Spatial Mind

Course: Spatial Data Analysis

A system designed to automatically generate precise spatial SQL queries based on the user's question and execute them within QGIS.

Members: Amjad Awad, Abdelrahman Jbr

Instructor: Eehab Hijazi

Abstraction

Spatial Mind is an AI-powered application that converts natural language into spatial SQL queries that can be executed in QGIS. It enables students, teachers, spatial scientists, and researchers to perform spatial analysis quickly and easily.

Introduction

Spatial analysis is a core component of modern geographic information systems, yet writing spatial SQL queries can be challenging for many users especially those without advanced technical backgrounds. Spatial Mind bridges this gap by transforming natural language questions into accurate spatial SQL queries that can be executed directly in QGIS.

By combining AI-driven language understanding with spatial database expertise, Spatial Mind makes complex analysis accessible to students, educators, researchers, and GIS professionals. Whether you want to filter features, calculate distances, analyze intersections, or perform advanced geospatial operations, the system simplifies the process and accelerates your workflow.

Methodology

The Spatial Mind system's methodology describes how natural-language spatial requests are transformed into executable spatial SQL queries in QGIS. As illustrated in *Figure 5* *,* The system integrates a QGIS plugin, a backend server, Large Language Models (LLMs), and a spatial database. The workflow is divided into three main phases: **Session Initialization**, **Query Processing**, and **Query Execution**.

Session Initialization

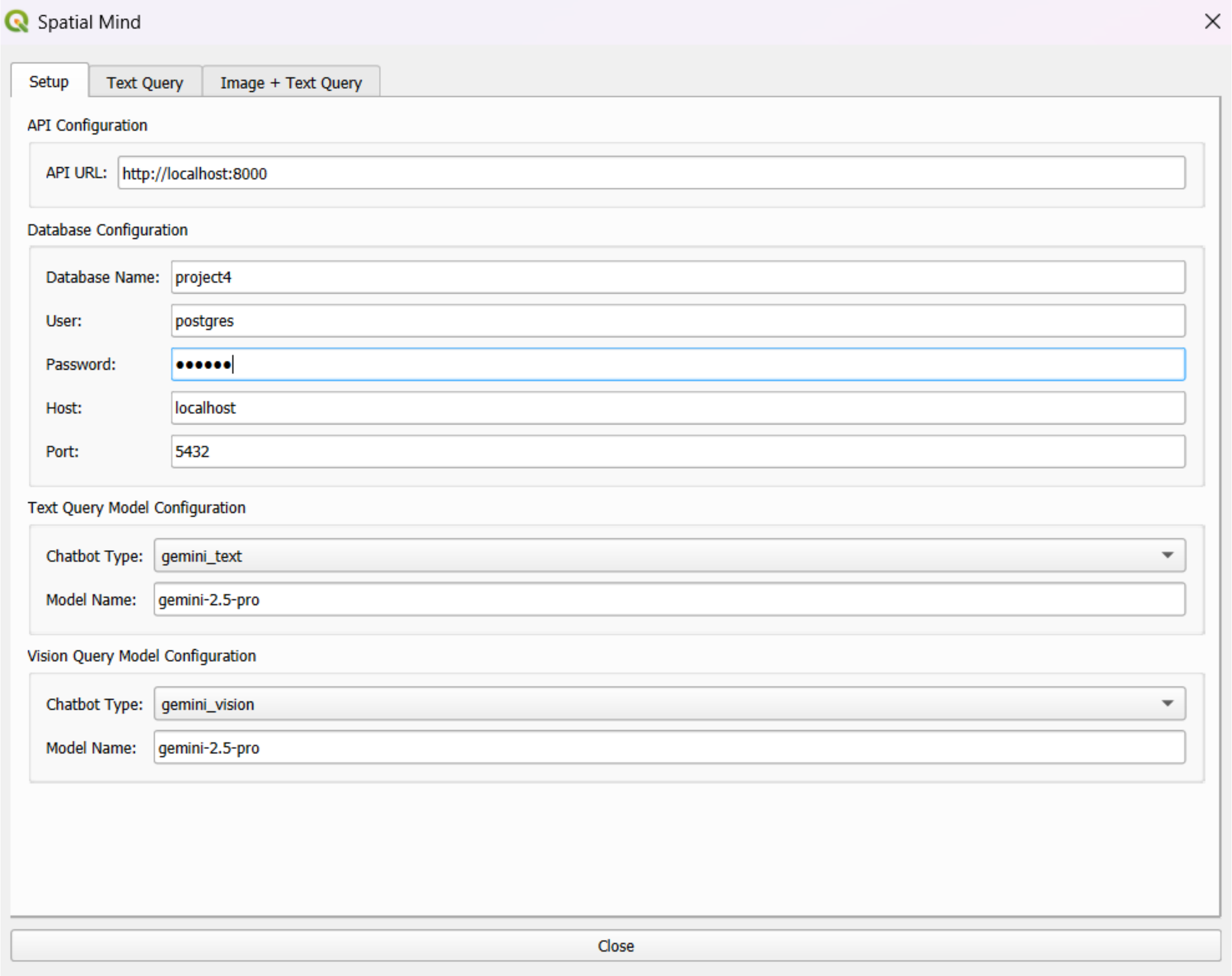
When the user launches the Spatial Mind plugin in QGIS, the system begins by establishing a connection with the spatial database (e.g., PostgreSQL/PostGIS), as illustrated in *Figure 1*. During this initialization phase, the user configures the interaction mode:

