Retail Store Sales Trend Analysis using Time-Series Data

Problem Title:

"Retail Store Sales Trend Analysis using Time-Series Data"

Name: Khushi Agrawal

EMAIL: khushiagrawal230705@gmail.com

DATE: 2ND Sep 2025

Project Description:

This project provides an interactive web dashboard to analyze, visualize, and forecast retail store sales using Streamlit, Pandas, SQLAlchemy, and Statsmodels. The system supports data ingestion, cleaning, aggregation, anomaly detection, promo impact analysis, and forecasting with Holt-Winters Exponential Smoothing. It allows businesses to gain actionable insights from their sales data, enabling better decision-making.

Index:

1. Introduction:

Retail businesses generate massive amounts of sales data daily. Analyzing this data helps identify trends, anomalies, and future forecasts. This project builds a Retail Sales Analytics Dashboard that integrates data ingestion, visualization, anomaly detection, and forecasting into one interactive platform.

2. Problem Statement:

Raw sales datasets are often unstructured, lacking features for analysis. Businesses face challenges in detecting anomalies, measuring promo impacts, and forecasting future sales. The aim is to design a system that ingests data, processes it into meaningful insights, and presents results interactively.

3. Objectives:

- Ingest CSV sales datasets into a database.
- Perform data cleaning and feature engineering.
- Provide interactive charts for total sales, monthly averages, weekday trends, and anomalies.
- Forecast future sales using time-series models.
- Support promo analysis and top-performing stores identification.
- Allow users to download insights for reporting.

4. Dataset Description:

The dataset includes sales transactions from retail stores.

Columns include:
- Date → Sales transaction date
- StoreID → Store identifier
- Sales → Daily sales revenue

Engineered features include:
- Weekday, Month
- Sales Category (Low, Medium, High)
- Cumulative Sales
- Promo Day
- Anomaly (Z-score method)
- Footfall Estimation(customer estimate)

5.Use-Case Explanation:

•Data Ingestion: The user uploads a raw sales data CSV file. The system processes it and stores the cleaned version in a database.

• Data Cleaning: The system automatically handles missing values by filling them, removes duplicate rows, and standardizes column names.

• Feature Engineering: The system derives new, useful columns like Weekday, Month, and Cumulative Sales to enrich the dataset for analysis.

•Anomalies & KPIs: The system flags unusual sales days and displays key performance indicators such as total sales per store and average sales by month and weekday.

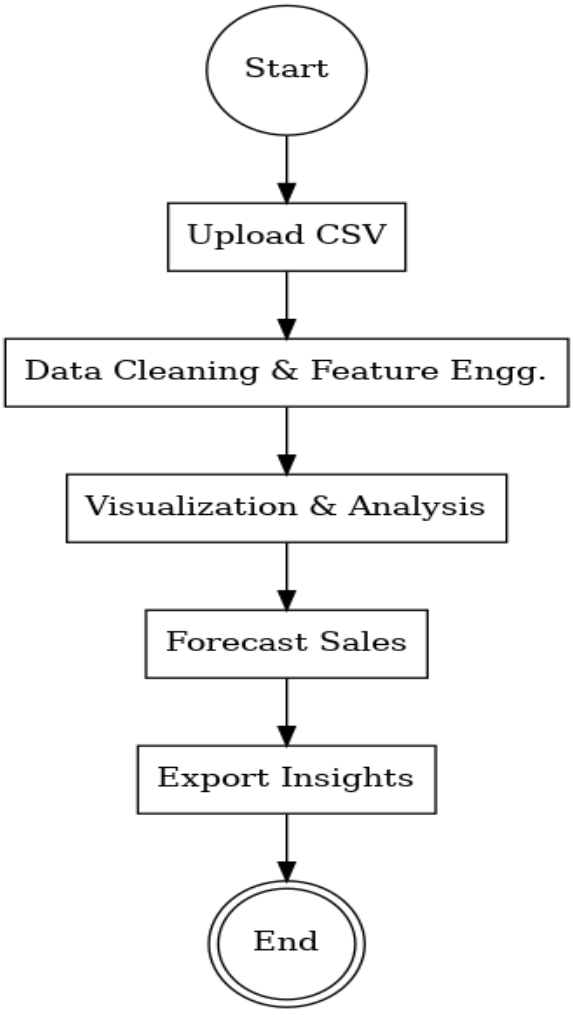
•Forecasting: The system generates a sales forecast for a specified number of future days.

•Reporting: The user can export various insights and reports into a downloadable Excel file.

