College of Excellence, .nirf 2024- 7thRank

Autonomous and Affiliated to Bharathiar University

Accredited with A++ grade by NAAC, An ISO 9001:2015 Certified Institution

Peelamedu, Coimbatore – 641004.

RETAIL SALES ANALYSIS AND FORECASTING USING MACHINE LEARNING

Internship work submitted to PSGR Krishnammal College for women
in partial fulfilment of the requirements for the award of the degree of
Master of Science in Data Analytics
Bharathiar University, Coimbatore – 641046

Done By

SHALINI PREETHA A D

24SMDA041

Guided By

Ms. M. NANDHINI, M.Sc., M.Phil., (Ph.D).,

Assistant Professor, Department of Data Analytics (PG)

PSGR Krishnammal College for Women

Coimbatore-641004.

JUNE 2025

CERTIFICATE

This is to certify that this internship work entitled **"RETAIL SALES ANALYSIS AND FORECASTING USING MACHINE LEARNING"** submitted to PSGR Krishnammal College For Women, Coimbatore in partial fulfilment of the requirement for the award of Master of Science in Data Analytics is a record of original work done **SHALINI PREETHA A D** (24SMDA041) during her period of study in the Department of Data Analytics (PG), PSGR Krishnammal College for Women, Coimbatore under my supervision and guidance and project work has not formed the basis for the award of any other Degree /Diploma/Associateship/Fellowship or any similar title to any candidate of any University.

Forwarded By

Ms. M. NANDHINI, M.Sc., M.Phil., (Ph.D.).,	**Dr. S. SASIKALA, M.SC., M.Phil., Ph.D.,**
Faculty Guide	**Head of the Department**

DECLARATION

I hereby declare that this internship work entitled "**RETAIL SALES ANALYSIS AND FORECASTING USING MACHINE LEARNING**" submitted to PSGR Krishnammal College For Women, Coimbatore, for the award of the Degree of Master of Science in Data Analytics, is a record of original work done by **SHALINI PREETHA A D (24SMDA041)** under the supervision and guidance of **Ms.M.NANDHINI MSc.,M.Phil., (Ph.D).,** Department of Data Analytics (PG), PSGR Krishnammal College for Women, Coimbatore, and that this internship work has not formed the basis for the award of any other Degree/Diploma/Associateship/Fellowship or any similar title to any candidate of any University.

SHALINI PREETHA A D
24SMDA041

Endorsed By

PLACE: COIMBATORE

DATE:

_____

Ms. M. NANDHINI, MSc., M.Phil., (Ph.D).,
Faculty Guide

ACKNOWLEDGEMENT

"Success is to be measured not so much by the position that one has reached in life but as by the obstacle which he had overcome while trying to success.

I extend my thanks to **Dr. (Mrs.) R. NANDINI,** Chairperson**,** PSGR Krishnammal College for women, and Coimbatore for her full support and for all the resources provided.

I express my whole hearted thanks to **Dr. (Mrs.) N.YESODHA DEVI, M.com, M.Phil., Ph.D.,** and Secretary PSGR Krishnammal College for women, and Coimbatore for having given me the opportunity to undertake this internship work.

I extend my thanks to **Dr. (Mrs.) P. HARATHI, M.SC., M.Phil., Ph.D., Principal,** PSGR Krishnammal College for women, and Coimbatore for her full support and for all the resources provided.

I am extremely grateful to **Dr.S.SASIKALA, M.SC., M.Phil., Ph.D., Head of the Department of Data Analytics (PG),** PSGR Krishnammal college for women, and Coimbatore for her sustained interest and advice that have contributed to a great extent to the completion of the internship work.

I wish my indebtedness to my guide **Ms.M.NANDHINI, MSc.,M.Phil.,(Ph.D), Assistant Professor**, Department of Data Analytics (PG), and PSGR Krishnammal College for women Coimbatore for her appropriate guidance, suggestions and support in the completion of this internship work.

I extend my sincere thanks to SKYPARK ITECH PRIVATE LIMITED, Coimbatore, for their kind support and contribution to the successful completion of the internship work.

My sincere thanks to all my staff of Department of Data Analytics (PG) for their timely support and encouragement.

Finally, I place on record my deep sense of gratitude to my beloved parents and to my friends for their timely support in completing this internship work.

ABSTRACT

Retail businesses, particularly those dealing with traditional products such as sarees, often face challenges in accurately predicting future sales due to fluctuating customer preferences, seasonal variations, festival-driven demand, and market competition. These uncertainties can result in inefficient inventory management, overstocking, or missed sales opportunities, ultimately impacting profitability and customer satisfaction. To address these challenges, Retail sales analysis applies machine learning techniques, specifically Linear Regression, to analyse historical sales data and forecast future sales for a local saree retail shop. The primary objective is to build a data-driven solution that empowers retailers to make informed decisions about inventory planning and sales strategies.

Phase I includes the integration of customer, purchase, and product data into a structured relational database using MySQL, the system enables seamless data management and analysis. Retail sales analysis follows a complete pipeline starting from data collection, preprocessing, and exploratory data analysis, to model training, forecasting, and deployment through a web-based dashboard using Flask. The Linear Regression model was chosen for its efficiency in modelling linear trends within time-series sales data. The predictions generated by the model offer valuable insights into upcoming demand trends, which are visualized through an interactive web application. Phase II includes the dashboard that displays key metrics such as monthly revenue, top-selling saree types, high-value customers, and remaining stock levels, thus helping store owners make better decisions in real time and demonstrates the practical application of machine learning in the retail sector, enabling businesses to improve operational efficiency, enhance customer satisfaction, and drive sustained growth.

1. INTRODUCTION

In today's rapidly evolving business landscape, the retail industry plays a crucial role in shaping consumer experiences and driving economic growth. However, one of the main challenges faced by retailers, especially in traditional and small-scale businesses such as local saree shops is the unpredictability of sales patterns. These fluctuations are often driven by numerous factors, including seasonal changes, festival-driven demand, regional preferences, promotional events, and market competition. For businesses that lack access to advanced analytics tools, this unpredictability can lead to poor inventory management, missed sales opportunities, and increased operational costs.

The traditional saree retail sector is highly dependent on specific cultural and seasonal events that influence customer buying behaviour. Retailers often make stocking and marketing decisions manually or based on past experiences rather than data. While this approach may have worked in the past, the increasing availability of digital tools and data presents a compelling opportunity to shift towards more intelligent and automated decision-making processes. Leveraging data science and machine learning can enable these businesses to transform their operations from reactive to proactive.

The system is built using widely adopted technologies including Python for data processing and machine learning, MySQL for structured data storage, and Flask for developing a web-based interface that enables real-time interaction with the sales and customer data. The ultimate goal of this project is to demonstrate the value of machine learning in a traditional retail context, making predictive analytics accessible and actionable for small business owners.

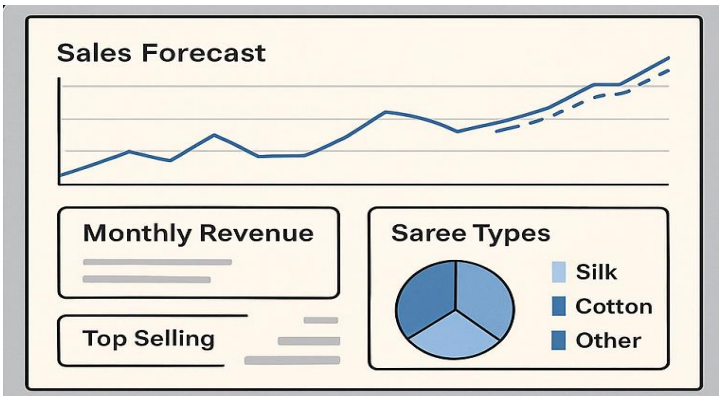


Fig 1 – Sample image for Retail sales Analysis and Forecasting

1.1 ORGANIZATION PROFILE

The internship project was undertaken at Skypark Itech, a Coimbatore-based training and IT services company that has rapidly emerged as a leading company in the technology space since its inception in 2021. The organization was founded with the mission to empower businesses and individuals by delivering high-quality, future-ready technological solutions. Skypark Itech has focusing on innovation, customer-centric development, and continuous learning, especially in the fields of Information Technology and Drone Technology. Skypark Itech is a forwardthinking organization dedicated to advancing technology through innovation. It offers an ideal platform for students to explore their technical capabilities while contributing to meaningful and impactful industry projects.

1.2 PROBLEM STATEMENT

- In the competitive saree retail market, sales are heavily influenced by seasonal festivals, regional preferences, promotions, and market competition. Many small and medium retailers rely on intuition rather than data for stocking decisions, often leading to overstocking or understocking, financial losses, and customer dissatisfaction.
- Retail sales analysis aims to address these challenges by developing a data-driven sales forecasting system using historical sales data and Linear Regression to improve inventory management and business efficiency.

1.3 TOOL DESCRIPTION

PYTHON

Python was the primary programming language used throughout the project, known for its readability and extensive library support. For data handling and preprocessing, Python's pandas library allows efficient extraction, cleaning, and aggregation of sales data from the MySQL database. For machine learning, Python provides the scikit-learn library, which is used to implement the Linear Regression model to analyse historical sales trends and forecast future sales. In terms of data visualization, libraries like Plotly and Matplotlib allow the creation of interactive and visually appealing charts for monthly revenue, top-selling sarees, and customer spending patterns.

