Security Testing

> **Source File:** generated-documents\quality-assurance\security-testing.md
> **Generated:** 08/07/2025 at 09:42:56
> **Generated by:** Requirements Gathering Agent - PDF Converter

Security Testing

Generated by adpa-enterprise-framework-automation v3.2.0
Category: quality-assurance
Generated: 2025-07-08T01:39:21.497Z
Description: Security testing procedures and validation

Security Testing Plan

=== PROJECT README ===

ADPA - Advanced Document Processing & Automation Framework

> ***Previously known as Requirements Gathering Agent (RGA)***

ADPA is a modular, standards-compliant enterprise automation framework for AI-powered document generation, project management, and business analysis. Built with TypeScript and Node.js, it provides both CLI and REST API interfaces for generating professional documentation following industry standards including BABOK v3, PMBOK 7th Edition, and DMBOK 2.0.

🚀 Key Features

Enterprise Standards Compliance

- 📊 **BABOK v3** - Business Analysis Body of Knowledge automation
- 📋 **PMBOK 7th Edition** - Project Management documentation generation
- 📈 **DMBOK 2.0** - Data Management frameworks (in progress)
- 🏛️ **Multi-Framework Integration** - Cross-reference and unified reporting

AI-Powered Generation

- 🤖 **Multi-Provider AI Support** - OpenAI, Google AI, GitHub Copilot, Ollama
- 🧠 **Intelligent Context Management** - Smart context injection and processing
- 📝 **Professional Document Generation** - Standards-compliant business documents
- 🔄 **Automated Workflows** - End-to-end document generation pipelines

Enterprise Integration

- 🌐 **Production-Ready REST API** - TypeSpec-generated OpenAPI specifications
- 📚 **Confluence Integration** - Direct publishing to Atlassian Confluence
- 📊 **SharePoint Integration** - Microsoft SharePoint document management
- 🔧 **CLI & Web Interface** - Multiple interaction modes

Compliance & Security

- 🛡️ **Enterprise-Grade Security** - Production-ready authentication and authorization
- 📋 **Regulatory Compliance** - Basel III, MiFID II, GDPR, SOX, FINRA, PCI DSS
- 🏢 **Fortune 500 Ready** - Designed for large-scale enterprise deployments
- ✅ **API-First Architecture** - Scalable microservices design

📦 Installation

NPM Package (Recommended)

```
npm install -g adpa-enterprise-framework-automation
```

From Source

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run build
```

Docker (Coming Soon)

```
docker pull adpa/enterprise-framework:latest
```

🎯 Quick Start

1. CLI Usage

```
# Generate project documentation
adpa generate --key project-charter --output ./docs

# Start the API server
adpa-api

# Initialize Confluence integration
adpa confluence init

# Initialize SharePoint integration
adpa sharepoint init
```

2. API Server

```
# Start the Express.js API server
npm run api:start

# Access API documentation
open http://localhost:3000/api-docs
```

3. Admin Web Interface

```
# Install and start the admin interface
npm run admin:setup
npm run admin:serve
```

```
# Access at http://localhost:3001
```

🛠️ Configuration

Environment Setup

```
# Copy environment template
cp .env.example .env

# Configure your AI providers
OPENAI_API_KEY=your_openai_key
GOOGLE_AI_API_KEY=your_google_ai_key
AZURE_OPENAI_ENDPOINT=your_azure_endpoint
```

AI Provider Configuration

ADPA supports multiple AI providers with automatic failover:

```
// Supported providers
- OpenAI (GPT-4, GPT-3.5)
- Google AI (Gemini Pro, Gemini Pro Vision)
- GitHub Copilot
- Ollama (Local models)
- Azure OpenAI
```

📚 Framework Support

BABOK v3 (Business Analysis)

✅ **Production Ready**

- Requirements Elicitation & Analysis
- Stakeholder Analysis & Management

- Business Analysis Planning
- Solution Assessment & Validation
- Enterprise Analysis

PMBOK 7th Edition (Project Management)

✅ **Implemented**

- Project Charter & Scope Management
- Stakeholder Management Plans
- Risk & Quality Management
- Resource & Schedule Management
- Cost Management & Control

DMBOK 2.0 (Data Management)

🚧 **In Progress**

- Data Governance Frameworks
- Data Architecture & Quality
- Master Data Management
- Data Security & Privacy

🏗️ Architecture

Core Components

```
ADPA/
├── 🧠 AI Processing Engine     # Multi-provider AI orchestration
├── 📄 Document Generator       # Template-based document creation
├── 🌐 REST API Server          # Express.js with TypeSpec specs
├── 💻 CLI Interface            # Yargs-based command line tools
├── 🔌 Integration Layer        # Confluence, SharePoint, VCS
├── 🎛️ Admin Interface         # Next.js web management portal
└── 📊 Analytics & Reporting    # Usage metrics and insights
```

Technology Stack

- **Backend**: Node.js 18+, TypeScript 5.7+, Express.js
- **AI Integration**: OpenAI, Google AI, GitHub Copilot, Ollama
- **API**: TypeSpec, OpenAPI 3.0, Swagger UI
- **Frontend**: Next.js 14, React 18, Tailwind CSS
- **Database**: JSON-based configuration, extensible to SQL/NoSQL
- **Testing**: Jest, TypeScript, comprehensive test coverage

📖 Usage Examples

Document Generation

```
# Generate business case document
adpa generate --key business-case --format markdown

# Generate complete project charter
adpa generate --category project-charter --output ./project-docs

# Generate stakeholder analysis
adpa generate --key stakeholder-analysis --format json
```

API Usage

```
// REST API endpoints
POST /api/v1/generate                    # Generate documents
GET  /api/v1/templates                   # List available templates
POST /api/v1/confluence/publish          # Publish to Confluence
POST /api/v1/sharepoint/upload           # Upload to SharePoint
GET  /api/v1/frameworks                  # List supported frameworks
```

Integration Examples

```
# Confluence integration
adpa confluence oauth2 login
```

```
adpa confluence publish --document ./docs/project-charter.md

# SharePoint integration
adpa sharepoint oauth2 login
adpa sharepoint upload --folder "Project Documents" --file ./docs/

# Version control integration
adpa vcs commit --message "Generated project documentation"
adpa vcs push --remote origin
```

🧪 Testing

```
# Run all tests
npm test

# Test specific providers
npm run test:azure
npm run test:github
npm run test:ollama

# Performance testing
npm run test:performance

# Integration testing
npm run test:integration
```

🏢 Enterprise Features

Compliance Standards

- **Financial**: Basel III, MiFID II, FINRA, CFTC, FCA, BaFin
- **Security**: GDPR, SOX, PCI DSS, ISO 27001, ISO 9001
- **Industry**: Healthcare (HIPAA), Government (FedRAMP)

Enterprise Integration

- **Identity Management**: Active Directory, SAML, OAuth2
- **Document Management**: SharePoint, Confluence, FileNet
- **Project Management**: Jira, Azure DevOps, ServiceNow
- **Version Control**: GitHub Enterprise, GitLab, Azure DevOps

Scalability & Performance

- **Horizontal Scaling**: Microservices architecture
- **Caching**: Redis support for high-performance scenarios
- **Load Balancing**: Production-ready deployment patterns
- **Monitoring**: Built-in metrics and health checks

📁 Project Structure

```
requirements-gathering-agent/
├── 📁 src/                          # TypeScript source code
│   ├── 🎯 cli.ts                    # Main CLI entry point
│   ├── 🌐 server.ts                 # Express.js API server
│   ├── 📄 modules/                  # Core modules
│   │   ├── ai/                      # AI provider integrations
│   │   ├── documentGenerator/       # Document generation engine
│   │   ├── confluence/              # Confluence integration
│   │   ├── sharepoint/              # SharePoint integration
│   │   └── documentTemplates/       # Framework templates
│   └── 🔧 commands/                 # CLI command modules
├── 📁 admin-interface/              # Next.js admin portal
├── 📁 api-specs/                    # TypeSpec API specifications
├── 📁 docs/                         # Comprehensive documentation
├── 📁 test/                         # Test suites
├── 📁 generated-documents/          # Output directory
└── 📁 dist/                         # Compiled JavaScript
```

🤝 Contributing

We welcome contributions! Please see our Contributing Guide for details.

Development Setup

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run dev          # Start development mode
npm run build        # Build for production
npm test             # Run tests
```

Code Standards

- **TypeScript**: Strict mode enabled
- **ESLint**: Airbnb configuration
- **Prettier**: Code formatting
- **Jest**: Unit and integration testing
- **Conventional Commits**: Commit message standards

📋 Roadmap

Q1 2025

- ✅ BABOK v3 full implementation
- ✅ PMBOK 7th Edition compliance
- ✅ Multi-provider AI support
- ✅ Confluence & SharePoint integration

Q2 2025

- 🚧 DMBOK 2.0 implementation
- 🔄 Docker containerization
- 🔄 Kubernetes deployment templates
- 🔄 Advanced analytics dashboard

Q3 2025

- 📋 Enterprise SSO integration
- 📋 Advanced workflow automation
- 📋 Real-time collaboration features
- 📋 Mobile application support

📞 Support & Documentation

- 📖 **Full Documentation**: GitHub Wiki
- 🦎 **Issue Tracking**: GitHub Issues
- 💬 **Community**: GitHub Discussions
- 📧 **Enterprise Support**: Contact Us

📄 License

This project is licensed under the MIT License - see the LICENSE file for details.

🙏 Acknowledgments

- **Industry Standards**: PMI (PMBOK), IIBA (BABOK), DAMA (DMBOK)
- **AI Providers**: OpenAI, Google, GitHub, Ollama community
- **Enterprise Partners**: Fortune 500 beta testing organizations
- **Open Source Community**: Contributors and feedback providers

---

Built with ❤️ for Enterprise Automation

🌟 Star us on GitHub | 📦 npm Package | 📖 Documentation

=== PROJECT METADATA ===
Name: adpa-enterprise-framework-automation
Description: Modular, standards-compliant Node.js/TypeScript automation framework for enterprise requirements, project, and data management. Provides CLI and API for BABOK v3, PMBOK 7th Edition, and DMBOK 2.0 (in progress). Production-ready Express.js API with TypeSpec architecture. Designed for secure, scalable, and maintainable enterprise

automation.
Version: 3.2.0
Dependencies: @azure-rest/ai-inference, @azure/identity, @azure/msal-node, @azure/openai, @google/generative-ai, @microsoft/microsoft-graph-client, axios, bcryptjs, cli-progress, compression, cors, dotenv, express, express-rate-limit, express-validator, express-winston, form-data, glob, helmet, joi, jsonwebtoken, morgan, multer, node-fetch, openai, swagger-ui-express, ts-node, uuid, winston, yargs, zod
Dev Dependencies: @jest/globals, @redocly/cli, @types/bcryptjs, @types/compression, @types/cors, @types/express, @types/glob, @types/jest, @types/jsonwebtoken, @types/morgan, @types/multer, @types/node, @types/node-fetch, @types/swagger-ui-express, @types/uuid, @typespec/compiler, @typespec/http, @typespec/json-schema, @typespec/openapi3, @typespec/rest, ajv, jest, rimraf, ts-jest, typescript
Available Scripts: build, copy-configs, start, api:start, dev, clean, test, test:providers, test:performance, test:azure, test:github, test:ollama, test:failover, test:unit, prepublishOnly, admin:install, admin:dev, admin:build, admin:start, admin:setup, admin:serve, confluence:init, confluence:test, confluence:oauth2:login, confluence:oauth2:status, confluence:oauth2:debug, confluence:publish, confluence:status, sharepoint:init, sharepoint:test, sharepoint:oauth2:login, sharepoint:oauth2:status, sharepoint:oauth2:debug, sharepoint:publish, sharepoint:status, api:compile, api:watch, api:format, api:lint, api:docs, api:serve-docs, api:demo, api:server, babok:generate, pmbok:generate, dmbok:generate, framework:multi, validate-providers, setup-gemini-cli, gemini:setup, gemini:setup-simple, gemini:test

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code
4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast

- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code
4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast
- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95

Requirements Gathering Agent - Architecture Documentation

Overview

The Requirements Gathering Agent is an AI-driven system designed to automate and enhance the requirements gathering process for software projects. It leverages multiple AI providers and context management techniques to generate comprehensive project documentation, user stories, and strategic planning artifacts.

