🤖 Retrieval-Augmented Generation (RAG) Chatbot

Clean, production-style notebook. Concise sections, minimal narration, technical comments only.

Sections

1. Imports
2. Data Ingestion
3. Text Preprocessing
4. Chunking
5. Embedding Generation
6. Vector Indexing
7. Retrieval + Generation (RAG QA)
8. RAG Pipeline
9. Chatbot Interaction
10. Evaluation

1. Imports

In [39]:
```
import os
from langchain.embeddings import OpenAIEmbeddings
from langchain.vectorstores import FAISS
from langchain.text_splitter import CharacterTextSplitter
from langchain.document_loaders import TextLoader
from langchain.chains import RetrievalQA
from langchain.llms import OpenAI
from langchain.docstore.document import Document
from openai import OpenAI
```

2. Data Ingestion

In [40]:
```
path_file = "../data/Manual_RRHH_Absurdo.txt"
loader = TextLoader(path_file)
docs=loader.load()
print(docs[0].page_content[:800])
```

MANUAL DE RECURSOS HUMANOS – EMPRESA "CAFÉ Y SIÉSTA S.L."

Bienvenido a bordo, querido mortal. Si estás leyendo esto, probablemente fui
ste lo suficientemente astuto como para engañar al departamento de selecció
n. Felicidades.

1. INGRESO A LA EMPRESA

El primer día de trabajo se te entregará una taza personalizada con tu nombr
e mal escrito, una silla giratoria que no gira y una lista de contraseñas qu
e no funcionan. Tu computadora tardará entre 2 y 17 días en ser configurada.
Durante ese tiempo, se espera que medites sobre el propósito de la vida labo
ral.

Tu jefe te saludará con un "ah, ¿hoy empezabas?", y te asignará una tarea ur
gente sobre un proyecto del que nadie quiere hablar. No preguntes, simplemen
te asiente con cara de experto.

2. HORARIOS

El horario oficial es de 9:00 a 23:00.

3. Text Preprocessing

Text Cleaning / Normalization

In [41]:
```
import re
import unicodedata

def clean_text(text):
    text = text.lower()  #
    text = re.sub(r"http\S+", "", text)  # URLs
    text = re.sub(r"\S+@\S+", "", text)  # emails
    text = re.sub(r"[^\w\s]", " ", text)  # special characters
    text = re.sub(r"\d{2}/\d{2}/\d{4}", "", text)  # dates
    text = unicodedata.normalize("NFKD", text).encode("ASCII", "ignore").dec
    text = re.sub(r"[^\w\s\n]", " ", text)  # extra whitespace
    return text.strip()

cleaned_text = clean_text(docs[0].page_content)
print(cleaned_text[:800])
```

```
manual de recursos humanos   empresa  cafe y siesta s l

bienvenido a bordo  querido mortal  si estas leyendo esto  probablemente fui
ste lo suficientemente astuto como para enganar al departamento de seleccion
felicidades

1  ingreso a la empresa

el primer dia de trabajo se te entregara una taza personalizada con tu nombr
e mal escrito  una silla giratoria que no gira y una lista de contrasenas qu
e no funcionan  tu computadora tardara entre 2 y 17 dias en ser configurada
durante ese tiempo  se espera que medites sobre el proposito de la vida labo
ral

tu jefe te saludara con un  ah   hoy empezabas    y te asignara una tarea ur
gente sobre un proyecto del que nadie quiere hablar  no preguntes  simplemen
te asiente con cara de experto

2  horarios

el horario oficial es de 9 00 a 23 00
```

4. Chunking

Document Splitting

In [42]:
```python
docs = [Document(page_content=cleaned_text)]
splitter = CharacterTextSplitter(chunk_size=500, chunk_overlap=50)
chunks = splitter.split_documents(docs)
print("Chars in cleaned text:", len(cleaned_text))
print("Total_chunks:", len(chunks))
```

```
Chars in cleaned text: 3393
Total_chunks: 9
```

5. Embedding Generation

In [43]:
```python
from langchain.embeddings import HuggingFaceEmbeddings
embedding = HuggingFaceEmbeddings(model_name="all-MiniLM-L6-v2")
```

6. Vector Indexing

FAISS Index Building

In [44]:
```python
vectorstore = FAISS.from_documents(chunks, embedding)
print("Index created with:", vectorstore.index.ntotal, "vectors")
```

```
Index created with: 9 vectors
```

