# Test Cases

> **Source File:** generated-documents\quality-assurance\test-cases.md
> **Generated:** 08/07/2025 at 09:43:07
> **Generated by:** Requirements Gathering Agent - PDF Converter

## Test Cases

**Generated by adpa-enterprise-framework-automation v3.2.0**
**Category:** quality-assurance
**Generated:** 2025-07-08T01:39:13.754Z
**Description:** Detailed test cases and test scenarios

---

## Test Cases

**Project:** === 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 Version:** 1.0
**Date:** 08/07/2025
**Status:** Draft

## 1. Test Case Overview

### 1.1 Purpose
This document contains comprehensive test cases for the === PROJECT RE
# 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 |

| Feature | Priority | Status |
|---|---|---|
| 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 |

| Feature | Priority | Status |
|---|---|---|
| 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]

 project. These test cases are designed to validate all functional and

### 1.2 Scope
The test cases cover:
- Functional testing of all user stories and requirements
- User interface and user experience validation
- Integration testing between system components
- Performance and security testing scenarios
- Error handling and edge case validation
```

```
### 1.3 Test Case Structure
Each test case includes:
- **Test Case ID:** Unique identifier for traceability
- **Test Case Title:** Descriptive name of the test scenario
- **Objective:** Purpose and goal of the test
- **Preconditions:** Setup requirements before test execution
- **Test Steps:** Detailed step-by-step actions
- **Expected Results:** Expected system behavior
- **Test Data:** Required input data and values
- **Priority:** Critical, High, Medium, or Low
- **Requirements Traceability:** Link to specific requirements

### 1.4 Test Case Categories
- **Functional Test Cases:** Core business functionality
- **UI/UX Test Cases:** User interface and experience
- **Integration Test Cases:** Component and system integration
- **Performance Test Cases:** Load, stress, and performance validation
- **Security Test Cases:** Authentication, authorization, and data pro
- **Negative Test Cases:** Error handling and edge cases

## 2. Functional Test Cases

### TC_FUNC_001: User Login - Valid Credentials
- **Objective:** Verify successful user login with valid credentials
- **Priority:** Critical
- **Preconditions:** User account exists in the system
- **Test Steps:**
  1. Navigate to login page
  2. Enter valid username in the username field
  3. Enter valid password in the password field
  4. Click the "Login" button
- **Expected Results:**
  - User is successfully authenticated
  - User is redirected to the dashboard
  - Welcome message is displayed
- **Test Data:**
  - Username: testuser@example.com
  - Password: ValidPass123!
- **Requirements Traceability:** REQ_AUTH_001

### TC_FUNC_002: User Login - Invalid Credentials
- **Objective:** Verify appropriate error handling for invalid login c
```

```
- **Priority:** High
- **Preconditions:** Login page is accessible
- **Test Steps:**
  1. Navigate to login page
  2. Enter invalid username in the username field
  3. Enter invalid password in the password field
  4. Click the "Login" button
- **Expected Results:**
  - Authentication fails
  - Error message "Invalid username or password" is displayed
  - User remains on login page
- **Test Data:**
  - Username: invalid@example.com
  - Password: WrongPassword
- **Requirements Traceability:** REQ_AUTH_002

### TC_FUNC_003: User Registration - Valid Data
- **Objective:** Verify successful user registration with valid inform
- **Priority:** Critical
- **Preconditions:** Registration page is accessible
- **Test Steps:**
  1. Navigate to registration page
  2. Enter valid first name
  3. Enter valid last name
  4. Enter valid email address
  5. Enter valid password
  6. Confirm password
  7. Accept terms and conditions
  8. Click "Register" button
- **Expected Results:**
  - User account is created successfully
  - Confirmation message is displayed
  - Verification email is sent
- **Test Data:**
  - First Name: John
  - Last Name: Doe
  - Email: johndoe@example.com
  - Password: SecurePass123!
- **Requirements Traceability:** REQ_REG_001

### TC_FUNC_004: Data Entry - Form Validation
- **Objective:** Verify form validation for required fields
- **Priority:** High
```

```
- **Preconditions:** User is logged in and form is accessible
- **Test Steps:**
  1. Navigate to data entry form
  2. Leave required fields empty
  3. Enter data in optional fields
  4. Click "Save" button
- **Expected Results:**
  - Validation errors are displayed for required fields
  - Form is not submitted
  - Error messages are clear and helpful
- **Test Data:**
  - Required fields: Name, Email, Phone
  - Optional fields: Address, Notes
- **Requirements Traceability:** REQ_VAL_001

### TC_FUNC_005: Data Search and Filter
- **Objective:** Verify search and filter functionality
- **Priority:** Medium
- **Preconditions:** User is logged in, test data exists in system
- **Test Steps:**
  1. Navigate to search page
  2. Enter search criteria in search field
  3. Apply additional filters if available
  4. Click "Search" button
- **Expected Results:**
  - Search results are displayed correctly
  - Results match the search criteria
  - Pagination works correctly if applicable
- **Test Data:**
  - Search term: "test data"
  - Filters: Date range, Category
- **Requirements Traceability:** REQ_SEARCH_001

