Software Requirements Specification

For

Galava: A Scalable Search Engine for Organizational Database Pools using Large Language Models

Prepared by

Name	Roll No	Branch
SAI ANKITA KASIBATLA	R2142210910	BTECH CSE AIML
SAAHIL SINGH RATHORE	R2142211304	BTECH CSE CCVT

Department of Systemics
School Of Computer Science
UNIVERSITY OF PETROLEUM & ENERGY STUDIES,
DEHRADUN- 248007. Uttarakhand

Table of Contents

1. INTRODUCTION

In the era of information, data has become the lifeblood of organizations, driving decision-making, innovation, and strategic planning across various sectors. The ability to efficiently store, manage, and retrieve data is paramount to leveraging its full potential. Over the years, advancements in database technologies have led to the development of diverse systems tailored to specific types of data and use cases, ranging from traditional relational databases to more flexible NoSQL databases designed to handle unstructured data. While these technological advancements have significantly enhanced data storage and management capabilities, they have also introduced a new set of challenges in data retrieval, particularly in organizational environments characterized by a multitude of disparate database systems.

The proliferation of diverse databases within organizations has led to data silos, where information is stored in separate, often incompatible systems. This fragmentation hinders the ability to perform comprehensive searches across the entire data landscape, limiting the visibility and accessibility of valuable information. Moreover, the complexity of query languages specific to each database system necessitates a high degree of technical expertise, restricting data access to a select group of individuals within the organization. This barrier not only slows down the decision-making process but also diminishes their ability to fully capitalize on its data assets.

Furthermore, as organizations continue to grow and their data ecosystems expand, scalability becomes a critical concern. Traditional data retrieval methods struggle to keep pace with the increasing volume of data and the demand for real-time access, leading to delays and inefficiencies that can hamper operational agility and responsiveness.

Recognizing these challenges, the need for a unified, efficient, and user-friendly solution for navigating the complex web of organizational databases has become more apparent than ever. Such a solution would not only break down data silos and provide a holistic view of the data assets but also simplify the data retrieval process, making it accessible to a broader range of users regardless of their technical expertise.

In response to this need, we introduce Galava, a novel search engine designed to bridge the gap between diverse database systems and the users seeking to access their data. Galava stands out by offering a low-latency, scalable, and intuitive interface that allows for seamless navigation across a plethora of databases, effectively democratizing data access within organizations. By leveraging advanced natural language processing (NLP) techniques and knowledge graphs, Galava aims to transform the data retrieval landscape, making it more efficient, inclusive, and agile.

This project outlines the development of Galava, detailing its innovative approach to overcoming the challenges of data fragmentation, complexity, and scalability in organizational database environments. Through a comprehensive examination of its architecture, functionality, and potential impact, we aim to contribute to the ongoing discourse on data management and retrieval solutions, offering insights and perspectives that could shape the future of organizational data strategies.

1.1 Purpose of the Project

Galava is designed as an innovative solution aimed at addressing the fragmentation and complexity of data retrieval across multiple database systems within organizations. Utilizing a combination of advanced natural language processing and a sophisticated indexing algorithm, Galava facilitates a user-friendly interface that allows non-technical users to perform complex data searches. The system is scalable and adept at managing large volumes of data, enabling efficient and timely data access to enhance informed decision-making.

1.2 Target Beneficiary

The primary beneficiaries of Galava are business analysts, researchers, and decision-makers within organizations who lack technical database querying skills. Galava serves as a bridge between complex data pools and end-users, democratizing data access and allowing for informed decision-making without the need for extensive technical training. IT departments also benefit from reduced overhead in managing and facilitating data access requests across the organization.

1.3 Project Scope

Galava's scope encompasses the development of a scalable interface to conduct searches across distributed databases, using a hierarchical navigable small world (HNSW) algorithm for quick and precise data retrieval and a write-ahead caching mechanism for enhanced efficiency. It aims to significantly reduce the time and expertise required to access and analyze data. The integration of Galava with existing data systems provides a seamless transition from traditional query methods to a more intuitive, language-driven approach to data retrieval. The project's innovative use of natural language processing will potentially set a new standard for how data is accessed and utilized across various sectors.

