RSET
RAJAGIRI SCHOOL OF
ENGINEERING & TECHNOLOGY
(AUTONOMOUS)

Project Phase II Report On

Advanced Supply and Trade Resource Optimisation

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology

in

Computer Science and Business Systems

By

Amel Chandra (U2109009)

Bharath S. (U2109018)

Joepaul Vilsan (U2109033)

Shane George Salphie (U2109063)

Under the guidance of

Dr. Nikhila T. Bhuvan

Department of Computer Science and Business Systems
Rajagiri School of Engineering & Technology (Autonomous)
(Parent University: APJ Abdul Kalam Technological University)
Rajagiri Valley, Kakkanad, Kochi, 682039
April 2025

CERTIFICATE

This is to certify that the project report entitled ***”Advanced Supply and Trade Resource Optimisation”*** *is a bonafide record of the work done by* ***Amel Chandra (U2109009), Bharath S (U2109018), Joepaul Vilsan (U2109033), Shane George Salphie (U2109063)****, submitted to the Rajagiri School of Engineering & Technology (RSET) (Autonomous) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology (B. Tech.) in "Computer Science and Business Systems" during the academic year 2024-2025.*

Dr. Nikhila T. Bhuvan	Dr. Nikhila T. Bhuvan	Dr. Divya James
Project Guide	Project Co-ordinator	Head Of Department
Associate Professor	Associate Professor	Associate Professor
Dept. of CSBS	Dept. of CSBS	Dept. of CSBS
RSET	RSET	RSET

ACKNOWLEDGMENT

We wish to express our sincere gratitude towards **Rev. Dr. Jaison Paul Mulerikkal**, Principal of RSET, and **Dr. Divya James**, Head of the Department of Computer Science and Business Systems, for providing us with the opportunity to undertake our project, **"Advanced Supply and Trade Resource Optimisation"**.

We are highly indebted to our project coordinators, **Dr. Nikhila T. Bhuvan**, Associate Professor, Department of Computer Science and Business Systems, **Ms. Ancy C. A.**, Assistant Professor, Department of Computer Science and Business Systems, **Mr. Mahesh K. M.**, Assistant Professor, Department of Computer Science and Business Systems, for their valuable support.

It is indeed our pleasure and a moment of satisfaction for us to express our sincere gratitude to our project guide **Dr. Nikhila T. Bhuvan** for her patience and all the priceless advice and wisdom she has shared with us.

Last but not least, we would like to express our sincere gratitude towards all other teachers and friends for their continuous support and constructive ideas.

Amel Chandra

Bharath S.

Joepaul Vilsan

Shane George Salphie

Abstract

Small-scale vendors face significant challenges in competing with large retail chains due to limited purchasing power, inefficient logistics, and restricted access to financial services. These barriers often hinder their ability to manage demand fluctuations and make informed business decisions, threatening their long-term sustainability. This project aims to address these issues by developing a comprehensive platform that empowers small vendors through collaborative purchasing, optimized logistics, and data-driven decision-making. By enabling vendors to combine orders, the platform facilitates bulk purchasing, unlocking competitive discounts and cost savings. Advanced algorithms optimize logistics by streamlining delivery routes, thereby reducing transportation costs and enhancing operational efficiency. These features empower vendors to make strategic decisions that align with market demands and their business goals. The ultimate goal is to level the playing field for small vendors, enabling them to compete effectively with larger retail chains while fostering sustainable growth in local economies. The deliverables of this project include a fully functional platform supporting vendor collaboration and an optimized logistics module to reduce operational costs. By promoting economic resilience, enhancing competitiveness, and driving sustainable development, this project seeks to strengthen the local vendor ecosystem and contribute to the overall development of communities.

Contents

Acknowledgment **i**

Abstract **ii**

List of Abbreviations **vi**

List of Figures **vii**

List of Tables **viii**

1 Introduction **1**

1.1 Background 2

1.2 Problem Definition 3

1.3 Scope and Motivation 4

1.4 Objectives 5

1.5 Relevance 6

2 System Architecture **10**

2.1 Dataset 12

2.2 Preprocessing Steps 13

2.3 Feature Extraction 15

2.3.1 Time-Based Features 15

2.3.2 Lagged Variables 16

2.4 Project Modules and Their Integration 17

2.4.1 Order Aggregation Module 17

2.4.2 Route Optimisation Module 17

2.4.3 Demand Forecasting Module 18

2.5 Order Aggregation Using Hybrid DBSCAN with Genetic Algorithm 19

2.5.1 Vendor Clustering Module and Order Aggregation Module 19

2.5.2 Parameter Selection with Genetic Algorithm 20
2.5.3 DBSCAN Algorithm for Clustering Orders 21
2.5.4 Output and Implementation 21
2.5.5 Integration with the Platform 22
2.6 Logistics Using Environment Adaptive Genetic Algorithm (EAGA) 22
2.6.1 Mechanics of Logistics Optimisation 22
2.6.2 Advantages of Using EAGA for Route Optimization 25
2.6.3 Integration with the Platform 26
2.7 Demand Forecasting with Neural Prophet 26
2.7.1 Inputs for Neural Prophet 26
2.7.2 Neural Prophet Structure 27
2.7.3 Outputs 28
2.7.4 Integration with Platform Workflow 28

3 Implementation **30**
3.1 System Overview 30
3.2 Backend Implementation 30
3.2.1 FastAPI Framework 30
3.2.2 Database Schema 31
3.2.3 API Endpoints 31
3.3 Frontend Implementation 32
3.3.1 React and TypeScript Application 32
3.3.2 Application Architecture 33
3.3.3 Key UI Components 33
3.4 Integration and Communication 34
3.4.1 API Integration 34
3.4.2 Real-time Notifications 34
3.5 Security Implementation 35
3.5.1 Authentication and Authorization 35
3.5.2 Data Encryption 35
3.6 Summary 36

4 Conclusion **37**

Appendix B: Vision, Mission, PO, PSO, and CO **42**

Appendix C: CO-PO-PSO Mapping **46**

List of Abbreviations

ASTRO - Advanced Supply and Trade Resource Optimisation

SDG - Sustainable Development Goals

ARIMA - Auto Regressive Integrated Moving Average

ACO - Ant Colony Optimisation

VAM - Vogel's Approximation Method

MACOA - Modified Ant Colony Optimisation Algorithm

MINLP - Modified Integer Non-Linear Programming

IMFO - Improved Moth-Flame Optimization

GA - Genetic Algorithm

PSO - Particle Swarm Optimization

GWO - Grey Wolf Optimizer

AHP - Analytic Hierarchy Process

MILP - Modified Integer Linear Programming

SME - Small to Medium-sized Enterprises

WOA - Whale Optimisation Algorithm

VRP - Vehicle Routing Problem

VRPTW - Vehicle Routing Problem with Time Windows

SIH - Solomn Insertion Heuristic

HGA - Hybrid Genetic Algorithm

CPLA - Co-operative Population Learning Algorithm

PITSH - Population-based Iterated Tabu Search Heuristic

API - Application Programming Interfaces

MOQ - Minimum Order Quantity

UML - Unified Modelling Language

OAS - Order Aggregation System

List of Figures

2.1 System Architecture 10

4.1 NeuralProphet Demand Forecasting 40

4.2 NeuralProphet Long-term Trend Analysis 40

4.3 NeuralProphet Weekly Seasonality Analysis 41

4.4 NeuralProphet Yearly Seasonality Analysis 41

List of Tables

2.1 Example of Vendor Clustering in Order Aggregation 21
2.2 Key Inputs for Neural Prophet 27
2.3 Example of Forecasted Sales and Smoothed Trend 28

4.1 Cost Comparison: Traditional vs. Collaborative Route Optimization 38

Chapter 1

Introduction

Small-scale vendors are essential contributors to local economies, offering unique products and personalized services that enrich communities. However, they face significant challenges competing against large retail chains, which benefit from greater purchasing power, efficient logistics, and easier access to financial resources. Due to limited resources, small vendors often struggle with high supply costs, fragmented logistics, and restricted access to credit, hindering their ability to grow and compete effectively. The ASTRO (Advanced Supply and Trade Resource Optimisation) platform addresses these challenges by providing a comprehensive solution that empowers small vendors through collaborative purchasing, logistics optimisation, and data-driven financial services. By allowing vendors to pool orders, ASTRO enables them to achieve bulk pricing similar to large retailers, reducing per-unit costs and enhancing profitability. The platform also incorporates advanced logistics optimisation algorithms, improving delivery efficiency and reducing operational expenses. Additionally, ASTRO offers financial services based on transaction data, enabling vendors to access credit and make informed decisions for business expansion.

ASTRO's functionality extends further with real-time demand forecasting tools, equipping vendors with predictive insights that help manage inventory and adapt to changing market demands. Through these features, ASTRO not only improves individual vendor competitiveness but also fosters economic resilience in local communities by supporting sustainable growth and operational efficiency. As a result, ASTRO is positioned as a vital resource for small vendors, helping them navigate a competitive retail landscape while contributing to the stability and diversity of local economies.

