

1. INTRODUCTION

1.1 Project Overview

IntelliSQL is an AI-powered application that enables users to interact with databases using natural language queries. The system leverages Google's Gemini Pro large language model to convert user questions written in plain English into accurate and optimized SQL queries. IntelliSQL simplifies database interaction by automatically generating SQL statements, executing them on the connected database, and presenting structured results through an intuitive user interface.

- **Scenario 1: Business Data Analysis**

A sales manager uses Intellisql to ask "Show total sales for last month by region." The system converts the requests into an SQL query and display summarized results.

- **Scenario 2: Student Database**

A college administrator asks, "List students with marks above 80 in Datastructures" "IntelliSQL generates the appropriate SQL command and retrieves accurate records

- **Scenario 3: Real-Time Report**

A company executive requests, "Give top 5 products with highest revenue" IntelliSQL processes the request, execute the query and presents output

1.2 Objectives

The primary purpose of the **IntelliSQL project** is to develop an intelligent Generative AI-based system that translates natural language into SQL queries using Gemini Pro. IntelliSQL project is to develop an intelligent system that bridges the gap between natural language and structured database queries using Gemini Pro. The detailed objectives are as follows The key objectives include:

Key Objectives:

- To design a natural language interface for database interaction
- To integrate Gemini Pro for accurate SQL query generation
- To ensure secure and efficient query execution
- To reduce manual SQL coding effort
- To ensure secure and efficient query execution
- To reduce manual SQL coding effort

2. PROJECT INITIALIZATION AND PLANNING PHASE

- 2.1 Defining Problem Statement
- 2.2 Project Proposal (Proposed Solution)
- 2.3 Initial Project Planning

3. DATA COLLECTION AND PREPROCESSING PHASE

- 3.1 Data Collection Plan and Raw Data Sources Identified
- 3.2 Data Quality Report
- 3.3 Data Preprocessing

4. MODEL DEVELOPMENT PHASE

- 4.1 Model Selection Report
- 4.2 Initial Model Training Code, Model Validation and Evaluation Report

5. MODEL OPTIMIZATION AND TUNING PHASE

- 5.1 Tuning Documentation
- 5.2 Final Model Selection Justification

6. RESULTS

6.1 Output Screenshots

The complete execution of the Intelligent SQL quer application is shown in the images step by step as shown below.

Step 1: Run the app.py code using **`streamlit run app.py`** and you will get a link in terminal <http://localhost:8501> which opens automatically in the browser.

```
PS C:\Users\pavan\Downloads\IntelliSQL-Intelligent-SQL-Querying-with-LLMs-Using-Gemini-Pro_pa1-main
\SQL-LLM> streamlit run app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://10.65.2.222:8501
```

