SmartSQL Insight: Natural Language to SQL Query Converter

Group Name: THUNDER BUDDIES

Team Members:

- NAGALLA DEVISRI PRASAD (22CS10045)
- MANTRI JASWANTH (22CS10041)
- ADAVATH THARUN KUMAR (22CS10001)
- BHUKYA MAHESH (22CS10017)
- GULLOLLA VAMSHINATH (22CS10027)

---

1. Problem Statement

In the contemporary data-driven landscape, the ability to effectively query and analyze data through SQL remains a critical skill. However, this requirement presents a significant challenge: while data analysis is increasingly becoming a necessity across various professional domains, the technical expertise required to write SQL queries creates a substantial barrier for non-technical users.

This challenge is particularly evident in organizations where data-driven decision making is crucial, but the technical skills required to access and analyze the data are concentrated among a limited number of technical staff. The complexity of SQL query writing, combined with the need to understand database schemas and relationships, often leads to bottlenecks in data access and analysis. This situation not only slows down decision-making processes but also creates a dependency on technical staff for even simple data queries. Additionally, the traditional approach to database access lacks user-friendly interfaces and feedback mechanisms that could help improve query accuracy and user learning over time.

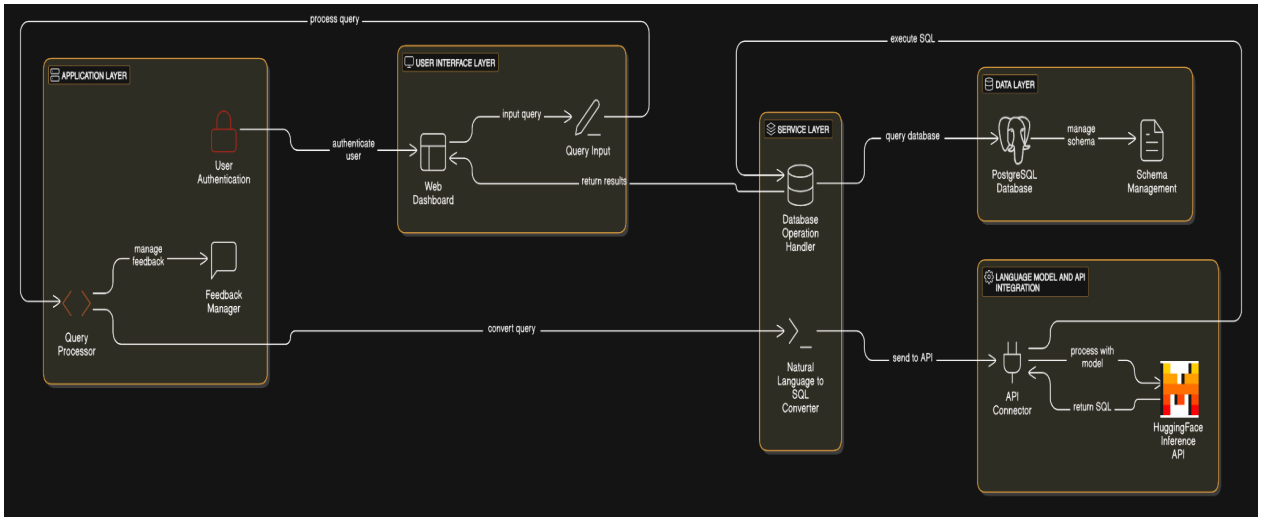
2. Methodology

Our solution, SmartSQL Insight, addresses these challenges through a comprehensive approach that combines modern web technologies with advanced language processing capabilities. The system is built on Django 4.2.7, a robust Python web framework, which provides the foundation for our application's architecture. At its core, the system employs a sophisticated natural language processing pipeline powered by HuggingFace's language models, which enables the translation of everyday language into precise SQL queries.

