© Volume 2, Issue 10, October 2026 | JATIR
e-ISSN: 2839-5786, p-ISSN: 3139-8464

AI Data Analyst Agent for Indian SMEs

[1]Shivani Shitole, [2]Krishnakumar Prajapati, [3]Apeksha Chavhan, [4]Yash Bhoi
[5]Rutuja Holmuke

[1]Assistant Professor, Department of Information Technology P.G. Moze College of Engineering (PGMCOE), Wagholi, Pune, Maharashtra, India

[2,3,4,5]Student, Department of Information Technology P.G. Moze College of Engineering (PGMCOE), Wagholi, Pune, Maharashtra, India

***Abstract*—Small and medium enterprises (SMEs) and educational institutions in India generate large volumes of tabular data, such as sales records, GST and Tally exports, academic results and placement statistics, yet most of them lack the expertise needed to analyze it. Existing chat-with-your-data tools are generic and are not designed for the inconsistent formatting of real Indian business data. This paper proposes an AI Data Analyst Agent, a web-based agentic system in which a user uploads a tabular dataset and asks questions in natural language. The system profiles the dataset and uses a multi-agent architecture, consisting of a Query Planner, a Code Generator, an Insight Explainer and a Chart Recommender, to plan the analysis, generate code and explain the result. AI-generated code runs inside an isolated Docker sandbox, while dedicated machine learning models, namely Prophet, Isolation Forest and K-Means, handle forecasting, anomaly detection and clustering. Results are returned as plain-language explanations with automatically selected visualizations. Because the intelligence lies in the analysis process rather than in dataset-specific logic, the same agent can serve sales, academic and institutional data without redevelopment. The proposed system aims to make computation-backed, explainable data analysis accessible to non-technical users through an open-source technology stack.**

***Index Terms*—Agentic AI, data analysis, large language models, multi-agent systems, sandboxed execution, SME analytics, natural language interface.**

I. INTRODUCTION

Data has become one of the most valuable assets of any organization, regardless of its size. Every business transaction, student enrollment and survey response produces data that can reveal patterns and guide better decisions when it is analyzed correctly. However, turning raw data into insight has traditionally required a trained data analyst or data scientist, a resource that most small businesses and educational institutions in India cannot afford or do not have access to.

Recent advances in Large Language Models (LLMs) and agentic AI systems make it possible to build software that does not merely respond to fixed commands but reasons through a problem in several steps: it understands the intent, plans an approach, executes it and explains the outcome, much like a human analyst. This shift from static, rule-based software to autonomous, reasoning-capable agents opens the door to democratizing data analysis.

This paper presents the design of an AI Data Analyst Agent for Indian SMEs and institutional data. The system is a web-based application in which a user uploads a dataset and interacts with it conversationally. Behind this simple interface, a coordinated set of AI agents decides what kind of analysis is needed, writes and executes the code that performs it, converts the numeric result into a plain-language explanation and recommends a suitable chart. Machine learning techniques such as time-series forecasting and anomaly detection are invoked whenever the question calls for them.

The system is intentionally domain-agnostic. The same agent can analyze sales data, academic records or institutional statistics without being redesigned for each case, because the intelligence lies in the process of analysis rather than in any dataset-specific logic.

II. PROBLEM STATEMENT

SMEs and educational institutions in India hold large amounts of tabular data but lack analysts to interpret it, and existing chat-with-your-data tools cannot handle its inconsistent formatting. An accessible, domain-agnostic system is needed that lets non-technical users get accurate, explainable answers from their own data.

III. AIM, OBJECTIVES AND SCOPE

Aim

To design and develop an AI-driven Data Analyst Agent that enables users, particularly Indian SMEs and educational institutions, to upload tabular data and interact with it in natural language, receiving accurate, explainable insights and visualizations without in-house data analysis expertise.

