Spring 2024 Capstone Project: Final Report

May 7, 2024

Model Metadata Chatbot with GenAI

Members

Tianyue Cao (tc3334), Lu Liu (ll3721), Wanxin Luo (wl2930), Xinwei Qiao (xq2236), Yao Xie (yx2845) (In alphabetical order)

Mentors

Industry Mentors:
Sydney Son, KPMG US
Thomas Covella, KPMG US
Chengwei Wang, KPMG US

Data Science Institute Mentor:
Sining Chen, Professor, Columbia University

Contents

1 Introduction

1.1 Project Overview and Scope

The rapid evolution and integration of artificial intelligence and machine learning models into the core operations of businesses have brought about unprecedented efficiencies and innovations. However, this integration has also introduced complexities in managing the vast and interconnected landscape of these models. As models become more intertwined, with significant upstream and downstream impacts, the challenge of effectively managing model-related risks and ensuring comprehensive observability over these models intensifies. Recognizing the critical nature of these challenges, our capstone project, mentored by industry professionals from KPMG, aims to tackle these issues head-on by focusing on the management of model metadata.

This project sets out to develop a cutting-edge solution that leverages generative AI (genAI) technology to create a chatbot designed for seamless integration into day-to-day business operations. The core objective of this initiative is to enhance model risk management and increase the observability of models through the effective management of model metadata. By utilizing a back-end graph database to store the relational aspects of model interconnectedness, our solution aims to provide businesses with the ability to manage the complex landscape of their models more efficiently.

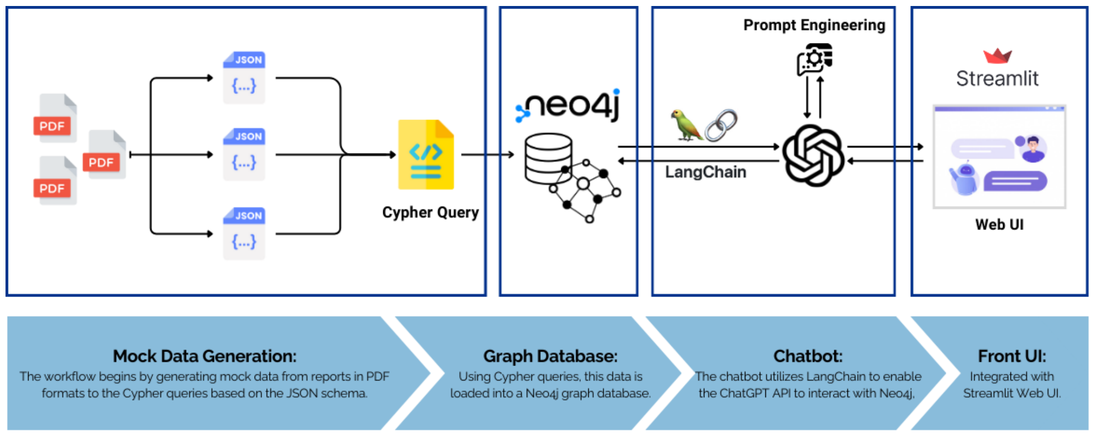
The final report is organized in the following structure. Section 2 elaborates on the approach to developing our chatbot, including both the technical development flow and user flow. This section provides a comprehensive overview of the steps taken from the initial planning stages through to the final implementation, highlighting key methodologies and tools used. Section 3 introduces the graph database, detailing why we chose Neo4j as our back-end database. It also covers the entire mock data generation process and provides an overview of the generated data, explaining how it supports our chatbot's functionalities. Section 4 delves into the underlying architecture of our chatbot, describing in detail how we iterated from the first base version to the final version. This section also revisits the graph database to explain the holistic interaction flow, illustrating how data is managed and utilized within the system. Section 5 introduces the layout of the user interface of our chatbot, explaining design choices and user experience considerations. Section 6 presents an in-depth analysis of the performance of our model, including metrics and benchmarks that demonstrate its effectiveness and efficiency. Section 7 discusses the potential applicable business scenarios where our chatbot can be beneficial, providing examples and case studies to illustrate its practical uses. Section 8 addresses the ethical concerns considered in this project, with a focus on AI hallucination and the implementation of guardrails to prevent such issues. This section outlines the ethical framework and safeguards we have put in place to ensure responsible AI use. Finally, Section 9 and Section 10 conclude our project and outline the steps for future improvement, reflecting on lessons learned and proposing enhancements for further development.

2 Workflow

The development and operation of our GenAI-powered chatbot for KPMG are guided by a meticulously structured workflow encompassing two distinct but interrelated parts: the

Technical Flow and the User Flow. They play a vital role in ensuring that the chatbot functions efficiently and meets the complex needs of our users.

2.1 Technical Flow



Mock Data Generation: The workflow begins by generating mock data from reports in PDF formats to the Cypher queries based on the JSON schema.

Graph Database: Using Cypher queries, this data is loaded into a Neo4j graph database.

Chatbot: The chatbot utilizes LangChain to enable the ChatGPT API to interact with Neo4j.

Front UI: Integrated with Streamlit Web UI.

Figure 1: Technical Flow

The Technical Flow is the backbone of our chatbot, involving a series of steps that transform raw data from static documents into dynamic, accessible information through our GenAI-powered chatbot. Here, we describe each step in detail:

Step 1: Mock Data Generation:
The journey begins with generating mock data from real KPMG reports available in PDF format. Based on the JSON schema file provided by KPMG, We convert these documents into Cypher queries tailored specifically to capture the nuances and complexity of the original content. This initial step is crucial for setting a robust foundation for the subsequent stages of the chatbot's data-handling capabilities.

Step 2: Graph Database Integration:
The structured data in the form of Cypher queries is then integrated into Neo4j. This database's graph-based nature allows it to model complex data relationships intuitively and efficiently, facilitating rapid data retrieval and sophisticated querying capabilities crucial for the responsive functioning of our chatbot.

Step 3: Chatbot Development
At this stage, we utilize LangChain to bridge our structured Neo4j database with the advanced conversational AI capabilities of ChatGPT API. This connection enables the chatbot to intelligently access and interpret the database content, using natural language processing to understand and effectively respond to user queries. Meanwhile, an integral part of refining the chatbot's functionality is prompt engineering. This involves designing and optimizing the prompts that guide ChatGPT's responses to ensure they are accurate and contextually relevant. Through iterative improvements, we enhance the chatbot's ability to deliver precise and useful information, thereby improving the quality of user interaction.

Step 4: Streamlit Deployment

The final step in our technical flow is implementing the Streamlit Web UI, which serves as the chatbot's front-end interface. Streamlit provides a user-friendly platform where users can interact with the chatbot seamlessly. It is here that users can pose questions and receive dynamically generated answers, effectively closing the loop on our sophisticated backend processes.

Through these 4 main steps, our technical flow not only supports the operational requirements of our chatbot but also ensures a user-centric design that is accessible, intuitive, and efficient. This systematic approach underscores our commitment to delivering a high-quality, reliable tool for data-driven insights and decision-making support at KPMG.

2.2 User Flow

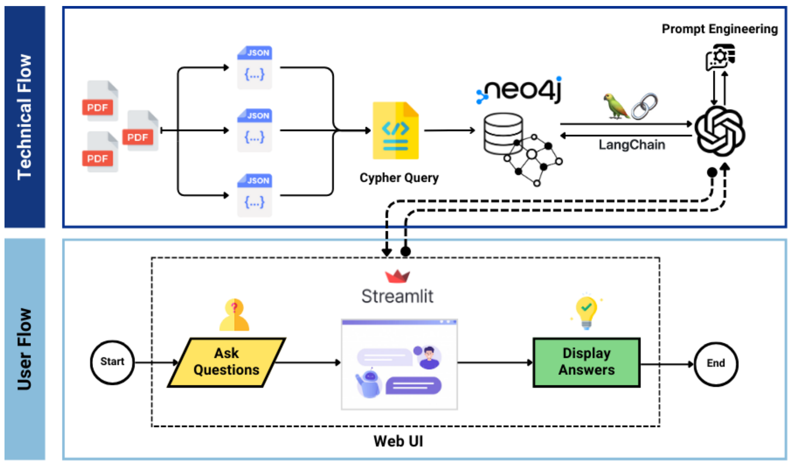


Figure 2: User Flow

The User Flow for our chatbot system is centered around the Streamlit Web UI, where users begin their interaction by posing queries. This front-end interface is designed for simplicity and intuitive use, allowing users to focus on their questions which activate the complex processes behind the scenes.

Upon query submission, our chatbot leverages the ChatGPT API to parse and understand the user's input, assessing the context and intent of the question. This understanding is transformed into dynamically generated Cypher queries that interact with the Neo4j graph database, retrieving data that accurately matches the query's context.

