

Infosys Springboard Virtual Internship 6.0 Completion Report

Team Details <Do not mention any personally identifiable information like email ID, institute details, mobile phone number etc.>

Batch Number: 2 **SmartStock Inventory Optimization for Retail Stores**

Start date : 10-11-2025

Names:

1. AMAN KUMAR
2. SMRITI
3. H. VAISHNAVI
4. GIRIJA PAVANI
5. DURGA PRASAD

Internship Duration: 8 Weeks

1. Project Title

SmartStock Inventory Optimization for Retail Stores

2. Project Objective

The primary objective of SmartStock is to develop a comprehensive, AI-powered inventory optimization system that leverages Machine Learning to accurately predict sales demand and automate stock management, thereby minimizing revenue loss from stockouts and capital waste from overstocking. By integrating advanced analytics—including ABC classification, anomaly detection, and market basket analysis—with a real-time alerting mechanism (Email/SMS) and an intelligent inter-store transfer optimizer, the project aims to replace manual guesswork with data-driven precision. Ultimately, the system provides retail managers with a unified, interactive dashboard and a natural language chatbot to proactively monitor store health, balance inventory efficiently across locations, and drive operational excellence in a dynamic retail environment.

3. Project description in detail

SmartStock is a comprehensive, full-stack web application designed to modernize retail inventory management through the power of Artificial Intelligence and robust data analytics. In the competitive retail landscape, balancing supply and demand is a constant challenge; traditional reactive methods often lead to costly **stockouts** or wasteful **overstocking**. SmartStock addresses these inefficiencies by transitioning inventory control to a predictive, data-driven model. Built on a secure 3-tier architecture, the system utilizes a high-performance **FastAPI (Python)** backend, a persistent **MySQL** database, and a responsive, interactive frontend to provide a unified solution for chain-store operations.

At the core of the system lies an intelligent **Machine Learning forecasting engine**. Leveraging the HistGradientBoostingRegressor algorithm, SmartStock processes historical sales data, seasonal trends, and promotional factors to predict future demand with high accuracy. This predictive capability allows store managers to anticipate sales volume for specific items days in advance, rather than relying on gut feelings or static spreadsheets. By forecasting demand at both the store and item level, the application enables proactive decision-making, ensuring that the right products are available on the shelves exactly when customers need them.

To ensure operational continuity, the system features a rigorous **Real-Time Monitoring and Alerting** module. The application continuously analyzes the gap between current stock levels and predicted demand, automatically classifying inventory into critical statuses such as "Stockout," "Understock," or "Overstock." When a critical threshold is breached, the system triggers immediate notifications via **Email** and **SMS (Twilio integration)**. This automated watchdog ensures that administrators are instantly aware of supply chain risks, allowing them to intervene rapidly before revenue is lost or capital is tied up in excess inventory.

Beyond basic tracking, SmartStock integrates a suite of **Advanced Analytics and Optimization** tools to uncover deeper business insights. This includes **ABC Analysis** to prioritize high-value items, **Anomaly Detection** to flag irregular sales spikes or potential theft, and **Market Basket Analysis** to discover correlation patterns between products.

Furthermore, the system includes a unique **Inter-Store Transfer Optimizer**, which algorithmically identifies stores with excess inventory and pairs them with stores facing shortages. This feature recommends efficient stock transfers, allowing the retail chain to balance its inventory internally without incurring the cost of new purchase orders.

The user experience is centralized in a modern, theme-aware **Web Dashboard** that visualizes these complex data points through intuitive charts and dynamic tables. To make data accessibility even more seamless, the platform includes an **AI-Powered Chatbot**. This natural language interface allows users to simply ask questions—such as *"Predict sales for Grocery 1 in Store 1"* or *"Show me stock levels"*—and receive instant, data-backed responses. By combining predictive AI, automated alerts, and strategic analytics, SmartStock empowers businesses to minimize costs, maximize revenue, and achieve operational excellence.