Table 1.3.1 Tool Description

TOOLS	DESCRIPTION
PYCHARM	• PyCharm, developed by JetBrains, is an Integrated Development Environment (IDE) tailored for Python development. It was used to write, test, and debug all Python scripts efficiently. • It's user-friendly interface, code completion, error highlighting, and support for virtual environments made it easier to manage the project's modules and maintain code quality.
FLASK	• Flask is a lightweight and flexible web framework in Python used to develop the web interface of the application. • In Retail sales analysis, Flask was used to connect the backend machine learning logic and the MySQL database with a user-friendly frontend that displays results, forecasts, and sales dashboards.
MYSQL	• MySQL is an open-source Relational Database Management System (RDBMS) used to store and manage the project's data. • In Retails sales analysis, it is used to handle structured data efficiently across four main tables: customers, purchases, purchased items, and sarees. • MySQL's robustness and query capabilities made it suitable for storing large volumes of sales data, retrieving filtered metrics, and supporting real-time interactions with the Flask web application.

PACKAGES

Table 1.3.2 Packages

PANDAS	• Pandas is a powerful data manipulation and analysis library. • In Retail sales analysis, it was used for cleaning, transforming, and analysing data retrieved from the MySQL database, as well as preparing datasets for modeling. • Syntax for importing: "**import pandas as pd".**
MATPLOTLIB	• Matplotlib is a comprehensive plotting library for static charts. • It is used for generating basic sales visualizations and data analysis graphs during the exploratory data analysis phase. • Syntax for importing: **import matplotlib.py as plt.**
PLOTLY	• Plotly is a graphing library used to create interactive and visually appealing charts for the web interface. • It helps display sales trends, customer metrics, and forecast data on the dashboard. • Syntax for importing: **import plotly.express as px.**
UUID	• This module is used to generate universally unique identifiers (UUIDs), which can be assigned to customer or purchase IDs to ensure data uniqueness and integrity. • Syntax for importing: **import uuid.**
MYSQL CONNECTOR	• This package enables Python to connect directly to MySQL databases. • It is used for executing SQL queries, fetching data, and performing insert/update operations from the Flask backend. • Syntax for importing: **import mysqlconnector.**

2.LITERATURE REVIEW

S.NO	Author(s)	Title	Objectives	Findings
1	Chopra & Ghosh (2020) [1]	Retail Sales Forecasting Using Machine Learning Techniques	Compared regression and tree-based models for predicting store-level sales using historical data.	Supports the use of regression (Linear Regression) for retail forecasting.
2	Suresh & Ramya (2021) [2]	Predictive Analysis in Retail Industry Using Python	Demonstrated forecasting models implemented in Python with real-world datasets.	Validates use of Python and ML libraries for small retail data analytics.
3	Jain & Singh (2020) [3]	Retail Analytics using Python and MySQL	Built a system integrating sales data with a MySQL backend for reporting and analysis.	Backs the project architecture: MySQL for data + Python for analytics.
4	Mishra & Rajan (2021) [4]	Inventory Optimization Using Data Science in Retail	Explored predictive analytics to reduce stockouts and overstocking in small retail environments.	Highlights the importance of sales prediction in inventory management.
5	Smith & Kumar (2022) [5]	Lightweight Forecasting Models for Small-Scale Retailers	Showed that simple models like Linear Regression can be effective when data is limited, offering interpretability and low computational cost.	Supports the choice of Linear Regression for saree sales forecasting with small and structured datasets.

3. DOMAIN - RETAIL ANALYTICS

The domain of this project lies in Retail Analytics, a crucial and rapidly growing area within the broader field of Business Analytics and Data Science. Retail Analytics involves the collection, analysis, and interpretation of data generated from retail operations to support better business decision-making. This includes evaluating sales performance, forecasting future demand, managing inventory, understanding customer behaviour, and improving overall operational efficiency. In the context of this project, the focus is specifically on the application of machine learning techniques to analyse historical sales data and forecast future performance for a local saree retail business.

Retail sales analysis also intersects with Sales Forecasting, a key subdomain within retail analytics. Sales forecasting involves using past transactional data to predict future sales trends. Accurate sales forecasting enables retailers to plan better for upcoming demand, manage stock levels efficiently, allocate budgets strategically, and avoid financial losses due to overstocking or stockouts. By using Linear Regression, a widely used machine learning algorithm, this project develops a predictive model that estimates the retailer's sales for the next three months.

Retail sales analysis belongs to the domain of Retail Analytics, with strong ties to Sales Forecasting and Applied Machine Learning. It demonstrates how traditional retail businesses can using the power of data science to become more agile, efficient, and competitive in a fast-evolving marketplace.

4.DATA MODELLING

4.1 DATA COLLECTION

In Retail sales analysis, data was collected from a local saree retail shop, which maintains sales and customer records as part of its daily operations. The data consisted of real or simulated transactional entries representing various aspects of the retail environment, including customer details, purchase history, product information, and item-level purchase data.

A	B	C	D	E	F	G
customer_id	customer_name	customer_dob	customer_number	customer_	customer_zipcode	customer_gend
CUST0001	Keya Nadkarni	03-10-1971	7028613734	Mumbai	400001	F
CUST0002	Gaurang Vaidya	16-03-1962	8982850698	Salem	636003	M
CUST0003	Veer Arya	19-12-1960	9943546883	Dindigul	624001	M
CUST0004	Sai Banerjee	18-06-1973	6683789558	Madurai	625003	F
CUST0005	Gaurangi Nagi	04-05-1980	6995878561	Madurai	625004	M
CUST0006	Laban Sodhi	15-06-1985	7288572524	Hyderabac	500002	F
CUST0007	Jeremiah Khanna	03-04-1963	7528751681	Bangalore	560002	F
CUST0008	Ganga Batra	30-09-1960	6527492376	Erode	638002	M
CUST0009	Joshua Sethi	17-12-1970	7651574833	Coimbator	641002	F

Fig 4.1 Sample Dataset

Fig 4.1 represents a sample customer dataset created for the project, containing essential details about different customers in a structured format. Each record includes a unique customer ID, customer name, date of birth, contact number, city of residence, corresponding zip code, and gender.

4.2 FEATURE EXTRACTION

4.2.1 Customer table

Table 4.2.1 Customer Table

Field Name	Description
customer_id	Unique identifier for each customer
customer_name	Full name of the customer
customer_dob	Date of birth of the customer
customer_gender	Gender of the customer (M/F)
customer_number	Mobile number of the customer
customer_location	City or town where the customer resides
customer_zipcode	Postal or ZIP code of the customer's location

	A	B	C	D	E	F	G
1	customer_id	customer_name	customer_dob	customer_number	customer_	customer_zipcode	customer_gender
2	CUST0001	Keya Nadkarni	03-10-1971	7028613734	Mumbai	400001	F
3	CUST0002	Gaurang Vaidya	16-03-1962	8982850698	Salem	636003	M
4	CUST0003	Veer Arya	19-12-1960	9943546883	Dindigul	624001	M
5	CUST0004	Sai Banerjee	18-06-1973	6683789558	Madurai	625003	F
6	CUST0005	Gaurangi Nagi	04-05-1980	6995878561	Madurai	625004	M
7	CUST0006	Laban Sodhi	15-06-1985	7288572524	Hyderabac	500002	F
8	CUST0007	Jeremiah Khanna	03-04-1963	7528751681	Bangalore	560002	F
9	CUST0008	Ganga Batra	30-09-1960	6527492376	Erode	638002	M
10	CUST0009	Joshua Sethi	17-12-1970	7651574833	Coimbator	641002	F
11	CUST0010	Oviya Bhatt	07-09-1960	6854241017	Thanjavur	613002	M
12	CUST0011	Abhiram Narang	11-07-1993	6695229350	Hyderabac	500002	M
13	CUST0012	Xiti Chand	08-12-1956	9911244175	Mumbai	400001	F
14	CUST0013	Inaya Kumer	24-01-2000	8304378225	Erode	638003	F
15	CUST0014	Frederick Mutti	01-12-1961	6562309352	Salem	636002	M
16	CUST0015	Amruta Gole	15-10-2002	7559361018	Salem	636003	F
17	CUST0016	Omya Balakrishnan	20-06-1958	7685802933	Tirunelveli	627001	M
18	CUST0017	Chanakya Buch	18-03-1991	7526526038	Mumbai	400001	M
19	CUST0018	Omisha Vora	12-06-1990	6555454435	Thanjavur	613002	M
20	CUST0019	Sneha Dar	16-10-1976	6792969244	Tirunelveli	627002	M

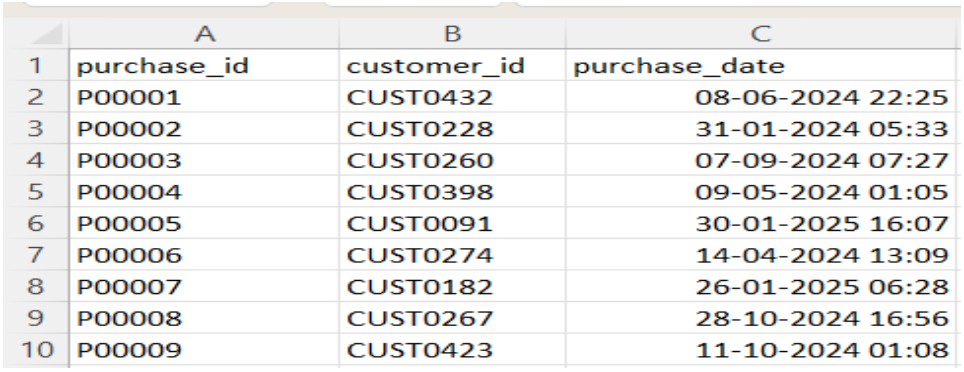
Fig 4.2.1 Customer Table

Fig 4.2.1 represents the customer demographic and contact details, including unique IDs, names, dates of birth, phone numbers, locations with ZIP codes, and gender. It provides a structured view of customer profiles, useful for segmentation, regional analysis, and supporting business applications like marketing, sales analysis, and customer relationship management.