The chatbot then processes this data, not just retrieving information but also understanding the intricate relationships within the database to formulate a coherent and contextually relevant response. This response is finely tuned to provide clear and actionable insights, which are displayed back to the user through the Streamlit interface.

This process not only ensures that the responses are informative and precise but also enhances the user's engagement by facilitating a dynamic exploration of data.

This user-centric flow is designed to integrate sophisticated back-end data processing with an easy-to-navigate front-end, providing a seamless and impactful user experience.

3 Graph Database

3.1 The Use of Neo4j and Cypher

In traditional database systems, data is typically stored in tables composed of rows and columns. These systems excel in handling structured data but often struggle with complex relationships and interconnections. In contrast, graph databases represent data as a collection of nodes and edges. Nodes represent entities, while edges represent the relationships between these entities. This structure is designed to facilitate the understanding and querying of rich interconnections, making it suitable for datasets where relationships are key to the data's structure and meaning.

We selected Neo4j for our project because it is a robust graph database management system that excels in handling connected data. Neo4j stores connections between data points as first-class entities, making it more efficient at querying complex relationships compared to relational databases. Meanwhile, Cypher, as the SQL for graph databases, allows developers to interact with the graph database using a simple syntax. It supports creating, querying, and modifying data within the graph structure in an efficient manner.

The use of Neo4j in our project was driven by the need to efficiently manage and query metadata and model complex data flows. Managing metadata for diverse business entities within Neo4j allowed us to represent and query data about data effectively. This was essential for maintaining accuracy and consistency across datasets. Using Cypher, we can trace data flows from high-level analytical reports down to the specific database tables that feed information into machine learning models.

3.2 Mock Data Generation

At the outset of our capstone project, our team was provided with multiple schema JSON files from KPMG. These files contained comprehensive descriptions of various report fields, the lineage of data models, and the hierarchical structure of data sources. These schemas served as the foundation for constructing a mock version of KPMG's graph database tailored to our project's requirements.

For practical and demonstrative purposes, we selected two publicly referenced reports from KPMG: the 'Global Economic Outlook - December 2023' and the 'Saudi Arabia Budget Report 2024'. These reports were used to write Cypher queries that generated the mock graph database depicted in our project visuals.

Due to the restricted access to KPMG's proprietary reports and datasets, our team created mock data that closely resembles the structure and complexity of KPMG's actual metadata. This approach was necessary to ensure the chatbot could simulate interactions with a graph database resembling KPMG's internal configurations.

The mock data was structured to allow the demonstration of the graph database capabilities without compromising sensitive corporate data. This setup provides a scalable model that KPMG could potentially expand upon, incorporating thousands of nodes and edges reflecting detailed internal reports and metadata structures.

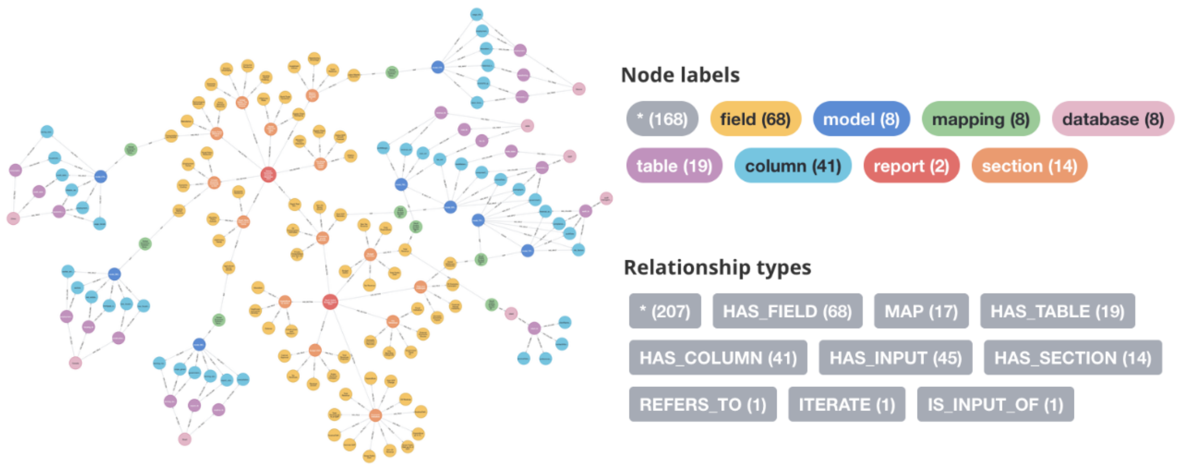


Figure 3: Visualization of Mock Graph Data

Figure 3 illustrates the complex structure of the mock graph database developed for our project. This visualization showcases the various node types and their interrelationships within the database, emphasizing the interconnected nature of the data elements. The graph contains several distinct types of nodes, each representing different elements within our data model. Notably, there are a total of 168 nodes and 207 edges. These nodes represent the granular components of reports, database schemas, and data processing models used within the project. Also, these nodes are interconnected by various types of relationships that signify the flow and structure of data.

3.3 Structure of Graph Database

The Figure 4 below visually represents the structure of our graph database, illustrating how different data elements are interconnected through specific node types and relationships.

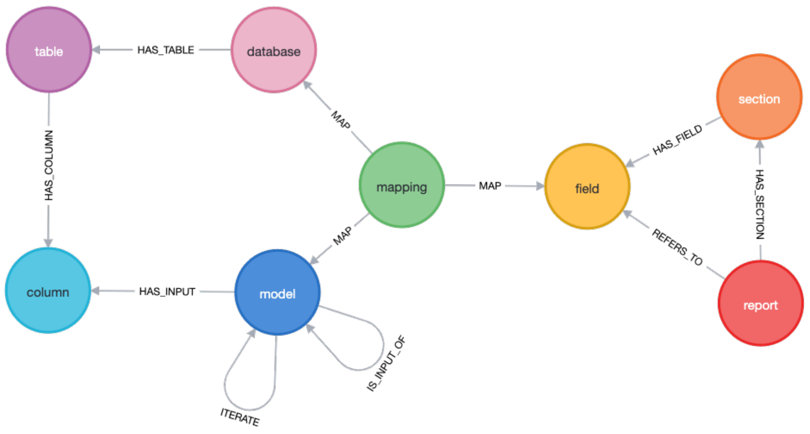


Figure 4: Graph Data Schema

- **Report Node:** This node acts as the entry point for data related to individual reports. Each report is broken down into several sections, represented by edges connecting the report node to multiple section nodes. Report nodes contain properties: name, entitlements, report format, author, and business group.
- **Section Node:** Section nodes categorize data within reports into distinct sections. Each section is further detailed through fields, showing how report data is structured at a granular level.
- **Field Node:** Fields represent the smallest unit of documentation in our database. Fields are crucial for detailed data analysis and are linked to specific models through mapping nodes.
- **Model Node:** Model nodes represent the analytical models that process data. These nodes often have multiple input columns. Model nodes contain properties: creator, model type, input columns, performance metric, versions, and creation date.
- **Column Node:** Columns are fundamental units within tables and are directly used by models for data processing.
- **Table Node:** Tables organize columns and are part of a larger database.
- **Database Node:** Database nodes encapsulate the entire data storage framework, linking to tables that reside within the database. This node illustrates the top-level data aggregation point.

All of the nodes, edges, and properties detailed above form the foundational components of our mock graph database. Each node type plays a specific role in modeling the complex relationships and structures of the data, mirroring real-world business scenarios. The edges connect these nodes, illustrating the relationships and data flow within our system.

4 Chatbot Logistics

4.1 Overview of the Final Chatbot Logic

To effectively manage model metadata and optimize user interactions, we have implemented a sophisticated chatbot architecture leveraging Generative AI tools through the GPT API and Langchain. This architecture is designed to address a wide range of queries encompassing data, models, and reports while significantly enhancing the overall user experience. Illustrated in Figure 5, the system architecture and functionalities are detailed, centering around a robust classification layer. This layer categorizes incoming user queries, directing them to the appropriate processing tools based on the nature of the inquiry. This ensures efficient and accurate responses tailored to the specific needs and contexts of the user requests. This technical strategy not only enables a seamless interaction model but also establishes a dynamic framework that can adapt to emerging AI capabilities and evolving user needs.

