

Project Report

1. INTRODUCTION

1.1 Project Overview

1.2 Purpose

2. IDEATION PHASE

2.1 Problem Statement

2.2 Empathy Map Canvas

2.3 Brainstorming

3. REQUIREMENT ANALYSIS

3.1 Customer Journey map

3.2 Solution Requirement

3.3 Data Flow Diagram

3.4 Technology Stack

4. PROJECT DESIGN

4.1 Problem Solution Fit

4.2 Proposed Solution

4.3 Solution Architecture

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

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.

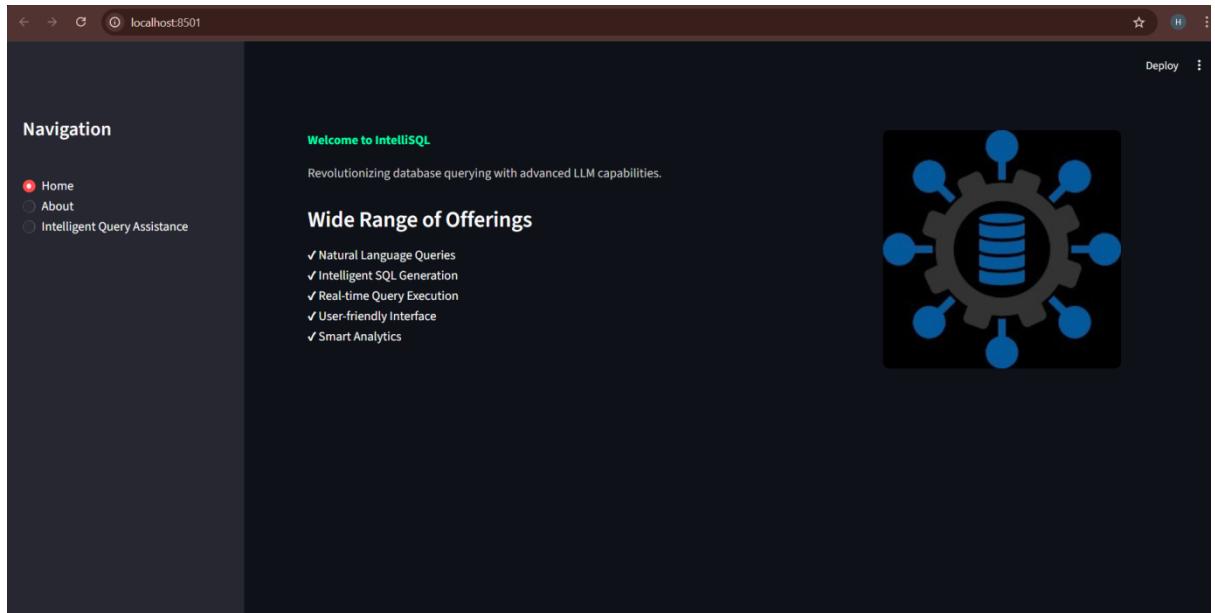
The testing results showed that the system provides fast query conversion and quick response time. The application remained stable during continuous usage and handled multiple queries without errors or crashes. This confirms that IntelliSQL meets the required performance standards and works efficiently.

