Database Schema

> **Source File:** generated-documents\technical-design\database-schema.md
> **Generated:** 30/07/2025 at 07:01:58
> **Generated by:** Requirements Gathering Agent - PDF Converter

DatabaseSchema

Generated by adpa-enterprise-framework-automation v3.2.0
Category: technical-design
Generated: 2025-07-14T21:01:38.763Z
Description:

Database Schema Design Document

Project: ADPA - Advanced Document Processing & Automation Framework
Version: 3.2.0
Prepared by: [Your Name / Team]
Date: [Current Date]

---

1. Database Overview

ADPA is an enterprise automation framework for AI-powered document generation, project management, and business analysis, supporting standards such as BABOK v3, PMBOK 7, and DMBOK 2.0. The platform is multi-tenant, API-first, supports multiple AI provider integrations, and features workflow automation, document management, and robust compliance/security controls.

The database must support:

- Multi-user, multi-team collaboration
- Standards-compliant document templates and generated documents
- Integration metadata (Confluence, SharePoint, Adobe, etc.)

- Robust access control and auditability
- Workflow automation and job tracking
- Extensibility for new standards and enterprise integrations

Recommended RDBMS: PostgreSQL (for its scalability, JSONB support, and enterprise features).

---

2. Entity Relationships

High-level ERD Placeholder:

```
[User] --< [TeamMember] >-- [Team] --< [Project] --< [Document] >-- [DocumentVersion]
                                   |               |
                              [ProjectRole]      [DocumentJob]
                                   |
                            [Role] (with permissions)
[User] --< [AuthProvider]
[Document] >-- [Template]
[Template] --< [Framework]
[Document] >-- [IntegrationMetadata]
[AuditLog] (references most entities)
```

- **User**: Can belong to multiple teams, own projects, and initiate jobs.
- **Team**: Aggregates users for collaboration and project grouping.
- **Project**: Contains documents, templates, and workflow jobs.
- **Document**: Versioned, can be exported via integrations.
- **Template**: Linked to standards/frameworks (BABOK, PMBOK, etc.).
- **Role/Permission**: Fine-grained RBAC for enterprise security.
- **AuditLog**: Tracks all critical actions for compliance.

---

3. Table Definitions

3.1 Users and Teams

users

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Unique user identifier

Column	Type	Constraints	Description
email	VARCHAR(255)	NOT NULL, UNIQUE, INDEX	User email
password_hash	VARCHAR(255)	NOT NULL	Bcrypt hash of password
name	VARCHAR(100)		Full name
is_active	BOOLEAN	DEFAULT TRUE	User status
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp
updated_at	TIMESTAMP		Last update timestamp

teams

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Unique team identifier
name	VARCHAR(100)	NOT NULL, UNIQUE	Team name
description	TEXT		Team details
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp
updated_at	TIMESTAMP		Update timestamp

team_members

Column	Type	Constraints	Description
team_id	UUID	PK, FK → teams(id), NOT NULL	Team reference

Column	Type	Constraints	Description
user_id	UUID	PK, FK → users(id), NOT NULL	User reference
role_id	UUID	FK → roles(id), NOT NULL	Membership role
joined_at	TIMESTAMP	DEFAULT NOW()	When joined
permissions	JSONB		Custom permissions

roles

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Unique role ID
name	VARCHAR(50)	NOT NULL, UNIQUE	Role name (admin, analyst, etc.)
description	TEXT		Role description
permissions	JSONB	NOT NULL	Permission matrix

3.2 Projects and Collaboration

projects

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Unique project ID
team_id	UUID	FK → teams(id), NOT NULL	Owning team
name	VARCHAR(150)	NOT NULL	Project name
description	TEXT		Project description

Column	Type	Constraints	Description
status	VARCHAR(30)	DEFAULT 'active'	Project status
created_by	UUID	FK → users(id), NOT NULL	Creator
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp
updated_at	TIMESTAMP		Last update

project_roles (optional, for per-project overrides)

project_id	UUID	FK → projects(id), PK
user_id	UUID	FK → users(id), PK
role_id	UUID	FK → roles(id)
assigned_at	TIMESTAMP	

