FSE 570: Capstone Project
Final Report presentation
Submitted to: Prof. Joshua Loughman
Arizona State University

PharmaFlow
Data-Driven Supply Chain Insights

Prajwal Anand, Swati Kashyap, Sudarshan Reddy S, Kalyani Tidke, Yogitha Reddy P
Arizona State University
Tempe, USA

Abstract

PharmaFlow is a Streamlit-based pharmaceutical supply chain analytics platform that integrates Google Sheets as a live backend, offering real-time data updates across its modules. The modular design includes demand forecasting (using SARIMA with grid search), price prediction (via SARIMAX), freight cost analysis, and a chatbot-powered analytics assistant that combines Gemini 1.5 Pro for deep queries and MiniLM with RAG for faster responses. The platform ensures robust data cleaning and outlier handling while enabling users to select specific SKUs for custom forecasts and cost analysis. While Google Sheets provides ease of use, scalability can be a concern; migrating to Firebase or BigQuery could improve performance. The NLP component could benefit from vector databases like FAISS for scalable document retrieval, and memory-aware conversations would improve user experience. Freight analysis can be extended with regression models and sustainability metrics. The platform would benefit from Streamlit multi-page navigation, session state, and user authentication. Audit logs, inventory optimization (EOQ, safety stock), and lead time simulations could further enhance value. Overall, PharmaFlow is a strong, extensible foundation for pharma analytics, combining classical time series forecasting, NLP, and a user-friendly UI, with room to grow into a production-grade decision support system. A REST API layer could enable broader integrations with ERP systems.

1. Introduction

The pharmaceutical supply chain operates as an intricate data-driven system which demands accurate management of demand forecasting and pricing strategies alongside shipment planning and operational analytics. The financial performance and patient outcomes suffer from delays and mispricing together with logistics inefficiencies. The Streamlit-based analytics platform PharmaFlow serves as a solution to these problems by delivering an intelligent and user-friendly interface for pharmaceutical supply chain real-time analysis. The platform uses Google Sheets as its dynamic backend for live data integration to provide modular functionalities that include SARIMA model-based demand forecasting and SARIMAX-based price prediction and shipment mode and freight cost analysis and a chatbot analytics assistant that uses Gemini 1.5 Pro and MiniLM models with retrieval-augmented generation (RAG). The system provides strong data cleaning capabilities and outlier management and SKU-specific analysis through a lightweight and scalable architecture. The report shows the entire process starting from data intake and preparation through to forecasting and NLP implementation and user interface development. The paper identifies potential system improvements through vector database integration and cloud database scalability and advanced inventory planning methods. The pharmaceutical supply chain benefits from PharmaFlow as a major advancement in digital supply chain optimization.

2. Problem statement and objectives

2.1. Problem statement

Multiple essential inefficiencies exist within the pharmaceutical supply chain which reduce its operational effectiveness. The process of inaccurate demand forecasting leads to either excessive stockpiling or inventory depletion which disrupts supply operations and elevates costs. Financial inefficiencies become worse because pricing strategies that do not adjust to market signals in real time. The selection of shipment modes without standardization results in delivery delays and increased costs which affect timely delivery. The supply chain suffers from two main problems: the lack of real-time intelligent analytics which reduces operational visibility and prevents data-driven decision-making.

2.2. Objectives

Develop an interactive web-based analytics dashboard using Streamlit, integrating live data from Google Sheets as a backend to enable dynamic updates. Implement SARIMA-based demand forecasting and SARIMAX-based price prediction to enhance decision-making accuracy. The platform should also support shipment mode analysis and provide clear visualizations of freight costs. To further enrich user experience, incorporate a chatbot interface powered by Gemini 1.5 Pro and MiniLM using Retrieval-Augmented Generation (RAG) for contextual insights. The system should be built on a modular architecture to ensure scalability and flexibility for future expansion.

