# **Technical Requirements Document (TRD)**

## **Multi-Document Program Analyzer → Automated Excel KPI Dashboard Generator**

### **Document Version: 2.0 (Integrated Final)**

### **Date: January 2025**

### **Classification: Technical Specification - Production Ready**

## **1. EXECUTIVE SUMMARY**

### **1.1 Purpose**

This document defines the comprehensive technical requirements for an automated system that ingests diverse program documents (PDF/XLSX/DOCX/etc.), extracts requirements & KPIs, and automatically produces a funder-ready Excel dashboard workbook with full automation, conditional formatting, and cross-referenced data linkages.

### **1.2 System Capabilities**

* **Multi-format document ingestion** with intelligent classification
* **Natural language processing** for requirement extraction
* **Automated KPI identification** and metric calculation
* **Excel workbook generation** with zero manual intervention
* **Complete formula automation** with error handling
* **Professional visualization** with conditional formatting
* **Audit trail** via JSON export of parsed structures

### **1.3 Success Criteria**

* ✅ Dashboard mirrors program requirements with zero Excel repair prompts
* ✅ All KPIs show Monthly and Cumulative progress with thresholds
* ✅ Executive Command Center summarizes status, risks, budget at-a-glance
* ✅ Calendar displays all deadlines/milestones color-coded by site
* ✅ 100% formula accuracy with complete error handling
* ✅ Passes all acceptance tests (Section 13)

## **2. SCOPE & OBJECTIVES**

### **2.1 In-Scope**

* Accept multiple files describing a program (logic models, timelines, staffing, training, risks, budgets, MOUs, prior dashboards)
* Analyze and normalize content into unified data model
* Generate single Excel workbook with fully wired formulas, conditional formatting, data validation, calendar views
* Provide JSON extract of parsed assumptions, targets, KPIs for auditability
* Support for 200+ pages across 25 documents per run

### **2.2 Out-of-Scope (v1)**

* Real-time API to third-party analytics (stubbed via CSV import)
* Live write-back from Excel to databases (supported via exported CSVs)
* Creation of native pivot caches (use formula-based rollups instead)
* VBA macros (unless specifically requested)

### **2.3 Stakeholders & Roles**

| **Role** | **Responsibilities** |
| --- | --- |
| **Program Owner** | Supplies documents, approves KPI taxonomy, signs off on dashboard |
| **Data Engineer** | Maintains parsers, data model, workbook builder |
| **Program Analyst** | Verifies mappings, sets targets, defines thresholds |
| **Compliance Lead** | Reviews risk, releases, and approval fields |
| **Quality Assurance** | Validates output accuracy and formula correctness |

## **3. INPUT SPECIFICATIONS**

### **3.1 Supported Document Formats**

| **Format** | **Extensions** | **Use Cases** | **Processing Method** |
| --- | --- | --- | --- |
| **PDF** | .pdf | Logic models, reports, timelines | pdfminer.six/PyMuPDF + OCR |
| **Excel** | .xlsx, .xls, .xlsm | Budgets, prior dashboards, rosters | pandas/openpyxl |
| **Word** | .docx, .doc | Policies, training outlines, SOWs | python-docx |
| **CSV** | .csv | Data exports, metrics | pandas |
| **Text** | .txt, .md | Notes, requirements | Direct parsing |
| **Images** | .png, .jpg | Schedules, flyers | pytesseract OCR |
| **Data** | .json, .xml | Structured data | Native parsers |

