# 📚 App Innovacion Educativa con Sistema RAG - Estructura Completa

## 📁 Estructura del Proyecto Legal

```

Educa-rag-app/

│

├── 📁 backend/

│ ├── 📁 rag\_system/

│ │ ├── \_\_init\_\_.py

│ │ ├── document\_processor.py

│ │ ├── embedding\_manager.py

│ │ ├── pedagogo\_chunking.py

│ │ ├── search\_engine.py

│ │ └── knowledge\_base.py

│ │

│ ├── 📁 models/

│ │ ├── sentence\_transformers/

│ │ │ ├── paraphrase-multilingual-mpnet-base-v2/

│ │ │ ├── legal-bert-base-uncased/

│ │ │ └── hiiamsid\_sentence\_similarity\_spanish\_es/

│ │ │

│ │ ├── faiss\_indexes/

│ │ │ ├── pedagogia\_codes.index

│ │ │ ├── basic\_education.index

│ │ │ ├── problemas-pedagógicos.index

│ │ │ └── legal\_docs.index

│ │ │

│ │ └── metadata/

│ │ ├── legal\_metadata.pkl

│ │ ├── jurisprudence\_metadata.pkl

│ │ └── contracts\_metadata.pkl

│ │

│ ├── 📁 api/

│ │ ├── \_\_init\_\_.py

│ │ ├── razonamiento-científico.py

│ │ ├── document\_handler.py

│ │ ├── response\_generator.py

│ │ └── memory\_manager.py

│ │

│ └── 📁 utils/

│ ├── text\_preprocessing.py

│ ├── legal\_patterns.py

│ ├── validation.py

│ └── export\_utils.py

│

├── 📁 frontend/

│ ├── 📁 assets/

│ │ ├── 📁 css/

│ │ │ ├── main.css

│ │ │ ├── legal-components.css

│ │ │ └── rag-interface.css

│ │ │

│ │ ├── 📁 js/

│ │ │ ├── core/

│ │ │ │ ├── app.js

│ │ │ │ ├── rag-client.js

│ │ │ │ ├── document-analyzer.js

│ │ │ │ └── memory-system.js

│ │ │ │

│ │ │ ├── modules/

│ │ │ │ ├── legal-search.js

│ │ │ │ ├── contract-analyzer.js

│ │ │ │ ├── jurisprudence-finder.js

│ │ │ │ └── document-composer.js

│ │ │ │

│ │ │ └── workers/

│ │ │ ├── embedding-worker.js

│ │ │ └── search-worker.js

│ │ │

│ │ └── 📁 images/

│ │ ├── icons/

│ │ └── logos/

│ │

│ ├── 📁 components/

│ │ ├── header.html

│ │ ├── sidebar-legal.html

│ │ ├── search-interface.html

│ │ ├── document-uploader.html

│ │ ├── rag-chat.html

│ │ └── results-viewer.html

│ │

│ └── index.html

│

├── 📁 data/

│ ├── 📁 legal\_knowledge/

│ │ ├── 📁 codes/

│ │ │ ├── codigo-civil.pdf

│ │ │ ├── codigo-penal.pdf

│ │ │ ├── codigo-comercio.pdf

│ │ │ └── codigo-procedimientos.pdf

│ │ │

│ │ ├── 📁 jurisprudence/

│ │ │ ├── scjn/

│ │ │ ├── tribunales-colegiados/

│ │ │ └── plenos-regionales/

│ │ │

│ │ ├── 📁 contracts/

│ │ │ ├── templates/

│ │ │ ├── precedents/

│ │ │ └── analysis/

│ │ │

│ │ ├── 📁 regulations/

│ │ │ ├── federal/

│ │ │ ├── state/

│ │ │ └── municipal/

│ │ │

│ │ └── 📁 doctrine/

│ │ ├── civil/

│ │ ├── penal/

│ │ ├── comercial/

│ │ └── constitucional/

│ │

│ ├── 📁 user\_documents/

│ │ ├── 📁 cases/

│ │ ├── 📁 contracts/

│ │ ├── 📁 briefs/

│ │ └── 📁 analysis/

│ │

│ └── 📁 templates/

│ ├── 📁 contracts/

│ ├── 📁 briefs/

│ ├── 📁 motions/

│ └── 📁 analysis/

│

├── 📁 config/

│ ├── rag\_config.json

│ ├── legal\_patterns.json

│ ├── model\_settings.json

│ └── user\_preferences.json

│

├── 📁 scripts/

│ ├── setup\_models.py

│ ├── build\_indexes.py

│ ├── update\_knowledge.py

│ └── backup\_data.py

│

├── requirements.txt

├── setup.py

├── README.md

└── run\_app.py

