

# **Agentic BI- 295A Final Report**

A Project Report  
Presented to  
The Faculty of the College of  
Engineering

San Jose State University  
In Partial Fulfillment  
Of the Requirements for the Degree  
**Master of Science in Artificial Intelligence**

By

Apoorva Adimulam, Aditya Chawla, Kushagra Kshatri, Shivang Patel

May 2026

Copyright © 2026

Apoorva Adimulam, Aditya Chawla, Kushagra Kshatri, Shivang Patel  
ALL RIGHTS RESERVED

**APPROVED**

---

Dr. Bernardo Flores, Project Advisor

## ABSTRACT

Agentic BI

By Apoorva Adimulam, Aditya Chawla, Kushagra Kshatri, Shivang Patel

Data Analysis serves as the foundation of Business Intelligence by providing systematic methods that enable evidence-based decision making for formulating corporate strategies and data-driven actions to solve problems at hand. It allows companies to identify latest trends, monitor performance indicators, and satisfy the ever volatile customer needs, giving companies a significant competitive advantage in fast-paced markets.

While data analysis provides significant upsides, it is a difficult skill to master and requires analysts to possess significant domain-specific expertise. While analysts are equipped with data visualization and dashboarding tools like PowerBI, Tableau and Redash, they also present some notable challenges. These platforms can be cumbersome to use and are heavily reliant on extensive hands-on experience and manual setup. Query generation on these platforms requires in-depth understanding of databases, development of visually appealing dashboards is time consuming, getting meaningful insights from these visualizations requires experience, KPI (Key Performance Indicator) monitoring and performance tracking is tedious.

In this project, we propose an end-to-end automation of the Business Intelligence process to solve the previously mentioned challenges through a Multi-Agent architecture where each Agent is in charge of a specialized sub-process. The expected result would be an interactive multi-agent AI analytics software that decision-makers can interact with to achieve automated high-quality dashboard visualization, help ease the monitoring process of these dashboards, and generate intelligent and explainable insights. This software aims to simplify data analysis and business insights by reducing complexity, saving time, and making informed decision-making smarter.

### **Acknowledgments**

The authors are deeply indebted to Professor Bernardo Flores for his invaluable comments and assistance in the preparation of this study.

## Table of Contents

<b>Chapter 1. Project Overview</b>	<b>1</b>
Introduction	1
Proposed Areas of Study and Academic Contribution	1
Current State of the Art	2
<b>Chapter 2. Project Architecture</b>	<b>4</b>
Introduction	4
Architecture Subsystems	6

## **List of Figures**

Figure 1 - System Architecture	5
Figure 2- Mock User Interface	7
Figure 3: Sequence Diagram of Data Ingestion and SQL Generation	8
Figure 4: Sequence diagram of SQL Execution	9
Figure 5: Sequence diagram of Data Visualization	10

## **List of Tables**

## Chapter 1. Project Overview

### 1.1. Introduction

Business Intelligence (BI) has become indispensable for organizations seeking to compete in data-driven markets. And yet, even with the established availability of powerful tools like PowerBI, Tableau, Redash, it's clear that many businesses are still finding it difficult to fully leverage the real potential of their own data. A major issue is that these existing platforms tend to demand a significant depth of technical expertise, still rely heavily on the manual crafting of complex queries, and require constant, sometimes tedious, oversight to maintain effectiveness. For companies that don't have the luxury of maintaining a large, specialized analytics team, this dependence directly translates into a few critical problems: high operational costs, increased response times, and missed opportunities in the fast-moving commercial sector. BI techniques are becoming modernised with the rise of LLMs and the subsequent development of LLM-based BI models like SiriusBI but there has not been any industry-level adaptation of LLMs in BI frameworks. With other domains rapidly adapting to the advancements in AI, this seems to be the perfect time to incorporate AI-based automation in Business Intelligence.

