Project Report

7. RESULTS

7.1 Output Screenshots

8. ADVANTAGES & DISADVANTAGES

9. CONCLUSION

10. FUTURE SCOPE

11. APPENDIX

Source Code(if any)

Dataset Link

GitHub & Project Demo Link

IntelliSQL: Intelligent SQL Querying with LLMs Using Gemini Pro

1. INTRODUCTION

1.1 Project Overview

IntelliSQL is an intelligent SQL querying system that uses Google Gemini Pro, a powerful Large Language Model (LLM), to convert natural language queries into SQL queries. This system allows users to interact with databases using simple English instead of writing complex SQL commands manually.

The application is built using Streamlit for the user interface, SQLite3 as the database, and Gemini Pro API for query conversion. IntelliSQL improves accessibility for beginners and enhances productivity for experienced users by simplifying database interactions.

1.2 Purpose

The purpose of IntelliSQL is to eliminate the complexity of writing SQL queries by allowing users to interact with databases using natural language.

The key purposes include:

- Helping beginners query databases without SQL knowledge
- Reducing errors in SQL query writing
- Improving productivity and efficiency
- Providing intelligent query suggestions
- Enabling faster data analysis

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Performance testing was conducted to evaluate the response time and stability of the IntelliSQL system. Multiple natural language queries were given as input, and the system successfully converted them into SQL queries using the Gemini Pro model. The generated SQL queries were executed on the SQLite database, and results were displayed correctly.

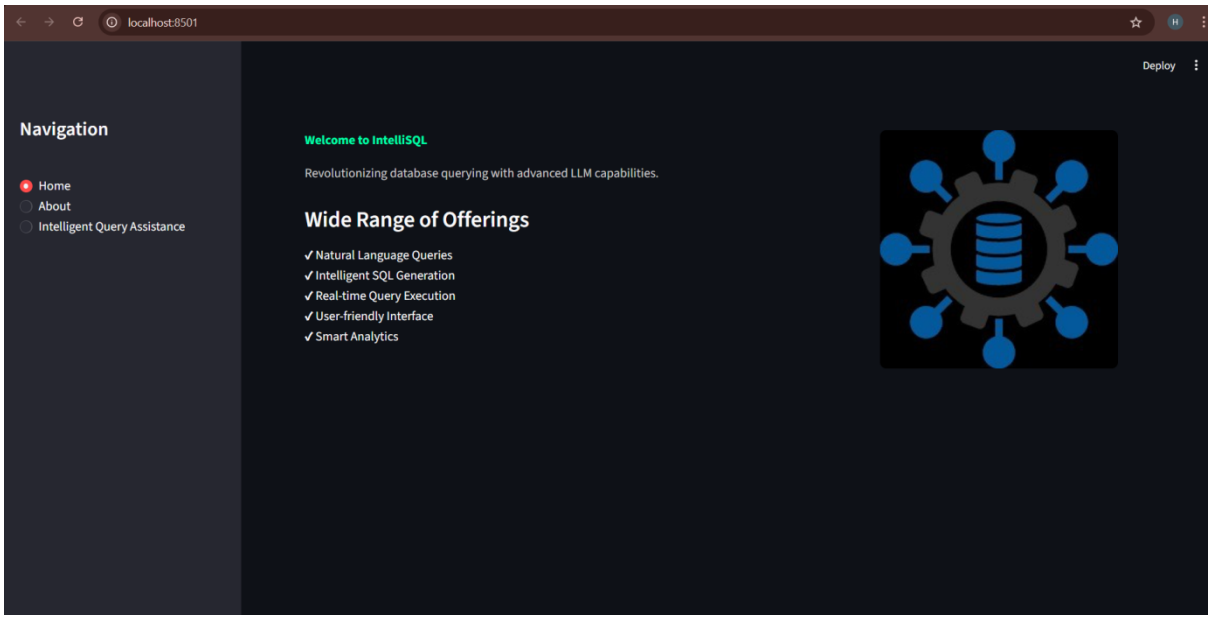
The testing results showed that the system provides fast query conversion and quick response time. The application remained stable during continuous usage and handled multiple queries without errors or crashes. This confirms that IntelliSQL meets the required performance standards and works efficiently.

