Documentazione di Sviluppo per la Soluzione di Scraping e Analisi degli Articoli

Link alla Demo

[Guarda la demo su YouTube](https://youtu.be/3DhYOJO5-2U)

Obiettivo del Progetto

L'obiettivo di questa soluzione è creare uno scraper per raccogliere articoli legali da un database, analizzarli e gestirli. La soluzione consente inoltre di:

1. Estrarre articoli rilevanti sulla base di una domanda dell'utente.
2. Analizzare i contenuti degli articoli utilizzando l'API di OpenAI GPT per estrarre parole chiave.
3. Fornire la possibilità di salvare i risultati in file `.txt` personalizzabili.

Passaggi Sviluppati

1. Configurazione Iniziale

Tecnologie utilizzate:

- **Python**: linguaggio principale per lo sviluppo.
- **SQL Server Express**: per la gestione del database.
- **Librerie principali**:

- `requests` e `BeautifulSoup`: per il web scraping.
- `pyodbc`: per la connessione a SQL Server.
- `openai`: per l'interazione con l'API di OpenAI GPT.

2. Creazione del Database

1. Installazione e configurazione di SQL Server Express.
2. Creazione del database `LexAI` e della tabella `Articoli` con la seguente struttura:

```sql
CREATE TABLE Articoli (
    Id INT IDENTITY(1,1) PRIMARY KEY,
    Fonte NVARCHAR(255) NOT NULL,
    Titolo NVARCHAR(255) NOT NULL,
    Contenuto NVARCHAR(MAX),
    Link NVARCHAR(500),
    Keywords NVARCHAR(500),
    EstrazioneKeywords NVARCHAR(1),
    Errore NVARCHAR(100),
    DataInserimento DATETIME DEFAULT GETDATE()
);
```

Note importanti:

- `EstrazioneKeywords` è inizializzato con il valore `'N'` per indicare che le parole chiave non sono ancora state estratte.

- `Errore` è usato per segnalare eventuali problemi con gli articoli (es. contenuto mancante).

3. Web Scraper per Raccogliere Articoli

Lo scraper è stato progettato per:

- Visitare il sito di riferimento.
- Estrarre i link degli articoli.
- Salvare i dettagli degli articoli nel database.

Codice principale dello scraper:

- Connessione a SQL Server tramite `pyodbc`.
- Salvataggio degli articoli con controllo dei duplicati utilizzando `WHERE NOT EXISTS`.

4. Analisi dei Contenuti con OpenAI GPT

Obiettivo: estrarre parole chiave rilevanti dai contenuti degli articoli.

Procedura:

1. Seleziona gli articoli con `EstrazioneKeywords = 'N'`.
2. Usa un prompt personalizzato per richiedere le parole chiave all'API di OpenAI.
3. Aggiorna i record nel database con le parole chiave estratte.

Prompt utilizzato:

"Estrai fino a 20 parole chiave significative dal seguente testo. Rispondi elencando solo le keywords

separate da una virgola."

Codice per l'analisi:

```python
response = openai.ChatCompletion.create(
    model="gpt-4",
    messages=[
        {"role": "system", "content": "Sei esperto nell'analizzare testi giuridici e identificare le parole chiave."},
        {"role": "user", "content": f"Estrai fino a 20 parole chiave dal seguente testo:

{contenuto}"}
    ]
)
```

Note:

- L'uso del modello GPT consente analisi semantiche accurate.
- L'output è salvato nel campo `Keywords` del database e il flag `EstrazioneKeywords` è aggiornato a `'Y'`.

5. Ricerca e Salvataggio dei Risultati

Funzionalità:

- Ricerca degli articoli nel database basandosi su una domanda dell'utente.
- Mostra i risultati trovati (titolo, fonte, link, contenuto).
- Permette di salvare i risultati in file `.txt` in una directory scelta dall'utente.

Codice per il salvataggio personalizzato:

```python
file_name = f"{article['Titolo']}.txt".replace(" ", "_").replace("/", "_")
directory = input("Inserisci la directory dove vuoi salvare il file: ").strip()
if not os.path.exists(directory):
    print("Directory non valida.")
    return
file_path = os.path.join(directory, file_name)
with open(file_path, 'w', encoding='utf-8') as file:
    file.write(f"Titolo: {article['Titolo']}
")
    file.write(f"Fonte: {article['Fonte']}
")
    file.write(f"Link: {article['Link']}
")
    file.write(f"Contenuto:
{article['Contenuto']}
")
print(f"Risultato salvato in: {file_path}")
```

7. Risultati

- Una soluzione robusta per gestire il scraping, l'analisi e la gestione di articoli legali.
- Possibilità di espandere il progetto aggiungendo nuove funzionalità, come l'integrazione di dashboard o reportistica.

Prossimi Passi

1. **Espansione del dataset**: Integrare altri siti rilevanti in materia di legge.
2. **Metodologia di indicizzazione vettoriale**: Implementare un sistema di indicizzazione vettoriale per utilizzare Retrieval-Augmented Generation (RAG) in modo più efficace, consentendo ricerche più "profonde" e semantiche nei contenuti, senza limitarsi al match esatto tra richiesta e parole chiave.
3. **Ottimizzazione dell'analisi**: Integrare ulteriori modelli NLP per classificare gli articoli.

Uso di GPT-4 per Prototipazione Rapida

Durante lo sviluppo, è stato utilizzato il modello GPT-4 per la generazione e documentazione del codice. Questo approccio ha permesso di prototipare rapidamente soluzioni, verificando iterativamente il funzionamento e apportando correzioni basate su test concreti.