The application's architecture is designed with a clear separation of concerns, ensuring maintainability and scalability:

- The **user interface layer** provides an intuitive web-based dashboard where users can input their queries in natural language
- The **application layer** handles user authentication, query processing, and feedback management
- The **service layer** manages the conversion of natural language to SQL and handles database operations
- The **data layer**, powered by PostgreSQL, stores user queries, feedback, and maintains the database schema information

The natural language to SQL conversion process is particularly sophisticated. When a user submits a query, the system first extracts the current database schema, including table structures and relationships. This information is then combined with the user's natural language query to create a context-rich prompt for the language model. The model generates an SQL query, which is then executed against the database, and the results are presented to the user in a formatted manner.



Language Model and API Integration

The core of SmartSQL Insight's natural language processing capabilities is powered by the Mistral-7B-Instruct-v0.2 model, hosted on HuggingFace's Inference API. This state-of-the-art language model, developed by Mistral AI, represents a significant advancement in natural language understanding and generation. The model's architecture, based on a 7-billion parameter transformer, enables it to understand complex natural language queries and generate accurate SQL statements.

The integration with HuggingFace's Inference API is implemented through a sophisticated service layer that handles the communication between our application and the language model. The system employs a carefully crafted prompt engineering approach that combines three key elements:

1. The database schema information, which provides the model with the necessary context about available tables, columns, and relationships
2. The user's natural language query
3. Specific instructions for SQL query generation

The prompt template is designed to ensure that the model generates only valid SQL queries without additional explanations or text. This is achieved through a structured prompt format that includes:

Given the following PostgreSQL database schema:
{schema_info}
Convert this natural language question to a valid SQL query:
Question: {question}
Return only the SQL query without any explanation or additional text.
if the question is not relevant to database schema then give only the text "NOT RELEVANT TO DATABASE"

The API integration is implemented with specific parameters optimized for SQL generation:

- `max_new_tokens`: 512 (ensures sufficient length for complex queries)
- `temperature`: 0.1 (maintains high consistency in output)
- `top_p`: 0.9 (balances between creativity and accuracy)
- `return_full_text`: False (focuses on the generated query)

The system includes robust error handling and response processing:

1. Authentication is handled through secure API key management
2. Response validation ensures the generated text contains valid SQL
3. Regular expression pattern matching extracts the SQL query from the response
4. Error handling manages API failures and unexpected response formats

Query Processing Pipeline

The natural language to SQL conversion process follows a sophisticated pipeline:

1. **Schema Extraction**: The system first retrieves the complete database schema, including:
    - Table structures
    - Column names and types
    - Foreign key relationships
    - Primary keys
2. **Prompt Construction**: The schema information is formatted into a clear, structured format that the model can understand, including:
    - Table definitions
    - Column specifications
    - Relationship mappings
3. **Query Generation**: The model processes the prompt and generates an SQL query, with the system:
    - Validating the response format
    - Extracting the SQL query
    - Ensuring query completeness
4. **Query Execution**: The generated SQL is executed against the database, with results:
    - Formatted for display
    - Stored in the query history
    - Made available for export

This enhanced implementation provides several advantages:

- High accuracy in SQL generation
- Consistent query formatting
- Robust error handling
- Efficient processing of complex queries
- Scalable architecture for future improvements

The use of the Mistral-7B-Instruct model specifically brings several benefits to the system:

- Superior understanding of natural language nuances
- Better handling of complex query requirements
- Improved accuracy in SQL generation
- Robust performance across various query types
- Ability to handle ambiguous or poorly formed queries
- Despite being a 7B parameter model, Mistral delivers performance comparable to much larger models while requiring fewer computational resources, enabling faster response times and lower hosting costs
- The model's strong contextual understanding allows it to generate SQL that respects database schema constraints and relationships even when the natural language query doesn't explicitly mention them
- Mistral's instruction-tuned design enables it to maintain context across follow-up queries, allowing users to refine or build upon previous questions without restating the entire context