Fig 6.1.1: Code running in Terminal

Step 2: The web application automatically launches in the default browser

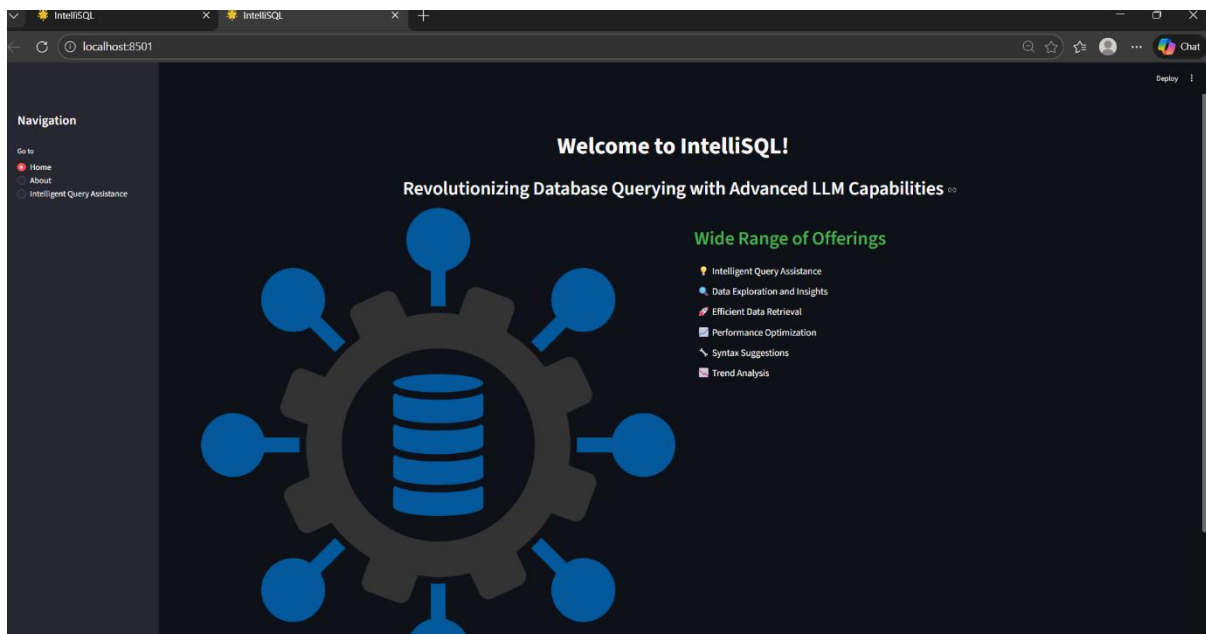


Fig 6.1.2: IntelliSQL Landing

INTELLISQL: INTELLIGENT SQL QUERYING WITH LLMS USING GEMINI PRO

Step 3: Click on **Browse Files** option to upload database file.

