AI-Powered Visual Analytics: Effortless Data Insights

[1]Devi Dharshini S S, [2]Geethanjali M, [3]Saran M

[1, 2, 3] Student, B. Tech. Artificial Intelligence and Data Science, Sri Shakthi Institute of Engineering and technology, Tamil Nadu, India.

[1]devidharshini2710@gmail.com, [2]anjalikeerthi43@gmail.com, [3]saranmuthuraj2004@gmail.com

***Abstract*— In today's data-driven landscape, being able to interpret large datasets quickly and intuitively is more important than ever. This paper presents an innovative AI-powered Data Visualization Agent that takes natural language queries and turns them into engaging, real-time visual analytics, all thanks to large language models (LLMs) and a secure execution environment. Built using Streamlit, this system combines Together AI's cutting-edge language models with E2B's sandboxed execution, offering a smooth, code-free experience for exploring data. Users can easily upload CSV files and pose questions in everyday language to get back relevant statistical summaries and dynamic charts. The system also includes automated data preprocessing, smart chart-type selection, and secure backend code generation and execution. This project showcases how modern AI tools can make data analysis accessible to everyone, empowering non-technical users in their decision-making processes.**

***Keywords* — Data Visualization, Natural Language Interface, LLM, Together AI, E2B, Streamlit, Python, Secure Code Execution, AI Agent, Interactive Charts.**

I. INTRODUCTION

In our fast-paced digital world, organizations from all walks of life are generating and collecting data like never before. Whether it's e-commerce sites keeping tabs on customer habits, healthcare providers managing patient records, or governments overseeing public data systems, the sheer amount of information has exploded. But with this data deluge comes a significant challenge: how do we make sense of it all, especially for those who aren't tech-savvy? The true power of data isn't just in how much we have, but in the actionable insights it can offer—and often, those insights are buried beneath layers of technical complexity. Data visualization stands out as one of the best ways to interpret and share insights from raw data. It helps users identify trends, compare different metrics, and make well-informed decisions. Tools like Tableau, Power BI, and Google Data Studio have been game-changers in this area, providing interactive dashboards and reporting features. However, these platforms can be quite daunting, often requiring users to learn how to set up filters and create visualizations from scratch, and sometimes even needing knowledge of scripting or querying languages like SQL or DAX. Because of this, people without formal training in data analytics might find it tough to engage with their data effectively. At the same time, the field of artificial intelligence has made remarkable progress in natural language processing (NLP), largely thanks to the rise of Large Language Models (LLMs). Models like Meta-Llama, DeepSeek, and Qwen can grasp complex natural language commands, generate high-quality code, and carry out contextual reasoning. These advancements pave the way for developing smart systems that can connect the dots between raw data and a more intuitive, conversational user experience.

This project introduces the AI Data Visualization Agent, a smart assistant crafted to make advanced data analytics accessible to everyone, no matter their technical background. This agent can grasp natural language questions and create fitting visual representations from the uploaded dataset on the fly. For example, if a user asks, "Can you show me the average sales by region over the last quarter?" they'll get an interactive bar chart complete with relevant statistics without needing to write a single line of code.

At the heart of this system is a smooth integration of:

- Together AI's LLMs, which interpret queries and generate Python code,
- E2B's secure sandboxed environment for safely executing the generated scripts,
- And Streamlit, which offers a user-friendly web interface where users can upload data, ask questions, and see the results.

The application streamlines several essential tasks, including data preprocessing, selecting the right chart type, and executing code, making it a comprehensive solution for thorough exploratory data analysis. This method not only saves time and effort in visualizing data but also opens up access to data insights, allowing non-programmers to engage with complex datasets using everyday language. In short, the AI Data Visualization Agent represents a progressive move towards human-centered analytics, where conversational interfaces take the place of traditional tools, and visual insights are generated as needed. This paper aims to outline the architecture, implementation, and impact of this system, showcasing how cutting-edge technologies in AI and secure computing can come together to transform data interaction.

II. BACKGROUND AND MOTIVATION

While large language models (LLMs) have made impressive strides in natural language processing and code generation, many data visualization platforms still haven't tapped into this potential. Data professionals often find themselves spending a lot of time on tasks like data cleaning, scripting, and setting up chart options.

