

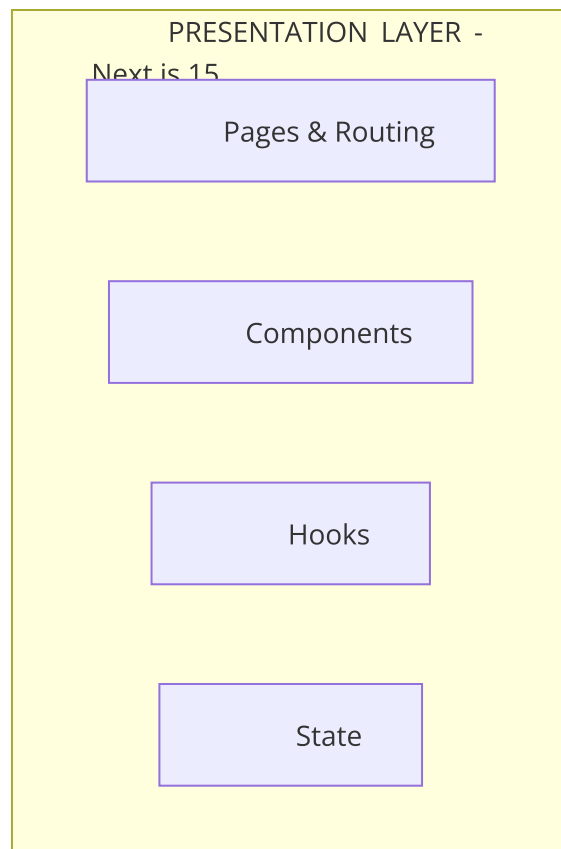
## 6. DESIGN

---

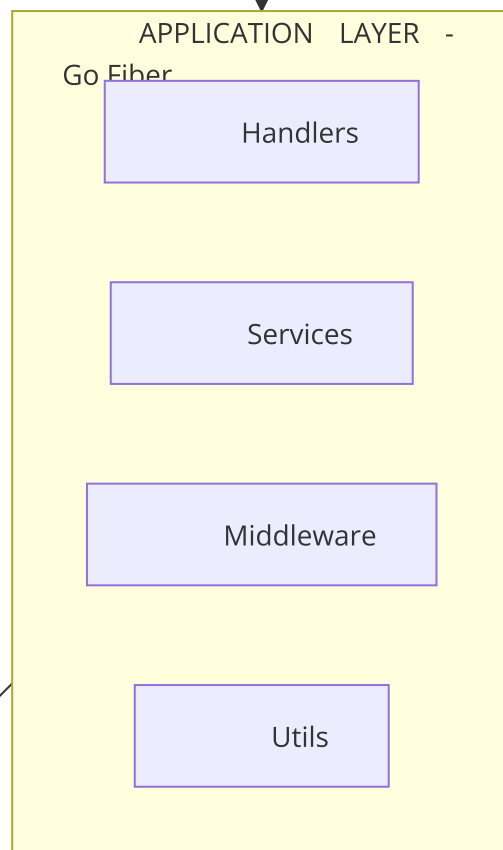
### 6.1 System Architecture

---

Three-tier architecture: Presentation (Next.js), Application (Go Fiber API), Data (PostgreSQL, Redis, DigitalOcean Spaces). Communication via RESTful APIs, SSE for streaming, S3 for storage.