4.2.2 Purchase table

Table 4.2.2 Purchase Table

Field Name	Description
purchase_id	Unique identifier for each purchase transaction
customer_id	Foreign key linking to the customers table
purchase_date	Date and time when the purchase was made



	A	B	C
1	purchase_id	customer_id	purchase_date
2	P00001	CUST0432	08-06-2024 22:25
3	P00002	CUST0228	31-01-2024 05:33
4	P00003	CUST0260	07-09-2024 07:27
5	P00004	CUST0398	09-05-2024 01:05
6	P00005	CUST0091	30-01-2025 16:07
7	P00006	CUST0274	14-04-2024 13:09
8	P00007	CUST0182	26-01-2025 06:28
9	P00008	CUST0267	28-10-2024 16:56
10	P00009	CUST0423	11-10-2024 01:08

Fig 4.2.2 Purchase Table

Fig 4.2.2 represents the purchase transactions, with each row representing a unique purchase. It includes a purchase ID (unique identifier for each transaction), the customer ID (linking the purchase to a specific customer), and the purchase date and time. This data can be used to track customer buying behaviour, analyse purchase frequency, identify peak transaction times, and study customer retention or seasonal trends.

4.2.3 Saree table

Table 4.2.3 Saree Table

Field Name	Description
saree_id	Unique identifier for each saree item
saree_name	Name or description of the saree
saree_type	Fabric or category of the saree (e.g., Silk, Cotton)
saree_base_price	Base price of the saree before discount
stock_quantity	Quantity of the saree available in stock

	saree_id	saree_name	saree_type	saree_base_price	stock_quantity
2	SR001	Rayon Striped Pattern	Rayon	1688	79
3	SR002	Luxurious Silk Kanjeevara	Silk	9031	59
4	SR003	Elegant Banarasi Zari	Banarasi	4494	22
5	SR004	Linen Check Design	Linen	1416	39
6	SR005	Gorgeous Velvet Look	Velvet	7740	79
7	SR006	Elegant Satin Finish	Satin	1864	28
8	SR007	Traditional Kanjeevaram '	Kanjeevaram	11292	28
9	SR008	Soft Linen Texture	Linen	1804	95
10	SR009	Cotton Block Print	Cotton	1393	43

Fig 4.2.3 Saree Table

Fig 4.2.3 represents the details about saree products available in stock. Each record includes a saree ID (unique identifier), saree name (descriptive name of the design), saree type (such as Rayon, Silk, Banarasi, Linen, etc.), base price (the listed price of the saree), and stock quantity (the number of units available). This dataset can be used for inventory management, sales tracking, and price analysis to monitor availability, manage stock levels, and understand the distribution of saree types and their pricing.

4.2.4 Purchase items table

Table 4.2.4 Purchaseditems Table

Field Name	Description
purchase_id	Foreign key referencing the purchases table
saree_id	Foreign key referencing the sarees table
quantity	Number of units purchased
saree_selling_price	Selling price per unit at the time of purchase
discount	Discount applied to the item (e.g., 0.10 for 10%)
final_price	Total price after discount
payment_method	Mode of payment used (e.g., Cash, UPI, Card)

A	B	C	D	E	F	G
purchase_id	saree_id	saree_selling_price	quantity	discount	final_price	payment_method
P00001	SR028	8723.95	1	0.11	7764.32	Cash
P00002	SR015	4203.5	4	0.06	15805.16	Debit Card
P00002	SR014	3750.32	5	0.1	16876.44	UPI
P00003	SR019	2571.21	1	0.13	2236.95	Net Banking
P00003	SR025	2822.21	5	0.13	12276.61	Credit Card
P00003	SR019	2379.61	1	0.15	2022.67	UPI
P00003	SR022	1402.53	1	0.1	1262.28	Debit Card
P00004	SR026	2422.81	3	0.03	7050.38	Net Banking
P00004	SR015	4237.43	4	0.15	14407.26	UPI
P00005	SR033	3121.5	1	0.09	2840.56	UPI
P00005	SR031	3204.21	5	0.08	14739.37	UPI
P00005	SR029	2426.2	3	0.06	6841.88	UPI
P00006	SR007	14408.2	4	0.12	50716.86	Cash
P00006	SR031	2993.83	3	0.05	8532.42	Cash

Fig 4.2.4 Purchase item Table

Fig 4.2.4 represents sales transaction details for saree purchases. Each entry is linked to a purchase ID and a saree ID, showing which product was bought. It includes the selling price per saree, quantity purchased, the discount applied, and the resulting final price after discount calculation. Additionally, it records the payment method (such as Cash, Debit Card, UPI, or Net Banking). This dataset is useful for analyzing sales performance, customer purchasing behavior, discount effectiveness, and payment preferences, as well as for generating business insights like revenue tracking and product demand trends.

4.3 RSAF DATASET

- The data was structured into four primary tables: customers, purchases, purchased items, and sarees using MYSQL.
- Each table was created with appropriate attributes and keys to ensure data integrity and facilitate easy querying.
- This structured dataset enables accurate sales trend analysis, customer behaviour insights, and efficient training of the forecasting model.

4.4 PROCESS FLOW

Phase I

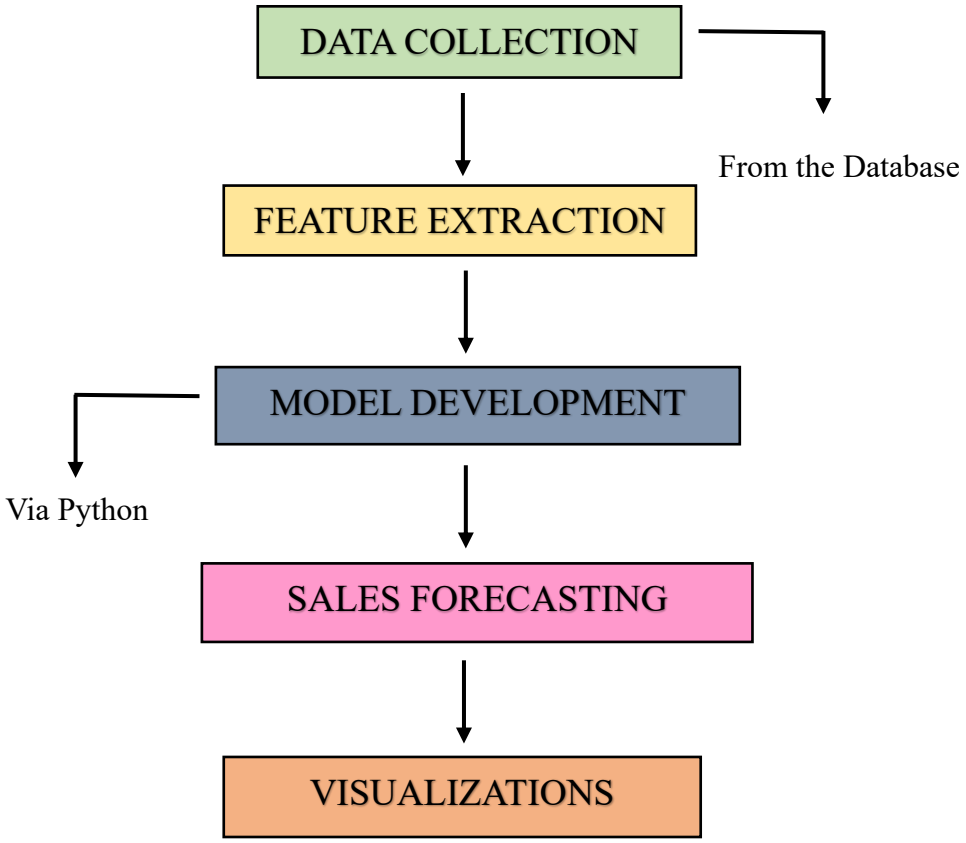


Fig 4.4.1

Fig 4.4.1 illustrates the process of building a sales forecasting system, beginning with data collection from the database, which provides the raw information required for analysis. This is followed by feature extraction, where important attributes such as product details, pricing, and sales history are identified for use in modeling. The next step is model development, carried out using Python, where machine learning techniques are applied to uncover sales patterns. Based on these models, sales forecasting is performed to predict future demand and trends. Finally, the outcomes are presented through visualizations, enabling clear interpretation of results and supporting informed business decisions.