This project introduces an Agentic Business Intelligence system that leverages multi-agent architectures to automate the complete BI pipeline—from natural language query interpretation to SQL execution, visualization generation, monitoring, Anomaly detection and insight extraction. By deploying specialized autonomous agents coordinated through frameworks like CrewAI, the system reduces dependency on technical personnel, enables real-time anomaly detection, and scales efficiently without proportional increases in human resources. The architecture integrates guardrails for privacy and security, semantic layers for business logic consistency, and AI-native visualization frameworks - Vizro to deliver enterprise-grade analytics capabilities.

### 1.2. Proposed Areas of Study and Academic Contribution

Current BI automation research predominantly explores monolithic LLM applications, yet research evidence suggests complex analytical workflows benefit from self-managed, dynamic resilient systems. This project operationalizes the theoretical principles of multi-agent system through a production-oriented architecture where autonomous agents handle distinct responsibilities—schema understanding, query formulation, execution validation, visualization, monitoring and insight extraction. The research quantifies performance gains of this distributed approach against single-agent baselines, measuring

accuracy, semantic correctness, and scalability across the complete analytics pipeline. By implementing Agent-to-Agent (A2A) communication protocols and Model Context Protocol (MCP) integration through frameworks like CrewAI, the work provides replicable design patterns for enterprise agentic BI systems.

This research also addresses critical gaps where current single-agent systems fall short in enterprise data analytics by advancing individual components across the complete BI pipeline—including NL2SQL translation, Text-to-Visualization generation, guardrails for AI agents, multi-agent orchestration, seamless tool calling, and inter-agent communication. The implementation leverages modern approaches such as Schema-Graph with Integrated Syntax (SGIS) for context-aware query generation and the LLM-Enhanced Visual Analytics (LEVA) framework for interactive data exploration, while integrating production-ready enterprise-grade tools like Vizro for AI-native dashboard development. Furthermore, the architecture adopts emerging open standards including Agent-to-Agent (A2A) protocol for direct agent communication and Model Context Protocol (MCP) for structured tool access and resource management, establishing a foundation for state-of-the-art agentic analytics systems.

## Current State of the Art

The state-of-the-art tools and technologies in BI and Analytics have gone through a fundamental shift from reactive to proactive intelligence, with leading platforms converging on conversational interfaces, real-time processing capabilities, and autonomous AI systems. This evolution positions business intelligence as the operational nervous system of modern enterprises, democratizing data access through integrated AI-native platforms that eliminate traditional barriers between users, applications and insights.

Microsoft the market leader and technology powerhouse, had released Azure Fabric's Real-Time Intelligence [1] an end-to-end solution for ingesting, processing, analyzing, visualizing, monitoring, alerting, and acting on data which has become the fastest-growing workload with 24,000+ customers and 6x adoption growth in the past year [2]. The platform enables users to build real-time dashboards using natural language with Copilot, eliminating technical barriers to real-time analytics. Power BI by Microsoft which is the market leader in BI and analytics industry is geared to be transformed into an AI-powered business intelligence system with Copilot Default-On Experience [3] to provide a full-screen, chat-based AI interface that allows users to explore data and generate insights across reports without traditional interface barriers.

Tableau by salesforce the second most popular BI software in its latest release focuses on agentic analytics with Enhanced Q&A with Multilingual Support [4] allowing users to ask questions in their preferred language with improved text formatting and multiple

entry points. Correlated Metrics Insights provides new AI-driven insight types that automatically identify significant relationships between metrics.

Redash is a commonly used Open-source BI platform that has been democratizing data access for over a decade, with a SQL oriented approach rather than relying on GUI. Redash offers enterprise-grade capabilities for visualization and dashboarding without licensing constraints thereby offering similar performance for budget constrained companies and is a solid alternative to closed source products available in the market.

Vizro is a modern open source AI native BI framework developed by McKinsey for low-code/no-code approach to create production ready enterprise grade dashboards. It is developed using modern web standards (react and python) and also supports native and deep integration with AI/ML frameworks such as Crew AI, MCP (Model Context Protocol), A2A(Agent to agent protocol) etc. It is a forward-looking framework as a modern alternative to traditional BI tools.