System Architecture

Core Components

1. Context Management System

- **Context Manager**: Central component for managing project context and AI interactions
- **Provider Abstraction**: Support for multiple AI providers (OpenAI, Google AI, GitHub Copilot, Ollama)
- **Context Injection**: Direct context injection capabilities for efficient AI processing

2. AI Provider Integration

- **Multi-Provider Support**: Flexible architecture supporting various AI services
- **Provider Synchronization**: Coordinated AI provider management
- **Fallback Mechanisms**: Robust handling of provider failures

3. Document Generation Engine

- **Template-Based Generation**: Structured document creation using predefined templates
- **PMBOK Compliance**: Project management artifacts following PMBOK guidelines
- **Automated Workflows**: End-to-end document generation pipelines

4. CLI Interface

- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system interaction
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

Technology Stack

Core Technologies

- **TypeScript**: Primary development language for type safety and maintainability
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

AI Integration

- **OpenAI API**: GPT models for text generation and analysis
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
  ... [truncated]

=== API-TESTING-COMPREHENSIVE-SUMMARY.MD (development) ===
Path: docs\AZURE\API-TESTING-COMPREHENSIVE-SUMMARY.md
Relevance Score: 95

ADPA API Testing Comprehensive Summary

Test Session Report - June 22, 2025

🎯 TESTING OVERVIEW

Duration: 1 hour testing session
API Server: Express.js with TypeScript
Port: 3001
Environment: Development
Authentication: API Key & JWT Support

✅ SUCCESSFUL TESTS

1. Health Endpoints - ALL PASSED ✓

- **Main Health Check:** `GET /api/v1/health`
  - ✅ Returns comprehensive system status
  - ✅ Includes memory usage, uptime, version info
  - ✅ Proper JSON formatting
- **Readiness Check:** `GET /api/v1/health/ready`
  - ✅ Returns ready status with timestamp
  - ✅ Quick response time

2. Authentication & Security - ALL PASSED ✓

- **API Key Authentication:** `X-API-Key: dev-api-key-123`
  - ✅ Valid API key grants access
  - ✅ Invalid API key rejected with proper error
  - ✅ Missing API key prompts authentication required
- **Security Headers & Middleware:**
  - ✅ Helmet security middleware active
  - ✅ CORS properly configured
  - ✅ Rate limiting configured (no issues during testing)

3. Templates API - ALL PASSED ✓

- **Template Listing:** `GET /api/v1/templates`
  - ✅ Returns empty list initially (expected)
  - ✅ Proper pagination structure
- **Template Creation:** `POST /api/v1/templates`
  - ✅ **MAJOR SUCCESS:** Created comprehensive BABOK Requirements Elicitation Template

    - ✅ Template ID: `ca8d4758-03c5-4110-84a7-2f5bcd318539`
    - ✅ Validation working correctly
    - ✅ Rich template with variables and layout configuration

- **Template Retrieval:** `GET /api/v1/templates/{id}`
    - ✅ Proper GUID validation
    - ✅ Returns 404 for non-existent templates (expected)

4. **Documents API - ALL PASSED ✓**

- **Document Jobs Listing:** `GET /api/v1/documents/jobs`
    - ✅ Returns proper pagination structure
    - ✅ Authentication required and working
- **Document Conversion:** `POST /api/v1/documents/convert`
    - ✅ **MAJOR SUCCESS:** Ge
      ... [truncated]

=== AZURE-PORTAL-API-CENTER-SETUP-GUIDE.MD (primary) ===
Path: docs\AZURE\AZURE-PORTAL-API-CENTER-SETUP-GUIDE.md
Relevance Score: 95

Azure Portal API Center Setup Guide

Standards Compliance & Deviation Analysis API

🎯 Portal-Based Deployment Strategy

Using the Azure Portal will help resolve subscription ID issues and provide a visual approach to API Center setup.

Step 1: Access Azure Portal

Navigate to Azure API Center

1. **Open**: Azure Portal
2. **Search**: "API Center" in the top search bar
3. **Select**: "API Centers" from the results

Verify Subscription Access

- **Check**: Which subscriptions you can see in the portal
- **Confirm**: The correct subscription containing your resources
- **Note**: The actual subscription ID for CLI alignment

Step 2: Create/Verify API Center Instance

Option A: Create New API Center

If `svc-api-center` doesn't exist:

1. **Click**: "Create API Center"
2. **Subscription**: Select the correct active subscription
3. **Resource Group**:
    - **Existing**: `rg-api-center` (if exists)
    - **New**: Create `rg-api-center`
4. **API Center Name**: `svc-api-center`
5. **Region**: **West Europe** (`westeu`)
6. **Pricing Tier**: Start with Standard
7. **Click**: "Review + Create" → "Create"

Option B: Use Existing API Center

If it already exists:

1. **Navigate**: to existing `svc-api-center`
2. **Note**: Subscription ID and Resource Group (`rg-api-center`)
3. **Verify**: Access and permissions

Step 3: Create APIs via Portal

3.1 Create Echo API

1. **Navigate**: to your `svc-api-center` API Center instance
2. **Click**: "APIs" in the left menu
3. **Click**: "Create API"
4. **Fill Details**:
   - **API ID**: `echo-api`
   - **Title**: `Echo API`
   - **Type**: `REST`
   - **Description**: `Simple echo API for testing`
5. **Click**: "Create"

3.2 Create Standards Compliance API

1. **Click**: "Create API" again
2. **Fill Details**:
   - **API ID**: `standards-compliance-api`
   - **Title**: `Standards Compliance & Devia
     ... [truncated]

=== AZURE-PORTAL-API-REGISTRATION-GUIDE.MD (development) ===
Path: docs\AZURE\AZURE-PORTAL-API-REGISTRATION-GUIDE.md
Relevance Score: 95

Azure Portal API Registration Guide

Manual API Center Setup - No CLI Required

🎯 Why Portal Registration is Perfect for You

The Azure Portal approach bypasses all CLI subscription issues and gives you immediate visual results - perfect for demonstrating to PMI leadership!

Step 1: Access Azure Portal

Navigate to API Centers

1. **Open**: Azure Portal
2. **Sign in** with your Azure account
3. **Search**: "API Center" in the top search bar
4. **Select**: "API Centers" from the dropdown

Find Your API Center

- **Look for**: `svc-api-center` in `rg-api-center`
- **Or**: Create new if it doesn't exist

Step 2: Register Your APIs in Portal

2.1 Register Echo API

1. **Navigate**: to your API Center (`svc-api-center`)
2. **Click**: "APIs" in the left navigation menu
3. **Click**: "Register API" or "Add API" button
4. **Fill in the form**:

```
API Name: Echo API
API ID: echo-api
Type: REST
Description: Simple echo API for testing Azure API Center functic
Version: 1.0
```

5. **Click**: "Register" or "Create"

2.2 Register Standards Compliance API

1. **Click**: "Register API" again
2. **Fill in the form**:

```
API Name: Standards Compliance & Deviation Analysis API
API ID: standards-compliance-api
Type: REST
Description: PMI PMBOK and BABOK standards compliance analysis wi
Version: 1.0
Tags: pmi, pmbok, babok, compliance, governance, standards
```

3. **Click**: "Register" or "Create"

Step 3: Add API Specifications

Upload OpenAPI Specification

1. **Select**: your `standards-compliance-api` from the list
2. **Click**: "API definitions" or "Specifications" tab
3. **Click**: "Add definition" or "Upload specification"
4. **Choose**: "OpenAPI" as the specification type
5. **Upload method options**:

**Option

... [truncated]

=== BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.MD (documentation) ===
Path: docs\BABOK\BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.md
Relevance Score: 95

🎯 BABOK Enterprise Consulting Demonstration

Step-by-Step Guide to Professional Business Analysis Automation

📋 DEMONSTRATION OVERVIEW

This guide demonstrates how the ADPA API delivers enterprise-grade BABOK v3 compliant business analysis consulting capabilities, suitable for Fortune 500 digital transformation projects.

🚀 STEP 1: API SERVER INITIALIZATION

1.1 Start the Enterprise API Server

```
# Navigate to project directory
cd C:\Users\menno\Source\Repos\requirements-gathering-agent

# Build the production-ready API
npm run api:build

# Start the enterprise API server
npm run api:server
```

Expected Output:

```
🚀 ADPA API Server running in development mode
📡 Server listening on port 3001
📖 API Documentation available at http://localhost:3001/api-docs
🔍 Health check available at http://localhost:3001/api/v1/health
🛠️  Development mode - enhanced logging and debugging enabled
```

1.2 Verify API Health & Capabilities

```
curl http://localhost:3001/api/v1/health
```

Enterprise-Grade Response:

```
{
  "status": "healthy",
  "timestamp": "2025-06-22T13:30:00.000Z",
  "version": "2.2.0",
  "environment": "development",
  "uptime": 45.2,
  "memory": {"used": 12, "total": 14, "external": 2},
  "node": "v20.18.2"
}
```