Phase II

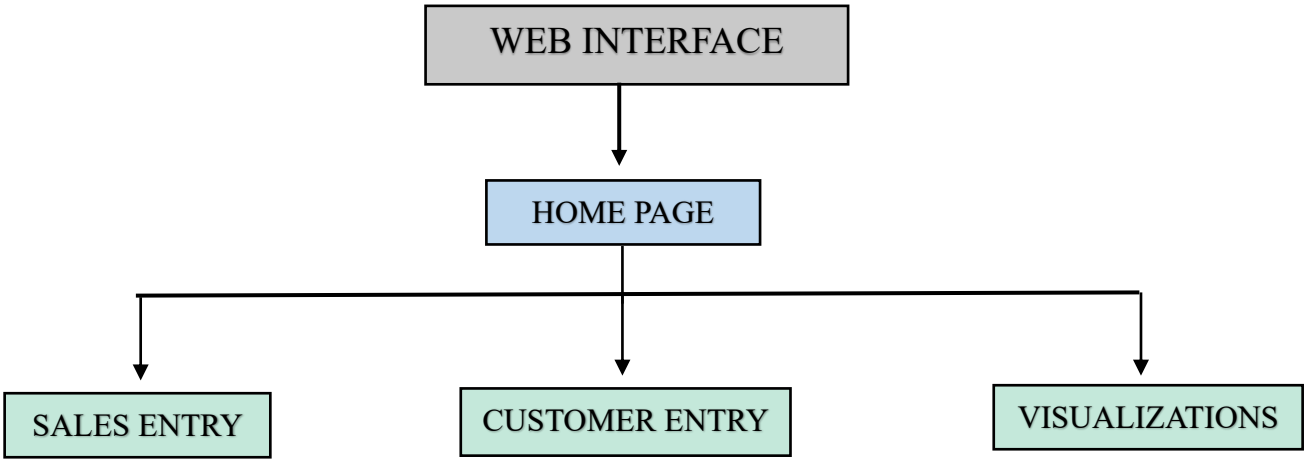


Fig 4.4.2

Fig 4.4.2 represents the web interface design of the system. It begins with the web interface, which serves as the platform for user interaction. From there, users are directed to the home page, which acts as the central hub of navigation. The home page provides three main functionalities: Sales Entry, where sales transactions can be recorded; Customer Entry, where customer details are added or updated; and Visualizations, which allow users to view analytical insights and graphical reports. This structure ensures that the system is user-friendly, organized, and provides easy access to essential operations.

5.MODEL SELECTION - RSAF MODEL

5.1 RSAF MODEL DEVELOPMENT

Linear Regression is a supervised machine learning algorithm used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the observed data. It is one of the simplest yet effective techniques for forecasting and predictive analytics.

In Retail analysis, Linear Regression was used to forecast future sales for a saree retail store based on historical monthly sales data. The data was first collected and aggregated from the MySQL database, where each transaction's total sales were grouped by month. Using the scikit-learn library in Python, a Linear Regression model was trained on this data to learn the trend in sales over time. Once trained, the model was used to predict sales for the next three months, helping the retailer plan inventory and promotions more effectively. These predictions were then visualized using Plotly and integrated into a web-based dashboard built with Flask. The simplicity, efficiency, and interpretability of Linear Regression made it a suitable choice for this project, especially given the linear nature of the sales trend and the limited size of the dataset.

PSEUDO ALGORTIHM

START

Connect to MySQL database

Fetch sales data (purchase date and final price)

Convert dates to months and group sales by month

Prepare data: X = month index, y = total sales

Train Linear Regression model on (X, y)

Predict sales for next 3 months using the model

Visualize actual and predicted sales

Display results on the web dashboard (Flask/Streamlit)

END

6.RSAF- WEB INTERFACE

6.1 HOME PAGE

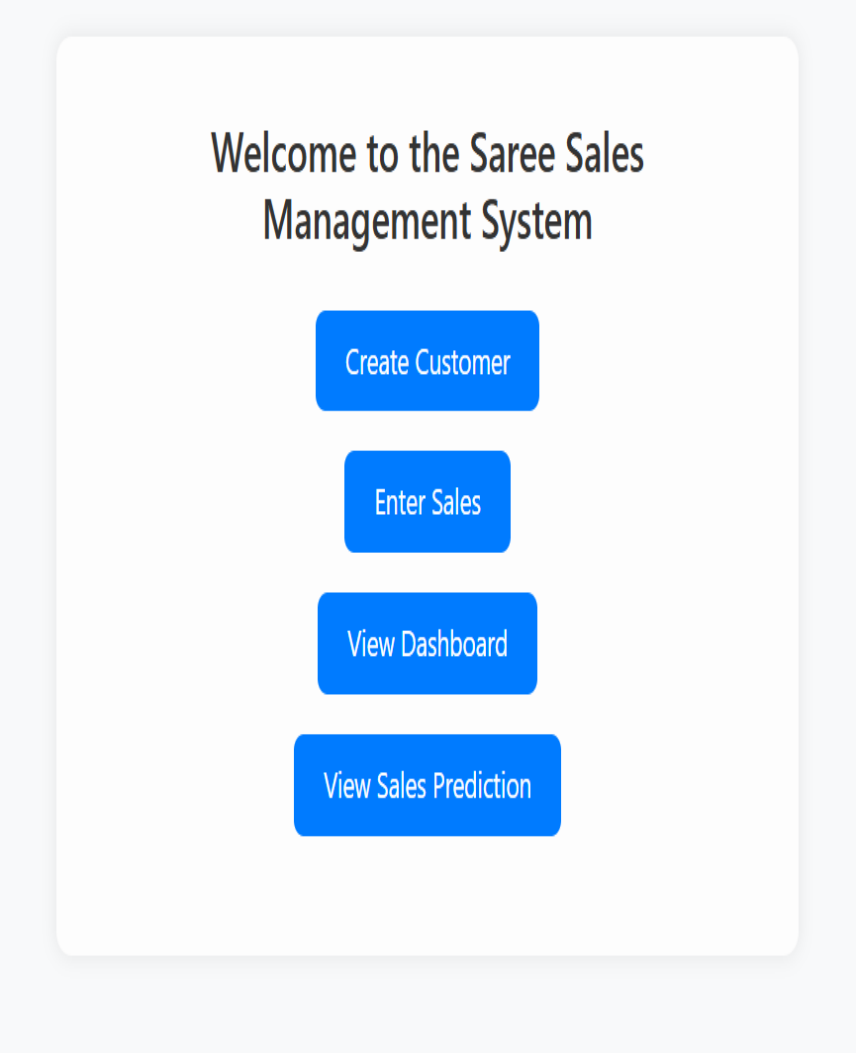


Fig 6.1 Home Page

Fig 6.1 represents the homepage of the web application serves as the main screen that provides users with easy navigation to various functionalities of the system. It typically includes links or buttons that guide users to different sections such as Customer Entry, Sales Entry, Sales Dashboard, and Prediction Results.

6.2 CUSTOMER ENTRY

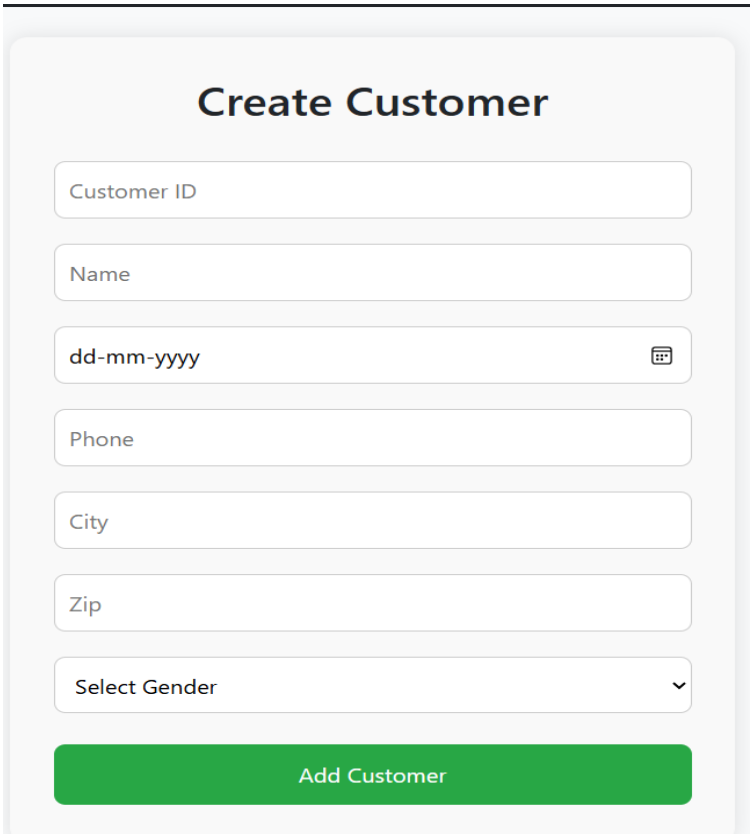


Fig 6.2 Customer Entry Page

Fig 6.2 represents the customer entry page, the user is presented with a structured form containing fields such as Customer Name, Date of Birth, Gender, Mobile Number, City, and ZIP Code. Each input field includes basic validation rules to ensure the accuracy and completeness of the data, such as requiring the mobile number to follow a 10-digit format or restricting the date of birth to a invalid calendar range. Once the form is filled and submitted, the data is processed through a Flask route and inserted into the customers table in the MySQL database using the MySQL-connector-python library.