1.1 Background

Small-scale vendors are integral to the fabric of local economies, providing unique products, personalized services, and fostering vibrant community interactions that larger retail chains often cannot replicate. These vendors contribute significantly to economic diversity, cultural richness, and employment within their communities. However, despite their crucial role, small-scale vendors face persistent challenges that impede their ability to compete effectively with larger retail entities. These challenges include limited purchasing power, inefficient logistics, restricted access to financial services, and a lack of data-driven decision-making tools. In today's highly competitive retail landscape, large retail chains leverage economies of scale to negotiate bulk purchasing discounts, implement sophisticated logistics networks, and utilize advanced data analytics to optimize their operations and marketing strategies. In contrast, small-scale vendors typically operate with constrained resources, making it difficult for them to achieve similar efficiencies and cost-effectiveness. This disparity not only affects their profitability but also limits their capacity to respond to market fluctuations and evolving consumer demands. The ASTRO platform (Advanced Supply and Trade Resource Optimisation) is designed to address these disparities by providing small-scale vendors with tools and services that enhance their competitive edge. By enabling collaborative purchasing, optimizing logistics, and offering data-driven financial services, ASTRO empowers local vendors to achieve bulk pricing, streamline their operations, and make informed business decisions. This platform aims to bridge the gap between small vendors and large retail chains, fostering economic resilience and sustainable growth within local communities. Moreover, the advent of digital transformation has revolutionized various industries, including retail. The integration of technology into traditional business models is essential for small-scale vendors to remain relevant and competitive. ASTRO leverages modern technological advancements, such as machine learning algorithms for logistics optimisation and predictive analytics for demand forecasting, to deliver practical solutions tailored to the specific needs of small vendors. The significance of ASTRO extends beyond individual business success; it contributes to the overall economic health of communities by supporting the sustainability and growth of small-scale enterprises.

1.2 Problem Definition

Small-scale vendors face a significant challenge in today's competitive retail landscape, where large retail chains dominate with bulk pricing, advanced logistics, and extensive resources. This disparity puts small vendors at a disadvantage, as they often lack the economies of scale and infrastructure necessary to compete effectively. The problem is that, despite being crucial to local economies, small vendors struggle with high operational costs, inefficient logistics, and limited access to affordable credit, all of which hinder their ability to grow sustainably. This issue is exacerbated by the lack of platforms that cater to the specific needs of small-scale businesses, especially when it comes to aggregating demand, optimizing supply chains, and improving cash flow.

Without access to bulk purchasing and efficient logistics, small vendors pay higher prices for goods, limiting their profit margins and making it difficult to reinvest in their businesses. Furthermore, the absence of accurate demand forecasting and inventory management tools leads to overstocking or stockouts, reducing customer satisfaction and revenue. These issues underscore the need for a system that enables small vendors to pool their purchasing power, streamline their supply chains, and make data-driven decisions to improve their competitiveness.

The Advanced Supply and Trade Resource Optimisation (ASTRO) platform seeks to address this gap by providing small-scale vendors with the tools to compete with larger retail chains. ASTRO enables collaborative purchasing, allowing vendors to aggregate their orders and access bulk discounts, thus reducing product costs. Additionally, the platform offers real-time demand forecasting, helping vendors optimize inventory levels to avoid both overstock and stockouts.

By facilitating these services, ASTRO aims to foster economic resilience and sustainable growth in local communities. Its focus on empowering small vendors with advanced tools levels the playing field, promoting a more equitable and competitive market environment. This approach not only ensures that small vendors can thrive but also contributes to the overall health and diversity of the retail ecosystem.

1.3 Scope and Motivation

The ASTRO platform is a multifaceted solution designed to support small-scale vendors by addressing key operational challenges through technological integration and collaborative strategies. The scope of ASTRO encompasses the development and implementation of several core functionalities:

1. **Collaborative Purchasing:** ASTRO facilitates group purchasing among small vendors, allowing them to pool their orders to achieve bulk pricing discounts. This feature enhances purchasing power, enabling vendors to reduce their cost of goods sold and improve profit margins.

2. **Logistics Optimisation:** Utilizing advanced algorithms, ASTRO optimizes delivery routes to reduce transportation costs and improve delivery efficiency. By streamlining logistics, the platform helps vendors achieve timely deliveries and minimize operational expenses.

3. **Real-Time Demand Forecasting:** Leveraging predictive analytics, ASTRO provides real-time demand forecasting tools that help vendors anticipate market demand and manage inventory levels effectively. Accurate demand forecasting reduces the risk of overstocking or stockouts, ensuring that vendors can meet customer needs efficiently.

4. **User-Friendly Interface:** ASTRO is designed with an intuitive user interface that ensures ease of use for vendors with varying levels of technical expertise. This feature maximizes user engagement and ensures that vendors can effectively utilize the platform's functionalities to enhance their business operations.

The motivation behind ASTRO stems from the urgent need to empower small-scale vendors in an increasingly competitive retail environment. As large retail chains continue to dominate the market through superior purchasing power and advanced logistics, small vendors risk marginalization and potential business failure. ASTRO addresses these challenges by providing a robust platform that enhances operational efficiency, reduces costs, and increases access to financial resources, thereby enabling small vendors to compete on a more equal footing.

Additionally, ASTRO is motivated by the broader goal of fostering economic resilience and sustainability within local communities. Small-scale vendors are pivotal to the economic diversity and vitality of local markets, contributing to job creation and community development. By supporting these vendors, ASTRO not only enhances individual business success but also strengthens the overall economic fabric of communities, promoting long-term sustainable growth and economic stability.

1.4 Objectives

The ASTRO project is guided by a set of clear and actionable objectives aimed at addressing the challenges faced by small-scale vendors. These objectives are designed to enhance operational efficiency, reduce costs, and provide financial support, thereby empowering vendors to compete effectively in the marketplace. The primary objectives of ASTRO include:

1. **Enable Collaborative Purchasing:**
    - **Objective:** Develop a robust system that allows small vendors to combine their orders, thereby achieving bulk purchasing discounts.
    - **Outcome:** Increased purchasing power, reduced cost of goods sold, and improved profit margins for vendors.
2. **Optimize Logistics:**
    - **Objective:** Implement advanced route optimisation algorithms to streamline delivery processes, reduce transportation costs, and enhance operational efficiency.
    - **Outcome:** Cost-effective logistics operations, timely deliveries, and reduced operational expenses.
3. **Provide Real-Time Demand Forecasting:**
    - **Objective:** Integrate predictive analytics tools to offer real-time demand forecasting, enabling vendors to anticipate market demand and manage inventory levels effectively.

- **Outcome:** Minimization of overstocking and stockouts, improved inventory management, and enhanced ability to meet customer demands.

4. **Promote Economic Resilience and Sustainability:**

   - **Objective:** Foster economic resilience by providing small vendors with the tools and resources needed to sustain and grow their businesses.
   - **Outcome:** Long-term sustainability, increased competitiveness, and strengthened economic stability within local communities.

5. **Enhance User Experience and Accessibility:**

   - **Objective:** Design a user-friendly interface that ensures easy access to the platform's features, regardless of vendors' technical expertise.
   - **Outcome:** High user engagement, effective utilization of platform functionalities, and improved overall user satisfaction.

6. **Support Community Development:**

   - **Objective:** Encourage collaboration and mutual support among vendors, fostering a sense of community and shared growth.
   - **Outcome:** Collective bargaining power, knowledge sharing, and resource optimisation, leading to a stronger and more cohesive vendor network.

By achieving these objectives, ASTRO aims to create a comprehensive solution that empowers small-scale vendors to overcome their operational challenges, enhance their competitiveness, and contribute to the sustainable growth of local economies. Each objective is strategically aligned to address specific pain points, ensuring that the platform delivers tangible benefits and drives meaningful improvements in vendors' business operations.

1.5 Relevance

The relevance of the ASTRO project is multifaceted, addressing critical needs within the retail ecosystem and contributing to broader economic and social goals. Key aspects of ASTRO's relevance include:

1. **Economic Empowerment of Small Vendors:**

    - **Impact:** ASTRO directly addresses the economic challenges faced by small-scale vendors, enhancing their purchasing power and optimizing logistics cost. This empowerment enables vendors to operate more efficiently, reduce costs, and increase profitability, thereby improving their economic standing and sustainability.

2. **Enhancing Local Economies:**

    - **Impact:** By supporting small vendors, ASTRO contributes to the vitality and resilience of local economies. Small businesses are integral to economic diversification and job creation, promoting the long-term sustainability of local markets.

3. **Technological Advancement and Digital Transformation:**

    - **Impact:** ASTRO exemplifies the role of technology in transforming traditional business models. By leveraging advanced algorithms for logistics optimisation, predictive analytics for demand forecasting, ASTRO introduces modern technological solutions to enhance the operational capabilities of small vendors. This digital transformation is crucial for small businesses to remain competitive in an increasingly digital and data-driven marketplace.

4. **Sustainable Growth and Environmental Impact:**

    - **Impact:** ASTRO's logistics optimisation not only reduces operational costs but also minimizes the environmental footprint of delivery processes. By streamlining routes and reducing transportation distances, ASTRO contributes to more sustainable business practices, aligning with global efforts to reduce carbon emissions and promote environmental sustainability.

5. **Competitive Parity:**

    - **Impact:** ASTRO helps small vendors achieve competitive parity with larger retail chains by providing tools that enhance their operational efficiency and cost-effectiveness. This parity is crucial for maintaining market diversity and preventing the monopolistic dominance of large retailers, ensuring a healthy and competitive marketplace that benefits both vendors and consumers.

6. **Adaptation to Market Trends:**

   - **Impact:** In a rapidly evolving retail environment, the ability to adapt to changing market trends and consumer behaviors is essential. ASTRO's real-time demand forecasting and data-driven insights enable small vendors to stay ahead of market trends, respond promptly to consumer needs, and adapt their inventory and marketing strategies accordingly.

7. **Community Building and Collaboration:**

   - **Impact:** ASTRO fosters a collaborative ecosystem where small vendors can work together to achieve common goals. This sense of community and mutual support enhances collective bargaining power, knowledge sharing, and resource optimisation, contributing to the overall strength and cohesion of the local vendor network.

8. **Scalability and Replicability:**

   - **Impact:** ASTRO's platform is designed to be scalable, allowing it to be replicated in various regions and adapted to different market conditions. This scalability ensures that the platform can benefit a wide range of small-scale vendors across different industries, promoting widespread economic empowerment and community resilience.