4. Timeline Overview

Week	Activities Planned	Activities Completed
Week 1	Requirement gathering, project scope definition, and resource allocation	Requirements finalized for inventory optimization, project scope documented, team roles assigned
Week 2	Research on forecasting models and selection of development tools/tech stack	Selected HistGradientBoostingRegressor, finalized stack (Python, FastAPI, MySQL, Chart.js)
Week 3	Database design and setup for sales and inventory data	MySQL schema designed for SalesRecord/Inventory tables and tested connection
Week 4	Data collection, preprocessing, and integration with dataset	Historical sales dataset loaded from Hugging Face, cleaned, and baseline inventory levels calculated
Week 5	Model development: Training ML algorithms on demand forecasting	ML pipeline built, model trained on sales data, and evaluated for accuracy (RMSE/MAE)
Week 6	Development of user interface (dashboard, charts, visualization)	Frontend built with Login, Auto Stock Alerts, and Interactive Charts
Week 7	System integration: Linking ML model, database, and UI	Integrated FastAPI with Frontend, enabled real-time Email/SMS alerts and Chatbot logic
Week 8	Testing, debugging, documentation, and final project deployment	API testing completed, bugs fixed, code documented, and project ready for demo

5a. Key Milestones

Milestone	Description	Date Achieved
Project Kickoff	Project kickoff defined objectives, scope, and tech stack (FastAPI, ML, MySQL).	14 – Nov - 2025
Prototype/First Draft	Developed the prototype for demand forecasting and real-time stock alerting.	20 - Nov - 2025
Mid-Term Review	Completed mid-term review to assess ML model accuracy and refine dashboard UI.	4 - Dec - 2025
Final Submission	Submitted the completed SmartStock project with full analytics, chatbot, and optimization features.	20 - Dec - 2025
Presentation	Prepared and delivered the project presentation showcasing the inventory optimization outcomes.	30 - Dec - 2025

5b. Project execution details

1. Requirement Analysis – Gathered retail inventory needs, defined project scope for demand forecasting, and finalized optimization objectives.

2. Technology Selection – Chose AI/ML models (HistGradientBoostingRegressor), Python (FastAPI) for backend, HTML/JS for frontend, and MySQL for database.

3. System Design – Designed database schema for sales and inventory, workflow diagrams for stock alerts, and UI mockups.

4. Data Collection – Gathered historical store sales datasets from Hugging Face and generated inventory baseline records for testing.

5. Data Preprocessing – Cleaned data, engineered date-based features, and structured datasets for machine learning training.

6. Model Development – Trained demand forecasting models, evaluated accuracy (MAE, RMSE), and implemented prediction pipelines.

7. Frontend Development – Built interactive dashboards with real-time stock alert tables, analytical charts (ABC/Anomaly), and a chatbot interface.

8. Backend Development – Developed RESTful APIs, linked ML models with the UI, and integrated Email/SMS notification services.

9. Testing & Integration – Conducted API testing, system integration for transfer logic, and user acceptance testing; resolved bugs.

10. Deployment & Documentation – Configured local deployment servers, prepared API documentation, and presented the final project demo.

6. Snapshots / Screenshots

Step 1 : Go to Github and clone the Repository in your system then open in VS Code to do the execution.

[AMAN KUMAR/SmartStock-Inventory-Optimization-for-Retail-Stores-Infosys](#)

Step 2 : Start the backend server with `uvicorn main:app --reload` and open `index.html` (or `auth.html`) via a local server (e.g., `python -m http.server`).

```
○ amankumar@Amans-MacBook-Pro backend % uvicorn main:app --reload
INFO: Will watch for changes in these directories: ['/Users/amankumar/Desktop/untitled folder/backend']
INFO: Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO: Started reloader process [54416] using WatchFiles
INFO: Started server process [54419]
INFO: Waiting for application startup.
ML model trained. Metrics: {'mae': 92.36543528961042, 'rmse': 256.8087894047711, 'r2': 0.8299335560352237}
INFO: Application startup complete.
```

Step 3: Here is the **Login/Register Page** where the user needs to authenticate to access the secure inventory dashboard.

Login

Email

you@example.com

[Forgot password?](#)

Password

.....

Login

[New user? Register here](#)

Register

Name

Your name

Email

you@example.com

[Forgot password?](#)

Password

.....

Register

[Already have an account? Login](#)

Step 4: After logging in, the user lands on the **Auto Stock Alerts** view. Here, they can toggle Email/SMS notifications and trigger immediate stock checks. Users can click Export to CSV to generate a downloadable report of current stock alerts and critical shortages for offline analysis.