Objectives

1) To design a multi-agent architecture that interprets a natural language question and autonomously plans the analysis.
2) To develop a data profiling module that inspects the structure, data types and quality (missing values, inconsistencies) of an uploaded dataset before analysis.
3) To implement a code generation and execution pipeline that runs the analysis plan in a secure, sandboxed environment.
4) To integrate time-series forecasting, anomaly detection and clustering to support a range of analytical questions.
5) To design an explanation module that converts numeric results into plain-language insights for non-technical users.
6) To recommend charts automatically, based on the nature of the result, so that insights are presented both textually and visually.

7) To ensure the system generalizes across domains (sales, academic, institutional) without redevelopment.
8) To incorporate security safeguards, namely sandboxed execution, input sanitization and access control, for safe handling of user data and AI-generated code.

Scope

The work covers a functional web-based prototype that accepts structured tabular uploads (CSV and Excel) and supports natural language question answering through a conversational interface. It includes automated profiling, analysis planning, code generation and sandboxed execution; machine learning analysis for forecasting, anomaly detection and clustering questions; plain-language explanations with matching visualizations; and session context so that follow-up questions can be asked on the same dataset. The prototype is to be demonstrated on representative Indian SME-style data (for example synthetic Tally or sales exports) and institutional data (for example academic and placement records).

IV. MOTIVATION

The motivation for this work is a simple observation: data has no value unless it can be interpreted, and interpretation has been gated behind specialized skill. Business owners and administrators often want to know why sales dropped in a particular month, which student cohort is underperforming or which product is losing traction, but lack the tools to ask. Rather than building another static dashboard that requires the user to know what to look for, this work aims to build a system that reasons about the question itself. It also aims to move beyond the shallow upload-a-CSV-and-chat implementations common in tutorials, by grounding the system in an underserved real-world use case and by adding genuine safeguards such as sandboxed code execution.

V. PROPOSED SYSTEM AND METHODOLOGY

The proposed system is a complete, self-contained web application rather than a developer library. Its core is a multi-agent architecture in which distinct AI agents each handle one part of the analysis: understanding and planning the question, generating and executing code, applying machine learning where appropriate and explaining the result. This mirrors how a human analyst approaches an unfamiliar question, instead of relying on a single AI call to produce an answer directly.

Well-defined analytical tasks such as forecasting and anomaly detection are handled by dedicated machine learning models, not only by AI-generated code, which gives more reliable and reproducible results than a purely LLM-driven approach. All AI-generated code runs in an isolated, sandboxed environment. Each user query follows a seven-step workflow:

Step 1, Data ingestion: the user uploads a CSV or Excel file, which is validated and stored temporarily for the session.

Step 2, Data profiling: columns, data types, missing values and basic statistics are analyzed to create a dataset profile.

Step 3, Query understanding: the Query Planner Agent interprets the question and identifies the required analysis, such as trends, forecasting, anomalies or correlations.
Step 4, Code generation and execution: the Code Generator Agent writes Python and Pandas code, which is executed in an isolated Docker container.
Step 5, ML processing: for applicable queries, Prophet, Isolation Forest or K-Means is used for forecasting, anomaly detection or clustering.
Step 6, Insight and visualization: the Insight Explainer converts results into simple language, and the Chart Recommender selects a suitable visualization.
Step 7, Response delivery: insights and visualizations are shown in the chat interface, and session context is kept for follow-up questions.
Overall, the workflow ensures that responses are based on actual computation on the uploaded dataset rather than on unsupported AI-generated assumptions.

VI. SYSTEM ARCHITECTURE AND MODULES

The architecture is shown in Fig. 1. The user interacts with a React frontend, which communicates with a FastAPI backend. The backend performs data profiling, routes requests and stores data in a PostgreSQL database. Requests are passed to the agent orchestrator, which plans, generates and explains, and which delegates execution to two components: the sandboxed code execution environment and the machine learning models.