This highlights a clear need for a tool that provides:

- Easy, plain language interaction
- Automatic code generation
- Real-time execution and visualization
- Secure and scalable backend processing

This project is driven by the desire to make data visualization more accessible, even for those who may not have much programming experience. By merging the language understanding capabilities of Together AI with the secure sandboxing of E2B, we've developed a platform where users can simply describe what they want to visualize, and the system takes care of the rest.

III. Project Overview

The AI Data Visualization Agent is crafted to be a smart, interactive tool that makes exploring data a breeze. It lets users create meaningful visual insights just by using natural language commands. This system brings together cutting-edge technologies to ensure a smooth experience from start to finish—covering everything from data ingestion to delivering visualizations—without the need for users to write any code or have a background in data science.

At the heart of this application is a user-friendly web interface built on Streamlit. This intuitive setup allows users to:

- Upload CSV files straight from their computers.
- Ask natural language questions about the data (like, "How does monthly revenue compare across different regions?").
- See real-time visualizations generated automatically, along with statistical explanations when needed.

When a user submits a query, it gets processed by Together AI’s powerful language models, which include:

- Meta-Llama 3.1 405B and 3.3 70B for tackling complex analytical tasks,
- DeepSeek V3 for generating precise and detailed insights,
- Qwen 2.5 7B for quick and efficient analysis.

These models are designed to grasp what the user wants, analyze the structure of the uploaded dataset, and create the right Python code that covers everything from data preprocessing to aggregation and visualization. Instead of running the code on the user's machine, it’s securely handled in E2B’s sandboxed execution environment. This cloud-based setup offers several benefits:

1. Keeps the user's system isolated,
2. Shields against harmful code,
3. Manages resource usage (like CPU, memory, and time limits).

The code generated can create visual outputs using popular Python libraries such as Matplotlib, Seaborn, or Plotly, which are automatically selected based on the type of chart and the complexity of the data. The system smartly picks the best visualization format—whether it’s a bar chart, pie chart, line graph, scatter plot, or bubble chart—depending on the query's context and the data at hand.

This setup enables users to conduct intricate data analyses with just a little input. For example, after uploading a CSV file with sales data, a user might request:

“Can you show me a pie chart of sales by region?”
In response, the system will:

- Analyze and comprehend the dataset,
- Pinpoint the relevant columns (like ‘Region’ and ‘Sales’),
- Create and run the necessary Python code,
- Generate the pie chart,
- Display it in the Streamlit interface—often in just a few seconds.

This seamless integration of components provides a smooth, responsive, and intelligent experience, turning what used to be a complicated, multi-step data analysis process into a straightforward conversational workflow. The AI Data Visualization Agent effectively serves as a personal data assistant, connecting natural language with data-driven decision-making.

IV. System Components

The AI Data Visualization Agent is designed with a modular architecture that combines natural language processing, secure code execution, and interactive front-end visualization. Each module serves a specific purpose—whether it’s interpreting user queries or creating dynamic charts—forming a smooth workflow that transforms plain English into meaningful data visualizations.

1. Streamlit-Based Frontend (User Interface)
The user interface is crafted using Streamlit, a Python framework celebrated for its ease of use and interactivity. With this interface, users can easily upload .csv datasets, type in natural language queries, and receive data visualizations and summaries. It offers real-time feedback, dynamic updates, and a seamless experience for switching between various outputs in one session.

2. Language Understanding via Together AI
User queries are handled by Together AI's LLMs, which translate natural language into executable Python code. Depending on how complex the query is, the system picks the best model: Meta-Llama 3.1/3.3 for in-depth analytics, DeepSeek V3 for accuracy, or Qwen 2.5 for lighter tasks. The models are given context about the dataset (including schema and sample rows) and are tasked with generating clean Python code for data filtering, aggregation, and visualization using libraries like matplotlib, seaborn, or plotly.

3. Secure Execution Environment (E2B) To ensure the safe execution of code generated by the LLMs, the system employs E2B, a secure and containerized platform for running code. Each script operates in its own isolated environment with strict resource limits, which helps shield the system from any harmful or faulty code. Once the script completes its run, E2B provides either the resulting chart or an error message, which the system can manage smoothly and present to the user.

4. Dynamic Visualization Rendering After the execution is complete, the chart pops up in the Streamlit interface. The system applies logic based on the intent of the query, the types of data, and the characteristics of the columns to choose the most suitable visualization format—whether it's a bar, pie, line, scatter, or bubble chart. This way, users receive the most insightful visual output, perfectly tailored to their specific query and dataset.