## 3. User Interface Test Cases

### TC_UI_001: Navigation Menu Functionality
- **Objective:** Verify all navigation menu items work correctly
- **Priority:** High
- **Preconditions:** User is logged in
- **Test Steps:**
  1. Verify main navigation menu is visible
  2. Click each menu item
  3. Verify correct page loads for each menu item
```

```
  4. Check submenu functionality if applicable
- **Expected Results:**
  - All menu items are clickable
  - Correct pages load for each menu item
  - Navigation is consistent across pages
- **Test Data:** N/A
- **Requirements Traceability:** REQ_NAV_001

### TC_UI_002: Responsive Design Validation
- **Objective:** Verify application works correctly on different scree
- **Priority:** Medium
- **Preconditions:** Application is accessible
- **Test Steps:**
  1. Open application in desktop browser
  2. Resize browser window to tablet size
  3. Resize browser window to mobile size
  4. Test key functionality at each size
- **Expected Results:**
  - Layout adjusts correctly for each screen size
  - All functionality remains accessible
  - Text remains readable at all sizes
- **Test Data:**
  - Desktop: 1920x1080
  - Tablet: 768x1024
  - Mobile: 375x667
- **Requirements Traceability:** REQ_RESP_001

### TC_UI_003: Form Input Validation Display
- **Objective:** Verify form validation messages are displayed correct
- **Priority:** High
- **Preconditions:** Form with validation is accessible
- **Test Steps:**
  1. Navigate to form with validation
  2. Enter invalid data in various fields
  3. Attempt to submit form
  4. Verify validation messages
- **Expected Results:**
  - Validation messages appear for invalid fields
  - Messages are clear and specific
  - Visual indicators highlight invalid fields
- **Test Data:**
  - Invalid email: "notanemail"
  - Invalid phone: "123"
```

```
  - Invalid date: "32/13/2023"
- **Requirements Traceability:** REQ_VAL_002

## 4. Integration Test Cases

### TC_INT_001: Database Operations
- **Objective:** Verify CRUD operations work correctly with database
- **Priority:** Critical
- **Preconditions:** Database is accessible, test data available
- **Test Steps:**
  1. Create new record through application
  2. Read/retrieve the created record
  3. Update the record with new information
  4. Delete the record
  5. Verify record no longer exists
- **Expected Results:**
  - All CRUD operations complete successfully
  - Data integrity is maintained
  - No orphaned records remain
- **Test Data:**
  - Test record with all required fields
- **Requirements Traceability:** REQ_DB_001

### TC_INT_002: API Integration Testing
- **Objective:** Verify integration with external APIs
- **Priority:** High
- **Preconditions:** External API is available, authentication configu
- **Test Steps:**
  1. Make API call from application
  2. Verify request is properly formatted
  3. Verify response is received
  4. Verify response data is processed correctly
- **Expected Results:**
  - API calls are successful
  - Data is exchanged correctly
  - Error handling works for API failures
- **Test Data:**
  - Valid API endpoints and test data
- **Requirements Traceability:** REQ_API_001

### TC_INT_003: Third-Party Service Integration
- **Objective:** Verify integration with third-party services
- **Priority:** Medium
```

```
- **Preconditions:** Third-party service is configured and accessible
- **Test Steps:**
  1. Trigger integration with third-party service
  2. Verify data is sent correctly
  3. Verify response is received and processed
  4. Check error handling for service unavailability
- **Expected Results:**
  - Integration works as expected
  - Data exchange is accurate
  - Graceful degradation when service unavailable
- **Test Data:**
  - Service configuration and test payloads
- **Requirements Traceability:** REQ_INT_001

## 5. Performance Test Cases

### TC_PERF_001: Page Load Time
- **Objective:** Verify page load times meet performance requirements
- **Priority:** High
- **Preconditions:** Application is deployed in test environment
- **Test Steps:**
  1. Clear browser cache
  2. Navigate to application pages
  3. Measure page load times
  4. Record performance metrics
- **Expected Results:**
  - Pages load within 2 seconds
  - Performance is consistent across browsers
  - No performance degradation over time
- **Test Data:**
  - Various page types and sizes
- **Requirements Traceability:** REQ_PERF_001

### TC_PERF_002: Concurrent User Load
- **Objective:** Verify system handles concurrent users appropriately
- **Priority:** High
- **Preconditions:** Load testing tools configured
- **Test Steps:**
  1. Configure load test for 100 concurrent users
  2. Execute typical user workflows
  3. Monitor system performance
  4. Gradually increase load to 500 users
- **Expected Results:**
```

```
  - System maintains performance under load
  - Response times remain within limits
  - No system crashes or errors
- **Test Data:**
  - User scenarios and test accounts
- **Requirements Traceability:** REQ_PERF_002

### TC_PERF_003: Database Query Performance
- **Objective:** Verify database queries perform within acceptable lim
- **Priority:** Medium
- **Preconditions:** Database with representative data volume
- **Test Steps:**
  1. Execute complex database queries
  2. Measure query execution times
  3. Test with various data volumes
  4. Monitor database resource usage
- **Expected Results:**
  - Queries complete within 1 second
  - Performance scales appropriately with data volume
  - Database resources are used efficiently
- **Test Data:**
  - Large datasets for testing
- **Requirements Traceability:** REQ_PERF_003