Notification Control Center
 Click the boxes below to toggle Email or SMS on/off. Then click "Send Now".

Email Alerts
OFF
 Inactive

SMS Alerts
OFF
 Inactive

Action
Send Now
 Trigger Alerts

Auto Stock Alerts (Overstock / Understock)

Load Alerts
 ☐ Auto-refresh
 Every 30 sec
 Search store or family
 Search
 Clear

All statuses
 Sort: default

Last updated: 4:51:03 PM
Shortage / Excess (units)
 Overstock Understock OK / Balanced

Details

Store	Family	Current Stock	Predicted Demand	Status	Shortage / Excess
1	AUTOMOTIVE	2.57	2.46	OK	0.11
1	BABY CARE	0	2.46	UNDERSTOCK	-2.46
1	BEAUTY	1.66	2.46	UNDERSTOCK	-0.8
1	BOOKS	0	2.46	UNDERSTOCK	-2.46
1	BREAD/BAKERY	329.98	288.35	OK	41.63
1	CELEBRATION	0	2.46	UNDERSTOCK	-2.46
1	CLEANING	776.11	816.02	OK	-39.91
1	DAIRY	452.61	229.65	OVERSTOCK	222.96

Export to CSV

Critical Alerts (High Risk Items)

All stores ↕

All critical ↕

Showing 1621 critical items.

Store	Family	Derived Status	Current Stock	Predicted Demand	Shortage / Excess
45	GROCERY I	OVERSTOCK	8,070.26	3,055.99	5,014.27
48	GROCERY I	OVERSTOCK	6,297.81	2,822.54	3,475.27
50	GROCERY I	OVERSTOCK	5,889.04	2,576.72	3,312.32
44	GROCERY I	OVERSTOCK	9,738.34	7,544.57	2,193.77
44	CLEANING	OVERSTOCK	3,142.2	979.93	2,162.27
21	GROCERY I	STOCKOUT	0	2,157.93	-2,157.93
6	GROCERY I	OVERSTOCK	4,289.23	2,496.23	1,793
47	CLEANING	OVERSTOCK	2,687.01	926.07	1,760.94

Step 5: The **Summary Dashboard** showcases high-level KPIs, including total Stockout, Understock, and Overstock items, along with a visual breakdown of critical inventory health.

Summary Dashboard

All stores ↕

All statuses ↕

Summary is calculated from the latest Auto Stock Alerts data. Please load alerts first if everything shows "-".