3. Results and Demonstration

User Authentication

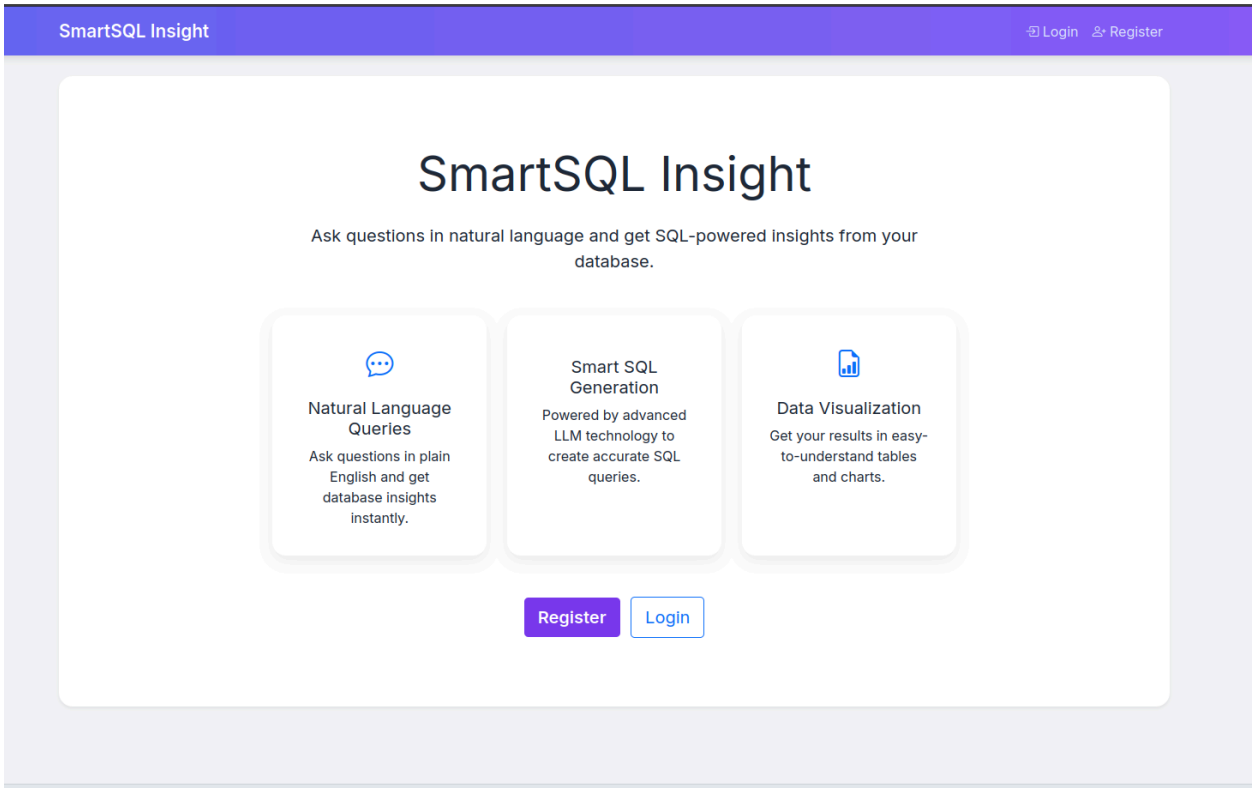
- Secure registration and login
- User-specific query history
- Protected routes and resources