3. Literature Review / Background

Data science has revolutionized supply chain operations during the last few years particularly in pharmaceuticals because precision remains essential for this sector. The traditional ERP systems maintain transactional data storage functions yet they lack the dynamic capabilities of AI-driven analytics. The combination of Python with Streamlit and Google Sheets cloud collaboration tools enables developers to create modern lightweight analytics solutions that scale efficiently. Time series forecasting models SARIMA and SARIMAX demonstrate effective results in forecasting pharmaceutical market demand together with pricing trends. The NLP models Gemini and MiniLM provide advanced capabilities for data interaction through natural language-based queries. The project unifies these tools into a pharmaceutical supply chain platform which meets specific industry requirements.

4. Architecture Overview

PharmaFlow functions as a modular Streamlit application which links to a real-time Google Sheets backend. The system contains multiple essential architectural components. The Streamlit dashboard of the frontend contains multiple tabs which display visualizations and text entry fields and dropdown selection options for user interaction. The application uses gspread and pandas libraries to manage data retrieval and modification functions which enable Google Sheets to operate as a simple database system. The application uses SARIMAX models to predict prices and SARIMA models to forecast demand and analyze costs for analytics purposes. The Retrieval-Augmented Generation (RAG) system uses Gemini and MiniLM to power its chatbot interface which allows users to query domain-specific data through intelligent queries. The modular design provides both scalability capabilities and enables straightforward addition of new features during upcoming development phases. The system uses Google Sheets as its backend which enables non-technical users to upload and review data through a simple interface that requires no additional infrastructure.

4.1 Storage Solutions

PharmaFlow uses Google Sheets as a cloud-hosted, spreadsheet-based backend to facilitate real-time data synchronization. The Sheets API is accessed securely through a service account managed via a JSON key. All data fetches and writes are performed using the gspread and pandas libraries.

A dedicated backend script handles:

- Authentication and authorization using OAuth2 credentials.
- Converting spreadsheet tables into structured pandas DataFrames for analysis.
- Appending new entries row-wise via validated inputs from the frontend.
- Supporting CSV bulk uploads with field mapping, logging failed rows, and enabling preview before final commit.

All updates made via the front end or directly in the Google Sheet are automatically synchronized across all tabs through reactive data polling mechanisms. This design removes the need for database hosting, making the system lightweight and easy to manage, especially for academic or MVP settings.

4.2 Forecasting Module (SARIMA & SARIMAX)

The platform incorporates time-series forecasting using SARIMA (Seasonal ARIMA) for demand prediction and SARIMAX (ARIMA with exogenous variables) for price forecasting. These models are implemented using the statmodels library and optimized via grid search for hyperparameter tuning.

The forecasting pipeline includes:

- Aggregation of shipment records into monthly or weekly series.
- Optional data transformation using log or Box-Cox methods.
- Model fitting and residual analysis.
- Future prediction over user-defined horizons.

Forecast outputs are displayed as Plotly line charts with confidence intervals and volatility bands. Performance metrics such as RMSE, MAE, and MAPE are computed and shown for model comparison. This module assists stakeholders in identifying upcoming trends and preparing for potential demand fluctuations or pricing shifts.

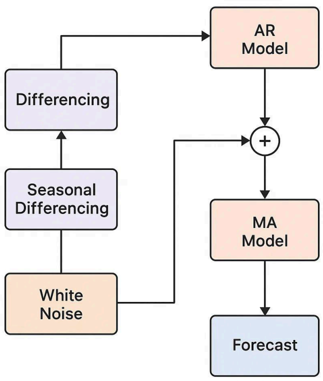

Fig 1: Workflow of SARIMA model

4.3 NLP-Assisted Chatbot Module

An integrated AI chatbot uses Retrieval-Augmented Generation (RAG) to provide instant answers based on underlying data. MiniLM embeddings are generated and indexed, while Gemini 1.5 Pro handles query generation. This allows users to ask domain-specific questions in natural language, removing the need for dashboard navigation.