1.4 References

1. Bazaga, A., Gunwant, N., & Micklem, G. (2021). Translating synthetic natural language to database queries with a polyglot deep learning framework. *Scientific Reports*, (Bazaga et al., 2021)

2. Dar, H. S., Lali, M. I., Din, M. U., Malik, K. M., & Bukhari, S. A. (2019). Frameworks for Querying Databases Using Natural Language: A Literature Review. (Dar et al., 2019)

3. Papenmeier, A., Kern, D., Hienert, D., Sliwa, A., Aker, A., & Fuhr, N. (2023). Starting Conversations with Search Engines -- Interfaces that Elicit Natural Language Queries. (Papenmeier et al., 2021)

2. PROJECT DESCRIPTION

2.1 Data/ Data structure

- Graph Data Structure: This structure is employed to represent the relationships between entities in the database systems, forming a knowledge graph. Nodes in the graph represent entities or individual records, while edges denote relationships or associations between these entities. This structure is crucial for understanding the complex connections within the data and for performing efficient queries.

- Arrays: Arrays are used to store configurations and metadata information retrieved from the databases. They provide a means to hold a collection of elements, typically of the same data type, in an indexed format. This allows for quick access and manipulation of configuration details and metadata elements necessary for initializing connections and processing queries.

2.2 SWOT Analysis

Strengths:

- **Innovative Interface**: Use of NLP to enable complex queries in natural language greatly simplifies user interaction, removing the need for specialized database query knowledge.
- **Scalability**: The system's design incorporates a scalable architecture that can handle an increasing volume of queries and data without significant performance degradation.
- **Real-Time Processing**: The capability for real-time processing and immediate feedback on queries is a strong advantage in decision-making scenarios.
- **Use of Advanced Algorithms**: The use of HNSW graphs ensures that Galava can perform quick and precise data retrieval, even in large-scale database environments.

Weaknesses:

- **System Complexity**: The sophistication of the system could make troubleshooting and maintenance challenging, especially in the face of integrating various database systems with different schemas and query languages.
- **Initial Setup and Integration**: The initial setup of Galava and its integration into existing systems could be complex and time-consuming.
- **Dependence on External Services**: Galava's performance is reliant on third-party services like AWS, which could pose risks if these services experience outages or changes.

Opportunities:

- **Expansion to New Markets**: As organizations continue to accumulate vast amounts of data, the need for efficient retrieval systems grows, creating opportunities for Galava to expand into new markets.
- **Strategic Partnerships**: Forming partnerships with database providers could lead to optimized integration and enhance Galava's compatibility and reach.

Threats:

- **Competition:** Other companies might develop similar or more advanced solutions, which could threaten Galava's market position.
- **Technological Advances:** Rapid changes in technology could render Galava's current algorithms obsolete if not continuously updated.
- **Data Privacy Regulations**: Stricter data privacy laws could impact the way Galava collects, processes, and stores data, necessitating continuous legal adjustments.