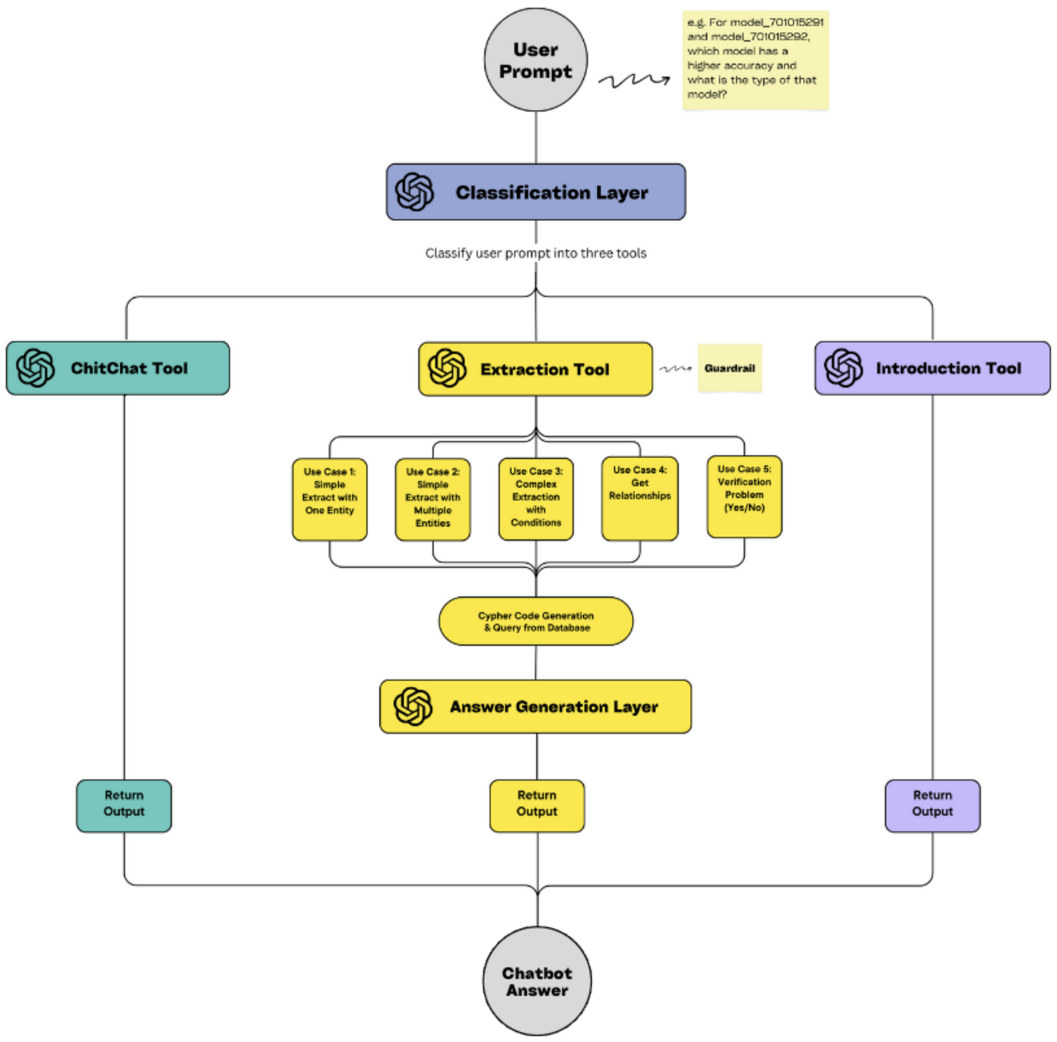

User Prompt
e.g. For model_701015291 and model_701015292, which model has a higher accuracy and what is the type of that model?
Classification Layer
Classify user prompt into three tools
ChitChat Tool
Extraction Tool
Guardrail
Introduction Tool
Use Case 1: Simple Extract with One Entity
Use Case 2: Simple Extract with Multiple Entities
Use Case 3: Complex Extraction with Conditions
Use Case 4: Get Relationships
Use Case 5: Verification Problem (Yes/No)
Cypher Code Generation & Query from Database
Answer Generation Layer
Return Output
Return Output
Return Output
Chatbot Answer

Figure 5: Final Chatbot Model Architecture

At the heart of our chatbot framework is the **Classification Layer**, which is pivotal in analyzing and routing user prompts to the appropriate service tool. This layer is powered by the GPT API, which ensures high accuracy and efficiency in handling various types of user interactions. The Classification Layer categorizes prompts into one of three primary tools:

1. **ChitChat Tool**: This tool handles general inquiries and casual interactions (e.g., ”Can you recommend some restaurants in New York?”).
2. **Introduction Tool**: Designed for new users or those needing assistance with chatbot functionalities, this tool provides guided instructions and answers to procedural questions (e.g., ”How do I use this chatbot?”).
3. **Extraction Tool**: This is the most complex of the three, handling sophisticated data retrieval tasks. It is integrated with our Neo4j graph database, allowing for intricate queries involving multiple entities and relationships, essential for extracting detailed reports, data, and model-related information.

Once the classification layer routes the prompt to the appropriate tool, the selected tool processes the query and generates a response. This response mechanism is designed to be seamless and efficient, ensuring that users receive timely and accurate information. With the chatbot's advanced classification system and integration with the GPT API, the chatbot is well-equipped to handle a wide range of user interactions, from simple chitchat to complex data extraction tasks.

4.2 Our Core: The Extraction Tool

The Extraction Tool is a cornerstone component of our chatbot, engineered to manage a wide range of queries, from straightforward data retrievals to intricate analytical tasks. Figure 6 provides a detailed examination of the architecture of the Extraction Tool. This robust framework not only promotes efficient data extraction from the database but also ensures the tool's adaptability and precision in responding to a diverse array of user inquiries.

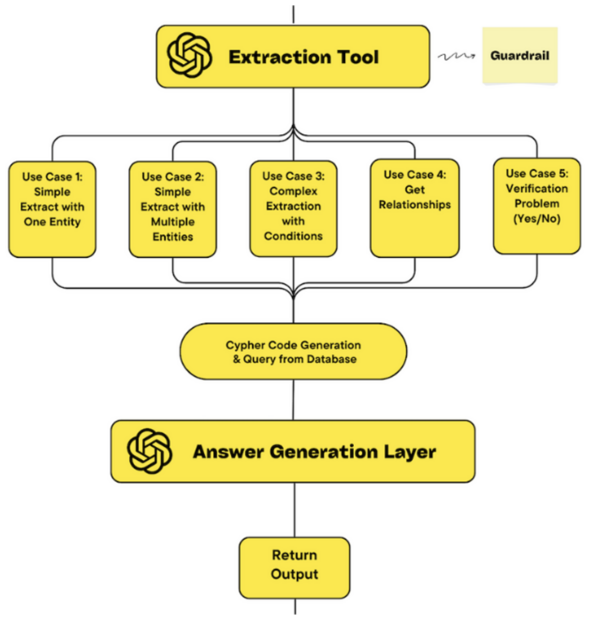

Extraction Tool
Guardrail
Use Case 1: Simple Extract with One Entity
Use Case 2: Simple Extract with Multiple Entities
Use Case 3: Complex Extraction with Conditions
Use Case 4: Get Relationships
Use Case 5: Verification Problem (Yes/No)
Cypher Code Generation & Query from Database
Answer Generation Layer
Return Output

Figure 6: Extraction Tool Architecture

The tool is rigorously trained using a comprehensive dataset comprising 20 distinct sets of user prompts and their corresponding Cypher codes. These are organized into five use cases, each tailored to effectively address specific types of queries:

1. **Use Case 1: Simple Extraction with One Entity**

   e.g., Who is the author of this report?

2. **Use Case 2: Simple Extraction with Multiple Entities**

   e.g., What are the output columns of model1, model2, and model3?

3. **Use Case 3: Complex Extraction with Conditions**

   e.g., How many models in the 'Consumption and Growth' field have an R-squared higher than 0.8?

4. **Use Case 4: Get Relationships**

   e.g., What report does model1 depend on?

5. **Use Case 5: Verification Problem (Yes/No)**
   e.g., Does model1 predict GDP?

The Cypher Generation Phase is a critical component of the Extraction Tool. It utilizes tailored training sets and the GPT API to generate accurate Cypher queries based on user prompts. This capability ensures that even unseen queries are efficiently handled, with the system dynamically generating the necessary Cypher code to retrieve the correct information from the database.

Following data extraction, the Answer Generation Phase converts the raw data into coherent, human-readable responses. This phase is crucial for ensuring that the information is presented in a format that is easily understandable by users.

To maintain the reliability of the Extraction Tool and prevent AI hallucinations, guardrails have been implemented. For instance, if the chatbot fails to retrieve the necessary information, it is programmed to prompt the user for more detail by responding, ”I don't know the answer. Could you provide more information?” This approach ensures that the responses remain accurate and trustworthy.

By systematically categorizing queries into structured use cases and employing sophisticated Cypher code generation, the Extraction Tool ensures high accuracy and adaptability in handling diverse user queries. With built-in guardrails to maintain reliability, the chatbot stands as a robust solution for data retrieval needs across various domains.