7. RESULTS

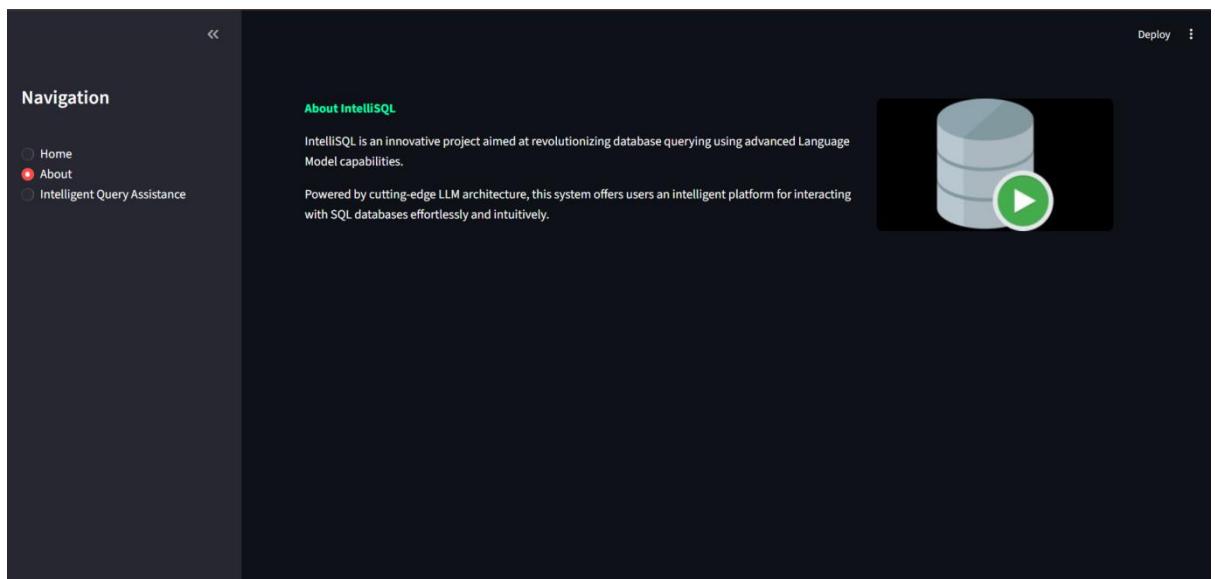
7.1 Output Screenshots

Results include:

Home Page :



About Page :



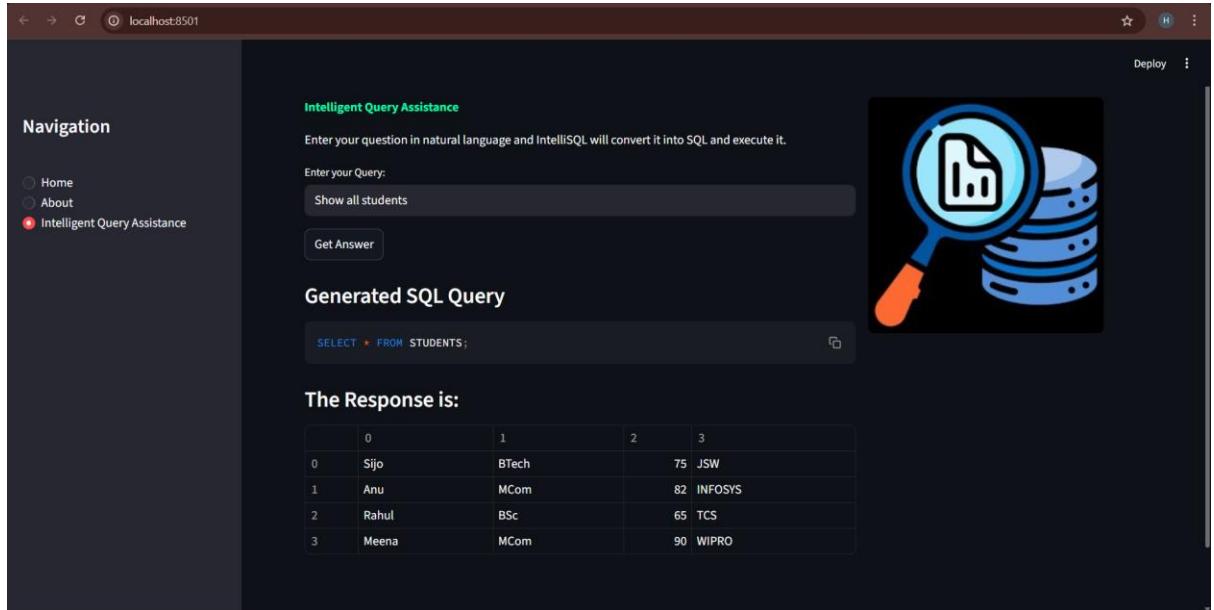
Query Input Page :

The screenshot shows a dark-themed web application interface. On the left, a vertical navigation sidebar titled "Navigation" contains three items: "Home", "About", and "Intelligent Query Assistance", with the third item being the active tab. The main content area has a title "Intelligent Query Assistance" and a sub-instruction "Enter your question in natural language and IntelliSQL will convert it into SQL and execute it.". Below this is a text input field with placeholder text "Example: Show students working in Infosys" and a "Get Answer" button. To the right of the input field is a decorative icon featuring a magnifying glass over a stack of coins.