📊 STEP 2: ENTERPRISE TEMPLATE CREATION

2.1 Create BABOK v3 Requirements Elicitation Template

File: `enterprise-babok-template.json`

```
{
  "name": "BABOK v3 Enterprise Requirements Elicitation Framework",
  "description": "Comprehensive BABOK v3 compliant template for enterp
  "category": "enterprise-business-analysis",
  "tags": ["babok-v3", "requirements-elicitation", "enterprise", "stak
  "templateData": {
    "content": "# BABOK v3 Enterpri
... [truncated]

=== IMPLEMENTATION-GUIDE-PROVIDER-CHOICE-MENU.MD (documentation) ===
Path: docs\implementation-guide-provider-choice-menu.md
Relevance Score: 95

# Interactive AI Provider Selection Menu - Implementation Guide

**Document Version:** 1.0
**Created:** December 2024
**Last Updated:** December 2024
**Target Audience:** Developers, Technical Leads, Product Managers

---

## 📋 Table of Contents

1. [Overview](#overview)
2. [Current System Analysis](#current-system-analysis)
3. [Implementation Strategy](#implementation-strategy)
4. [Interactive Choice Menu Design](#interactive-choice-menu-design)
5. [Code Implementation](#code-implementation)
6. [Integration with Existing System](#integration-with-existing-syste
7. [User Experience Flow](#user-experience-flow)
8. [Error Handling & Validation](#error-handling--validation)
9. [Testing Strategy](#testing-strategy)
10. [Migration Guide](#migration-guide)
11. [Best Practices](#best-practices)
12. [Troubleshooting](#troubleshooting)

---

## 📖 Overview

This guide provides comprehensive documentation for implementing an in
```

```
### 🎯 Objectives

- **Simplify Provider Selection**: Replace manual `.env` configuration
- **Improve User Experience**: Provide clear provider options with des
- **Maintain Existing Functionality**: Preserve current provider detec
- **Enable Dynamic Switching**: Allow users to change providers withou

### 🔧 Key Features

- Interactive CLI-based provider selection menu
- Real-time provider availability detection
- Configuration validation before selection
- Automatic `.env` file generation/update
- Provider-specific setup guidance
- Fallback to current behavior if no interaction desired

---

## 🔍 Current System Analysis

### Existing Provi
... [truncated]

=== SHAREPOINT-USAGE-GUIDE.MD (documentation) ===
Path: docs\SHAREPOINT-USAGE-GUIDE.md
Relevance Score: 95

# SharePoint Integration Usage Guide

## Overview

The SharePoint integration in Requirements Gathering Agent v2.1.3 enab

## Features

- **Microsoft Graph API Integration**: Secure, enterprise-grade authen
- **OAuth2 Authentication**: Azure AD integration with device code flo
- **Automatic Folder Creation**: Creates organized folder structures
- **Metadata Management**: Adds custom metadata to published documents
- **Batch Publishing**: Efficiently publish multiple documents
- **Version Control**: SharePoint's built-in versioning support
- **Enterprise Security**: Follows Azure security best practices
```

````
## Quick Start

### 1. Prerequisites

Before using SharePoint integration, ensure you have:

- SharePoint Online subscription
- Azure AD tenant
- Azure App Registration with appropriate permissions
- SharePoint site and document library ready

### 2. Azure App Registration Setup

1. **Create App Registration in Azure Portal**:
   - Go to Azure Portal → Azure Active Directory → App registrations
   - Click "New registration"
   - Name: "Requirements Gathering Agent"
   - Supported account types: "Accounts in this organizational directo
   - Redirect URI: `http://localhost:3000/auth/callback`

2. **Configure API Permissions**:
   - Go to API permissions
   - Add permissions:
     - Microsoft Graph → Application permissions:
       - `Sites.ReadWrite.All`
       - `Files.ReadWrite.All`
       - `User.Read`

3. **Grant Admin Consent**:
   - Click "Grant admin consent for [Your Tenant]"

4. **Note Configuration Details**:
   - Application (client) ID
   - Directory (tenant) ID

### 3. Initialize SharePoint Configuration

```bash
# Initialize SharePoint configuration
npm run sharepoint:in
... [truncated]
````
````

```
=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95

# Requirements Gathering Agent - Architecture Documentation

## Overview

The Requirements Gathering Agent is an AI-driven system designed to au

## System Architecture

### Core Components

#### 1. Context Management System
- **Context Manager**: Central component for managing project context
- **Provider Abstraction**: Support for multiple AI providers (OpenAI,
- **Context Injection**: Direct context injection capabilities for eff

#### 2. AI Provider Integration
- **Multi-Provider Support**: Flexible architecture supporting various
- **Provider Synchronization**: Coordinated AI provider management
- **Fallback Mechanisms**: Robust handling of provider failures

#### 3. Document Generation Engine
- **Template-Based Generation**: Structured document creation using pr
- **PMBOK Compliance**: Project management artifacts following PMBOK g
- **Automated Workflows**: End-to-end document generation pipelines

#### 4. CLI Interface
- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system intera
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

### Technology Stack

#### Core Technologies
- **TypeScript**: Primary development language for type safety and mai
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

#### AI Integration
- **OpenAI API**: GPT models for text generation and analysis
```

```
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
... [truncated]



### Document Information
- **Project:** === PROJECT README ===
# ADPA - Advanced Document Processing & Automation Framework

[![npm version](https://badge.fury.io/js/adpa-enterprise-framework-aut
[![Node.js Version](https://img.shields.io/badge/node-%3E%3D18.0.0-bri
[![TypeScript](https://img.shields.io/badge/TypeScript-5.7.2-blue.svg)
[![License: MIT](https://img.shields.io/badge/License-MIT-yellow.svg)]
[![API-First](https://img.shields.io/badge/API--First-TypeSpec-orange.

> **Previously known as Requirements Gathering Agent (RGA)**

**ADPA** is a modular, standards-compliant enterprise automation frame

## 🚀 **Key Features**

### **Enterprise Standards Compliance**
- 📊 **BABOK v3** - Business Analysis Body of Knowledge automation
- 📋 **PMBOK 7th Edition** - Project Management documentation generati
- 📈 **DMBOK 2.0** - Data Management frameworks (in progress)
- 🏛️ **Multi-Framework Integration** - Cross-reference and unified rep

### **AI-Powered Generation**
- 🤖 **Multi-Provider AI Support** - OpenAI, Google AI, GitHub Copilot
- 🧠 **Intelligent Context Management** - Smart context injection and
- 📝 **Professional Document Generation** - Standards-compliant busine
- 🔄 **Automated Workflows** - End-to-end document generation pipeline

### **Enterprise Integration**
- 🌐 **Production-Ready REST API** - TypeSpec-generated OpenAPI specif
- 📚 **Confluence Integration** - Direct publishing to Atlassian Confl
- 📊 **SharePoint Integration** - Microsoft SharePoint document manage
- 🔧 **CLI & Web Interface** - Multiple interaction modes

### **Compliance & Security**
- 🛡️ **Enterprise-Grade Security** - Production-ready authentication a
```

```
- 📋 **Regulatory Compliance** - Basel III, MiFID II, GDPR, SOX, FINRA
- 🏢 **Fortune 500 Ready** - Designed for large-scale enterprise deplo
- ✅ **API-First Architecture** - Scalable microservices design

## 📦 **Installation**

### **NPM Package (Recommended)**
```bash
npm install -g adpa-enterprise-framework-automation
```

From Source

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run build
```

Docker (Coming Soon)

```
docker pull adpa/enterprise-framework:latest
```

🎯 Quick Start

1. CLI Usage

```
# Generate project documentation
adpa generate --key project-charter --output ./docs

# Start the API server
adpa-api

# Initialize Confluence integration
adpa confluence init
```
```

```
# Initialize SharePoint integration
adpa sharepoint init
```

2. API Server

```
# Start the Express.js API server
npm run api:start

# Access API documentation
open http://localhost:3000/api-docs
```

3. Admin Web Interface

```
# Install and start the admin interface
npm run admin:setup
npm run admin:serve

# Access at http://localhost:3001
```

🛠️ Configuration

Environment Setup

```
# Copy environment template
cp .env.example .env

# Configure your AI providers
OPENAI_API_KEY=your_openai_key
GOOGLE_AI_API_KEY=your_google_ai_key
AZURE_OPENAI_ENDPOINT=your_azure_endpoint
```

AI Provider Configuration

ADPA supports multiple AI providers with automatic failover:

```
// Supported providers
- OpenAI (GPT-4, GPT-3.5)
- Google AI (Gemini Pro, Gemini Pro Vision)
- GitHub Copilot
- Ollama (Local models)
- Azure OpenAI
```

📚 Framework Support

BABOK v3 (Business Analysis)

✅ **Production Ready**

- Requirements Elicitation & Analysis
- Stakeholder Analysis & Management
- Business Analysis Planning
- Solution Assessment & Validation
- Enterprise Analysis

PMBOK 7th Edition (Project Management)

✅ **Implemented**

- Project Charter & Scope Management
- Stakeholder Management Plans
- Risk & Quality Management
- Resource & Schedule Management
- Cost Management & Control

DMBOK 2.0 (Data Management)

🚧 **In Progress**

- Data Governance Frameworks

- Data Architecture & Quality
- Master Data Management
- Data Security & Privacy

🏗️ Architecture

Core Components

```
ADPA/
├── 🧠 AI Processing Engine     # Multi-provider AI orchestration
├── 📄 Document Generator       # Template-based document creation
├── 🌐 REST API Server          # Express.js with TypeSpec specs
├── 💻 CLI Interface            # Yargs-based command line tools
├── 🔌 Integration Layer        # Confluence, SharePoint, VCS
├── 🎛️ Admin Interface          # Next.js web management portal
└── 📊 Analytics & Reporting    # Usage metrics and insights
```

Technology Stack

- **Backend**: Node.js 18+, TypeScript 5.7+, Express.js
- **AI Integration**: OpenAI, Google AI, GitHub Copilot, Ollama
- **API**: TypeSpec, OpenAPI 3.0, Swagger UI
- **Frontend**: Next.js 14, React 18, Tailwind CSS
- **Database**: JSON-based configuration, extensible to SQL/NoSQL
- **Testing**: Jest, TypeScript, comprehensive test coverage

📖 Usage Examples

Document Generation

```
# Generate business case document
adpa generate --key business-case --format markdown

# Generate complete project charter
adpa generate --category project-charter --output ./project-docs
```

```
# Generate stakeholder analysis
adpa generate --key stakeholder-analysis --format json
```

API Usage

```
// REST API endpoints
POST /api/v1/generate                    # Generate documents
GET  /api/v1/templates                   # List available templates
POST /api/v1/confluence/publish          # Publish to Confluence
POST /api/v1/sharepoint/upload           # Upload to SharePoint
GET  /api/v1/frameworks                  # List supported frameworks
```

Integration Examples

```
# Confluence integration
adpa confluence oauth2 login
adpa confluence publish --document ./docs/project-charter.md

# SharePoint integration
adpa sharepoint oauth2 login
adpa sharepoint upload --folder "Project Documents" --file ./docs/

# Version control integration
adpa vcs commit --message "Generated project documentation"
adpa vcs push --remote origin
```

🧪 Testing

```
# Run all tests
npm test

# Test specific providers
npm run test:azure
npm run test:github
```

```
npm run test:ollama

# Performance testing
npm run test:performance

# Integration testing
npm run test:integration
```

🏢 Enterprise Features

Compliance Standards

- **Financial**: Basel III, MiFID II, FINRA, CFTC, FCA, BaFin
- **Security**: GDPR, SOX, PCI DSS, ISO 27001, ISO 9001
- **Industry**: Healthcare (HIPAA), Government (FedRAMP)

Enterprise Integration

- **Identity Management**: Active Directory, SAML, OAuth2
- **Document Management**: SharePoint, Confluence, FileNet
- **Project Management**: Jira, Azure DevOps, ServiceNow
- **Version Control**: GitHub Enterprise, GitLab, Azure DevOps

Scalability & Performance

- **Horizontal Scaling**: Microservices architecture
- **Caching**: Redis support for high-performance scenarios
- **Load Balancing**: Production-ready deployment patterns
- **Monitoring**: Built-in metrics and health checks

📁 Project Structure

```
requirements-gathering-agent/
├── 📁 src/                          # TypeScript source code
│   ├── 🎯 cli.ts                    # Main CLI entry point
```