1. **Text-only mode:**
    - The user can choose between **Ollama** or **Gemini**.
    - **Ollama** returns a **short, brief schema**.
    - **Gemini (text)** returns a **full, detailed schema**.

2. **Text + Image mode:**

    Only **Gemini** is available, as it supports multimodal inputs.

This step ensures that the LLM selection and schema format align with user preferences and available model capabilities.



API Configuration

API URL: http://localhost:8000

Database Configuration

Database Name: project4

User: postgres

Password: ••••••

Host: localhost

Port: 5432

Text Query Model Configuration

Chatbot Type: gemini_text

Model Name: gemini-2.5-pro

Vision Query Model Configuration

Chatbot Type: gemini_vision

Model Name: gemini-2.5-pro

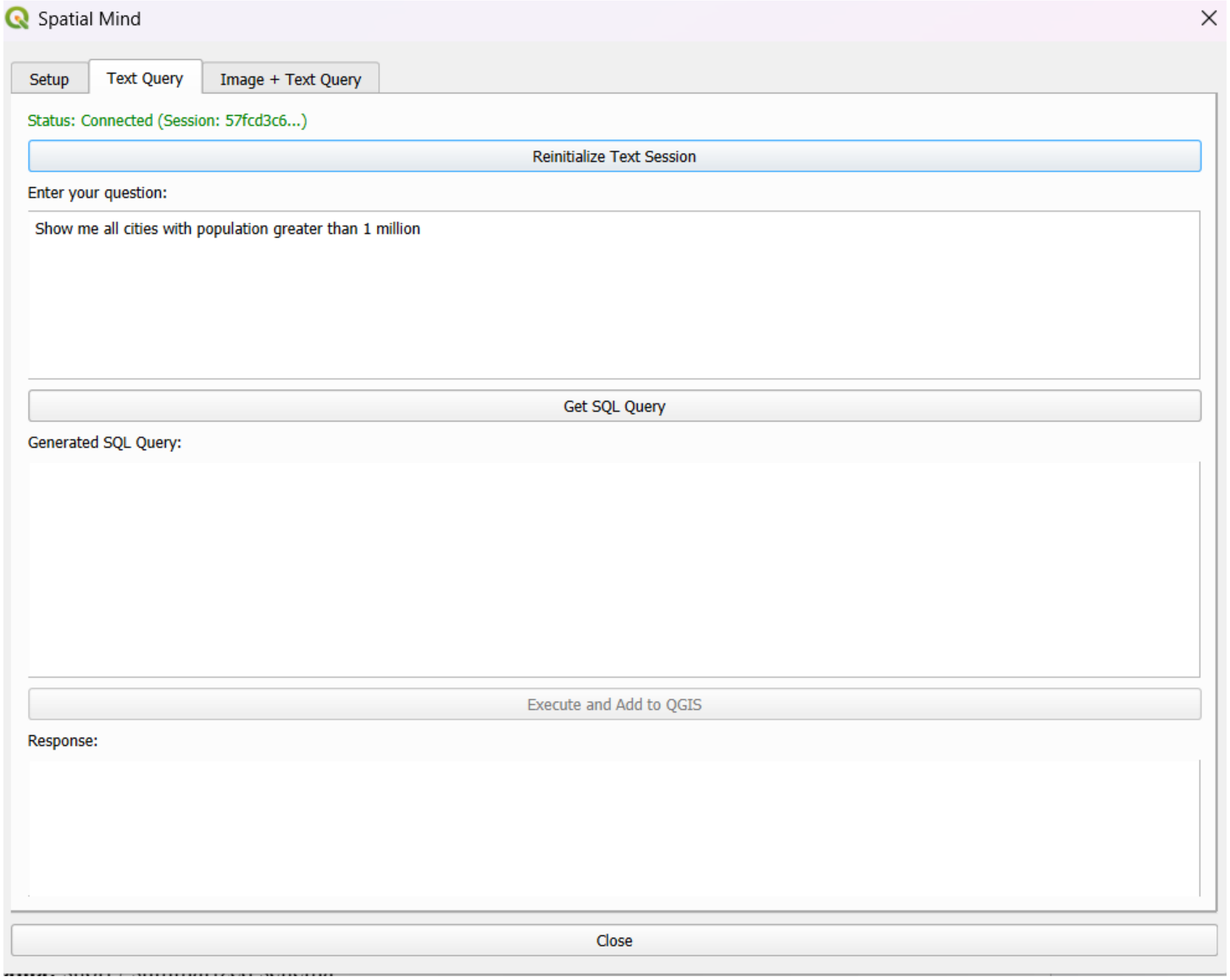
Figure 1: Session Initialization

Query Processing Pipeline

After the session is set up, the user interacts with the system by sending natural-language spatial queries through the QGIS plugin.

2.1 User Prompt Submission

The user enters a prompt describing the spatial analysis task they want to perform (e.g., *"Find all hospitals within 3 km of schools"*), as illustrated in *Figure 2* .
The QGIS plugin forwards this prompt to the backend server.



Status: Connected (Session: 57fcd3c6...)

Enter your question:

Show me all cities with population greater than 1 million

Generated SQL Query:

Response:

Figure 2: User Prompt Submission

2.2 Schema Retrieval

At the same time, the backend retrieves the **database schema** from the connected spatial database.
The schema format depends on the chosen model:

1. **Ollama:** short / summarized schema
2. **Gemini:** detailed / descriptive schema

This schema is essential for grounding the LLM's reasoning and ensuring SQL accuracy.

2.3 Contextual Prompt Engineering

To improve the LLM's output, the backend enriches the input by combining:

- the latest user prompt
- **chat history** (previous conversations in the session)
- the **database schema**