9. **Support for Economic Diversity and Stability:**

   - **Impact:** By enabling small vendors to thrive, ASTRO supports economic diversity, which is essential for the stability and resilience of local economies. Diverse economic activities reduce dependence on a single sector or large corporations, making communities more adaptable to economic shocks and changes.

10. **Promoting Entrepreneurship and Innovation:**

    - **Impact:** ASTRO encourages entrepreneurship by lowering the barriers to entry for small vendors. By providing the necessary tools and resources, ASTRO enables aspiring entrepreneurs to start and grow their businesses, fostering a culture of innovation and economic dynamism within local communities.

11. **Enhanced Customer Experience:**

    - **Impact:** By improving operational efficiency and reducing costs, ASTRO enables small vendors to offer better pricing and more reliable services to their customers. Enhanced customer experiences lead to increased customer loyalty and satisfaction, driving repeat business and positive word-of-mouth referrals.

12. **Economic Resilience and Crisis Management:**

    - **Impact:** ASTRO equips small vendors with the tools to better manage economic fluctuations and crises. By optimizing inventory levels, forecasting demand accurately, and accessing financial support, vendors are better prepared to navigate economic downturns, supply chain disruptions, and other unforeseen challenges.

13. **Alignment with Sustainable Development Goals (SDGs):**

    - **Impact:** ASTRO aligns with key United Nations Sustainable Development Goals, particularly Decent Work and Economic Growth (SDG 8) and Responsible Consumption and Production (SDG 12). By empowering small vendors, ASTRO promotes inclusive economic growth and decent work opportunities in local communities. Through optimized logistics and collaborative purchasing, the platform reduces resource waste and transportation emissions, supporting more sustainable consumption and production patterns in retail supply chains.

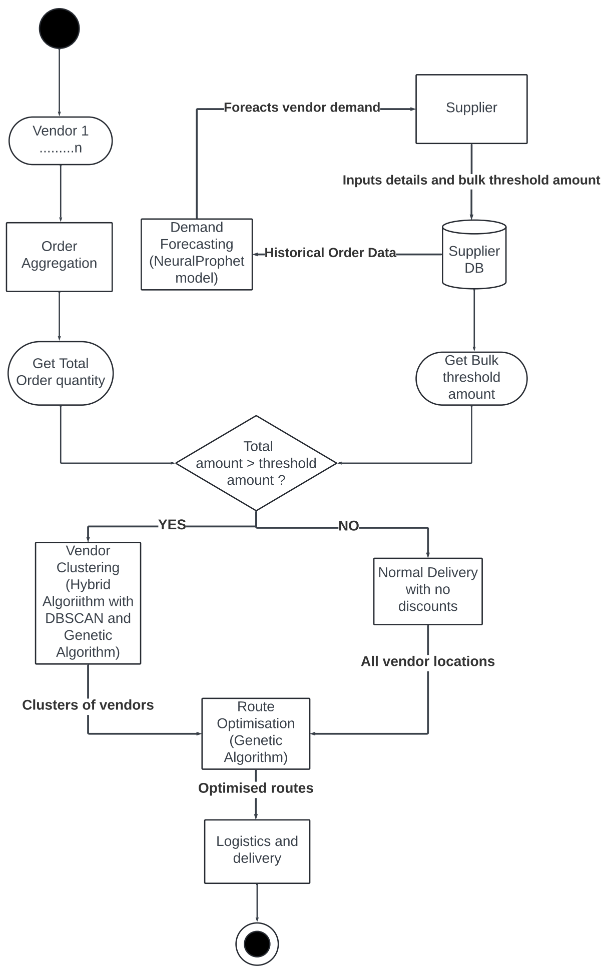
14. **Enhancing Vendor Autonomy and Empowerment:**

    - **Impact:** ASTRO empowers small vendors by giving them greater control over their purchasing, logistics, and financial decisions. This autonomy fosters a sense of ownership and confidence among vendors, encouraging them to take proactive steps towards business improvement and growth.

Chapter 2

System Architecture

Figure 2.1 illustrates the system architecture for the ASTRO Platform, designed to support small-scale vendors by facilitating bulk order discounts, demand forecasting, and logistics optimisation. This architecture provides a clear overview of the platform's components and how they work together to streamline the entire process from order initiation to optimized delivery. Here's a breakdown of the key aspects and importance of this architecture:



Vendor 1 ....n
Order Aggregation
Get Total Order quantity
Foreacts vendor demand
Supplier
Inputs details and bulk threshold amount
Demand Forecasting (NeuralProphet model)
Historical Order Data
Supplier DB
Get Bulk threshold amount
Total amount > threshold amount ?
YES
NO
Vendor Clustering (Hybrid Algoriithm with DBSCAN and Genetic Algorithm)
Normal Delivery with no discounts
All vendor locations
Clusters of vendors
Route Optimisation (Genetic Algorithm)
Optimised routes
Logistics and delivery

Figure 2.1: System Architecture

1. **High-Level Overview:** The architecture serves as a roadmap, showing the main stages of the platform's workflow. From initial product viewing and order placement by vendors, through demand forecasting, order aggregation, and logistics optimisation, this high-level overview provides all stakeholders—including vendors, suppliers, and platform administrators—with a clear understanding of the system's core functionality.

2. **Clarifies Data Flow:** The architecture clarifies the flow of data within the system, beginning with the acquisition of order and sales data. This data undergoes preprocessing and feature extraction, followed by demand forecasting with the Neural Prophet model. The forecast data then flows into the Order Aggregation module, which combines vendor orders to meet bulk purchase thresholds. Finally, optimized delivery routes are created through the Logistics Optimisation module, using Green Routing techniques. This sequential data flow enhances system transparency and makes it easy to identify potential bottlenecks.

3. **Enables Collaborative Order Aggregation:** The architecture highlights how order aggregation enables small-scale vendors to achieve bulk discounts. By aggregating orders from multiple vendors, the platform ensures Minimum Order Quantities (MOQs) are met, enabling vendors to access bulk discounts typically reserved for larger chains. This aspect of the system is crucial for enhancing the competitiveness of small vendors.

4. **Supports Logistics Optimisation through Green Routing:** The architecture emphasizes the logistics optimisation approach used in the platform. The Green Routing component incorporates route flexibility and service time windows to create efficient and environmentally friendly delivery routes. This optimisation minimizes transportation costs and enhances delivery reliability for vendors, contributing to a streamlined and cost-effective supply chain.

5. **Enables Real-Time Notifications and Updates:** The system architecture includes a notification mechanism that provides real-time updates to vendors on order status, inventory levels, and logistics. These notifications foster transparency, enabling vendors to make informed decisions about inventory and delivery schedules.

6. **Facilitates Collaboration and Communication Among Stakeholders:** The system architecture acts as a shared framework, providing a common language for platform users, developers, and suppliers. It allows stakeholders to better understand the interactions between components and collaborate effectively toward improving platform functionality. This collaborative framework is essential for ensuring smooth operations and adapting to vendor needs.

Overall, the system architecture is crucial for understanding the ASTRO Platform. It provides a clear visual representation of the platform's workflow, highlighting key features such as order aggregation, demand forecasting, logistics optimisation, and real-time notifications. These components work together to enhance vendor competitiveness by offering cost-saving opportunities, efficient delivery solutions, and improved collaboration with suppliers.

2.1 Dataset

Purpose: This dataset supports research in sales forecasting, specifically at the item and store levels. The objective is to predict the next three months of sales for individual items at various store locations, applicable for demand forecasting, inventory management, and sales optimisation.

Data Collection: This dataset was collected from multiple store locations, representing real-world sales data with variations in store performance and item demand. It provides diverse daily sales records for individual items across different stores, enabling demand pattern analysis.

Data Fields:

date: Date of each sale record. There are no adjustments for holidays or store closures, simplifying the analysis but possibly requiring external data for event-based effects.

store: A unique identifier for each store, allowing for store-level analysis and capturing unique characteristics and item demand differences.

item: A unique identifier for each item sold, enabling item-level demand forecasting and cross-store analysis.

sales: The number of items sold at a particular store on a given date, which is the target variable for forecasting models.

File Descriptions:

train.csv: Contains historical sales data used for training forecasting models.

test.csv: Provides the test data where future sales predictions are needed, with a time-based split for realistic scenario testing.

sample_submission.csv: A sample submission file showing the required format for submitting sales forecasts.

Usage in Research: This dataset supports the development and validation of time series forecasting models. It allows researchers to explore seasonality, trends, and external influences on sales. Models like Neural Prophet or ARIMA can be applied, along with machine learning techniques for enhanced forecasting accuracy.

Challenges and Opportunities: Although the dataset lacks explicit holiday or store event information, researchers may incorporate external data to improve forecasts. This provides an opportunity to explore techniques that account for seasonality and trend modelling, such as Fourier series for seasonality or external regressors.

Access: This dataset is easily accessible, enabling collaborative studies on sales forecasting and providing a resource for testing and improving demand forecasting algorithms.

2.2 Preprocessing Steps

To ensure the data is prepared for accurate demand forecasting and that the model interprets the features effectively, the following preprocessing steps are applied:

Tasks

Handling Missing Values: Missing data may exist in historical datasets due to reporting errors or system downtime. The Neural Prophet model employs a data imputation mechanism to avoid excessive data loss when working with incomplete data. Missing events are filled in with zeros, indicating their absence. For other real-valued variables, including the time series itself (when autoregression is enabled), the following imputation procedure is applied:

1. **First Step:** Missing values are approximated using bi-directional linear interpolation. The missing values are filled by interpolating between the last known value before and after the missing data. This process is applied to a maximum of 5 missing values in each direction.

2. **Second Step:** Remaining missing values are imputed using a centered rolling average. A window of 30 is used, and up to 20 consecutive missing values can be filled.

3. **Third Step:** If more than 30 consecutive missing values remain, these data points are dropped from the dataset.

This imputation process ensures minimal loss of data while maintaining the integrity of the time series.