HTTPS/REST + SSE



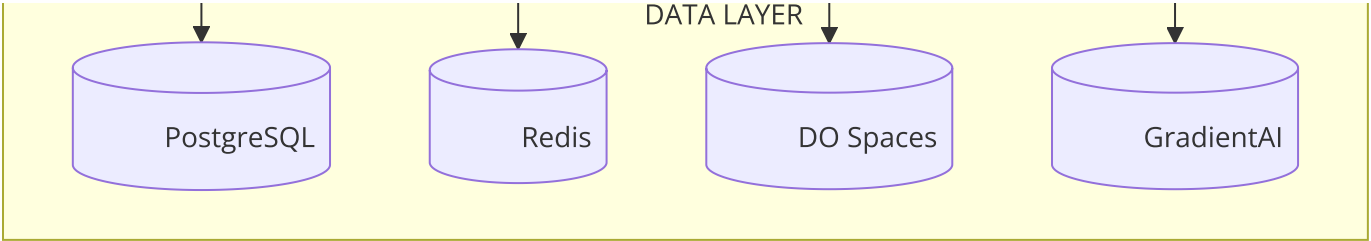


Figure 6.1: Three-tier System Architecture

## 6.2 Design Patterns Used

Pattern	Description
Repository Pattern	GORM-based data access layer abstracts database operations
Service Layer	Business logic separated from handlers and data access
Middleware Chain	JWT auth, CORS, rate limiting, logging as composable middleware
Provider Pattern	React context providers for auth, theme, and query state
Observer Pattern	SSE for real-time streaming of AI responses and job status
Factory Pattern	Service initialization with dependency injection

## 6.3 Data Flow Summary

### 6.3.1 Context Diagram (Level 0)

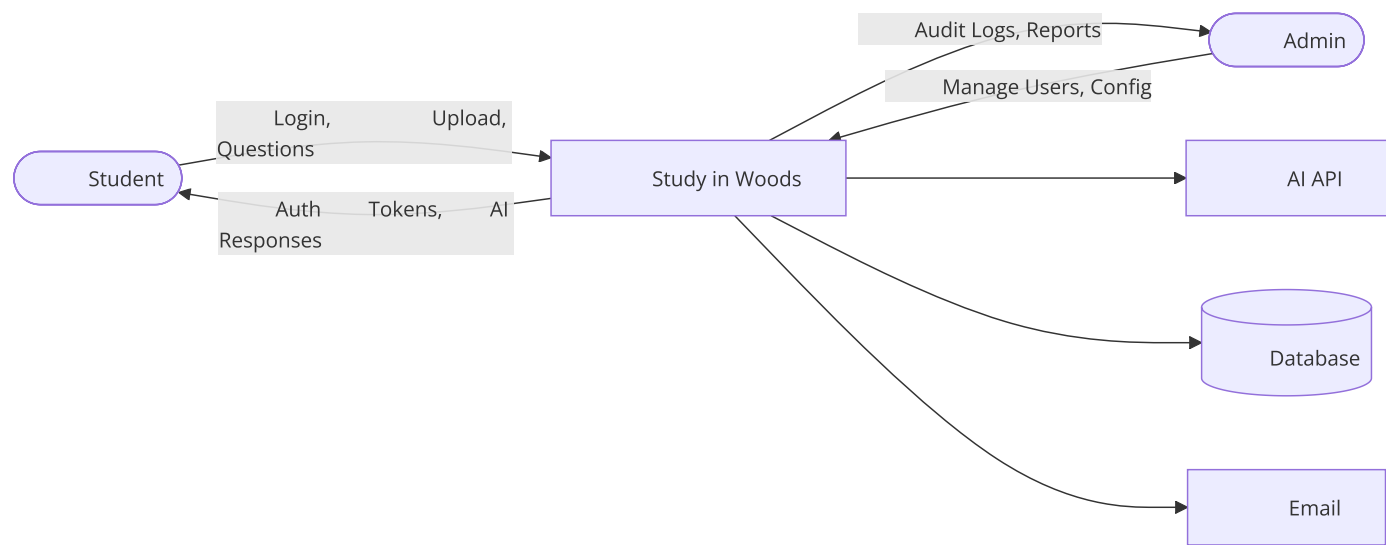


Figure 6.2: Context Diagram (Level 0 DFD)