4.4 Modular Code Structure

All functionalities are encapsulated in independent modules (e.g., forecasting_module.py, chatbot_module.py), managed through a central routing logic in app.py. This ensures code maintainability and makes it easy to introduce new modules without disrupting existing flows.

4.5 Security and Configuration

Sensitive information is stored securely in environment variables loaded via python-dotenv. Google API access is restricted to authorized service accounts. HTTPS and input validation mechanisms are used to ensure platform integrity. Logs capture any backend failures for troubleshooting.

5. Dataset

5.1 Dataset Overview and Statistics

The core dataset contains more than 2800 historical pharmaceutical shipments which are distributed across 35 countries while covering three product groups: ARV, HRDT and ACT. The dataset contains 2800 purchase order line items which store shipment date information along with quantity details and pricing data and freight expenses and additional metadata. The initial data cleaning process aimed to achieve both consistency and usability. The date fields received conversion through `pd.to_datetime(..., errors="coerce")` while rows with `NaT` in essential fields were removed. The numeric fields received processing through `pd.to_numeric(..., errors="coerce")` while missing values received imputation from median or default values. The freight cost data required a custom function `clean_freight_cost_column_with_id_priority()` and IQR-based capping which affected less than 10% of the data to prevent distortion. The "Unknown" value was used to fill missing entries in categorical variables so that row integrity would remain intact.

```
df.head()
```

	Project Code	PQ #	PO / SO #	ASN/DN #	Country	Managed By	Fulfill Via	Vendor INCO Term	Shipment Mode	PQ First Sent to Client Date	...	Manufacturing Site	First Line Designation	Weight (Kilograms)	Freight Cost (USD)	Line Item Insurance (USD)
0	100-CI-T01	Pre-PQ Process	SCMS-4	ASN-8	Côte d'Ivoire	PMO - US	Direct Drop	EXW	Air	Pre-PQ Process	...	Ranbaxy Fine Chemicals LTD	Yes	13.0	780.34	NaN
1	108-VN-T01	Pre-PQ Process	SCMS-13	ASN-85	Vietnam	PMO - US	Direct Drop	EXW	Air	Pre-PQ Process	...	Aurobindo Unit III, India	Yes	358.0	4521.50	NaN
2	100-CI-T01	Pre-PQ Process	SCMS-20	ASN-14	Côte d'Ivoire	PMO - US	Direct Drop	FCA	Air	Pre-PQ Process	...	ABBVIE GmbH & Co.KG Wiesbaden	Yes	171.0	1653.78	NaN
3	108-VN-T01	Pre-PQ Process	SCMS-78	ASN-50	Vietnam	PMO - US	Direct Drop	EXW	Air	Pre-PQ Process	...	Ranbaxy, Paonta Shahib, India	Yes	1855.0	16007.06	NaN
4	108-VN-T01	Pre-PQ Process	SCMS-81	ASN-55	Vietnam	PMO - US	Direct Drop	EXW	Air	Pre-PQ Process	...	Aurobindo Unit III, India	Yes	7590.0	45450.08	NaN

5 rows × 36 columns

Fig 2: Dataset

5.2 Data Preprocessing

The dataset underwent a systematic pre-processing workflow to ensure consistency, accuracy, and readiness for analysis and forecasting. Data was initially retrieved from a centralized Google Sheet using secure API authentication. Upon loading, column headers were standardized and stripped of unnecessary whitespace to ensure consistency across processing steps. Date fields such as delivery and scheduling dates were converted into proper datetime objects using pd.to_datetime(), with invalid entries safely coerced to NaT. Numeric columns, including weight and freight cost, were converted using pd.to_numeric(), and missing values were filled using either zeros or column medians based on domain context.