Understock Items

185

Overstock Items

1066

Stockout Items

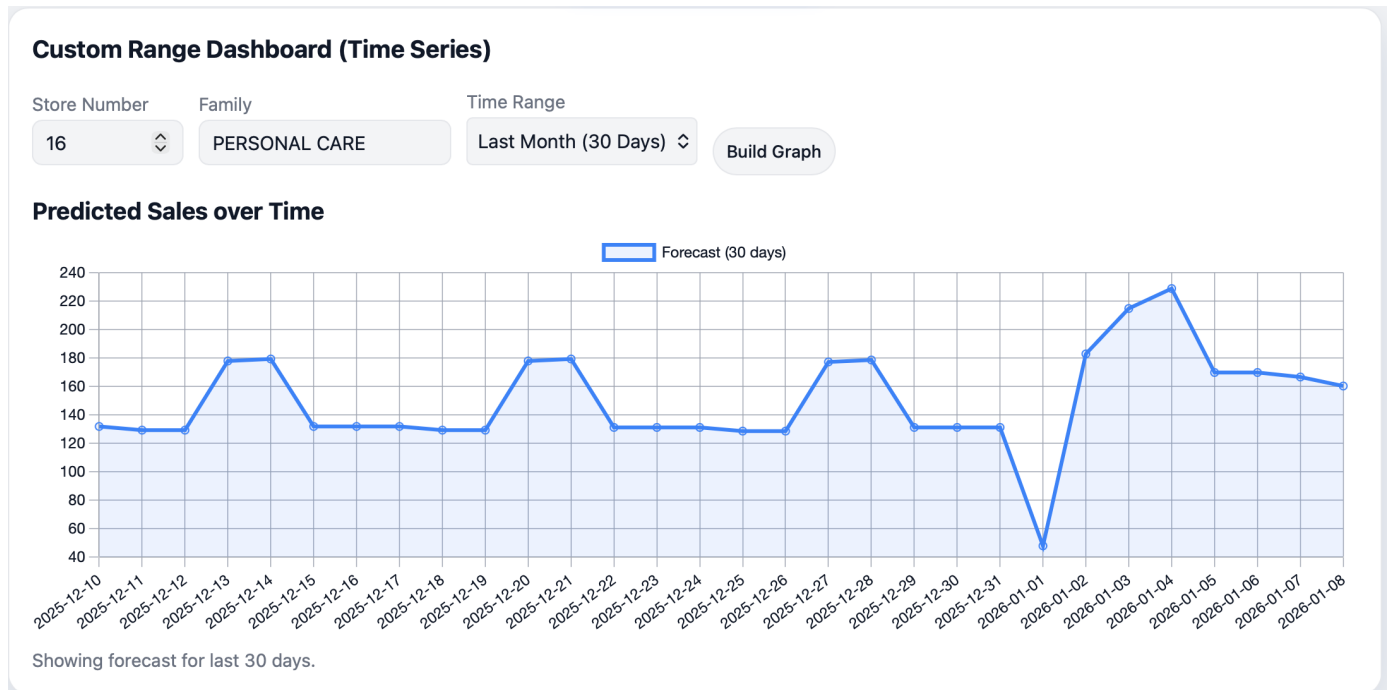
370

Stores Impacted

54**Filtered Critical Items**

Store	Family	Derived Status	Current Stock	Predicted Demand	Shortage / Excess
45	GROCERY I	OVERSTOCK	8,070.26	3,055.99	5,014.27
48	GROCERY I	OVERSTOCK	6,297.81	2,822.54	3,475.27
50	GROCERY I	OVERSTOCK	5,889.04	2,576.72	3,312.32
44	GROCERY I	OVERSTOCK	9,738.34	7,544.57	2,193.77
44	CLEANING	OVERSTOCK	3,142.2	979.93	2,162.27
21	GROCERY I	STOCKOUT	0	2,157.93	-2,157.93
6	GROCERY I	OVERSTOCK	4,289.23	2,496.23	1,793
47	CLEANING	OVERSTOCK	2,687.01	926.07	1,760.94

Step 6: Here is the Custom Range Dashboard. In this tab, users can select specific dates (e.g., Last 30 Days) to view a time-series graph of predicted sales versus actual stock for specific items.



Step 7: In the Manage Items tab, the user can add new inventory stock, update existing quantities, or remove obsolete items from the database.

Manage Inventory Items

Add new inventory items, update current stock or delete items. Click a row to edit it.

Search family (press Enter)

Search

Refresh

Item Form

Store Number: 1

Family: BOOKS

Current Stock: 9

Add Item Update Item Delete Item Clear

Items

Store	Family	Current Stock
1	AUTOMOTIVE	2.57
1	BABY CARE	0
1	BEAUTY	1.66
1	BOOKS	9
1	BREAD/BAKERY	329.98
1	CELEBRATION	0
1	CLEANING	776.11
1	DAIRY	452.61
1	DELI	132.46
1	EGGS	177.02