7. RESULTS

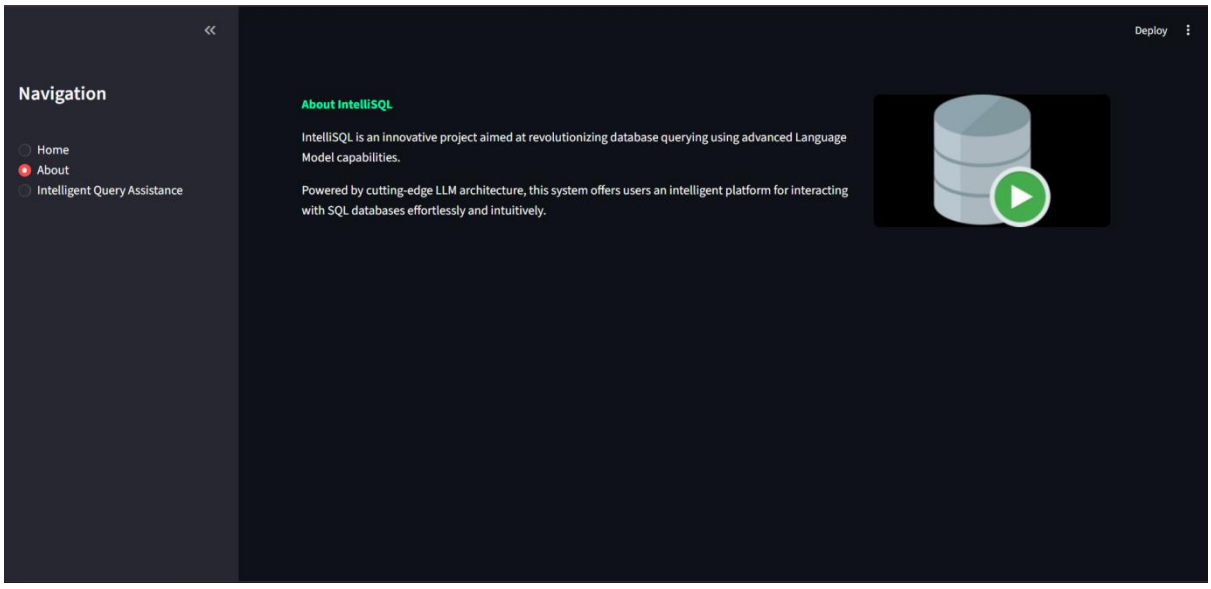
7.1 Output Screenshots

Results include:

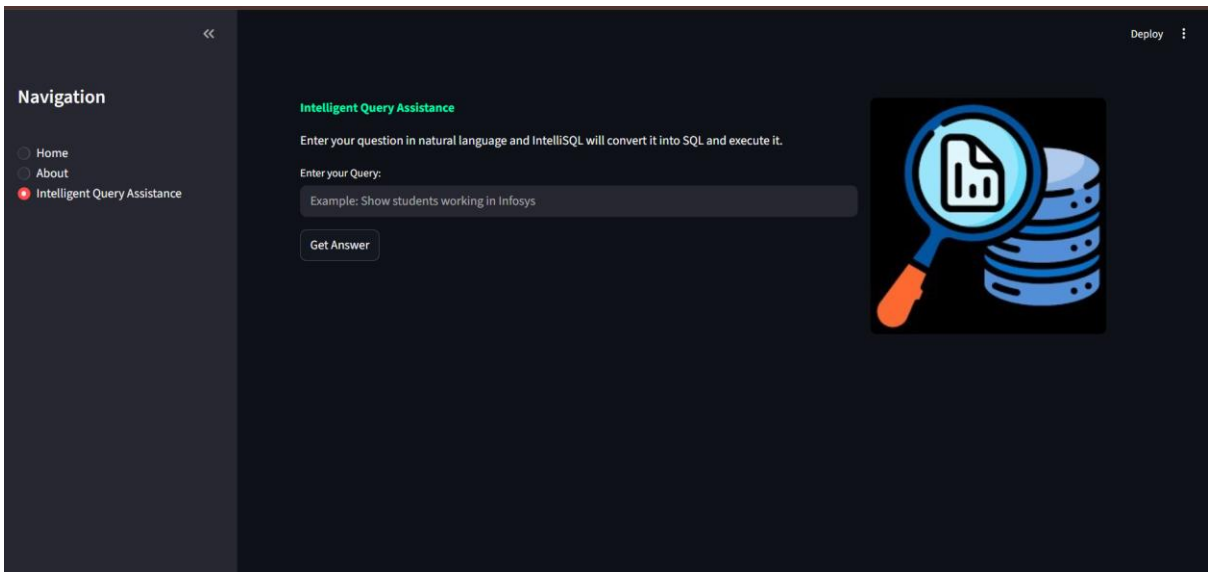
Home Page :