Freight cost normalization involved interpreting non-numeric values like “Freight included” or “Invoiced separately,” as well as resolving references such as “See ASN” or “ID#”. A custom routine, clean_freight_cost_column_with_id_priority, was employed to standardize these values and link them to appropriate records where needed. After interpretation, missing freight costs were imputed using the median to maintain distributional integrity.

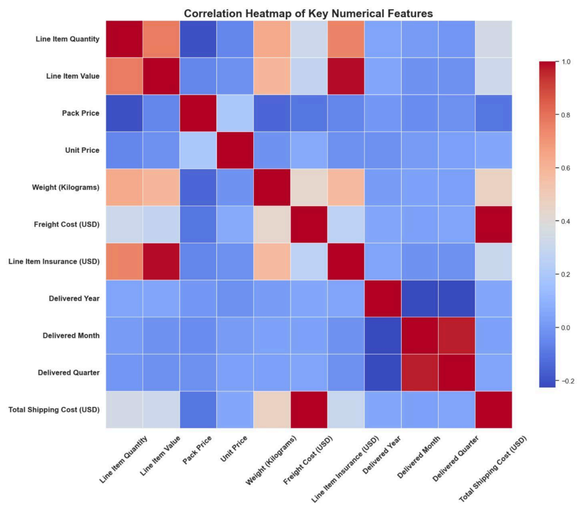

Fig 3: Correlation Heatmap of Key Numerical Features

For time series forecasting, the data was resampled to a weekly or monthly frequency, depending on use-case requirements. Outliers were identified and capped using an interquartile range (IQR)-based filtering approach to reduce skew from extreme values. Where necessary, log or Box-Cox transformations were applied to stabilize variance, and seasonal decomposition was used to isolate trends and residuals. The clean time series was then split into training and test datasets using an 80-20 ratio.

Fig 4: Scatter plot for Shipment weight vs Freight Cost

Finally, forecasting models were retrained on the full dataset to produce final predictions. Any statistical transformations were reversed, and seasonal patterns reintroduced to ensure interpretability. Additionally, missing values in categorical columns were filled with the placeholder “Unknown” to retain data completeness and avoid unintended exclusions during model training.

6. Approach and Methodologies

6.1 Live Data Sync via Google Sheets

PharmaFlow uses Google Sheets as its lightweight cloud-based database to achieve real-time data synchronization across the platform. The backend system relies on gspread, pandas and Streamlit to create a smooth connection between the frontend dashboard and the live data source. The google_sheets_loader.py module performs authentication tasks and transforms sheet data into pandas DataFrames for efficient processing. The append_row_to_sheet() function enables users to add new data points to the Google Sheet while showing UI feedback about submission success or failure.

The Streamlit frontend interface features a data-entry form with validated input fields and a CSV bulk upload function which allows users to preview uploaded data and logs any upload failures for transparency. The dashboard features a DataFrame that automatically refreshes its content so updates from form entries and bulk uploads appear instantly across all relevant tabs for real-time data visibility and consistency.

6.2 Front-End Interface (Streamlit)

The platform’s user interface is built using Streamlit, a Python-based framework that is known for its rapid deployment and clean layout design. Users interact with PharmaFlow through a multi-tab dashboard that houses key tools such as Demand Forecasting, Price Prediction, Freight Cost Analysis, Data Upload, and an AI Chat Assistant.