Scaling and Normalization: To improve model efficiency, numerical features, particularly sales data, are normalized. The Neural Prophet model offers various normalization options, with the default being the soft method. This approach scales the minimum value to 0.0 and the 95th quantile to 1.0, ensuring that the time series values are appropriately transformed for the model.

The formula for soft normalization is as follows:

$$x' = \frac{x - \min(x)}{Q_{95} - \min(x)}$$

where Q_{95} represents the 95th quantile of the time series values.

Time-Based Feature Engineering: To capture temporal and seasonal patterns in the data, key features like Day of the Week, Month, and a binary IsWeekend indicator are created. These features help the model understand weekly sales patterns, broader seasonal trends (e.g., holiday spikes), and the impact of weekends on sales. The IsWeekend feature is binary, set to 1 if the date is a Saturday or Sunday and 0 otherwise.

Lagged Sales Features: Since past sales are directly predictive of future demand, lagged sales features are created for the last 7 and 30 days to capture both short-term and longer-term patterns. Additionally, rolling statistics such as the rolling mean and rolling standard deviation over the past 30 days are computed. These features smooth out fluctuations and help the model capture trends more accurately. The rolling mean is calculated as:

$$\text{Rolling Mean}_{30} = \frac{1}{30} \sum_{i=t-30}^{t-1} \text{Sales}(i)$$

and the rolling standard deviation as:

$$\text{Rolling Std}_{30} = \sqrt{\frac{1}{30} \sum_{i=t-30}^{t-1} (\text{Sales}(i) - \text{Rolling Mean}_{30})^2}$$

Output

The result of these preprocessing steps is a clean, normalized dataset with imputed missing values and encoded categorical features. The dataset is now ready for model training, ensuring that the features are aligned and consistent, helping the model make accurate demand forecasts.

2.3 Feature Extraction

Feature extraction is a critical process in the ASTRO platform as it transforms raw sales data into meaningful representations, enabling accurate demand forecasting and efficient operational management. By identifying and engineering relevant features from the dataset, we capture essential patterns, trends, and relationships that enhance the performance of predictive models.

In the context of the dataset, which contains store-level and item-level sales information over time, feature extraction is designed to uncover temporal patterns, historical dependencies, and interactions between stores and items. The extracted features are categorized into two major types: time-based features and lagged variables, both of which are integral to understanding and predicting demand fluctuations.

2.3.1 Time-Based Features

Time-based features play a crucial role in capturing cyclical and seasonal patterns inherent in sales data. By decomposing the date field into granular components, these features allow the model to understand how sales vary across different times of the week, month, and year. For instance, the day of the week feature represents weekdays as integers, enabling the model to differentiate between regular weekday and weekend shopping behaviours. Similarly, the month feature identifies seasonal effects that often influence demand, such as increased sales during holiday months or specific seasonal trends for particular items.

Another important time-based feature is the week of the year, which highlights demand patterns recurring annually, such as back-to-school shopping or end-of-year festive sales. Additionally, a binary is weekend feature isolates the typically higher weekend sales from weekday trends, ensuring that the model appropriately weighs these differences when forecasting.

These features collectively form the temporal backbone of the forecasting model, allowing it to detect repeating cycles and shifts in demand driven by the calendar.

2.3.2 Lagged Variables

Lagged variables focus on incorporating historical sales data to enhance the model's ability to forecast future demand. These features represent the past performance of sales, which is often predictive of future trends. For example, a 1-day lag feature captures the sales volume from the previous day, providing immediate short-term context for the current demand. Similarly, a 7-day lag feature reflects weekly demand patterns by showing the sales from the same day of the prior week, which is especially useful for capturing weekly seasonality.

To understand longer-term patterns, a 30-day lag feature is introduced, representing sales from one month earlier. This helps in identifying recurring monthly trends or evaluating the effectiveness of past promotions. In addition to specific lags, rolling statistical features such as the rolling average (7-day) and rolling standard deviation (7-day) are calculated. These features smooth short-term fluctuations and provide insights into the stability or variability of demand over a week.

A cumulative sales feature further extends this approach by summing all sales up to the current date for a given store-item combination. This feature captures overall sales momentum and growth trends, offering a long-term perspective on performance.

Lagged variables are particularly effective in making the model aware of historical dependencies, enabling it to account for demand inertia, recent trends, and periodic changes.

The combination of time-based features and lagged variables provides a comprehensive representation of the dataset, balancing temporal patterns with historical sales behaviour. These features enable the demand forecasting models to detect and learn from intricate relationships within the data, ensuring robust predictions and informed decision-making for vendors. Through this customized feature extraction approach, the platform empowers small-scale vendors with insights that drive collaborative efficiency and competitiveness.

2.4 Project Modules and Their Integration

ASTRO platform comprises three core modules: Demand Forecasting, Order Aggregation, and Route Optimisation. These modules work in harmony to streamline vendor operations, optimize costs, and enhance efficiency in the supply chain. By integrating these modules, the platform empowers vendors to achieve bulk order discounts, coordinate deliveries, and reduce logistics expenses, creating a collaborative ecosystem that benefits all participants.

2.4.1 Order Aggregation Module

The Order Aggregation Module builds on the insights provided by the demand forecasting module. Once multiple vendors agree to participate in a joint order, this module combines their requirements into a single bulk order. The process ensures that the MOQ thresholds are met, unlocking discounts from suppliers that would be unattainable for individual vendors.

The aggregation process also categorizes and organizes the joint order to ensure that products are distributed efficiently. Each vendor's share of the bulk order is clearly delineated, and the aggregated data serves as input for the subsequent logistics operations.

Notifications play a key role in this module: vendors are informed about the opportunity to join an order and, once confirmed, receive updates about the order status and expected delivery timelines. The module ensures transparency and coordination among vendors, simplifying the process of collective purchasing.

2.4.2 Route Optimisation Module

The Route Optimisation Module is responsible for ensuring efficient delivery of the aggregated orders. After the order is finalized and fulfilled by the supplier, this module uses

Green Route Optimisation algorithms to design an optimal delivery route.

Since the joint order often involves multiple vendors located in different areas, the Route optimisation module focuses on clustering delivery locations and minimizing logistics costs. The Route Flexibility and Service Time Window feature ensures that deliveries are scheduled efficiently while adhering to time constraints.

The module optimizes the route for a single vehicle tasked with delivering products to multiple vendors. Factors such as load capacity, distance between vendors, and delivery time windows are considered to achieve the most cost-effective and eco-friendly route. By reducing unnecessary travel and fuel consumption, this module contributes to both financial savings and environmental sustainability.

2.4.3 Demand Forecasting Module

The Demand Forecasting Module lies at the heart of the platform, enabling vendors to predict future sales and order cycles. By analysing historical sales data, the module leverages the Neural Prophet model to identify trends, seasonal patterns, and demand fluctuations. The predictions provide vendors with actionable insights into when and how much to order, ensuring they maintain optimal inventory levels while avoiding overstocking or stockouts.

Additionally, this module facilitates order cycle prediction, which is critical for synchronizing vendor orders. When a vendor places an order with a supplier, the platform uses the demand forecasting output to identify similar demand cycles for other vendors. This allows the platform to notify these vendors about the opportunity to join the order, achieving the Minimum Order Quantity (MOQ) threshold required for bulk discounts. By doing so, the demand forecasting module not only supports individual vendors but also sets the foundation for collective collaboration.

The seamless integration of demand forecasting, order aggregation, and route optimisation modules creates a powerful, collaborative platform that addresses the challenges faced by small-scale vendors. The demand forecasting module drives proactive order management, the order aggregation module facilitates cost-effective purchasing, and the route optimisation module ensures efficient delivery. Together, these modules enable vendors to compete with large retail chains by reducing costs, improving operations, and fostering a cooperative ecosystem.

2.5 Order Aggregation Using Hybrid DBSCAN with Genetic Algorithm

Order aggregation is a crucial module in the platform, enabling small-scale vendors to achieve bulk discounts by combining their orders with others. This process ensures that the Minimum Order Quantities (MOQ) required by suppliers are met, leading to cost savings and operational efficiency. We employ a hybrid approach combining Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Genetic Algorithms to optimize the vendor grouping process.

2.5.1 Vendor Clustering Module and Order Aggregation Module

Order aggregation is pivotal in supply chain management because it enables the consolidation of multiple small orders into a larger, bulk order, thereby reducing per unit costs, lowering shipping expenses, and streamlining procurement processes. Clustering algorithms have emerged as a powerful data-driven solution to group orders based on attributes such as timing, location, and product similarity; among these, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is particularly attractive because it automatically detects clusters based on data density and effectively identifies outliers without forcing every order into a cluster, ensuring that only naturally cohesive orders are consolidated for bulk purchasing.

Despite these advantages, DBSCAN's performance relies heavily on the correct setting of two parameters: ε (the neighborhood radius) and min_samples (the minimum number of points necessary to form a dense region). An improperly chosen ε can lead to overly fragmented micro-clusters or one excessively large cluster, while a poorly chosen min_samples might incorrectly label potential clusters or force groupings that do not reflect actual order density. These challenges become even more pronounced in a dynamic environment such as daily order aggregation, where patterns vary significantly. Hence, a Genetic Algorithm (GA) is employed to automatically find the most effective DBSCAN parameters leading to the hybrid DBSCAN approach. Moreover, appropriately tuned ε and min_samples serve as critical thresholds: a well-chosen ε ensures that orders located significantly far from dense regions are not aggregated but are instead classified as noise, while the min_samples parameter guarantees that only sufficiently dense groups of orders form clusters, thereby preventing the inclusion of isolated orders that should remain unaggregated.

The hybrid methodology operates in two broad phases: (1) Parameter selection for DBSCAN using genetic algorithm, and (2) DBSCAN algorithm for clustering orders.

