ChatGPT

LangChain SQL Chatbot – Project Documentation

Project Overview

This project enables natural language interaction with a SQL database using a Streamlit UI, LangChain, and the Groq LLM (LLaMA3-8b). Users can ask questions about data in a relational database (`student.db` or a MySQL DB), and the LLM translates queries to SQL behind the scenes, fetches results, and presents them conversationally.

Tech Stack

Component	Description
LangChain	Framework for building LLM-powered agents
Groq LLM	Ultra-fast inference model (LLaMA3) via API
Streamlit	Python library for web UI
SQLite / MySQL	Backend databases
SQLAlchemy	DB abstraction & connectivity
Pydantic	Internal validation via LangChain

Folder Structure

```
├── app.py          # Streamlit frontend + LLM agent + DB orchestration
├── sqlite.py       # Local script to initialize and populate SQLite DB
├── student.db      # SQLite database file
```

Logical Flow of the Application

1. User Launches Streamlit App

```
st.set_page_config(...)
st.title(...)
```

- Initializes browser UI

2. Database Selection via Sidebar

```
radio_opt = ['Use SQLite3...', 'Connect to your MySQL DB']
selected_opt = st.sidebar.radio(...)
```

- Lets users pick between SQLite and MySQL

3. Groq API Key Input

```
api_key = st.sidebar.text_input(...)
```

- Securely captures the Groq API key

4. Database Configuration Function

```
@st.cache_resource(ttl='2h')
def configure_db(...):
    ...
```

- Connects to SQLite or MySQL - Caches DB engine for performance

5. Initialize LangChain Agent

```
llm = ChatGroq(...)
toolkit = SQLDatabaseToolkit(...)
agent = create_sql_agent(...)
```

- Configures Groq LLM, SQL toolkit, and LangChain agent

6. Stateful Chat UI

```
if 'messages' not in st.session_state ...
```

- Maintains persistent conversation history

7. Running Queries and Handling Response

```
user_query = st.chat_input(...)
response = agent.run(user_query, ...)
```

- Translates natural language to SQL - Executes and displays results

SQLite DB Initialization (sqlite.py)

```
CREATE TABLE IF NOT EXISTS Student (...)
```

- Creates Student table - Populates sample records

Sample User Queries

- "Show all students in class A"
- "What is the average marks in Data Science?"
- "Who scored the highest in DEVOPS?"

Environment and Deployment Tips

.env usage

```
GROQ_API_KEY=your_key_here
```

Run

```
streamlit run app.py
```

Deployment

- Streamlit Cloud
- Docker
- Localhost

Future Enhancements

- Schema auto-visualization
- Support for .csv upload
- Export chat history

Summary

This project merges the power of LLMs with SQL databases using a no-code UI, enabling intelligent, natural-language data interaction.