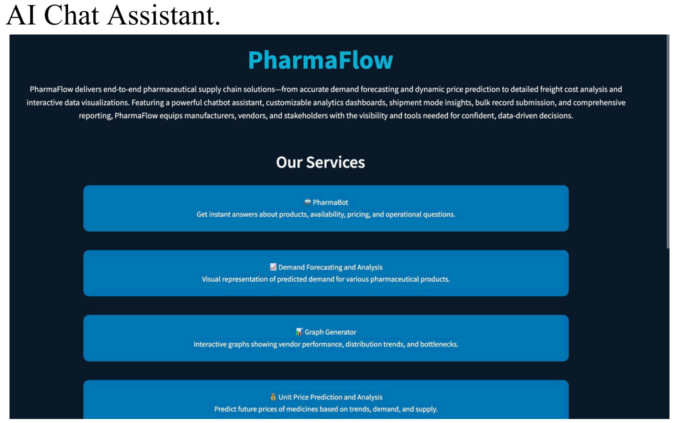

Fig 5: StreamLit web page

Each tab has intuitive elements such as dropdowns for filters, sliders for parameters, and upload buttons for bulk CSV submissions to reduce the friction in data exploration. The interface updates dynamically based on user input, ensuring real-time feedback without page reloads. To enhance usability and visual coherence, custom styling is applied, and essential state variables are retained across user sessions.

6.3 Chatbot-Assisted Analytics (RAG)

PharmaFlow implements Retrieval-Augmented Generation (RAG) to power its chatbot-assisted analytics system which operates through a secure efficient intelligent framework. The system uses credential-based protection through a secure API and environment variables stored in a `.env` file. The `SentenceTransformer('all-MiniLM-L6-v2')` model generates sentence embeddings for natural language understanding which the system caches to optimize its performance.

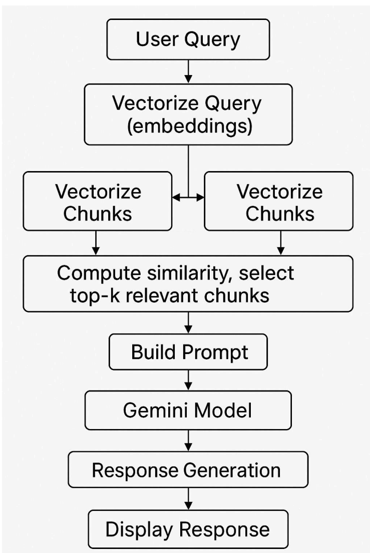

Fig 6: Workflow of chat-bot

The data processing pipeline implements chunking strategy by dividing the DataFrame into segments of 200 rows which gets converted to JSON for field-aware and structured data segmentation. The system retrieves similarities through cosine similarity and selects the top-K chunks for relevance. The system uses Gemini 2.0 Flash as its core model which users access through `google.generativeai.GenerativeModel`.

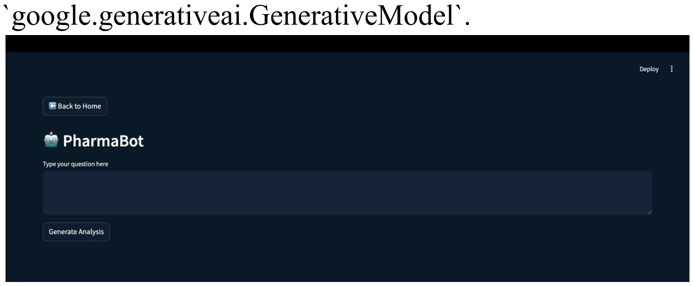

Fig 7:Chat-bot UI

The model receives prompts that both recall the embedded schema and offer specific instructions to prevent hallucinations while generating output that remains concise analytical and fact-based. Streamlit enables real-time AI-generated insight rendering through its dynamic chat UI which provides users immediate access to actionable intelligence from the underlying data.

6.4 Demand Forecasting

PharmaFlow starts its demand forecasting process by running an extensive preprocessing phase to achieve data quality and consistency at the highest level. The filtering process removes null values while IQR-based methods handle outliers to maintain dataset integrity below 10% distortion. The delivery quantities undergo resampling for monthly calendar periods to create meaningful time intervals for analysis. The data preparation for modeling requires variance stabilization through logarithmic and Box-Cox transformation methods which store the transformation parameter (λ) for future reproducibility. The `seasonal_decompose()` function removes seasonal patterns to reveal trends and eliminate cyclical noise in the data.

