IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro

Project Documentation format

1. Introduction

· **Project Title:** IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro

· **Team Members**: Shaik Mohammad Ashfaq **(Project Manager & AI Specialist)**
Shaik Ziaur Rahaman **(Backend Developer & Data Engineer)**
Srinivasula Samyuktha **(Frontend Developer & UI/UX Designer)**
Tammisetty Lakshmi Prasanna Kumar **(QA Engineer)**

2. Project Overview

· **Purpose:** To provide a seamless digital interface for non-technical users to interact with databases using natural language.

· **Features:**

- **NLP Interface**: Converts plain English into SQL commands via Gemini 1.5 Flash.
- **Secure Access**: Utilizes environment variables for API key masking.
- **Live Data Preview**: Displays database records in interactive tables.
- **Error Handling**: Implements Regex to clean AI responses for safe SQL execution.

3. Architecture

- **Frontend:**
  • Built using **Streamlit**, which allows for rapid deployment of a data-driven web interface.
  • The architecture is component-based, using a sidebar for multi-page navigation (Home, About, Query).

- **Backend:**
  • Built with **Python** and the **Google Generative AI SDK** to handle LLM processing.
  • Includes a logic layer that performs **Regex-based sanitization** to isolate raw SQL from AI responses.

- **Database:**
  • Uses **SQLite** for lightweight, relational data storage (replacing MongoDB).

- The schema consists of a `STUDENTS` table with columns for **NAME**, **CLASS**, **SECTION**, **MARKS**, and **COMPANY**.

4. Setup Instructions

- **Prerequisites:**
  - Python 3.9+
  - Google Gemini API Key
  - Pip (Python package manager)

- **Installation:**
  1. **Clone the repository**: git clone [GitHub-URL]
  2. **Install dependencies**: pip install streamlit google-generativeai python-dotenv
  3. **Environment Variables**: Create a .env file in the root directory and add GOOGLE_API_KEY="your_api_key_here".

5. Folder Structure

- **Client:**
- **`app.py`:** Main entry point containing the Streamlit UI and AI integration logic.
- **`readme.md`**: Local documentation providing instructions on how to interact with the user interface.
- **`requirements.txt`**: List of dependencies required for the client-side libraries, such as `streamlit` and `google-generativeai`.

- **Server:**
- **`.env`**: The server configuration file that securely stores the `GOOGLE_API_KEY`.
- **`sql.py`**: The backend database engineering script used to initialize the server-side database and seed it with initial data.
- **`data.db`**: The SQLite database file where the server stores and retrieves all relational records.
- **`.gitignore`**: Server-side configuration to ensure sensitive files like `.env` are not exposed to version control.
- **pyvenv.cfg**: Server configuration file that manages the Python environment version and paths.

6. Running the Application

Step 1: Database Setup:

Bash

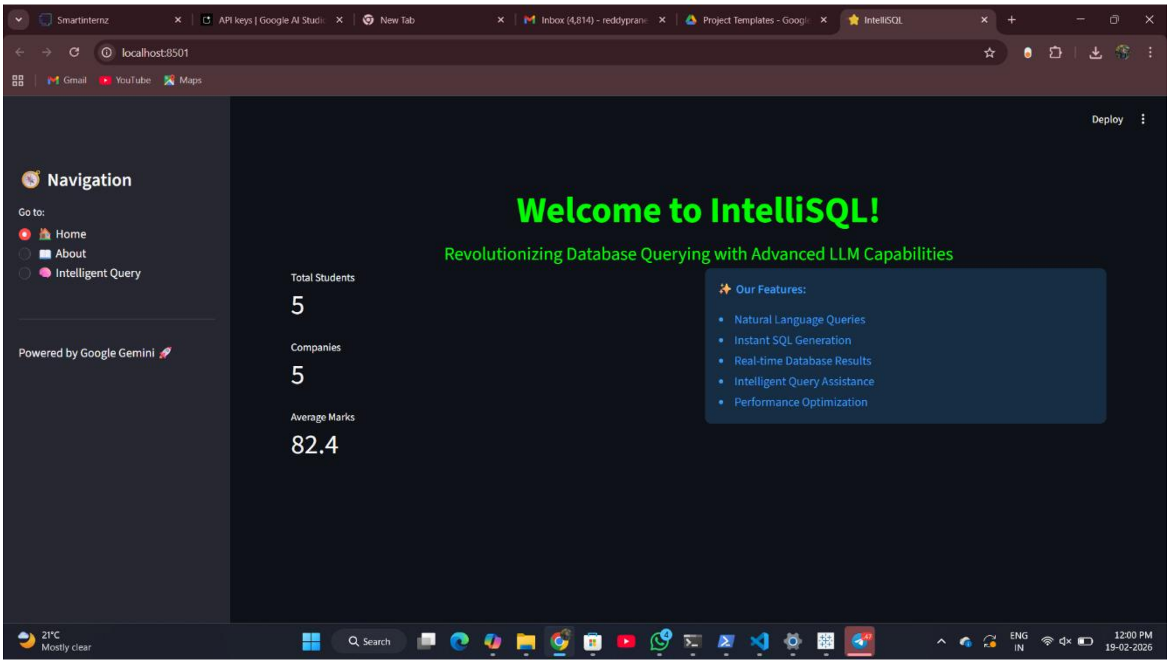
```
python sql.py
```

``` [cite: 148]

**Step 2: Start Application:**

**Bash**