3.3 Document Templates, Documents, and Versions

templates

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Template ID
name	VARCHAR(150)	NOT NULL	Template name
description	TEXT		Template description
category	VARCHAR(50)	NOT NULL, INDEX	E.g., 'babok', 'pmbok'
framework_id	UUID	FK → frameworks(id)	Linked standard/framework
tags	VARCHAR[]		Searchable tags
template_data	JSONB	NOT NULL	Template content/structure

Column	Type	Constraints	Description
is_active	BOOLEAN	DEFAULT TRUE	Active status
created_by	UUID	FK → users(id)	Creator
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp

frameworks

Column	Type	Constraints	Description
id	UUID (PK)		Framework ID
name	VARCHAR(75)	NOT NULL, UNIQUE	'BABOK v3', 'PMBOK 7', etc.
description	TEXT		
version	VARCHAR(20)		
is_active	BOOLEAN	DEFAULT TRUE	

documents

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Document ID
project_id	UUID	FK → projects(id), NOT NULL	Associated project
template_id	UUID	FK → templates(id)	Source template
title	VARCHAR(150)	NOT NULL	Document title
status	VARCHAR(30)	DEFAULT 'draft'	Document status

Column	Type	Constraints	Description
format	VARCHAR(20)	NOT NULL	markdown/pdf/docx/json
created_by	UUID	FK → users(id), NOT NULL	Creator
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp
updated_at	TIMESTAMP		Last update
current_version	INT	NOT NULL, DEFAULT 1	Latest version number

document_versions

Column	Type	Constraints	Description
id	UUID (PK)	PRIMARY KEY	Version ID
document_id	UUID	FK → documents(id), NOT NULL	Parent document
version	INT	NOT NULL	Version number
content	JSONB	NOT NULL	Document content
created_by	UUID	FK → users(id)	Author
created_at	TIMESTAMP	DEFAULT NOW()	Creation timestamp
change_note	TEXT		Version change summary

3.4 Document Generation Jobs (Workflow)

document_jobs

Column	Type	Constraints	Description
id	UUID (PK)	NOT NULL, PRIMARY KEY	Job ID
document_id	UUID	FK → documents(id), NOT NULL	Target document
status	VARCHAR(30)	NOT NULL	queued/running/completed/failed
started_at	TIMESTAMP		Start time
completed_at	TIMESTAMP		End time
initiated_by	UUID	FK → users(id)	Who started job
ai_provider	VARCHAR(50)		Used AI provider
logs	TEXT		Execution logs
result_url	VARCHAR(255)		Output file location

3.5 Integrations and Metadata

integration_metadata

Column	Type	Constraints	Description
id	UUID (PK)		Row ID
document_id	UUID	FK → documents(id), NOT NULL	Related document
integration	VARCHAR(30)	NOT NULL	'confluence', 'sharepoint', etc.
external_id	VARCHAR(150)		External system reference

Column	Type	Constraints	Description
metadata	JSONB		Integration-specific data
published_at	TIMESTAMP		Publish time

auth_providers (for SSO/OAuth2, etc.)

Column	Type	Constraints	Description
id	UUID (PK)		Row ID
user_id	UUID	FK → users(id)	User reference
provider	VARCHAR(30)	NOT NULL	'google', 'github', 'azuread', etc.
provider_id	VARCHAR(150)	NOT NULL	ID at provider
access_token	TEXT		Encrypted access token
refresh_token	TEXT		
expires_at	TIMESTAMP		
created_at	TIMESTAMP	DEFAULT NOW()	

3.6 Audit Logging

audit_logs

Column	Type	Constraints	Description
id	UUID (PK)		Row ID
user_id	UUID	FK → users(id)	Acting user
action	VARCHAR(100)	NOT NULL	E.g. 'document.create'

Column	Type	Constraints	Description
entity_type	VARCHAR(30)	NOT NULL	E.g. 'document', 'team', etc.
entity_id	UUID		Target entity
details	JSONB		Detailed context
created_at	TIMESTAMP	DEFAULT NOW()	

---

4. Data Types

- **UUID**: For all primary/foreign keys; ensures uniqueness (Postgres: `uuid` type).
- **VARCHAR(N)**: For short text fields (name, status, provider, etc.).
- **TEXT**: For long text (descriptions, logs).
- **JSONB**: For flexible schema data (template data, metadata, permissions, logs).
- **TIMESTAMP**: For time fields (always with time zone).
- **BOOLEAN**: For status/flags.
- **INT**: For versioning and counters.
- **ARRAY**: For tags (Postgres: `varchar[]`).
- **ENUM**: (optional, for fields like status—can be implemented as VARCHAR with check constraints, or native enums).

