LLM Agents for SQL / Pandas Query Generation

Proposed By:

Abhishek Sureddy
Akshay Kumar Sureddy
Avinash Reddy Vasipalli
Muhammad Yusuf Hassan
Dhanush Kanth Anand

Objective

- Web-based tool to convert user inputs to relevant SQL/ Pandas queries
- Leverage Large Language Models (LLMs) for code generation

Motivation

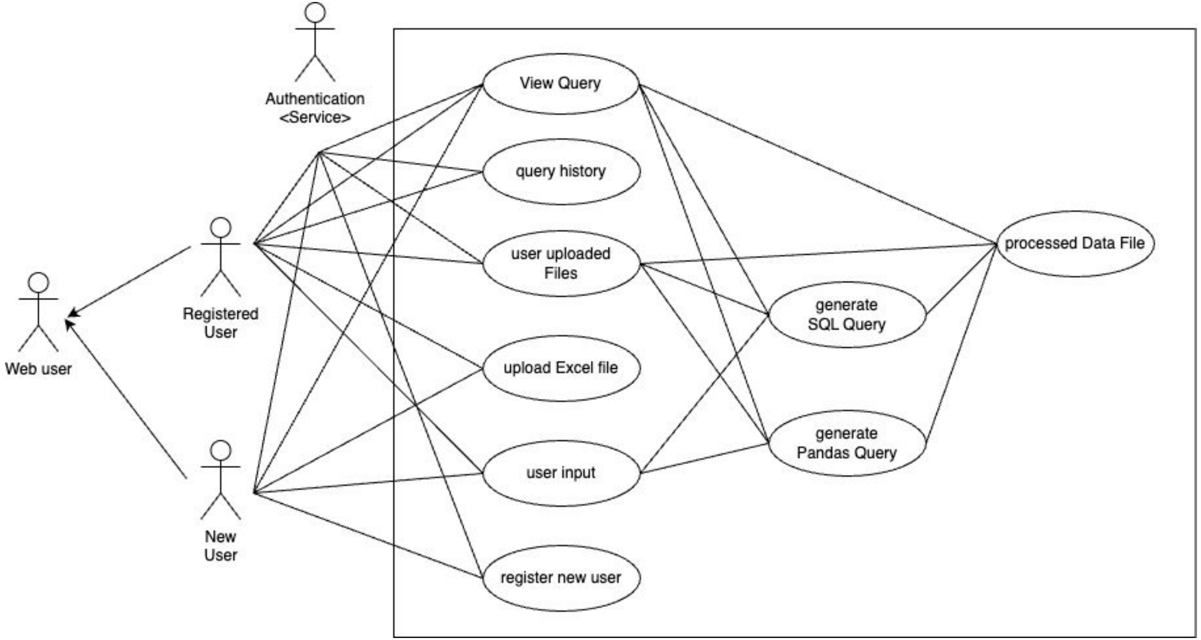
1. **Non-technical users**
   Enable those without coding knowledge to perform complex data analysis tasks easily.

2. **Data analysts**
   Help data analysts streamline their workflow by generating SQL/Pandas queries.

3. **Efficient querying**
   Generate optimized queries to handle large datasets more effectively.

4. **Time-saving tool**
   Reduce the time spent on repetitive query writing, especially for common data operations.

