1. INTRODUCTION

1.1 Project Overview

The Text-to-SQL Query Generator using Google Gemini AI is an AI-based application that converts natural language questions into SQL queries. Users can ask database-related questions in simple English, and the system automatically generates accurate SQL queries using Google Gemini AI. The generated queries are executed on a SQLite database, and the results are displayed through a Streamlit web interface.

This project demonstrates how Large Language Models (LLMs) can simplify database interactions for non-technical users.

1.2 Purpose

The purpose of this project is to:

Reduce dependency on manual SQL writing

Allow non-technical users to interact with databases

Demonstrate real-world usage of Generative AI

Provide a simple and user-friendly interface

2. IDEATION PHASE

2.1 Problem Statement

Many users need information from databases but lack SQL knowledge. Writing SQL queries requires technical expertise. This project solves this issue by converting English questions into SQL queries automatically.

2.2 Empathy Map Canvas

User Thinks:

“I need student data but I don’t know SQL.”

User Feels:

Confused about writing SQL queries.

User Says:

“Can I just ask in English?”

User Does:

Types a simple English question.

2.3 Brainstorming

During brainstorming, the following ideas were discussed:

Use AI model for query generation

Use Streamlit for UI

Use SQLite for database

Secure API key using dotenv

Provide real-time results display

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

User opens application

User types question

System processes question

SQL query generated

Query executed

Results displayed

3.2 Solution Requirement

Functional Requirements

Accept natural language input

Convert text to SQL query

Execute query on database

Display results

Non-Functional Requirements

Fast response time

Secure API key handling

User-friendly interface

3.3 Data Flow Diagram

User Input → Gemini AI → SQL Query → SQLite Database → Results → Display

3.4 Technology Stack

Python

Streamlit

Google Gemini AI

SQLite

python-dotenv

4. PROJECT DESIGN

4.1 Problem Solution Fit

The system successfully solves the problem of SQL complexity by enabling natural language interaction with databases.

4.2 Proposed Solution

The proposed system uses Google Gemini AI to convert English text into SQL queries dynamically. These queries are executed on a predefined student database.

4.3 Solution Architecture

User

↓

Streamlit Interface

↓

Gemini AI API

↓

SQL Query

↓

SQLite Database

↓

Results Display

5. PROJECT PLANNING & SCHEDULING

Phase

Description

Week 1

Problem Analysis & Research

Week 2

Database Creation

Week 3

AI Integration

Week 4

Testing & Documentation

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

Tested with multiple queries

Response time within few seconds

Correct SQL generation for basic queries

7. RESULTS

The application successfully:

Converts English text into SQL

Executes queries

Displays accurate student data

7.1 Output Screenshots

(Add screenshots of:)

Streamlit Interface

SQL Query Generated

Output Table

Terminal execution

8. ADVANTAGES & DISADVANTAGES

Advantages

Easy for beginners

Saves time

AI-powered automation

User-friendly

Disadvantages

Requires internet connection

Limited to predefined database

May generate incorrect SQL for complex queries

9. CONCLUSION

The Text-to-SQL Query Generator using Google Gemini AI successfully demonstrates the practical use of Large Language Models in database systems. It simplifies database interaction and makes data access easier for non-technical users.

10. FUTURE SCOPE

Support multiple databases

Add voice input

Deploy on cloud platform

Improve error handling

Add authentication system

11. APPENDIX

Source Code

GitHub repository link

https://github.com/chandana-859/Gemini-sql-project