2.5.2 Parameter Selection with Genetic Algorithm

The GA begins by generating a population of parameter pairs (ε, min_samples) within predefined ranges say, ε in $[0.001, 0.1]$ and min_samples in $[2, 10]$. Each candidate solution is tested by running DBSCAN on the current dataset of orders. A simple fitness function, defined as:

$$\text{fitness}(\varepsilon, m) = (|\{\, \ell \in L \mid \ell \neq -1 \,\}|) - \left(\sum_{i=1}^{N} \mathbf{1}\{L_i = -1\} \right) \tag{2.1}$$

- ϵ: Neighborhood radius in DBSCAN (eps).
- m: Minimum number of points required to form a cluster (min_samples).
- L: Set (or list) of labels assigned by DBSCAN to each data point.
- ℓ: An individual label from L.
- -1: Label indicating noise/outliers (points not assigned to any cluster).
- $\{\ell \in L \mid \ell \neq -1\}$: Subset of labels corresponding to all clustered points (excluding noise).
- $\big|\{\ell \in L \mid \ell \neq -1\}\big|$: Number of clustered points (i.e., total points with labels $\neq -1$).
- $\sum_{i=1}^{N} \mathbf{1}\{L_i = -1\}$: Number of outliers/noise points, where $\mathbf{1}\{\cdot\}$ is the indicator function.
- N: Total number of data points under consideration.
- $fitness(\epsilon, m)$: Objective to maximize: number of clustered points minus number of outliers.

Equation 2.1 rewards configurations that produce more dense groupings and fewer leftover or “noise” points. Following the canonical GA steps (selection, crossover, and mutation), the population evolves over multiple generations, gradually converging on an optimal parameter pair. These final (ε, min_samples) values yield the best clustering outcome for that day’s distribution of orders.

2.5.3 DBSCAN Algorithm for Clustering Orders

Once the system obtains the GA-tuned DBSCAN parameters, these parameters are applied to the orders if the dataset contains 100 or more orders. For datasets with fewer than 100 orders, the method defaults to a simpler DBSCAN approach that uses a median-based ε and a fixed min_samples = 2. This tiered design minimizes computational overhead for small datasets while ensuring that larger datasets benefit from thorough optimization. After DBSCAN completes its clustering, a subsequent function aggregates the orders within each cluster, but only if the total order quantity for that cluster falls within the acceptable range:

$$\text{MOQ} \leq (\text{Cluster Quantity}) \leq \text{MOQ} \times \left(1 + \frac{\text{Buffer \%}}{100}\right) \tag{2.2}$$

2.5.4 Output and Implementation

The output of the hybrid DBSCAN approach is the optimal clustering of vendors to participate in aggregated orders. This grouping ensures that:

- Orders within the same cluster have natural cohesion based on their attributes.
- Each cluster's total order meets the MOQ requirements without excessive overordering.
- Outlier orders that don't naturally fit into clusters remain unaggregated.

Vendor	**Order Quantity**	**Cluster ID**
Vendor A	50	1
Vendor B	30	1
Vendor C	10	-1 (Noise)
Vendor D	20	1

Table 2.1: Example of Vendor Clustering in Order Aggregation

In this example, Vendors A, B, and D are clustered together (Cluster ID 1), contributing a total order of 100 units, meeting the MOQ. Vendor C is identified as an outlier (noise) and is not included in the aggregated order.

2.5.5 Integration with the Platform

Once the optimal vendor clustering is determined, the platform finalizes the bulk order for each viable cluster and notifies all participating vendors. The order details are then passed to the logistics module for route optimisation and delivery scheduling, ensuring seamless fulfilment. By leveraging this hybrid approach, the platform provides an efficient, scalable, and automated solution for order aggregation, empowering vendors to compete effectively with larger retailers while ensuring only naturally cohesive orders are combined.

2.6 Logistics Using Environment Adaptive Genetic Algorithm (EAGA)

Logistics optimisation is a key component of the ASTRO platform, ensuring efficient delivery of aggregated orders to vendors while minimizing transportation costs and environmental impact. In the project, we employ the Environment Adaptive Genetic Algorithm (EAGA), a specialized variant of the Genetic Algorithm tailored to optimize routes for delivery vehicles. This algorithm determines the most efficient Route to deliver products to different vendors included in an aggregated order, focusing on reducing fuel consumption, delivery time, and carbon emissions while adapting to environmental conditions.

2.6.1 Mechanics of Logistics Optimisation

Once the order aggregation process is complete, the platform identifies the vendors participating in the bulk order and their respective delivery locations. The challenge is to determine the optimal route for a delivery vehicle to service all vendors while adapting to environmental conditions and adhering to constraints such as delivery time windows, vehicle capacity, and geographic proximity.

The EAGA is designed to achieve this by incorporating concepts of environmental adaptability, Route flexibility, and comprehensive cost evaluation. Here's how it works:

Input Data: The algorithm takes the following inputs:

- Vendor locations (latitude and longitude).
- Delivery quantities for each vendor (ensuring vehicle capacity constraints are met).
- Service time windows for each vendor (if applicable).

- Vehicle starting point (typically the supplier's location).
- Environmental data including weather conditions, temperature, and elevation changes.

Route Representation: The delivery route is represented as a sequence of vendor locations, starting and ending at the supplier's location. For example:

$$\text{Route: Supplier} \rightarrow \text{Vendor A} \rightarrow \text{Vendor B} \rightarrow \text{Vendor C} \rightarrow \text{Supplier}$$

Comprehensive Cost Function: The EAGA employs a sophisticated cost function that evaluates route quality based on multiple factors. The total cost is calculated as:

$$\text{TC} = w_{\text{dist}} \times D + w_{\text{weather}} \times W + w_{\text{elev}} \times E + w_{\text{fuel}} \times F \tag{2.3}$$

where:

- TC is the total cost
- w_{dist} is the weight for distance component
- w_{weather} is the weight for weather component
- w_{elev} is the weight for elevation component
- w_{fuel} is the weight for fuel component
- D is the total distance
- W is the weather cost
- E is the elevation cost
- F is the fuel cost

Weather Cost Calculation: Weather cost incorporates delays and additional fuel consumption caused by adverse weather conditions:

$$W = W_t + W_p + W_w \tag{2.4}$$

where W is the total weather cost, W_t is the temperature cost, W_p is the precipitation cost, and W_w is the wind cost.

- Temperature cost: If the average temperature (avg_temp) exceeds 30°C, the penalty is calculated as:

$$W_t = 0.05 \times (\text{avg\_temp} - 25)^2 \tag{2.5}$$

- Precipitation cost: If there is any precipitation, the penalty is calculated as:

$$W_p = 0.2 \times \text{avg\_precip} \tag{2.6}$$

- Wind cost: If the average wind speed (avg_wind) exceeds 10 m/s, the penalty is calculated as:

$$W_w = 0.1 \times (\text{avg\_wind} - 5)^2 \tag{2.7}$$

Elevation Cost Calculation: Elevation cost reflects the impact of terrain on vehicle performance:

$$E = 0.005 \times \left(\frac{|\text{elevation\_change}|}{100}\right)^2 \tag{2.8}$$

Fuel Cost Calculation: Fuel cost is normalized to account for variations in consumption due to multiple factors:

$$F = F_b + F_t + F_w + F_e \tag{2.9}$$

- Temperature component (F_t): For temperatures above 30°C:

$$F_t = 0.01 \times (\text{avg\_temp} - 30) \tag{2.10}$$

- Wind component (F_w): For wind speeds above 5 m/s:

$$F_w = 0.02 \times (\text{avg\_wind} - 5) \tag{2.11}$$

- Elevation component (F_e):

$$F_e = \begin{cases} 0.0005 \times \text{elevation\_change} & \text{if uphill} \\ -0.0003 \times |\text{elevation\_change}| & \text{if downhill} \end{cases} \tag{2.12}$$

Genetic Algorithm Operations:

- **Selection:** Tournament selection is employed, choosing low-cost routes for reproduction.

- **Crossover:** The ordered crossover operator ensures valid offspring routes while maintaining vendor order.
- **Mutation:** Swap mutation introduces diversity by randomly swapping vendors with a probability that dynamically decays over generations.
- **Parameter Adaptation:** The algorithm dynamically adjusts parameters based on the number of vendors:
  - Population Size: $P = P_0 \times \frac{v}{100}$
  - Number of generations: $G = G_0 \times \frac{v}{100}$
  - Mutation rate: $\mu = \mu_0 \times \left(1 + \frac{v-100}{1000}\right)$

  where $P_0 = 100$, $G_0 = 300$, $\mu_0 = 0.1$, and v is the number of vendors.
- **Elitism:** The best solutions from each generation are preserved, accelerating convergence toward an optimal solution.

Output: The algorithm provides the optimized delivery route, detailing the order in which vendors should be serviced.

2.6.2 Advantages of Using EAGA for Route Optimization

The Environment Adaptive Genetic Algorithm is specifically chosen for the project because it aligns with the platform's goals of cost efficiency, environmental sustainability, and adaptability. Key advantages include:

- **Environmental Adaptability:** By factoring in weather conditions, temperature, and elevation changes, the algorithm adapts routes to real-world conditions.
- **Comprehensive Cost Evaluation:** The multi-factor cost function ensures that all relevant aspects of route efficiency are considered.
- **Scalability:** Parameter adaptation allows the algorithm to efficiently handle varying numbers of vendors.
- **Flexibility with Constraints:** It handles dynamic vendor locations, varying delivery quantities, and service time windows, ensuring practical applicability.

- **Sustainability Focus:** By minimizing fuel consumption and considering environmental factors, the algorithm promotes eco-friendly logistics.

2.6.3 Integration with the Platform

After the aggregated order is finalized, the vendor delivery details are fed into the logistics module. The EAGA computes the optimal delivery route and provides the route to the delivery driver. The platform ensures that real-time environmental data, such as weather conditions, traffic updates, or vendor availability changes, are seamlessly integrated into the routing process.