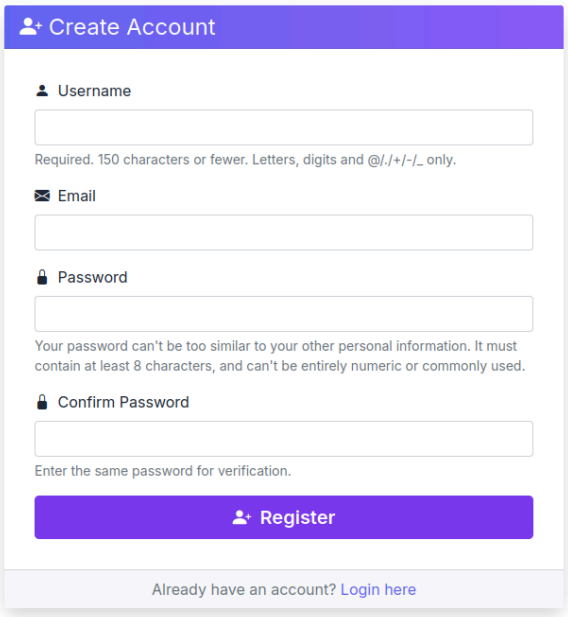
Query Interface

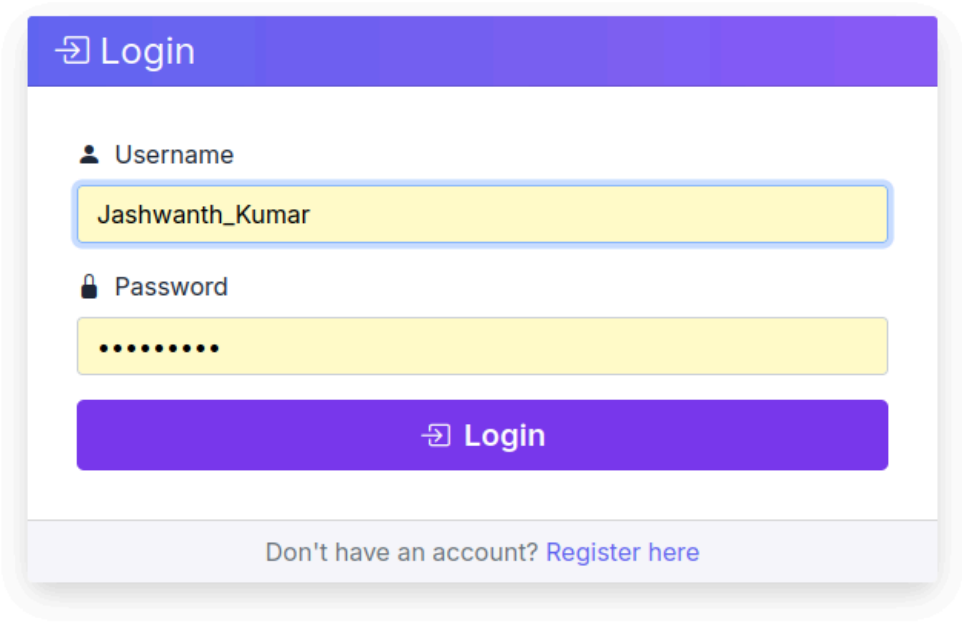
- Natural language input
- Real-time SQL generation
- Result display as tables
- Error handling