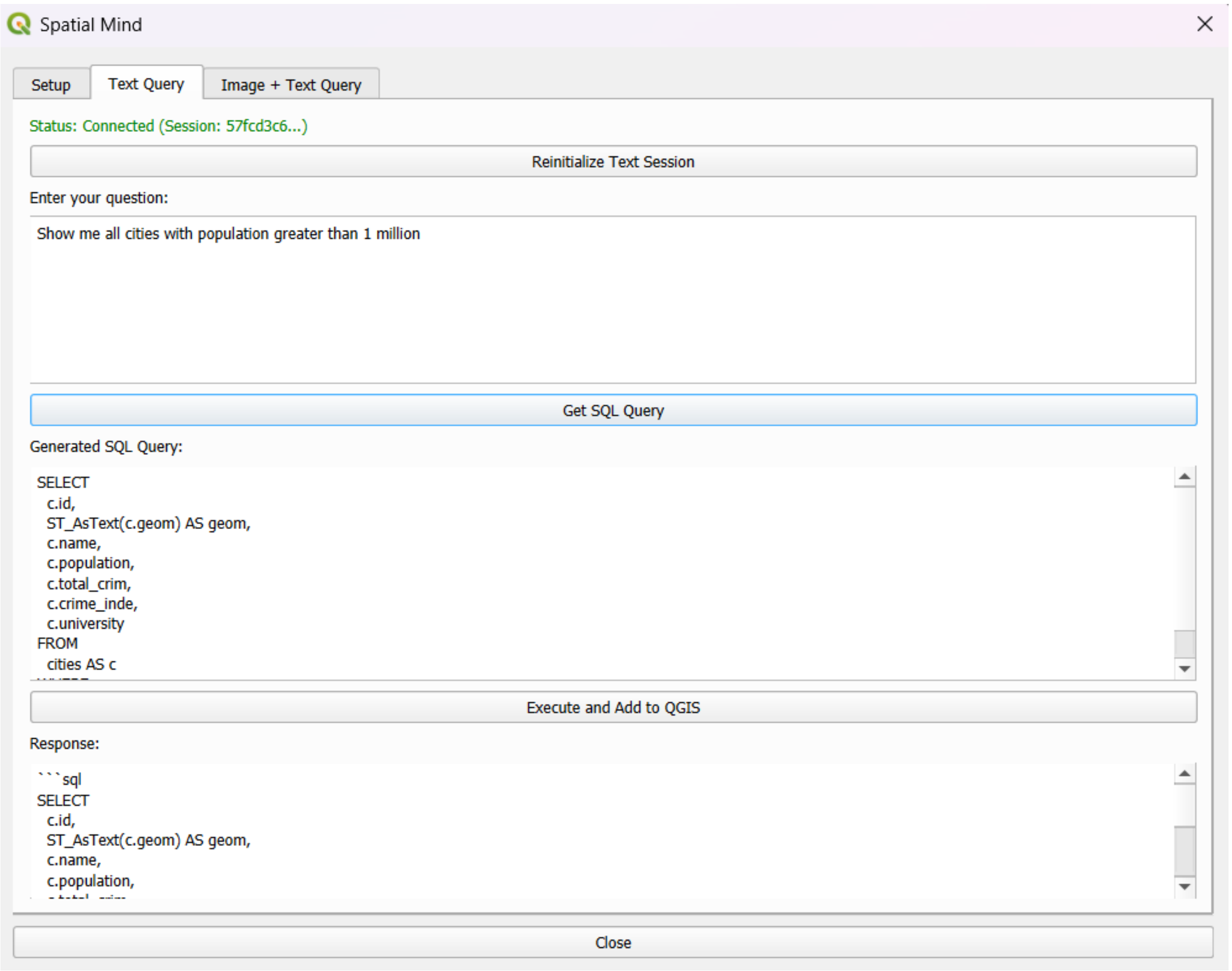
This ensures better context retention and higher-quality SQL generation.

2.4 LLM SQL Generation

The backend sends the contextualized prompt to the selected LLM.
The LLM processes the information and returns a valid **spatial SQL query** suitable for QGIS/PostGIS.

2.5 Delivery to the User

The backend sends the generated SQL query back to the QGIS plugin, allowing the user to review, modify, or execute it, as illustrated in *Figure 3*.



Status: Connected (Session: 57fcd3c6...)

Enter your question:

Show me all cities with population greater than 1 million

Generated SQL Query:

```
SELECT
  c.id,
  ST_AsText(c.geom) AS geom,
  c.name,
  c.population,
  c.total_crim,
  c.crime_inde,
  c.university
FROM
  cities AS c
```