The business intelligence industry is going through unprecedented transformation, driven by generative AI integration, real-time intelligence capabilities, and conversational analytics. Leading vendors are rapidly deploying cutting-edge features that fundamentally reshape how organizations interact with data.

## Chapter 2. Project Architecture

### 2.1. Introduction

This Agentic BI system is based on a layered, event-driven, and service-oriented architecture featuring isolation of user interactions, robust agent orchestration, and scalable analytics pipelines. This architecture moves beyond traditional static dashboards by combining conversational AI with enterprise-grade agentic orchestration, thus enabling automation of sophisticated business intelligence workflows all the way from natural language understanding to data retrieval and visualization.

This system is organized into four major logical layers: the UI Layer, the Guardrail Layer, the central Orchestrator, and the Multi-Agent Layer. Given the complexity that needs to be dealt with while building and deploying this project, following such modular architecture allows for high scalability while providing consistency across all operations. Separating the user interface from the agentic processing logic would ensure that complex queries can be handled asynchronously, without interfering with client-side rendering. A guardrail layer on top of this entire architecture enhances the privacy of the overall product.

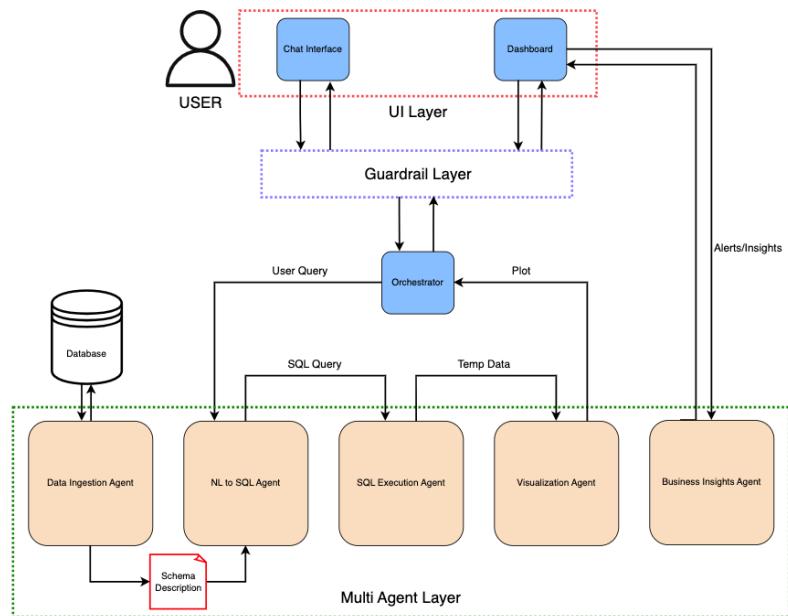


Fig. 1 - Project Architecture

Figure 1 gives a high-level visual description of the architecture of Agentic BI. The system adopts a layered, event-driven, and service-oriented architecture designed to isolate user interactions, ensure robust agent orchestration, support scalable analytics pipelines, and maintain strict enterprise-grade security and privacy controls. This architecture embodies the modern paradigms of business intelligence by seamlessly integrating conversational AI and enterprise agentic orchestration, aligning with both current academic research and leading industry deployments.

Guardrails sits between the UI and core system, providing real-time input validation, output filtering, and enforcement of privacy policies. Implements authentication, authorization, zero-trust practices, privacy controls, and compliance auditing.

The Agent Orchestration Layer is the heart of the solution and uses the CrewAI orchestration framework for workflow management. It supports Agent-to-Agent (A2A) protocol for direct agent-to-agent communication and manages the lifecycle of individual task-specialized agents. Specialized Agents for individual tasks : the Monitoring Agent handles event streaming, real-time anomaly detection, and efficient alerting via stream processing pipelines. Analytics Agent performs statistical analysis and machine-learning powered insight extraction. Visualization Agent manages chart generation and dashboard components, integrating with modern frameworks like Vizro for low-code, real-time BI. Data Agent is responsible for data discovery, query processing, connecting seamlessly to varied underlying data sources. Model Context Protocol (MCP) Integration Agent leverages the Model Context Protocol as a tool gateway, enabling structured resource access and secure external API/resource management. Insight Analysis & Narrative Generation Agent provide contextual explanations and data storytelling, potentially leveraging public/private LLMs.