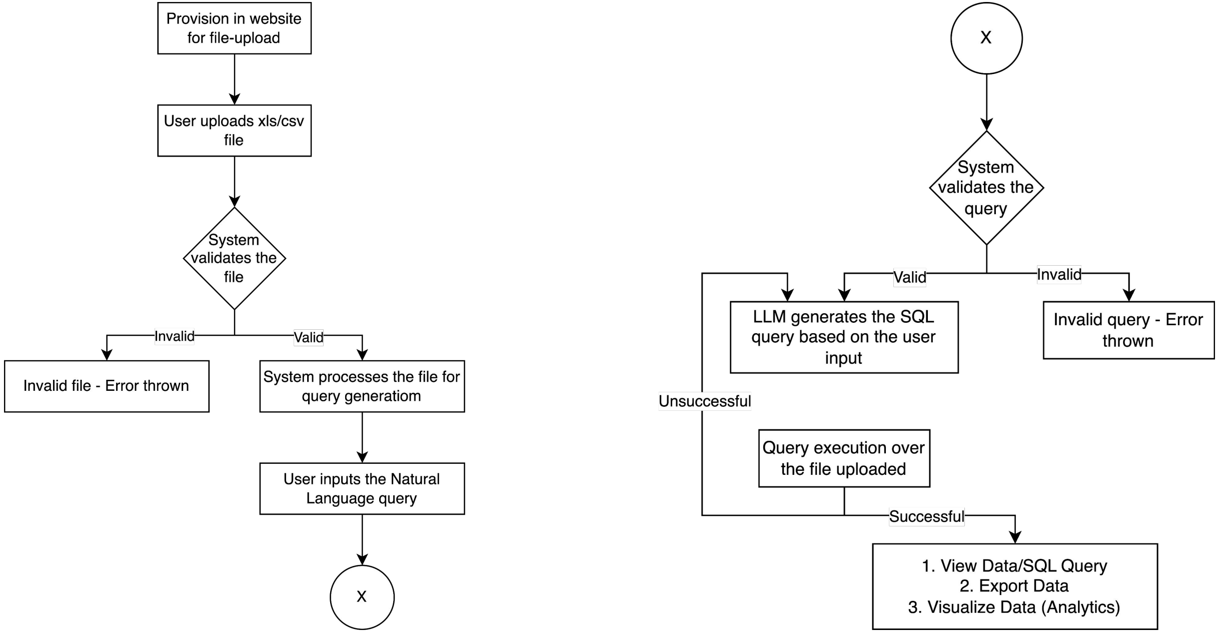
Use Case Diagram



High-Level Features

1. **Natural Language Interface**: Users can input data and queries in natural language.
2. **Query Generation:** Generate both SQL and Pandas queries based on user input.
3. **Data Processing:** Automatically process Excel or CSV files based on the generated queries.
4. **Cloud Deployment:** Use AWS/GCP for scalable cloud-based deployment and query execution.
5. **User Spaces:** Allow users to save preferences or files securely.
6. **Results Visualization:** Display/ Download processed data in a clean and user-friendly manner.

Flow of Execution



Provision in website for file-upload

User uploads xls/csv file

System validates the file

Invalid

Valid

Invalid file - Error thrown

System processes the file for query generatiom

User inputs the Natural Language query

X

X

System validates the query

Valid

Invalid

LLM generates the SQL query based on the user input

Invalid query - Error thrown

Unsuccessful

Query execution over the file uploaded

Successful

1. View Data/SQL Query
2. Export Data
3. Visualize Data (Analytics)

Use Cases

1. Upload Data:

- User uploads an Excel file via the web interface.
- The system checks the file’s format (Excel/CSV).
- If the format is valid, the system processes the file, else it requests a correct file from the user.

2. Enter Natural Language Query:

- **User Inputs Query in Natural Language**: The user types a query or analysis request using everyday language
- **System Interprets the Query**: The system leverages a large language model (LLM) to understand the user's input and translate it into a technical query format (like SQL or Pandas).
- **Prepare for Query Generation**: Once interpreted, the system gets ready to generate the corresponding query that will operate on the uploaded data.

Use Cases

3. Generate and Execute Query:

- We prompt the LLM with user input and a few rows from the data.
- LLM generates the SQL/Pandas query and check for vulnerabilities and executes it on the uploaded data if it is safe.

4. Visualize Results:

- The user can view and copy (to clipboard) the LLM-generated query.
- The processed data will be displayed onto the screen so that the user can see the results.
- User can also create visualizations on the data like line charts, pie charts, etc. by selecting appropriate columns and visualization options.

5. Export Results:

- The user can export and download the processed data in multiple formats (xlsx, csv, etc.)
- They can also download any visualization charts that they have created on the data.

Technologies Proposed

We plan to use the following technologies to build and optimize our project. These tools could be improvised as need during the development process:

1. **Frontend:** React or Angular for developing an interactive and user-friendly UI
2. **Backend:** Flask or FastAPI, Integrating LLMs and LangChain for Natural Language processing and query generation
3. **DataBase:** SQLite or PostgreSQL for local query execution and testing. AWS DynamoDB for scalable cloud data storage
4. **Development:** Docker for containerization and environment management. AWS or GCP for scalable and reliable cloud deployment

Thank You!!