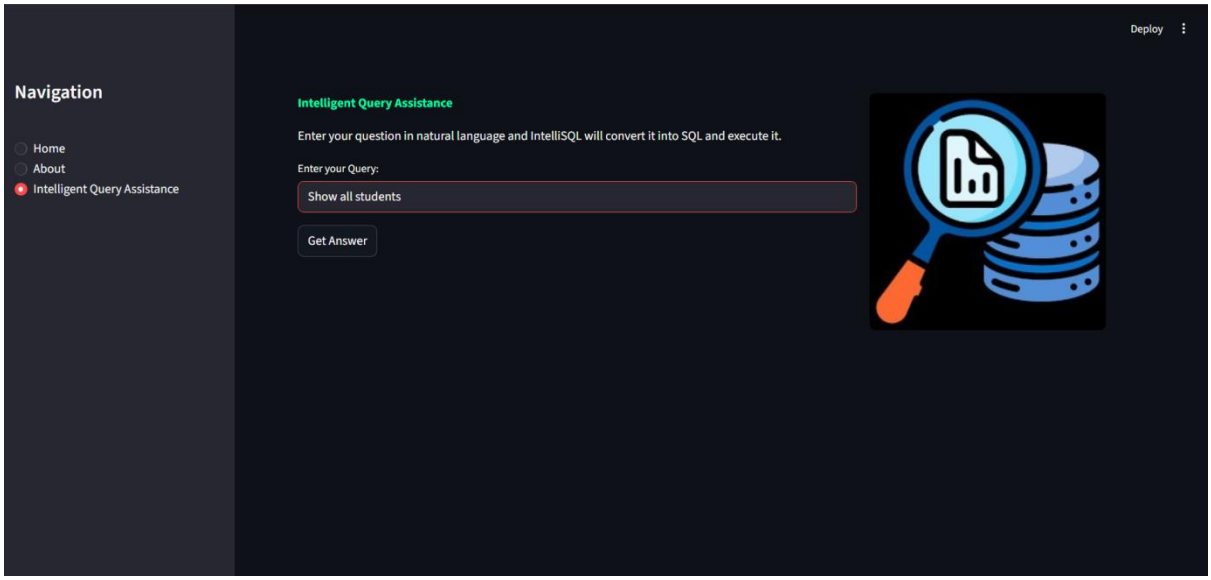
About Page :



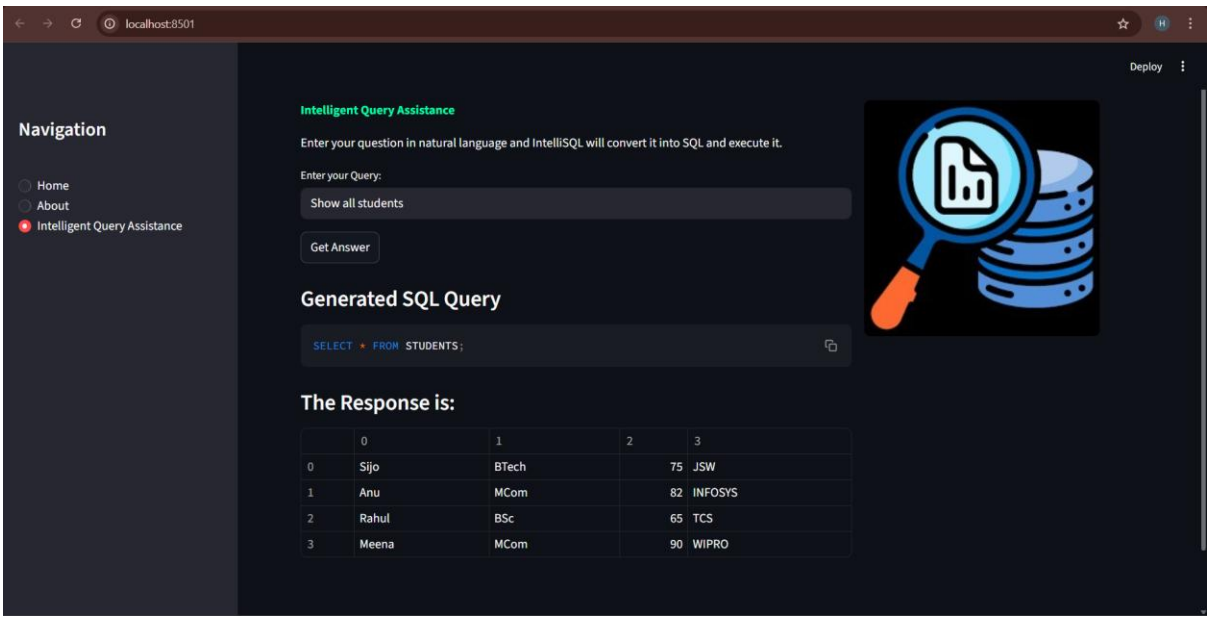
Query Input Page :