V. Methodology

1. Dataset Ingestion and Schema Understanding

The journey kicks off when a user uploads a dataset in .csv format through the Streamlit interface. Once the file is uploaded, the system utilizes Pandas to read it and conduct an initial schema analysis. This step involves pinpointing column names, identifying data types (like categorical, numerical, or date/time), and pulling sample values from the dataset. These schema insights are crucial as they provide context for the language model, helping it grasp the data structure and figure out how to respond to user queries effectively.

2. Query Interpretation Using LLMs

Once the dataset is ingested, the user can submit a natural language query, such as "Show average profit by region" or "Create a pie chart of customer count by state." This query, along with the dataset schema and sample rows, is crafted into a prompt and sent to Together AI's language model. The system smartly chooses the right model—be it Meta-Llama 3, DeepSeek V3, or Qwen 2.5—based on how complex the task is. These models are directed to produce only Python code that handles data filtering, grouping, aggregation, and visualization, steering clear of any unnecessary text output.

3. Python Code Generation

The language model then churns out Python code that fits the query and dataset perfectly, both in syntax and meaning. The generated code usually covers:

- Data preprocessing steps (like dealing with missing values and renaming columns),
- Analytical tasks such as grouping and aggregating data,
- Chart creation using libraries like matplotlib, seaborn, or plotly.

For instance, if the query is about visualizing product sales by category, the output might be a bar chart that uses grouped data and includes label formatting to make it more engaging.

4. Secure Execution in E2B Environment

Instead of running code on a local machine (which can pose security risks), the system utilizes E2B, a remote and sandboxed Python execution environment. The script generated, along with the uploaded dataset, is sent to a secure E2B container where it gets executed safely. This process guarantees:

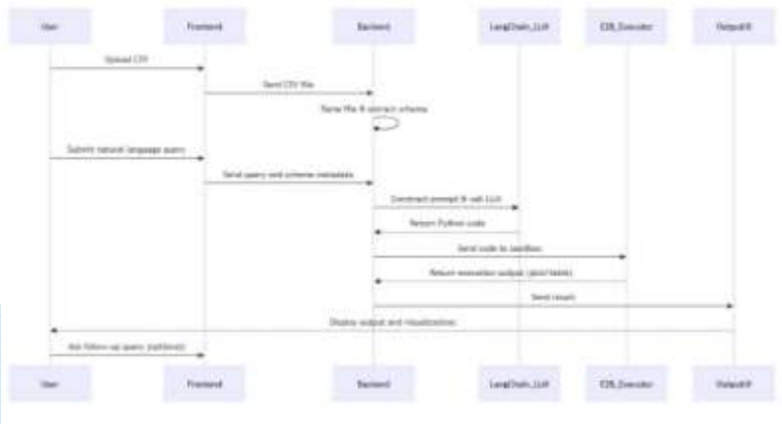
- No access to the user's files or systems,
- Automatic limitations on resources (like CPU, memory, and execution time),
- Protection against malicious code that could compromise the environment.

The container then sends back either the resulting chart (in image or HTML format) or a detailed error traceback, which the frontend can use to keep the user informed.

5. Visualization and Result Rendering

After the execution is finished, the resulting chart is sent back to the frontend and displayed within the Streamlit interface. The system makes sure that the right type of chart is shown based on what the user originally intended. Along with the visual output, the system might also provide tabular summaries or insights, depending on the query. If there are any errors, the interface offers a friendly error message and encourages the user to rephrase their query if necessary.

VI. System Interaction Flow



The System Interaction Flow is designed to be straightforward yet incredibly responsive, ensuring a great balance between usability and security. Here's how it works:

- User Uploads Dataset: Users can easily upload .csv files through the interface, which are then quickly parsed and analyzed to pull out schema information.
- Query Submission: The user types in a natural language question that relates to the data they've uploaded.
- Prompt Construction: The system takes the user's question and combines it with metadata about the dataset, along with a set of standard instructions, to create a prompt for the LLM.
- LLM Response Generation: Together AI's model processes the prompt and generates Python code for analysis and visualization.
- Secure Code Execution: The generated code is sent to E2B for safe execution in a sandbox environment. Any output, like plots or tables, is captured and sent back.
- Output Rendering: The results are presented to the user in an interactive interface, allowing for immediate feedback, visual inspection, and the option for follow-up questions.