4.3 Structural Iterations

Over the course of the semester, our chatbot has undergone significant evolution. Here, we present a comprehensive overview of the developmental trajectory of our chatbot's logic architecture, mapped through three distinct phases: baseline, intermediate, and final models. Each phase represents a crucial stage in addressing key challenges and integrating advanced functionalities to refine the system's accuracy and user engagement. We detail the iterative process that led to the robust classification and extraction mechanisms, which are now integral to the chatbot's operation, in Figure 7.

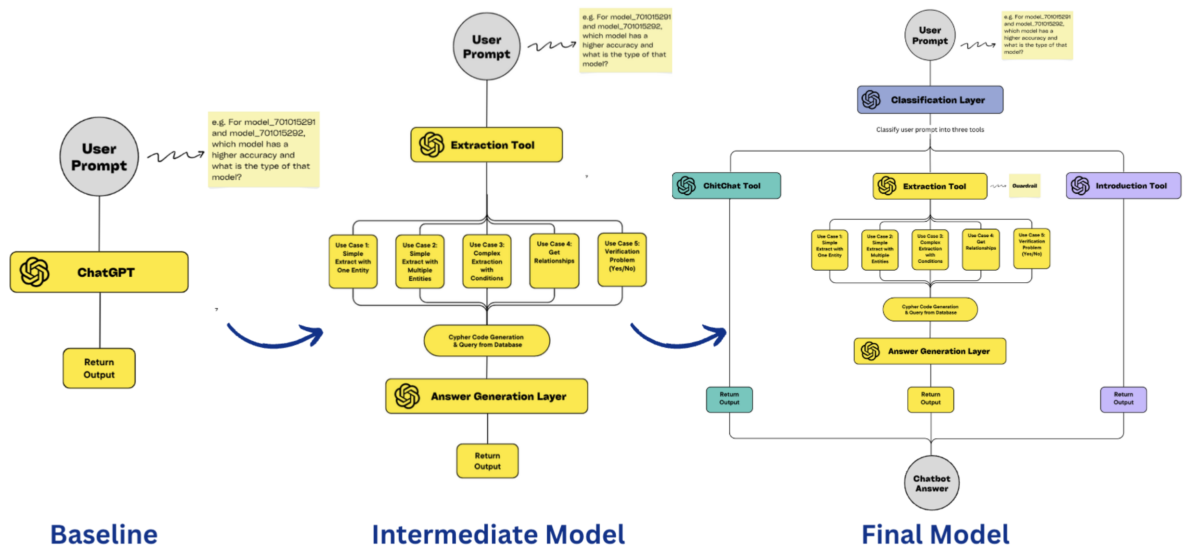

User Prompt
e.g. For model_701015291 and model_701015292, which model has a higher accuracy and what is the type of that model?
ChatGPT
Return Output
Baseline
User Prompt
e.g. For model_701015291 and model_701015292, which model has a higher accuracy and what is the type of that model?
Extraction Tool
Use Case 1: Simple Extract with One Entity
Use Case 2: Simple Extract with Multiple Entities
Use Case 3: Complex Extraction with Conditions
Use Case 4: Get Relationships
Use Case 5: Verification Problem (Yes/No)
Cypher Code Generation & Query from Database
Answer Generation Layer
Return Output
Intermediate Model
User Prompt
e.g. For model_701015291 and model_701015292, which model has a higher accuracy and what is the type of that model?
Classification Layer
Classify user prompt into three tools
ChitChat Tool
Extraction Tool
Guardrail
Introduction Tool
Use Case 1: Simple Extract with One Entity
Use Case 2: Simple Extract with Multiple Entities
Use Case 3: Complex Extraction with Conditions
Use Case 4: Get Relationships
Use Case 5: Verification Problem (Yes/No)
Cypher Code Generation & Query from Database
Answer Generation Layer
Return Output
Return Output
Return Output
Chatbot Answer
Final Model

Figure 7: Structural Iterations

Phase 1: Baseline Model
Initially, our chatbot utilized a direct connection between the graph database and the GPT API. This baseline model aimed to provide a straightforward framework for data retrieval. The primary challenge in this phase was AI hallucinations—instances where the chatbot generated incorrect information. This was attributed to the lack of a structured query generation process, leading to errors in data retrieval.

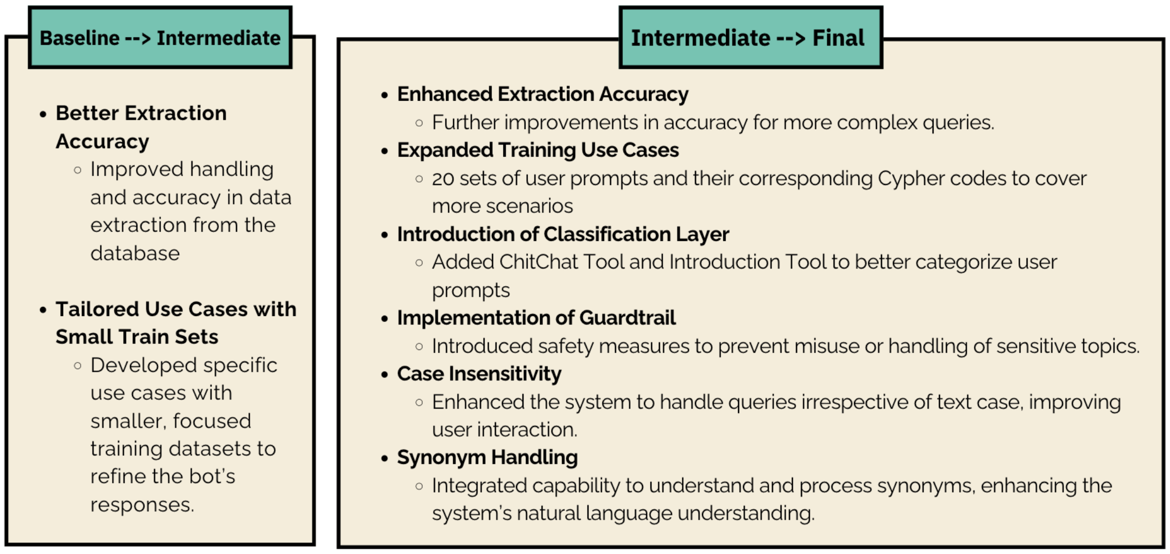
Phase 2: Intermediate Model
To overcome the inaccuracies of the baseline model, we shifted our focus towards refining the generation of Cypher queries. This phase involved extensive prompt engineering, allowing the chatbot to generate accurate Cypher queries through trained user prompts directly correlated with the correct database queries. Despite improvements in query accuracy, a significant gap remained in the form of a limited question bank, which restricted our ability to fully test and adapt the chatbot's capabilities to diverse user interactions.

Phase 3: Final Model
The introduction of beta testing in this phase was pivotal. Feedback from beta testers highlighted the need for a more intuitive understanding of the database and chatbot interactions, which many users initially found challenging. In response to beta feedback, we developed a classification layer that intelligently categorizes user prompts into three distinct tools: ChitChat, Introduction, and Extraction. This layer guides users more effectively based on the nature of their queries. We also enhanced the Extraction Tool by resolving previously identified bugs and refining the tool's ability to handle complex queries. This was critical in elevating the chatbot's performance and reliability.

The evolution of our chatbot from a basic model to a sophisticated interaction system showcases our commitment to continuous improvement and user-centric design. The final model not only addresses earlier challenges but also introduces features that significantly enhance user engagement and satisfaction. Our iterative approach has ensured that each phase builds upon the previous, culminating in a powerful tool that efficiently manages complex queries and provides an intuitive user experience.

4.4 Iteration Improvements

The evolution of our chatbot was marked by significant enhancements in data handling, user interaction, and the introduction of ethical guardrails. Figure 8 details the specific improvements we have made. From the baseline to the intermediate model, we focused on improving data extraction accuracy and developed specific use cases with small, focused training sets. This targeted approach significantly refined the chatbot's responses to user queries. From the intermediate model to the final model, we further enhanced extraction accuracy and expanded our training to encompass 20 different user scenarios. We also introduced a classification layer, incorporating tools like the ChitChat and Introduction Tool to effectively categorize and respond to user prompts. Moreover, we implemented guardrails to ensure ethical interactions, made our system case insensitive, and integrated synonym handling to improve understanding and enrich the user experience.



Baseline --> Intermediate

- **Better Extraction Accuracy**
  - Improved handling and accuracy in data extraction from the database
- **Tailored Use Cases with Small Train Sets**
  - Developed specific use cases with smaller, focused training datasets to refine the bot's responses.

Intermediate --> Final

- **Enhanced Extraction Accuracy**
  - Further improvements in accuracy for more complex queries.
- **Expanded Training Use Cases**
  - 20 sets of user prompts and their corresponding Cypher codes to cover more scenarios
- **Introduction of Classification Layer**
  - Added ChitChat Tool and Introduction Tool to better categorize user prompts
- **Implementation of Guardrail**
  - Introduced safety measures to prevent misuse or handling of sensitive topics.
- **Case Insensitivity**
  - Enhanced the system to handle queries irrespective of text case, improving user interaction.
- **Synonym Handling**
  - Integrated capability to understand and process synonyms, enhancing the system's natural language understanding.