Fig 8: Demand Forecasting UI

The forecasting depends on a Seasonal AutoRegressive Integrated Moving Average (SARIMA) model. The model selection process uses a systematic grid search of $(p,1,q) \times (P,1,Q,12)$ parameter space to find the best model according to the Akaike Information Criterion (AIC) which balances model fit and complexity. Backtesting with an 80/20 training-test split is used for model validation while standard evaluation metrics RMSE, MAE and MAPE assess performance. The forecast duration extends from 1 month to 6 months with the default base forecast set at 3 months.

PharmaFlow produces interpretable visualizations through charts that show average monthly delivery quantities and breakdowns of the top five manufacturers and vendors and site–vendor pairs. The visual aids assist stakeholders to recognize patterns and trends and essential demand influencers which makes the forecasting results both actionable and simple to understand.

6.5 Price Prediction

PharmaFlow's price prediction module provides exact time-accurate unit price predictions for pharmaceutical shipments. The system starts with thorough data cleaning to maintain data consistency by reusing harmonized freight values. The interquartile range (IQR) trimming technique is used to remove outliers from the `Unit Price`, `Weight`, and `Freight Cost` columns without distorting the data distribution. The cleaning phase prepares reliable representative data for model training.

The cleaned data undergoes aggregation through weekly unit price resampling which uses the "Delivered-to-Client" date. Weekly resampling of the data smooths away daily fluctuations while preserving important patterns in pricing behavior across time. The model calculates average unit prices for each week to establish a stable time series which functions well for predictions.

SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous variables) functions as the forecasting model with parameters set to SARIMAX(1,1,1)x(0,1,1,52). Weekly data benefits from this specification because the seasonal component (52) detects annual patterns. The model includes:

The AR(1) component represents an autoregressive relationship which tracks price changes between successive weeks. The I(1) parameter implements differencing to achieve stationarity in the series. The MA(1) component functions to represent both shocks and noise in the data. The Seasonal (0,1,1,52) component uses seasonal differencing and MA components to detect annual repeating patterns.

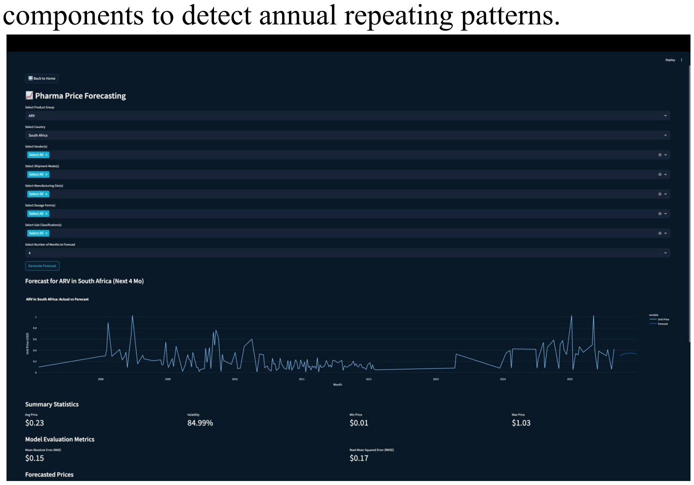

Fig 9: Price Prediction UI

The SARIMAX model receives selection because it handles trends alongside seasonality and supports exogenous variable extensions despite their current non-use. Backtesting of the model occurs before deployment when historical data exceeds 20 weeks in length. The dataset undergoes partitioning into training data and testing data with an 80/20 distribution. The model's performance evaluation depends on Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) calculations to determine forecast precision.

Through PharmaFlow's user interface, users can interact with the model by using filters that include product group and country and vendor and more. The results display actual unit price data alongside forecasted values through line charts and include projected price tables for future weeks along with statistical data. A seasonal profile chart reveals the typical unit price for each month throughout the year which helps users detect recurring price patterns and long-term price changes. The comprehensive statistical approach creates pricing forecasts for PharmaFlow which both users and analysts can understand and use effectively.