The Semantic Layer ensures business meaning and logic are consistently applied, translating user queries and agent requests into context-aware SQL/statements. It supports modular updates and versioning, promoting business logic consistency across the platform.

Vizro framework provides AI-native, configuration-driven tools that can be called using modern A2A and MCP protocols by AI agents for development of enterprise grade dashboards and visualizations from data which can be stored in a variety of formats in the data source such as pandas DataFrames, SQL, JSON etc.

Data Sources & Pipeline Layer supports connection to databases, warehouses, lakes, and APIs. Event streaming infrastructure (Kafka, Flink) manages real-time analytics, feeding both anomaly detection and dashboard visualizations.

The system's architecture is a modular, multi-layered framework designed to process natural language user queries, retrieve data, and present visualizations and insights. The architecture is organized into four primary layers: the UI Layer, the Guardrail Layer, the central Orchestrator, and the Multi Agent Layer.

## 2.2. Architecture Subsystems

As portrayed in Figure 1, Agentic BI is proposed to consist of multiple sub-services accounting for separate features of the systems. These subsystems would collaborate with each other to effectively process incoming requests from users. Below is a detailed breakdown of all the subsystems proposed to be used in Agentic BI.

### 1. User Interface Layer

The User Interface (UI) Layer works as the entrypoint for users interacting with Agentic BI. It mainly consists of two components- the Chat Interface and the Dashboard. Figure 2 shows the proposed mock to the UI setup that would be used in the Agentic BI system.

- **Chat Interface:** The chat section is the primary source through which a user would input their natural language queries to interact with the agentic backend. Here, the user would describe the plots they want to add or the problem they are facing and ask for a data-backed visualization to deep dive into the same business problem. The user would expect the agentic backend to return plot(s) through the dashboard component of the UI layer.
- **Dashboard:** This section would serve as the display area in which the final outputs of the agentic backend will be rendered. The plots must be interactive and descriptive in nature.

