ADPA Implementation Plan

> **Source File:** generated-documents\ADPA-Implementation-Plan.md
> **Generated:** 16/07/2025 at 13:55:45
> **Generated by:** Requirements Gathering Agent - PDF Converter

ADPA Implementation Plan for Coding Agent

Overview

This implementation plan is based on a review of the generated documentation for the ADPA (Advanced Document Processing & Automation Framework) project. The plan is designed to guide a coding agent (or team) in delivering the core features, integrations, and technical requirements described in the project documentation.

---

1. Project Vision & Goals

- Deliver a unified, AI-powered platform for generating, managing, and integrating business-critical documentation.
- Ensure compliance with BABOK v3, PMBOK 7th Edition, and DMBOK 2.0 standards.
- Provide robust integrations with Confluence, SharePoint, Adobe Document Services, and VCS tools.
- Support multiple interfaces: CLI, REST API, and web admin portal.

---

2. Core Feature Implementation Steps

2.1 Standards-Compliant Document Generation

- [ ] Implement document templates for all required standards (BABOK, PMBOK, DMBOK).
- [ ] Build/extend the document generation engine to support CLI, API, and web triggers.
- [ ] Ensure output formats include Markdown, PDF, and JSON.
- [ ] Integrate acceptance criteria and user story validation.

2.2 Multi-Provider AI Orchestration

- [ ] Integrate OpenAI, Google AI, GitHub Copilot, and Ollama as AI providers.
- [ ] Implement provider failover and health checks.
- [ ] Support context-aware prompt engineering and smart template selection.

2.3 Enterprise Integrations

- [ ] Implement Confluence integration (OAuth2, page publishing, metadata sync).
- [ ] Implement SharePoint integration (OAuth2, document library sync, metadata).
- [ ] Integrate Adobe Document Services for PDF generation and manipulation.
- [ ] Provide VCS integration for document versioning and audit trails.

2.4 Security & Compliance

- [ ] Enforce RBAC, OAuth2, SAML, and API key authentication.
- [ ] Implement audit logging and compliance reporting.
- [ ] Ensure all generated documents are standards-compliant and traceable.

2.5 API & Web Portal

- [ ] Build REST API endpoints for all document and integration workflows.
- [ ] Develop the Next.js/React admin portal for project and document management.
- [ ] Implement user management, project dashboards, and workflow triggers.

---

3. Technical Architecture & Stack

- Node.js (≥18.x), TypeScript (≥5.7), Express.js (API), Next.js/React (admin portal)
- Tailwind CSS, Axios, Docker, Kubernetes (for scalable deployment)
- Multi-provider AI SDKs, OAuth2 libraries, OpenAPI 3.0 for API docs

---

4. Implementation Phases

Phase 1: Foundation

- [ ] Set up monorepo structure and CI/CD pipeline
- [ ] Implement core document generation engine and basic templates
- [ ] Integrate at least one AI provider (OpenAI or Google AI)
- [ ] Build initial CLI and REST API endpoints

Phase 2: Integrations & Security

- [ ] Add Confluence, SharePoint, and Adobe integrations
- [ ] Implement authentication, RBAC, and compliance features
- [ ] Expand template library and standards coverage

Phase 3: Advanced Features & Scaling

- [ ] Add provider failover, advanced context management, and smart template selection
- [ ] Build admin web portal and dashboards
- [ ] Optimize for scalability, monitoring, and enterprise deployment

---

5. Acceptance Criteria

- All features must pass user story acceptance criteria as defined in `user-stories.md`.
- All integrations must be tested in enterprise-like environments.
- Documentation, security, and compliance must be validated by stakeholders.

---

6. Next Steps for Coding Agent

- [ ] Review and prioritize user stories and requirements.
- [ ] Set up the development environment as per the technical stack.
- [ ] Begin implementation with Phase 1 tasks, tracking progress in project management tools.
- [ ] Regularly sync with stakeholders for feedback and validation.

This plan should be updated as features are delivered and requirements evolve. For detailed requirements, see the generated documentation in each subdirectory.

---

7. Ongoing Best Practices & Recommendations

- **Iterative Delivery:** Break down large features into small, testable increments. Use feature branches and pull requests for all changes.
- **Documentation:** Update technical and user documentation as features are delivered. Maintain clear API docs and integration

guides.

- **Testing:** Ensure all new code is covered by unit, integration, and end-to-end tests. Use CI/CD to enforce test coverage and code quality.
- **Security:** Regularly review authentication, authorization, and data handling. Rotate secrets and audit dependencies for vulnerabilities.
- **Compliance:** Validate all outputs and integrations against BABOK, PMBOK, and DMBOK standards. Document compliance checks and traceability.
- **Stakeholder Feedback:** Schedule regular demos and feedback sessions with stakeholders. Adjust priorities based on feedback and evolving requirements.
- **Continuous Improvement:** Review and refine processes, tools, and architecture. Encourage team learning and adoption of new best practices.