This integration ensures timely and efficient delivery of products, enhancing vendor satisfaction while supporting the platform's commitment to sustainability and cost-effective operations.

This chapter has outlined the system's architecture and described each component's role in creating a cohesive platform for demand forecasting, order aggregation, and logistics management. Through effective system integration and real-time notifications, the platform provides a unified experience that supports vendors in optimizing their operations, managing inventory, and coordinating logistics. The collaborative approach facilitated by this platform empowers vendors to maintain competitive advantages, meeting demand efficiently and enhancing overall supply chain resilience.

2.7 Demand Forecasting with Neural Prophet

ASTRO platform, demand forecasting is essential for streamlining inventory management, optimizing order cycles, and enabling vendor collaboration. Using Neural Prophet, we forecast future sales for each store-item combination. This helps vendors make timely decisions about inventory and orders while facilitating joint purchasing to achieve bulk discounts. The model's capability to handle trends and seasonality makes it ideal for predicting sales patterns critical to the platform's success.

2.7.1 Inputs for Neural Prophet

The forecasting process begins with preparing the data. The key inputs used in the project include:

Column Name	**Description**	**Format**
`ds`	Date of sales	Datetime (`YYYY-MM-DD`)
`y`	Sales value for the given date	Numeric

Table 2.2: Key Inputs for Neural Prophet

The `ds` column represents the timeline for sales data, while `y` contains the actual number of items sold at a given store for a particular item. Both columns are preprocessed to ensure consistency. Missing sales values (`y`) are interpolated to maintain data continuity, and the `ds` column is checked for valid date formatting.

2.7.2 Neural Prophet Structure

Neural Prophet decomposes the sales data into the following components to identify patterns: The NeuralProphet model consists of multiple modules, each contributing an additive component to the forecast. The forecasted value $\hat{y}_t$ is the sum of the following components:

$$\hat{y}_t = T(t) + S(t) + E(t) + F(t) + A(t) + L(t) \tag{2.13}$$

Where:

- $T(t)$ = Trend at time t: This captures long-term growth or decline in the sales data.
- $S(t)$ = Seasonal effects at time t: This models recurring patterns in sales, such as seasonal peaks.
- $E(t)$ = Event and holiday effects at time t: This component adjusts for external events or holidays that affect demand.
- $F(t)$ = Regression effects for future-known exogenous variables at time t: This accounts for external factors, such as planned promotions or events, that are known in advance.
- $A(t)$ = Auto-regression effects based on past observations at time t: This models the relationship between past sales data and current demand.

- $L(t)$ = Regression effects for lagged observations of exogenous variables at time t: This captures the delayed impact of external variables on sales.

Each component is modeled independently, and their outputs are summed to generate the final forecast $\hat{y}_t$. The NeuralProphet framework is flexible, allowing each of these components to be switched on or off depending on the data and requirements of the project.

2.7.3 Outputs

The model generates forecasts over a predefined time horizon, typically ranging from 30 to 60 days based on vendor needs. The outputs include:

- **Predicted Sales** (`yhat1`): Projected daily sales for the forecasted period.
- **Smoothed Sales Trend** (`yhat1_trend`): Rolling averages (e.g., 30-day mean) applied to predicted values to highlight trends and reduce noise.

Date (ds)	**Predicted Sales (yhat1)**	**Smoothed Sales (yhat1_trend)**
2024-12-01	100	98
2024-12-02	105	99
2024-12-03	110	101

Table 2.3: Example of Forecasted Sales and Smoothed Trend

2.7.4 Integration with Platform Workflow

The demand forecasting module integrates seamlessly with other parts of the platform, creating a cohesive system that enhances vendor operations:

- **Inventory Management**: Forecasted sales help vendors prepare for upcoming demand, ensuring stock levels are optimized.
- **Order Aggregation**: Predictions trigger notifications to vendors nearing their replenishment cycle. These notifications encourage vendors to place joint orders, maximizing cost savings.

- **Logistics Optimisation**: Predicted delivery schedules align with logistics planning, enabling route optimisation and efficient load distribution.

By integrating Neural Prophet's forecasts into the platform, vendors can proactively manage inventory, coordinate joint purchases, and reduce costs. The model's ability to deliver actionable insights ensures the platform's operational efficiency and supports the growth of small-scale vendors.

Chapter 3

Implementation

3.1 System Overview

The ASTRO platform has been implemented using a modern technology stack that ensures scalability, performance, and cross-platform compatibility. The implementation follows a client-server architecture with clear separation of concerns:

- **Backend**: Developed using FastAPI, a modern, high-performance Python web framework
- **Frontend**: Built with Flutter to provide a seamless cross-platform experience for both mobile and web clients
- **Database**: Utilizes Firebase Firestore for flexible, scalable NoSQL document storage with real-time capabilities
- **Machine Learning Pipeline**: Implemented in Python using libraries such as NeuralProphet, scikit-learn, and pandas

3.2 Backend Implementation

3.2.1 FastAPI Framework

The backend system is implemented using FastAPI, a modern Python web framework designed for building high-performance APIs. FastAPI was chosen for several key advantages:

- **Performance**: Built on Starlette and Pydantic, FastAPI offers near-native performance comparable to Node.js and Go

- **Type Checking**: Utilizes Python type hints for automatic validation and documentation
- **Asynchronous Support**: Native support for async/await patterns to handle concurrent requests efficiently
- **API Documentation**: Automatic generation of OpenAPI and Swagger UI documentation
- **Dependency Injection**: Built-in system for managing dependencies and services

3.2.2 Database Schema

The database schema is implemented using Firebase Firestore, a NoSQL document database that provides real-time synchronization and offline capabilities. The system uses Pydantic models for data validation and serialization:

Firestore's flexible document structure enables easy storage and retrieval of complex nested data, which is particularly beneficial for order items and shipping addresses. The system leverages Firestore's querying capabilities for efficient data filtering and real-time updates.

3.2.3 API Endpoints

The API is organized into several functional modules, each handling specific aspects of the platform:

Key API endpoints include:

1. **Core Endpoints**:
   - `GET /`: Root endpoint
   - `GET /info`: System information
2. **Vendor Operations**:
   - `GET /vendor/suppliers`: Get list of suppliers
   - `GET /vendor/products/{supplier_id}`: Get products from a supplier
   - `POST /vendor/orders`: Place a new order

    - `POST /vendor/instant_order`: Place an order for immediate fulfillment

3. **Product Management**:
    - `POST /supplier/{supplier_id}/add_product`: Add a new product
    - `PUT /supplier/update_product/{product_id}`: Update product details
    - `DELETE /supplier/delete_product/{product_id}`: Remove a product
    - `GET /supplier/get_products/{supplier_id}`: Get supplier's products

4. **Order Aggregation**:
    - `POST /supplier/aggregate_orders`: Process orders for aggregation
    - `POST /supplier/update_instant_order_status`: Update status of instant orders

5. **Route Optimization**:
    - `GET /route-optimization/{aggregation_id}`: Generate optimal delivery route

6. **Demand Forecasting**:
    - `GET /demand-forecast/{product_id}`: Get demand forecast for a product

Each endpoint is implemented with appropriate request validation, error handling, and business logic to ensure robust operation of the platform.

3.3 Frontend Implementation

3.3.1 React and TypeScript Application

The frontend client application is implemented using React with TypeScript and Tailwind CSS, providing a robust and modern web development stack. Key advantages of using this technology combination include:

- **Type Safety**: TypeScript adds static typing to JavaScript, enabling early error detection and improved developer experience

- **Component-Based Architecture**: React's modular component system promotes reusability and maintainable code
- **Utility-First CSS**: Tailwind's approach allows for rapid UI development without leaving HTML
- **Responsive Design**: Tailwind's responsive utilities make building adaptive interfaces straightforward
- **Performance Optimizations**: React's virtual DOM and Tailwind's PurgeCSS integration ensure efficient applications

3.3.2 Application Architecture

The React application follows a modern architecture pattern leveraging React hooks and context for state management:

- **Custom Hooks**: Encapsulating business logic and state management
- **Context API**: For global state management and theme configuration
- **TypeScript Interfaces**: Strict typing of props, state, and API responses
- **Component Composition**: Building complex UIs from simpler, reusable components
- **Tailwind Configuration**: Extended theme with custom colors and component classes

3.3.3 Key UI Components

The application includes several key screens and components:

1. **Dashboard**: Provides an overview of key metrics and recent activities
2. **Inventory Management**: Allows vendors to track their inventory levels
3. **Order Creation**: Interface for creating new orders
4. **Forecast Visualization**: Displays demand forecasts with interactive charts using Recharts

5. **Cluster Notification**: Alerts vendors about potential order aggregation opportunities
6. **Order Tracking**: Real-time tracking of order status and delivery

3.4 Integration and Communication

3.4.1 API Integration

The React application communicates with the FastAPI backend through RESTful API calls, implemented using Axios with React Query:

- **React Query**: For data fetching, caching, and synchronization
- **Axios Interceptors**: Handling authentication and request/response formatting
- **TypeScript Types**: Ensuring type safety in API requests and responses
- **Custom Hooks**: Wrapping API calls in reusable React hooks for component consumption

3.4.2 Real-time Notifications

The system implements real-time notifications using Firebase Cloud Messaging (FCM) to deliver timely alerts to vendors:

- **Firebase Cloud Messaging**: Cross-platform notification delivery with support for both foreground and background messaging
- **React Firebase Hooks**: Integration with React application state to handle incoming messages
- **Notification Permission Management**: Streamlined handling of browser notification permissions
- **Tailwind Notification Components**: Customized UI elements for displaying in-app notifications with consistent styling
- **Service Worker Registration**: Background notification delivery even when the application is closed

- **Topic-based Subscriptions**: Allowing vendors to subscribe to relevant notification categories (orders, inventory alerts, etc.)