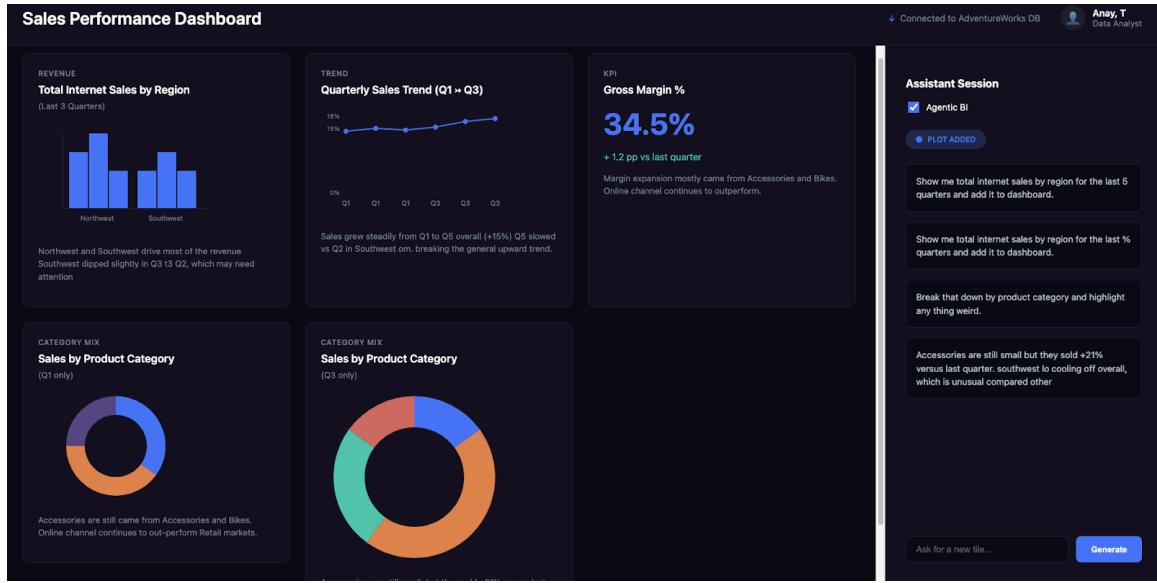


Fig. 2- Mock User Interface for Agentic BI

## 2. Guardrail Layer

The Guardrail Layer is placed between the Chat Interface and the core system logic, maintaining security and governance. It is responsible for performing checks on input user queries and outgoing plots, enforcing privacy and business policies. It is required to identify and flag any incoming threats and attacks, ensuring the durability and security of customer data, while ensuring full policy compliance.

## 3. Orchestrator

The orchestrator acts as the ‘manager’ for the specialized agents in the multi-agent Agentic BI system. It will leverage technologies like CrewAI, LangGraph and Agent-to-Agent (A2A) protocol for handling task distribution among the agents. It receives the sanitized query from the user, identifies tasks that need to be performed and delegates them among the available specialized agents. It is also responsible for routing information between the agents, ensuring the right piece of context goes to the right agent.

#### 4. Database Layer

The database layer is not in-house for the Agentic BI system. It would be hosted on the customer's side through their warehousing technologies. Agentic BI would require the users to connect the system with their data through a relational database structure. In this project, PostgreSQL databases hosted through a docker container would be used for this purpose.

Another data section required for the functioning is a vector database (VectorDB) which will be generated by the Data ingestion agent and required by other agents for their execution. This would be stored in-house in the Agentic BI system.

#### 5. Data Ingestion Agent

This is the first of many agents in the multi-agents system leveraged by Agentic BI. It is responsible for onboarding a user's database into the system. It directly connects with the docker container of the user's data, parses metadata and extracts structural information like table definitions, primary keys, joins, etc. and converts all the gathered information in a vector database. It ensures that the schema description is LLM-friendly and provides a semantic view of the user's underlying database, so that the agentic subsystem can efficiently interact with the data to form queries and plots.

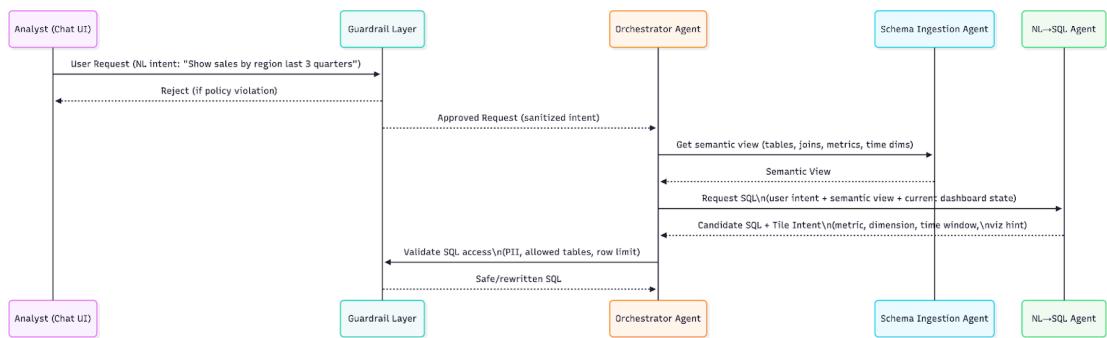


Fig. 3- Sequence Diagram of Data Ingestion and SQL Generation

## 6. NL-to-SQL Agent

The NL-to-SQL agent, along with the Visualization Agent is the most important individual agent of this system, acting as the central translation engine, turning the user's natural language problem into an executable SQL query that can be used by subsequent agents to interact with the underlying database. It takes as input the natural language intent of the user, the context of the dashboard, and the semantic view from the Ingestion Agent, and produces executable database queries. The agent outputs a Candidate SQL query along with a "tile intent," which summarizes the analytics intent-e.g., measures, dimensions, and time windows of interest.