### 6.3.2 System Decomposition (Level 1)

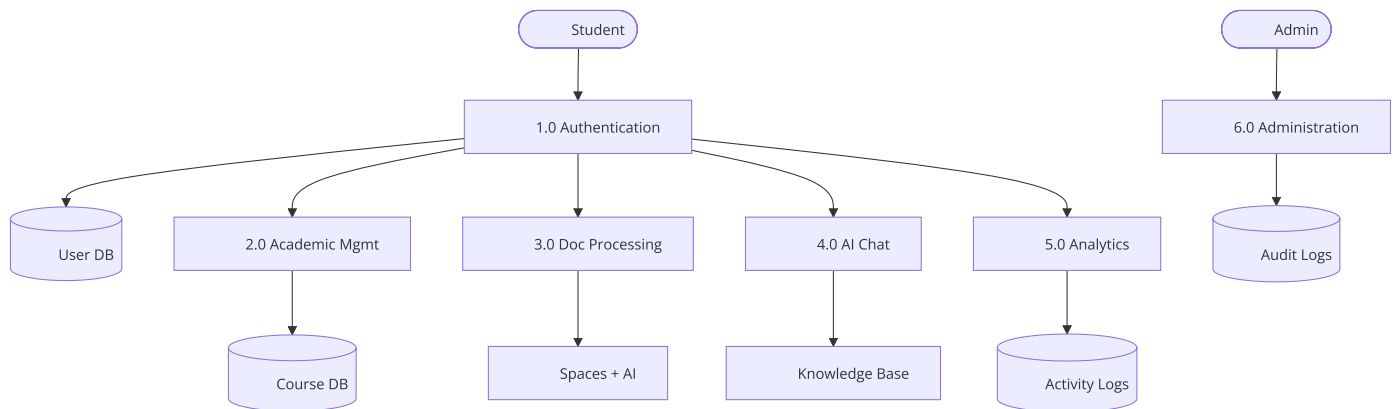


Figure 6.3: System Decomposition (Level 1)

## 6.4 Entity Relationship Diagram

Database: 30+ tables. Core hierarchy: University → Course → Semester → Subject → Documents/ChatSessions.

