**"Organizador Personal de Recursos Profesionales"** (**"KnowledgeHub"**). Es una aplicación web para centralizar información útil en tu carrera, con funcionalidades prácticas pero sin excesiva complejidad.

📝 Descripción del Proyecto

Un sistema donde puedas guardar, organizar y buscar recursos como:

* Enlaces web (artículos, tutoriales, documentación)
* Documentos locales (PDFs, imágenes)
* Notas rápidas
* Tareas pendientes  
  Con capacidad para etiquetar contenido y buscar eficientemente.

⚙️ Stack Tecnológico

| **Componente** | **Tecnología** |
| --- | --- |
| Backend | Python + Flask |
| Frontend | HTML + CSS + Bootstrap |
| Base de Datos | MongoDB |
| Contenedorización | Docker |
| Extra | Requests (descargas web) |

🧩 Funcionalidades Clave

1. **Gestión de Recursos**
   * Añadir URLs con captura automática de título
   * Subir documentos (PDF, imágenes)
   * Crear notas rápidas
   * Sistema de etiquetas (ej: "Python", "Docker", "Urgente")
2. **Búsqueda Inteligente**
   * Filtros por tipo (enlace/nota/documento)
   * Búsqueda por palabras clave en contenido
   * Búsqueda por etiquetas
3. **Descargas Web (Extra)**
   * Guardar copia local de páginas web (HTML)
   * Almacenar contenido textual para búsquedas offline
4. **Organización Visual**
   * Vista de tarjetas clasificables
   * Sistema de favoritos
   * Marcado de recursos "Para revisar"

🗄️ Estructura de la Base de Datos (MongoDB)

*// Colección: recursos*

{

"\_id": ObjectId,

"tipo": "enlace|nota|documento|tarea",

"titulo": String,

"descripcion": String,

"contenido": String, *// Texto para notas o HTML descargado*

"url": String, *// Para enlaces*

"ruta\_archivo": String, *// Para documentos subidos*

"etiquetas": [String],

"favorito": Boolean,

"fecha\_creacion": DateTime,

"para\_revisar": Boolean

}

🐳 Configuración Docker (docker-compose.yml)

version: '3.8'

services:

web:

build: .

ports:

- "5000:5000"

volumes:

- ./app:/app

- ./uploads:/app/uploads *# Para archivos subidos*

environment:

- MONGO\_URI=mongodb://db:27017/knowledgehub

depends\_on:

- db

db:

image: mongo:5.0

volumes:

- db\_data:/data/db

environment:

- MONGO\_INITDB\_DATABASE=knowledgehub

volumes:

db\_data:

📂 Estructura de Archivos

knowledgehub/

├── Dockerfile

├── docker-compose.yml

├── requirements.txt

└── app/

├── \_\_init\_\_.py

├── routes.py

├── models.py

├── utils.py # Funciones helper (ej: descargar webs)

├── templates/

│ ├── base.html

│ ├── index.html

│ ├── add\_resource.html

│ └── search.html

├── static/

│ ├── css/

│ └── js/

└── uploads/ # Carpeta para archivos subidos

🧠 Conceptos que Practicarás

1. **Flask**:
   * Rutas, formularios, sesiones
   * Manejo de archivos estáticos
   * Renderizado de templates Jinja2
2. **MongoDB**:
   * Operaciones CRUD
   * Consultas con filtros compuestos
   * Gestión de documentos binarios (GridFS opcional)
3. **Docker**:
   * Contenedorización de app y DB
   * Volúmenes persistentes
   * Comunicación entre contenedores
4. **Frontend**:
   * Diseño responsive con Bootstrap
   * Interacciones AJAX (para búsquedas dinámicas)

💡 Ejemplo de Código (Flask + MongoDB)

*# routes.py - Añadir recurso*

@app.route('/add', methods=['POST'])

def add\_resource():

tipo = request.form['tipo']

nuevo\_recurso = {

"tipo": tipo,

"titulo": request.form['titulo'],

"etiquetas": request.form.getlist('etiquetas'),

"fecha\_creacion": datetime.utcnow()

