Architecture

This project is a **Document Search** where user can search inside pdf documents using a simple UI. For frontend I used **React.js**, for backend we use **Spring Boot**, and for storing and searching and indexing i used **OpenSearch**.

Main idea is that user will type some keywords, system will find related documents in local folder which is on my desktop by searching inside them, and show results fast. Later on I added semantic search using AI embeddings and given a tick option to enable embedding search.

High Level Design

1. **Frontend (React.js)**
    - React app shows a search bar and results section below search bar.
    - User types search text ,it calls backend API.
    - Results are displayed in simple list view.
    - It also shows loading and error state for better user experience.
2. **Backend (Spring Boot)**
    - Backend exposes REST API endpoints like /api/search and /api/index.
    - When indexing, it reads PDF files using **Apache PDFBox** and extracts text.
    - Extracted text is pushed to **OpenSearch** for indexing.
    - On search request , backend queries OpenSearch and returns matched docs.
    - Main components are:
        - **DocumentController.java** → REST endpoints.
        - **DocumentService.java** → business logic like indexing, searching.
3. **OpenSearch (Search Engine)**
    - Stores the documents in indexed format.
    - Supports full text search, filtering, ranking.
    - Communicates with backend using OpenSearch REST client.

Workflow

1. **Indexing Flow**

- Admin/user selects a folder with pdfs.
- Backend extracts content from every pdf.
- Each document is sent to OpenSearch for indexing.

2. **Searching Flow**
   - User enters query on frontend.
   - Frontend calls backend API with query.
   - Backend queries OpenSearch index.
   - Matching results are returned and shown in UI.

Local host URL :

Frontend runs on http://localhost:3000

Backend runs on http://localhost:8080

OpenSearch runs on http://localhost:9200