Cloud Architecture & Deployment Plan

Components	Azure Service
Prototype	Hosted on AZ vm
Authenticated user login	Az appservice (Azure identity provider)
Frontend (Web APP)	Az app service
Backend API	Az Function
Document Storage	Az Blob Storage
Metadata / logs	Az cosmos DB
Vector Embeddings	Az AI Search
LLMs (Agents)	Az OpenAI Service
Monitoring and logs	Az Monitor + Az Application Insights

Phase 1: Prototype Deployment

Initially, the entire application will be hosted on a **single Azure Virtual Machine (VM)**. This allows rapid iteration and validation before modularizing services into dedicated Azure resources.

Component	Deployment (Prototype Phase)
Full Stack Prototype	Azure Virtual Machine (Linux)
All services	Containerized/locally hosted on VM

Phase 2: Modular Cloud Services Deployment (Post-Prototype)

Once validated, each component will be migrated to its respective managed Azure service for scalability, security, and cost efficiency.

Frontend (Document Upload + Chat Interface)

- **Service**: Azure App Service (Web App)
- **Features**:
  - Scalable web hosting for UI and APIs
  - Use Standard or Basic Tier for cost savings
- **Cost Optimization**:

- Auto-scale during business hours only
- Use CDN (Azure Front Door or Azure CDN) for static assets

API Layer (Document Processing & Orchestration)

- **Service**: Azure Functions (Serverless)
- **Use for**:
  - Upload handling
  - API endpoints
  - Triggering workflows
- **Cost Optimization**:
  - Pay-per-execution
  - Best for variable workloads (spiky traffic)

NLP Processing Engine

- **Service**: Azure Machine Learning or Azure Container Instances
- **Model Hosting Options**:
  - ~~Use **Azure Kubernetes Service (AKS)** with spot instances for LayoutLM~~
  - Use **Azure OpenAI Service** for GPT-based processing if available

---

Vector Search (RAG System)

- **Service**: Azure Cognitive Search + Azure Cosmos DB
- **Details**:
  - Use Cosmos DB for metadata + hierarchical JSON storage
  - Use Azure Search with vector index for semantic search

Storage Systems

- **Service**: Azure Blob Storage
- **Usage**:
  - Raw documents
  - Processed JSON
  - Generated videos, audio, and images
- **Cost Optimization**:

- Use **Hot tier** for recent uploads, **Cool/Archive** for older assets
- Use lifecycle rules to auto-move blobs to cheaper tiers

Post 4 months

phase 3 : Full Production on AKS (Kubernetes)

(only if needed) we will use microservices

With the help of AKS for high scalability!

Component	Description
Ingress + DNS	Azure-managed ingress routes external traffic to frontend pods
Namespaces	Logical isolation: `frontend`, `backend`, `agents`, etc.
Frontend Pod	UI interface, possibly React/Next.js app
API Gateway Pod	Handles routing and API exposure internally
Agent Pods	Micro-agents for orchestrating tasks (Manager, Generator, Narrator, etc.)
Image Prompt Agents	Interact with DALL·E or Stability API for image generation
Storage	Azure Blob for storing media & JSON; secured via Azure Key Vault
Monitoring	Separate node for observability: Grafana + Prometheus or SigNoz
Communication	Notifications via Discord or Slack

Components Breakdown

Agent System (Namespace: `agents`)

- **Manager Agent**: Central router, delegates tasks
- **Generator Agent**: Scene planning for visuals or narratives
- **Narrator Agent**: Calls LLM APIs (e.g., GPT) for script/text
- **Editor Agent**: Edits generated content (text/image) as per user or generator logic
- **Image Prompt Agent**: Connects to AI image models like DALL·E / Stability

Storage

- Media and JSON content goes to **Azure Blob Storage**

- Secure secrets/keys via **Azure Key Vault**

Monitoring (Node 2)

- Prometheus for metrics collection
- Grafana for dashboarding
- Optional: Use **SigNoz** as an all-in-one observability tool
- Alerts to Discord/Slack or email

Benefits of AKS-Based Phase 3

Feature	Description
High Scalability	Auto-scale pods based on workload (HPA + cluster autoscaler)
Better Isolation	Use namespaces to separate environments or services
Custom Agents	Deploy micro-agents independently and scale as needed
Secure Secrets	Use Azure Key Vault and Kubernetes secrets
CI/CD Ready	Integrate GitHub Actions or Azure DevOps pipelines
Unified Monitoring	Centralized logging/metrics with Grafana/Prometheus or SigNoz

With the help of aks for high scalability !!!!