Project Number: 63

Project Title: SAT Decisions Scraper & Search Application with RAG (Retrieval-Augmented Generation)

Project Clients: Josh Tedjasaputra (Customer Experience Insight Pty Ltd)

Project specializations: Software Development;Web Application Development;Artificial Intelligence (Machine/Deep Learning, NLP);Human Computer Interaction (HCI);Internet of Things (IoTs);Cloud Computing;Big data Analytics and Visualization;Computer Science and Algorithms;Mobile Application Development;

Number of groups: 4 groups

Main contact: Josh Tedjasaputra (Customer Experience Insight Pty Ltd)

Background:

The project aims to develop an agentic Python application that automatically scrapes, processes, and indexes SAT (State Administrative Tribunal) decisions. The application should allow natural language queries using Retrieval-Augmented Generation (RAG) via AI Language Model to provide contextually relevant results, summaries, and insights. Additionally, the system must allow authorised users to manually upload new SAT decision documents, which will be processed, embedded, and immediately searchable. The primary goal is to enable efficient search and analysis of tribunal decisions using AI-driven insights (democratisation of law in WA)

Requirements and Scope:

- Develop a web-based search and retrieval system for SAT decisions.

- Implement an automated web scraper to extract tribunal decisions, metadata, and summaries.

- Integrate Retrieval-Augmented Generation (RAG) using AI Model to generate contextualised summaries of decisions.

- Provide a secure user interface for natural language queries and decision retrieval.

- Enable manual document addition with real-time indexing and searchability.

- Implement error handling, logging, and robust documentation for maintainability.

Required Knowledge and skills:

Data Acquisition & Processing:

- Scrape and store SAT decisions with metadata (date, case title, involved parties, summary).

- Handle structured and unstructured data cleaning (removal of duplicates, normalisation).

- Store the processed decisions in a searchable vector database (e.g., FAISS, Pinecone).

Search & Retrieval Interface:

- Web-based UI (Flask/FastAPI) allowing natural language queries (e.g., “Show decisions about environmental regulations”).

- Query results should include decision links, metadata, and AI Model-generated summaries.

RAG & AI Integration:

- Use AI model embeddings for vector-based search and response generation.

- Implement context-aware retrieval of case details and similar decisions.

Manual Document Addition:

- Allow authenticated users to upload new decisions (PDF/DOCX).

- Process uploaded documents, generate embeddings, and make them searchable instantly.

System Scalability & Security:

- The system should run on a Linux-based environment with secured API endpoints.

- User authentication for manual document upload and administrative controls.

Expected outcomes/deliverables:

Functional Web Application that allows SAT decision search and retrieval.

Automated Scraper that collects SAT decisions and stores them in a structured format.

AI-Powered Search Engine using RAG with Llama embeddings.

Manual Document Addition Feature with real-time indexing.

Comprehensive Documentation (README, API instructions, user guide).

Error Handling & Logging Framework for system maintainability.