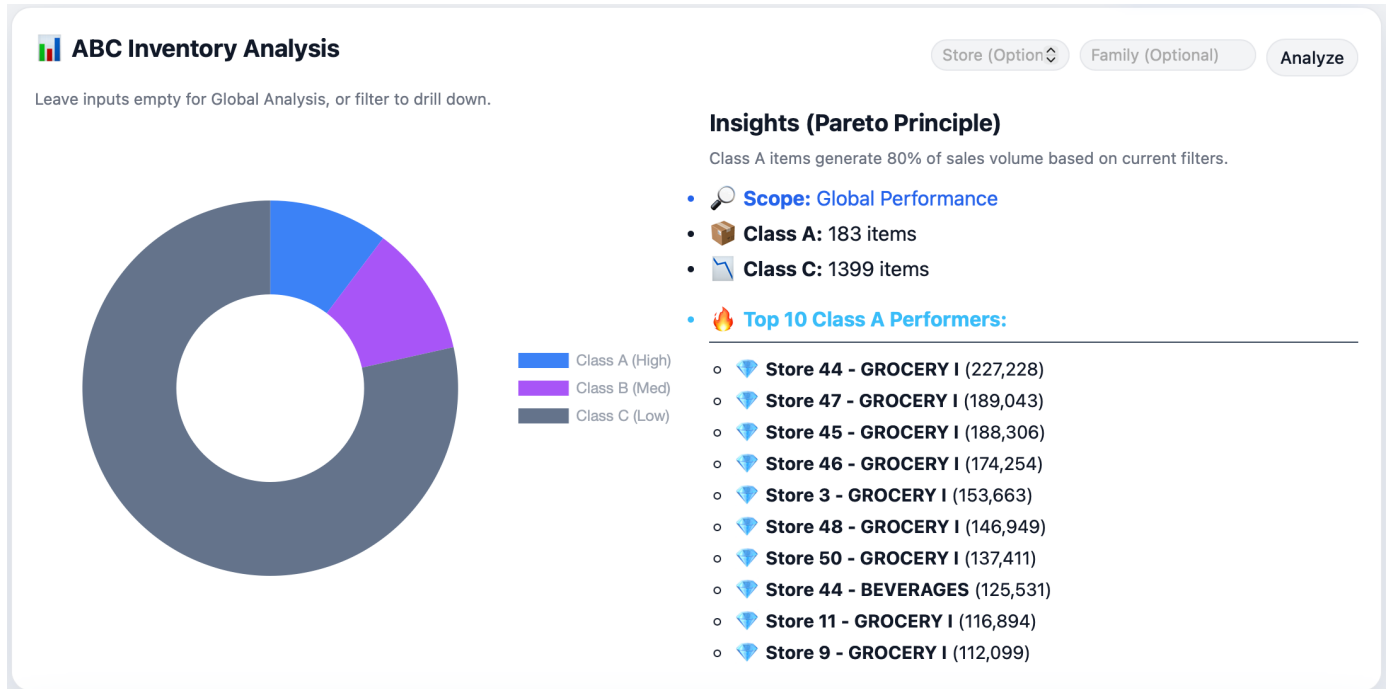
Recent Inserts

Recent inserts: 1

Store	Family	Current Stock
1	Mobile	100

Clear Recent

Step 8: The **Advanced Analytics** section starts with **ABC Analysis**, which categorizes inventory into Class A (High Value), B, and C to prioritize management efforts.



Step 9: Scrolling down in Analytics, the **Anomaly Detection** table highlights irregular sales spikes or potential theft by analyzing historical deviations (Z-Scores).



Anomaly Detection (Theft/Spike Alert)

Store (Option) Family (Optional) Scan All

Found 336 anomalies in the last 1 year. Showing all.

Store	Item	Date	Sales (Qty)	Avg Sales	Deviation
2	HARDWARE	2013-01-28	3	0.61	2.6x spike
31	LIQUOR,WINE,BEER	2013-01-28	109	21.71	3.0x spike
3	BEAUTY	2013-01-27	18	6.21	3.0x spike
4	BEAUTY	2013-01-27	10	3.46	2.6x spike
13	LINGERIE	2013-01-27	18	5.96	2.5x spike
26	LINGERIE	2013-01-27	24	8.82	2.5x spike
28	BEAUTY	2013-01-27	4	1.07	2.5x spike
38	LINGERIE	2013-01-27	80	33.93	3.0x spike
39	LAWN AND GARDEN	2013-01-27	7	1.75	2.8x spike
41	POULTRY	2013-01-27	133.29	68.49	2.6x spike
44	LINGERIE	2013-01-27	59	19.68	2.7x spike
48	BEAUTY	2013-01-27	22	7.71	2.6x spike
54	AUTOMOTIVE	2013-01-27	10	2.96	2.6x spike

Step 10: Here is the **Market Basket Patterns** section. This feature analyzes sales data to discover hidden correlations between items (e.g., items frequently bought together), aiding in layout optimization.

Market Basket Patterns
Store (Optional)
Min Conf %: 85
Find Patterns

Found 114 patterns (Min Confidence: 85%). Showing all.