### **3.2 File Limits (Configurable)**

yaml

Maximum Files: 25 per run

Maximum Pages: 200 total across all documents

Maximum File Size: 50MB per file

Total Processing Size: 500MB per batch

OCR Page Limit: 50 pages per run

### **3.3 Document Classification Taxonomy**

python

DOCUMENT\_TYPES = {

"Logic Model": ["logic", "theory", "change", "framework"],

"Timeline": ["timeline", "schedule", "gantt", "milestone"],

"Budget": ["budget", "financial", "cost", "expense"],

"Risk Guide": ["risk", "mitigation", "issue", "threat"],

"Training": ["training", "curriculum", "module", "onboarding"],

"Role Specs": ["role", "position", "staff", "responsibility"],

"School Strategy": ["school", "campus", "site", "partnership"],

"Prior Dashboard": ["dashboard", "kpi", "metric", "report"],

"Status Report": ["status", "progress", "update", "monthly"],

"Participant Data": ["participant", "youth", "enrollment", "roster"]

}



## **4. PROCESSING PIPELINE**

### **4.1 Pipeline Architecture**

mermaid

graph LR

A[Document Ingestion] --> B[Parsing & OCR]

B --> C[Classification]

C --> D[Entity Extraction]

D --> E[Normalization]

E --> F[Rules Engine]

F --> G[Validation]

G --> H[Excel Generation]

H --> I[Quality Assurance]

I --> J[Output & Audit]

### **4.2 Detailed Processing Stages**

#### **4.2.1 Ingestion**

python

def ingest\_documents(file\_list):

"""

Store originals with metadata

"""

for file in file\_list:

metadata = {

'filename': file.name,

'size': file.size,

'checksum': compute\_sha256(file),

'pages': count\_pages(file),

'type': detect\_type(file),

'timestamp': datetime.now()

}

store\_with\_metadata(file, metadata)

#### **4.2.2 Parsing**

* **PDF:** Text extraction + table detection using camelot/tabula-py
* **DOCX:** Structured text + embedded tables + styles
* **XLSX:** Multi-sheet analysis with formula preservation
* **Images:** OCR with confidence scoring

#### **4.2.3 Information Extraction**

yaml

Entities to Extract:

- Goals and Objectives

- Activities and Tasks

- KPIs and Metrics

- Targets and Thresholds

- Deadlines and Milestones

- Sites and Locations

- Roles and Responsibilities

- Budget Lines and Costs

- Risks and Mitigations

- Training Modules

- Deliverables

#### **4.2.4 Normalization Rules**

python

DATE\_NORMALIZATION = {

'input\_formats': ['MM/DD/YYYY', 'YYYY-MM-DD', 'DD-MMM-YY'],

'output\_format': 'YYYY-MM-DD',

'timezone': 'UTC'

}

STATUS\_MAPPING = {

'complete': 'Completed',

'done': 'Completed',

'in-progress': 'In Progress',

'ongoing': 'In Progress',

'pending': 'Not Started',

'not started': 'Not Started'

}



## **5. UNIFIED DATA MODEL (UDM)**

### **5.1 Core Objects**

python

@dataclass

class Program:

name: str

year: int

sites: List[Site]

budget\_total: Decimal

currency: str = "USD"

@dataclass

class Site:

name: str

color: str *# Hex color for calendar*

district: Optional[str]

liaison: Optional[Person]

approval\_artifact: Optional[str]

@dataclass

class Activity:

goal\_ref: str

type: str

description: str

site\_ref: Optional[Site]

owner\_ref: Optional[Person]

kpi\_refs: List[str]

@dataclass

class KPI:

name: str

category: str

unit: str

target\_monthly: float

target\_total: float

direction: Literal["Maximize", "Minimize"]

thresholds: Dict[str, float] *# {"pass": 1.0, "warn": 0.8, "fail": 0.5}*

formula: Optional[str]

@dataclass

class Task:

goal: str

action: str

kpi\_ref: Optional[str]

owner\_ref: Optional[Person]

site\_ref: Optional[Site]

deadline: date

progress\_pct: float

status: str

barriers: Optional[str]

school\_partnership: Optional[str]

### **5.2 Relationships & Business Rules**

* Every Activity maps to ≥1 KPI (traceability requirement)
* Every Task references a KPI or Activity
* Every deadline appears on Calendar with site color
* Budget totals must equal sum(BudgetLine.planned\_cost) ±0.5%
* All high-severity risks require mitigation plan