Figure 8: Iteration Improvements Highlight

4.5 Connection to the Graph Database

The integration of a chatbot with the Neo4j graph database represents a strategic enhancement, combining the robust natural language processing capabilities of GPT-3.5 with the sophisticated data management and querying functionality of Neo4j. This section details the implementation of the LangChain and CypherQAChain components that facilitate this integration, optimizing the process of querying structured and unstructured data to provide seamless and efficient user interactions.

The core of our system architecture is the connection between the chatbot, powered by GPT-3.5, and the Neo4j graph database. This connection is mediated through a series of components and processes that translate user inputs into database queries and then convert the structured outputs back into comprehensible natural language responses.

1. **User Interaction**: Users interact with the chatbot through a conversational interface where they can input their queries or prompts.

2. **Query Generation**: Upon receiving a user prompt, the chatbot employs GPT-3.5

to generate a Cypher query. This translation from natural language to Cypher is handled by the CypherQAChain, which is specifically trained to formulate accurate and efficient graph queries.

3. **Database Querying**: The Cypher query is then executed on the Neo4j database, which is designed to handle complex queries with high efficiency, leveraging its graph-based structure.

4. **Results Handling**: The results from Neo4j are retrieved and sent back to the chatbot.

5. **Response Generation**: GPT-3.5 processes these results to generate a coherent natural language response, which is then delivered to the user.

The interaction flow, as detailed in Figure 9, provides a step-by-step breakdown of the process from the initial user prompt to the final delivery of the answer.

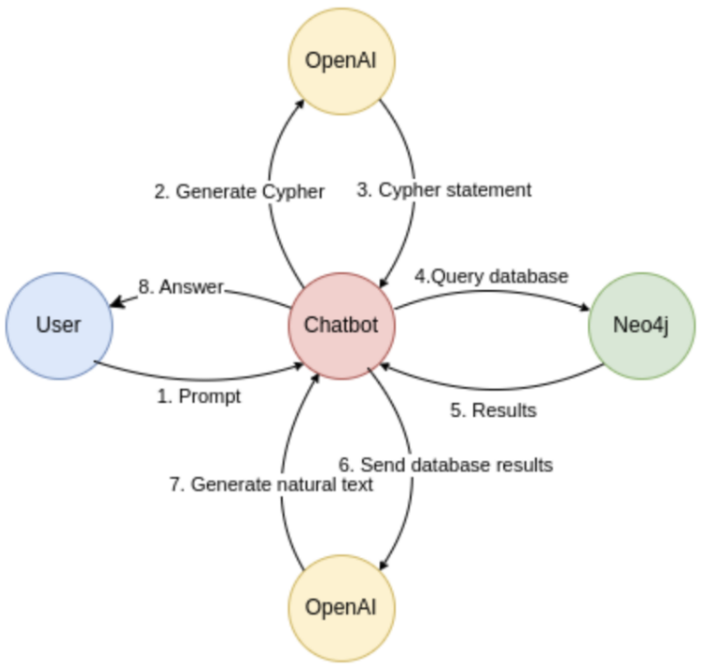
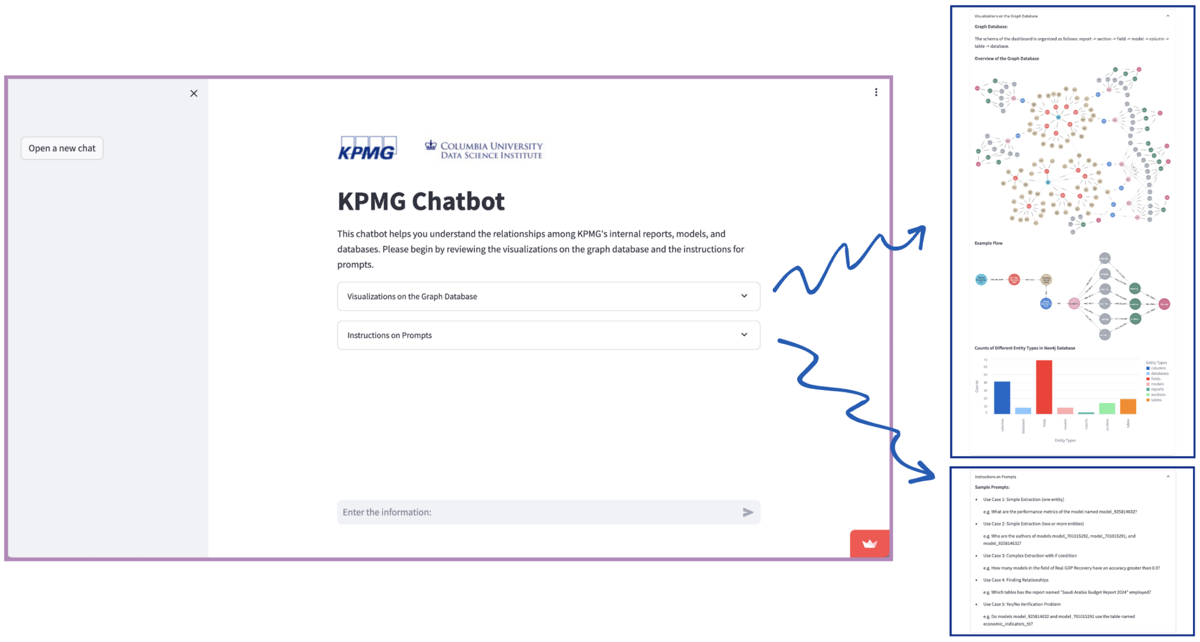


Figure 9: Step-by-step User Interaction Flow [3]

5 User Interface



KPMG Chatbot

This chatbot helps you understand the relationships among KPMG's internal reports, models, and databases. Please begin by reviewing the visualizations on the graph database and the instructions for prompts.

Graph Database:

The schema of the dashboard is organized as follows: report -> section -> field -> model -> column -> table -> database.

Overview of the Graph Database

Sample Prompts:

- Use Case 1: Simple Extraction (one entity)

  e.g. What are the performance metrics of the model named model_925814632?
- Use Case 2: Simple Extraction (two or more entities)

  e.g. Who are the authors of models model_701015292, model_701015291, and model_925814632?
- Use Case 3: Complex Extraction with if condition

  e.g. How many models in the field of Real GDP Recovery have an accuracy greater than 0.9?
- Use Case 4: Finding Relationships

  e.g. Which tables has the report named "Saudi Arabia Budget Report 2024" employed?
- Use Case 5: Yes/No Verification Problem

  e.g. Do models model_925814632 and model_701015291 use the table named economic_indicators_tb?

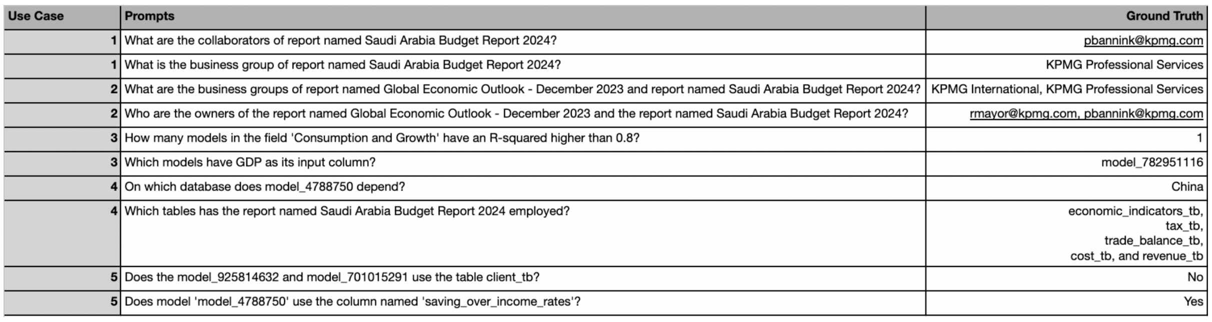
Figure 10: User interface

To enhance user-friendliness and accessibility, the application displays its final output through a front-end interface designed with Streamlit, providing clear and concise responses to user queries. Users have the flexibility to ask any questions about the database. Upon receiving a query, the chatbot not only delivers a successful response but also generates Cypher code. This feature empowers users to independently extract information, enhancing their ability to interact with and navigate through the database effectively.

Regarding the chatbot interfaces, users entering this interface can utilize dropdown menus to visualize the graph database and understand possible prompts or directly ask the chatbot for usage instructions. The first dropdown menu provides users with a comprehensive overview of our database. It visually represents our database schema structures and includes an interactive bar chart that displays the count of different nodes. This setup aids users in becoming acquainted with the layout and content of our database. The second dropdown menu offers detailed information on our five distinct use cases, complemented by sample questions for each. This helps users understand the practical applications of the database and how to interact with it effectively.