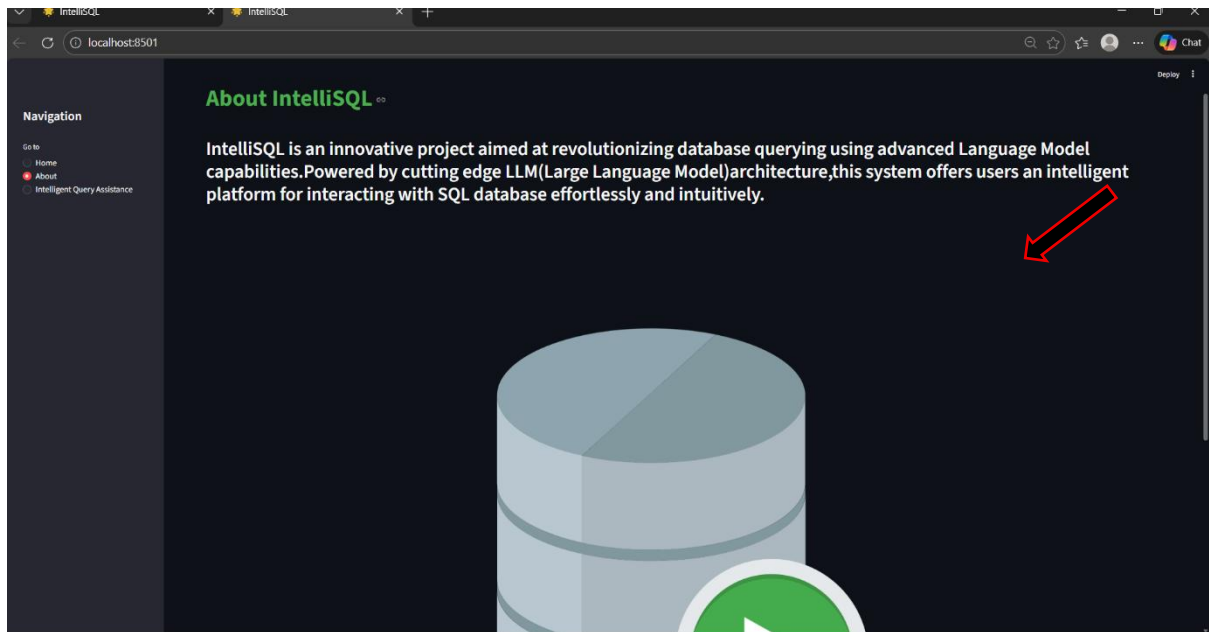


Fig 6.1.3: Database Page Intellisql project

Step 4: Click on choose file option to upload the database file

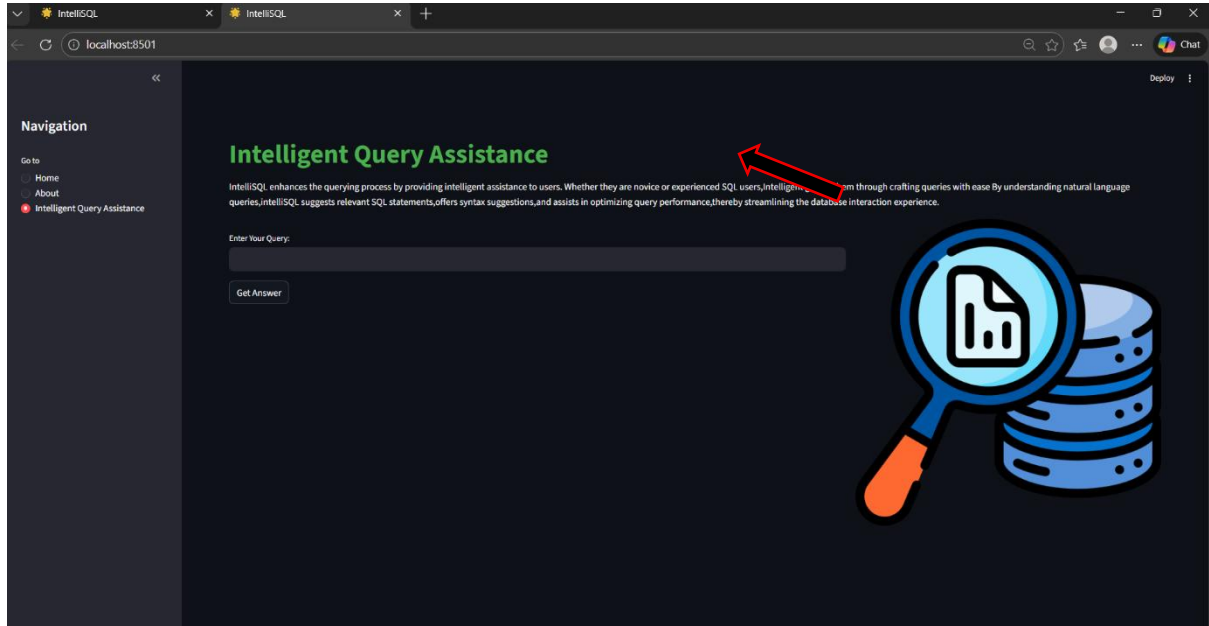


Fig 6.1.4: Window to upload Database file

INTELLISQL: INTELLIGENT SQL QUERYING WITH LLMS USING GEMINI PRO

Step 5: Database File Loads into the Application

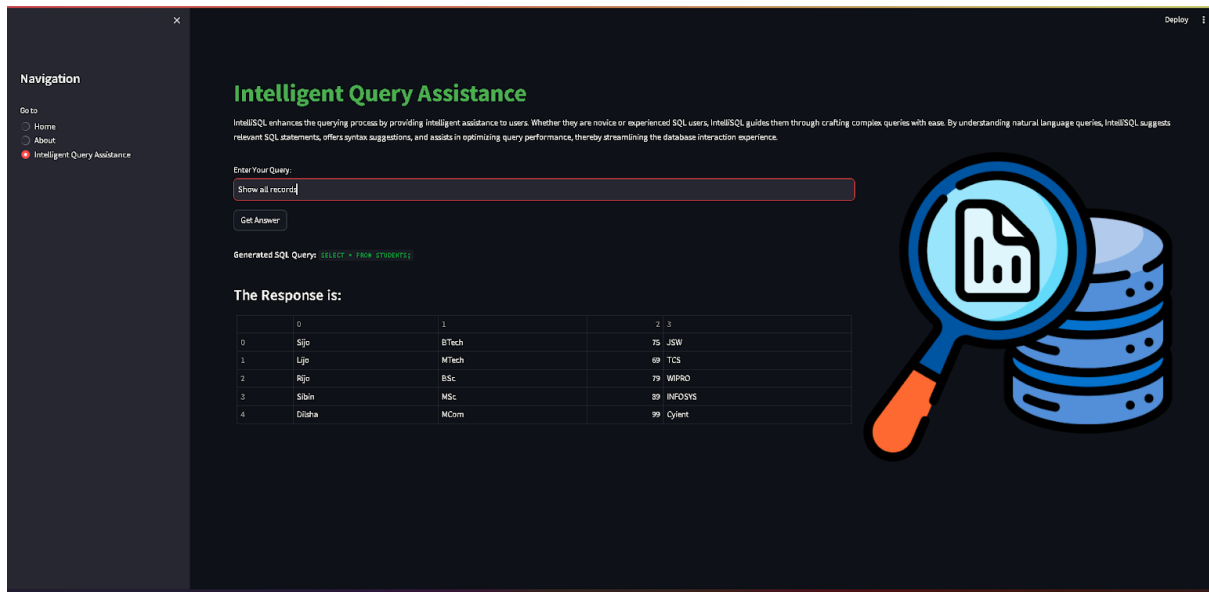