## **6. KPI TAXONOMY & METRICS**

### **6.1 Standard KPI Categories**

yaml

Content Production:

metrics:

- "# Reels Created"

- "# Live Sessions"

- "# Long-form Content"

- "% Content Approved"

- "Average Engagement Rate"

thresholds:

pass: "≥100% of target"

warn: "≥80% of target"

fail: "<80% of target"

Youth Engagement:

metrics:

- "# Events Hosted"

- "Total Attendance"

- "# Pop-ups/Outreach"

- "Retention Rate %"

- "Satisfaction Score"

thresholds:

pass: "≥85%"

warn: "≥70%"

fail: "<70%"

Training & Development:

metrics:

- "Onboarding Completion %"

- "Module Completion Rate"

- "Training Hours Delivered"

- "Competency Assessment Score"

School Partnerships:

metrics:

- "# Active Sites"

- "% Sites with Liaison"

- "% Sites with MOUs"

- "Touchpoints per Month"

Budget & Resources:

metrics:

- "Budget Utilization %"

- "Cost per Participant"

- "Staff Hours vs Plan"

- "Burn Rate"

Risk & Compliance:

metrics:

- "% Releases on File"

- "# Open High Risks"

- "Average Resolution Time"

- "Compliance Score"

### **6.2 KPI Calculation Formulas**

excel

Retention Rate = COUNTIFS(Status,"Completed",Criteria,"Yes")/COUNTA(Participants)

Completion % = Tasks\_Completed/Total\_Tasks

Budget Variance = (Actual-Planned)/Planned

Engagement Rate = (Likes+Comments+Shares)/Impressions

Risk Score = COUNT(High\_Risks)\*3 + COUNT(Medium\_Risks)\*2 + COUNT(Low\_Risks)\*1



## **7. EXCEL WORKBOOK SPECIFICATION**

### **7.1 File Naming Convention**

****<ProgramName>\_KPI\_Dashboard\_YYYYMMDD\_v<Version>.xlsx

Example: EYC\_KPI\_Dashboard\_20250120\_v2.xlsx

### **7.2 Required Sheet Structure (Order Matters)**

| **#** | **Sheet Name** | **Purpose** | **Key Formulas** |
| --- | --- | --- | --- |
| 1 | Executive Command Center | High-level rollup | Aggregations from all sheets |
| 2 | KPI Dashboard | Monthly & cumulative metrics | COUNTIFS, SUMIFS with date ranges |
| 3 | Project Management | Task tracking with schools | Status calculations, overdue flags |
| 4 | Impact KPIs | Outcome measurements | Target vs actual comparisons |
| 5 | Logic Framework KPIs | Input→Output mapping | Linked to activities |
| 6 | Roster | People/roles/sites/rates | VLOOKUP references |
| 7 | School Partnerships | Site-specific tracking | Partnership metrics |
| 8 | Training Tracker | Module completion | Progress calculations |
| 9 | Content Log | Media production tracking | Approval rates, engagement |
| 10 | Event Log | Event management | Attendance summations |
| 11 | Risk & Mitigation | Risk register | Severity scoring |
| 12 | Budget Tracker | Financial management | Variance calculations |
| 13 | Calendar View | Timeline visualization | Date-based filtering |
| 14 | Program Performance | Efficiency metrics | Complex KPI formulas |
| 15 | Barriers & Needs | Issue tracking | Status counts |
| 16 | Top Successes | Achievement highlights | Top N calculations |
| 17 | Formula Documentation | Reference guide | Text descriptions |
| 18 | Reference Lists | Hidden dropdowns | Named ranges |

### **7.3 Column Schemas**

#### **Project Management Sheet**

****| Goal | Specific Action | KPI/Outcome | KPI Measurement | Owner | School Partnership | Deadline | Progress% | Status | Barriers/Notes |