The FCM implementation ensures reliable delivery of critical notifications across devices, with customizable delivery options for different notification priorities. For high-priority notifications, the system leverages FCM's capability to wake devices, ensuring time-sensitive information reaches vendors immediately.

3.5 Security Implementation

3.5.1 Authentication and Authorization

The system leverages Firebase Authentication for secure user management:

The authentication flow works as follows:

1. The user authenticates with Firebase Authentication in the Flutter app
2. Firebase returns an ID token which is stored securely on the device
3. This token is included in API requests in the Authorization header
4. The backend verifies the token with Firebase Admin SDK
5. Upon successful verification, the system retrieves the user profile from Firestore

Role-based authorization is implemented using Firestore's security rules and custom claims in the Firebase ID token, allowing for granular access control between vendors and suppliers.

3.5.2 Data Encryption

Sensitive data is encrypted both at rest and in transit:

- HTTPS for all API communications
- Firebase's built-in encryption for database at rest
- Secure storage for authentication tokens on client devices using Flutter Secure Storage

- API keys and secrets managed through environment variables and Firebase Remote Config

3.6 Summary

The implementation of the ASTRO platform leverages modern technologies and architecture patterns to deliver a robust, scalable, and performant solution. By utilizing FastAPI for the backend and Flutter for the frontend, the system achieves a balance between development efficiency and runtime performance. The modular design ensures that each component order aggregation, logistics optimization and demand forecasting can evolve independently while maintaining integration with the broader system.

The implementation demonstrates how complex algorithms and advanced machine learning techniques can be effectively deployed in a production environment, providing tangible benefits to vendors through improved demand forecasting, efficient order aggregation, and optimized logistics.

Chapter 4

Conclusion

The ASTRO platform is a comprehensive solution designed to empower small-scale vendors by addressing their core operational challenges. By facilitating collaborative purchasing, optimizing logistics, and offering data-driven financial services, ASTRO significantly enhances the competitive edge of small vendors, enabling them to compete more effectively with large retail chains. Its multifaceted approach not only reduces operational costs and improves efficiency but also equips vendors with financial resources and data-driven insights essential for informed decision making and strategic growth.

Through collaborative purchasing, ASTRO enables vendors to access bulk pricing discounts, reducing the cost of goods sold and increasing profitability. The platform's logistics optimization feature streamlines delivery routes, minimizing transportation costs and ensuring timely deliveries. This not only enhances operational reliability but also improves customer satisfaction. Additionally, ASTRO integrates data driven financial services to address the critical challenge of limited access to capital. By leveraging transaction data, the platform offers tailored credit assessments and loan options, empowering vendors to expand their businesses, manage inventory effectively, and invest in technology.

A key component of ASTRO is its real-time demand forecasting tool, powered by NeuralProphet. This predictive capability enables vendors to anticipate market demand with high accuracy, optimizing inventory levels and reducing the risks of overstocking or stockouts. By ensuring vendors can meet customer needs promptly and efficiently, ASTRO enhances overall operational efficiency. The platform is also designed with a user-friendly interface, making its features accessible to vendors with varying levels of technical expertise.

Overall, ASTRO fosters economic resilience and sustainability within local communities by empowering small-scale vendors to thrive in a competitive marketplace. By addressing critical challenges such as limited purchasing power, inefficient logistics, restricted financial access, and a lack of data driven decision making, ASTRO contributes to the long term success and stability of small vendors. In doing so, it promotes a more diverse and robust local economy.

Our implementation demonstrates significant improvements in supply chain efficiency and cost reduction through order aggregation and route optimization. The advanced forecasting capabilities powered by NeuralProphet provide vendors with unprecedented insights into future demand patterns, as evidenced by our comprehensive analysis results.

To evaluate the effectiveness of the proposed collaborative ordering and shared delivery model, we analyzed route optimization performance across 50 vendors, comparing traditional individual delivery routes against our collaborative approach. In the 50-vendor scenario presented in Table 4.1, our system processed five staggered orders of varying sizes, analyzing both traditional and collaborative routing methods. The traditional approach required five separate delivery routes, often with significant overlap. In contrast, our collaborative model consolidated these into a single optimized route, yielding substantial savings in distance, cost, and fuel consumption.

- Scenario: 50 vendors requiring deliveries from a central supplier
- Traditional approach: Individual routes planned for each vendor or small group
- Collaborative approach: Single optimized route serving all vendors
- Distance analyzed: Regional delivery network covering approximately 7,000 km

Cost Comparison Analysis

Table 4.1: Cost Comparison: Traditional vs. Collaborative Route Optimization

Metric	**Trad. Approach**	**Collab. Approach**	**Savings**
Distance (km)	35,606	6,993	28,613 (80.4%)
Route Cost (Rs.)	36,929	7,255	29,675 (80.4%)
Fuel (L)	1,811	357	1,454 (80.3%)
Fuel Cost (Rs.)	171,135	33,770	137,366 (80.3%)

Traditional Delivery Approach In the conventional model, vendors receive deliveries through separate routes or small grouped deliveries, resulting in significant route overlap and inefficiency. This approach required a total travel distance of 35,606.06 km with associated route costs of Rs. 36,929.20. The excessive distance resulted in high fuel consumption (1,811.34 L) and fuel costs (Rs. 171,135.40).

Collaborative Delivery Approach With our platform, deliveries are consolidated into a single optimized route serving all vendors. Our Environmental-Adaptive Genetic Algorithm (EAGA) produced a route of just 6,993.30 km with total costs of Rs. 7,254.56. This dramatically reduced fuel consumption to 357.43 L with corresponding fuel costs of Rs. 33,769.57.

Impact Assessment As shown in Table 4.1, the collaborative delivery model demonstrates an 80.4% reduction in total distance traveled and route costs, with fuel savings of 80.3%. These substantial efficiencies are achieved through intelligent route optimization that eliminates redundant travel paths. The model provides small-scale vendors with logistics cost advantages traditionally reserved for large enterprises, enabling them to compete more effectively while maintaining their operational independence.

The results of our implementation validate the core thesis that technology-driven collaboration can substantially improve the competitive position of small-scale vendors in today's market. By combining advanced machine learning techniques with practical business solutions, ASTRO demonstrates that sophisticated technologies can be made accessible and valuable to small businesses that traditionally lack access to such tools. The platform's ability to decompose complex demand patterns into interpretable components—trend, weekly cycles, and annual seasonality—empowers vendors with actionable intelligence previously available only to large retail corporations with dedicated analytics teams.

