documents
- Mitigate malicious upload risk

3.4 Download Logic

Function:

```
async def _download_bytes(url: str, filename: str) -> bytes
```

Security controls:

- HTTP status validation
- Empty file detection
- File size limit enforcement (MAX_UPLOAD_BYTES)
- Signature validation

3.5 Analyzer POST Logic

Function:

```
async def _post_to_analyzer(filename: str, file_bytes: bytes, content_type: str)
```

Behavior:

- Builds multipart/form-data request
- Sends file to ANALYZE_SOW_URL
- Forwards correct MIME type
- Returns JSON response

---

4. Backend API (app.py)

4.1 Endpoint Definition

```
@app.post("/analyze-sow")
async def analyze_sow(file: UploadFile = File(...))
```

Validation:

- Content-Type allowlist
- Extension allowlist
- Empty file detection

Allowed MIME types:

- application/pdf
- application/msword
- application/vnd.openxmlformats-officedocument.wordprocessingml.document
- application/octet-stream (fallback)

4.2 Content-Type Mapping Logic

```
if filename.endswith(".docx"):
    content_type = "application/vnd.openxmlformats-
officedocument.wordprocessingml.document"
elif filename.endswith(".doc"):
    content_type = "application/msword"
else:
    content_type = "application/pdf"
```

Purpose:

- Ensure correct format handling by Document Intelligence
- Override unreliable client-provided MIME types

---

5. Document Extraction (layout_client.py)

5.1 Extraction Function

```
def extract_layout_text(file_bytes: bytes, content_type: str) -> str
```

Uses:

- Azure Document Intelligence
- prebuilt-layout model

Call Pattern:

```
poller = client.begin_analyze_document(
    "prebuilt-layout",
    document=file_bytes,
    content_type=content_type
)
```

Output:

- Structured page-aware text
- Preserves headings, paragraphs, and tables

- Returns full reconstructed textual representation

---

6. GPT Comparison Engine

6.1 Function

```
compare_against_best_template(
    uploaded_sow_text,
    golden_templates
)
```

Purpose:

- Compare uploaded SOW text against approved template corpus
- Identify missing clauses
- Detect semantic deviations
- Provide structured gap analysis

Characteristics:

- No fine-tuning
- No persistent training
- Inference-only AI reasoning
- Deterministic template baseline

---

7. Golden Template Management

Golden templates are loaded at application startup and held in memory.

Design Principles:

- Immutable baseline
- Controlled governance
- Centralized review logic
- No dynamic template training

---

8. Security Architecture

8.1 No Persistent Storage

Uploaded files:

- Not written to disk
- Not stored in blob storage
- Not logged
- Not cached

Processing is memory-bound only.

8.2 Input Hardening

Controls implemented:

- Extension allowlist
- MIME type enforcement
- Signature verification
- File size limits
- HTTP status validation
- Timeout configuration

8.3 AI Isolation

Service separation:

- Teams Bot → Presentation layer
- FastAPI → Orchestration layer
- Document Intelligence → Extraction layer
- Azure OpenAI → Reasoning layer

Benefits:

- Clear responsibility boundaries
- Easier auditing
- Reduced lateral movement risk
- Improved troubleshooting clarity

---

9. Configuration Reference

Required Environment Variables:

- ANALYZE_SOW_URL
- Azure Document Intelligence endpoint + key
- Azure OpenAI endpoint + key
- Model deployment name

Timeouts:

- Download: 60s
- Analyzer call: 180s

File Size Limit:

- Defined via MAX_UPLOAD_BYTES

---

10. Error Handling Behavior

Error Categories:

- Unsupported file type → User-facing validation message
- Empty file → 400 response
- Layout extraction error → 500 with diagnostic message
- GPT comparison error → 500
- Analyzer HTTP failure → surfaced to Teams user

Errors are intentionally explicit to simplify debugging in production.

---

11. Extension Points

Potential enhancements:

- Risk scoring model
- Structured JSON scoring output
- Audit log integration
- Template versioning
- Clause-level similarity scoring
- Confidence scoring metrics

---

12. Operational Characteristics

Performance Factors:

- Document size
- Azure Document Intelligence latency
- GPT token size of uploaded document
- Template corpus size

Scalability:

- Horizontally scalable via Azure Web App scaling
- Stateless backend design

- No session persistence required

---

13. Compliance Considerations

- No long-term data retention
- No model fine-tuning on customer data
- Deterministic template baseline
- Azure-hosted inference services
- Explicit file validation controls

---

14. Summary

The SOW Analyzer is a stateless, AI-driven SOW validation engine embedded directly in Microsoft Teams. It combines structured document extraction and semantic reasoning while maintaining strong governance controls and zero persistent storage of uploaded contractual documents.

It is designed for:

- Secure enterprise AI adoption
- Operational contract standardization
- Reduced review cycle time
- Enforced template compliance
- Scalable AI document validation