#### **Content Log Sheet**

****| Date | Site | Owner | Type | Title | Link | Approved(Y/N) | Reach | Saves | Engagement | Notes |

#### **Budget Tracker Sheet**

****| Role/Site | Planned Hours | Actual Hours | Rate | Planned $ | Actual $ | Variance $ | Variance % | Burn Rate |

### **7.4 Formula Standards**

excel

# COUNTING FORMULAS

Total Tasks: =COUNTA('Project Management'!C:C)-1

Completed: =COUNTIF('Project Management'!H:H,"Completed")

Overdue: =COUNTIFS('Project Management'!G:G,"<"&TODAY(),'Project Management'!H:H,"<>Completed")

# MATHEMATICAL FORMULAS

Budget Variance: =IFERROR((Actual-Planned)/Planned,0)

Burn Rate: =IFERROR(Actual/Planned,0)

Average Score: =IFERROR(AVERAGE(ScoreRange),0)

# LOGICAL FORMULAS

Status: =IF(Progress=1,"Completed",IF(Progress>0,"In Progress","Not Started"))

RAG Status: =IF(Value>=Target,"PASS",IF(Value>=Target\*0.8,"WARN","FAIL"))

Overdue Flag: =AND(TODAY()>Deadline,Status<>"Completed")

# TEXT FORMULAS

Summary: =CONCATENATE(COUNT(Tasks)," tasks, ",COUNTIF(Status,"Completed")," completed")

Month Label: =TEXT(Date,"MMM YYYY")

# ERROR HANDLING

All Division: =IFERROR(numerator/denominator,0)

All Lookups: =IFERROR(VLOOKUP(...),"Not Found")

All Averages: =IFERROR(AVERAGE(range),0)

### **7.5 Conditional Formatting Rules**

#### **Status Column Formatting**

| **Condition** | **Fill Color** | **RGB Code** | **Formula** |
| --- | --- | --- | --- |
| Completed | Green | #C6EFCE | =$H2="Completed" |
| In Progress | Yellow | #FFEB9C | =$H2="In Progress" |
| Not Started | Red | #FFC7CE | =$H2="Not Started" |
| Overdue | Dark Red | #FF6B6B | =AND(TODAY()>$G2,$H2<>"Completed") |

#### **Progress Indicators**

excel

100% Complete: Green fill (#C6EFCE)

80-99% Complete: Light Green (#E6F4EA)

50-79% Complete: Yellow (#FFEB9C)

25-49% Complete: Light Yellow (#FFF4E6)

1-24% Complete: Light Red (#FFEAA7)

0% Complete: Red (#FFC7CE)

#### **KPI Thresholds (3-Color Scale)**

yaml

Pass Zone: Green (*#C6EFCE) when ≥100% of target*

Warn Zone: Yellow (*#FFEB9C) when 80-99% of target*

Fail Zone: Red (*#FFC7CE) when <80% of target*

### ****7.6 Data Validation Rules**

yaml

Status Dropdown:

Values: [Completed, In Progress, Not Started, Blocked]

InputMessage: "Select the current status of this task"

Priority Dropdown:

Values: [High, Medium, Low]

InputMessage: "Select task priority level"

Site Dropdown:

Source: =Sites (named range)

InputMessage: "Select the school or site"

Risk Severity:

Values: [High, Medium, Low]

InputMessage: "Assess risk severity level"

Date Fields:

Type: Date

Format: MM/DD/YYYY

InputMessage: "Enter date in MM/DD/YYYY format"

ErrorMessage: "Invalid date format"

### **7.7 Visual Design Standards**

css

*/\* Color Palette \*/*

Primary Blue: #004C97

Secondary Blue: #0070C0

Light Blue: #DDEBF7

Success Green: #C6EFCE

Warning Yellow: #FFEB9C

Danger Red: #FFC7CE

Dark Text: #000000

Light Text: #FFFFFF

Gray Text: #666666

*/\* Font Specifications \*/*

Headers: Calibri Bold 14pt White on Blue

Subheaders: Calibri Bold 11pt Dark Blue

Body: Calibri Regular 10pt Black

Numbers: Calibri Regular 10pt with number format



## **8. TECHNICAL ARCHITECTURE**

### **8.1 System Components**

python

class DashboardGenerator:

"""

Core system for automated dashboard generation

"""

def \_\_init\_\_(self):

self.config = ConfigManager()

self.ingester = DocumentIngester()

self.parser = MultiFormatParser()

self.classifier = DocumentClassifier()

self.extractor = EntityExtractor()

self.normalizer = DataNormalizer()

self.rules\_engine = RulesEngine()

self.validator = DataValidator()

self.excel\_builder = ExcelBuilder()

self.formatter = FormatManager()

self.qa\_checker = QualityAssurance()

def process(self, documents: List[Path]) -> Tuple[Path, Dict]:

"""

Main processing pipeline

Returns: (excel\_path, audit\_json)

"""

*# Phase 1: Ingest and classify*

ingested = self.ingester.ingest\_all(documents)

classified = self.classifier.classify(ingested)

*# Phase 2: Parse and extract*

parsed = self.parser.parse\_all(classified)

entities = self.extractor.extract(parsed)

*# Phase 3: Normalize and validate*

normalized = self.normalizer.normalize(entities)

validated = self.validator.validate(normalized)

*# Phase 4: Apply rules and build*

processed = self.rules\_engine.apply(validated)

workbook = self.excel\_builder.build(processed)

*# Phase 5: Format and QA*

formatted = self.formatter.apply(workbook)

final = self.qa\_checker.verify(formatted)

*# Phase 6: Generate outputs*

excel\_path = self.save\_excel(final)

audit\_json = self.generate\_audit(processed)

return excel\_path, audit\_json

### **8.2 Technology Stack**

yaml

Language: Python 3.11+

Core Libraries:

PDF Processing:

- pdfminer.six: Text extraction

- PyMuPDF: Page rendering

- camelot-py: Table extraction

- pytesseract: OCR for scanned docs

Document Processing:

- python-docx: Word documents

- pandas: Data manipulation

- openpyxl: Excel I/O

NLP & Extraction:

- spaCy: Entity recognition

- dateutil: Date parsing

- regex: Pattern matching

Excel Generation:

- openpyxl: Workbook creation

- xlsxwriter: Advanced charting

- numpy: Calculations

Configuration:

- YAML: Settings and templates

- JSON: Data exchange

- dotenv: Environment variables

### **8.3 Configuration Management**

yaml

*# config.yml*

program:

name: "EYC"

year: 2025

sites:

- name: "Lincoln High"

color: "#FF6B6B"

- name: "Washington Middle"

color: "#4ECDC4"

kpi\_thresholds:

default:

pass: 1.0

warn: 0.8

fail: 0.5

retention\_rate:

pass: 0.85

warn: 0.70

fail: 0.50

excel\_options:

max\_rows: 1000000

enable\_charts: true

include\_calendar: true

protect\_formulas: false

processing:

enable\_ocr: true

ocr\_confidence\_threshold: 0.7

parallel\_processing: true

max\_workers: 4



## **9. QUALITY ASSURANCE & VALIDATION**

### **9.1 Formula Validation**

python

FORMULA\_TESTS = {

"no\_errors": "All formulas compile without #ERROR",

"no\_circular": "No circular references detected",

"ranges\_exist": "All referenced ranges are valid",

"iferror\_wrapped": "Division operations wrapped in IFERROR",

"no\_emoji": "No emoji characters in formulas",

"performance": "No volatile functions in large ranges"

}

### **9.2 Workbook Validation**

python

def validate\_workbook(wb: Workbook) -> ValidationResult:

"""