```

## 🔧 Componentes Principales

### 1. Sistema RAG Backend

\*\*document\_processor.py:\*\*

```python

class LegalDocumentProcessor:

def \_\_init\_\_(self):

self.model = SentenceTransformer('paraphrase-multilingual-mpnet-base-v2')

self.legal\_patterns = self.load\_legal\_patterns()

def process\_legal\_document(self, file\_path, doc\_type):

# Extractores específicos por tipo

if doc\_type == 'codigo':

return self.process\_legal\_code(file\_path)

elif doc\_type == 'jurisprudencia':

return self.process\_jurisprudence(file\_path)

elif doc\_type == 'contrato':

return self.process\_contract(file\_path)

def legal\_chunking(self, text, chunk\_type='article'):

# Chunking especializado para documentos legales

pass

def create\_embeddings(self, chunks):

# Crear embeddings con modelo legal

pass

```

\*\*legal\_chunking.py:\*\*

```python

def structured\_legal\_chunking(text, document\_type):

"""Chunking inteligente según tipo de documento"""

chunkers = {

'codigo': article\_chunking,

'jurisprudencia': thesis\_chunking,

'contrato': clause\_chunking,

'demanda': section\_chunking,

'sentencia': ruling\_chunking

}

return chunkers.get(document\_type, general\_chunking)(text)

def article\_chunking(text):

"""Para códigos legales - por artículos"""

pass

def thesis\_chunking(text):

"""Para jurisprudencia - por tesis"""

pass

def clause\_chunking(text):

"""Para contratos - por cláusulas"""

pass

```

### 2. Motor de Búsqueda Especializado

\*\*search\_engine.py:\*\*

```python

class LegalSearchEngine:

def \_\_init\_\_(self):

self.indexes = {

'codes': faiss.read\_index('models/faiss\_indexes/legal\_codes.index'),

'jurisprudence': faiss.read\_index('models/faiss\_indexes/jurisprudence.index'),

'contracts': faiss.read\_index('models/faiss\_indexes/contracts.index')

}

def legal\_search(self, query, search\_type='all', filters=None):

"""Búsqueda especializada por tipo legal"""

pass

def precedent\_search(self, legal\_issue, jurisdiction=None):

"""Búsqueda de precedentes"""

pass

def article\_search(self, code\_name, article\_number=None):

"""Búsqueda específica de artículos"""

pass

def contextual\_search(self, query, context\_docs=None):

"""Búsqueda con contexto de caso"""

pass

```

### 3. Frontend con RAG Integration

\*\*rag-client.js:\*\*

```javascript

class LegalRAGClient {

constructor() {

this.searchEngine = new LegalSearchEngine();

this.documentAnalyzer = new LegalDocumentAnalyzer();

this.memorySystem = new LegalMemorySystem();

}

async searchLegalContent(query, type = 'all') {

// Búsqueda en base de conocimientos legal

}

async analyzeDocument(document, analysisType) {

// Análisis de documento con contexto legal

}

async generateLegalResponse(query, context) {

// Generación de respuesta con RAG

}

}

```

### 4. Configuración RAG

\*\*rag\_config.json:\*\*

```json

{

"embedding\_models": {

"primary": "paraphrase-multilingual-mpnet-base-v2",

"legal\_specialist": "nlpaueb/legal-bert-base-uncased",

"spanish\_specialist": "hiiamsid/sentence\_similarity\_spanish\_es"

},

"search\_settings": {

"top\_k": 5,

"similarity\_threshold": 0.7,

"rerank": true,

"context\_window": 3000

},

"chunking\_strategies": {

"codigo": {

"method": "article\_based",

"max\_chunk\_size": 1500,

"overlap": 100

},

"jurisprudencia": {

"method": "thesis\_based",

"max\_chunk\_size": 2000,

"overlap": 150

},

"contrato": {

"method": "clause\_based",

"max\_chunk\_size": 1000,

"overlap": 50

}

}

}

```

### 5. Templates Legales con RAG

\*\*legal\_patterns.json:\*\*

```json

{

"document\_types": {

"demanda": {

"structure": ["proemio", "hechos", "derecho", "petitorio"],

"legal\_references": ["articulos", "jurisprudencia", "doctrine"],

"search\_context": "litigation"

},

"contrato": {

"structure": ["partes", "antecedentes", "clausulas", "firmas"],

"legal\_references": ["codigo\_civil", "codigo\_comercio"],

"search\_context": "contracts"

}

}

}

```

## 🚀 Características Avanzadas

### 1. \*\*Búsqueda Semántica Legal\*\*

- Búsqueda por conceptos jurídicos

- Encuentra precedentes similares

- Contextualiza con jurisprudencia

### 2. \*\*Análisis Inteligente de Documentos\*\*

- Identifica cláusulas problemáticas

- Sugiere mejoras basadas en precedentes

- Extrae argumentos legales

### 3. \*\*Generación Asistida\*\*

- Contratos con cláusulas optimizadas

- Escritos con fundamentos sólidos

- Análisis con jurisprudencia relevante

### 4. \*\*Sistema de Memoria Legal\*\*

- Aprende de tus casos anteriores

- Mejora con cada documento procesado

- Base de conocimientos personalizada

¿Te gusta esta estructura? ¿Empezamos implementando algún componente específico?