```
streamlit run app.py
~~~

``` [cite: 146]

**App URL:** The dashboard is accessible at **http://localhost:8501**

## 7. API Documentation

**SQL Generation**:

- **Method**: Internal POST to Gemini 1.5 Flash.
- **Parameters**: prompt_context, user_question.
- **Response**: A SQL string (e.g., SELECT * FROM STUDENTS;).

**Database Execution**:

- **Function**: read_query(sql, db).
- **Parameters**: Cleaned SQL string and database path.

## 8. Authentication

- **API Authentication**: Handled via **API Key-based authentication** using the Google Generative AI SDK.
- **Security**: Keys are stored in an encrypted/masked .env file and loaded into the environment at runtime using python-dotenv.

## 9. User Interface

- **Registration/Login**: Simplified via API key validation upon app startup.
- **Query Interface**: A professional text-input area where users type natural language questions.
- **Data Visualization**: Real-time rendering of results in interactive data tables.

## 10. Testing

- **Strategy**: Functional testing was performed on diverse natural language phrasings to ensure SQL accuracy.
~~~

- **Tools**: **Manual UAT** and **Python Debugger** were used to verify that Regex correctly strips AI conversational text.

## 11. Screenshots of project

## GitHub Repository:



Navigation

Go to:

- ● 🏠 Home
- ○ 📖 About
- ○ Intelligent Query

Powered by Google Gemini 🚀

Welcome to IntelliSQL!

Revolutionizing Database Querying with Advanced LLM Capabilities

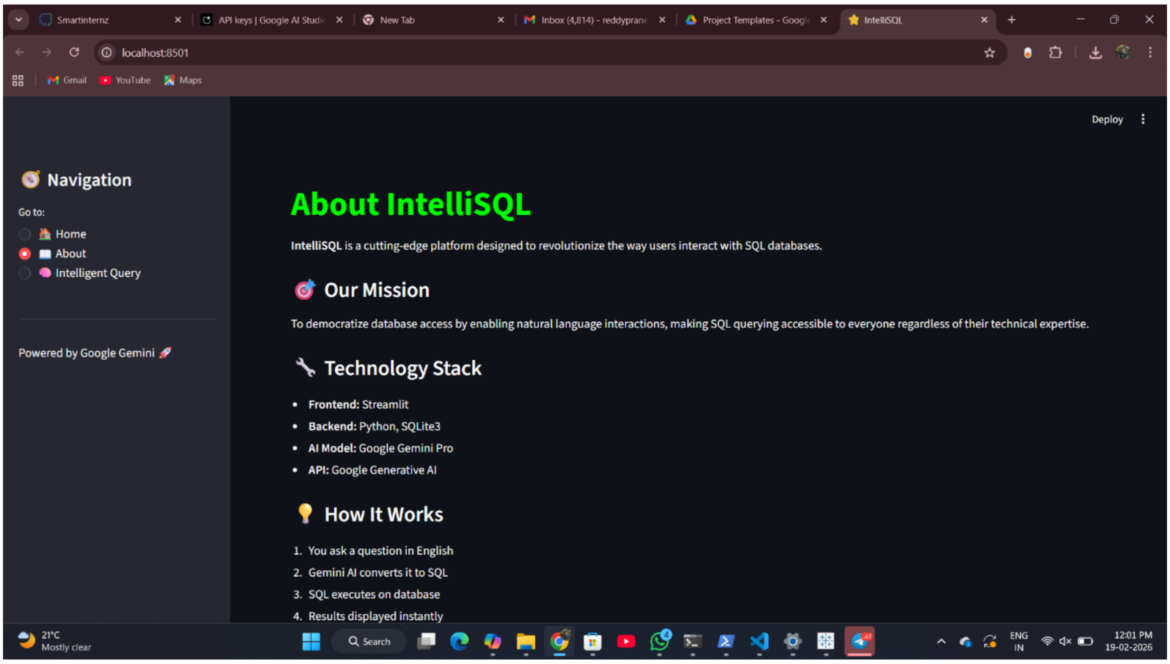
Total Students
5

Companies
5

Average Marks
82.4

✨ Our Features:

- Natural Language Queries
- Instant SQL Generation
- Real-time Database Results
- Intelligent Query Assistance
- Performance Optimization



Navigation

Go to:

- ○ 🏠 Home
- ● 📖 About
- ○ Intelligent Query

Powered by Google Gemini 🚀

About IntelliSQL

**IntelliSQL** is a cutting-edge platform designed to revolutionize the way users interact with SQL databases.

🎯 Our Mission

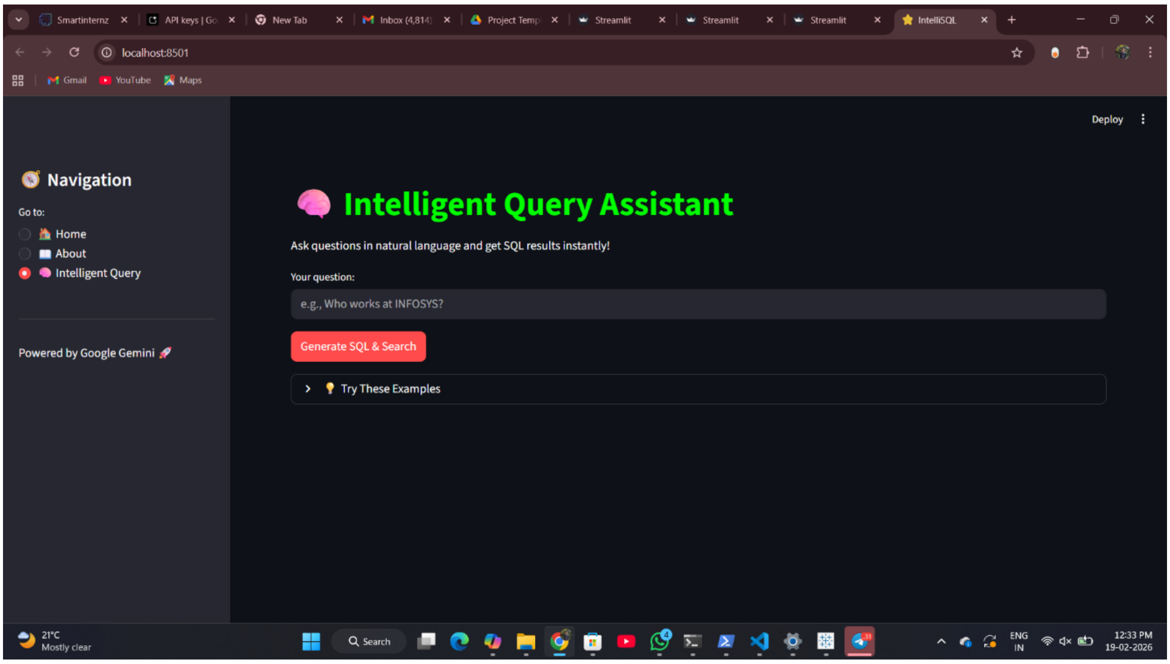
To democratize database access by enabling natural language interactions, making SQL querying accessible to everyone regardless of their technical expertise.

🔧 Technology Stack

- **Frontend:** Streamlit
- **Backend:** Python, SQLite3
- **AI Model:** Google Gemini Pro
- **API:** Google Generative AI

💡 How It Works

1. You ask a question in English
2. Gemini AI converts it to SQL
3. SQL executes on database
4. Results displayed instantly



## 🧭 Navigation

Go to:

○ 🏠 Home
○ 📖 About
● 🧠 Intelligent Query