Comprehensive workbook validation

"""

checks = {

"sheet\_count": len(wb.worksheets) >= 15,

"sheet\_order": validate\_sheet\_order(wb),

"formulas\_valid": validate\_all\_formulas(wb),

"formatting\_applied": check\_conditional\_formats(wb),

"validation\_rules": check\_data\_validation(wb),

"named\_ranges": verify\_named\_ranges(wb),

"no\_repair\_needed": test\_excel\_open(wb),

"merged\_cells\_valid": check\_merged\_cells(wb)

}

return ValidationResult(checks)

### **9.3 Data Integrity Rules**

yaml

Validation Rules:

- No duplicate task IDs

- All KPIs linked to activities

- Budget lines sum to total ±0.5%

- All dates in valid range

- All percentages 0-100%

- No orphaned references

- All required fields populated

- Status values from allowed list



## **10. ERROR HANDLING & RESILIENCE**

### **10.1 Parser Error Handling**

python

ERROR\_STRATEGIES = {

"pdf\_corrupt": "Skip page, log error, continue",

"ocr\_fail": "Flag low confidence, provide manual entry",

"table\_parse\_fail": "Attempt alternate parser, fallback to text",

"date\_parse\_fail": "Use document date as default",

"formula\_error": "Wrap in IFERROR, log for review"

}

### **10.2 Extraction Confidence Scoring**

python

CONFIDENCE\_THRESHOLDS = {

"high": 0.9, *# Auto-accept*

"medium": 0.7, *# Flag for review*

"low": 0.5, *# Require manual verification*

"reject": 0.5 *# Below this, do not use*

}

### **10.3 Graceful Degradation**

* Missing data: Populate with defaults, flag in audit
* Partial extraction: Generate what's possible, document gaps
* Format issues: Attempt recovery, maintain structure
* Formula errors: Provide static values as fallback

## **11. PERFORMANCE & SCALABILITY**

### **11.1 Performance Targets**

yaml

Processing Metrics:

Document Ingestion: <5 seconds per document

OCR Processing: <10 seconds per page

Entity Extraction: <2 seconds per document

Excel Generation: <30 seconds for full workbook

Total Pipeline: <2 minutes for 20 documents

Resource Limits:

Max Memory: 4GB

Max CPU: 4 cores

Max Temp Storage: 1GB

Scalability:

Concurrent Documents: 10

Max Excel Rows: 1,048,576

Max Excel Columns: 16,384

Max Sheets: 255

### **11.2 Optimization Strategies**

* Cache parsed documents for re-runs
* Parallel processing for independent documents
* Lazy loading for large datasets
* Incremental formula calculation
* Batch writing for Excel operations

## **12. SECURITY & COMPLIANCE**

### **12.1 Data Security**

yaml

Encryption:

At Rest: AES-256

In Transit: TLS 1.3

Access Control:

Authentication: OAuth 2.0 / SAML

Authorization: Role-based (RBAC)

Audit Logging: All operations logged

PII Handling:

Detection: Automated PII scanning

Redaction: Configurable by field

Storage: Encrypted, time-limited

Export: Opt-in for PII fields

### **12.2 Compliance Requirements**

* FERPA compliance for educational data
* HIPAA considerations for health information
* GDPR for international participants
* State privacy laws (CCPA, etc.)

## **13. ACCEPTANCE TESTS**

### **13.1 Test Suite**

python

class AcceptanceTests:

def AT\_1\_document\_ingestion(self):

"""

Provide 8-15 files of various formats

Pass: System classifies each correctly, extracts ≥90% entities

"""

def AT\_2\_workbook\_generation(self):

"""

Generate complete XLSX

Pass: No repair prompts, all sheets present, order correct

"""

def AT\_3\_kpi\_calculations(self):

"""

Add test data, verify calculations

Pass: Monthly & cumulative update correctly

"""

def AT\_4\_calendar\_integration(self):

"""

Check deadline mapping

Pass: 100% of deadlines appear on calendar with colors

"""

def AT\_5\_budget\_formulas(self):

"""