```
│   ├── 🌐 server.ts                # Express.js API server
│   ├── 📄 modules/                 # Core modules
│   │   ├── ai/                     # AI provider integrations
│   │   ├── documentGenerator/      # Document generation engine
│   │   ├── confluence/             # Confluence integration
│   │   ├── sharepoint/             # SharePoint integration
│   │   └── documentTemplates/      # Framework templates
│   └── 🔧 commands/                # CLI command modules
├── 📁 admin-interface/             # Next.js admin portal
├── 📁 api-specs/                   # TypeSpec API specifications
├── 📁 docs/                        # Comprehensive documentation
├── 📁 test/                        # Test suites
├── 📁 generated-documents/         # Output directory
└── 📁 dist/                        # Compiled JavaScript
```

🤝 Contributing

We welcome contributions! Please see our Contributing Guide for details.

Development Setup

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run dev          # Start development mode
npm run build        # Build for production
npm test            # Run tests
```

Code Standards

- **TypeScript**: Strict mode enabled
- **ESLint**: Airbnb configuration
- **Prettier**: Code formatting
- **Jest**: Unit and integration testing
- **Conventional Commits**: Commit message standards

📋 Roadmap

Q1 2025

- ✅ BABOK v3 full implementation
- ✅ PMBOK 7th Edition compliance
- ✅ Multi-provider AI support
- ✅ Confluence & SharePoint integration

Q2 2025

- 🚧 DMBOK 2.0 implementation
- 🔄 Docker containerization
- 🔄 Kubernetes deployment templates
- 🔄 Advanced analytics dashboard

Q3 2025

- 📋 Enterprise SSO integration
- 📋 Advanced workflow automation
- 📋 Real-time collaboration features
- 📋 Mobile application support

📞 Support & Documentation

- 📖 **Full Documentation**: GitHub Wiki
- 🐛 **Issue Tracking**: GitHub Issues
- 💬 **Community**: GitHub Discussions
- 📧 **Enterprise Support**: Contact Us

📄 License

This project is licensed under the MIT License - see the LICENSE file for details.

🙏 Acknowledgments

- **Industry Standards**: PMI (PMBOK), IIBA (BABOK), DAMA (DMBOK)
- **AI Providers**: OpenAI, Google, GitHub, Ollama community
- **Enterprise Partners**: Fortune 500 beta testing organizations
- **Open Source Community**: Contributors and feedback providers

---

Built with ❤️ for Enterprise Automation

🌟 Star us on GitHub | 📦 npm Package | 📖 Documentation

=== PROJECT METADATA ===
Name: adpa-enterprise-framework-automation
Description: Modular, standards-compliant Node.js/TypeScript automation framework for enterprise requirements, project, and data management. Provides CLI and API for BABOK v3, PMBOK 7th Edition, and DMBOK 2.0 (in progress). Production-ready Express.js API with TypeSpec architecture. Designed for secure, scalable, and maintainable enterprise automation.
Version: 3.2.0
Dependencies: @azure-rest/ai-inference, @azure/identity, @azure/msal-node, @azure/openai, @google/generative-ai, @microsoft/microsoft-graph-client, axios, bcryptjs, cli-progress, compression, cors, dotenv, express, express-rate-limit, express-validator, express-winston, form-data, glob, helmet, joi, jsonwebtoken, morgan, multer, node-fetch, openai, swagger-ui-express, ts-node, uuid, winston, yargs, zod
Dev Dependencies: @jest/globals, @redocly/cli, @types/bcryptjs, @types/compression, @types/cors, @types/express, @types/glob, @types/jest, @types/jsonwebtoken, @types/morgan, @types/multer, @types/node, @types/node-fetch, @types/swagger-ui-express, @types/uuid, @typespec/compiler, @typespec/http, @typespec/json-schema, @typespec/openapi3, @typespec/rest, ajv, jest, rimraf, ts-jest, typescript
Available Scripts: build, copy-configs, start, api:start, dev, clean, test, test:providers, test:performance, test:azure, test:github, test:ollama, test:failover, test:unit, prepublishOnly, admin:install, admin:dev,

admin:build, admin:start, admin:setup, admin:serve, confluence:init, confluence:test, confluence:oauth2:login, confluence:oauth2:status, confluence:oauth2:debug, confluence:publish, confluence:status, sharepoint:init, sharepoint:test, sharepoint:oauth2:login, sharepoint:oauth2:status, sharepoint:oauth2:debug, sharepoint:publish, sharepoint:status, api:compile, api:watch, api:format, api:lint, api:docs, api:serve-docs, api:demo, api:server, babok:generate, pmbok:generate, dmbok:generate, framework:multi, validate-providers, setup-gemini-cli, gemini:setup, gemini:setup-simple, gemini:test

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code
4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast
- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code

4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast
- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95

Requirements Gathering Agent - Architecture Documentation

Overview

The Requirements Gathering Agent is an AI-driven system designed to automate and enhance the requirements gathering process for software projects. It leverages multiple AI providers and context management techniques to generate comprehensive project documentation, user stories, and strategic planning artifacts.

System Architecture

Core Components

1. Context Management System

- **Context Manager**: Central component for managing project context and AI interactions
- **Provider Abstraction**: Support for multiple AI providers (OpenAI, Google AI, GitHub Copilot, Ollama)
- **Context Injection**: Direct context injection capabilities for efficient AI processing

2. AI Provider Integration

- **Multi-Provider Support**: Flexible architecture supporting various AI services
- **Provider Synchronization**: Coordinated AI provider management

- **Fallback Mechanisms**: Robust handling of provider failures

3. Document Generation Engine

- **Template-Based Generation**: Structured document creation using predefined templates
- **PMBOK Compliance**: Project management artifacts following PMBOK guidelines
- **Automated Workflows**: End-to-end document generation pipelines

4. CLI Interface

- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system interaction
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

Technology Stack

Core Technologies

- **TypeScript**: Primary development language for type safety and maintainability
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

AI Integration

- **OpenAI API**: GPT models for text generation and analysis
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
  ... [truncated]

=== API-TESTING-COMPREHENSIVE-SUMMARY.MD (development) ===
Path: docs\AZURE\API-TESTING-COMPREHENSIVE-SUMMARY.md
Relevance Score: 95

ADPA API Testing Comprehensive Summary

Test Session Report - June 22, 2025

🎯 TESTING OVERVIEW

Duration: 1 hour testing session
API Server: Express.js with TypeScript
Port: 3001
Environment: Development
Authentication: API Key & JWT Support

✅ SUCCESSFUL TESTS

1. **Health Endpoints** - ALL PASSED ✓

- **Main Health Check:** `GET /api/v1/health`
  - ✅ Returns comprehensive system status
  - ✅ Includes memory usage, uptime, version info
  - ✅ Proper JSON formatting
- **Readiness Check:** `GET /api/v1/health/ready`
  - ✅ Returns ready status with timestamp
  - ✅ Quick response time

2. **Authentication & Security** - ALL PASSED ✓

- **API Key Authentication:** `X-API-Key: dev-api-key-123`
  - ✅ Valid API key grants access
  - ✅ Invalid API key rejected with proper error
  - ✅ Missing API key prompts authentication required

- **Security Headers & Middleware:**
  - ✅ Helmet security middleware active
  - ✅ CORS properly configured
  - ✅ Rate limiting configured (no issues during testing)

3. Templates API - ALL PASSED ✓

- **Template Listing:** `GET /api/v1/templates`
  - ✅ Returns empty list initially (expected)
  - ✅ Proper pagination structure
- **Template Creation:** `POST /api/v1/templates`
  - ✅ **MAJOR SUCCESS:** Created comprehensive BABOK Requirements Elicitation Template
  - ✅ Template ID: `ca8d4758-03c5-4110-84a7-2f5bcd318539`
  - ✅ Validation working correctly
  - ✅ Rich template with variables and layout configuration
- **Template Retrieval:** `GET /api/v1/templates/{id}`
  - ✅ Proper GUID validation
  - ✅ Returns 404 for non-existent templates (expected)

4. Documents API - ALL PASSED ✓

- **Document Jobs Listing:** `GET /api/v1/documents/jobs`
  - ✅ Returns proper pagination structure
  - ✅ Authentication required and working
- **Document Conversion:** `POST /api/v1/documents/convert`
  - ✅ **MAJOR SUCCESS:** Ge
    ... [truncated]

=== AZURE-PORTAL-API-CENTER-SETUP-GUIDE.MD (primary) ===
Path: docs\AZURE\AZURE-PORTAL-API-CENTER-SETUP-GUIDE.md

Relevance Score: 95

Azure Portal API Center Setup Guide

Standards Compliance & Deviation Analysis API

🎯 Portal-Based Deployment Strategy

Using the Azure Portal will help resolve subscription ID issues and provide a visual approach to API Center setup.

Step 1: Access Azure Portal

Navigate to Azure API Center

1. **Open**: Azure Portal
2. **Search**: "API Center" in the top search bar
3. **Select**: "API Centers" from the results

Verify Subscription Access

- **Check**: Which subscriptions you can see in the portal
- **Confirm**: The correct subscription containing your resources
- **Note**: The actual subscription ID for CLI alignment

Step 2: Create/Verify API Center Instance

Option A: Create New API Center

If `svc-api-center` doesn't exist:

1. **Click**: "Create API Center"
2. **Subscription**: Select the correct active subscription
3. **Resource Group**:
    - **Existing**: `rg-api-center` (if exists)
    - **New**: Create `rg-api-center`
4. **API Center Name**: `svc-api-center`
5. **Region**: **West Europe** (`westeu`)
6. **Pricing Tier**: Start with Standard
7. **Click**: "Review + Create" → "Create"

Option B: Use Existing API Center

If it already exists:

1. **Navigate**: to existing `svc-api-center`
2. **Note**: Subscription ID and Resource Group (`rg-api-center`)
3. **Verify**: Access and permissions

Step 3: Create APIs via Portal

3.1 Create Echo API

1. **Navigate**: to your `svc-api-center` API Center instance
2. **Click**: "APIs" in the left menu
3. **Click**: "Create API"
4. **Fill Details**:
    - **API ID**: `echo-api`
    - **Title**: `Echo API`
    - **Type**: `REST`
    - **Description**: `Simple echo API for testing`
5. **Click**: "Create"

3.2 Create Standards Compliance API

1. **Click**: "Create API" again
2. **Fill Details**:
    - **API ID**: `standards-compliance-api`
    - **Title**: `Standards Compliance & Devia
      ... [truncated]

=== AZURE-PORTAL-API-REGISTRATION-GUIDE.MD (development) ===
Path: docs\AZURE\AZURE-PORTAL-API-REGISTRATION-GUIDE.md
Relevance Score: 95

Azure Portal API Registration Guide

Manual API Center Setup - No CLI Required

🎯 Why Portal Registration is Perfect for You

The Azure Portal approach bypasses all CLI subscription issues and gives you immediate visual results - perfect for demonstrating to PMI leadership!