## 7. SQL Execution Agent

The SQL Execution Agent is an operational runtime for data retrieval. It executes the candidate SQL against the target data warehouse and manages the retrieval process. The key features of this agent include a Validation and Retry Loop wherein-if the database returns syntax errors, join inconsistencies, or filter mismatches-the agent automatically tries to correct the query and re-executes it until valid data is retrieved. If the execution is successful, the agent cleans, aggregates, and stages the results in a temporary tile store and assigns a unique identifier for the processing steps downstream.

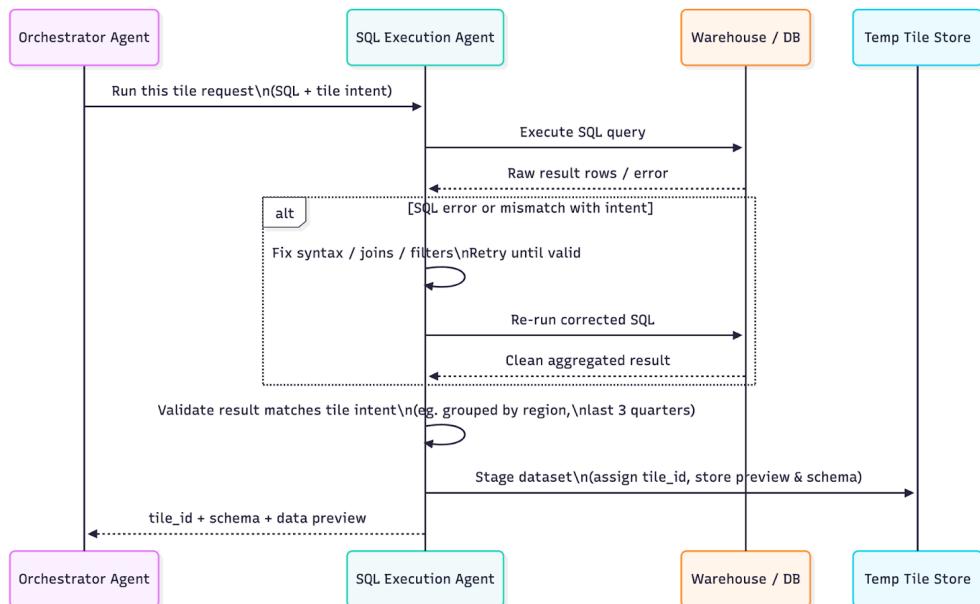


Fig. 4- Sequence diagram of SQL Execution

## 8. Visualization Agent

The Visualization Agent is responsible for the presentation layer, transforming staged data into visual artifacts. Based on data type and granularity, it selects the most effective visualization format to communicate the results properly-for example, bar charts when comparing values, line charts to show trends. It creates accurate chart specifications using staged data that are compatible with frameworks like Plotly (Vizro). Equally, it generates a short interpretive summary-a text in natural language that describes the analytical outcome.

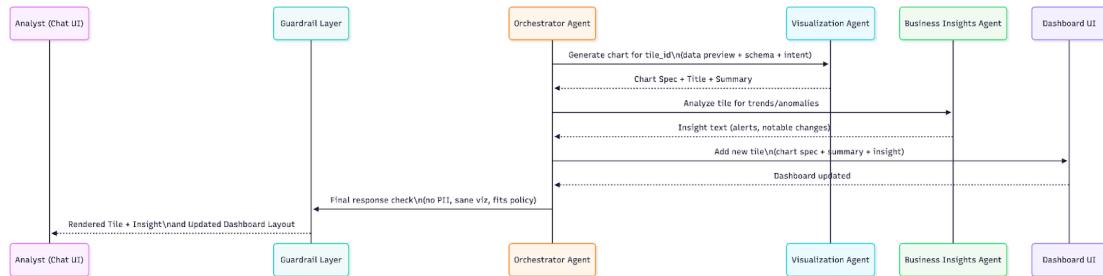


Fig. 5- Sequence diagram of Data Visualization

## 9. Business Insights Agent

This optional agent provides high-level analytical reasoning, scanning one or more generated dashboard tiles for deeper patterns not immediately obvious from the visual chart alone, such as anomalies, correlations, or emerging trends. The agent synthesizes these statistical findings into human-readable narratives that link up with visual tiles to provide contextual "data storytelling" capabilities to the end user.