Fig 6.1.5: File loads in the Application

Step 6: Click on submit button to process the natural language query

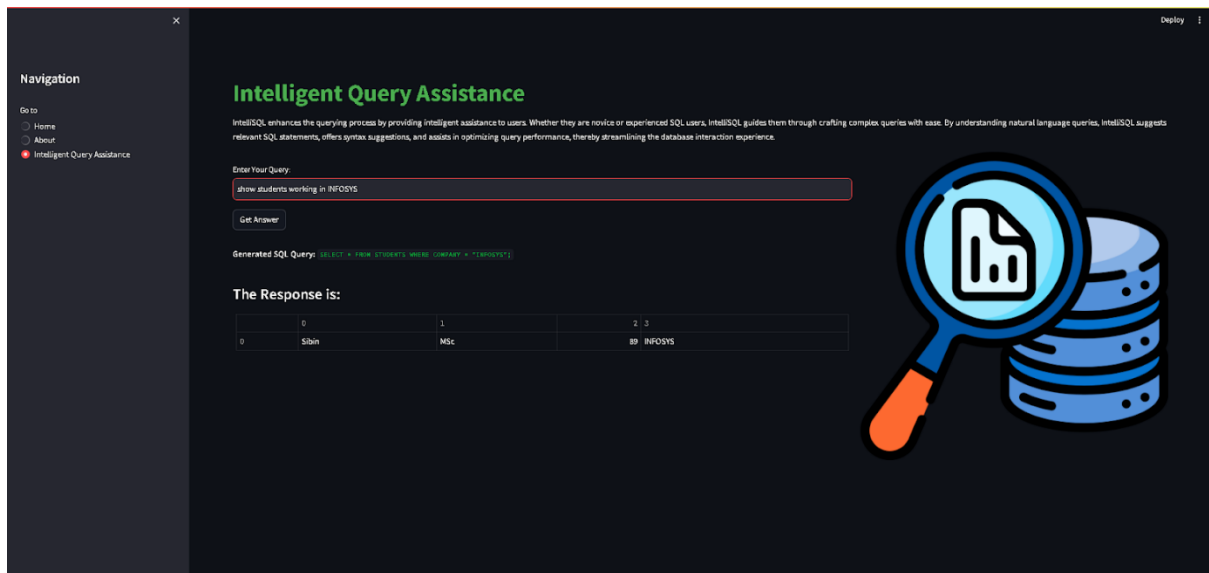


Fig 6.1.6: Click on submit to predict

INTELLISQL: INTELLIGENT SQL QUERYING WITH LLMS USING GEMINI PRO

Step 7: This step ensures that the user's simple English question is accurately transformed. System generates SQL and retrieves the query results.