7. Retrieval

In [ ]:
```
client = OpenAI(api_key="OPENROUTER_API_KEY", base_url="https://openrouter.a
```

In [46]:
```
def retrieve(query: str, k: int = 3):
    return vectorstore.similarity_search(query, k=k)
```

8. RAG Pipeline

In [47]:
```
def build_context(docs):
    return "\n\n".join(d.page_content for d in docs)
```

In [48]:
```
def generate_answer(query: str, docs):
    context = build_context(docs)
    resp = client.chat.completions.create(
        model="deepseek/deepseek-r1:free",
        messages=[
            {"role": "system", "content": "Responde SOLO usando el contexto
            {"role": "user", "content": f"Contexto:\n{context}\n\nPregunta:\
        ]
    )
    return resp.choices[0].message.content
```

In [49]:
```
def rag_answer(query: str, k: int = 3):
    docs = retrieve(query, k=k)
    answer = generate_answer(query, docs)
    return answer, docs
```

9. Chatbot Interaction

In [50]:
```
demo_questions = [
    "¿Como debo solicitar mis vacaciones?",
    "Cual es el horario oficial?",
    "Que pasa el primer dia?",
]

for q in demo_questions:
    ans, used = rag_answer(q, k=3)
    print(f"\nQ: {q}\nA: {ans}\nDocs usados: {len(used)}")
```

```
Q: ¿Como debo solicitar mis vacaciones?
A: Para solicitar tus vacaciones, según el manual, debes redactar una solici
tud formal escrita **en pergamino**, sellarla con **cera caliente** y entreg
arla en mano al **unicornio de la oficina**. Si el unicornio está ocupado, p
uedes intentar usar el **formulario online**, aunque se aclara que este "jam
ás funciona".

No se mencionan más opciones ni detalles adicionales en el contexto proporci
onado.
Docs usados: 3

Q: Cual es el horario oficial?
A: El horario oficial mencionado en el contexto es de **9:00 a 23:00**. Así
lo establece el punto 2, aunque también se aclara que el "horario real" es m
ás flexible, sujeto a sutiles presiones sociales.
Docs usados: 3

Q: Que pasa el primer dia?
A: Según el manual, **el primer día de trabajo** recibirás:
- Una taza personalizada con tu nombre mal escrito.
- Una silla giratoria que no gira.
- Una lista de contraseñas que no funcionan.

Además, tu computadora tardará entre **2 y 17 días en ser configurada**, y d
urante ese tiempo debes **meditar sobre el propósito de la vida laboral** (c
ontexto punto 1).
Docs usados: 3
```

10. Evaluation

In [51]:

```python
EVAL_SET = [
    {
        "query": "¿Como debo solicitar mis vacaciones?",
        "keywords": ["pergamino", "cera", "unicornio"]
    },
    {
        "query": "Cual es el horario oficial?",
        "keywords": ["9", "23", "constructo", "murmura"]
    },
    {
        "query": "Que sucede el primer dia de trabajo?",
        "keywords": ["taza", "contraseña", "giratoria"]
    }
]

def evaluate(eval_set):
    results = []
    for item in eval_set:
        ans, _ = rag_answer(item["query"])
        text = ans.lower()
        hit = sum(1 for kw in item["keywords"] if kw in text)
        coverage = hit / len(item["keywords"])
        results.append((item["query"], round(coverage, 2), ans[:140] + ("...
    return results
```

```python
print("\n=== Evaluation ===")
for q, cov, preview in evaluate(EVAL_SET):
    print(f"[{cov}] {q} -> {preview}")
```

```
=== Evaluation ===
[1.0] ¿Como debo solicitar mis vacaciones? -> Para solicitar vacaciones, deb
es presentar una solicitud formal escrita en pergamino, sellada con cera cal
iente, y entregarla en mano al uni...
[0.5] Cual es el horario oficial? -> El horario oficial mencionado en el con
texto es de **9:00 a 23:00**.

Fuente: Sección **2. Horarios** del manual proporcionado.
[1.0] Que sucede el primer dia de trabajo? -> Según el contexto proporcionad
o, **el primer día de trabajo suceden las siguientes cosas**:
1. Se entrega una **taza personalizada con tu ...
```