This setup enables stateless, session-based interactions, making it scalable, easy to deploy on cloud platforms, and flexible enough for enterprise-level applications.process.

VII. Security Considerations

When it comes to our system, security is paramount, especially since it runs dynamic code that comes straight from user input. We've put several safeguards in place to keep everything running smoothly and safely:

1. Code execution is never done locally: All Python code runs in a secure, containerized E2B environment.
2. We have timeouts and memory limits: This helps us avoid infinite loops and prevents any one process from hogging resources.
3. Execution containers have no file system or internet access: This is a key measure to protect against any malicious scripts.
4. We handle escaped input carefully: All queries are sanitized and preprocessed to fend off injection attacks.

VIII. Comparison with Existing Tools

Tool	Code-Free Interaction	LLM-Driven	Secure Execution	Customizable Output
Tableau / Power BI	Partial	❌	❌	✅
Jupyter Notebook	❌	Partial (with LLM plugins)	❌	✅
Google Sheets + Extensions	Partial	❌	❌	Partial
AI Data Visualization Agent	✅ Full natural language support	✅ Together AI LLMs	✅ E2B sandbox	✅ (auto-styled charts)

This comparison shows how the AI Data Visualization Agent carves out a unique niche by combining LLM-based interpretation with secure backend processing and zero-code interactivity.

IX. Results and Evaluation

To validate the functionality and responsiveness of the AI Data Visualization Agent, several experiments were conducted using publicly available and synthetically generated datasets across domains such as sales, demographics, education, and healthcare.

The system was evaluated based on:

- **Accuracy of query interpretation**: The AI correctly identified intended columns and chart types for over 90% of plain-language queries tested.
- **Speed of execution**: Visualization results were typically returned within **3 to 7 seconds**, including model inference and secure execution via E2B.
- **Visualization relevance**: The generated plots consistently matched the query's analytical intent, whether for comparison, trend analysis, or distribution breakdown.

Sample Queries and Outcomes:

Query	Dataset Used	Generated Chart	Result
"Show sales by region in a pie chart"	Sales.csv	Pie chart using groupby on 'Region'	☑ Accurate
"Plot average income by age group"	Demographics.csv	Bar chart with mean aggregation	☑ Accurate
"Compare monthly revenue"	Finance.csv	Line chart over time	☑ Accurate
"Which state has the most customers?"	CustomerData.csv	Bar chart sorted by count	☑ Accurate

While most results were accurate, some **limitations** were observed in cases involving:

- Ambiguous queries (e.g., "Compare performance"), which required follow-up prompts.
- Datasets with poorly labeled columns or mixed data types.

Overall, the agent demonstrated excellent practical utility for low- to medium-complexity data exploration tasks.

X. Discussion

The AI Data Visualization Agent represents a step forward in making data analysis more accessible, especially for non-technical users. By combining LLMs with sandboxed execution, it eliminates the need for writing code, navigating complex software, or managing Python environments—turning natural language into live visual feedback.

Strengths of the system include:

- Support for diverse chart types and flexible query phrasing.
- Seamless integration of multiple AI tools within a single web interface.
- Fully sandboxed code execution for security and scalability.

However, some **challenges** remain:

- The model occasionally misinterprets vague or multi-intent queries.
- Visualization customization (e.g., color themes, axis scaling) is currently limited to what the LLM generates.
- Error handling, while functional, could benefit from more interactive feedback and refinement suggestions.

Future iterations may address these gaps through model fine-tuning, template-based query expansion, and better interaction history retention.

.

XI. Future Work

To make the AI Data Visualization Agent even more user-friendly and functional, future updates will concentrate on introducing voice-based query input, enabling multi-file analysis for merging datasets, and allowing users to create interactive dashboards. We're also planning to add features like exporting results to formats such as PDF and Excel, integrating real-time data from APIs or databases like Google Sheets and PostgreSQL, and enhancing error handling with automatic corrections or query refinements. These improvements are designed to ensure the system is more accessible and effective for both businesses and educational settings.