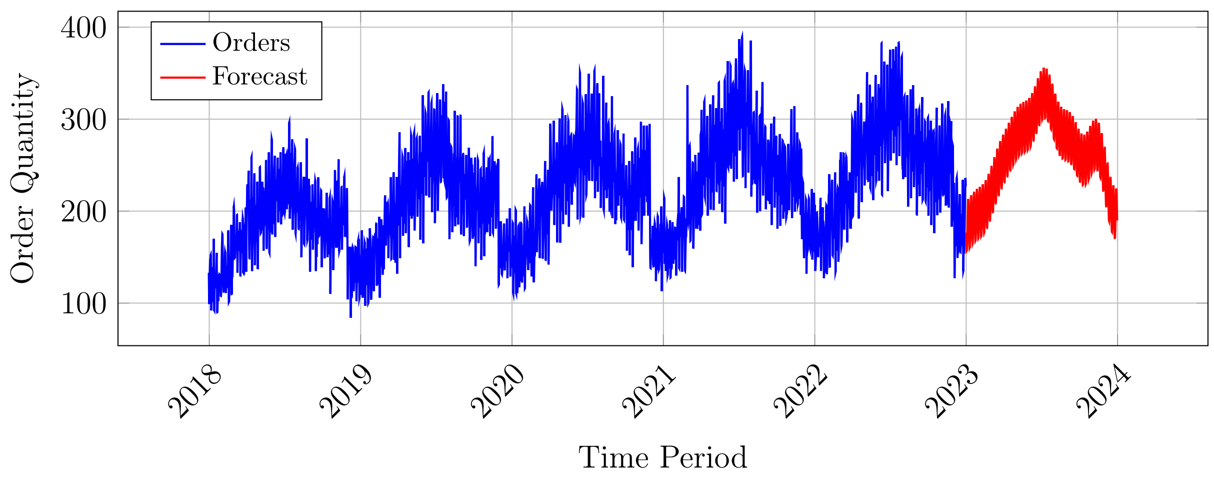


Figure 4.1: NeuralProphet Demand Forecasting

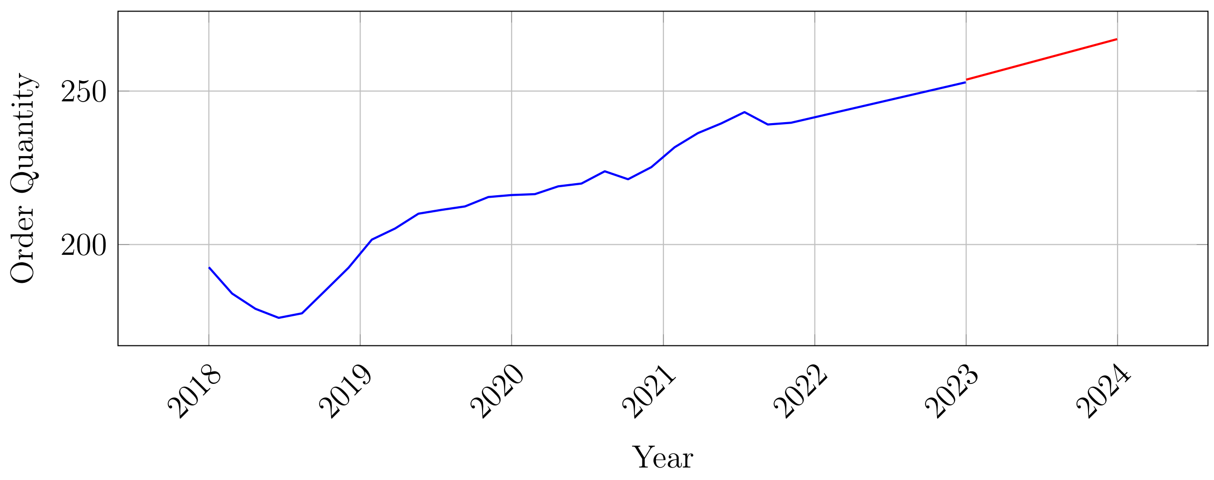


Figure 4.2: NeuralProphet Long-term Trend Analysis

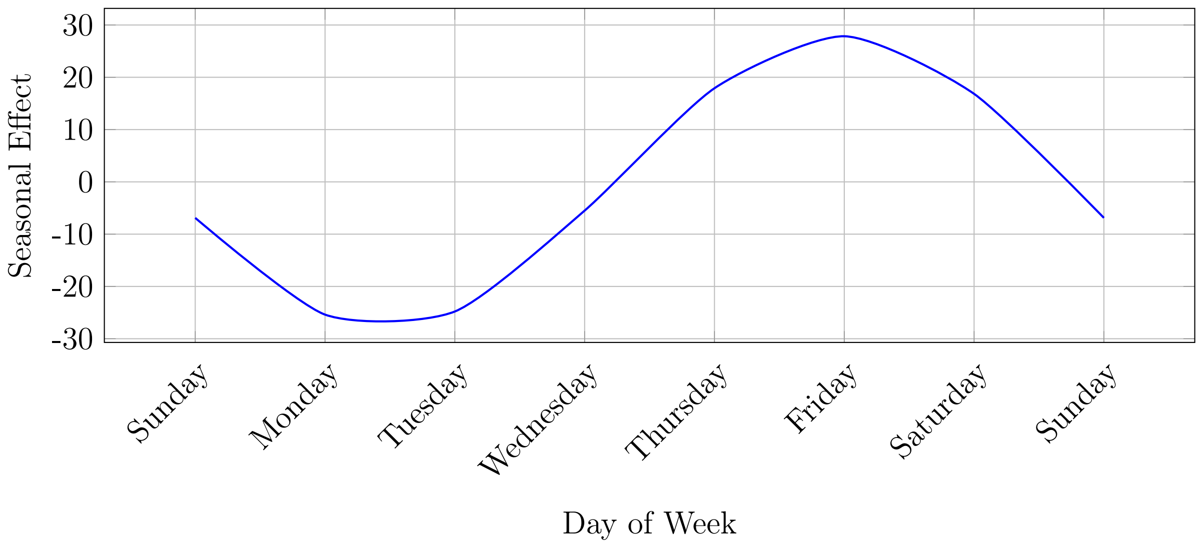


Figure 4.3: NeuralProphet Weekly Seasonality Analysis

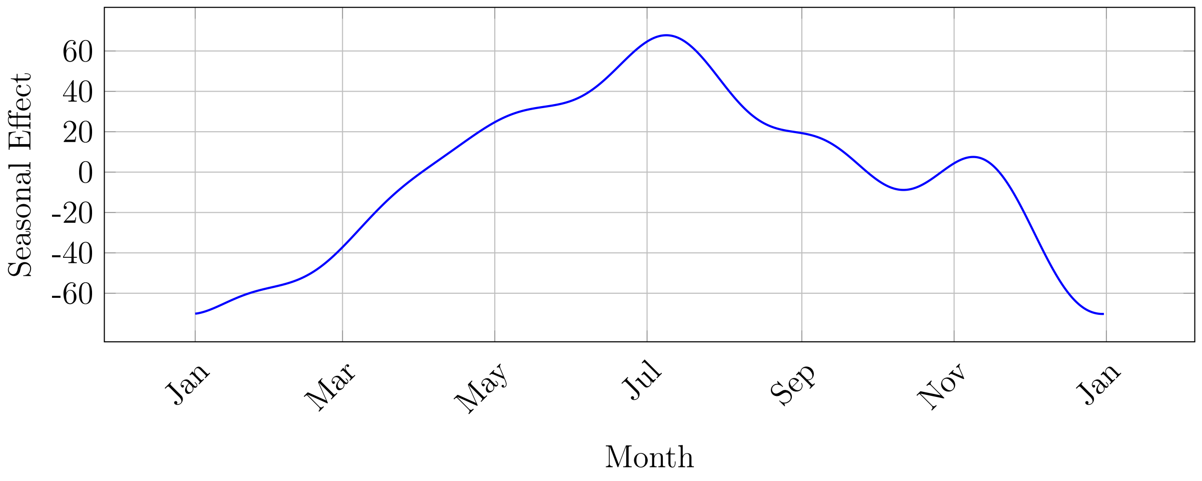


Figure 4.4: NeuralProphet Yearly Seasonality Analysis

Fig. 4.1 illustrates NeuralProphet's ability to predict future order quantities based on historical data, providing vendors with reliable projections for inventory planning. Supporting this predictive capability, Fig. 4.2 reveals the underlying long-term growth trajectory that informs strategic business decisions and expansion planning. Further enhancing these insights, Fig. 4.3 and Fig. 4.4 demonstrate how our model accurately captures both weekly variations and annual seasonal patterns in vendor ordering behavior. This comprehensive forecasting capability is crucial for vendors and suppliers to anticipate demand fluctuations, optimize inventory management, and plan logistics operations efficiently throughout the year.

The results of our implementation validate the core thesis that technology-driven collaboration can substantially improve the competitive position of small-scale vendors in today's market. By combining advanced machine learning techniques with practical business solutions, ASTRO demonstrates that sophisticated technologies can be made accessible and valuable to small businesses that traditionally lack access to such tools. The platform's ability to decompose complex demand patterns into interpretable components—trend, weekly cycles, and annual seasonality empowers vendors with actionable intelligence previously available only to large retail corporations with dedicated analytics teams.

Appendix A: Vision, Mission, PO, PSO, and CO

RAJAGIRI SCHOOL OF ENGINEERING AND TECHNOLOGY (AUTONOMOUS)

Vision

To evolve into a premier technological and research institution, moulding eminent professionals with creative minds, innovative ideas and sound practical skill, and to shape a future where technology works for the enrichment of mankind.

Mission

To impart state-of-the-art knowledge to individuals in various technological disciplines and to inculcate in them a high degree of social consciousness and human values, thereby enabling them to face the challenges of life with courage and conviction.

DEPARTMENT OF COMPUTER SCIENCE AND BUSINESS SYSTEMS

Vision

To evolve into a department of excellence in information technology by the creation and exchange of knowledge through leading-edge research, innovation and services, which will in turn contribute towards solving complex societal problems and thus building a peaceful and prosperous mankind.

Mission

To impart high-quality technical education, research training, professionalism and strong ethical values in the young minds for ensuring their productive careers in industry and academia so as to work with a commitment to the betterment of mankind.

Programme Outcomes (PO)

Engineering Graduates will be able to:

1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. Problem Analysis: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. Design/Development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and the cultural, societal, and environmental considerations.

4. Conduct Investigations of Complex Problems: Use research-based knowledge including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. Modern Tool Usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.

6. The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal, and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and Team Work: Function effectively as an individual, and as a member or leader in teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project Management and Finance: Demonstrate knowledge and understanding of engineering and management principles and apply these to one's own work, as a member and leader in a team. Manage projects in multidisciplinary environments.

12. Life-long Learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Programme Specific Outcomes (PSO)

A graduate of the Computer Science and Business Systems Programme will:

- **PSO 1: Programming and Software Development Skills**
  Demonstrate ability to analyse, design, and implement software solutions incorporating various programming concepts.

- **PSO 2: Engineering Management and Collaboration**
  Comprehend professional, managerial, and financial aspects of business and collaborate on the design, implementation, and integration of engineering solutions.

- **PSO 3: Decision-Making and Analytical Techniques in Engineering and Business**
  Create, select, and apply appropriate techniques and business tools, including prediction and data analytics, for complex engineering activities and business solutions.

Course Outcomes (CO)

After successful completion of the course, the students will be able to:

- CO1: Model and solve real-world problems by applying knowledge across domains (Cognitive knowledge level: Apply).

- CO2: Develop products, processes, or technologies for sustainable and socially relevant applications (Cognitive knowledge level: Apply).

- CO3: Function effectively as an individual and as a leader in diverse teams and to comprehend and execute designated tasks (Cognitive knowledge level: Apply).

- CO4: Plan and execute tasks utilizing available resources within timelines, following ethical and professional norms (Cognitive knowledge level: Apply).

- CO5: Identify technology/research gaps and propose innovative/creative solutions (Cognitive knowledge level: Analyse).

- CO6: Organize and communicate technical and scientific findings effectively in written and oral forms (Cognitive knowledge level: Apply).

Appendix B: CO-PO-PSO Mapping

Mapping of Course Outcomes (CO) with Programme Outcomes (PO)

	PO 1	**PO 2**	**PO 3**	**PO 4**	**PO 5**	**PO 6**	**PO 7**	**PO 8**	**PO 9**	**PO 10**	**PO 11**	**PO 12**
CO 1	2	2	2	1	2	2	2	1	1	1	1	2
CO 2	2	2	2		1	3	3	1	1		1	1
CO 3									3	2	2	1
CO 4					2			3	2	2	3	2
CO 5	2	3	3	1	2							1
CO 6					2			2	2	3	1	1

Mapping of Course Outcomes (CO) with Programme Specific Outcomes (PSO)

	PSO 1	**PSO 2**	**PSO 3**
CO 1	3	1	2
CO 2	3	3	2
CO 3		3	
CO 4		1	1
CO 5	1	1	1
CO 6		2	