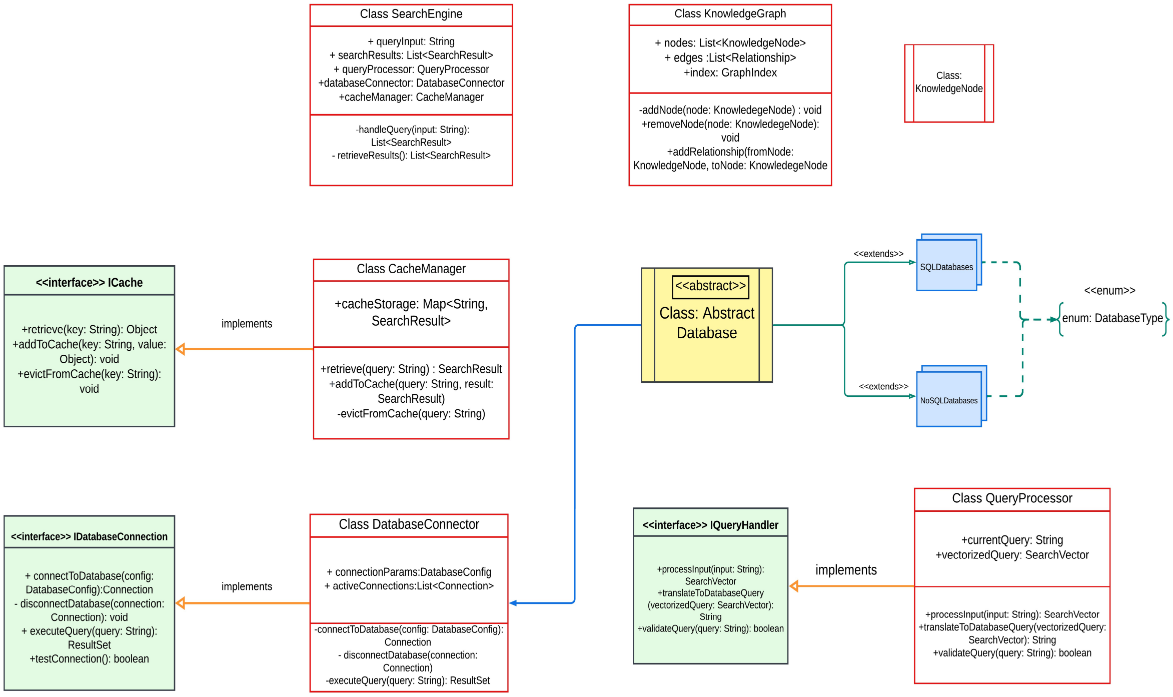
2.3 Project Features

The principal functionality of the Galava project is to streamline complex data retrieval processes using natural language queries. The system features a real-time monitoring interface for database searches, capable of interpreting user input and transforming it into actionable database queries. It also includes an intelligent alert system that notifies users if query responses deviate from expected parameters, as specified by predefined thresholds within the algorithm. By leveraging this innovative platform, users can access and analyze data from various databases effortlessly. All data interactions and user queries are logged securely, ensuring that information can be reviewed and utilized to refine future search operations and system optimizations.

2.4 Design and Implementation Constraints

- Networking Variability: Ensuring consistent connectivity across distributed databases, which may be located in different network environments with varying levels of stability and speed.
- Data Storage Limitations: Balancing the need for immediate, rapid access to data against the costs and technical limitations associated with scaling storage solutions in line with growing data volumes.
- System Complexity: The challenge of integrating disparate data systems with potentially different schemas, query languages, and access protocols within a unified search interface.

2.5 Design diagram


Class SearchEngine
+ queryInput: String
+ searchResults: List<SearchResult>
+ queryProcessor: QueryProcessor
+databaseConnector: DatabaseConnector
+cacheManager: CacheManager
-handleQuery(input: String): List<SearchResult>
- retrieveResults(): List<SearchResult>
Class KnowledgeGraph
+ nodes: List<KnowledgeNode>
+ edges :List<Relationship>
+index: GraphIndex
-addNode(node: KnowledegeNode) : void
+removeNode(node: KnowledegeNode): void
+addRelationship(fromNode: KnowledgeNode, toNode: KnowledegeNode
Class: KnowledgeNode
<<interface>> ICache
+retrieve(key: String): Object
+addToCache(key: String, value: Object): void
+evictFromCache(key: String): void
implements
Class CacheManager
+cacheStorage: Map<String, SearchResult>
+retrieve(query: String) : SearchResult
+addToCache(query: String, result: SearchResult)
-evictFromCache(query: String)
<<abstract>>
Class: Abstract Database
<<extends>>
SQLDatabases
<<extends>>
NoSQLDatabases
<<enum>>
enum: DatabaseType
<<interface>> IDatabaseConnection
+ connectToDatabase(config: DatabaseConfig):Connection
- disconnectDatabase(connection: Connection): void
+ executeQuery(query: String): ResultSet
+testConnection(): boolean
implements
Class DatabaseConnector
+ connectionParams:DatabaseConfig
+ activeConnections:List<Connection>
-connectToDatabase(config: DatabaseConfig): Connection
- disconnectDatabase(connection: Connection)
-executeQuery(query: String): ResultSet
<<interface>> IQueryHandler
+processInput(input: String): SearchVector
+translateToDatabaseQuery (vectorizedQuery: SearchVector): String
+validateQuery(query: String): boolean
implements
Class QueryProcessor
+currentQuery: String
+vectorizedQuery: SearchVector
+processInput(input: String): SearchVector
+translateToDatabaseQuery(vectorizedQuery: SearchVector): String
+validateQuery(query: String): boolean

3. SYSTEM REQUIREMENTS

3.1 User Interface

For Galava, an intuitive user interface is essential for users to conduct searches and view data with ease. The front-end will be developed using the Streamlit library, to provide a responsive and accessible web application. Users will be able to input natural language queries and receive data in a well-organized format that enhances readability and analysis.