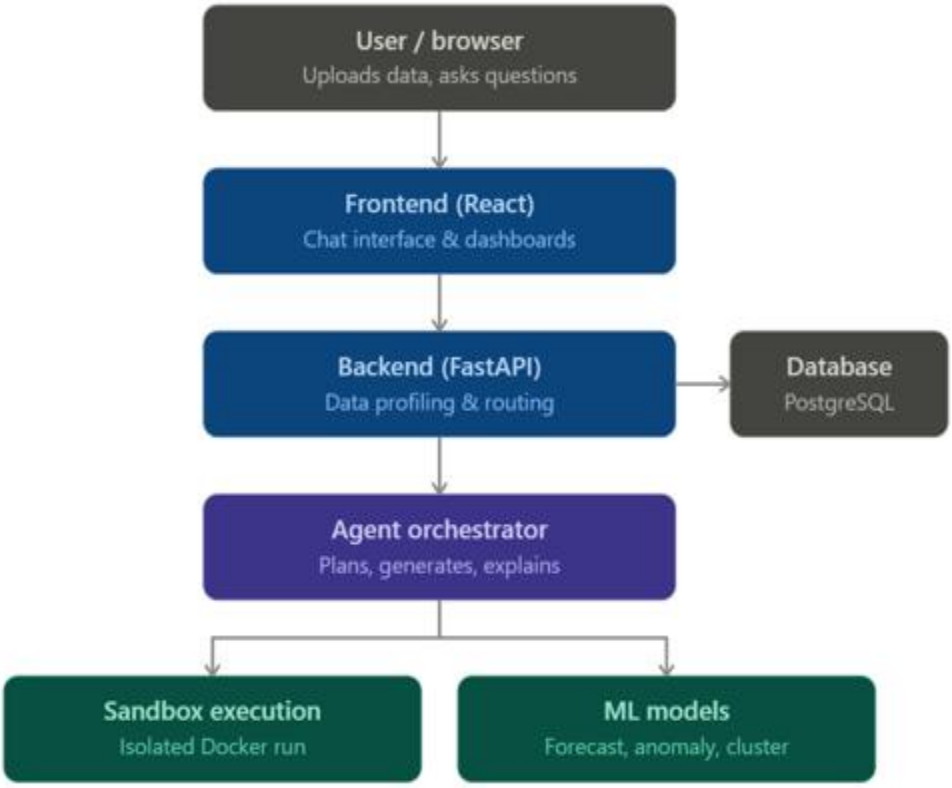

Fig. 1. System architecture of the proposed AI Data Analyst Agent

The responsibilities of each module are listed in Table 1.

Table 1 Module descriptions

Module	Responsibility
Frontend (React)	Chat interface for asking questions, file upload controls, and rendering of the dashboards and charts returned by the backend.
Backend (FastAPI)	Handles authentication, receives uploaded files, performs data profiling, routes requests to the agent orchestrator and manages database interactions.
Agent orchestrator	Coordinates the four sub-agents (Query Planner, Code Generator, Insight Explainer and Chart Recommender) and manages the reasoning flow from question to final explanation.
Sandbox execution	Runs all AI-generated code inside an isolated Docker container with no external network access and enforced time limits, ensuring safety regardless of code origin.
ML models	Provides dedicated statistical and ML capability: Prophet for forecasting, Isolation Forest for anomaly detection and K-Means for clustering, trained fresh on the uploaded dataset when required.
Database (PostgreSQL)	Stores user accounts, uploaded dataset metadata and chat and session history for context retention across follow-up questions.

VII. IMPLEMENTATION ENVIRONMENT

The proposed implementation uses an entirely free and open-source stack, with the exception of calls to the hosted language model.

Frontend: React.js, Recharts and Axios.

Backend: Python 3.10 or later with FastAPI.

AI agent and LLM: Gemini API (Flash) with LangGraph or AutoGen for agent orchestration.

Machine learning: Scikit-learn [7], Prophet [6], Pandas and NumPy.

Database and security: PostgreSQL for storage and Docker for sandboxed execution.

Tools and platforms: Git and GitHub for version control; VS Code or PyCharm; Windows 10 or 11, Linux or macOS.