6.3 SALES ENTRY

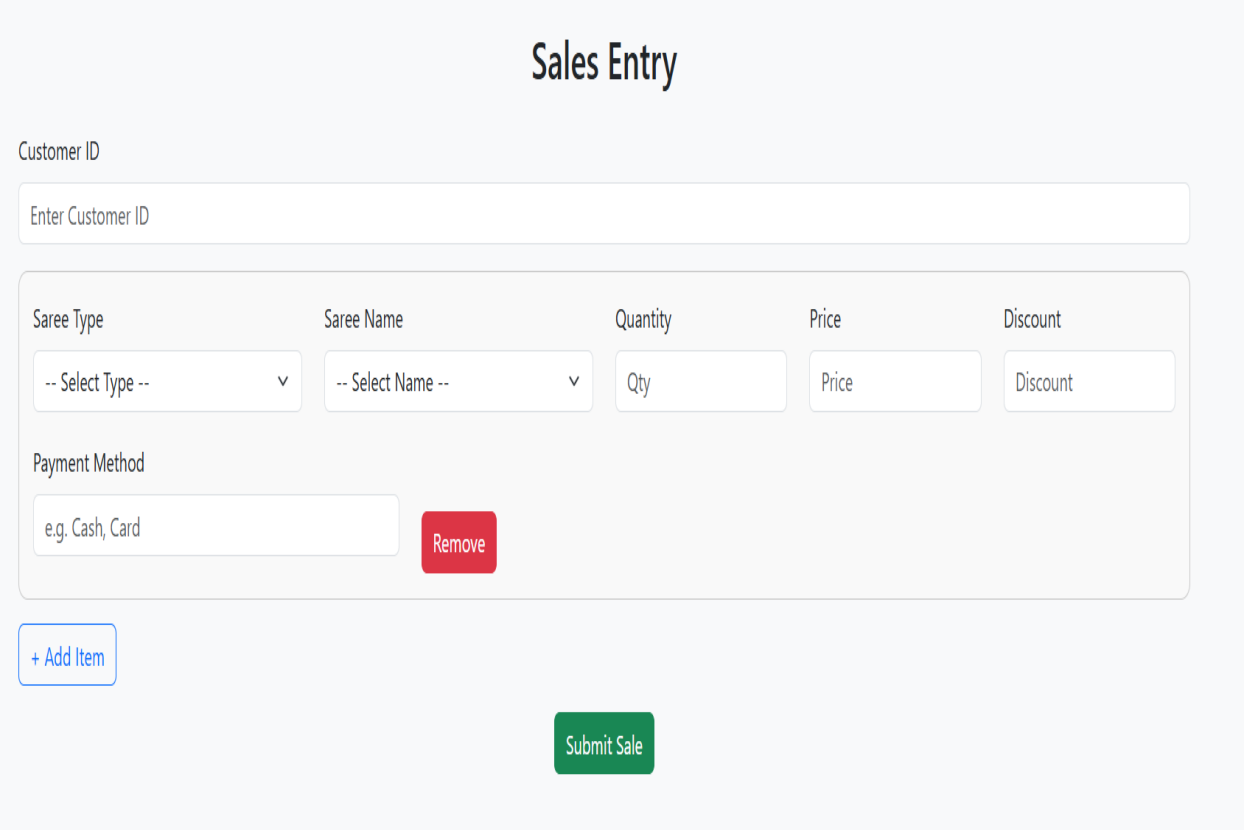


Fig 6.3 Sales Entry Page

Fig 6.3 represents the Sales Entry Page allows shop staff to record customer purchases efficiently into the system. This web form, developed using Flask and HTML, captures essential sales data such as the customer ID, purchase date, selected saree ID(s), quantity, selling price, discount (if any), and payment method. Upon form submission, the entered data is processed and stored into the purchases and purchase items tables in the MySQL database. The system also automatically calculates the final price after applying discounts and updates the stock quantity for each saree purchased. A confirmation message is displayed once the transaction is successfully recorded.

6.4 VISUALIZATIONS

Dashboard contains Total Revenue, Total Sales and Total number of Customers. It also contains the insights such as Monthly revenue trend, Top selling sarees, Gender-wise revenue, Payment mode revenue etc…

Monthly Revenue Trend

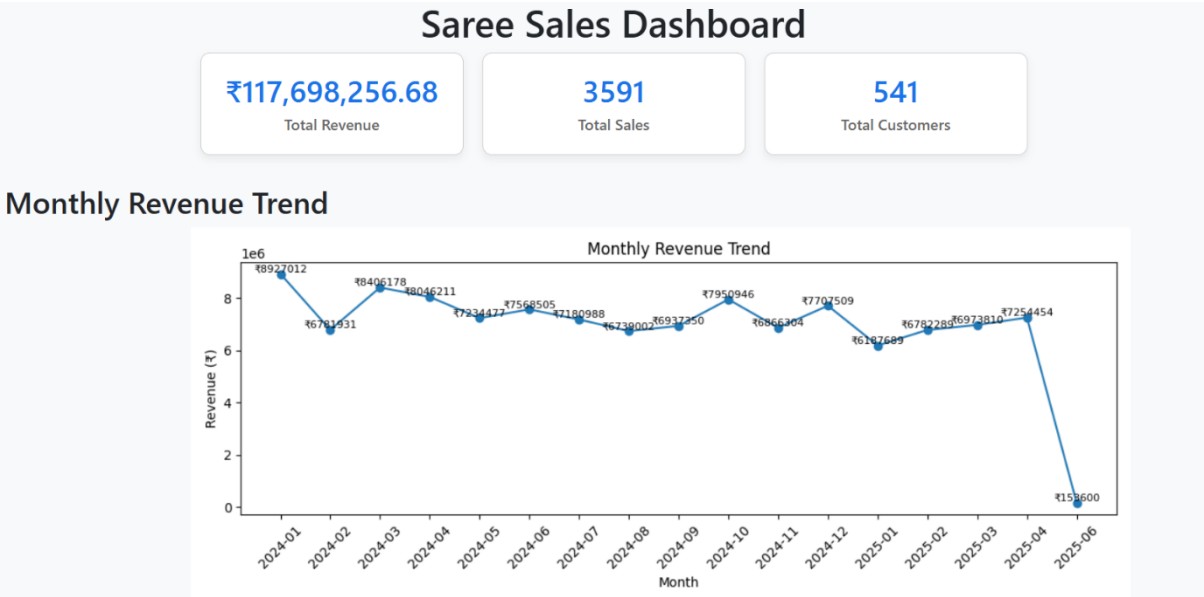


Fig 6.4.1 Monthly Revenue Trends

Fig 6.4.1 shows the monthly revenue for each month from Jan 2024 to Jun 2025. By analysing this graph retailers get to know high and low performing months and seasonal sales spikes. For example, Jan 2024 has high revenue while Jun 2025 has low revenue.

Gender-wise Revenue

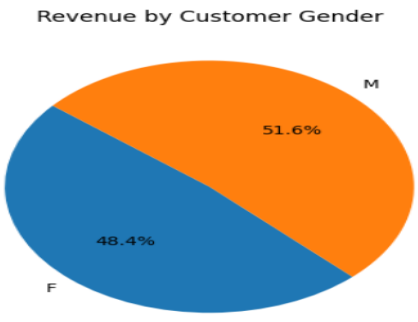


Fig 6.4.2 Gender-wise Revenue

Fig 6.4.2 shows the revenue based on the Customer gender. Retailers can visualize the gender-wise sales and target customers based on the gender.

Payment Method Revenue

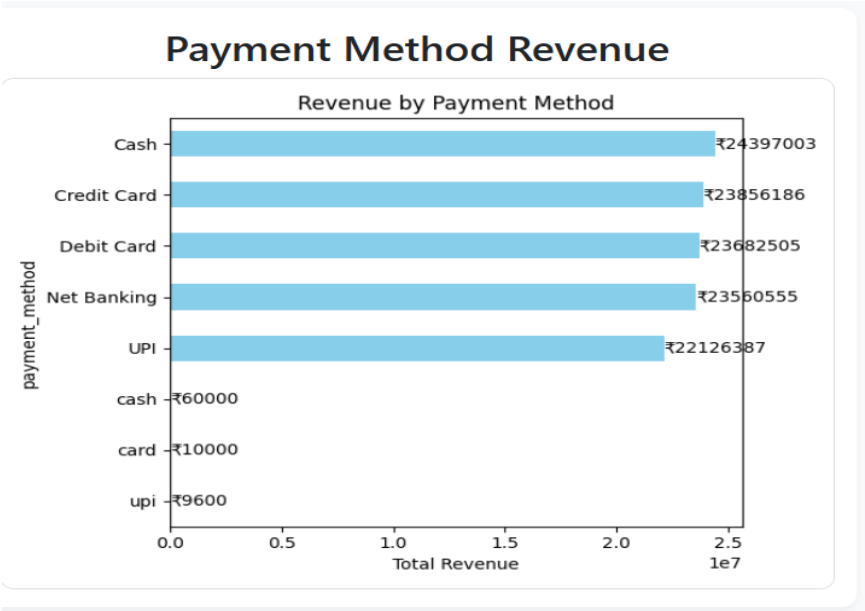


Fig 6.4.3 Payment Method Revenue

Fig 6.4.3 shows the revenue according to the payment method such as cash, card, UPI etc…Here high revenue achieved through cash payment.