University		
int	id	PK
string	name	
string	code	

Course		
--------	--	--

Semester		
----------	--	--

Subject		
int	id	PK
string	kb_uuid	

Document		
int	id	PK
string	status	

User		
int	id	PK
string	email	
string	role	

ChatSession		
-------------	--	--

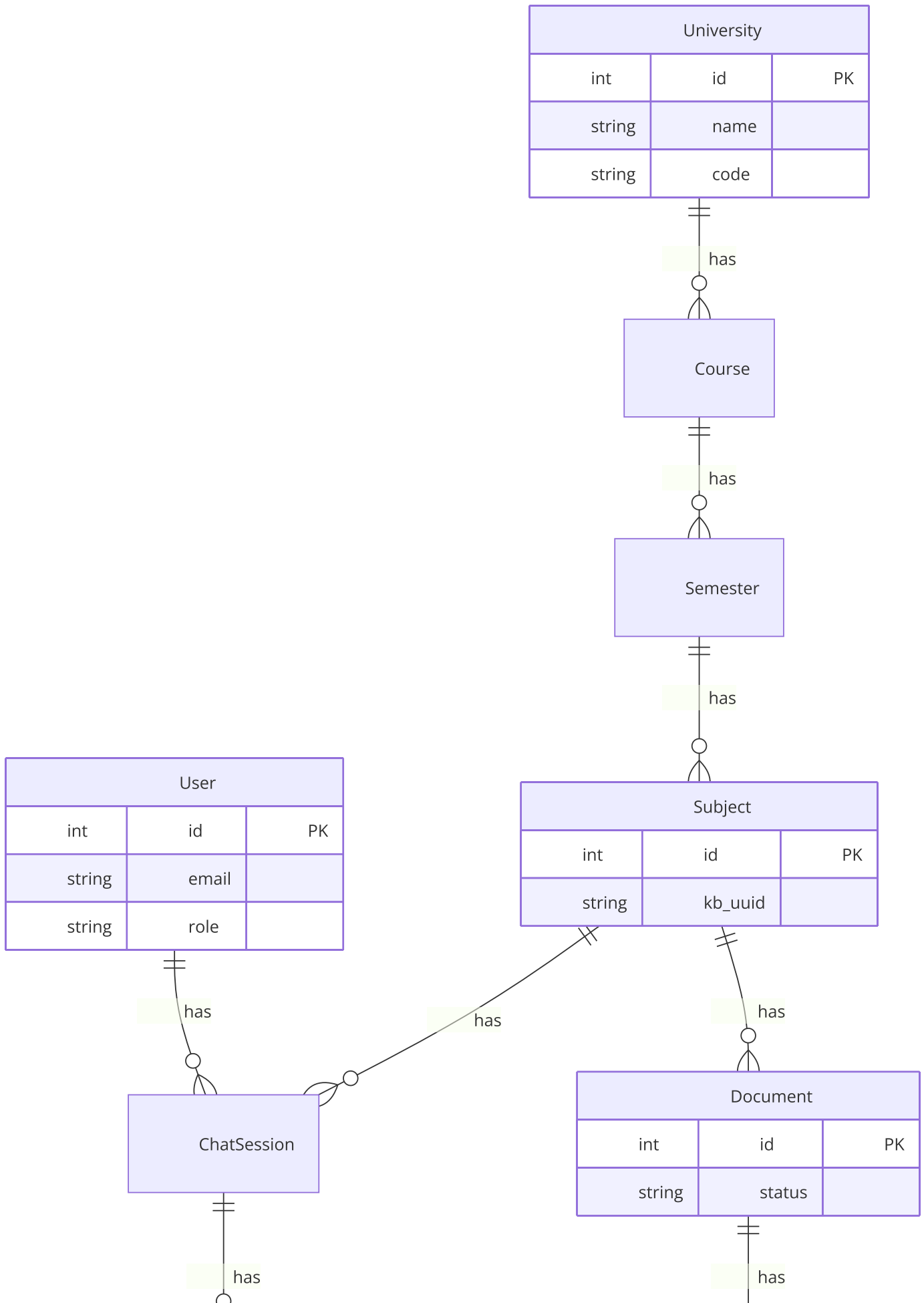




Figure 6.4: Entity Relationship Diagram

## 6.5 Key Sequence Flows

### 6.5.1 Authentication Flow