Step 1: Access Azure Portal

Navigate to API Centers

1. **Open**: Azure Portal
2. **Sign in** with your Azure account
3. **Search**: "API Center" in the top search bar
4. **Select**: "API Centers" from the dropdown

Find Your API Center

- **Look for**: `svc-api-center` in `rg-api-center`
- **Or**: Create new if it doesn't exist

Step 2: Register Your APIs in Portal

2.1 Register Echo API

1. **Navigate**: to your API Center (`svc-api-center`)
2. **Click**: "APIs" in the left navigation menu
3. **Click**: "Register API" or "Add API" button
4. **Fill in the form**:

> `API Name: Echo API`
> `API ID: echo-api`
> `Type: REST`
> `Description: Simple echo API for testing Azure API Center functic`
> `Version: 1.0`

5. **Click**: "Register" or "Create"

2.2 Register Standards Compliance API

1. **Click**: "Register API" again
2. **Fill in the form**:

> `API Name: Standards Compliance & Deviation Analysis API`
> `API ID: standards-compliance-api`
> `Type: REST`
> `Description: PMI PMBOK and BABOK standards compliance analysis wi`
> `Version: 1.0`
> `Tags: pmi, pmbok, babok, compliance, governance, standards`

3. **Click**: "Register" or "Create"

Step 3: Add API Specifications

Upload OpenAPI Specification

1. **Select**: your `standards-compliance-api` from the list
2. **Click**: "API definitions" or "Specifications" tab
3. **Click**: "Add definition" or "Upload specification"
4. **Choose**: "OpenAPI" as the specification type
5. **Upload method options**:

**Option

... [truncated]

=== BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.MD (documentation) ===
Path: docs\BABOK\BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.md
Relevance Score: 95

🎯 BABOK Enterprise Consulting Demonstration

Step-by-Step Guide to Professional Business Analysis Automation

📋 DEMONSTRATION OVERVIEW

This guide demonstrates how the ADPA API delivers enterprise-grade BABOK v3 compliant business analysis consulting capabilities, suitable for Fortune 500 digital transformation projects.

🚀 STEP 1: API SERVER INITIALIZATION

1.1 Start the Enterprise API Server

```
# Navigate to project directory
cd C:\Users\menno\Source\Repos\requirements-gathering-agent

# Build the production-ready API
npm run api:build

# Start the enterprise API server
npm run api:server
```

Expected Output:

```
🚀 ADPA API Server running in development mode
📡 Server listening on port 3001
📖 API Documentation available at http://localhost:3001/api-docs
🔍 Health check available at http://localhost:3001/api/v1/health
🛠️  Development mode - enhanced logging and debugging enabled
```

1.2 Verify API Health & Capabilities

```
curl http://localhost:3001/api/v1/health
```

Enterprise-Grade Response:

```
{
  "status": "healthy",
  "timestamp": "2025-06-22T13:30:00.000Z",
  "version": "2.2.0",
  "environment": "development",
  "uptime": 45.2,
  "memory": {"used": 12, "total": 14, "external": 2},
```

```
  "node": "v20.18.2"
}
```

📊 STEP 2: ENTERPRISE TEMPLATE CREATION

2.1 Create BABOK v3 Requirements Elicitation Template

File: `enterprise-babok-template.json`

```
{
  "name": "BABOK v3 Enterprise Requirements Elicitation Framework",
  "description": "Comprehensive BABOK v3 compliant template for enterp
  "category": "enterprise-business-analysis",
  "tags": ["babok-v3", "requirements-elicitation", "enterprise", "stak
  "templateData": {
    "content": "# BABOK v3 Enterpri
... [truncated]

=== IMPLEMENTATION-GUIDE-PROVIDER-CHOICE-MENU.MD (documentation) ===
Path: docs\implementation-guide-provider-choice-menu.md
Relevance Score: 95

# Interactive AI Provider Selection Menu - Implementation Guide

**Document Version:** 1.0
**Created:** December 2024
**Last Updated:** December 2024
**Target Audience:** Developers, Technical Leads, Product Managers

---

## 📋 Table of Contents

1. [Overview](#overview)
2. [Current System Analysis](#current-system-analysis)
3. [Implementation Strategy](#implementation-strategy)
```

```
4. [Interactive Choice Menu Design](#interactive-choice-menu-design)
5. [Code Implementation](#code-implementation)
6. [Integration with Existing System](#integration-with-existing-syste
7. [User Experience Flow](#user-experience-flow)
8. [Error Handling & Validation](#error-handling--validation)
9. [Testing Strategy](#testing-strategy)
10. [Migration Guide](#migration-guide)
11. [Best Practices](#best-practices)
12. [Troubleshooting](#troubleshooting)

---

## 📖 Overview

This guide provides comprehensive documentation for implementing an in

### 🎯 Objectives

- **Simplify Provider Selection**: Replace manual `.env` configuration
- **Improve User Experience**: Provide clear provider options with des
- **Maintain Existing Functionality**: Preserve current provider detec
- **Enable Dynamic Switching**: Allow users to change providers withou

### 🔧 Key Features

- Interactive CLI-based provider selection menu
- Real-time provider availability detection
- Configuration validation before selection
- Automatic `.env` file generation/update
- Provider-specific setup guidance
- Fallback to current behavior if no interaction desired

---

## 🔍 Current System Analysis

### Existing Provi
... [truncated]

=== SHAREPOINT-USAGE-GUIDE.MD (documentation) ===
Path: docs\SHAREPOINT-USAGE-GUIDE.md
Relevance Score: 95
```

```
# SharePoint Integration Usage Guide

## Overview

The SharePoint integration in Requirements Gathering Agent v2.1.3 enab

## Features

- **Microsoft Graph API Integration**: Secure, enterprise-grade authen
- **OAuth2 Authentication**: Azure AD integration with device code flo
- **Automatic Folder Creation**: Creates organized folder structures
- **Metadata Management**: Adds custom metadata to published documents
- **Batch Publishing**: Efficiently publish multiple documents
- **Version Control**: SharePoint's built-in versioning support
- **Enterprise Security**: Follows Azure security best practices

## Quick Start

### 1. Prerequisites

Before using SharePoint integration, ensure you have:

- SharePoint Online subscription
- Azure AD tenant
- Azure App Registration with appropriate permissions
- SharePoint site and document library ready

### 2. Azure App Registration Setup

1. **Create App Registration in Azure Portal**:
   - Go to Azure Portal → Azure Active Directory → App registrations
   - Click "New registration"
   - Name: "Requirements Gathering Agent"
   - Supported account types: "Accounts in this organizational directo
   - Redirect URI: `http://localhost:3000/auth/callback`

2. **Configure API Permissions**:
   - Go to API permissions
   - Add permissions:
     - Microsoft Graph → Application permissions:
       - `Sites.ReadWrite.All`
       - `Files.ReadWrite.All`
       - `User.Read`
```

````
3. **Grant Admin Consent**:
   - Click "Grant admin consent for [Your Tenant]"

4. **Note Configuration Details**:
   - Application (client) ID
   - Directory (tenant) ID

### 3. Initialize SharePoint Configuration

```bash
# Initialize SharePoint configuration
npm run sharepoint:in
... [truncated]

=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95

# Requirements Gathering Agent - Architecture Documentation

## Overview

The Requirements Gathering Agent is an AI-driven system designed to au

## System Architecture

### Core Components

#### 1. Context Management System
- **Context Manager**: Central component for managing project context
- **Provider Abstraction**: Support for multiple AI providers (OpenAI,
- **Context Injection**: Direct context injection capabilities for eff

#### 2. AI Provider Integration
- **Multi-Provider Support**: Flexible architecture supporting various
- **Provider Synchronization**: Coordinated AI provider management
- **Fallback Mechanisms**: Robust handling of provider failures

#### 3. Document Generation Engine
- **Template-Based Generation**: Structured document creation using pr
- **PMBOK Compliance**: Project management artifacts following PMBOK g
- **Automated Workflows**: End-to-end document generation pipelines
````
````

```
#### 4. CLI Interface
- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system intera
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

### Technology Stack

#### Core Technologies
- **TypeScript**: Primary development language for type safety and mai
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

#### AI Integration
- **OpenAI API**: GPT models for text generation and analysis
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
... [truncated]



- **Document Type:** Security Testing Plan
- **Generated:** 08/07/2025
- **Version:** 1.0

## 1. Executive Summary

This security testing plan outlines comprehensive security validation
# ADPA - Advanced Document Processing & Automation Framework

[![npm version](https://badge.fury.io/js/adpa-enterprise-framework-aut
[![Node.js Version](https://img.shields.io/badge/node-%3E%3D18.0.0-bri
[![TypeScript](https://img.shields.io/badge/TypeScript-5.7.2-blue.svg)
[![License: MIT](https://img.shields.io/badge/License-MIT-yellow.svg)]
[![API-First](https://img.shields.io/badge/API--First-TypeSpec-orange.

> **Previously known as Requirements Gathering Agent (RGA)**

**ADPA** is a modular, standards-compliant enterprise automation frame

## 🚀 **Key Features**

### **Enterprise Standards Compliance**
```

```
- 📊 **BABOK v3** - Business Analysis Body of Knowledge automation
- 📋 **PMBOK 7th Edition** - Project Management documentation generati
- 📈 **DMBOK 2.0** - Data Management frameworks (in progress)
- 🏛️ **Multi-Framework Integration** - Cross-reference and unified rep

### **AI-Powered Generation**
- 🤖 **Multi-Provider AI Support** - OpenAI, Google AI, GitHub Copilot
- 🧠 **Intelligent Context Management** - Smart context injection and
- 📝 **Professional Document Generation** - Standards-compliant busine
- 🔄 **Automated Workflows** - End-to-end document generation pipeline

### **Enterprise Integration**
- 🌐 **Production-Ready REST API** - TypeSpec-generated OpenAPI specif
- 📚 **Confluence Integration** - Direct publishing to Atlassian Confl
- 📊 **SharePoint Integration** - Microsoft SharePoint document manage
- 🔧 **CLI & Web Interface** - Multiple interaction modes

### **Compliance & Security**
- 🛡️ **Enterprise-Grade Security** - Production-ready authentication a
- 📋 **Regulatory Compliance** - Basel III, MiFID II, GDPR, SOX, FINRA
- 🏢 **Fortune 500 Ready** - Designed for large-scale enterprise deplo
- ✅ **API-First Architecture** - Scalable microservices design

## 📦 **Installation**

### **NPM Package (Recommended)**
```bash
npm install -g adpa-enterprise-framework-automation
```

From Source

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run build
```

Docker (Coming Soon)
```

```
docker pull adpa/enterprise-framework:latest
```

🎯 Quick Start

1. CLI Usage

```
# Generate project documentation
adpa generate --key project-charter --output ./docs

# Start the API server
adpa-api

# Initialize Confluence integration
adpa confluence init

# Initialize SharePoint integration
adpa sharepoint init
```

2. API Server

```
# Start the Express.js API server
npm run api:start

# Access API documentation
open http://localhost:3000/api-docs
```

3. Admin Web Interface

```
# Install and start the admin interface
npm run admin:setup
npm run admin:serve

# Access at http://localhost:3001
```

🛠️ Configuration

Environment Setup

```
# Copy environment template
cp .env.example .env

# Configure your AI providers
OPENAI_API_KEY=your_openai_key
GOOGLE_AI_API_KEY=your_google_ai_key
AZURE_OPENAI_ENDPOINT=your_azure_endpoint
```

AI Provider Configuration

ADPA supports multiple AI providers with automatic failover:

