1. CHATDB FOR LARGE-SCALE ENTERTAINMENT DATASETS

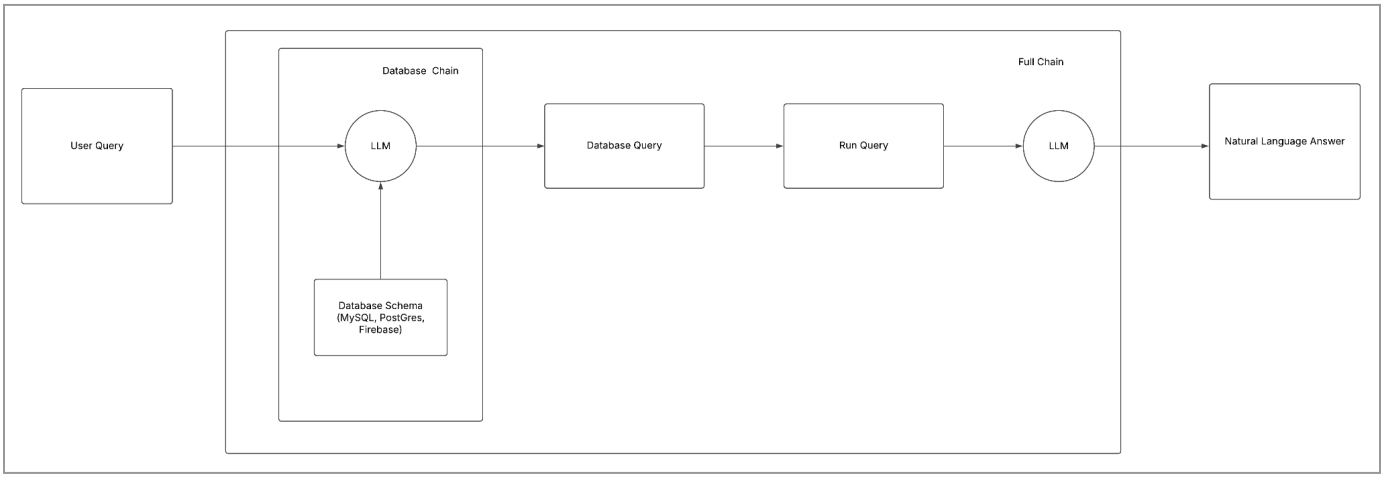
2. Team Details

- Quynh Tran: Background in Business,
    - My name is Quynh Tran and my undergraduate is Business Administration. I have experience in managing and making business decisions based on statistics. I just transferred to Data Science and am doing my master's in person at USC.
- Kevin Bui:
    - My name is Kevin Bui, and I am currently a DEN Student majoring in Computer Science Data Science for my Master of Science. I work full-time as an Application Support Specialist at Center for Creative Leadership. This is a fully remote position, and I work heavily within SQL Server to maintain live data in many different applications and API's. This involves performing data fixes, creating/executing stored procedures, and creating reports for data analysis. I studied at Azusa Pacific University with a Bachelor of Science in Computer Science. I am comfortable with Python, Java, SQL, CSS, PHP, React, and Javascript.
- Riya Berry:
    - My name is Riya Berry, and I am currently an in-person Masters in Applied Data Science student at USC. I completed my BA in Data Science at University of California, Berkeley, and then worked for 2 years as a Data Analyst at a market research consulting firm called Protagonist. I have significant experience in building and developing ML systems, NLP, statistical modeling, and data mining in Python/R/SQL. I am also currently conducting Computer Science Research for the Norman Lear Center at USC Annenberg. As a researcher, I am developing a web-interface platform that allows users to search for key terms within a large-scale database of film and television scripts. Through this role, I have gained increased exposure to front and back-end development, AWS EC2 and ElasticBeanstalk, and distributed database management.

3. Implementation Questions:

- **Tech Stack Used:**
    - To create a Natural Language Interface (NLI), we used HuggingFace (OpenAI 3.5) and LangChain, which allows us to develop NLI applications using LLMs
    - For our databases, we used MySQL and PostgreSQL along with Firebase (No SQL database)
    - To create a front-end interface for users to interact with ChatDB, we used Streamlit
    - Additionally, much of our code relies on Python – particularly Pandas, Numpy, JSON, pymysql, SQLalchemy
- **Query Syntax Implementation Plan:** Describe your approach to converting natural language into database queries.

- Our approach will be to take the user query, which is in natural language, and the database schema, then process these two pieces of information through the LLM to produce a database query. We will extract and interpret the user query using OpenAI's API.
- We will take that database query and run the query against the database to either explore, query, or modify data in the database. To do so, we will integrate LangChain with our database schemas (MySQL, PostgreSQL, Firebase)
- We will then take the results of the database query and process it through the LLM again to produce a natural language answer that is interpretable to human users.



- **Database Selection:** List the databases you are using for your project
  - MySQL - basic SQL operations, lightweight, and free.
  - PostgreSQL - additional SQL database and open source.
  - Firebase - no SQL database and easy API calls.

