📚 NLQ-to-SQL Library System

Complete Technical Documentation

React Frontend Express Backend MySQL Database Gemini AI

🧬 System Architecture

Frontend (React)

User Interface

Material-UI Components

↓ HTTP Requests ↓

Backend (Express)

API Endpoints

Business Logic

↙ SQL ↓ AI ↘

MySQL DB

Aiven Cloud

Gemini AI

NLQ → SQL

🔄 Complete API Flow

1 **User enters natural language question**

`QueryInput.jsx` captures input

↓

2 **Frontend sends POST request**

`api.js → POST /api/query`

↓

3 **Backend receives request**

`server.js → queryRoutes.js`

↓

4 **Extract database schema**

`schemaExtractor.js → MySQL INFORMATION_SCHEMA`

↓

5 **Generate SQL using Gemini AI**

`llmService.js → Gemini API (NLQ + Schema → SQL)`

↓

6 **Validate & execute SQL**

`sqlService.js` → MySQL database

↓

7 Return results to frontend

`ResultsTable.jsx` displays data

📁 Project File Structure

```
nlq-2-sql/
├── backend/
│   ├── api/
│   │   └── index.js              # Vercel serverless entry point
│   ├── config/
│   │   └── db.js                 # MySQL connection pool
│   ├── routes/
│   │   └── queryRoutes.js        # API route handlers (/query, /schema)
│   ├── services/
│   │   ├── llmService.js         # Gemini AI integration
│   │   └── sqlService.js         # SQL execution & retry logic
│   ├── utils/
│   │   └── schemaExtractor.js    # Extract DB schema
│   ├── server.js                 # Express app configuration
│   ├── package.json
│   └── vercel.json               # Vercel deployment config
│
└── frontend/
    ├── src/
    │   ├── components/
    │   │   ├── QueryInput.jsx     # User input component
    │   │   ├── ResultsTable.jsx   # Display query results
    │   │   └── Navbar.jsx         # Navigation bar
    │   ├── pages/
    │   │   ├── HomePage.jsx       # Main query page
    │   │   ├── TablesPage.jsx     # Database tables view
    │   │   └── DocsPage.jsx       # Documentation page
    │   ├── services/
```

```
│   │   └── api.js              # Axios API calls
│   ├── App.jsx                 # Main app & routing
│   ├── theme.js                # Material-UI dark theme
│   └── main.jsx                # React entry point
├── package.json
└── vercel.json                 # SPA routing config
```

⚙️ Backend Files Explained

1. server.js

Purpose: Main Express application configuration

```
// Sets up Express, CORS, routes // Exports app for Vercel serverless // Health check endpoint: /api/health
```

Responsibility	Details
CORS Configuration	Allows requests from frontend (nlq-frontend.vercel.app)
Route Registration	Maps /api/* to queryRoutes
Error Handling	Catches database connection errors

2. config/db.js

Purpose: MySQL connection pool management

```
// Creates mysql2 connection pool // Handles both local (localhost) and production (Aiven) // SSL enabled for production only
```

3. routes/queryRoutes.js

Purpose: API endpoint definitions

Endpoint	Method	Function
/api/query	POST	Execute natural language query
/api/schema	GET	Get database schema
/api/health	GET	Check database connection

4. services/llmService.js

Purpose: Gemini AI integration for NLQ → SQL conversion

```
// Sends prompt: "Convert '{question}' to SQL for {schema}" // Cleans markdown formatting from response // Returns pure SQL query string
```

5. services/sqlService.js

Purpose: SQL execution with retry logic

```
// 1. Generate SQL from NLQ // 2. Validate query (no DROP, DELETE, UPDATE) // 3. Execute on MySQL // 4. If fails: Retry with error context (self-correction) // 5. Return results or error
```

6. utils/schemaExtractor.js

Purpose: Extract database schema dynamically

```
// Queries INFORMATION_SCHEMA.TABLES // DESCRIBE each table for columns // Returns formatted schema string for AI
```

7. api/index.js

Purpose: Vercel serverless function entry point

```
// Imports Express app from server.js // Exports for Vercel's serverless environment // Handles all incoming requests
```

🎨 Frontend Files Explained

1. App.jsx

Purpose: Main application component with routing

Feature	Implementation
React Router	Routes: /, /tables, /docs
Theme Provider	Material-UI dark theme
Health Check	Tests backend connection on load

2. components/QueryInput.jsx

Purpose: User input interface with sample queries

```
// TextField for natural language input // Sample query chips for quick testing // Sends question to api.js → /api/query
```

3. components/ResultsTable.jsx

Purpose: Display query results in formatted table

```
// Shows generated SQL query // Renders data table with all columns // Error handling with retry option // Row count &
```

```
attempt number display
```

4. pages/TablesPage.jsx

Purpose: View all database tables with caching

```
// Tabs for 4 tables: Books, Members, Staff, Transactions // localStorage caching (5 min expiry) // Refresh button for manual update // Shows cache status & last updated time
```

5. services/api.js

Purpose: Axios HTTP client for API calls

Function	API Call
executeQuery(question)	POST /api/query
getSchema()	GET /api/schema
checkHealth()	GET /api/health

🗄️ Database Schema

Books Table

Column	Type	Description
book_id	INT (PK)	Unique book identifier
title	VARCHAR(255)	Book title

author	VARCHAR(255)	Author name
isbn	VARCHAR(13)	ISBN code (unique)
category	VARCHAR(100)	Fiction, Technology, etc.
available_copies	INT	Currently available
status	ENUM	available | unavailable | reserved

Members Table

Column	Type	Description
member_id	INT (PK)	Unique member identifier
name	VARCHAR(255)	Member's full name
email	VARCHAR(255)	Email address (unique)
membership_type	ENUM	student | faculty | public
join_date	DATE	Membership start date

Staff Table

Column	Type	Description
staff_id	INT (PK)	Unique staff identifier
name	VARCHAR(255)	Staff member name
position	VARCHAR(100)	Librarian, Assistant, etc.
email	VARCHAR(255)	Email address (unique)

salary	DECIMAL(10,2)	Monthly salary

Transactions Table (Relationships)

Column	Type	Foreign Key
transaction_id	INT (PK)	-
book_id	INT (FK)	→ books.book_id
member_id	INT (FK)	→ members.member_id
status	ENUM	issued | returned | overdue
fine_amount	DECIMAL(10,2)	Late fee

🛠️ Technology Stack

React 18

Frontend UI library with hooks

Material-UI

Component library & theming

Node.js + Express

MySQL

Backend REST API

Relational database (Aiven)

Gemini Flash 2.0

AI for NLQ → SQL

▲

Vercel

Serverless deployment

🔐 Environment Variables

Backend (.env)

```
DB_HOST=your-aiven-host.aivencloud.com DB_USER=avnadmin
DB_PASSWORD=your_password DB_NAME=defaultdb DB_PORT=3306
GEMINI_API_KEY=your_gemini_key NODE_ENV=production
```

Frontend (.env.production)

```
VITE_API_URL=https://nlq-2-sql.vercel.app/api
```

🚀 Deployment Architecture

Component	Platform	URL

Frontend	Vercel (Static)	nlq-frontend.vercel.app
Backend API	Vercel (Serverless)	nlq-2-sql.vercel.app/api
MySQL Database	Aiven (Cloud)	*.aivencloud.com:3306
AI Model	Google Cloud	Gemini API