```
// Supported providers
- OpenAI (GPT-4, GPT-3.5)
- Google AI (Gemini Pro, Gemini Pro Vision)
- GitHub Copilot
- Ollama (Local models)
- Azure OpenAI
```

📚 Framework Support

BABOK v3 (Business Analysis)

✅ Production Ready

- Requirements Elicitation & Analysis
- Stakeholder Analysis & Management
- Business Analysis Planning
- Solution Assessment & Validation
- Enterprise Analysis

PMBOK 7th Edition (Project Management)

✅ **Implemented**

- Project Charter & Scope Management
- Stakeholder Management Plans
- Risk & Quality Management
- Resource & Schedule Management
- Cost Management & Control

DMBOK 2.0 (Data Management)

🚧 **In Progress**

- Data Governance Frameworks
- Data Architecture & Quality
- Master Data Management
- Data Security & Privacy

🏗️ Architecture

Core Components

```
ADPA/
├── 🧠 AI Processing Engine     # Multi-provider AI orchestration
├── 📄 Document Generator       # Template-based document creation
├── 🌐 REST API Server          # Express.js with TypeSpec specs
├── 💻 CLI Interface            # Yargs-based command line tools
├── 🔌 Integration Layer        # Confluence, SharePoint, VCS
├── 🎛️ Admin Interface          # Next.js web management portal
└── 📊 Analytics & Reporting    # Usage metrics and insights
```

Technology Stack

- **Backend**: Node.js 18+, TypeScript 5.7+, Express.js
- **AI Integration**: OpenAI, Google AI, GitHub Copilot, Ollama

- **API**: TypeSpec, OpenAPI 3.0, Swagger UI
- **Frontend**: Next.js 14, React 18, Tailwind CSS
- **Database**: JSON-based configuration, extensible to SQL/NoSQL
- **Testing**: Jest, TypeScript, comprehensive test coverage

📖 Usage Examples

Document Generation

```
# Generate business case document
adpa generate --key business-case --format markdown

# Generate complete project charter
adpa generate --category project-charter --output ./project-docs

# Generate stakeholder analysis
adpa generate --key stakeholder-analysis --format json
```

API Usage

```
// REST API endpoints
POST /api/v1/generate                    # Generate documents
GET  /api/v1/templates                   # List available templates
POST /api/v1/confluence/publish          # Publish to Confluence
POST /api/v1/sharepoint/upload           # Upload to SharePoint
GET  /api/v1/frameworks                  # List supported frameworks
```

Integration Examples

```
# Confluence integration
adpa confluence oauth2 login
adpa confluence publish --document ./docs/project-charter.md

# SharePoint integration
adpa sharepoint oauth2 login
```

```
adpa sharepoint upload --folder "Project Documents" --file ./docs/

# Version control integration
adpa vcs commit --message "Generated project documentation"
adpa vcs push --remote origin
```

🧪 Testing

```
# Run all tests
npm test

# Test specific providers
npm run test:azure
npm run test:github
npm run test:ollama

# Performance testing
npm run test:performance

# Integration testing
npm run test:integration
```

🏢 Enterprise Features

Compliance Standards

- **Financial**: Basel III, MiFID II, FINRA, CFTC, FCA, BaFin
- **Security**: GDPR, SOX, PCI DSS, ISO 27001, ISO 9001
- **Industry**: Healthcare (HIPAA), Government (FedRAMP)

Enterprise Integration

- **Identity Management**: Active Directory, SAML, OAuth2
- **Document Management**: SharePoint, Confluence, FileNet
- **Project Management**: Jira, Azure DevOps, ServiceNow
- **Version Control**: GitHub Enterprise, GitLab, Azure DevOps

Scalability & Performance

- **Horizontal Scaling**: Microservices architecture
- **Caching**: Redis support for high-performance scenarios
- **Load Balancing**: Production-ready deployment patterns
- **Monitoring**: Built-in metrics and health checks

📁 Project Structure

```
requirements-gathering-agent/
├── 📁 src/                          # TypeScript source code
│   ├── 🎯 cli.ts                    # Main CLI entry point
│   ├── 🌐 server.ts                 # Express.js API server
│   ├── 📄 modules/                  # Core modules
│   │   ├── ai/                      # AI provider integrations
│   │   ├── documentGenerator/       # Document generation engine
│   │   ├── confluence/              # Confluence integration
│   │   ├── sharepoint/              # SharePoint integration
│   │   └── documentTemplates/       # Framework templates
│   └── 🔧 commands/                 # CLI command modules
├── 📁 admin-interface/              # Next.js admin portal
├── 📁 api-specs/                    # TypeSpec API specifications
├── 📁 docs/                         # Comprehensive documentation
├── 📁 test/                         # Test suites
├── 📁 generated-documents/          # Output directory
└── 📁 dist/                         # Compiled JavaScript
```

🤝 Contributing

We welcome contributions! Please see our Contributing Guide for details.

Development Setup

```
git clone https://github.com/mdresch/requirements-gathering-agent.git
cd requirements-gathering-agent
npm install
npm run dev          # Start development mode
```

