Website: https://cipherschools.com | **Email**: nitesh@cipherschools.com | **Contact**: +91-8080399840

CipherSQLStudio Assignment

Overview

Build **CipherSQLStudio** - a browser-based SQL learning platform where students can practice SQL queries against pre-configured assignments with real-time execution and intelligent hints.

Project Description

Create a web application that allows users to:

- View SQL assignment questions with pre-loaded sample data
- Write and execute SQL queries in a browser-based editor
- Get intelligent hints (not solutions) from an integrated LLM
- See query results in real-time

Important: This is NOT a database creation tool. Assignments and sample data will be pre-inserted by administrators into the database. Your focus is on building the user experience for attempting and solving SQL assignments.

What You'll Build

Core Features (Required - 90%)

1. Assignment Listing Page

- Display all available SQL assignments
- Show assignment difficulty, title, and brief description

- Allow users to select an assignment to attempt

2. Assignment Attempt Interface

- **Question Panel:** Display the selected assignment question and requirements
- **Sample Data Viewer:** Show the pre-loaded table schemas and sample data relevant to this assignment
- **SQL Editor:** Code editor for writing SQL queries (Monaco Editor recommended)
- **Results Panel:** Display query execution results in a formatted table
- **LLM Hint Integration:** A "Get Hint" button that uses an LLM API to provide guidance (not the full solution)

3. Query Execution Engine

- Execute user-submitted SQL queries against PostgreSQL
- Return results or error messages
- Implement query validation and sanitization for security

Optional Features (10%)

- Login/Signup system for users
- Save user's SQL query attempts for each assignment

Note: Submissions found to be built using AI-generated code will be disqualified. We are not evaluating completeness – we are evaluating understanding. Build what you can, but build it yourself.

Technical Requirements

Frontend Stack

Required: React.js
Styling: Vanilla SCSS with mobile-first responsive design approach

Why? We're specifically asking for vanilla SCSS to evaluate your fundamental styling abilities.

Required Approach

- Build mobile-first responsive layouts (320px, 641px, 1024px, 1281px)
- Use SCSS features: variables, mixins, nesting, and partials
- Follow BEM or similar CSS naming conventions
- Ensure touch-friendly UI elements for mobile devices

Backend Stack

Component	Technology
Runtime	Node.js / Express.js
Sandbox Database	PostgreSQL
Persistence DB	MongoDB (Atlas preferred)
Code Editor	Monaco Editor

LLM Integration

- Integrate any LLM API (OpenAI, Gemini, etc.)
- **Critical:** The LLM should provide hints, not complete solutions
- Implement proper prompt engineering to guide the LLM's responses

---

Deliverables

1. GitHub Repository
   - Frontend code and backend code
   - Clear folder structure
   - .env.example files with required variables
   - Installation and setup instructions
2. README.md
   - Project setup instructions
   - Environment variables needed
   - Technology choices explanation
3. Data-Flow Diagram (Compulsory)

    - User clicks "Execute Query" → Label every step → Result displays
    - Include: API calls, database queries, state updates
    - **Must be drawn by hand** (proves understanding)
4. Project Demo (Optional)
    - Show assignment selection
    - Query execution demo
    - Hint generation in action
    - Mobile responsive view

Evaluation Criteria

Category	Weight	What We're Looking For
Core functionality & Data-Flow Diagram	50%	Features work as expected, proper error handling
CSS (vanilla css)	15%	Mobile-first approach, proper use of SCSS features, responsive design
Code structure & readability	10%	Clean, readable, well-structured code with proper separation of concerns
UI/UX clarity	10%	Intuitive interface, good visual hierarchy, smooth user flow
LLM Integration	10%	Effective prompt engineering, hints are helpful but not revealing solutions
Demo Video	5%	

Reference & Helping Guide

A simplified helping guide is provided below with:

- Essential schema design hints
- Key architectural decisions

- Sandbox Guide

Note: The guide provides hints and direction, not complete solutions. You must figure out implementation details yourself. This is meant to clarify the project requirements and help you complete it faster.

https://docs.google.com/document/d/1nWE56xDx_Tw5ZE9z_QytgfFx-oyY7r3e2-Iv1LCZEWk/edit?usp=sharing

8. Submission Form

Deadline: 3 days from assignment start date

Google Form Link:
https://docs.google.com/forms/d/e/1FAIpQLScO35LDliPDBut4GRk05ah7yVxEXkQLJP1ZACq_0n2s1j0Kww/viewform?usp=dialog

---