Response:

~~~
```sql
SELECT
  c.id,
  ST_AsText(c.geom) AS geom,
  c.name,
  c.population,
~~~

Figure 3: How query is being delivered to the User

Query Execution and User Interaction

Once the user receives the generated SQL query, they are given two options:

Option A: Submit Another Query

The user may refine their question or request a new analysis.
The system loops back to **Step 2**, maintaining chat history for improved contextual reasoning.

Option B: Execute the SQL Query

The plugin sends the SQL query to the spatial database for execution.
The result is loaded back into QGIS and displayed as a **vector layer**, allowing the user to visualize or analyze the output immediately, as illustrated in *Figure 4*.

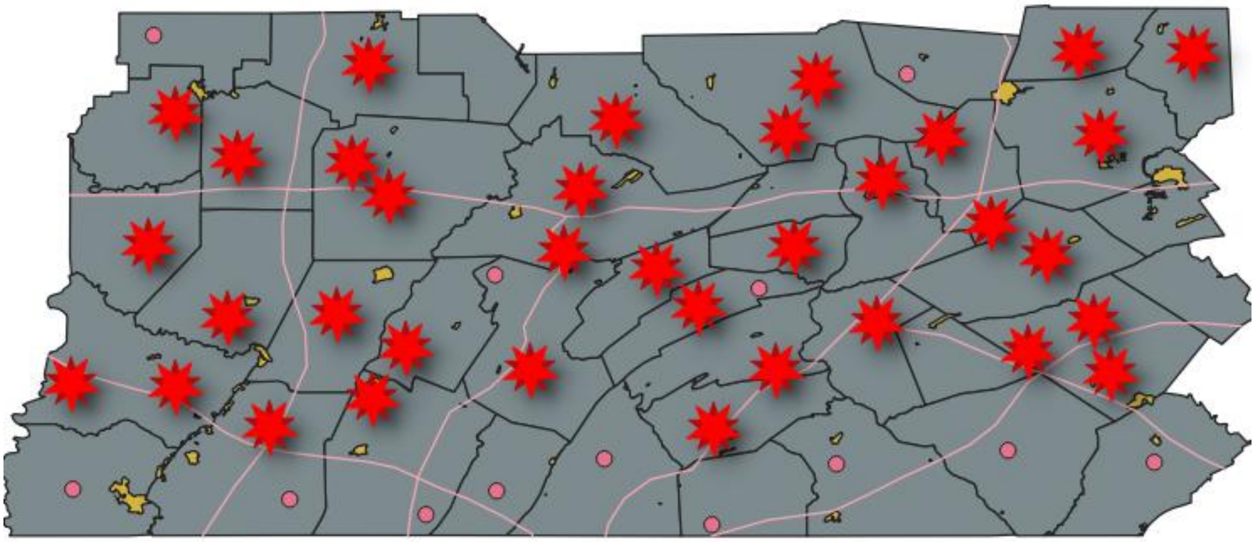


Figure 4: The result in QGIS

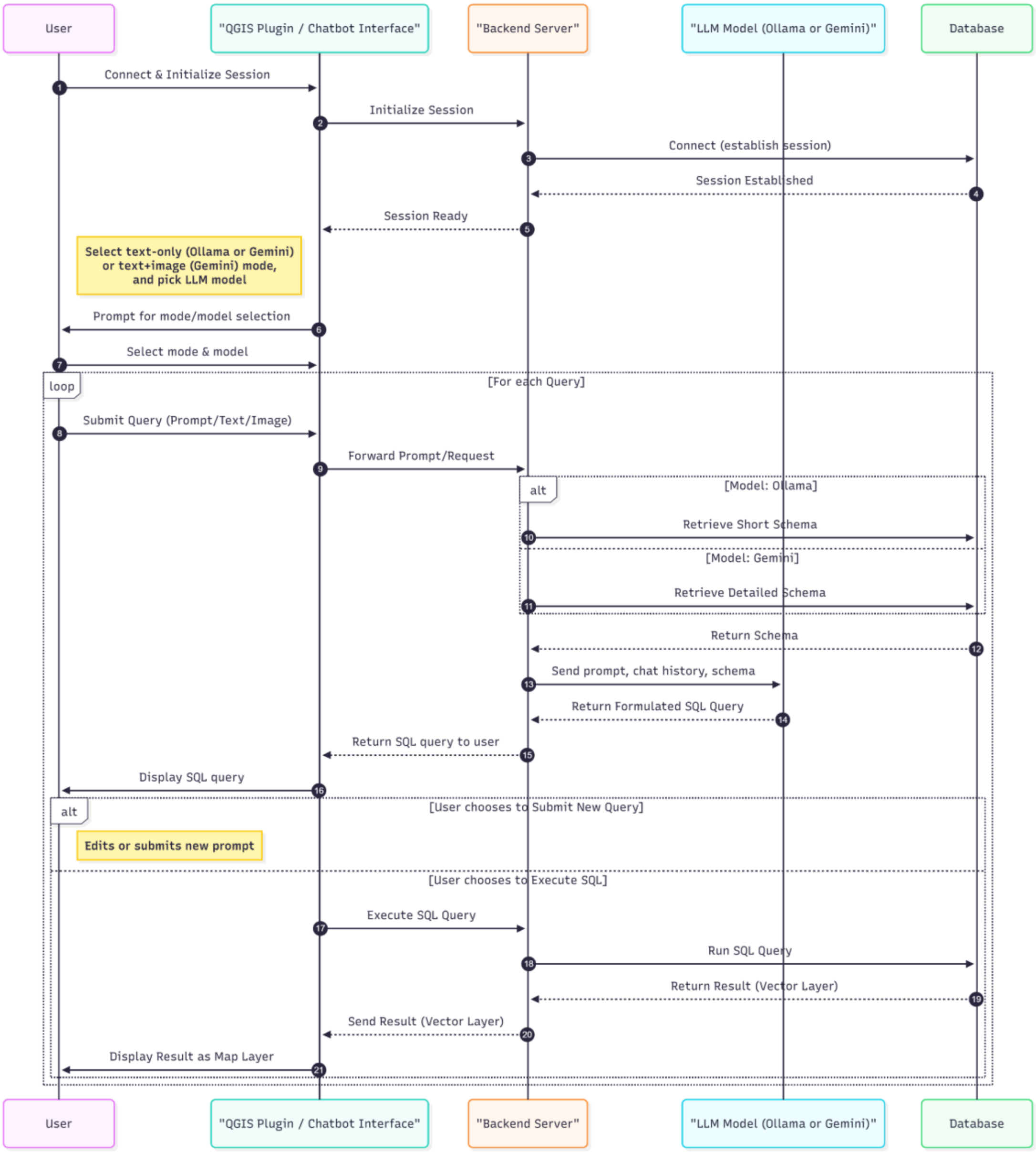

User
"QGIS Plugin / Chatbot Interface"
"Backend Server"
"LLM Model (Ollama or Gemini)"
Database
1 Connect & Initialize Session
2 Initialize Session
3 Connect (establish session)
4 Session Established
5 Session Ready
Select text-only (Ollama or Gemini) or text+image (Gemini) mode, and pick LLM model
6 Prompt for mode/model selection
7 Select mode & model
loop
[For each Query]
8 Submit Query (Prompt/Text/Image)
9 Forward Prompt/Request
alt
[Model: Ollama]
10 Retrieve Short Schema
[Model: Gemini]
11 Retrieve Detailed Schema
12 Return Schema
13 Send prompt, chat history, schema
14 Return Formulated SQL Query
15 Return SQL query to user
16 Display SQL query
alt
[User chooses to Submit New Query]
Edits or submits new prompt
[User chooses to Execute SQL]
17 Execute SQL Query
18 Run SQL Query
19 Return Result (Vector Layer)
20 Send Result (Vector Layer)
21 Display Result as Map Layer
User
"QGIS Plugin / Chatbot Interface"
"Backend Server"
"LLM Model (Ollama or Gemini)"
Database

Figure 5: Sequence diagram representing the workflow of the SpatialMind system.

Conclusion

Spatial Mind demonstrates how AI can simplify complex spatial analysis tasks. By combining natural language processing, schema interpretation, DE-9IM spatial reasoning, workflow image interpretation, and conversational context, the system bridges the gap between GIS technology and non-expert users.

This approach empowers users to explore spatial data efficiently, enhances learning and research workflows, and provides a foundation for future improvements such as multi-database support, advanced geospatial analytics, and integration with other GIS platforms.