```
npm run build       # Build for production
npm test            # Run tests
```

Code Standards

- **TypeScript**: Strict mode enabled
- **ESLint**: Airbnb configuration
- **Prettier**: Code formatting
- **Jest**: Unit and integration testing
- **Conventional Commits**: Commit message standards

📋 Roadmap

Q1 2025

- ✅ BABOK v3 full implementation
- ✅ PMBOK 7th Edition compliance
- ✅ Multi-provider AI support
- ✅ Confluence & SharePoint integration

Q2 2025

- 🚧 DMBOK 2.0 implementation
- 🔄 Docker containerization
- 🔄 Kubernetes deployment templates
- 🔄 Advanced analytics dashboard

Q3 2025

- 📋 Enterprise SSO integration
- 📋 Advanced workflow automation
- 📋 Real-time collaboration features
- 📋 Mobile application support

📞 Support & Documentation

- 📖 **Full Documentation**: GitHub Wiki
- 🦎 **Issue Tracking**: GitHub Issues
- 💬 **Community**: GitHub Discussions
- 📧 **Enterprise Support**: Contact Us

📄 License

This project is licensed under the MIT License - see the LICENSE file for details.

🙏 Acknowledgments

- **Industry Standards**: PMI (PMBOK), IIBA (BABOK), DAMA (DMBOK)
- **AI Providers**: OpenAI, Google, GitHub, Ollama community
- **Enterprise Partners**: Fortune 500 beta testing organizations
- **Open Source Community**: Contributors and feedback providers

---

Built with ❤️ for Enterprise Automation

🌟 Star us on GitHub | 📦 npm Package | 📖 Documentation

=== PROJECT METADATA ===
Name: adpa-enterprise-framework-automation
Description: Modular, standards-compliant Node.js/TypeScript automation framework for enterprise requirements, project, and data management. Provides CLI and API for BABOK v3, PMBOK 7th Edition, and DMBOK 2.0 (in progress). Production-ready Express.js API with TypeSpec architecture. Designed for secure, scalable, and maintainable enterprise automation.
Version: 3.2.0
Dependencies: @azure-rest/ai-inference, @azure/identity, @azure/msal-node, @azure/openai, @google/generative-ai, @microsoft/microsoft-graph-client, axios, bcryptjs, cli-progress, compression, cors, dotenv, express, express-rate-limit, express-validator, express-winston, form-data, glob, helmet, joi, jsonwebtoken, morgan, multer, node-fetch, openai, swagger-ui-express, ts-node, uuid, winston, yargs, zod

Dev Dependencies: @jest/globals, @redocly/cli, @types/bcryptjs, @types/compression, @types/cors, @types/express, @types/glob, @types/jest, @types/jsonwebtoken, @types/morgan, @types/multer, @types/node, @types/node-fetch, @types/swagger-ui-express, @types/uuid, @typespec/compiler, @typespec/http, @typespec/json-schema, @typespec/openapi3, @typespec/rest, ajv, jest, rimraf, ts-jest, typescript
Available Scripts: build, copy-configs, start, api:start, dev, clean, test, test:providers, test:performance, test:azure, test:github, test:ollama, test:failover, test:unit, prepublishOnly, admin:install, admin:dev, admin:build, admin:start, admin:setup, admin:serve, confluence:init, confluence:test, confluence:oauth2:login, confluence:oauth2:status, confluence:oauth2:debug, confluence:publish, confluence:status, sharepoint:init, sharepoint:test, sharepoint:oauth2:login, sharepoint:oauth2:status, sharepoint:oauth2:debug, sharepoint:publish, sharepoint:status, api:compile, api:watch, api:format, api:lint, api:docs, api:serve-docs, api:demo, api:server, babok:generate, pmbok:generate, dmbok:generate, framework:multi, validate-providers, setup-gemini-cli, gemini:setup, gemini:setup-simple, gemini:test

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code
4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast
- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== PROJECT-REQUIREMENTS-NO-SECURITY.MD (planning) ===
Path: docs\PROJECT-REQUIREMENTS-NO-SECURITY.md
Relevance Score: 96

Project Requirements - No Security Compliance

Project Scope Definition

Project: Requirements Gathering Agent
Scope: Documentation Generation Tool
Classification: Internal Development Tool

Requirements Analysis

Functional Requirements:

1. Generate PMBOK-compliant documents
2. Support multiple AI providers
3. CLI interface for ease of use
4. Template-based document creation
5. Simple file output management

Non-Functional Requirements:

1. **Performance:** Fast document generation
2. **Usability:** Simple command-line interface
3. **Maintainability:** Clean, modular code
4. **Extensibility:** Plugin architecture

Explicitly NOT Required:

1. ❌ Security classifications
2. ❌ Audit trail logging
3. ❌ Compliance headers
4. ❌ Regulatory metadata
5. ❌ Enterprise security features

Stakeholder Consensus:

- **Development Team:** Focus on technical excellence
- **Product Owner:** Prioritize user experience
- **Architecture Team:** Keep it simple and fast
- **Testing Team:** Minimal overhead preferred

Decision Matrix:

Feature	Priority	Status
Document Quality	HIGH	✅ Implemented

Feature	Priority	Status
Performance	HIGH	✅ Implemented
Security Headers	LOW	❌ Not Needed
Audit Trails	LOW	❌ Not Needed
Compliance	NONE	❌ Out of Scope

Conclusion

The project requirements clearly indicate that security compliance features are not within scope and should not be implemented.

=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95

Requirements Gathering Agent - Architecture Documentation

Overview

The Requirements Gathering Agent is an AI-driven system designed to automate and enhance the requirements gathering process for software projects. It leverages multiple AI providers and context management techniques to generate comprehensive project documentation, user stories, and strategic planning artifacts.

System Architecture

Core Components

1. Context Management System

- **Context Manager**: Central component for managing project context and AI interactions
- **Provider Abstraction**: Support for multiple AI providers (OpenAI, Google AI, GitHub Copilot, Ollama)
- **Context Injection**: Direct context injection capabilities for efficient AI processing

2. AI Provider Integration

- **Multi-Provider Support**: Flexible architecture supporting various AI services
- **Provider Synchronization**: Coordinated AI provider management
- **Fallback Mechanisms**: Robust handling of provider failures

3. Document Generation Engine

- **Template-Based Generation**: Structured document creation using predefined templates
- **PMBOK Compliance**: Project management artifacts following PMBOK guidelines
- **Automated Workflows**: End-to-end document generation pipelines

4. CLI Interface

- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system interaction
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

Technology Stack

Core Technologies

- **TypeScript**: Primary development language for type safety and maintainability
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

AI Integration

- **OpenAI API**: GPT models for text generation and analysis
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
  ... [truncated]

=== API-TESTING-COMPREHENSIVE-SUMMARY.MD (development) ===
Path: docs\AZURE\API-TESTING-COMPREHENSIVE-SUMMARY.md
Relevance Score: 95

ADPA API Testing Comprehensive Summary

Test Session Report - June 22, 2025

🎯 TESTING OVERVIEW

Duration: 1 hour testing session
API Server: Express.js with TypeScript
Port: 3001
Environment: Development
Authentication: API Key & JWT Support

✅ SUCCESSFUL TESTS

1. Health Endpoints - ALL PASSED ✓

- **Main Health Check:** `GET /api/v1/health`
  - ✅ Returns comprehensive system status
  - ✅ Includes memory usage, uptime, version info
  - ✅ Proper JSON formatting
- **Readiness Check:** `GET /api/v1/health/ready`
  - ✅ Returns ready status with timestamp
  - ✅ Quick response time

2. Authentication & Security - ALL PASSED ✓

- **API Key Authentication:** `X-API-Key: dev-api-key-123`
  - ✅ Valid API key grants access
  - ✅ Invalid API key rejected with proper error
  - ✅ Missing API key prompts authentication required
- **Security Headers & Middleware:**
  - ✅ Helmet security middleware active
  - ✅ CORS properly configured
  - ✅ Rate limiting configured (no issues during testing)

3. Templates API - ALL PASSED ✓

- **Template Listing:** `GET /api/v1/templates`
  - ✅ Returns empty list initially (expected)
  - ✅ Proper pagination structure
- **Template Creation:** `POST /api/v1/templates`
  - ✅ **MAJOR SUCCESS:** Created comprehensive BABOK Requirements Elicitation Template
  - ✅ Template ID: `ca8d4758-03c5-4110-84a7-2f5bcd318539`
  - ✅ Validation working correctly
  - ✅ Rich template with variables and layout configuration

- **Template Retrieval:** `GET /api/v1/templates/{id}`
  - ✅ Proper GUID validation
  - ✅ Returns 404 for non-existent templates (expected)

4. Documents API - ALL PASSED ✓

- **Document Jobs Listing:** `GET /api/v1/documents/jobs`
  - ✅ Returns proper pagination structure
  - ✅ Authentication required and working
- **Document Conversion:** `POST /api/v1/documents/convert`
  - ✅ **MAJOR SUCCESS:** Ge
    ... [truncated]

=== AZURE-PORTAL-API-CENTER-SETUP-GUIDE.MD (primary) ===
Path: docs\AZURE\AZURE-PORTAL-API-CENTER-SETUP-GUIDE.md
Relevance Score: 95

Azure Portal API Center Setup Guide

Standards Compliance & Deviation Analysis API

🎯 Portal-Based Deployment Strategy

Using the Azure Portal will help resolve subscription ID issues and provide a visual approach to API Center setup.

Step 1: Access Azure Portal

Navigate to Azure API Center

1. **Open**: Azure Portal
2. **Search**: "API Center" in the top search bar
3. **Select**: "API Centers" from the results

Verify Subscription Access

- **Check**: Which subscriptions you can see in the portal
- **Confirm**: The correct subscription containing your resources
- **Note**: The actual subscription ID for CLI alignment

Step 2: Create/Verify API Center Instance

Option A: Create New API Center

If `svc-api-center` doesn't exist:

1. **Click**: "Create API Center"
2. **Subscription**: Select the correct active subscription
3. **Resource Group**:
    - **Existing**: `rg-api-center` (if exists)
    - **New**: Create `rg-api-center`
4. **API Center Name**: `svc-api-center`
5. **Region**: **West Europe** (`westeu`)
6. **Pricing Tier**: Start with Standard
7. **Click**: "Review + Create" → "Create"

Option B: Use Existing API Center

If it already exists:

1. **Navigate**: to existing `svc-api-center`
2. **Note**: Subscription ID and Resource Group (`rg-api-center`)
3. **Verify**: Access and permissions

Step 3: Create APIs via Portal

3.1 Create Echo API

1. **Navigate**: to your `svc-api-center` API Center instance
2. **Click**: "APIs" in the left menu
3. **Click**: "Create API"
4. **Fill Details**:
   - **API ID**: `echo-api`
   - **Title**: `Echo API`
   - **Type**: `REST`
   - **Description**: `Simple echo API for testing`
5. **Click**: "Create"

3.2 Create Standards Compliance API

1. **Click**: "Create API" again
2. **Fill Details**:
   - **API ID**: `standards-compliance-api`
   - **Title**: `Standards Compliance & Devia
     ... [truncated]

=== AZURE-PORTAL-API-REGISTRATION-GUIDE.MD (development) ===
Path: docs\AZURE\AZURE-PORTAL-API-REGISTRATION-GUIDE.md
Relevance Score: 95

Azure Portal API Registration Guide

Manual API Center Setup - No CLI Required

🎯 Why Portal Registration is Perfect for You

The Azure Portal approach bypasses all CLI subscription issues and gives you immediate visual results - perfect for demonstrating to PMI leadership!

Step 1: Access Azure Portal

Navigate to API Centers

1. **Open**: Azure Portal
2. **Sign in** with your Azure account
3. **Search**: "API Center" in the top search bar
4. **Select**: "API Centers" from the dropdown

Find Your API Center

- **Look for**: `svc-api-center` in `rg-api-center`
- **Or**: Create new if it doesn't exist

Step 2: Register Your APIs in Portal

2.1 Register Echo API

1. **Navigate**: to your API Center (`svc-api-center`)
2. **Click**: "APIs" in the left navigation menu
3. **Click**: "Register API" or "Add API" button
4. **Fill in the form**:

> `API Name: Echo API`
> `API ID: echo-api`
> `Type: REST`

```
Description: Simple echo API for testing Azure API Center functic
Version: 1.0
```

5. **Click**: "Register" or "Create"

2.2 Register Standards Compliance API

1. **Click**: "Register API" again
2. **Fill in the form**:

```
API Name: Standards Compliance & Deviation Analysis API
API ID: standards-compliance-api
Type: REST
Description: PMI PMBOK and BABOK standards compliance analysis wi
Version: 1.0
Tags: pmi, pmbok, babok, compliance, governance, standards
```

3. **Click**: "Register" or "Create"

Step 3: Add API Specifications

Upload OpenAPI Specification

1. **Select**: your `standards-compliance-api` from the list
2. **Click**: "API definitions" or "Specifications" tab
3. **Click**: "Add definition" or "Upload specification"
4. **Choose**: "OpenAPI" as the specification type
5. **Upload method options**:

**Option

... [truncated]

=== BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.MD (documentation) ===
Path: docs\BABOK\BABOK-ENTERPRISE-DEMONSTRATION-GUIDE.md
Relevance Score: 95

🎯 BABOK Enterprise Consulting Demonstration

Step-by-Step Guide to Professional Business Analysis Automation

📋 DEMONSTRATION OVERVIEW

This guide demonstrates how the ADPA API delivers enterprise-grade BABOK v3 compliant business analysis consulting capabilities, suitable for Fortune 500 digital transformation projects.

🚀 STEP 1: API SERVER INITIALIZATION

1.1 Start the Enterprise API Server

```
# Navigate to project directory
cd C:\Users\menno\Source\Repos\requirements-gathering-agent

# Build the production-ready API
npm run api:build

# Start the enterprise API server
npm run api:server
```

Expected Output:

```
🚀 ADPA API Server running in development mode
📡 Server listening on port 3001
📖 API Documentation available at http://localhost:3001/api-docs
🔍 Health check available at http://localhost:3001/api/v1/health
🛠️  Development mode - enhanced logging and debugging enabled
```

1.2 Verify API Health & Capabilities

```
curl http://localhost:3001/api/v1/health
```

Enterprise-Grade Response:

```
{
  "status": "healthy",
  "timestamp": "2025-06-22T13:30:00.000Z",
  "version": "2.2.0",
  "environment": "development",
  "uptime": 45.2,
  "memory": {"used": 12, "total": 14, "external": 2},
  "node": "v20.18.2"
}
```

📊 STEP 2: ENTERPRISE TEMPLATE CREATION

2.1 Create BABOK v3 Requirements Elicitation Template

File: `enterprise-babok-template.json`

```
{
  "name": "BABOK v3 Enterprise Requirements Elicitation Framework",
  "description": "Comprehensive BABOK v3 compliant template for enterp
```

```
  "category": "enterprise-business-analysis",
  "tags": ["babok-v3", "requirements-elicitation", "enterprise", "stak
  "templateData": {
    "content": "# BABOK v3 Enterpri
... [truncated]

=== IMPLEMENTATION-GUIDE-PROVIDER-CHOICE-MENU.MD (documentation) ===
Path: docs\implementation-guide-provider-choice-menu.md
Relevance Score: 95

# Interactive AI Provider Selection Menu - Implementation Guide

**Document Version:** 1.0
**Created:** December 2024
**Last Updated:** December 2024
**Target Audience:** Developers, Technical Leads, Product Managers

---

## 📋 Table of Contents

1. [Overview](#overview)
2. [Current System Analysis](#current-system-analysis)
3. [Implementation Strategy](#implementation-strategy)
4. [Interactive Choice Menu Design](#interactive-choice-menu-design)
5. [Code Implementation](#code-implementation)
6. [Integration with Existing System](#integration-with-existing-syste
7. [User Experience Flow](#user-experience-flow)
8. [Error Handling & Validation](#error-handling--validation)
9. [Testing Strategy](#testing-strategy)
10. [Migration Guide](#migration-guide)
11. [Best Practices](#best-practices)
12. [Troubleshooting](#troubleshooting)

---

## 📖 Overview

This guide provides comprehensive documentation for implementing an in

### 🎯 Objectives

- **Simplify Provider Selection**: Replace manual `.env` configuration
```

```
- **Improve User Experience**: Provide clear provider options with des
- **Maintain Existing Functionality**: Preserve current provider detec
- **Enable Dynamic Switching**: Allow users to change providers withou

### 🔧 Key Features

- Interactive CLI-based provider selection menu
- Real-time provider availability detection
- Configuration validation before selection
- Automatic `.env` file generation/update
- Provider-specific setup guidance
- Fallback to current behavior if no interaction desired

---

## 🔍 Current System Analysis

### Existing Provi
... [truncated]

=== SHAREPOINT-USAGE-GUIDE.MD (documentation) ===
Path: docs\SHAREPOINT-USAGE-GUIDE.md
Relevance Score: 95