Saree type Revenue

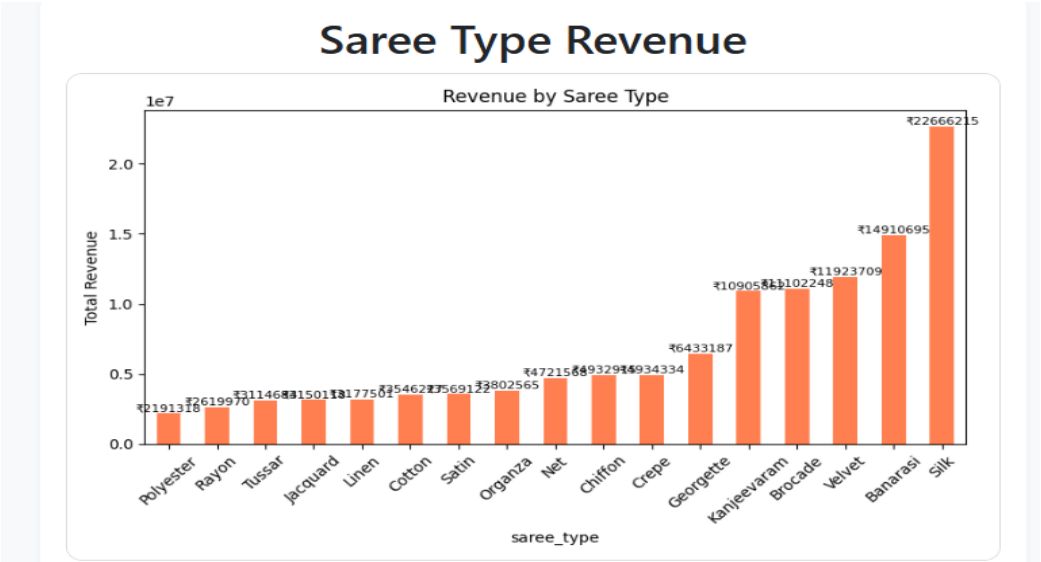


Fig 6.4.4 Saree Type Revenue

Fig 6.4.4 shows the revenue for each saree type. By analysing this graph silk saree has high demand.

Top 10 best-selling sarees

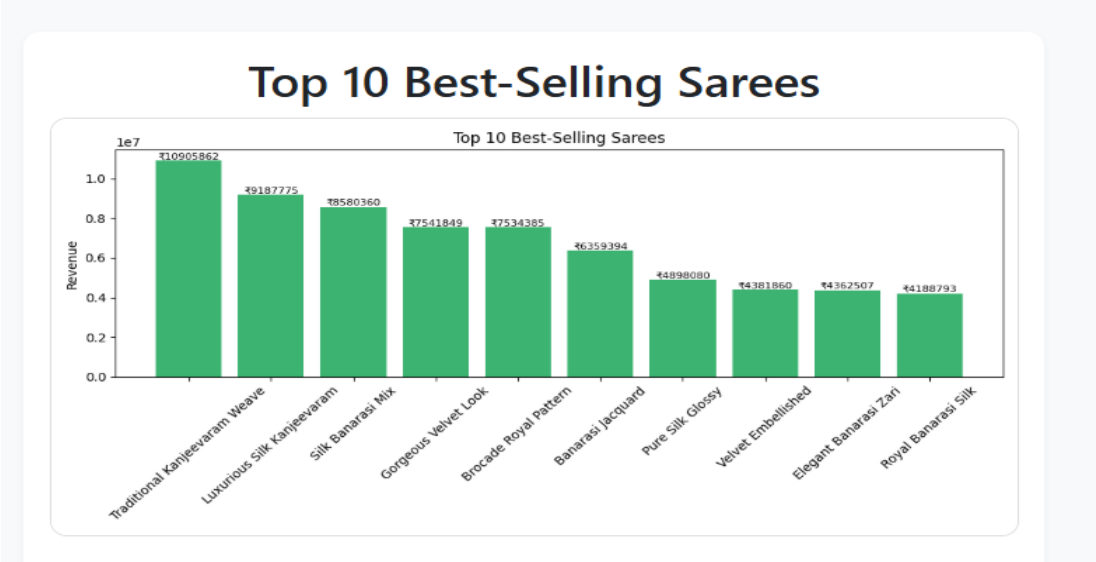


Fig 6.4.5 Top 10 best selling sarees

Fig 6.4.5 shows the top 10 best-selling sarees based on the total revenue. Here traditional kanjeevaram silk is the topmost best-selling saree.

Top saree sold

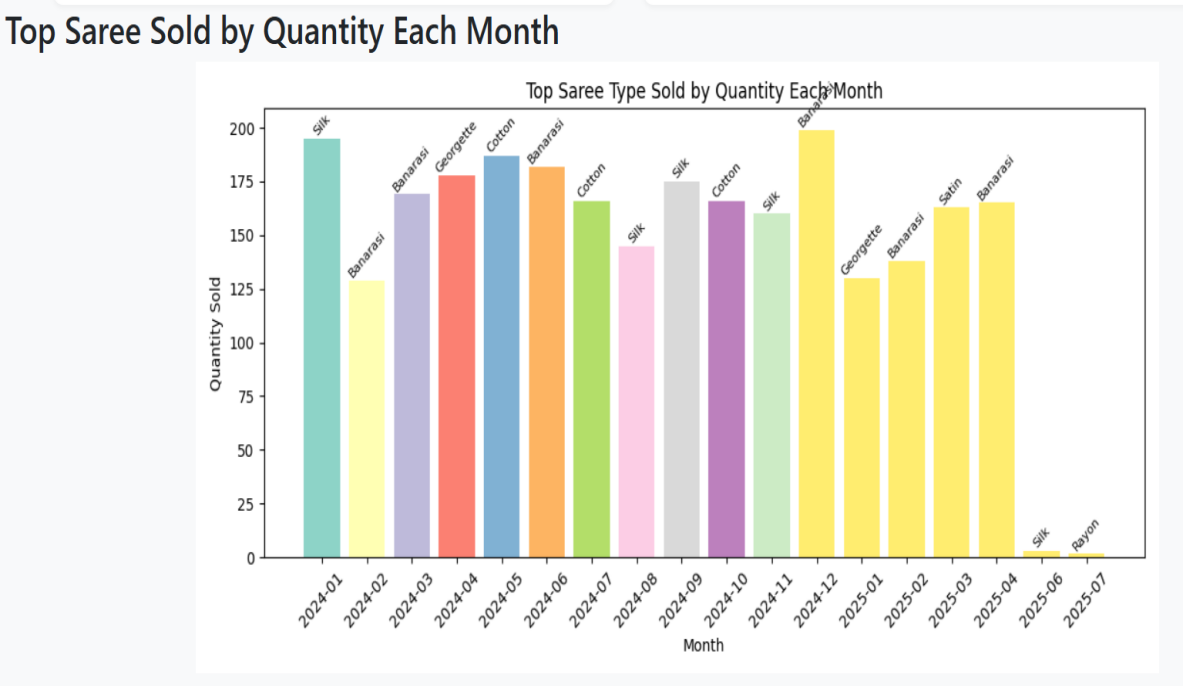
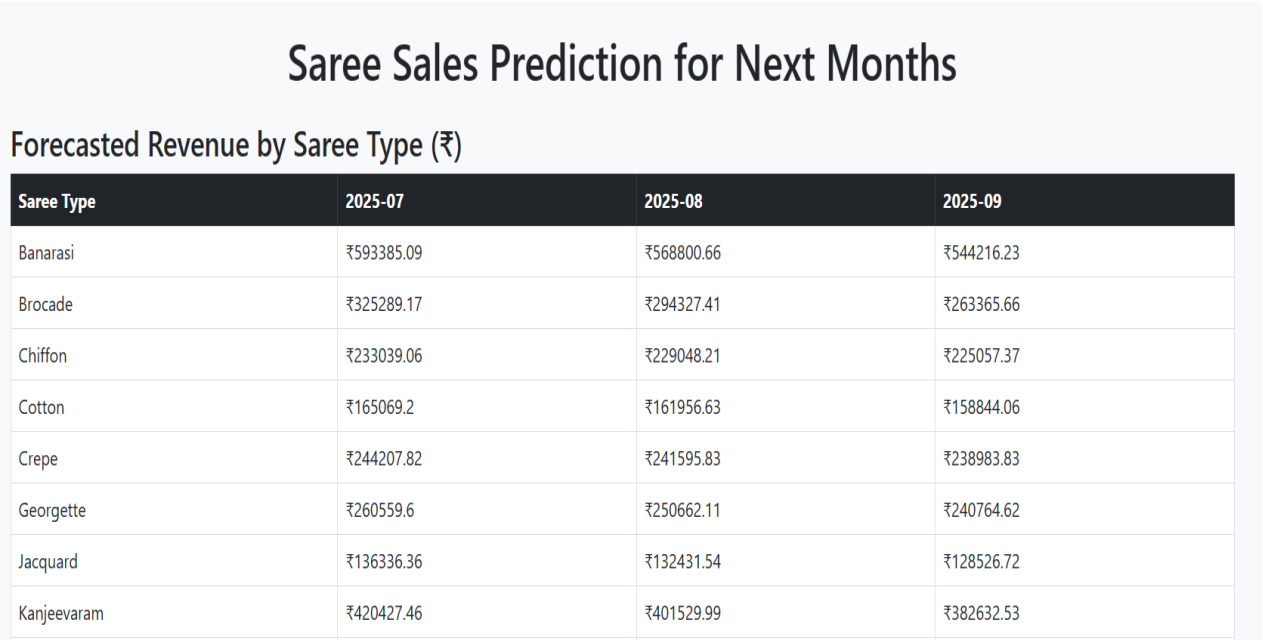


Fig 6.4.6 Top Saree Sold

Fig 6.4.6 shows the best-selling sarees based on the quantity sold for each month.

