**Barma-AI Frontend Architecture Overview**

**🧩 Project Structure**

**Root Folder: src/**

src/

├── components/

│ ├── Sidebar/

│ │ ├── DocumentList.jsx

│ │ ├── UploadButton.jsx

│ │ ├── PDFViewer.jsx

│ │ └── index.jsx

│ ├── ChatBox/

│ │ ├── ChatMessage.jsx

│ │ ├── ChatInput.jsx

│ │ └── index.jsx

│ ├── MindMapView/

│ │ ├── MindMap.jsx

│ │ ├── NotesEditor.jsx

│ │ └── index.jsx

│ └── App.js

├── styles/

│ ├── App.css

│ ├── ChatBox.css

│ ├── ChatInput.css

│ ├── ChatMessage.css

│ ├── MindMapView.css

│ ├── PDFViewer.css

│ ├── Sidebar.css

├── utils/

│ └── api.js

├── index.js

└── setupTests.js

**⚙️ Core Technologies**

* **React** with functional components and hooks
* **React Resizable Panels** for flexible 3-panel layout
* **Fetch API** and centralized api.js for backend communication
* **CSS Modules** with custom styles per component

**🧱 Component Architecture**

**App.js**

* Root layout using PanelGroup.
* Manages global state: selectedDocumentId.
* Connects Sidebar, ChatBox, and MindMapView.

**Sidebar**

* Sidebar/index.jsx: Loads and renders documents, handles selection.
* DocumentList.jsx: Lists uploaded PDFs.
* UploadButton.jsx: Handles file uploads.
* PDFViewer.jsx: Renders selected PDF using URL.
* Props: onSelectDocument() to lift selected document to App.

**ChatBox**

* ChatBox/index.jsx: Manages messages, handles chat logic.
* ChatInput.jsx: Text area for sending user queries.
* ChatMessage.jsx: Renders styled AI/user messages.
* Uses prop documentId to attach queries to selected document.

**MindMapView**

* Currently similar structure to ChatBox.
* MindMap.jsx and NotesEditor.jsx intended for legal brainstorming/summaries.

**🔁 Data Flow**

1. **Document Selection**
   * Sidebar fetches list from backend.
   * On select, updates state in App.js.
2. **Chat Requests**
   * ChatBox receives documentId from App.
   * Sends message and document\_ids to backend via api/chat.
3. **Responses**
   * Responses are rendered with ChatMessage.
   * Handles errors, loading states, and scroll-to-bottom.

**🌐 Backend Interaction (utils/api.js)**

* uploadDocument(file) → POST /api/upload
* fetchDocuments() → GET /api/documents
* chatWithDocument(document\_ids, message) → POST /api/chat
* generateMindMap(document\_id) → POST /api/mindmap

**🎯 Design Principles**

* Modular file organization for clarity and reuse.
* Minimal shared state (lifted to App.js).
* Clean separation of UI and API logic.
* Styled for clarity with responsiveness considered.

This document captures the current technical layout and implementation of the Barma-AI frontend system. It supports a scalable, maintainable foundation for legal document analysis and AI interaction.