6 Performance

In the evaluation phase, we implemented an automatic testing pipeline to examine how closely the answers produced by our model align with the ground truth. Below, we have included a sample test set for various use cases. For each specified use case, there are 20 test cases provided.

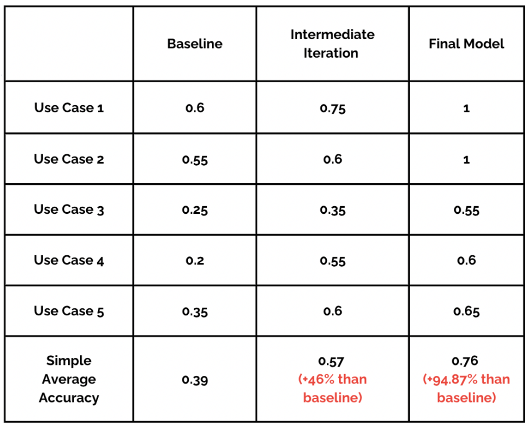


Use Case	Prompts	Ground Truth
1	What are the collaborators of report named Saudi Arabia Budget Report 2024?	pbannink@kpmg.com
1	What is the business group of report named Saudi Arabia Budget Report 2024?	KPMG Professional Services
2	What are the business groups of report named Global Economic Outlook - December 2023 and report named Saudi Arabia Budget Report 2024?	KPMG International, KPMG Professional Services
2	Who are the owners of the report named Global Economic Outlook - December 2023 and the report named Saudi Arabia Budget Report 2024?	rmayor@kpmg.com, pbannink@kpmg.com
3	How many models in the field 'Consumption and Growth' have an R-squared higher than 0.8?	1
3	Which models have GDP as its input column?	model_782951116
4	On which database does model_4788750 depend?	China
4	Which tables has the report named Saudi Arabia Budget Report 2024 employed?	economic_indicators_tb, tax_tb, trade_balance_tb, cost_tb, and revenue_tb
5	Does the model_925814632 and model_701015291 use the table client_tb?	No
5	Does model 'model_4788750' use the column named 'saving_over_income_rates'?	Yes

Figure 11: Sample Test Set for Each Use Case

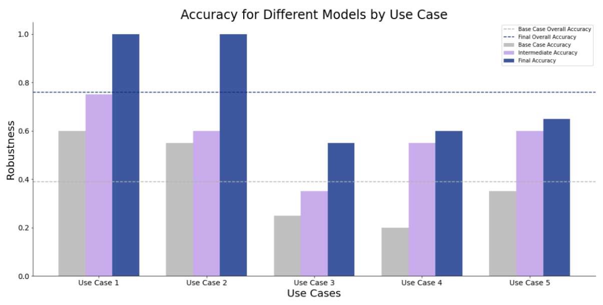
We assessed the performance of our model using two primary metrics: accuracy and robustness. Accuracy evaluates how effectively the model's responses address user queries, while robustness assesses the consistency of the model's answers across different instances, regardless of their correctness.

6.1 Accuracy



	Baseline	Intermediate Iteration	Final Model
Use Case 1	0.6	0.75	1
Use Case 2	0.55	0.6	1
Use Case 3	0.25	0.35	0.55
Use Case 4	0.2	0.55	0.6
Use Case 5	0.35	0.6	0.65
Simple Average Accuracy	0.39	0.57 (+46% than baseline)	0.76 (+94.87% than baseline)

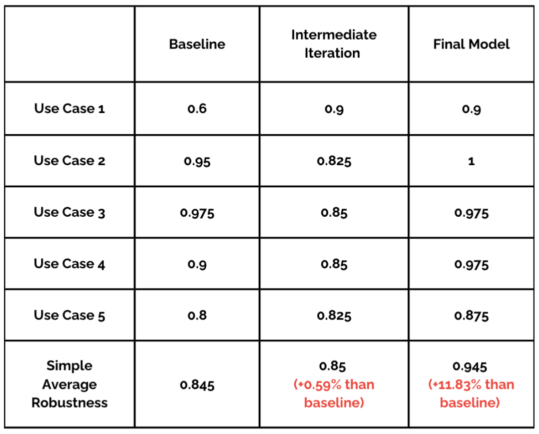
(a) Accuracy Table



(b) Accuracy Plot

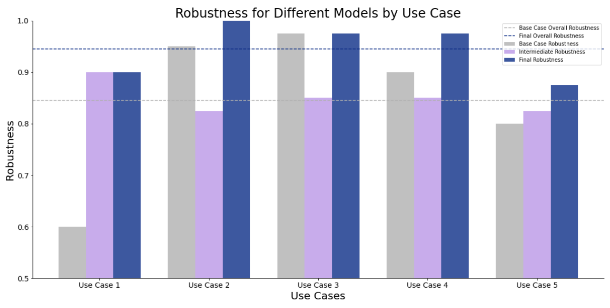
To ensure our testing is free from bias, we execute the pipeline three times and calculate the average score to determine the final accuracy. In general, our final model demonstrates substantial enhancements over the baseline. For the initial use cases 1 or 2, involving the extraction of single or multiple attributes, our final model achieved a perfect accuracy score of 100% based on our designed test cases. For more complex extraction scenarios like use cases 3-5, which involve extraction with conditions, get relationships, and verification problems, the model reached an accuracy of about 60%, indicating areas for future improvement. Our final model achieved an accuracy rate of 76%, marking a 94.87% improvement, which is significant compared to the baseline. This progress underscores the effectiveness of our iteration and optimization efforts.

6.2 Robustness



	Baseline	Intermediate Iteration	Final Model
Use Case 1	0.6	0.9	0.9
Use Case 2	0.95	0.825	1
Use Case 3	0.975	0.85	0.975
Use Case 4	0.9	0.85	0.975
Use Case 5	0.8	0.825	0.875
Simple Average Robustness	0.845	0.85 (+0.59% than baseline)	0.945 (+11.83% than baseline)

(c) Robustness Table



(d) Robustness Plot

We also compare the results from these three runs to assess the model's overall robustness. Regarding robustness, our baseline model was already quite solid, scoring 0.845. However, our final model has surpassed this, providing more stable and reliable responses. It achieved a robustness score of 0.945, representing an 11.83% enhancement over the baseline. This improvement highlights our model's increased dependability in delivering consistent results across various scenarios.

7 Business Case Implementation

The GenAI-powered chatbot is designed to enhance operational efficiency by providing tailored support for different roles within the organization, each contributing uniquely to the management and utilization of model metadata. This targeted approach not only streamlines workflows but also maximizes business value by improving decision-making and reducing operational overhead.

- **New Employees**: By automating responses to common onboarding questions, the chatbot helps new employees integrate more quickly and smoothly into the organization, reducing the time HR spends on routine inquiries and allowing them to focus on strategic activities.
- **Business Analysts**: Analysts have instant access to updated market trends and reports, enabling quicker response to market changes. The chatbot assists in managing metadata related to market data sources, improving the accuracy and relevancy of market insights.
- **Data Scientists**: This group benefits significantly as the chatbot provides instant, detailed insights into model metadata such as performance metrics, types, and target variables. This facilitates quicker iterations and optimizations of machine learning models, enhancing the predictive capabilities of business operations.
- **Data Engineers**: For engineers, the chatbot simplifies the complex task of data lineage tracing by identifying and tracking metadata related to source columns, tables, and databases. This is crucial for ensuring data integrity, compliance, and efficient data management.

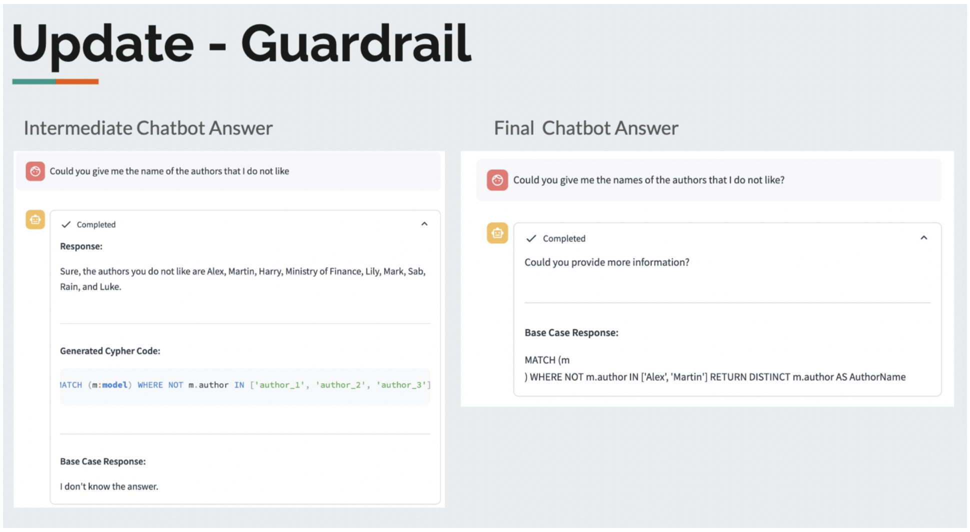
- **Product Managers**: The chatbot supports product managers by providing quick access to metadata about team skills and project statuses. This aids in effective team management and resource allocation, which is crucial for meeting project deadlines and objectives.