•Security: The application is protected with a basic user authentication system for secure access to the dashboard.

6. UML Diagrams:

- **Activity diagram:**



7. Explanation of the code:

Main File: app_streamlit.py (Dashboard):

This is the central file that handles login, dataset ingestion, and analytics. Below are important sections with explanations:

a) Imports & Setup:

```
import os, pandas as pd, streamlit as st, altair as alt
from db import engine, Base, SessionLocal, Dataset
from models import Sale
from ingest import ingest_csv
from utils import agg_store_total, forecast_store, ...
```

- Imports Streamlit for UI, Altair for visualization, pandas for data handling.
- Imports database connection (db.py), sales model (models.py), CSV ingestion logic (ingest.py) and utility functions (utils.py).

b) Login/Signup System:

```
if "users" not in st.session_state:
    st.session_state.users = {}
```

- Implements a temporary in-memory login system using st.session_state.
- Users can Sign Up or Login.
- Ensures app pauses (st.stop()) until login is successful.
- Sidebar allows logout, resetting session state.

c) Dataset upload and ingestion:

```
uploaded = st.sidebar.file_uploader("Upload retail_sales.csv", type=["csv"])
if uploaded and st.sidebar.button("Ingest & Process"):
    new_id = ingest_csv(tmp_path, promo_dates=promos, dataset_name=ds_name)
```

- This block lets the user upload a **CSV file** from the sidebar.
- On clicking **"Ingest & Process"**, the file is cleaned and inserted into the database using ingest_csv().

d) Data Loading:

```python
def load_latest_dataset() -> pd.DataFrame:
    session = SessionLocal()
    d = session.query(Dataset).order_by(Dataset.uploaded_at.desc()).first()
    df = pd.read_sql(f"SELECT * FROM sales WHERE dataset_id = {d.id}", engine)
    return normalize_df_columns(df)
```

- Always loads the **most recent dataset** from DB.
- normalize_df_columns() standardizes column names (Date, StoreID, Sales).

e) Sidebar Controls:

```python
store_selector = st.sidebar.selectbox("Select store", options=store_list)
days_forecast = st.sidebar.number_input("Forecast days", min_value=7, max_value=90, value=14)
```

- Allows users to **select store** and **forecast period**.
- Adds export button for insights (Excel).

f) Total Sales by Store:

```python
store_totals = agg_store_total(df)
chart = alt.Chart(store_totals).mark_bar().encode(
    x="StoreID:O", y="Sales:Q", tooltip=["StoreID","Sales"])
st.altair_chart(chart, use_container_width=True)
```

- Displays aggregated store-wise sales.

g) Monthly and Weekday Averages:

```python
monthly = agg_monthly_avg(df)
weekday = agg_weekday_avg(df)
```

- Shows trends over time.

h) Anomaly & Promo Analysis:

```python
anomalies = df[df['Anomaly'] == 'Anomaly']
promo_mean = df.groupby('PromoDay')['Sales'].mean().reset_index()
```

- Uses Z-score to detect unusually high/low sales.
- Compares sales on Promo vs Non-Promo days.

i)Drilldown & Forecast:

```
df_plot_idx = df_plot.set_index("Date").asfreq("D").fillna(0.0)
df_plot_idx["7day"] = df_plot_idx["Sales"].rolling(7).mean()
model = ExponentialSmoothing(ser, trend="add")
fit = model.fit()
pred = fit.forecast(days_forecast)
```

- Shows daily sales trend + 7-day rolling average.
- Forecasts future sales using Holt-Winters Exponential Smoothing.

j)Analysis:

```
clean_df = df.copy().drop_duplicates()
ts_df["CumulativeSales"] = ts_df.groupby("StoreID")["Sales"].cumsum()
```

- Step-by-step Pandas operations for cleaning, filtering, derived columns.
- Includes Top 3 stores, Weekday sales, Cumulative sales.

k) Export Results:

```
st.download_button("⬇ Download Cleaned Dataset", data=csv1, file_name="cleaned_retail_sales.csv"
```

- Exports cleaned datasets & summaries.

Supporting Files:

utils.py:

```
df['Zscore'] = (df['Sales'] - df['Sales'].mean()) / df['Sales'].std()
df['Anomaly'] = np.where(df['Zscore'].abs() > 2, "Anomaly", "Normal")
```

Ingest.py:

```
df_feat = clean_and_feature(df_raw, promo_dates=promo_dates)
session.add(Sale(...))
```

- Reads CSV → calls clean_and_feature() (from utils).
- Inserts cleaned rows into sales table.

db.py

```
engine = create_engine("sqlite:///retail_sales.db")
SessionLocal = sessionmaker(bind=engine)
```

- Sets up SQLite database engine.
- Defines Dataset model → tracks uploaded files.