6.6 Shipment Mode & Freight Cost Analysis

Through the Shipment Mode and Freight Cost Analysis module of PharmaFlow users can obtain detailed insights about logistical efficiency together with cost distribution across different transportation modes. The initial step involves data cleaning to standardize freight cost values and filling missing shipment mode entries with "Unknown" to preserve data integrity. Users can use filters which include country and product group to narrow down the analysis for particular operational segments.

The dashboard presents essential performance indicators (KPIs) that show both the complete shipment numbers and the distribution percentage between air, sea and land transportation modes. Users can perform quick cost benchmarking through a comparative bar chart that displays average freight costs by mode.

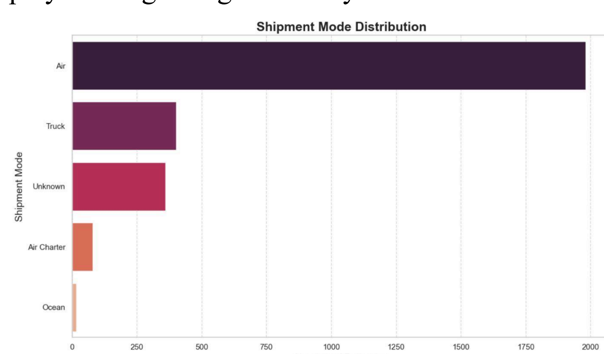

Fig 10: Shipment Mode Distribution

The module enables deeper temporal analysis through trend visualizations which present weekly freight cost changes for each shipment mode through line charts to help users detect time-based patterns or anomalies. The small-multiple charts display monthly shipment cost behavior for each mode to help identify recurring cost patterns which support better shipment planning and budgeting decisions. The module provides logistics teams and supply chain managers with essential analytical tools to optimize their shipment strategies while minimizing freight-related inefficiencies.

7. Supply Chain Stakeholder Benifits

PharmaFlow provides customized advantages to pharmaceutical supply chain management stakeholders through its real-time analytics and predictive modeling and operational transparency capabilities. Procurement Managers use PharmaFlow to predict product-country demand which helps them plan their inventory proactively. The system allows users to find reliable vendors and manufacturing sites through performance metrics that track on-time delivery results. The SARIMAX-powered unit price forecasting system allows Finance and Pricing Teams to monitor price volatility and freight costs by shipment mode to manage total landed costs effectively.

Logistics Coordinators can optimize their shipment strategies by analyzing historical and seasonal freight costs across different transportation modes to detect inefficient usage of expensive air freight. The real-time dashboards and chatbot-based insights of PharmaFlow enable Country Program Managers to answer donor and ministry inquiries without requiring dedicated data analysts for response. The system allows users to easily view delivery patterns and seasonal patterns which supports their ability to make informed decisions quickly.

Executive Leadership receives complete supply chain performance visibility through PharmaFlow across 35+ countries and multiple product categories. The platform provides executive leaders with summarized performance metrics that help them evaluate strategy and review network-wide performance. The platform's modular design together with its live clean dataset provides Data Analysts and Engineers with beneficial capabilities. The platform's configuration enables fast development and testing of new forecasting models while allowing users to integrate custom metrics and machine learning features which keeps PharmaFlow adaptable and forward-looking.

8. Future Enhancements

PharmaFlow's current architecture provides a strong base for future production-grade deployments with multiple planned enhancements to improve performance and scalability and enterprise integration. The backend system will transition from Google Sheets to Google BigQuery or Firebase as part of the planned upgrade to achieve faster querying and better support for large-scale operations. The analytics component will replace the current MiniLM-based vector search engine with more efficient solutions such as FAISS or Weaviate to improve both retrieval accuracy and response speed for complex queries.

Security and user experience improvements are also in focus. The planned features will introduce multi-user authentication and session persistence and audit logging capabilities to provide secure role-based access and traceability across the platform. The platform will integrate sustainability metrics including CO_2 emissions per shipment to enable environmentally conscious decision-making in logistics planning.