Modify hours, check cascade

Pass: Totals and variances recalculate correctly

"""

def AT\_6\_risk\_tracking(self):

"""

Add high-severity risk

Pass: Executive dashboard reflects, mitigation tracked

"""

def AT\_7\_conditional\_formatting(self):

"""

Test status changes

Pass: Colors update per rules

"""

def AT\_8\_data\_validation(self):

"""

Test all dropdowns

Pass: Lists work, messages appear

"""

def AT\_9\_cross\_references(self):

"""

Verify sheet linkages

Pass: All formulas reference correct sheets

"""

def AT\_10\_performance(self):

"""

Process maximum load

Pass: Complete in <2 minutes

"""

### **13.2 Test Data Requirements**

yaml

Test Dataset:

Documents:

- Logic Model PDF (10 pages)

- Budget XLSX (5 sheets)

- Timeline DOCX (Gantt chart)

- Prior Dashboard XLSX

- Training Guide PDF (20 pages)

- Risk Register CSV

- Participant Roster XLSX

- Status Reports (3 monthly)

Expected Outputs:

- 50+ KPIs identified

- 100+ tasks extracted

- 10+ risks documented

- 500+ data points total



## **14. DEPLOYMENT & OPERATIONS**

### **14.1 Deployment Configuration**

yaml

Environment Setup:

Python: 3.11+

Dependencies: requirements.txt

Config: config.yml

Templates: /templates/\*.yml

Execution:

CLI: python dashboard\_generator.py --config config.yml --input /docs

API: POST /api/generate with multipart/form-data

UI: Upload via web interface (optional)

Outputs:

Excel: /output/<program>\_dashboard\_<date>.xlsx

Audit: /output/<program>\_audit\_<date>.json

Logs: /logs/<date>\_processing.log

### **14.2 Operational Runbook**

markdown

## Daily Operations

1. **\*\*Morning Check\*\***

- Verify service health

- Check error logs

- Review pending jobs

2. **\*\*Processing New Documents\*\***

- Validate input documents

- Run generator with appropriate config

- Review confidence scores

- Approve/adjust extractions

3. **\*\*Quality Review\*\***

- Open generated Excel

- Spot-check formulas

- Verify conditional formatting

- Test data entry

4. **\*\*Delivery\*\***

- Final QA check

- Generate audit report

- Deliver to stakeholder

- Archive inputs/outputs



## **15. EXTENSIBILITY & CUSTOMIZATION**

### **15.1 Plugin Architecture**

python

class ExtractionPlugin(ABC):

"""Base class for custom extractors"""

@abstractmethod

def can\_handle(self, document: Document) -> bool:

"""Check if plugin handles this document type"""

@abstractmethod

def extract(self, document: Document) -> Entities:

"""Extract entities from document"""

class KPITemplatePlugin(ABC):

"""Base class for domain-specific KPI templates"""

@abstractmethod

def get\_kpi\_definitions(self) -> List[KPI]:

"""Return domain-specific KPIs"""

### **15.2 Template Packs**

yaml

Available Templates:

- Youth Development

- Workforce Training

- Health Programs

- Education Initiatives

- Arts & Culture

- Environmental Programs

- Community Development

Custom Templates:

Location: /templates/custom/

Format: YAML with KPI definitions

Override: Merge with defaults



## **16. MAINTENANCE & SUPPORT**

### **16.1 Maintenance Schedule**

yaml

Daily:

- Monitor error logs

- Check processing queue

Weekly:

- Review extraction accuracy

- Update confidence thresholds

Monthly:

- Formula optimization

- Template updates

- Performance analysis

Quarterly:

- Excel compatibility testing

- Security patches

- Feature enhancements

Annually:

- Major version upgrade

- Architecture review

- Compliance audit

### **16.2 Support Matrix**

| **Issue Type** | **Response Time** | **Resolution Target** |
| --- | --- | --- |
| Critical (System Down) | 1 hour | 4 hours |
| High (Incorrect Output) | 4 hours | 1 business day |
| Medium (Missing Features) | 1 business day | 3 business days |
| Low (Enhancement Request) | 3 business days | Next release |