Pattern DELI ↔ EGGS Strong correlation detected.	99.3%	Pattern CLEANING ↔ GROCERY I Strong correlation detected.	99%
Pattern DELI ↔ POULTRY Strong correlation detected.	98.8%	Pattern BREAD/BAKERY ↔ EGGS Strong correlation detected.	98.7%
Pattern DELI ↔ GROCERY I Strong correlation detected.	98.7%	Pattern BREAD/BAKERY ↔ DELI Strong correlation detected.	98.5%
Pattern EGGS ↔ POULTRY Strong correlation detected.	98.5%	Pattern EGGS ↔ GROCERY I Strong correlation detected.	98.4%
Pattern BEVERAGES ↔ POULTRY Strong correlation detected.	98.2%	Pattern BREAD/BAKERY ↔ GROCERY I Strong correlation detected.	98.1%

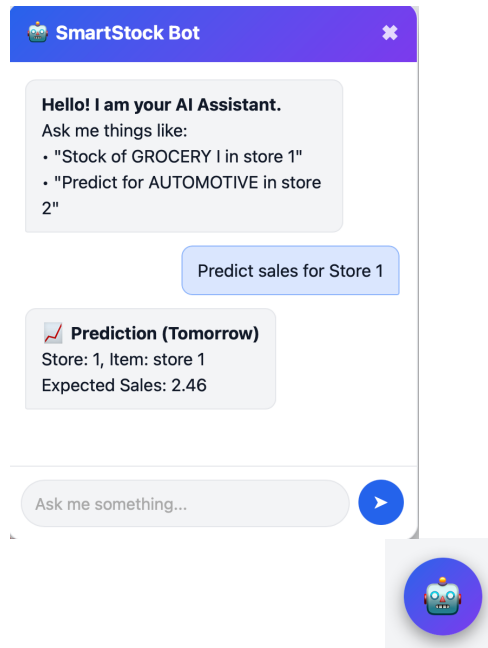
Step 11: The **Inter-Store Transfer Optimizer** identifies "Donor" and "Receiver" stores and recommends specific transfer quantities to balance stock levels efficiently.

Inter-Store Transfer Optimizer
Donor Stock > 40
Receiver Stock < 10
Find Matches

Found 4839 opportunities! Showing matches.

From (Donor)	To (Receiver)	Item Family	Transfer Qty	Action
Store 44	Store 21	BEVERAGES	50	<button>Approve</button>
Store 44	Store 22	BEVERAGES	50	<button>Approve</button>
Store 44	Store 29	BEVERAGES	50	<button>Approve</button>
Store 44	Store 36	BEVERAGES	50	<button>Approve</button>
Store 44	Store 42	BEVERAGES	50	<button>Approve</button>
Store 44	Store 52	BEVERAGES	50	<button>Approve</button>
Store 44	Store 53	BEVERAGES	50	<button>Approve</button>
Store 3	Store 21	BEVERAGES	50	<button>Approve</button>
Store 3	Store 22	BEVERAGES	50	<button>Approve</button>
Store 3	Store 29	BEVERAGES	50	<button>Approve</button>

Step 12: Using the **SmartStock Bot**, the user can communicate via natural language to ask questions like "Predict sales for Store 1" or "Check stock of Grocery I".



End : Finally, the application includes a **Footer** highlighting the technology stack (FastAPI, MySQL, ML) , Theme Change Button and a **Logout** button to securely end the session.



7. Challenges Faced

- **Data Simulation** – The available open-source sales dataset lacked real-time inventory levels, requiring the development of logic to derive "current stock" based on historical sales averages.
- **Data Preprocessing** – Handling missing dates and formatting inconsistencies in the sales records was necessary before feeding data into the Machine Learning model.
- **Model Tuning** – Achieving high accuracy with the `HistGradientBoostingRegressor` was challenging due to the impact of external factors like holidays and promotions on sales demand.
- **Algorithm Complexity** – Designing the **Inter-Store Transfer** logic was difficult, as it required efficiently matching "Donor" stores with "Receiver" stores without creating new shortages in the donor location.
- **System Integration** – Linking the scikit-learn ML pipeline with the FastAPI backend and ensuring the frontend dashboard updated dynamically created synchronization challenges.
- **Performance Optimization** – Processing **Market Basket Analysis** and **Anomaly Detection** on a full year of historical data initially caused API latency, requiring query optimization.
- **Alert Configuration** – Integrating external services like **Twilio (SMS)** and **SMTP (Email)** required careful error handling to ensure notifications were delivered reliably without crashing the server.
- **Chatbot Logic** – Implementing a robust Regex-based intent recognition system to accurately parse user queries for specific items and stores proved to be complex.
- **Time Management** – Balancing the development of the ML model, backend APIs, and the comprehensive frontend dashboard within the project timeline was demanding.