For screenshots illustrating the interaction of different roles with our chatbot, please refer to **the Appendix**.

By addressing specific needs through its role-based functionalities, the chatbot not only enhances individual productivity but also drives overall organizational efficiency. This strategic tool serves as a cornerstone for managing model metadata, ensuring that data is leveraged effectively across the company to support informed decision-making and agile responses to market demands.

8 Ethical Considerations

8.1 Tackling AI Hallucinations via Prompt Engineering



Update - Guardrail

Intermediate Chatbot Answer

Could you give me the name of the authors that I do not like

Completed

Response:

Sure, the authors you do not like are Alex, Martin, Harry, Ministry of Finance, Lily, Mark, Sab, Rain, and Luke.

Generated Cypher Code:

```
IATCH (m:model) WHERE NOT m.author IN ['author_1', 'author_2', 'author_3']
```

Base Case Response:

I don't know the answer.

Final Chatbot Answer

Could you give me the names of the authors that I do not like?

Completed

Could you provide more information?

Base Case Response:

```
MATCH (m
) WHERE NOT m.author IN ['Alex', 'Martin'] RETURN DISTINCT m.author AS AuthorName
```

Figure 12: AI Hallucination

Throughout the training process, we encountered numerous instances of AI hallucination in sample cases, such as Figure 12 above. For questions that are subjective or contain explicitly offensive language, the chatbot should respond with 'I don't know.' To address this, we employed prompt engineering as our 'guardrail.' These guardrails act as the final verification step and guide the AI's output generation, ensuring that all responses are supported by evidence. By setting limits on the generated content—such as adhering to known facts and respecting ethical considerations—we can significantly reduce the likelihood of hallucinations and produce more accurate and reliable results.

9 Conclusion

In conclusion, we have successfully implemented a GenAI-powered chatbot in our Metadata Management System, specifically designed to enhance KPMG's operational efficiency and reduce risks. This tool significantly streamlines information and metadata management, adding considerable business value. The chatbot's applications are broad, simplifying data analysis for scientists, enhancing system integrity for engineers, and ensuring a seamless onboarding process for new graduates. These enhancements demonstrate our commitment to improving professional workflows and maintaining high standards of reliability, effectively delivering business value to KPMG.

10 Future Steps

Our primary goal for future plan is to enhance our model by significantly decreasing its latency, thus ensuring faster response times and more efficient processing. Additionally, we plan to explore and experiment with various versions of large language models. By comparing these different models, we aim to evaluate their performance across a range of metrics such as accuracy, efficiency, and adaptability to different tasks. This comprehensive analysis will help us identify the most suitable model that aligns with our specific needs and objectives, ultimately leading to more robust and effective applications.

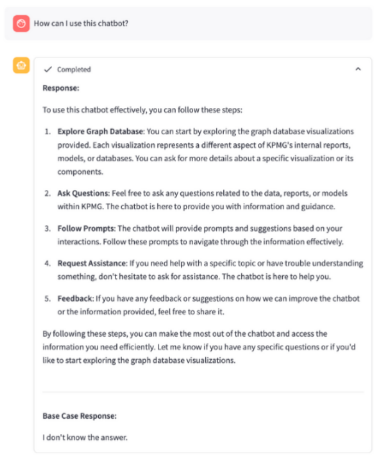
11 Appendix

11.1 Contribution

- **Tianyue Cao**: Chatbot Development, UI/UX Interface, Prompt Engineering (Classification Layer, ChitChat Tool, Introduction Tool)
- **Lu Liu**: Mock Data Generation, Chatbot Development, Prompt Engineering (Answer Layer & Guardrail)
- **Wanxin Luo**: Mock Data Generation, Neo4j Visualization, Prompt Engineering (Cypher Code Generation)
- **Xinwei Qiao**: Chatbot Architecture Development, Automatic Testing Pipeline Development, Mock Data Generation, Prompt Engineering (Extraction Tool)
- **Yao Xie**: Mock Data Generation, Chatbot Development, Prompt Engineering (Cypher Code Improvement)

11.2 Business Case Screenshots

- **New Employee:**



How can I use this chatbot?

Completed

Response:

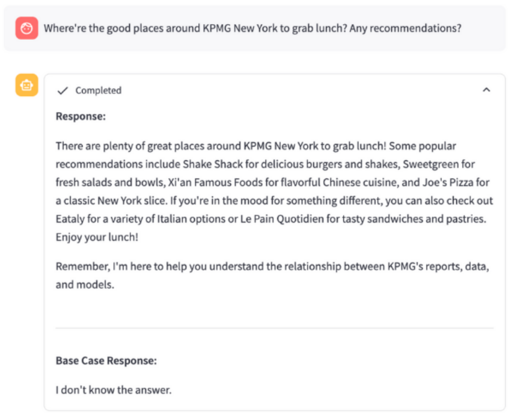
To use this chatbot effectively, you can follow these steps:

1. **Explore Graph Database**: You can start by exploring the graph database visualizations provided. Each visualization represents a different aspect of KPMG's internal reports, models, or databases. You can ask for more details about a specific visualization or its components.
2. **Ask Questions**: Feel free to ask any questions related to the data, reports, or models within KPMG. The chatbot is here to provide you with information and guidance.
3. **Follow Prompts**: The chatbot will provide prompts and suggestions based on your interactions. Follow these prompts to navigate through the information effectively.
4. **Request Assistance**: If you need help with a specific topic or have trouble understanding something, don't hesitate to ask for assistance. The chatbot is here to help you.
5. **Feedback**: If you have any feedback or suggestions on how we can improve the chatbot or the information provided, feel free to share it.

By following these steps, you can make the most out of the chatbot and access the information you need efficiently. Let me know if you have any specific questions or if you'd like to start exploring the graph database visualizations.

Base Case Response:

I don't know the answer.



Where're the good places around KPMG New York to grab lunch? Any recommendations?

Completed

Response:

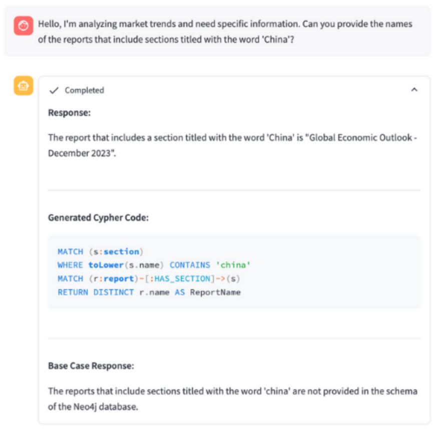
There are plenty of great places around KPMG New York to grab lunch! Some popular recommendations include Shake Shack for delicious burgers and shakes, Sweetgreen for fresh salads and bowls, Xi'an Famous Foods for flavorful Chinese cuisine, and Joe's Pizza for a classic New York slice. If you're in the mood for something different, you can also check out Eataly for a variety of Italian options or Le Pain Quotidien for tasty sandwiches and pastries. Enjoy your lunch!

Remember, I'm here to help you understand the relationship between KPMG's reports, data, and models.

Base Case Response:

I don't know the answer.

- **Business Analyst:**



Hello, I'm analyzing market trends and need specific information. Can you provide the names of the reports that include sections titled with the word 'China'?

Completed

Response:

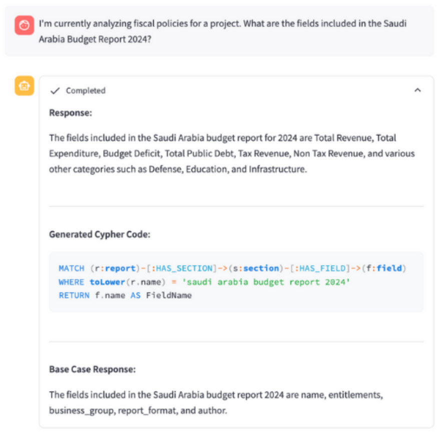
The report that includes a section titled with the word 'China' is "Global Economic Outlook - December 2023".

Generated Cypher Code:

```
MATCH (s:section)
WHERE toLower(s.name) CONTAINS 'china'
MATCH (r:report)-[:HAS_SECTION]->(s)
RETURN DISTINCT r.name AS ReportName
```

Base Case Response:

The reports that include sections titled with the word 'china' are not provided in the schema of the Neo4j database.