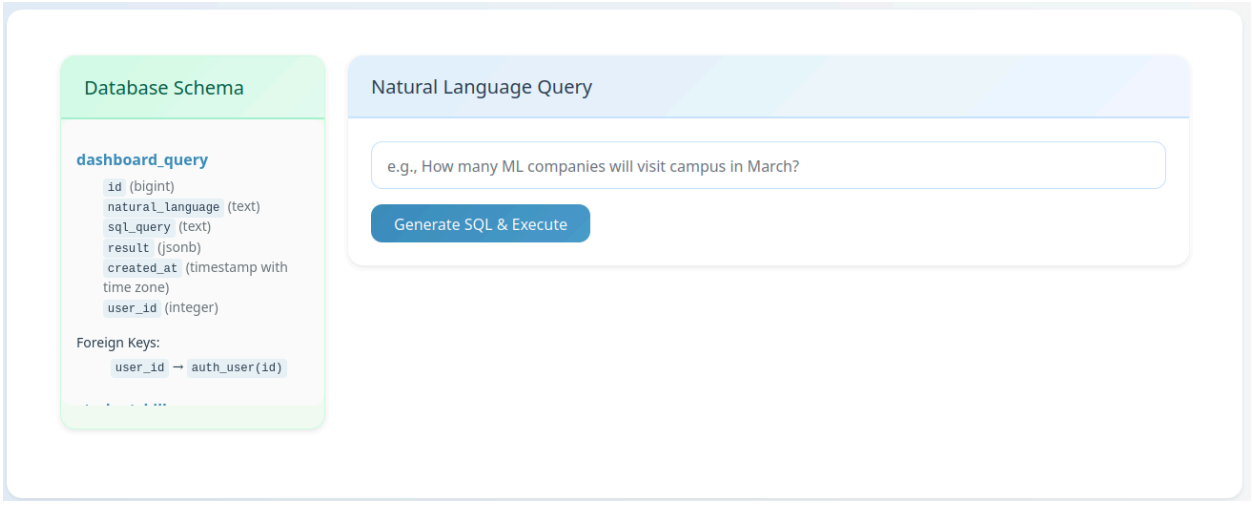
Query Management

- Historical query tracking
- Query rerun capability


SmartSQL Insight
Login
Register
SmartSQL Insight
Ask questions in natural language and get SQL-powered insights from your database.
Natural Language Queries
Ask questions in plain English and get database insights instantly.
Smart SQL Generation
Powered by advanced LLM technology to create accurate SQL queries.
Data Visualization
Get your results in easy-to-understand tables and charts.
Register
Login


Create Account
Username
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.
Email
Password
Your password can't be too similar to your other personal information. It must contain at least 8 characters, and can't be entirely numeric or commonly used.
Confirm Password
Enter the same password for verification.
Register
Already have an account? Login here


Login
Username
Jashwanth_Kumar
Password
Login
Don't have an account? Register here


Database Schema
dashboard_query
id (bigint)
natural_language (text)
sql_query (text)
result (jsonb)
created_at (timestamp with time zone)
user_id (integer)
Foreign Keys:
user_id → auth_user(id)
Natural Language Query
e.g., How many ML companies will visit campus in March?
Generate SQL & Execute

Natural Language Query

How many ML companies will visit campus in March?

Generate SQL & Execute

Generated SQL

```
SELECT COUNT(*) FROM companies WHERE industry = 'ML' AND visit_month = 3
```

