Developing ChatDB   
An Interactive Natural Language Database Query System

USC Viterbi School of Engineering

Sep 21, 2024

# **ChatDB Team 12**

**David Schirmer** :

**(USC ID (4858550670)**  
Studying Data Science, graduating Fall 2025. Masters in Business Analytics and currently work in cloud architecture.

**Harshal Borhade**

**(USC ID: 8976537629)**  
Studying Computer Science, graduating in May 2025, I have experience in systems programming and software development.  
  
**Fariha Sheikh:**

Studying Master's in Applied Data Science and expecting graduation in Spring of 2026.

# Project Requirements

## General Overview

* 1. **Purpose**: Assist users in learning how to query data in PostgreSQL (SQL) and MongoDB (NoSQL) database systems.
  2. **Key Functionality**: Users can select a database type (SQL or NoSQL), choose a table/collection, receive sample queries, and execute these queries with results visualized.
  3. **Platform**: A web-based user interface (UI) for user interaction.

## Functional Requirements

* 1. **Database Selection:** Users should be able to choose between PostgreSQL (SQL) or MongoDB (NoSQL) as their database system. Once a database type is selected, users can browse and select specific tables (PostgreSQL) or collections (MongoDB).
  2. **Predefined Patterns**: Provide users with sample queries for the selected database system, based on specific language constructs (e.g., JOIN, WHERE, GROUP BY in SQL; find(), aggregate() in NoSQL). The system will suggest and generate queries following a pattern.
  3. **Custom Queries**: Allow users to ask questions about their data (e.g., "How many users registered last month?") and automatically generate the corresponding query using a pattern-matching algorithm.
  4. **Query Execution**: Run the generated or custom-written queries against the selected database system (PostgreSQL/MongoDB) and display the query results directly on the web interface in an easy-to-read format.
  5. **Browser-Based UI**: A user-friendly web-based interface that allows users to access the system.

# Project Implementation

### Database System:

**SQL**: We have chosen **PostgreSQL** as our SQL database management system due to its well-structured syntax, comprehensive documentation, and significant market presence.

**NoSQL**: We will be implementing **MongoDB** for handling unstructured data and facilitating Big Data analysis. Its flexibility and widespread adoption make it an ideal choice for our NoSQL database needs.

### Dataset:

1. <https://www.kaggle.com/datasets/ahmedabbas757/coffee-sales?resource=download> The dataset includes the transaction date and timestamp, which captures the exact time each transaction happened; the location, indicating which of the three coffee shop locations the transaction took place; and product-level details that include info about the items purchased, quantities, and prices.
2. <https://www.kaggle.com/datasets/gregorut/videogamesales>  
   This dataset contains a list of video games with sales greater than 100,000 copies. Fields include ranking of overall sales, Platform of the game's release (i.e. PC, PS4, etc.), Year - Year of the game's release, Genre - Genre of the game, Publisher - Publisher of the game, and continent-wide sales metrics.
3. <https://www.kaggle.com/datasets/qinsights/pharma-sales-dataset>  
    Kaggle consists of pharmaceutical sales data, useful for analysis of sales trends, forecasting, and insights within the pharma industry. It contains features such as dates, sales quantities, product types, and possibly regions or categories for a more comprehensive understanding of the market. It is structured for tasks like time series forecasting or regression modeling for pharmaceutical sales.

### Languages Used for the Project:

1. **Python:** Excellent support for web development and natural language processing. We'll use it to build the natural language processing (NLP) engine, query generators, and database connectors. We will be using the Flask library to create a web interface.
2. **JavaScript along with HTML and CSS**: We will create an interactive and responsive web interface. We might use libraries like react or vue.js to create the user interface.

### 

### User Interaction with ChatDB:

#### User Interface and Interaction

The system will feature a web-based User Interface (UI) that enables users to interact with the database system. The UI will include the following components:

1. **Query Input**: A chat box where users can enter their queries.
2. **Suggested Queries**: The system will provide suggested queries to assist users in formulating their questions.
3. **Database Selection**: A dropdown menu that allows users to select the database to query (PostgreSQL or MongoDB).
4. **Dataset Selection**: A dropdown menu that enables users to choose the specific dataset to query.

#### Query Processing and Output

Once the user submits a query, the system will:

1. **Generate Query**: Display the generated query as output, if it matches a recognized pattern.
2. **Suggestion**: Provide a suggestion if the query does not match any patterns.
3. **Execution Results**: Display the results of the query execution, including a table or chart (if implemented).
4. **Data Visualization (Optional)**: If time permits, the system will include data visualization using charts to enhance the representation of query results.

### Implementation Plan:

#### Proposed Timeline:

**(Week one-two)**: Developing ChatDB involves several key steps to create a functional and user-friendly system within our project's timeline. Week one we will start with system design and setup, where we identify and plan the main parts of the application. These include the user interface, the natural language processing (NLP) engine, query generators for both SQL and NoSQL databases, and database connectors. Setting up the databases is also an important early step. Both MySQL and MongoDB will be configured with a dataset.

**(Week three-four)**: Once we have the basics in place, creating a NLP module using Python, which will parse natural language inputs, figure out what the user wants, and identify key details needed to form queries. At the same time, we'll develop functions to translate the parsed input into executable SQL and MongoDB queries, so the system can interact effectively with both types of databases. Secure database connectors will be set up to handle running queries and getting results back, keeping data secure and ensuring reliable communication between the app and the databases.

**(Week five-six)**: We will focus on the front end, we'll build the web interface using HTML, CSS, and JavaScript to make sure it's responsive and interactive. The interface will be designed to enhance user experience by being intuitive and easy to navigate. We'll integrate it with the backend using Flask, allowing for smooth handling of user inputs and dynamic display of results. We'll also add features like autocomplete, schema exploration tools, and data visualization components to help users formulate queries and interpret data effectively.

**(Week seven+)**: Testing and optimization are crucial throughout the development process. We'll write unit tests for individual modules to check they work properly, and integration testing will make sure all the parts work together seamlessly. We'll optimize performance by fine-tuning the NLP processing and query execution to make things faster and more efficient. Furthermore, we'll also conduct user acceptance testing by gathering feedback from test users, which will help us improve usability based on real-world interactions. Finally, we'll deploy the application on a web server to make it accessible to our intended users. This involves setting up the server environment, implementing necessary security measures, and ensuring the system stays stable when multiple users are on it.

**(Throughout development)**: We will need to keep up-to-date documentation as we work on our separate systems. This will include detailed records of the codebase, decisions we make, and how the system functions, which is important for future maintenance and scaling. By following these development steps, we're aiming to provide a robust and user-friendly platform for natural language database querying with ChatDB. Focusing on both backend and frontend development, along with thorough testing and documentation, helps ensure the final product meets our project's goals and offers a valuable tool for users with different levels of technical expertise.

### Roles and responsibilities:

* Harshal will be working on creating the NLP module for translating user input into NoSQL queries and execution using MongoDB, and also assisting with development of the user interface.
* David’s responsibility will be in developing the SQL portion of ChatDB. This includes the design of the SQL database schemas, the implementation of a module that will translate natural language inputs into executable SQL queries, and ensuring an optimized connection between the application and the database.
* Fariha’s responsibilities include developing the web-based user interface using HTML and JavaScript libraries and assisting with the NLP engine development.