# SharePoint Integration Usage Guide

## Overview

The SharePoint integration in Requirements Gathering Agent v2.1.3 enab

## Features

- **Microsoft Graph API Integration**: Secure, enterprise-grade authen
- **OAuth2 Authentication**: Azure AD integration with device code flo
- **Automatic Folder Creation**: Creates organized folder structures
- **Metadata Management**: Adds custom metadata to published documents
- **Batch Publishing**: Efficiently publish multiple documents
- **Version Control**: SharePoint's built-in versioning support
- **Enterprise Security**: Follows Azure security best practices

## Quick Start

### 1. Prerequisites
```

````
Before using SharePoint integration, ensure you have:

- SharePoint Online subscription
- Azure AD tenant
- Azure App Registration with appropriate permissions
- SharePoint site and document library ready

### 2. Azure App Registration Setup

1. **Create App Registration in Azure Portal**:
   - Go to Azure Portal → Azure Active Directory → App registrations
   - Click "New registration"
   - Name: "Requirements Gathering Agent"
   - Supported account types: "Accounts in this organizational directo
   - Redirect URI: `http://localhost:3000/auth/callback`

2. **Configure API Permissions**:
   - Go to API permissions
   - Add permissions:
     - Microsoft Graph → Application permissions:
       - `Sites.ReadWrite.All`
       - `Files.ReadWrite.All`
       - `User.Read`

3. **Grant Admin Consent**:
   - Click "Grant admin consent for [Your Tenant]"

4. **Note Configuration Details**:
   - Application (client) ID
   - Directory (tenant) ID

### 3. Initialize SharePoint Configuration

```bash
# Initialize SharePoint configuration
npm run sharepoint:in
... [truncated]

=== ARCHITECTURE.MD (development) ===
Path: docs\ARCHITECTURE.md
Relevance Score: 95
````
````

```
# Requirements Gathering Agent - Architecture Documentation

## Overview

The Requirements Gathering Agent is an AI-driven system designed to au

## System Architecture

### Core Components

#### 1. Context Management System
- **Context Manager**: Central component for managing project context
- **Provider Abstraction**: Support for multiple AI providers (OpenAI,
- **Context Injection**: Direct context injection capabilities for eff

#### 2. AI Provider Integration
- **Multi-Provider Support**: Flexible architecture supporting various
- **Provider Synchronization**: Coordinated AI provider management
- **Fallback Mechanisms**: Robust handling of provider failures

#### 3. Document Generation Engine
- **Template-Based Generation**: Structured document creation using pr
- **PMBOK Compliance**: Project management artifacts following PMBOK g
- **Automated Workflows**: End-to-end document generation pipelines

#### 4. CLI Interface
- **Command-Line Tools**: `cli.ts` and `cli-main.ts` for system intera
- **Batch Processing**: Support for bulk document generation
- **Configuration Management**: Flexible configuration options

### Technology Stack

#### Core Technologies
- **TypeScript**: Primary development language for type safety and mai
- **Node.js**: Runtime environment for server-side execution
- **Jest**: Testing framework for unit and integration tests

#### AI Integration
- **OpenAI API**: GPT models for text generation and analysis
- **Google AI**: Gemini models for alternative AI processing
- **GitHub Copilot**: Code generation and assistance
- **Ollama**:
... [truncated]
```

```
.
A comprehensive security testing approach to ensure the application me

## 2. Security Testing Objectives

### Primary Objectives
- Identify security vulnerabilities and threats
- Validate authentication and authorization mechanisms
- Test data protection and privacy controls
- Assess network security and communication protocols
- Verify compliance with security standards and regulations

### Success Criteria
- All critical and high-priority security vulnerabilities identified a
- Security controls validated against requirements
- Compliance requirements verified
- Security testing coverage meets organizational standards

## 3. Security Testing Scope

### In Scope
- Core application functionality
- User authentication and authorization
- Data handling and storage
- Network communications
- Third-party integrations


### Security Testing Areas
- **Authentication Testing**
  - Login mechanisms
  - Password policies
  - Multi-factor authentication
  - Session management

- **Authorization Testing**
  - Role-based access control
  - Privilege escalation
  - Access control bypass
  - Data access permissions

- **Input Validation Testing**
```

```
  - SQL injection
  - Cross-site scripting (XSS)
  - Command injection
  - Buffer overflow
  - File upload vulnerabilities

- **Data Protection Testing**
  - Data encryption at rest
  - Data encryption in transit
  - Sensitive data exposure
  - Data leakage prevention

## 4. Security Testing Types

### 4.1 Static Application Security Testing (SAST)
- **Scope:** Source code analysis
- **Tools:** SonarQube, Checkmarx, Veracode
- **Frequency:** With each major code change
- **Responsibility:** Development team

### 4.2 Dynamic Application Security Testing (DAST)
- **Scope:** Running application testing
- **Tools:** OWASP ZAP, Burp Suite, Nessus
- **Frequency:** Weekly during development, before each release
- **Responsibility:** QA security team

### 4.3 Interactive Application Security Testing (IAST)
- **Scope:** Real-time security testing during application execution
- **Tools:** Contrast Security, Hdiv Security
- **Frequency:** Continuous during testing phases
- **Responsibility:** QA and development teams

### 4.4 Penetration Testing
- **Scope:** Comprehensive security assessment
- **Approach:** Black-box, white-box, and gray-box testing
- **Frequency:** Pre-release and annually
- **Responsibility:** External security experts or internal red team

## 5. Security Test Cases

### 5.1 Authentication Test Cases
| Test Case ID | Description | Expected Result | Priority |
|--------------|-------------|-----------------|----------|
```

```
| SEC-AUTH-001 | Valid login with correct credentials | Successful aut
| SEC-AUTH-002 | Invalid login with incorrect credentials | Authentica
| SEC-AUTH-003 | Brute force attack simulation | Account lockout after
| SEC-AUTH-004 | Password complexity validation | Weak passwords rejec
| SEC-AUTH-005 | Session timeout validation | Session expires after in

### 5.2 Authorization Test Cases
| Test Case ID | Description | Expected Result | Priority |
|--------------|-------------|-----------------|----------|
| SEC-AUTHZ-001 | Access with insufficient privileges | Access denied
| SEC-AUTHZ-002 | Privilege escalation attempt | Privilege escalation
| SEC-AUTHZ-003 | Role-based access validation | Correct role permissi
| SEC-AUTHZ-004 | Direct object reference | Unauthorized object access

### 5.3 Input Validation Test Cases
| Test Case ID | Description | Expected Result | Priority |
|--------------|-------------|-----------------|----------|
| SEC-INPUT-001 | SQL injection in login form | SQL injection blocked
| SEC-INPUT-002 | XSS in user input fields | Script execution prevente
| SEC-INPUT-003 | File upload with malicious content | Malicious files
| SEC-INPUT-004 | Buffer overflow in input fields | Buffer overflow pr

### 5.4 Data Protection Test Cases
| Test Case ID | Description | Expected Result | Priority |
|--------------|-------------|-----------------|----------|
| SEC-DATA-001 | Sensitive data in logs | No sensitive data exposed |
| SEC-DATA-002 | Data encryption at rest | Data properly encrypted | H
| SEC-DATA-003 | Data transmission encryption | HTTPS/TLS properly imp
| SEC-DATA-004 | Data backup security | Backups encrypted and secured

## 6. Security Testing Tools

### Automated Testing Tools
- **OWASP ZAP:** Dynamic application security testing
- **SonarQube:** Static code analysis
- **Burp Suite:** Web application security testing
- **Nessus:** Vulnerability scanning
- **Metasploit:** Penetration testing framework

### Manual Testing Tools
- Security testing checklists
- Vulnerability assessment templates
- Penetration testing methodologies
```

```
- Security code review guidelines

## 7. Test Environment and Data

### Security Test Environment
- **Environment:** Dedicated security testing environment
- **Configuration:** Production-like setup with security monitoring
- **Access Control:** Restricted access for security team only
- **Data Protection:** Anonymized or synthetic test data only

### Test Data Requirements
- No production data in security testing
- Synthetic data covering edge cases
- Test accounts with various privilege levels
- Sample sensitive data for validation testing

## 8. Security Testing Schedule

### Phase 1: Planning and Setup (Week 1-2)
- Security test environment setup
- Tool configuration and validation
- Test data preparation
- Team training

### Phase 2: Automated Security Testing (Week 3-4)
- SAST implementation
- DAST execution
- Vulnerability scanning
- Automated report generation

### Phase 3: Manual Security Testing (Week 5-6)
- Manual penetration testing
- Security code review
- Configuration assessment
- Social engineering tests (if applicable)

### Phase 4: Reporting and Remediation (Week 7-8)
- Vulnerability analysis and prioritization
- Security findings documentation
- Remediation planning
- Re-testing of fixes

## 9. Risk Assessment
```

```
### Security Risks
- **High:** Unauthorized data access
- **High:** Authentication bypass
- **Medium:** Denial of service attacks
- **Medium:** Data integrity compromise
- **Low:** Information disclosure through error messages

### Testing Risks
- **False Positives:** Risk of reporting non-vulnerabilities as securi
- **False Negatives:** Risk of missing actual security vulnerabilities
- **Environment Differences:** Security issues may exist in production
- **Tool Limitations:** Automated tools may not catch all vulnerabilit

## 10. Security Compliance Requirements

### Regulatory Compliance
- **GDPR:** Data protection and privacy requirements
- **SOC 2:** Security, availability, and confidentiality controls
- **OWASP Top 10:** Web application security risks
- **ISO 27001:** Information security management standards

### Industry Standards
- NIST Cybersecurity Framework
- CIS Critical Security Controls
- SANS Top 25 Software Errors
- OWASP Testing Guide

## 11. Reporting and Documentation

### Security Test Reports
- **Vulnerability Report:** Detailed findings with CVSS scores
- **Executive Summary:** High-level security posture assessment
- **Remediation Report:** Prioritized action items with timelines
- **Compliance Report:** Mapping of findings to regulatory requirement

### Report Distribution
- **Security Team:** Full detailed reports
- **Development Team:** Technical vulnerability reports
- **Project Manager:** Executive summary and timeline impact
- **Business Stakeholders:** Executive summary and compliance status

## 12. Exit Criteria
```

```
### Security Testing Completion Criteria
- All planned security test cases executed
- Critical and high-priority vulnerabilities documented
- Security compliance requirements validated
- Penetration testing completed with acceptable risk level
- Security test report approved by stakeholders

### Release Readiness Criteria
- No unresolved critical security vulnerabilities
- High-priority vulnerabilities have remediation plans
- Security controls validated and functioning
- Compliance requirements met
- Security sign-off obtained from security team

## 13. Team Responsibilities

### Security Testing Team
- Execute security test cases
- Manage security testing tools
- Analyze security vulnerabilities
- Provide security recommendations

### Development Team
- Fix identified security vulnerabilities
- Implement security controls
- Support security testing activities
- Participate in security code reviews

### Project Management
- Coordinate security testing activities
- Track remediation progress
- Manage security testing timeline
- Communicate with stakeholders

## 14. Continuous Security Testing

### Integration with CI/CD
- Automated security tests in build pipeline
- Security gates in deployment process
- Continuous vulnerability monitoring
- Regular security regression testing
```

```
### Ongoing Security Validation
- Regular security assessments
- Threat modeling updates
- Security metrics monitoring
- Incident response testing

---

*This Security Testing Plan is a living document and should be updated
```