---

5. Indexes and Keys

- **Primary Key**: All tables use UUID PKs.
- **Foreign Keys**: Enforce referential integrity on all relationships.
- **Unique Index**: On `users.email`, `teams.name`, `roles.name`, `frameworks.name`.
- **Composite PK**: `team_members (team_id, user_id)`, `project_roles (project_id, user_id)`.
- **Indexes**:
  - `documents(project_id)`
  - `templates(category)`
  - `document_jobs(document_id)`
  - `integration_metadata(document_id, integration)`
  - `audit_logs(user_id, entity_type, entity_id)`
  - GIN index on `tags`, `permissions`, and JSONB columns for search.

---

6. Constraints

- **NOT NULL**: For all required fields (PKs, FKs, essential metadata).
- **UNIQUE**: For emails, names where appropriate.
- **FOREIGN KEY**: All relations; `ON DELETE CASCADE` or `SET NULL` as needed (e.g., cascade delete documents when project is deleted).
- **CHECK Constraints**: For status fields (e.g., status in documents, projects).
- **Default Values**: For flags (e.g., `is_active`), timestamps.

---

7. Normalization Strategy

- **3rd Normal Form (3NF)**: All tables are normalized; no repeating groups, all non-key attributes fully dependent on the key.
- **Denormalization**: Considered for:
  - Frequently accessed search fields (e.g., aggregate document metadata)
  - Caching computed values in reporting/analytics tables (if needed for performance)
  - JSONB columns for extensibility without schema change (e.g., custom template fields, integration metadata).

---

8. Performance Considerations

- **Indexing**: All FKs, search fields, and JSONB/array columns indexed.
- **Partitioning**: For large tables (documents, audit_logs), partition by time or project.
- **Caching**: For static or rarely changing reference data (frameworks, roles).
- **Bulk Operations**: Batched inserts/updates for document/job processing.
- **Connection Pooling**: For high concurrency in API workloads.
- **Scalability**: Multi-tenant support via team/project scoping, sharding feasible at project/team level if required.
- **Read Replicas**: For analytics/reporting workloads.

---

9. Data Migration Strategy

- **Initial Migration**: Use migration tools (e.g., Flyway, Liquibase, TypeORM, Prisma) to create schema.
- **Versioned Migrations**: All schema changes tracked and applied incrementally.
- **Legacy Data Import**: Scripts to convert old JSON configuration to normalized tables; validate referential integrity.
- **Data Validation**: Use tests/ETL pipelines to ensure data quality.

- **Rollback**: Down-migrations available for all schema changes.

---

10. Backup and Recovery

- **Automated Nightly Backups**: Full and incremental; stored in secure, offsite location.
- **Point-in-Time Recovery**: WAL archiving enabled (for PostgreSQL).
- **Backup Verification**: Regular restore drills and checksum validation.
- **Disaster Recovery Plan**: Documented, tested, with RTO/RPO targets.
- **Sensitive Data Encryption**: At rest and in transit; secrets and tokens stored encrypted in DB.

APPENDIX: ERD Diagram Placeholder

> *(Insert ERD image here—visualize Users, Teams, Projects, Documents, Templates, Integrations, Roles/Permissions, and AuditLogs with appropriate relationships and cardinalities.)*

Notes and Justification

- **Security**: RBAC, encrypted credentials, and audit logging are first-class.
- **Compliance**: Audit trails and data lineage for regulatory standards (GDPR, SOX, HIPAA, etc.).
- **Extensibility**: New frameworks, document types, integrations, and AI providers can be added without schema overhaul.
- **Multi-Tenancy**: Team/project scoping; all data is access-controlled.
- **Enterprise Integration**: Integration metadata supports external systems (Confluence, SharePoint, Adobe, etc.).
- **Collaboration**: Document versioning, roles per project, and team-based access.
- **Scalability**: Designed for large-scale enterprise deployments.

End of Document