## Glossary

Term	Definition
Agentic	Pertaining to a system composed of autonomous agents capable of self-managed, goal-oriented actions, decision-making, and communication to complete complex analytical workflows.
LLM	Large Language Model. The core reasoning engine used by the agents to understand natural language, generate code (like SQL), and provide contextual analysis and narrative insights.
NL2SQL	Natural Language to SQL. The process or capability of translating a user's natural language question into an executable SQL query against the underlying database.
Multi-Agent Architecture	A system design pattern where multiple specialized and autonomous AI agents are orchestrated to collaborate, perform distinct responsibilities, and solve a complex analytical problem.
Anomaly Detection	A capability of the Monitoring Agent to automatically identify rare events, sudden shifts, or deviations in real-time data streams that do not conform to expected patterns.
Tile Intent	The structured output from the NL-to-SQL Agent that summarizes the user's analytical goal (e.g., measures, dimensions, time range) alongside the generated SQL, guiding the Visualization Agent.
Orchestrator	The central 'manager' leveraging CrewAI that receives sanitized queries, identifies required tasks, delegates them to specialized agents, and manages the flow of information and collaboration.
Guardrail Layer	A security and governance component positioned between the UI and the core system that validates inputs and filters outputs, enforcing privacy policies, access control, and

	compliance.
Semantic Layer	A conceptual model that sits above the raw database structure, ensuring that business logic, metrics, and data relationships are consistently applied and understood by all agents.
Vector Database (VectorDB)	A specialized database that stores the LLM-friendly schema descriptions and metadata as high-dimensional embeddings for efficient context retrieval by other agents.
CrewAI	The specific orchestration framework used to define, manage, and coordinate the roles, responsibilities, and communication protocols for all specialized agents.
Vizro	The open-source, AI-native visualization framework leveraged by the Visualization Agent for creating interactive, production-ready enterprise-grade dashboards.
A2A Protocol	Agent-to-Agent Protocol. An open communication standard used for structured, direct, and efficient exchange of information and context between autonomous agents.
MCP	Model Context Protocol. An emerging open standard that facilitates structured tool access and secure external API/resource management.
SGIS	Schema-Graph with Integrated Syntax. A modern approach used by the NL-to-SQL Agent to integrate schema relationships into the query generation process, improving semantic correctness.
LEVA	LLM-Enhanced Visual Analytics. A framework utilized to empower agents to perform interactive data exploration and dynamically adjust visualization analysis based on user intent.

## References

- [1] Microsoft, "The foundation for powering AI-driven operations: Fabric real-time intelligence | Microsoft Fabric Blog | Microsoft Fabric," *Microsoft.com*, 2025. [Online]. Available:  
<https://blog.fabric.microsoft.com/en-us/blog/the-foundation-for-powering-ai-driven-operations-fabric-real-time-intelligence/>. Accessed: Sep. 27, 2025.
- [2] McKinsey & Company, "The state of AI: How organizations are rewiring to capture value," *McKinsey & Company*, Mar. 12, 2025. [Online]. Available:  
<https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai>.
- [3] JulCsc, "See what's new with the latest Power BI update - Power BI," *Microsoft.com*, Aug. 12, 2025. [Online]. Available:  
<https://learn.microsoft.com/en-us/power-bi/fundamentals/desktop-latest-update>.
- [4] Tableau, "Tableau Research | Tableau Blog," *Tableau.com*. [Online]. Available:  
<https://www.tableau.com/blog/research>.
- [5] J. Jiang et al., "SiriusBI: A Comprehensive LLM-Powered Solution for Data Analytics in Business Intelligence," Proc. VLDB Endow., vol. 18, no. 12, pp. 4860–4873, 2025, doi: 10.14778/3750601.3750610.
- [6] M. L. Bernardi, A. Casciani, and M. Cimitile, "Conversing with business process-aware large language models: the BPLLM framework," J. Intell. Inf. Syst., vol. 62, pp. 1607–1629, 2024, doi: 10.1007/s10844-024-00898-1.
- [7] L. Lawrence and J. Butler, "A case study of large language models' effectiveness in diverse business applications: Developing a universal integration framework," The Pinnacle: A Journal by Scholar-Practitioners, vol. 2, no. 1, 2024, doi: 10.61643/c38193.
- [8] Z. Zhan, E. Haihong, and M. Song, "Leveraging Large Language Model for Enhanced Text-to-SQL Parsing," IEEE Access, vol. 13, pp. 30497–30504, 2025, doi: 10.1109/ACCESS.2025.3540072.
- [9] Z. Hong et al., "Next-Generation Database Interfaces: A Survey of LLM-based Text-to-SQL," IEEE Trans. Knowl. Data Eng., 2025, doi: 10.1109/TKDE.2025.3609486.
- [10] X. Liu et al., "A Survey of Text-to-SQL in the Era of LLMs: Where Are We, and Where Are We Going?," IEEE Trans. Knowl. Data Eng., vol. 37, no. 10, pp. 5735–5754, Oct. 2025, doi: 10.1109/TKDE.2025.3592032.
- [11] A. Chopra and R. Azam, "Enhancing Natural Language Query to SQL Query Generation Through Classification-Based Table Selection," in Engineering Applications