Powered by Google Gemini 🚀

# 🧠 Intelligent Query Assistant

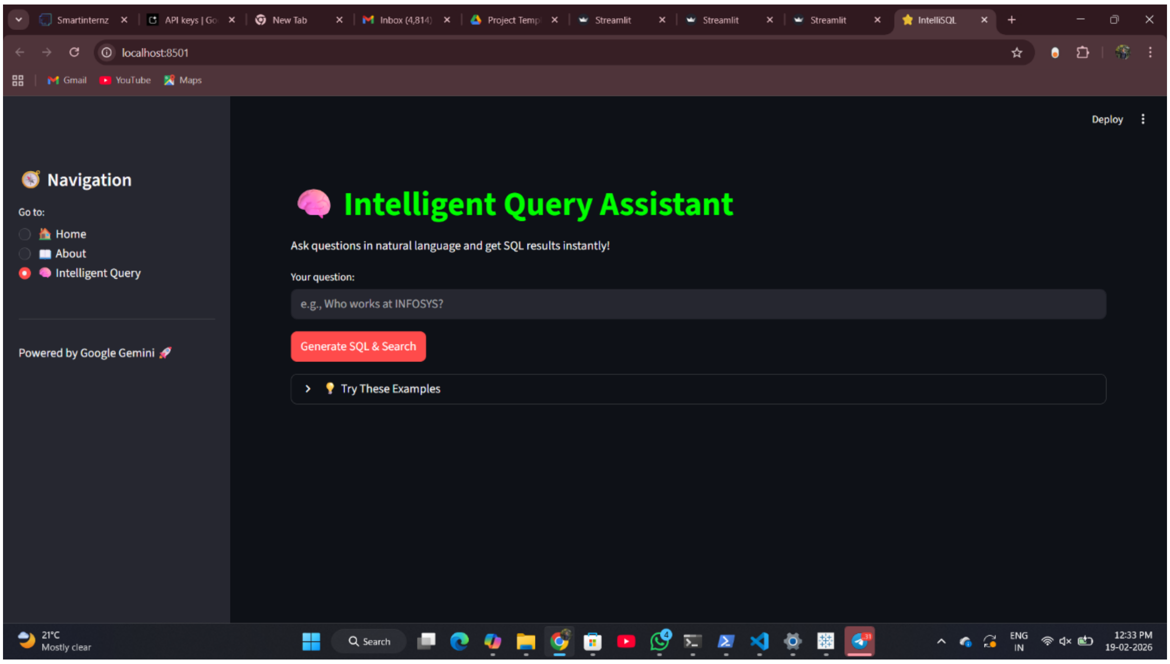
Ask questions in natural language and get SQL results instantly!

Your question:

e.g., Who works at INFOSYS?

Generate SQL & Search

› 💡 Try These Examples


Smartinternz
API keys | Go
New Tab
Inbox (4,814)
Project Temp
Streamlit
Streamlit
Streamlit
IntelliSQL
localhost:8501
Gmail
YouTube
Maps
Deploy

🧭 Navigation

Go to:

○ 🏠 Home
○ 📖 About
● 🧠 Intelligent Query

Powered by Google Gemini 🚀

🧠 Intelligent Query Assistant

Ask questions in natural language and get SQL results instantly!

Your question:

e.g., Who works at INFOSYS?

Generate SQL & Search

› 💡 Try These Examples

21°C
Mostly clear
Search
ENG
IN
12:33 PM
19-02-2026

- **Complex Joins**: The current model may struggle with multi-table queries if the schema isn't explicitly defined in the prompt.
- **Responsiveness**: Wide data tables may require horizontal scrolling on smaller mobile displays.

## 13. Future Enhancements

- **Voice-to-SQL**: Integration of a microphone icon to allow users to speak their queries.
- **AI Insights**: Using the AI to explain the data results in plain English below the table.
- **Multi-DB Support**: Adding connectors for PostgreSQL and MySQL.