4. Planned Implementation:

- Our approach started with setting up the necessary tools for code and Note sharing: Google Colab Notebooks, Github Repositories, and a shared Google Drive
- We initially wanted to analyze a large scale dataset related to entertainment and media
  - Thus, we decided to work with a Kaggle dataset on the "Most Watched" tv shows and movies in 2024
- For our database schema, we decided to use PostgreSQL and MySQL as our SQL databases
  - In our original proposal, we were considering multiple NoSQL databases – including MongoDB, Amazon DynamoDB, and Cassandra
  - We deviated from our original proposal and decided to use Firebase because of our exposure to this schema in DSCI 551. Additionally, Firebase has easy integration with Python through Firebase REST and Requests.
- We decided to configure our NLI using Hugging Face's OpenAI 3.5

    - This represents a deviation from our original proposal, as we wanted to use Mistral API via HuggingFace
    - However, we decided to use OpenAI because of its credibility and widespread usage as an LLM. Additionally, we opted for Open AI 3.5 version, which allows for over 16K context windows at a reasonable price.
- Our approach is to clean our “most-watched dataset” for easy database integration, then integrate it with our 3 SQL/NoSQL databases
- We are using Streamlit to create a front-end user interface where users and ask questions about our database
- Lastly, we are integrating LangChain with our database schemas and using HuggingFace (Open AI 3.5) to create our NLI

5. Project Status:

- We have set up the environment and all essential tools
- Then, we cleaned our “most-watched” dataset
    - To do so, we one-hot encoded categorical columns, removed duplicates and null values, and removed unnecessary data from the dataset.
    - Then, we converted the dataset from a CSV to JSON for easy integration with our SQL databases
    - Additionally, we uploaded our JSON dataset to be stored in Firebase with a unique key for easy indexing
- To create our NLI, we used LangChain with our database schemas in Python
    - Currently, we have developed a proof of concept using the “world-db” SQL database that Professor demoed in class.
    - Using Streamlit, we set up a front-end interface that allows users to input any message to explore/query/modify the world database. LangChain is used to convert the user input to SQL queries, and then the queries are executed on the MySQL database.
    - Then, we take the result of our database query and re-process it through using Hugging Face (OpenAI 3.5) to produce a natural language answer
- Our next step is to ensure our systems work with our “most watched tvs/movies” dataset, and incorporate the other database schemas
    - We also need to incorporate our backend API integration
    - Lastly, we need to finalize our NoSQL set up to ensure the NLI works on the Firebase database

6. Challenges Faced:

- We were challenged with how to connect our databases, NLI, and web interface into one application
    - We resolved this challenge by conducting significant research. We discovered that LangChain could help us address this problem because it provides text-to-SQL

translation, and then executes these queries onto our database. We also discovered that Streamlit allows us to easily transform Python scripts into user-friendly web applications.

- Cleaning the database
    - In converting our dataset to JSON, we wanted to use the “title” as a key – or unique identifier
    - However, we realized that the dataset stores multiple entries for each TV show – we believe one for each tv show season
        - As a result, the “title” is not a unique identifier
    - Thus, we are working to find a way to consolidate/summarize the data so there is only one entry for a given tv show while maintaining accuracy

7. Timeline:

Task #	Task Name	Assigned Team Member(s)- Team Responsibility	Start Date	End Date	Status	Notes - Planned Implementation
1	Project Kickoff & Planning	Everyone	2/06	2/12	C…	☑ Initial brainstorming, project scope finalized ☑ Research plan on implementing ChatDB ☑ Identify database source ☑ Set up Git ☑ Set up all the tools we need and do research how we are going to connect and use them
2	Submitting Proposal	Riya…	Feb …	Date	C…	☑ **Completed**
2	Dataset Preparation	Riya		2/12	C…	☑ Data cleaning using Pandas, export to CSV & JSON ☑ Convert to JSON for NoSQL
3	SQL & NoSQL Database Implementation	Riya, Rachel, Kevin		2/17	C…	☑ Firebase set up (Kevin) ☑ MySQL server set up (Riya and Rachel)
6	Backend Development	Kevin & Rachel		3/5	C…	☑ Develop API endpoints for SQL/NoSQL queries ☑ Set up MySQL server, test queries ☑ Integrate Hugging face- open ai 3.5 ☑ Firebase setup
8	Tools Setup	Kevin &		3/12	In…	☐ Set up ChatDB with LangChain

	Check+ Checking NLI	Riya				☐ Team Members need to make sure they can run it on local machine: ☐ Langchain, OpenAI, streamlite and Langchain to run it ☐ Ensure chatDB works with our “most watched tv shows/movies” dataset ☐ Build UI for query submission & visualization ☐ Riya - Further dataset cleaning in Python
9	Testing and Debugging	Everyone		04/01	N…	☐ Unit testing for query handling ☐ Prepare for the in-class Demo ☐ Debugging of edge cases
14	DEMO DAY	Everyone	D…	Apr 21, …	N…	☐ Live in-class demo (4/21 and 4/23, 10 points) ☐ Demonstrate usability of ChatDB, and how users can easily explore/query/modify data about the “most watched” tv shows and datasets
	FINAL TOUCH_UP	Everyone	D…	Apr 24, …	N…	☐ We should touch up on Final Report before the MAY 09 final deadline
15	FINAL REPORT	Everyone	Ma…	May 9, …	N…	☐ Compiling our design, implementation, and learning experiences into one comprehensive report