6.5 SALES PREDICTION



Saree Sales Prediction for Next Months

Forecasted Revenue by Saree Type (₹)

Saree Type	2025-07	2025-08	2025-09
Banarasi	₹593385.09	₹568800.66	₹544216.23
Brocade	₹325289.17	₹294327.41	₹263365.66
Chiffon	₹233039.06	₹229048.21	₹225057.37
Cotton	₹165069.2	₹161956.63	₹158844.06
Crepe	₹244207.82	₹241595.83	₹238983.83
Georgette	₹260559.6	₹250662.11	₹240764.62
Jacquard	₹136336.36	₹132431.54	₹128526.72
Kanjeevaram	₹420427.46	₹401529.99	₹382632.53

Fig 6.5.1 Sales Prediction

Fig 6.5.1 the Sales Prediction Page is a key feature of the web application that displays the forecasted sales for the upcoming three months based on the historical data and the trained Linear Regression model. This page also helps the retailer plan inventory, set sales targets, and prepare for high-demand periods with greater confidence and data-backed insights.

Inventory management

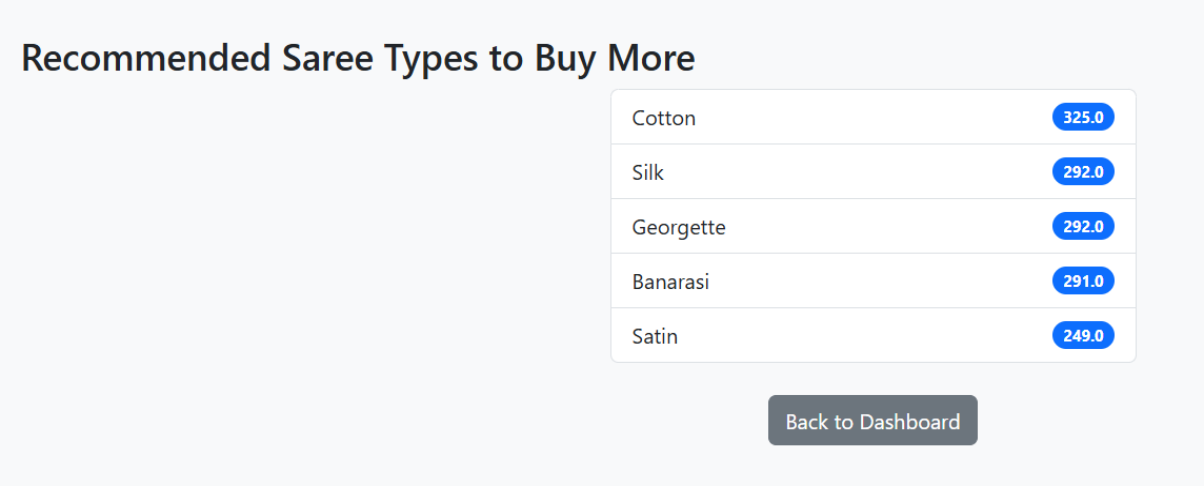


Fig 6.5.2 Inventory Managament

Fig 6.5.2 represents the saree types to buy more based on the previous months sales analysis

7.CONCLUSION

Retail sales analysis successfully demonstrates how traditional retail businesses can leverage modern data science techniques to improve operational efficiency and decision-making. By collecting and organizing historical sales data into a structured database and applying a supervised machine learning algorithm, Linear Regression, the system provides a practical and interpretable solution for forecasting future sales. It predicts upcoming sales trends for the next three months, enabling saree retailers to manage inventory more effectively, plan procurement, and prepare for peak demand periods with greater confidence.

The solution also integrates a dynamic Flask-based web application that supports real-time customer and sales entry, while displaying visual dashboards for insights such as top-selling sarees, revenue patterns, and stock levels. This end-to-end approach ensures that even small or non-technical business users can easily interact with and benefit from machine learning technology in a simple and accessible way.

FUTURE ENHANCEMENT

While the use of Linear Regression offers a good starting point for forecasting, future versions of the system can incorporate more advanced models such as ARIMA, LSTM, or Prophet to capture seasonality, trends, and irregular sales behaviour more accurately. The system can also be extended with real-time data integration, automated alerts, customer segmentation, and recommendation systems to further boost business intelligence. Overall, the project stands as a strong example of how data analytics and machine learning can empower even traditional, small-scale retail businesses to make smarter, data-driven decisions.

REFERENCES

Bibilography

1. Chopra, R., & Ghosh, A. Retail Sales Forecasting Using Machine Learning Techniques. International Journal of Computer Applications, (2020), 176(25), 15–21.
2. Suresh, K., & Ramya, M. Predictive Analysis in Retail Industry Using Python. International Journal of Engineering Research & Technology, (2021), 10(6), 450–456.
3. Jain, P., & Singh, R. Retail Analytics using Python and MySQL. International Journal of Advanced Computer Science, 11(3), (2020), 120–128.
4. Mishra, S., & Rajan, V. Inventory Optimization Using Data Science in Retail. Journal of Retail & Consumer Analytics, 5(2), (2021), 89–97.
5. Smith, J., & Kumar, P. Lightweight Forecasting Models for Small-Scale Retailers. International Journal of Data Science and Analytics, (2022), 8(4), 233–240.

Website References

https://towardsdatascience.com/sales-forecasting-using-machine-learning-9391f3c9803d

https://www.analyticsvidhya.com/blog/2021/06/how-to-forecast-retail-demand-with-machine-learning/

https://www.mckinsey.com/industries/retail/our-insights/how-data-and-analytics-are-transforming-retail

APPENDIX

#import necessary libraries

```
from flask import Flask, render_template, request, redirect, url_for, flash
import mysql.connector
import pandas as pd
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import io
import base64
import uuid
from mysql.connector.errors import IntegrityError
from sklearn.linear_model import LinearRegression
import numpy as np
```

 # Flask app initialization

```
app = Flask(__name__)
app.secret_key = 'your_secret_key'
```

 # Function to establish MySQL database connection

```
def get_db_connection():
    return mysql.connector.connect(
        host='localhost',
        user='root',
        password='root',
        database='saree_sales'
    )
```

 # Convert matplotlib plots to base64 so they can be displayed in HTML

```
def generate_plot_url(fig):
    img = io.BytesIO()
    fig.savefig(img, format='png')
```

```
    img.seek(0)
    return base64.b64encode(img.getvalue()).decode()


# Homepage route
@app.route('/')
def home():
    return render_template('home.html')


#customer entry
@app.route('/create-customer', methods=['GET', 'POST'])
def create_customer():
    if request.method == 'POST':
        data = request.form
        conn = get_db_connection()
        cursor = conn.cursor()
        try:
            cursor.execute('''
                INSERT INTO customers (customer_id, customer_name, customer_dob,
customer_number, customer_city, customer_zipcode, customer_gender)
                VALUES (%s, %s, %s, %s, %s, %s, %s)
            ''', (
                data['customer_id'], data['customer_name'], data['customer_dob'],
                data['customer_number'], data['customer_city'],
                data['customer_zipcode'], data['customer_gender']
            ))
            conn.commit()
            flash("Customer created successfully", "success")
            return redirect(url_for('home'))

        except IntegrityError as e:
            if "Duplicate entry" in str(e):
                flash(f" ❗ Customer ID '{data['customer_id']}' already exists. Please use a unique
ID.", "danger")
```

```
            else:
                flash(" ❗ An error occurred while creating the customer.", "danger")
            return redirect(url_for('create_customer'))

        finally:
            cursor.close()
            conn.close()

    return render_template('create_customer.html')
```

#sales entry

```
@app.route('/sales-entry', methods=['GET', 'POST'])
def sales_entry():
    conn = get_db_connection()
    cursor = conn.cursor()
    cursor.execute("SELECT DISTINCT saree_type FROM sarees")
    saree_types = [r[0] for r in cursor.fetchall()]
    if request.method == 'POST':
        data = request.form
        customer_id = data.get('customer_id')
        cursor.execute("SELECT 1 FROM customers WHERE customer_id = %s",
(customer_id,))
        if not cursor.fetchone():
            flash("Customer ID not found. Please create the customer first.", "danger")
            return redirect(url_for('create_customer'))
        purchase_id = f"P-{uuid.uuid4().hex[:8]}"
        cursor.execute("INSERT INTO purchases (purchase_id, customer_id) VALUES (%s,
%s)", (purchase_id, customer_id))
        i = 0
        while f'saree_type_{i}' in data:
            saree_type = data[f'saree_type_{i}']
            saree_name = data[f'saree_name_{i}']
            quantity = int(data[f'quantity_{i}'])
```

```
            price = float(data[f'price_{i}'])
            discount = float(data[f'discount_{i}'])
            payment = data[f'payment_{i}']
            cursor.execute("SELECT saree_id, stock_quantity FROM sarees WHERE
saree_name=%s AND saree_type=%s", (saree_name, saree_type))
            saree = cursor.fetchone()
            if not saree:
                flash(f"Saree not found: {saree_name} under {saree_type}", "warning")
                i += 1
                continue
            saree_id, stock = saree
            if stock < quantity:
                flash(f"Insufficient stock for {saree_name}. Available: {stock}", "warning")
                i += 1
                continue
            final_price = (price * quantity) - discount
            cursor.execute(
                "INSERT INTO purchase_items (purchase_id, saree_id, quantity,
saree_selling_price, discount, final_price, payment_method) "
                "VALUES (%s, %s, %s, %s, %s, %s, %s)",
                (purchase_id, saree_id, quantity, price, discount, final_price, payment)
            )
            cursor.execute("UPDATE sarees SET stock_quantity = stock_quantity - %s WHERE
saree_id = %s", (quantity, saree_id))
            i += 1
        conn.commit()
        flash(f'Sale recorded successfully with Purchase ID: {purchase_id}", "success")
        cursor.close()
        conn.close()
        return redirect(url_for('home'))
    cursor.close()
    conn.close()
    return render_template('sales_entry.html', saree_types=saree_types)
```

```
@app.route('/get-saree-names/<saree_type>')
def get_saree_names(saree_type):
    conn = get_db_connection()
    cursor = conn.cursor()
    cursor.execute("SELECT saree_name FROM sarees WHERE saree_type = %s",
(saree_type,))
    names = [r[0] for r in cursor.fetchall()]
    cursor.close()
    conn.close()
    return {"saree_names": names}
```

