AI Agent Documentation Storage Architecture

Overview

This document outlines the comprehensive storage and repository architecture for managing documentation, code, and data for 20+ AI agents in a production environment.

1. Primary Storage Locations

1.1 Git Repository (GitLab/GitHub Enterprise)

Purpose: Version control for code, configurations, and documentation **Location**: `https://git.company.com/ai-platform/agents`

```
ai-agents-platform/
├── documentation/
│   ├── architecture/
│   │   ├── system-design.md
│   │   ├── data-flow-diagrams/
│   │   └── api-specifications/
│   ├── development/
│   │   ├── getting-started.md
│   │   ├── best-practices.md
│   │   └── coding-standards.md
│   └── operations/
│       ├── deployment-guide.md
│       ├── monitoring-setup.md
│       └── incident-response.md
├── agents/
│   ├── customer-support-agent/
│   │   ├── src/
│   │   ├── prompts/
│   │   ├── tests/
│   │   ├── configs/
│   │   └── README.md
│   ├── data-analysis-agent/
│   ├── code-review-agent/
│   └── [17 more agents...]
├── shared-libraries/
├── evaluation-framework/
└── infrastructure-as-code/
```

1.2 Confluence/SharePoint

Purpose: Business documentation, runbooks, and team collaboration **Location**: `https://confluence.company.com/display/AIAGENTS`

- **Spaces Structure**:
  - AI Agents Overview
  - Agent Catalog
  - User Guides
  - Training Materials
  - Meeting Notes
  - Roadmaps
  - Post-mortems

1.3 AWS S3 / Azure Blob Storage

Purpose: Large datasets, model artifacts, logs, and backups **Buckets Structure**:

```
s3://ai-agents-prod/
├── models/
│   ├── customer-support-agent/
│   │   ├── v1.0.0/
│   │   ├── v1.1.0/
│   │   └── latest/
│   └── [other agents...]
├── datasets/
│   ├── training/
│   ├── evaluation/
│   └── feedback/
├── logs/
│   ├── performance/
│   ├── errors/
│   └── interactions/
└── backups/
    ├── daily/
    ├── weekly/
    └── monthly/
```

1.4 Vector Database (Pinecone/Weaviate)

Purpose: Agent knowledge bases and semantic search **Collections**:

- `agent-knowledge-base`
- `user-feedback-embeddings`
- `documentation-search`
- `conversation-history`

1.5 PostgreSQL/MongoDB

Purpose: Structured data, metadata, and configurations **Schemas**:

- `agents_metadata`
- `performance_metrics`
- `user_feedback`
- `deployment_history`
- `evaluation_results`

2. Documentation Types and Storage Mapping

Documentation Type	Primary Storage	Secondary Storage	Access Method
Source Code	Git Repository	S3 (backups)	Git CLI, IDE
API Documentation	Git + Swagger	Confluence	Web Portal
User Guides	Confluence	S3 (PDFs)	Web Browser
Training Data	S3	Git (samples)	SDK/API
Model Artifacts	S3	Model Registry	MLOps Platform
Logs & Metrics	CloudWatch/ELK	S3 (archives)	Monitoring Dashboard
Secrets & Configs	AWS Secrets Manager	HashiCorp Vault	API/CLI

3. Integration Architecture

3.1 CI/CD Pipeline Integration

```yaml
```

```yaml
# .gitlab-ci.yml example
stages:
  - validate
  - test
  - build
  - deploy
  - document

update-documentation:
  stage: document
  script:
    - python scripts/generate_docs.py
    - aws s3 sync ./docs s3://ai-agents-docs/
    - curl -X POST $CONFLUENCE_API/update
```

3.2 Documentation Generation Pipeline

1. **Code Changes** → Git commit triggers pipeline
2. **Auto-generate Docs** → Extract docstrings, generate API docs
3. **Sync to Storage** → Update S3, Confluence, and search index
4. **Notify Teams** → Slack/Teams notification of updates

3.3 Access Control Matrix

Role	Git	Confluence	S3	Database	Vector DB
Developer	Read/Write	Read/Write	Read	Read/Write	Read
Data Scientist	Read	Read/Write	Read/Write	Read	Read/Write
Operations	Read	Read	Read/Write	Read	Read
Business User	None	Read	None	None	None

4. Backup and Disaster Recovery

4.1 Backup Schedule

- **Git Repository**: Real-time replication to secondary region
- **Confluence**: Daily exports to S3
- **S3 Buckets**: Cross-region replication enabled
- **Databases**: Daily snapshots, point-in-time recovery enabled
- **Vector DB**: Weekly full backups, daily incremental

4.2 Recovery Time Objectives (RTO)

- Critical Documentation: < 1 hour
- Model Artifacts: < 2 hours
- Historical Data: < 24 hours

5. Search and Discovery

5.1 Unified Search Platform

- **Elasticsearch Index** aggregating:
  - Git repository content
  - Confluence pages
  - S3 object metadata
  - Database records
  - Vector similarity search

5.2 Documentation Portal

URL: https://ai-docs.company.com

- Single sign-on (SSO) enabled
- Role-based access control
- Full-text search across all sources
- Version history and change tracking

6. Monitoring and Compliance

6.1 Documentation Metrics

- **Coverage**: % of agents with complete documentation
- **Freshness**: Days since last update
- **Quality**: Automated quality checks
- **Usage**: Page views and search queries

6.2 Compliance Requirements

- **GDPR**: Personal data handling documentation
- **SOC2**: Security and access controls
- **ISO 27001**: Information security management

- **Industry-specific**: Healthcare (HIPAA), Finance (PCI-DSS)

7. Implementation Roadmap

Phase 1: Foundation (Months 1-2)

- Set up Git repository structure
- Configure S3 buckets and policies
- Establish Confluence spaces
- Implement basic CI/CD

Phase 2: Integration (Months 3-4)

- Deploy unified search
- Automate documentation generation
- Set up monitoring dashboards
- Implement backup procedures

Phase 3: Optimization (Months 5-6)

- Machine learning for documentation quality
- Advanced search with NLP
- Automated compliance checking
- Performance optimization

8. Best Practices

8.1 Documentation Standards

- Use Markdown for technical docs
- Follow semantic versioning
- Include examples and diagrams
- Maintain changelog for each agent

8.2 Version Control

- Feature branches for major changes
- Code reviews for documentation
- Automated testing of examples
- Tag releases appropriately

8.3 Access Management

- Principle of least privilege
- Regular access reviews
- Audit logging enabled
- Secret rotation policies

Conclusion

This storage architecture provides a robust, scalable foundation for managing documentation and artifacts for 20+ AI agents. The distributed approach ensures high availability, efficient collaboration, and comprehensive tracking of all agent-related information throughout their lifecycle.