8. Learnings & Skills Acquired

- **AI & Machine Learning** – Gained hands-on experience in implementing the `HistGradientBoostingRegressor` algorithm and Scikit-Learn pipelines for accurate retail demand forecasting.
- **Backend Development** – Mastered **FastAPI** for building high-performance RESTful APIs, managing dependency injection, and handling asynchronous database operations.
- **Data Engineering** – Improved ability to manage **MySQL** databases using `SQLAlchemy`, including complex queries for sales aggregation and inventory tracking.
- **Advanced Analytics** – Learned to implement statistical methods like **Z-Score analysis** for anomaly detection and Pareto Principle logic for **ABC Analysis**.
- **System Integration** – Successfully integrated third-party services like **Twilio** (SMS) and **SMTP** (Email) to create a robust real-time notification system.
- **Frontend Visualization** – Enhanced skills in creating dynamic, theme-aware user interfaces and interactive charts using **Chart.js** and vanilla JavaScript.
- **Algorithm Design** – Developed logical thinking by designing the **Inter-Store Transfer Optimizer** to solve complex inventory distribution problems.
- **Retail Domain Knowledge** – Understood key supply chain concepts such as safety stock, stockout prevention, and inventory turnover ratios.
- **Chatbot Implementation** – Learned to build a natural language interface using Regex logic to query database states dynamically.

9. Testimonials from team

Team Member 1 [Aman Kumar](ML Developer): "Working on the machine learning component gave me deep insights into predictive modeling. I learned how to implement and tune the `HistGradientBoostingRegressor` to handle historical sales data and seasonality, ensuring our demand forecasts were accurate and reliable."

Team Member 2 [Smriti](Frontend Developer): "Building the main dashboard was a great learning experience. I focused on creating interactive visualizations using `Chart.js` to display stock alerts and forecasting trends, ensuring the system provided clear, actionable insights for store managers."

Team Member 3 [H.Vaishnavi](Frontend Developer - Auth): "I was responsible for the authentication interface, designing the secure Login and Registration pages. implementing JWT

token handling on the client side taught me the importance of secure session management and protecting user routes in a web application."

Team Member 4 [Girija Pavani](Backend Developer): "Developing the backend with FastAPI was a significant milestone for me. I learned to build high-performance APIs that seamlessly connect the ML model with the frontend, and I successfully implemented the logic for the Inter-Store Transfer Optimizer."

Team Member 5 [Durga Prasad](Database Engineer): "Handling the SQL portion strengthened my database management skills. I designed the schema for efficient sales tracking and wrote complex queries for Advanced Analytics, such as ABC Analysis and Anomaly Detection, to derive meaningful business insights."

10. Conclusion

The SmartStock project successfully developed an AI-powered inventory optimization system for accurate demand forecasting and automated stock management. It combines machine learning algorithms, real-time alerting mechanisms, and advanced analytics to help retail managers minimize stockouts, prevent overstocking, and optimize inter-store distribution. The project enhanced technical proficiency in full-stack development, AI integration, and supply chain logic, while delivering a robust, user-friendly tool that drives data-driven operational efficiency.

11. Acknowledgements

Thank the organization, mentor, and any team members who supported your internship journey.

I would like to express my sincere gratitude to **Infosys** for providing me the opportunity to work on the **SmartStock Inventory Optimization project** and gain hands-on experience in AI and ML applications.

I am especially thankful to my **mentor, [shakthi Gopalakrishnan]**, for their constant guidance, support, and valuable insights throughout the internship. Their mentorship helped me overcome challenges and grow both technically and professionally.

I would also like to thank my **team members** for their collaboration, encouragement, and efforts in successfully executing this project. Their teamwork made the internship journey smooth and enriching.

Finally, I appreciate everyone who contributed directly or indirectly to my learning and the completion of this project.