#dashboard

```
@app.route('/dashboard')
def dashboard():
    conn = get_db_connection()
```

 # Query sales data with join

```
    purchases_query = """
        SELECT
            pi.purchase_id,
            p.purchase_date,
            p.customer_id,
            pi.saree_id,
            pi.quantity,
            pi.saree_selling_price,
            pi.discount,
            pi.final_price,
            pi.payment_method
        FROM purchase_items pi
        JOIN purchases p ON pi.purchase_id = p.purchase_id
    """
    df = pd.read_sql(purchases_query, conn)
    customers_df = pd.read_sql("SELECT * FROM customers", conn)
    sarees_df = pd.read_sql("SELECT * FROM sarees", conn)
```

```
    conn.close()
# Preprocess dates and calculate revenue
    df['purchase_date'] = pd.to_datetime(df['purchase_date'])
    df['month'] = df['purchase_date'].dt.to_period('M').astype(str)
    df['total_revenue'] = df['quantity'] * df['saree_selling_price'] - df['discount']
# KPI summary
    total_summary = {
        "total_revenue": round(df["total_revenue"].sum(), 2),
        "total_sales": df["purchase_id"].nunique(),
        "total_customers": df["customer_id"].nunique()
    }
# monthly revenue trend
    monthly_trend = df.groupby('month')['total_revenue'].sum().reset_index()
    fig1, ax1 = plt.subplots(figsize=(10, 4))
    ax1.plot(monthly_trend['month'], monthly_trend['total_revenue'], marker='o')
    for i, row in monthly_trend.iterrows():
        ax1.text(row['month'], row['total_revenue'], f"₹{row['total_revenue']:.0f}", ha='center',
va='bottom', fontsize=8)
    ax1.set_title('Monthly Revenue Trend')
    ax1.set_xlabel('Month')
    ax1.set_ylabel('Revenue (₹)')
    ax1.tick_params(axis='x', rotation=45)
    fig1.tight_layout()
    monthly_chart = generate_plot_url(fig1)
#Revenue by customer gender (pie chart)
    gender_df = df.merge(customers_df, on='customer_id')
    gender_summary = gender_df.groupby('customer_gender')['total_revenue'].sum()
    fig2, ax2 = plt.subplots()
    ax2.pie(gender_summary, labels=gender_summary.index, autopct='%1.1f%%',
startangle=140)
    ax2.set_title('Revenue by Customer Gender')
    gender_chart = generate_plot_url(fig2)
```

#Revenue by payment method

```
payment_summary = df.groupby('payment_method')['total_revenue'].sum().sort_values()
fig3, ax3 = plt.subplots()
payment_summary.plot(kind='barh', ax=ax3, color='skyblue')
for i, (label, value) in enumerate(payment_summary.items()):
    ax3.text(value, i, f"₹{value:.0f}", va='center')
ax3.set_title('Revenue by Payment Method')
ax3.set_xlabel('Total Revenue')
fig3.tight_layout()
payment_chart = generate_plot_url(fig3)
```

 # Revenue by saree type

```
saree_df = df.merge(sarees_df, on='saree_id')
type_summary = saree_df.groupby('saree_type')['total_revenue'].sum().sort_values()
fig4, ax4 = plt.subplots(figsize=(8, 5))
type_summary.plot(kind='bar', ax=ax4, color='coral')
for i, (label, value) in enumerate(type_summary.items()):
    ax4.text(i, value, f"₹{value:.0f}", ha='center', va='bottom', fontsize=8)
ax4.set_title('Revenue by Saree Type')
ax4.set_ylabel('Total Revenue')
ax4.tick_params(axis='x', rotation=45)
fig4.tight_layout()
type_chart = generate_plot_url(fig4)
```

 # Top 10 best-selling sarees

```
top_sarees = df.groupby('saree_id')['total_revenue'].sum().nlargest(10).reset_index()
top_sarees = top_sarees.merge(sarees_df, on='saree_id')
fig5, ax5 = plt.subplots(figsize=(10, 5))
ax5.bar(top_sarees['saree_name'], top_sarees['total_revenue'], color='mediumseagreen')
for i, v in enumerate(top_sarees['total_revenue']):
    ax5.text(i, v, f"₹{v:.0f}", ha='center', va='bottom', fontsize=8)
ax5.set_title('Top 10 Best-Selling Sarees')
ax5.set_ylabel('Revenue')
ax5.tick_params(axis='x', rotation=45)
fig5.tight_layout()
```

```
    top_saree_chart = generate_plot_url(fig5)
# Top saree type sold by quantity per month
    monthly_top_df = saree_df.copy()
    monthly_top_df['month'] =
pd.to_datetime(monthly_top_df['purchase_date']).dt.to_period('M').astype(str)

    monthly_quantity = monthly_top_df.groupby(['month',
'saree_type'])['quantity'].sum().reset_index()

    top_per_month = monthly_quantity.sort_values(['month', 'quantity'], ascending=[True,
False]) \
        .drop_duplicates('month')

    fig6, ax6 = plt.subplots(figsize=(10, 5))
    colors = plt.cm.Set3(range(len(top_per_month)))  # distinct color palette
    ax6.bar(top_per_month['month'], top_per_month['quantity'], color=colors)
    for i, row in top_per_month.iterrows():
        ax6.text(row['month'], row['quantity'], row['saree_type'], ha='center', va='bottom',
fontsize=8, rotation=45)

    ax6.set_title('Top Saree Type Sold by Quantity Each Month')
    ax6.set_ylabel('Quantity Sold')
    ax6.set_xlabel('Month')
    ax6.tick_params(axis='x', rotation=45)
    fig6.tight_layout()
    monthly_top_saree_chart = generate_plot_url(fig6)
# Render all charts on dashboard page
    return render_template(
        'dashboard_simple.html',
        total_summary=total_summary,
        monthly_chart=monthly_chart,
        gender_chart=gender_chart,
        payment_chart=payment_chart,
        type_chart=type_chart,
```

```
        top_saree_chart=top_saree_chart,
        monthly_top_saree_chart = monthly_top_saree_chart,

    )

@app.route('/sales-prediction')
def sales_prediction():
    conn = get_db_connection()

    # Query: Get monthly total quantity and price for each saree_type
    query = """
    SELECT
        DATE_FORMAT(p.purchase_date, '%Y-%m') as month,
        s.saree_type,
        SUM(pi.quantity) as total_quantity,
        SUM(pi.final_price) as total_revenue
    FROM purchase_items pi
    JOIN purchases p ON pi.purchase_id = p.purchase_id
    JOIN sarees s ON pi.saree_id = s.saree_id
    GROUP BY month, s.saree_type
    ORDER BY month, s.saree_type
    """
    df = pd.read_sql(query, conn)
    conn.close()

    # Create pivot tables
    quantity_pivot = df.pivot(index='month', columns='saree_type',
values='total_quantity').fillna(0)
    revenue_pivot = df.pivot(index='month', columns='saree_type',
values='total_revenue').fillna(0)

    forecast_horizon = 3  # next 3 months
    quantity_predictions = {}
    revenue_predictions = {}
```

```
    month_numbers = np.arange(len(quantity_pivot)).reshape(-1, 1)

    for saree_type in quantity_pivot.columns:
        # Quantity prediction
        y_qty = quantity_pivot[saree_type].values
        model_qty = LinearRegression().fit(month_numbers, y_qty)
        future_months = np.arange(len(quantity_pivot), len(quantity_pivot) +
forecast_horizon).reshape(-1, 1)
        forecast_qty = model_qty.predict(future_months)
        forecast_qty = [max(0, val) for val in forecast_qty]
        quantity_predictions[saree_type] = forecast_qty

        # Revenue prediction
        y_rev = revenue_pivot[saree_type].values
        model_rev = LinearRegression().fit(month_numbers, y_rev)
        forecast_rev = model_rev.predict(future_months)
        forecast_rev = [max(0, val) for val in forecast_rev]
        revenue_predictions[saree_type] = forecast_rev

    # Month labels for the forecast
    months = pd.period_range(start=quantity_pivot.index[-1], periods=forecast_horizon + 1,
freq='M')[1:]
    months = [str(m) for m in months]

    # Total forecasted quantity for recommendations
    total_forecast_quantity = {k: sum(v) for k, v in quantity_predictions.items()}
    recommended = sorted(total_forecast_quantity.items(), key=lambda x: x[1],
reverse=True)[:5]

    # Total forecasted revenue
    total_forecast_revenue = {k: sum(v) for k, v in revenue_predictions.items()}

    return render_template('sales_prediction.html',
```

```
                months=months,
                quantity_predictions=quantity_predictions,
                revenue_predictions=revenue_predictions,
                total_forecast_revenue=total_forecast_revenue,
                recommended=recommended)

if __name__ == "__main__":
    app.run(debug=True)
```