## 6. Security Test Cases

### TC_SEC_001: Authentication Security
- **Objective:** Verify authentication mechanisms are secure
- **Priority:** Critical
- **Preconditions:** Security testing tools available
- **Test Steps:**
  1. Test password complexity requirements
  2. Verify account lockout after failed attempts
  3. Test session timeout functionality
  4. Verify secure password storage
- **Expected Results:**
  - Strong password policies are enforced
  - Accounts lock after 5 failed attempts
  - Sessions timeout after inactivity
  - Passwords are properly hashed
- **Test Data:**
  - Various password combinations
- **Requirements Traceability:** REQ_SEC_001
```

```
### TC_SEC_002: Authorization Testing
- **Objective:** Verify user authorization and access controls
- **Priority:** Critical
- **Preconditions:** Multiple user roles configured
- **Test Steps:**
  1. Login with different user roles
  2. Attempt to access restricted resources
  3. Verify role-based permissions
  4. Test privilege escalation attempts
- **Expected Results:**
  - Users can only access authorized resources
  - Role-based permissions work correctly
  - Unauthorized access attempts are blocked
- **Test Data:**
  - User accounts with different roles
- **Requirements Traceability:** REQ_SEC_002

### TC_SEC_003: Data Protection
- **Objective:** Verify sensitive data is properly protected
- **Priority:** High
- **Preconditions:** Application handles sensitive data
- **Test Steps:**
  1. Submit sensitive data through application
  2. Verify data encryption in transit
  3. Verify data encryption at rest
  4. Test data masking in logs and displays
- **Expected Results:**
  - Data is encrypted during transmission
  - Sensitive data is encrypted in database
  - Data is masked in logs and error messages
- **Test Data:**
  - Sensitive test data (PII, financial)
- **Requirements Traceability:** REQ_SEC_003

## 7. Negative Test Cases

### TC_NEG_001: Invalid Input Handling
- **Objective:** Verify system handles invalid input appropriately
- **Priority:** High
- **Preconditions:** Forms and input fields are accessible
- **Test Steps:**
  1. Enter invalid data in input fields
```

```
  2. Submit forms with invalid data
  3. Test boundary conditions
  4. Verify error handling
- **Expected Results:**
  - Invalid input is rejected
  - Appropriate error messages are displayed
  - System remains stable
- **Test Data:**
  - SQL injection strings, XSS payloads, invalid formats
- **Requirements Traceability:** REQ_ERR_001

### TC_NEG_002: Error Recovery Testing
- **Objective:** Verify system recovers gracefully from errors
- **Priority:** Medium
- **Preconditions:** System components can be disrupted
- **Test Steps:**
  1. Simulate network interruption
  2. Simulate database connection loss
  3. Simulate service unavailability
  4. Verify system recovery
- **Expected Results:**
  - System detects errors appropriately
  - Error messages are user-friendly
  - System recovers when conditions improve
- **Test Data:**
  - Various error scenarios
- **Requirements Traceability:** REQ_ERR_002

### TC_NEG_003: Resource Exhaustion
- **Objective:** Verify system behavior under resource constraints
- **Priority:** Low
- **Preconditions:** Ability to limit system resources
- **Test Steps:**
  1. Limit available memory
  2. Limit disk space
  3. Limit network bandwidth
  4. Monitor system behavior
- **Expected Results:**
  - System degrades gracefully
  - Appropriate warnings are generated
  - Critical functions remain available
- **Test Data:**
  - Resource limitation scenarios
```

```
- **Requirements Traceability:** REQ_ERR_003

## 8. Test Execution Guidelines

### 8.1 Pre-Execution Setup
- Verify test environment is properly configured
- Ensure test data is available and current
- Confirm all required tools and access are available
- Review test case updates and dependencies

### 8.2 Execution Process
- Execute test cases in the specified order
- Document actual results for each test step
- Capture screenshots for UI-related tests
- Log any deviations from expected results

### 8.3 Result Documentation
- Mark test cases as Pass, Fail, or Blocked
- Provide detailed comments for failed tests
- Attach supporting evidence (screenshots, logs)
- Update test case status in test management tool

### 8.4 Defect Reporting
- Create defect reports for failed test cases
- Include detailed reproduction steps
- Attach supporting evidence and logs
- Assign appropriate severity and priority

---

**Document Control:**
- **Author:** Test Analyst Team
- **Reviewers:** Senior Test Analyst, Business Analyst, Development Le
- **Approval:** Test Manager
- **Next Review Date:** [Date + 2 weeks]
- **Distribution:** All testing team members, development team

**Revision History:**
| Version | Date | Author | Changes |
|---------|------|--------|---------|
| 1.0 | 08/07/2025 | Test Analyst | Initial test cases document |

**Test Case Statistics:**
```

```
- Total Test Cases: 18
- Critical Priority: 6
- High Priority: 8
- Medium Priority: 3
- Low Priority: 1
```

---