Query Results

count
2

Feedback

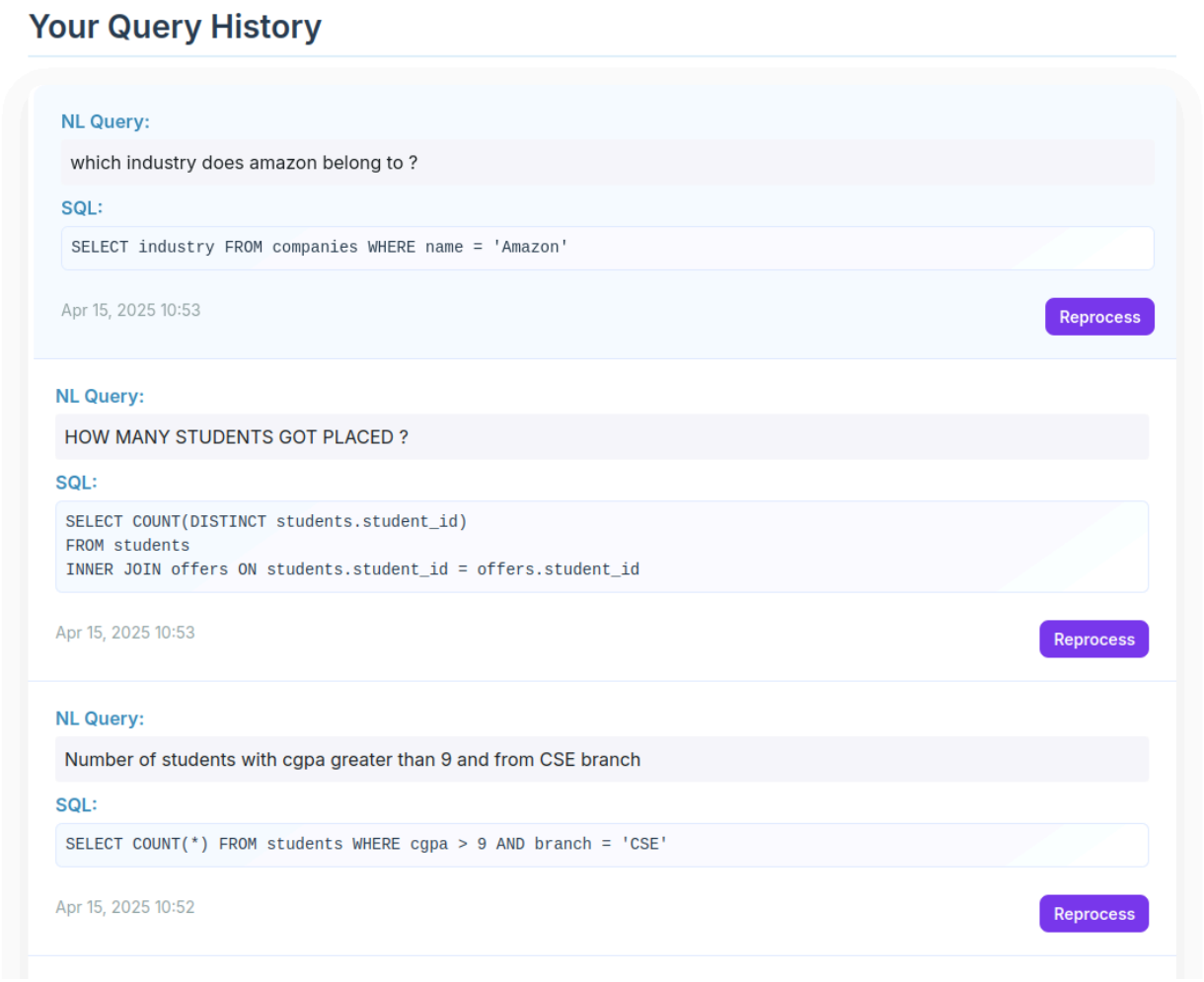
☐ Was this query helpful?

Rate this query: 1 2 3 4 5

Comments:

Any suggestions or feedback?

Submit Feedback



4. Conclusion and Future Scope

SmartSQL Insight represents a significant step forward in making database access more democratic and user-friendly. By eliminating the technical barrier of SQL syntax while maintaining the power and flexibility of SQL queries, the system enables a broader range of users to access and analyze data effectively. The implementation of user feedback and query history features ensures that the system not only serves immediate needs but also contributes to long-term improvement in query accuracy and user understanding.

Looking ahead, there are several promising directions for future development:

- Integration of additional language models and providers to enhance the system's ability to handle complex queries and improve accuracy
- Advanced features such as query optimization suggestions and visual query builders to further simplify the user experience
- Performance improvements through caching and query optimization to enhance the system's efficiency
- Enhanced security features to ensure safe and controlled access to sensitive data
- Add capability to handle multiple connected databases simultaneously
- Connect query results directly to visualization libraries for automatic chart generation

5. References

1. Django Project. (2023). Django Documentation. https://docs.djangoproject.com/en/4.2/

2. HuggingFace. (2023). Inference API Documentation. https://huggingface.co/docs/api-inference/

3. PostgreSQL Global Development Group. (2023). PostgreSQL Documentation. https://www.postgresql.org/docs/

4. The pandas development team. (2023). pandas: powerful Python data analysis toolkit. https://pandas.pydata.org/docs/

5. python-dotenv. (2023). Documentation. https://github.com/theskumar/python-dotenv

6. psycopg2. (2023). Documentation. https://www.psycopg.org/docs/

7. Mistral AI. (2023). Mistral-7B-Instruct-v0.2 Model. https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2