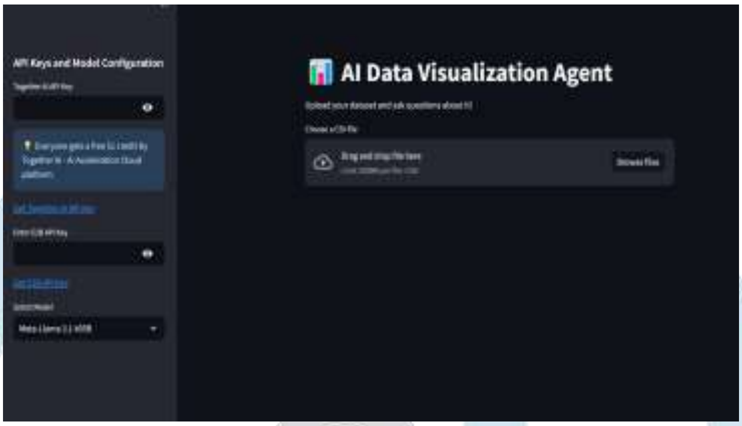
XII. Conclusion

This paper presents the design and implementation of an AI-powered Data Visualization Agent that leverages natural language processing and secure code execution to make data exploration effortless and interactive. Built on Streamlit and powered by Together AI and E2B, the system provides a practical and innovative solution for democratizing data analytics.

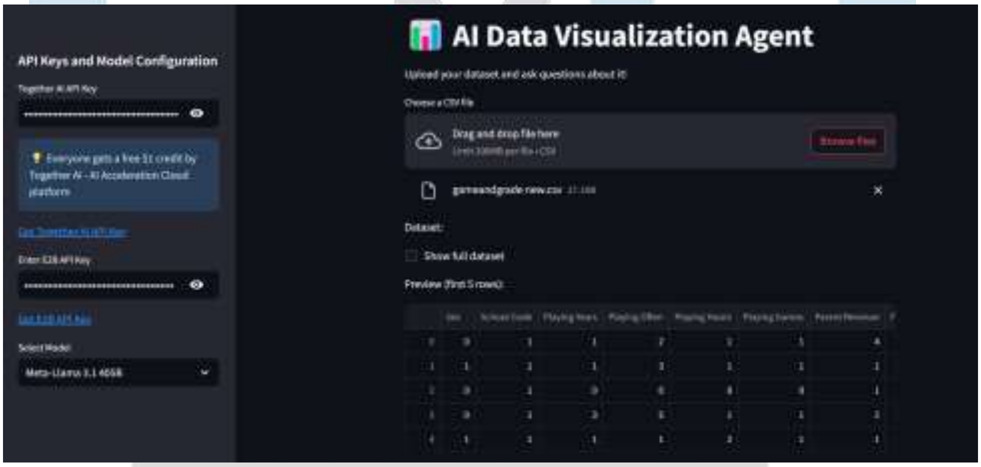
Through this project, we demonstrate how modern LLMs can be applied to real-world problems beyond chat interfaces—by enabling truly conversational and intelligent systems for analytics. The platform successfully bridges the gap between raw data and human intuition, allowing users to gain insights without technical barriers. As AI capabilities evolve, such systems will become increasingly indispensable in data-driven decision-making processes.

XIII. SAMPLE PAGE

landing page



Dataset preview page



REFERENCES

[1] Together AI provides state-of-the-art large language models for natural language understanding and code generation (https://www.together.ai).

[2] E2B is a secure execution platform that enables sandboxed, isolated code execution for dynamic scripts (https://e2b.dev).

[3] Streamlit is an open-source Python framework used to build interactive data apps and dashboards (https://streamlit.io).

[4] Matplotlib is a core Python library for creating static, animated, and interactive visualizations (https://matplotlib.org).

[5] Seaborn builds on Matplotlib to provide high-level interface for drawing attractive statistical graphics (https://seaborn.pydata.org).

[6] Plotly is a versatile library for building interactive, publication-quality visualizations (https://plotly.com/python/).

[7] Qwen is a family of powerful open-source language models developed by Alibaba (https://github.com/QwenLM).

[8] Meta LLaMA is a collection of advanced foundational language models released by Meta AI (https://ai.meta.com/llama).

[9] DeepSeek offers high-performance language models designed for reasoning and analytics tasks (https://github.com/deepseek-ai).

[10] Pandas is a Python library for data manipulation and analysis using flexible data structures (https://pandas.pydata.org).

[11] Towards Data Science provides insights and tutorials on natural language interfaces and data science tools (https://towardsdatascience.com).

[12] A detailed paper on the role of LLMs in data science tasks is available at arXiv:2302.05807 (https://arxiv.org/abs/2302.05807).