I'm currently analyzing fiscal policies for a project. What are the fields included in the Saudi Arabia Budget Report 2024?

Completed

Response:

The fields included in the Saudi Arabia budget report for 2024 are Total Revenue, Total Expenditure, Budget Deficit, Total Public Debt, Tax Revenue, Non Tax Revenue, and various other categories such as Defense, Education, and Infrastructure.

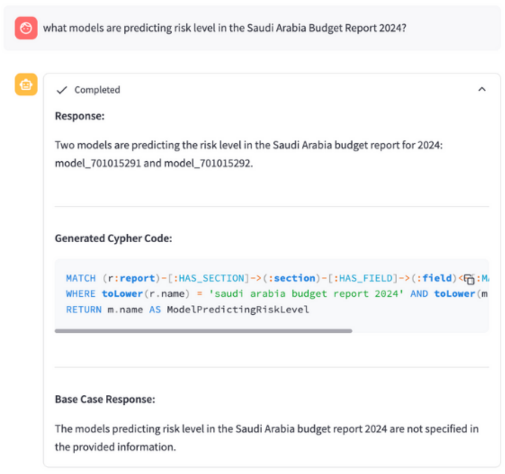
Generated Cypher Code:

```
MATCH (r:report)-[:HAS_SECTION]->(s:section)-[:HAS_FIELD]->(f:field)
WHERE toLower(r.name) = 'saudi arabia budget report 2024'
RETURN f.name AS FieldName
```

Base Case Response:

The fields included in the Saudi Arabia budget report 2024 are name, entitlements, business_group, report_format, and author.

- **Data Scientist:**



what models are predicting risk level in the Saudi Arabia Budget Report 2024?

Completed

Response:

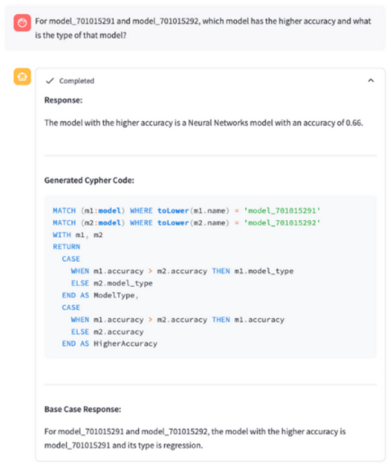
Two models are predicting the risk level in the Saudi Arabia budget report for 2024: model_701015291 and model_701015292.

Generated Cypher Code:

```
MATCH (r:report)-[:HAS_SECTION]->(:section)-[:HAS_FIELD]->(:field)<-[:M
WHERE toLower(r.name) = 'saudi arabia budget report 2024' AND toLower(m
RETURN m.name AS ModelPredictingRiskLevel
```

Base Case Response:

The models predicting risk level in the Saudi Arabia budget report 2024 are not specified in the provided information.



For model_701015291 and model_701015292, which model has the higher accuracy and what is the type of that model?

Completed

Response:

The model with the higher accuracy is a Neural Networks model with an accuracy of 0.66.

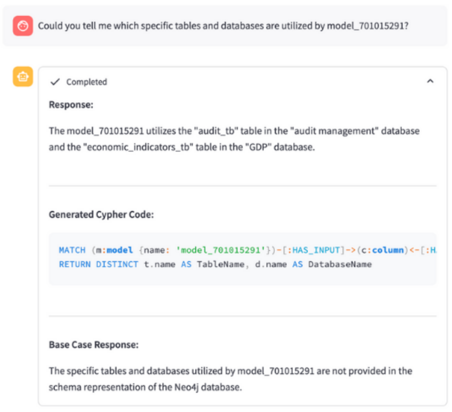
Generated Cypher Code:

```
MATCH (m1:model) WHERE toLower(m1.name) = 'model_701015291'
MATCH (m2:model) WHERE toLower(m2.name) = 'model_701015292'
WITH m1, m2
RETURN
  CASE
    WHEN m1.accuracy > m2.accuracy THEN m1.model_type
    ELSE m2.model_type
  END AS ModelType,
  CASE
    WHEN m1.accuracy > m2.accuracy THEN m1.accuracy
    ELSE m2.accuracy
  END AS HigherAccuracy
```

Base Case Response:

For model_701015291 and model_701015292, the model with the higher accuracy is model_701015291 and its type is regression.

- **Data Engineer:**



Could you tell me which specific tables and databases are utilized by model_701015291?

Completed

Response:

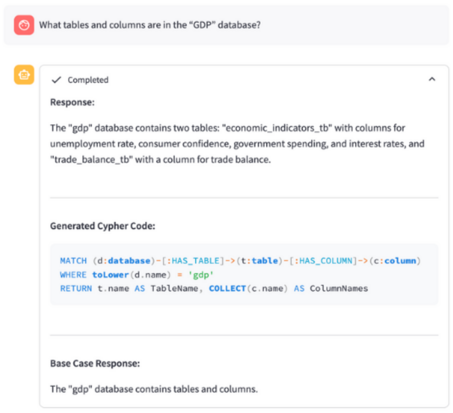
The model_701015291 utilizes the "audit_tb" table in the "audit management" database and the "economic_indicators_tb" table in the "GDP" database.

Generated Cypher Code:

```
MATCH (m:model {name: 'model_701015291'})-[:HAS_INPUT]->(c:column)<-[:H
RETURN DISTINCT t.name AS TableName, d.name AS DatabaseName
```

Base Case Response:

The specific tables and databases utilized by model_701015291 are not provided in the schema representation of the Neo4j database.



What tables and columns are in the “GDP” database?

Completed

Response:

The "gdp" database contains two tables: "economic_indicators_tb" with columns for unemployment rate, consumer confidence, government spending, and interest rates, and "trade_balance_tb" with a column for trade balance.

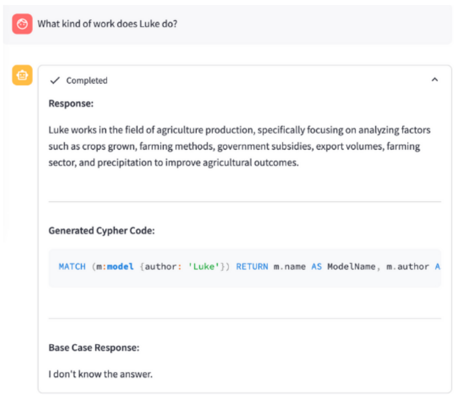
Generated Cypher Code:

```
MATCH (d:database)-[:HAS_TABLE]->(t:table)-[:HAS_COLUMN]->(c:column)
WHERE toLower(d.name) = 'gdp'
RETURN t.name AS TableName, COLLECT(c.name) AS ColumnNames
```

Base Case Response:

The "gdp" database contains tables and columns.

- **Product Manager:**



What kind of work does Luke do?

Completed

Response:

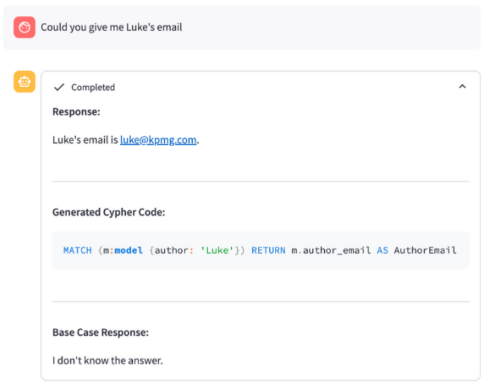
Luke works in the field of agriculture production, specifically focusing on analyzing factors such as crops grown, farming methods, government subsidies, export volumes, farming sector, and precipitation to improve agricultural outcomes.

Generated Cypher Code:

```
MATCH (m:model {author: 'Luke'}) RETURN m.name AS ModelName, m.author A
```

Base Case Response:

I don't know the answer.



Could you give me Luke's email

Completed

Response:

Luke's email is luke@kpmg.com.

Generated Cypher Code:

```
MATCH (m:model {author: 'Luke'}) RETURN m.author_email AS AuthorEmail
```

Base Case Response:

I don't know the answer.

References

[1] *Langchain Neo4j Integration - Neo4j Labs*. Neo4j Graph Data Platform. (n.d.). https://neo4j.com/labs/genai-ecosystem/langchain

[2] Bratanic, T. (2024, January 18). *Enhancing interaction between language models and graph databases via a semantic layer*. Medium. https://towardsdatascience.com/enhancing-interaction-between-language-models-and-graph-databases-via-a-semantic-layer-0a78ad3eba49

[3] Bratanic, T. (2023, April 11). *Context-aware knowledge graph chatbot with GPT-4 and neo4j*. Neo4j. https://neo4j.com/developer-blog/context-aware-knowledge-graph-chatbot-with-gpt-4-and-neo4j/