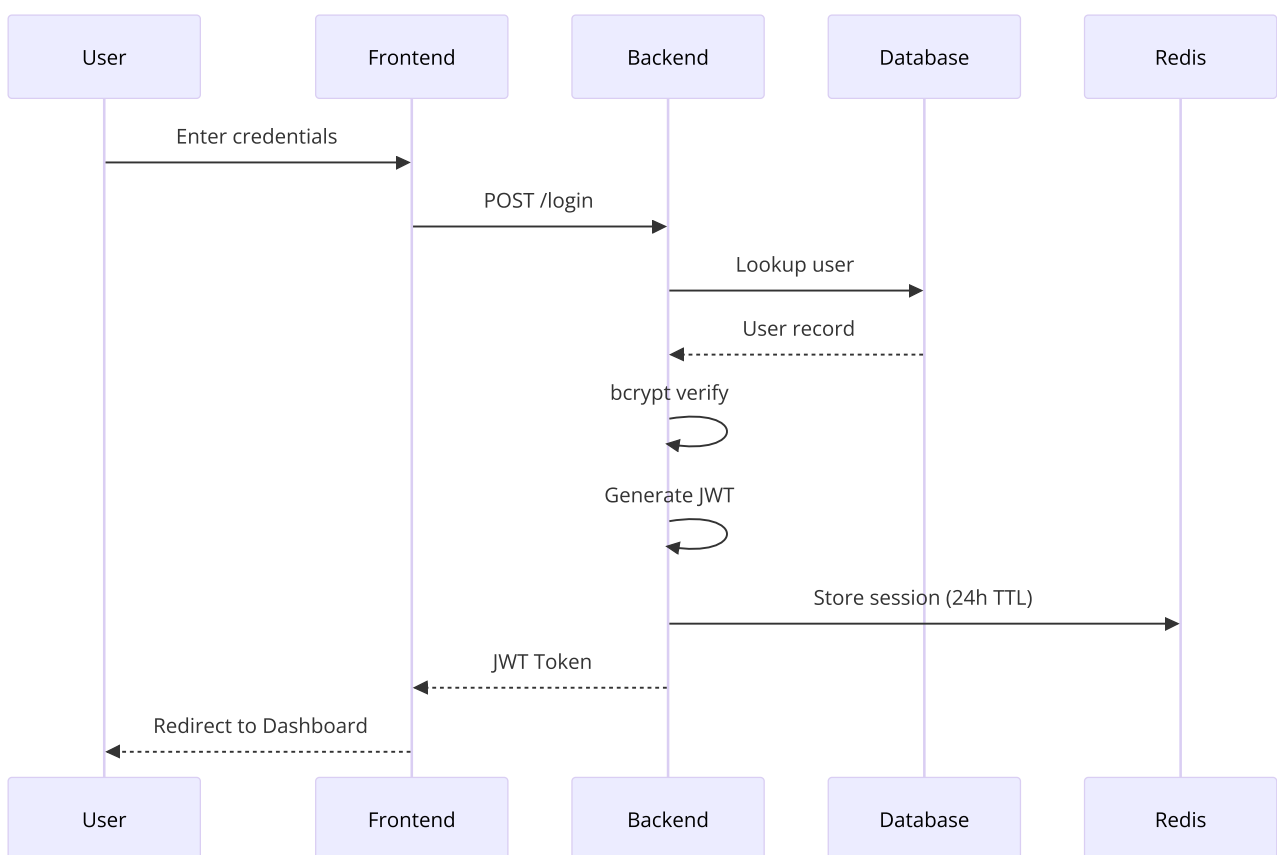


Figure 6.5: Authentication Sequence Diagram

6.5.2 Document Upload & AI Extraction

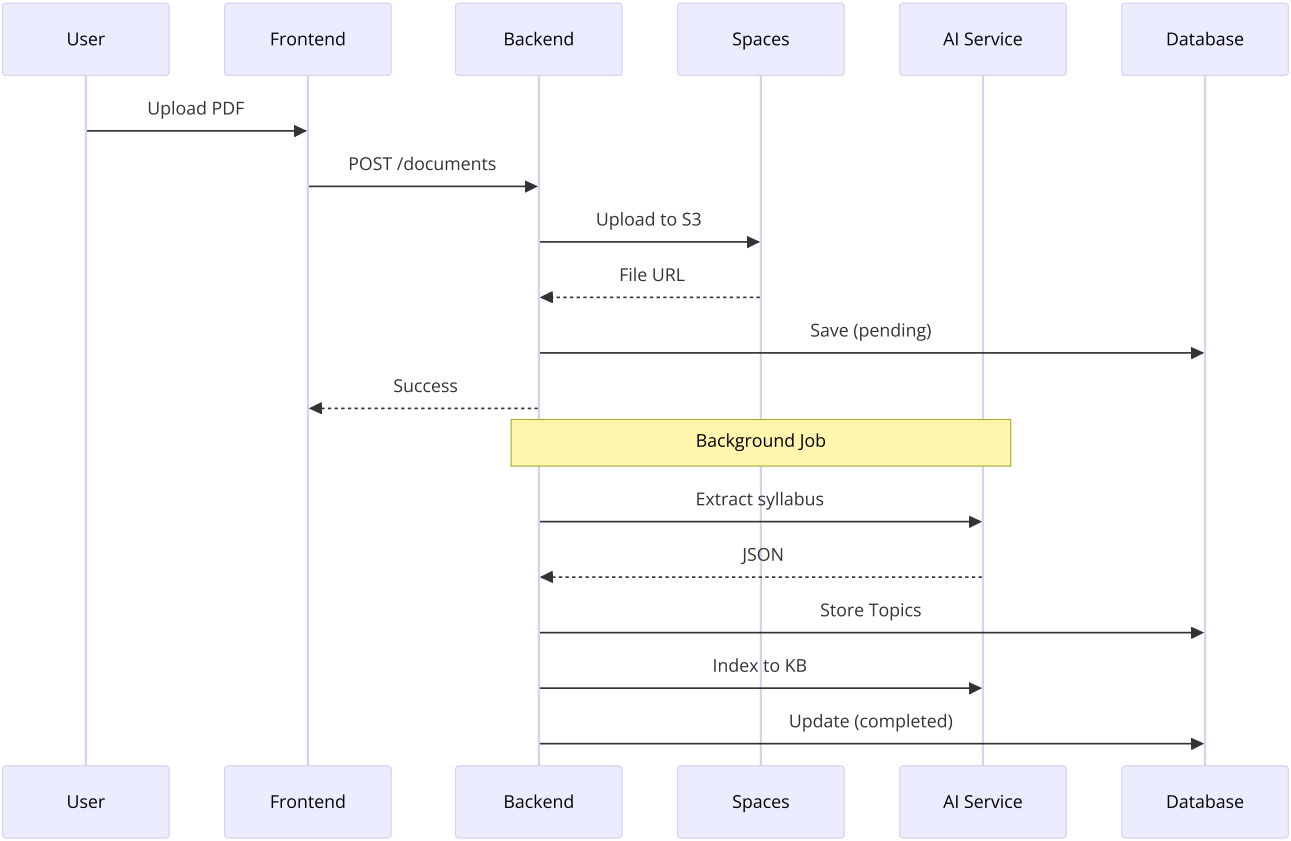
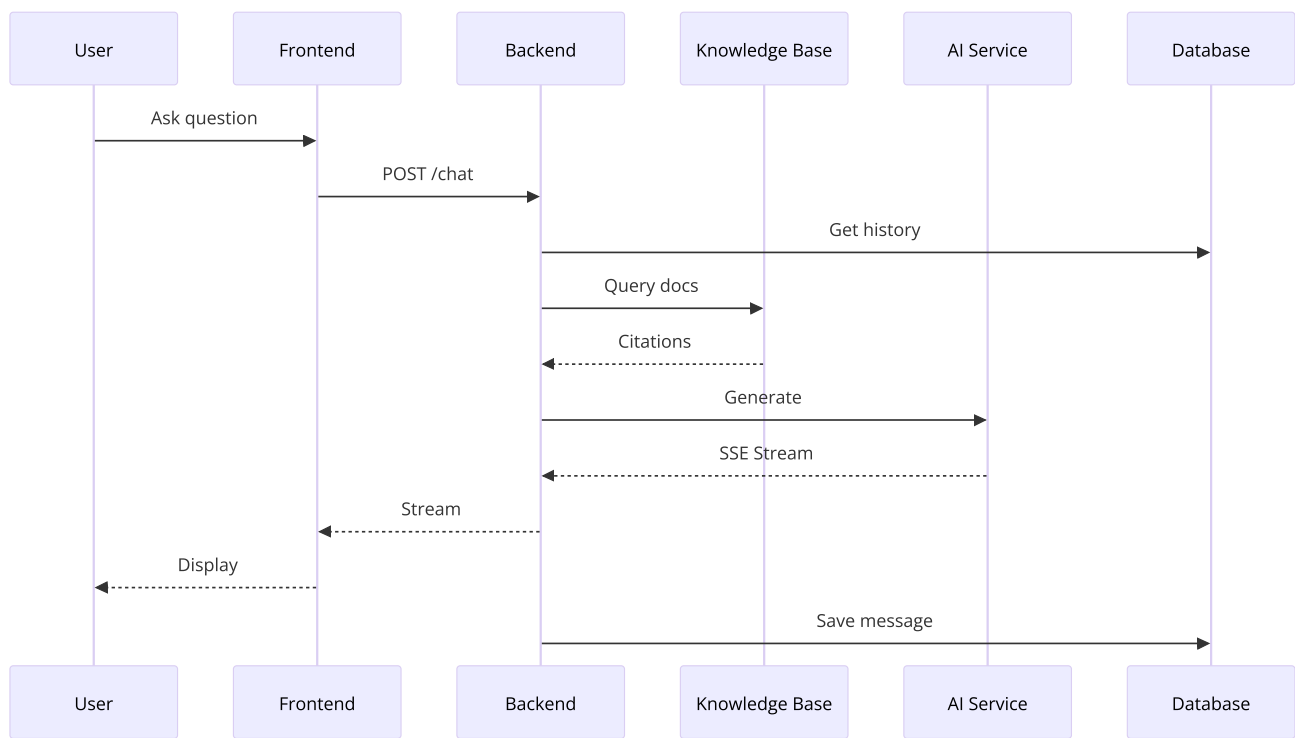