}

if tipo == "enlace":

nuevo\_recurso["url"] = request.form['url']

*# Opcional: Descargar contenido web automático*

if 'descargar' in request.form:

contenido = utils.descargar\_web(request.form['url'])

nuevo\_recurso["contenido"] = contenido

*# Insertar en MongoDB*

mongo.db.recursos.insert\_one(nuevo\_recurso)

return redirect(url\_for('index'))

🚀 Pasos para Implementar

1. Configurar entorno Docker con MongoDB
2. Crear sistema de autenticación básico (opcional pero recomendado)
3. Implementar CRUD para recursos
4. Desarrollar sistema de búsqueda con filtros
5. Añadir función de descarga web (usando requests + BeautifulSoup)
6. Diseñar interfaz con Bootstrap
7. Implementar subida de archivos (usar flask.secure\_filename)

🌟 Beneficios para tu Carrera

* **Organización centralizada**: Todo tu material profesional en un solo lugar
* **Acceso desde cualquier dispositivo**: Web = disponible siempre
* **Base para proyectos futuros**: Puedes expandirlo (ej: añadir APIs, IA para clasificación automática)
* **Portafolio tangible**: Demuestra tus habilidades full-stack

**Dificultad estimada**: ⭐⭐☆☆☆ (Nivel intermedio, ideal para consolidar conocimientos sin frustraciones)

🧭 Cómo funciona la aplicación actualmente:

1. **Página principal (index):**
   * URL: http://localhost:5000/
   * Muestra una lista de recursos (inicialmente vacía)
   * Tienes un botón "Añadir Nuevo Recurso" que lleva al formulario
2. **Formulario de añadir recurso:**
   * URL: http://localhost:5000/add
   * Permite seleccionar entre 3 tipos de recursos:
     + 📎 **Enlace web** (con opción para descargar contenido)
     + 📝 **Nota** (texto libre)
     + 📄 **Documento** (subir archivos)

📝 Cómo añadir información:

Paso 1: Accede al formulario

1. Ve a http://localhost:5000/
2. Haz clic en el botón **"Añadir Nuevo Recurso"**

Paso 2: Completa el formulario

* **Campos comunes a todos los recursos:**
  + **Tipo de recurso:** Selecciona uno de los 3 tipos
  + **Título:** Nombre descriptivo del recurso
  + **Descripción:** Información adicional
  + **Etiquetas:** Palabras clave separadas por comas (ej: python, docker, web)
* **Campos específicos:**
  + Para **Enlace web**:
    - URL: Dirección completa (ej: https://es.wikipedia.org/)
    - Opción: "Descargar contenido para búsqueda offline" (marca si quieres guardar el texto de la página)
  + Para **Nota**:
    - Contenido: Texto libre (puedes escribir lo que necesites)
  + Para **Documento**:
    - Subir archivo: Selecciona un archivo de tu computadora (PDF, imagen, etc.)

Paso 3: Guarda el recurso

* Haz clic en el botón **"Guardar"**
* Serás redirigido a la página principal donde verás:
  + Un mensaje de confirmación: "Recurso añadido correctamente!"
  + Tu nuevo recurso aparecerá en la lista

🔍 Ejemplos prácticos:

1. **Guardar un enlace útil:**
   * Tipo: Enlace web
   * Título: "Documentación oficial de Flask"
   * URL: https://flask.palletsprojects.com/
   * Etiquetas: flask, python, documentación
   * Marcar "Descargar contenido" para tenerlo disponible offline
2. **Crear una nota:**
   * Tipo: Nota
   * Título: "Comandos Docker importantes"
   * Contenido:

text

Copy

Download

docker-compose up --build

docker ps -a

docker exec -it nombre\_contenedor bash

* + Etiquetas: docker, comandos

1. **Subir un documento:**
   * Tipo: Documento
   * Título: "Apuntes MongoDB"
   * Seleccionar archivo: mongodb-cheatsheet.pdf
   * Etiquetas: mongodb, base de datos, cheatsheet