3.2 Protocols

- Data Transmission Protocol:

  RESTful API: Representational State Transfer (REST) API will be used to handle client-server communications. It allows for a stateless, client-server, cacheable communications protocol — the HTTP standard is used underneath.

- Data Security Protocol:

  HTTPS: Secure Hypertext Transfer Protocol (HTTPS) will be implemented for secure communication over the network. It provides encrypted and secure identification of a network web server to prevent data breaches.

4. NON-FUNCTIONAL REQUIREMENTS

4.1 Performance requirements

Galava must adhere to the following non-functional performance criteria:

1. Availability: The service must aim for high availability, with aspirations to achieve 24/7 uptime, minimizing service interruptions to ensure that users can access the platform whenever needed.

2. Scalability: The architecture should be designed to efficiently manage increasing numbers of users and data queries without degradation in performance, facilitating resource expansion in response to growing demands seamlessly.

3. Fault Tolerance: The system must be resilient to faults and capable of maintaining operational functionality in the event of component failures, ensuring continuous system availability and reliability.

4. Performance: The system should maintain swift response times for user queries, even under high load, to ensure a smooth and efficient user experience. Performance benchmarks should be established, aiming to process and return search results within a minimal and acceptable time frame.

4.2 Security requirements

To ensure the integrity and confidentiality of data within Galava, the following security measures are implemented:

- Virtual Private Cloud (VPC): Galava utilizes AWS VPC to establish a private network, ensuring that all resources are logically isolated from other networks. VPC provides advanced security features, such as security groups and network access control lists, which offer granular inbound and outbound filtering at the instance and subnet level.

- Network Segmentation: Network segmentation is rigorously enforced, dividing the network into separate segments or subnets. This minimizes the attack surface and restricts unauthorized access to sensitive components, ensuring that any compromise in one segment does not affect the others.

- Encryption: Data in transit and at rest will be encrypted using robust encryption standards. This ensures that sensitive data, including user queries and retrieved data, remain secure against interception and unauthorized access.

- Identity and Access Management (IAM): Strict IAM policies will be used to control access to AWS resources. IAM roles and policies will be defined to grant necessary permissions to users and services, enforcing the principle of least privilege.

APPENDIX A: GLOSSARY

- **Large Language Models (LLMs):** Advanced AI models trained on vast datasets of text from the internet or specific domains. They use deep learning techniques, particularly variants of the Transformer architecture, to understand, generate, and manipulate human-like text based on the input they receive. These models excel in a variety of natural language processing tasks, including but not limited to text generation, translation, summarization, and question-answering.

- **Knowledge Graph:** A sophisticated data structure that connects entities (such as objects, places, events, and concepts) and their interrelationships in a graph format. Each node in the graph represents an entity, and the edges denote the relationships between these entities. Knowledge graphs are instrumental in representing complex, interconnected information in a structured and understandable manner, enabling efficient data retrieval and analysis.

- **Vault:** A vault is a secure digital storage solution designed to protect sensitive information such as passwords, keys, tokens, and certificates. Vaults are built with robust encryption and access control mechanisms to ensure that sensitive data is stored, accessed, and managed securely, minimizing the risk of unauthorized access or breaches. "Galava" utilizes a vault to securely store critical information required for accessing and interacting with the diverse databases in the organizational network. This includes secret keys and URIs to object storage locations containing configuration files and metadata.

- **Cache Mechanism:** A cache is a high-speed data storage layer that stores a subset of data, typically transient in nature, so that future requests for that data can be served faster than by accessing the data's primary storage location. Caching mechanisms improve the efficiency and performance of a system by reducing the data access time.

- **Hierarchical Navigable Small World (HNSW) Graphs**: A type of graph algorithm used for efficient Approximate Nearest Neighbor (ANN) searches. HNSW utilizes a layered structure where each layer is a graph that connects the nearest neighbors. The algorithm is designed for high-dimensional spaces and is known for its balance between accuracy and search efficiency, making it suitable for tasks like vector search in large datasets.