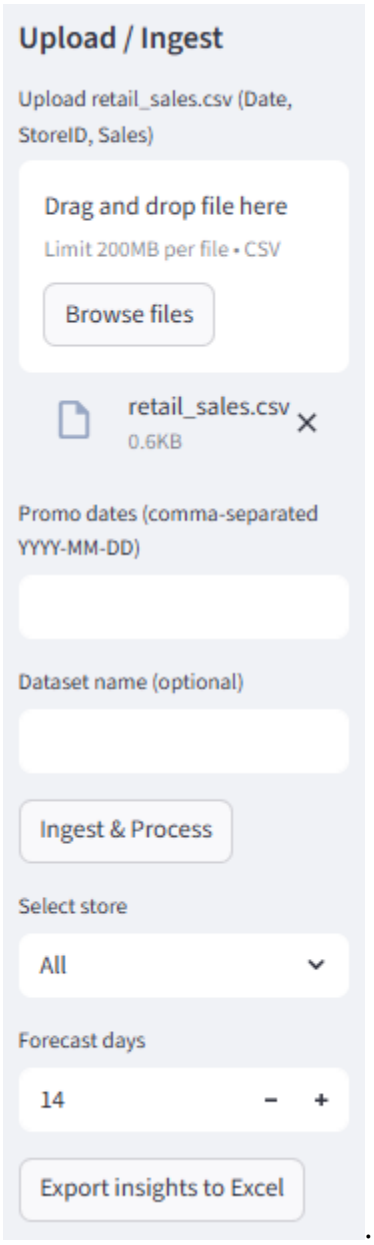
Models.py

```
class Sale(Base):
    __tablename__ = "sales"
    date = Column(Date)
    store_id = Column(Integer)
    sales = Column(Float)
```

- Defines Sales table schema.
- Stores raw + engineered features,

8)Output Screenshot with Explanation:

- Login/Signup screen for secure access to the analytics dashboard.
- User profile section indicating active login/session details.



- Screen to upload raw sales CSV files for data processing and analysis.

Retail Sales Data Analysis

Data Cleaning

	id	Date	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall
0	3017	2023-01-01 00:00:00	101	4500	Sunday	1	Medium	4500	0	-0.1316	Normal	
1	3018	2023-01-01 00:00:00	102	5200	Sunday	1	High	5200	0	0.6814	Normal	
2	3019	2023-01-01 00:00:00	103	3100	Sunday	1	Low	3100	0	-1.7576	Normal	
3	3020	2023-01-02 00:00:00	101	4700	Monday	1	Medium	9200	0	0.1007	Normal	
4	3021	2023-01-02 00:00:00	102	5000	Monday	1	Medium	10200	0	0.4491	Normal	

•Report on data quality after cleaning, showing handling of missing and duplicate data.

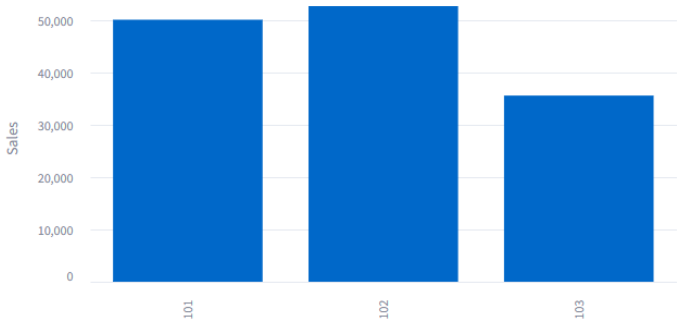
Time-Series Manipulation

Date	id	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall_Est	da
2023-01-01 00:00:00	3017	101	4500	Sunday	1	Medium	4500	0	-0.1316	Normal	9	
2023-01-01 00:00:00	3018	102	5200	Sunday	1	High	5200	0	0.6814	Normal	10	
2023-01-01 00:00:00	3019	103	3100	Sunday	1	Low	3100	0	-1.7576	Normal	6	
2023-01-02 00:00:00	3020	101	4700	Monday	1	Medium	9200	0	0.1007	Normal	9	
2023-01-02 00:00:00	3021	102	5000	Monday	1	Medium	10200	0	0.4491	Normal	10	

- It gives the data table based on time series.