SQL Query Generation:

This screenshot shows the same application interface as the previous one, but with a different query entered in the input field. The input field now contains the text "Show all students". The rest of the interface, including the navigation sidebar and the decorative icon, remains the same.

Query Results Display :



The screenshot shows a web application interface titled "Intelligent Query Assistance" running on localhost:8501. On the left, a dark sidebar labeled "Navigation" contains links for "Home", "About", and "Intelligent Query Assistance" (which is selected). The main content area has a dark background with green text. It includes a section for "Intelligent Query Assistance" with a placeholder "Enter your question in natural language and IntelliSQL will convert it into SQL and execute it." Below this is a search bar with the query "Show all students" and a "Get Answer" button. To the right is a large graphic of a magnifying glass over a stack of coins. A "Generated SQL Query" section shows the SQL command "SELECT * FROM STUDENTS;". The final section, "The Response is:", displays a table with four rows of student data:

	0	1	2	3
0	Sijo	BTech	75	JSW
1	Anu	MCom	82	INFOSYS
2	Rahul	BSc	65	TCS
3	Meena	MCom	90	WIPRO

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

```
❶ app.py  X
❷ app.py > ...
 1  import streamlit as st
 2  import os
 3  import sqlite3
 4  from dotenv import load_dotenv
 5  from google import genai
 6  def load_image(local_path, url, width=300):
 7      if os.path.exists(local_path):
 8          st.image(local_path, width=width)
 9      else:
10          st.image(url, width=width)
11
12 load_dotenv()
13
14 client = genai.Client(api_key=os.getenv("GOOGLE_API_KEY"))
15
16
17 st.set_page_config(
18     page_title="IntelliSQL",
19     page_icon="🧠",
20     layout="wide",
21     initial_sidebar_state="expanded"
22 )
23
24 st.markdown("""
25 <style>
26 body {
27     background-color: #0E1117;
28     color: white;
29 }
30
31 .big-title {
32     font-size: 40px;
33     font-weight: bold;
34     color: #00FFAA;
35 }
36
37 .sub-title {
38     font-size: 18px;
39     color: #CCCCCC;
40 }
41
42 .center {
43     text-align: center;
44 }
45
46 .sql-box {
47     background-color: #1E1E1E;
48     padding: 10px;
49     border-radius: 10px;
50 }
51 </style>
52 """ , unsafe_allow_html=True)
53 |
54 prompt = """
55 You are an expert in converting English questions to SQL queries.
56 The SQL database name is STUDENTS and has columns:
57 NAME, CLASS, MARKS, COMPANY.
--
```

sql.py

```
sql.py  X
sql.py > ...
1 import sqlite3
2
3 # create/connect database
4 connection = sqlite3.connect("data.db")
5 cursor = connection.cursor()
6
7 # create table
8 table = """
9 CREATE TABLE IF NOT EXISTS STUDENTS(
10     NAME VARCHAR(25),
11     CLASS VARCHAR(25),
12     MARKS INT,
13     COMPANY VARCHAR(25)
14 );
15 """
16 cursor.execute(table)
17
18 # insert sample records
19 cursor.execute("INSERT INTO STUDENTS VALUES('Sijo','BTech',75,'JSW')")
20 cursor.execute("INSERT INTO STUDENTS VALUES('Anu','MCom',82,'INFOSYS')")
21 cursor.execute("INSERT INTO STUDENTS VALUES('Rahul','BSc',65,'TCS')")
22 cursor.execute("INSERT INTO STUDENTS VALUES('Meena','MCom',90,'WIPRO')")
23
24 connection.commit()
25 connection.close()
26
27
28
```

requirements.txt

```
requirements.txt  X
requirements.txt
1 streamlit
2 google-generativeai
3 python-dotenv
4
5
6 -
```

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>