The development of a REST API layer will enable PharmaFlow to integrate smoothly with major Enterprise Resource Planning (ERP) systems including SAP and NetSuite. The planned enhancements will establish PharmaFlow as a powerful enterprise-ready decision support system which meets the changing requirements of worldwide pharmaceutical supply chains.

9. Challenges and Learnings

The core element of PharmaFlow's achievement was maintaining high data quality standards. The accuracy of forecasts and analyses depended heavily on resolving data inconsistencies and filling data gaps. The data preprocessing techniques included null value filtering and outlier management and data harmonization to create reliable and consistent data that reflected real-world conditions for modeling purposes.

The selection and tuning of suitable forecasting models required thorough experimentation to achieve the right balance between model complexity and interpretability. The SARIMA and SARIMAX models received individual optimization to reach their best performance levels. The implementation of grid searches together with backtesting allowed the selection of methods which provided the most accurate and actionable insights without showing signs of overfitting or underfitting.

The development of PharmaFlow required scalability to be a primary factor. The platform was built to manage growing data volumes while maintaining steady performance throughout its expansion. The models were designed for efficiency purposes to handle big datasets quickly without compromising accuracy. The system maintains its adaptability and robustness through its design which allows it to scale up for processing bigger supply chain datasets and new features in the future.

10. Conclusions

PharmaFlow AI successfully demonstrates how modern data science tools, combined with lightweight cloud infrastructure, can optimize pharmaceutical supply chain planning. Through robust data cleaning, live integration, AI-driven analytics, and predictive modeling, the platform enhances visibility, forecasts operational demand, and informs pricing strategies with statistical rigor. Its modular, scalable design positions it as a practical decision-support tool for pharmaceutical logistics management.

11. Github Link:

https://github.com/Praj460/Chatbot-for-supply-chain-Analysis/tree/main

References

[1] Box, G. E. P., Jenkins, G. M., & Reinsel, G. C. (2015). *Time Series Analysis: Forecasting and Control* (5th ed.). Wiley.

[2] Hyndman, R. J., & Athanasopoulos, G. (2019). *Forecasting: Principles and Practice* (2nd ed.). OTexts.

[3] Said, S. E., & Dickey, D. A. (1984). Testing for unit roots in autoregressive-moving average models of unknown order. *Biometrika*, 71(3), 599–607.

[4] Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with Python. *Proceedings of the 9th Python in Science Conference*.

[5] Wang, S., Wei, J., Wang, Y., Bao, H., Dong, L., & Zhou, M. (2020). MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *arXiv preprint arXiv:2002.10957*.

[6] Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. *arXiv preprint arXiv:1908.10084*.

[7] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[8] Google DeepMind. (2023). *Gemini: A Family of Multimodal LLMs*. Retrieved from https://deepmind.google/technologies/gemini

[9] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[10] Joblib Developers. (2023). *Joblib Documentation*. Retrieved from https://joblib.readthedocs.io/en/latest/

[11] McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*.

[12] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ... & Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585, 357–362.

[13] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95.

[14] Waskom, M. L. (2021). Seaborn: Statistical data visualization. *Journal of Open Source Software*, 6(60), 3021.

[15] Streamlit Inc. (2023). *Streamlit Documentation*. Retrieved from https://docs.streamlit.io/

[16] Google Developers. (2023). *Google Sheets API Documentation*. Retrieved from https://developers.google.com/sheets/api

[17] Gspread Project. (2023). *gspread Documentation*. Retrieved from https://docs.gspread.org/en/latest/

[18] Python Software Foundation. (2023). *python-dotenv Documentation*. Retrieved from https://pypi.org/project/python-dotenv/

[19] Hardeshana, D. (2021). *Supply Chain Shipment Pricing Data Analysis*. Kaggle. Retrieved from https://www.kaggle.com/code/divyeshardeshana/supply-chain-shipment-price-data-analysis