Hardware: Intel i5 (8th generation) or equivalent, or AMD Ryzen 5 or higher; 8 GB RAM minimum (16 GB recommended); 10 GB free storage; an internet connection for Gemini API calls.

VIII. NOVELTY OF THE APPROACH

Domain-focused: targets messy SME data such as Tally, GST and institutional records.

Multi-agent AI: four specialized agents handle planning, coding, insight generation and visualization.

Built-in machine learning: Prophet, Isolation Forest and K-Means provide forecasting, anomaly detection and clustering.
Secure execution: AI-generated code runs only inside a sandboxed environment.
Domain-independent: the same agent works across business, academic and institutional datasets.
Open source: a free and open-source stack keeps costs low for small organizations.

IX. EXPECTED OUTCOMES AND APPLICATIONS

The project is expected to deliver a functional web-based data analysis application that produces accurate, computation-backed answers with automatically generated charts, demonstrated on datasets from more than one domain, with secure handling of AI-generated code and an evaluation of forecasting and anomaly detection performance.
Potential applications include: (a) SMEs, for sales, inventory and financial analysis; (b) educational institutions, for academic and placement analysis; (c) retail and e-commerce, for sales trends and product performance; (d) non-profits, for survey and beneficiary data; (e) freelancers and consultants, for quick client data analysis; and (f) students and researchers, for exploratory data analysis.

X. ADVANTAGES AND LIMITATIONS

Advantages. Users need no coding or statistical expertise. The system works across domains without rebuilding. Answers are backed by real computation rather than AI-guessed numbers, sandboxed execution keeps AI-generated code safe, and the technology stack is free and open source.
Limitations. The system works only on structured tabular data (CSV and Excel), not on images or free text. Accuracy depends on data quality, and very messy files may need cleanup first. The free tier of the LLM API has rate limits. There is no real-time integration with live ERP or Tally systems, so data must be exported and uploaded. This version supports English-language questions only.

XI. CONCLUSION AND FUTURE SCOPE

This paper proposed an AI Data Analyst Agent that lets non-technical users in Indian SMEs and educational institutions analyze their own tabular data through natural language. By combining a multi-agent reasoning pipeline, dedicated machine learning models and sandboxed code execution, the design aims to deliver explainable results that are grounded in actual computation. The architecture and modules described here provide the basis for the prototype that will be built and evaluated on representative SME and institutional datasets.
Future work includes direct integration with accounting and ERP tools such as Tally and Zoho Books, multilingual support for Hindi and Marathi, additional file types such as PDF tables and Google Sheets, scheduled and automated recurring reports, and deployment as a multi-tenant SaaS product for real institutional use.

ACKNOWLEDGMENT

The authors thank the Department of Information Technology, P.G. Moze College of Engineering, Wagholi, Pune, for its support and guidance during this project.

REFERENCES

[1]. L. Xie, C. Zheng, H. Xia, H. Qu, and C. Zhu-Tian, "WaitGPT: Monitoring and steering conversational LLM agent in data analysis with on-the-fly code visualization," arXiv preprint arXiv:2408.01703, 2024.
[2]. "LLM-based data science agents: A survey of capabilities, challenges, and future directions," arXiv preprint arXiv:2510.04023, 2025.
[3]. F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in Proc. IEEE Int. Conf. Data Mining (ICDM), Pisa, Italy, 2008, pp. 413–422.
[4]. Sinaptik AI, "PandasAI: Chat with your data," GitHub repository. [Online]. Available: https://github.com/sinaptik-ai/pandas-ai. [Accessed: 18-Sep-2026].
[5]. Julius AI, "AI data analyst," Julius AI. [Online]. Available: https://julius.ai. [Accessed: 18-Sep-2026].
[6]. S. J. Taylor and B. Letham, "Forecasting at scale," The American Statistician, vol. 72, no. 1, pp. 37–45, 2018.
[7]. F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.