Requirement Analysis

Technology Stack (Architecture & Stack)

Date	February 18, 2026
Team ID	LTVIP2026TMIDS80425
Project Name	Smart Bridge – Intelligent SQL Querying
Maximum Marks	4 Marks

Technical Architecture

The deliverable shall include the architectural diagram and the information as per Table 1 & Table 2.

Architecture Overview:
The Smart Bridge Intelligent SQL Querying application follows a three-tier architecture with a React frontend, FastAPI backend, and SQLite database, integrated with Google Gemini AI for natural language processing.

System Architecture

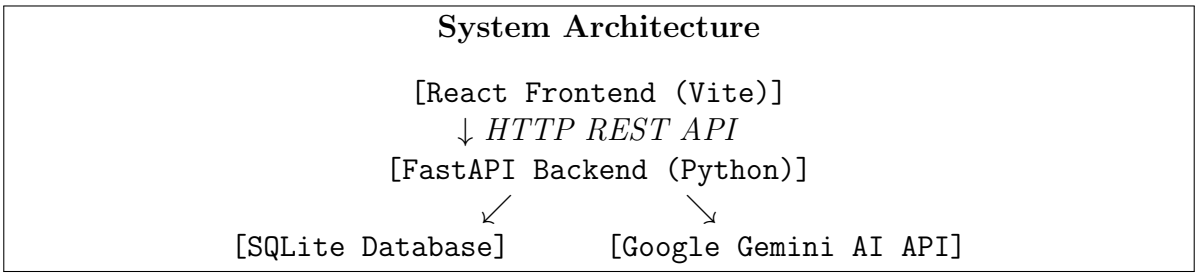

Table-1: Components & Technologies

S.No	Component	Description	Technology
1	User Interface	Modern, responsive web-based UI for natural language query input, schema browsing, and results display	React 18, Vite, CSS3, JavaScript (ES6+)
2	Application Logic – Query Processing	Core backend logic for receiving NL queries, coordinating AI generation, and executing SQL	Python 3.11, FastAPI, Uvicorn
3	Application Logic – AI Integration	Integration with Google Gemini for converting natural language to SQL queries	Google Generative AI SDK (google-generativeai)

4	Application Logic – Schema Extraction	Logic to extract and serve database schema information (tables, columns, types)	Python sqlite3 module, FastAPI
5	Database	Relational database storing sample e-commerce data (customers, products, orders, order_items)	SQLite 3
6	External API	Google Gemini AI service for natural language understanding and SQL generation	Google Gemini 1.5 Flash API
7	API Gateway	RESTful API layer handling frontend-backend communication with CORS support	FastAPI, Starlette CORS Middleware
8	Development Tools	Build tools, package managers, and development servers	npm, pip, Vite Dev Server, Uvicorn
9	Version Control	Source code management and collaboration	Git, GitHub
10	Infrastructure	Local development environment	Local Server (localhost:5173 frontend, localhost:8000 backend)

Table-2: Application Characteristics

S.No	Characteristics	Description	Technology
1	Open-Source Frameworks	React for UI, FastAPI for backend API, Vite for build tooling	React 18 (MIT), FastAPI (MIT), Vite (MIT)
2	Security Implementations	Read-only SQL execution (SELECT only), input sanitization, API key management via environment variables, CORS configuration	Python dotenv, FastAPI CORS Middleware, SQL sanitization
3	Scalable Architecture	Three-tier architecture with modular AI service layer. Database engine can be swapped (SQLite → PostgreSQL). AI provider is abstracted.	FastAPI modular routing, Service layer pattern

4	Availability	Frontend served as static SPA. Backend runs as standalone ASGI server. Both can be deployed independently.	Vite static build, Uvicorn ASGI server
5	Performance	Fast query response (< 5s including AI). Lightweight SQLite for low-latency local queries. React virtual DOM for efficient UI updates.	SQLite in-memory caching, React 18 concurrent features

References

- https://c4model.com/
- https://fastapi.tiangolo.com/
- https://react.dev/
- https://ai.google.dev/