Figure 6.6: Document Upload and AI Extraction Flow

6.5.3 AI Chat with RAG



*Figure 6.7: AI Chat with RAG Flow*



## 6.6 Component Architecture

### 6.6.1 Backend Layers

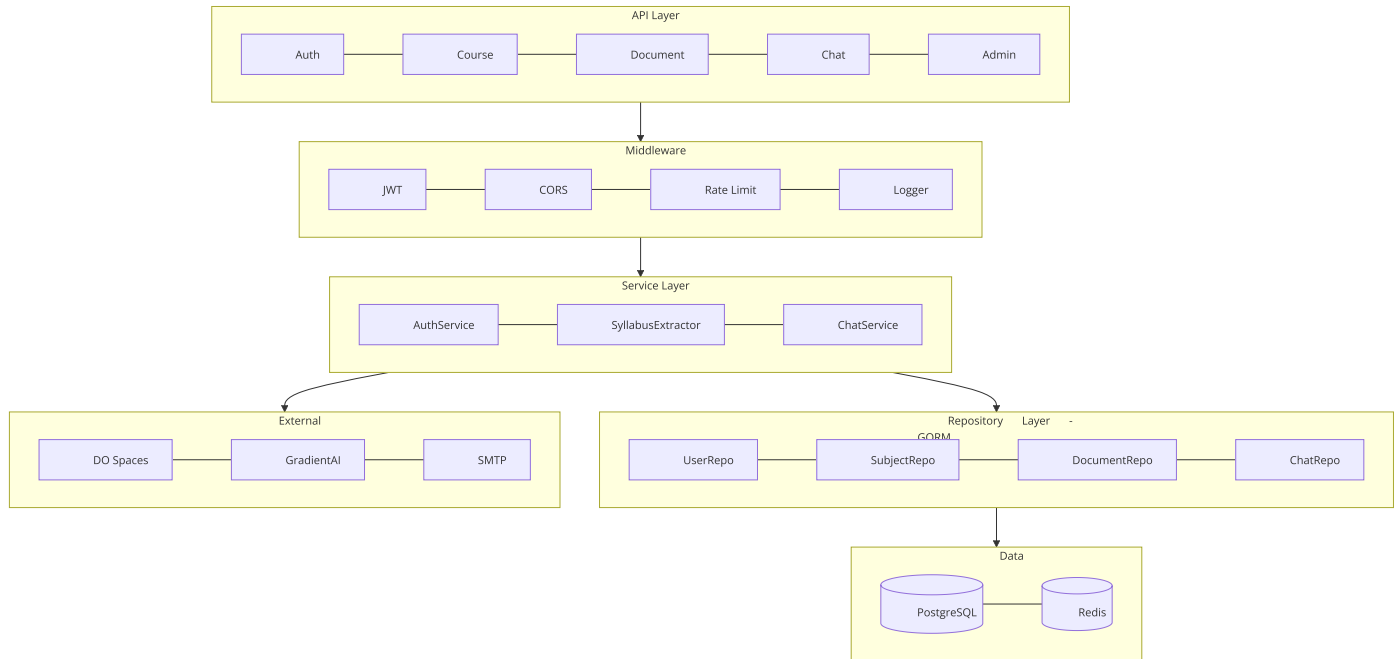


Figure 6.8: Backend Layered Architecture

### 6.6.2 Frontend Structure

The Next.js 15 frontend follows a modular architecture with clear separation of concerns:

Layer	Components	Purpose
App Router	Next.js 15 App Directory	File-based routing with layouts
Layout	Header, Sidebar, Footer	Consistent UI shell across pages
Auth Pages	/login, /register	User authentication flows
Dashboard	Stats, Activity, Actions	Overview and quick actions
Academic	/universities, /courses, /subjects	Academic hierarchy management
Chat	/chat/[subjectId]	AI chat with sessions, messages, citations
Analytics	Charts, Stats, Activity Table	Usage insights and metrics
Admin	Users, Settings, Audit Logs	System administration

Layer	Components	Purpose
Providers	Query, Theme, Auth	Global state and context
Utils	API Client (Axios), Hooks, Zod	Shared utilities and validation

Figure 6.9: Frontend Application Structure