## **17. RISK MITIGATION**

### **17.1 Technical Risks**

| **Risk** | **Impact** | **Mitigation** |
| --- | --- | --- |
| OCR Quality | Low extraction accuracy | Multiple OCR engines, manual review |
| Excel Limits | Cannot handle large datasets | Pagination, summary sheets |
| Formula Complexity | Performance issues | Optimization, caching |
| Version Compatibility | Excel features not supported | Compatibility mode, fallbacks |

### **17.2 Operational Risks**

| **Risk** | **Impact** | **Mitigation** |
| --- | --- | --- |
| Ambiguous Requirements | Wrong KPIs extracted | Confidence scoring, review process |
| Document Quality | Cannot parse | Manual entry interface |
| Staff Training | Incorrect usage | Comprehensive documentation |
| Data Loss | Cannot recover | Automated backups, versioning |

## **18. APPENDICES**

### **Appendix A: Formula Library**

excel

# Core Formulas

Task Count: =COUNTA(range)-1

Status Count: =COUNTIF(range,"status")

Multi-Criteria: =COUNTIFS(range1,criteria1,range2,criteria2)

Percentage: =IFERROR(completed/total,0)

Monthly Sum: =SUMIFS(values,dates,">="&EOMONTH(TODAY(),-1)+1,dates,"<"&EOMONTH(TODAY(),0)+1)

Weighted Average: =SUMPRODUCT(values,weights)/SUM(weights)

Dynamic Lookup: =INDEX(return\_range,MATCH(lookup\_value,lookup\_range,0))

### **Appendix B: Color Codes**

yaml

Status Colors:

Completed: *#C6EFCE*

In Progress: *#FFEB9C*

Not Started: *#FFC7CE*

Blocked: *#D3D3D3*

Priority Colors:

High: *#FF6B6B*

Medium: *#FFD93D*

Low: *#6BCF7F*

Site Colors:

Site 1: *#FF6B6B*

Site 2: *#4ECDC4*

Site 3: *#45B7D1*

Site 4: *#96CEB4*

Site 5: *#FFEAA7*

### ****Appendix C: Named Range Definitions**

excel

Sites: =Reference Lists!$A$2:$A$20

Roles: =Reference Lists!$B$2:$B$20

Statuses: =Reference Lists!$C$2:$C$10

Priorities: =Reference Lists!$D$2:$D$5

RiskLevels: =Reference Lists!$E$2:$E$5

ContentTypes: =Reference Lists!$F$2:$F$15

EventTypes: =Reference Lists!$G$2:$G$10



## **19. VERSION HISTORY**

| **Version** | **Date** | **Changes** | **Author** |
| --- | --- | --- | --- |
| 2.0 | Jan 2025 | Integrated comprehensive requirements | Technical Team |
| 1.5 | Jan 2025 | Added acceptance tests and UDM | Program Team |
| 1.0 | Jan 2025 | Initial requirements document | Requirements Team |

## **20. APPROVAL & SIGN-OFF**

### **Approval Matrix**

| **Role** | **Name** | **Signature** | **Date** |
| --- | --- | --- | --- |
| Technical Lead | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |
| Program Manager | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |
| Data Engineer | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |
| Quality Assurance | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |
| Compliance Officer | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |
| Program Owner | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_\_\_\_\_\_\_\_\_ | \_\_\_\_\_ |

### **Document Control**

* **Classification:** Technical Specification - Production Ready
* **Distribution:** Program Team, Technical Team, Stakeholders
* **Review Cycle:** Quarterly
* **Next Review:** April 2025

*This integrated document represents the complete technical requirements for the Multi-Document Program Analyzer → Automated Excel KPI Dashboard Generator system. It combines best practices from both source documents to create a comprehensive, production-ready specification.*