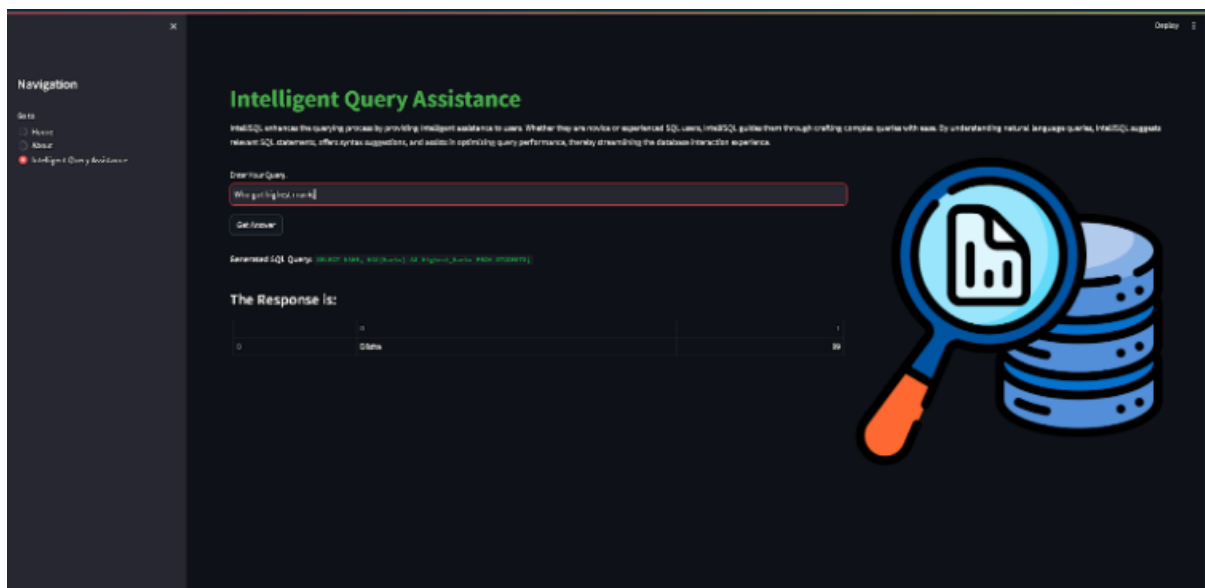
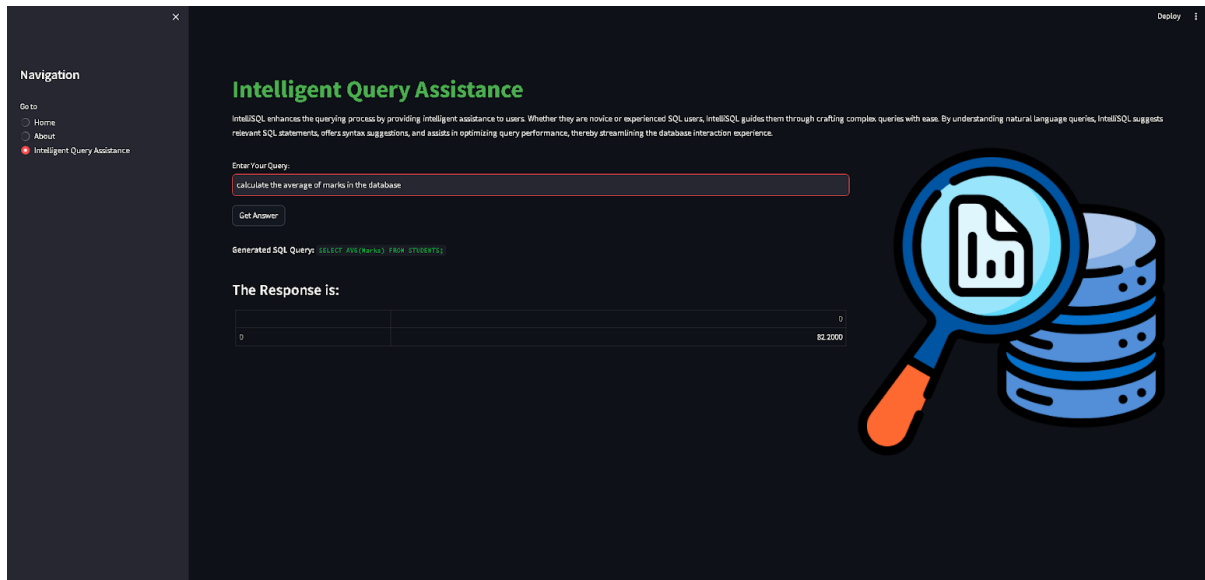