SQL Query Generation:



Query Results Display :



8. ADVANTAGES & DISADVANTAGES

Advantages :

1. Easy to Use
IntelliSQL is easy to use. Users can interact with the database using simple English.

2. No SQL Knowledge Required
Users do not need SQL knowledge. The system converts English into SQL automatically.

3. Fast Query Execution
The system generates and runs SQL queries quickly. This saves time for users.

Disadvantages :

1. Requires Internet Connection
The system needs internet access. Gemini Pro API works only online.

2. Depends on AI Accuracy
The system depends on AI responses. Sometimes results may not be fully accurate.

3. Limited to Database Structure
The system works based on the database schema. Changes in schema may affect results.

9. CONCLUSION

IntelliSQL is an intelligent database querying system that allows users to interact with databases using natural language. It uses the Gemini Pro Large Language Model to convert user questions into SQL queries automatically. This removes the need for manual SQL writing and makes database interaction easier for both beginners and experienced users. The system is built using Streamlit for the user interface and SQLite for database management, providing a simple and efficient platform.

The project successfully demonstrates how Artificial Intelligence can simplify complex database operations. IntelliSQL improves productivity, reduces errors, and saves time by generating accurate SQL queries. It provides a user-friendly and efficient solution for data retrieval and analysis. Overall, IntelliSQL shows the potential of AI-powered tools in making database systems more accessible and intelligent.

10. FUTURE SCOPE

In the future, IntelliSQL can be enhanced to support multiple databases such as MySQL, PostgreSQL, and Oracle. This will make the system more flexible and useful in real-world applications. Additional features like voice input and automatic query suggestions can also be added to improve user interaction and accessibility.

The system can also be improved by using more advanced AI models to increase accuracy and performance. IntelliSQL can be deployed on cloud platforms to allow access from anywhere. These improvements will make the system more powerful, scalable, and suitable for large-scale and enterprise-level applications.

11. APPENDIX

Source Code :

app.py

app.py

app.py > ...

```python
import streamlit as st
import os
import sqlite3
from dotenv import load_dotenv
from google import genai
def load_image(local_path, url, width=300):
    if os.path.exists(local_path):
        st.image(local_path, width=width)
    else:
        st.image(url, width=width)

load_dotenv()

client = genai.Client(api_key=os.getenv("GOOGLE_API_KEY"))


st.set_page_config(
    page_title="IntelliSQL",
    page_icon="🧠",
    layout="wide",
    initial_sidebar_state="expanded"
)

st.markdown("""
<style>
body {
    background-color: #0E1117;
    color: white;
}

.big-title {
    font-size: 40px;
    font-weight: bold;
    color: #00FFAA;
}

.sub-title {
    font-size: 18px;
    color: #CCCCCC;
}

.center {
    text-align: center;
}

.sql-box {
    background-color: #1E1E1E;
    padding: 10px;
    border-radius: 10px;
}
</style>
""", unsafe_allow_html=True)

prompt = """
You are an expert in converting English questions to SQL queries.
The SQL database name is STUDENTS and has columns:
NAME, CLASS, MARKS, COMPANY.
```

sql.py

```python
import sqlite3

# create/connect database
connection = sqlite3.connect("data.db")
cursor = connection.cursor()

# create table
table = '''
CREATE TABLE IF NOT EXISTS STUDENTS(
    NAME VARCHAR(25),
    CLASS VARCHAR(25),
    MARKS INT,
    COMPANY VARCHAR(25)
);
'''
cursor.execute(table)

# insert sample records
cursor.execute("INSERT INTO STUDENTS VALUES('Sijo','BTech',75,'JSW')")
cursor.execute("INSERT INTO STUDENTS VALUES('Anu','MCom',82,'INFOSYS')")
cursor.execute("INSERT INTO STUDENTS VALUES('Rahul','BSc',65,'TCS')")
cursor.execute("INSERT INTO STUDENTS VALUES('Meena','MCom',90,'WIPRO')")

connection.commit()
connection.close()
```

requirements.txt

```
streamlit
google-generativeai
python-dotenv
```

Dataset Link :

SQLite database (data.db)

Documentation Link:

https://drive.google.com/drive/folders/1pJOd1M96ZpUI_cN_6ZSF0JwJiuZZPx45

GitHub Link :

https://github.com/shaik-hafiza/intellisql-project

Demo Link :

https://drive.google.com/file/d/1Ke9QMuRdZ158yAXp6meW9QQtzAUqrZy9/view?usp=drivesdk