of Neural Networks (EANN 2024), L. Iliadis, I. Maglogiannis, A. Papaleonidas, E. Pimenidis, and C. Jayne, Eds., Cham: Springer, 2024, vol. 2141.

[12] W. Zhao et al., "Enhancing Interaction Graph of Data Schema and Syntactic Structure with Pre-trained Language Model for Text-to-SQL," in Big Data, vol. 2301, L. Kong et al., Eds., Springer, 2025, pp. 159–173.

[13] S. Chu and J. Liu, "Based on BERT-GPT-GNN converged architecture: intelligent generation engine for complex SQL queries in business intelligence," Discover Artif. Intell., vol. 5, no. 1, Art. no. 147, 2025, doi: 10.1007/s44163-025-00381-y.

[14] Y. Wu et al., "Automated Data Visualization from Natural Language via Large Language Models: An Exploratory Study," Proc. ACM Manag. Data, vol. 2, no. 3, Art. no. 115, 2024, doi: 10.1145/3654992.

[15] Y. Zhao et al., "LEVA: Using Large Language Models to Enhance Visual Analytics," IEEE Trans. Vis. Comput. Graph., vol. 31, no. 3, pp. 1830–1847, 2025, doi: 10.1109/TVCG.2024.3368060.

[16] V. Dhanoa, A. Wolter, G. M. Leon, H.-J. Schulz, and N. Elmquist, "Agentic Visualization: Extracting Agent-based Design Patterns from Visualization Systems," IEEE Comput. Graph. Appl., pp. 1–13, 2025, doi: 10.1109/MCG.2025.3607741.

[17] A. Bandi, B. Kongari, R. Naguru, S. Pasnoor, and S. V. Vilipala, "The Rise of Agentic AI: A Review of Definitions, Frameworks, Architectures, Applications, Evaluation Metrics, and Challenges," Future Internet, vol. 17, no. 9, p. 404, 2025, doi: 10.3390/fi17090404.

[18] A. Petrenko, "Agent-based approach to implementing artificial intelligence (AI) in service-oriented architecture (SOA)," Syst. Res. Inf. Technol., no. 1, pp. 104–123, 2025, doi: 10.20535/SRIT.2308-8893.2025.1.08.

[19] M. M. Karim et al., "Transforming Data Annotation with AI Agents: A Review of Architectures, Reasoning, Applications, and Impact," Future Internet, vol. 17, no. 8, p. 353, 2025, doi: 10.3390/fi17080353.

## Appendices

### Appendix A.

The PoC implementation of the project is available at <https://github.com/Shivang-Patel/AgenticBI>. The repository consists of:

- /src/agents - Dataset stored in PostgreSQL Docker container and source codes for NL2SQL agent, Schema Ingestion agent, SQL Execution agent.
- /tests - Test scripts for agents