Fig 6.1.7: Final Output(average & high score)

7. ADVANTAGES & DISADVANTAGES

7.1 Intellisql: Intelligent SQL querying Project

Advantage	Description
Natural Language Querying	Allows users to interact with databases using simple English instead of writing complex SQL queries.
Automated SQL Generation	Gemini Pro automatically converts user questions into accurate and optimized SQL queries.
Fast & Real-Time Results	Provides instant query execution and quick retrieval of database results.
User Friendly interface	Simple web-based interface that enables non-technical users to query databases easily..
Error Reduction	Minimizes syntax errors and logical mistakes in manual SQL writing.
Scalability	Can be extended to support large databases and multiple database management systems.
Time Saving	Reduces the time required to write, debug, and execute complex SQL queries.

Table 7.1: Advantages

7.2 Disadvantages / Limitations

Disadvantage	Description
Accuracy Limitations	Complex or ambiguous natural language queries may sometimes generate incorrect SQL statements.
Internet Dependency	Requires an active internet connection to access the Gemini Pro API for query processing.
API Cost	Using Gemini Pro or other LLM APIs may involve usage costs depending on the number of requests.
Security Risks	Improper validation may expose the database to risks if unsafe queries are generated.
Limited Context Understanding	Allows users to interact with databases using simple English instead of writing complex SQL queries..

Table 7.2: Disadvantages

8. CONCLUSION

Intelligent SQL Querying with LLMs using Gemini Pro presents an efficient and intelligent solution for converting natural language queries into accurate SQL statements. By leveraging the capabilities of Gemini Pro, the system simplifies database interaction, reduces manual coding effort, and improves accessibility for both technical and non-technical users. The project demonstrates how Generative AI can enhance productivity, ensure efficient data retrieval, and transform the way users interact with database systems in modern applications.

9. FUTURE SCOPE

In the future, **IntelliSQL** can be enhanced to support multiple database systems and handle more complex queries such as multi-table joins and nested operations. The system can be improved with conversational memory for follow-up queries and integrated data visualization features for automatic chart generation. Additionally, deploying the solution as a cloud-based SaaS platform with advanced security and role-based access control can make it scalable and enterprise-ready.

10. APPENDIX

10.1 Source Code

The complete source code for the **IntelliSQL** project is implemented using Python, Streamlit, SQLite/MySQL database integration, and the Google Gemini Pro API. The application consists of modules for natural language input processing, prompt engineering, SQL query generation, secure query execution, and structured output display.

10.2 GitHub & Project Demo Link

The source code and project demonstration video are hosted on GitHub. The repository contains all required files, including the application code, documentation, and setup instructions.

- **GitHubRepository:**
https://github.com/venkatapavan07/Intelligent_SQL_Querying_Pavan.git
- **ProjectDemoVideo:**<https://drive.google.com/file/d/1Q2vQDSiW4oSckBg-dzW7AE-E5jkXzX8Q/view?usp=drivesdk>