Deploy

Total Sales by Store



Anomalies

	id	Date	StoreID
5	3022	2023-01-02 00:00:00	103

Promo Impact

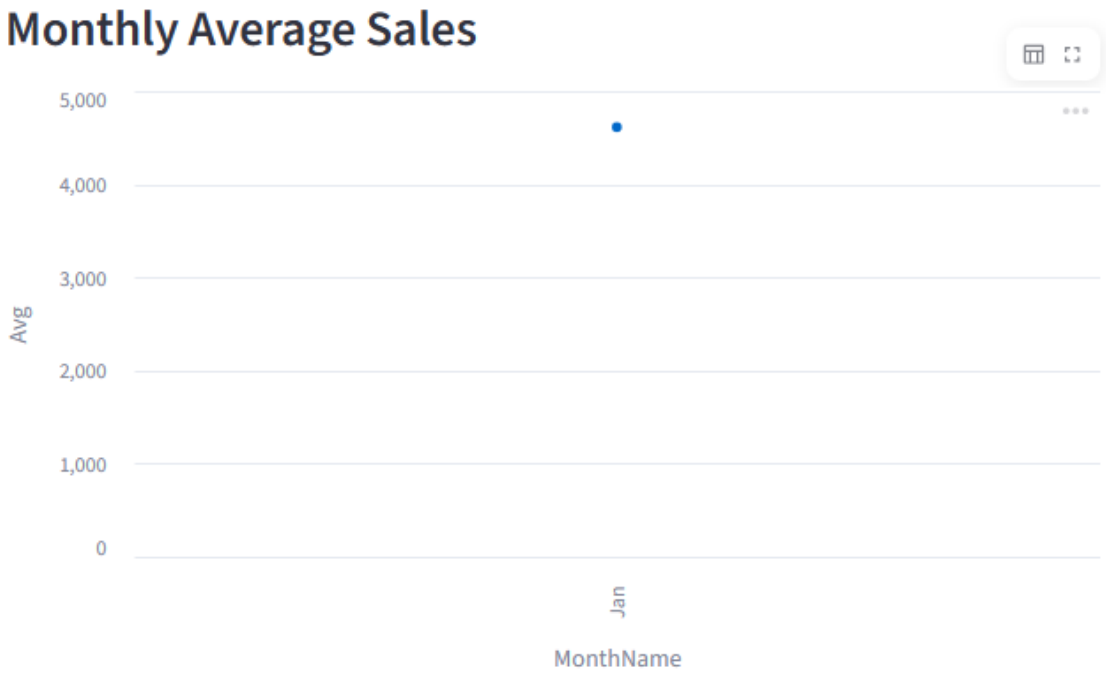
	Label	Sales
0	None	4611.1111
1	None	4633.3333

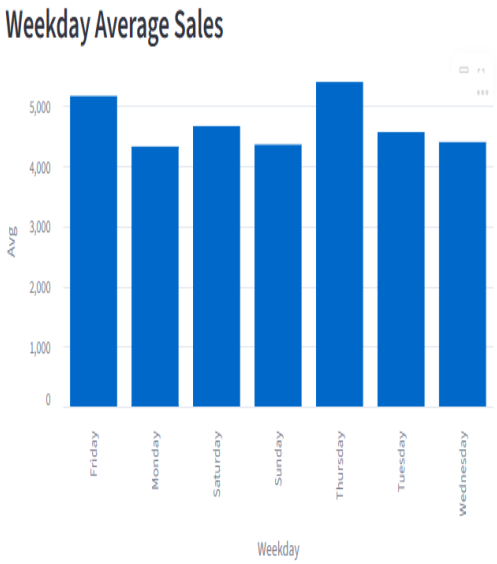
•Promo impact analysis visualized for both total and store-specific sales.

•Visualization of detected anomalies using Z-score method.

•Comparison of sales on promo and non-promo days to measure promo effectiveness.

•Summary chart displaying total sales by each store.





Weekday Average Sales

	Weekday	Avg
0	Friday	5166.6667
1	Monday	4325
2	Saturday	4666.6667
3	Sunday	4358.3333
4	Thursday	5400
5	Tuesday	4566.6667
6	Wednesday	4400

- Weekday-wise sales analysis to highlight top and slow days.Bar graph of sales by weekday for operational planning
- KPI dashboard summarizing key business metrics like sales averages.

Date	id	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall_Est	dataset_id
2023-01-01 00:00:00	3017	101	4500	Sunday	1	Medium	4500	0	-0.1316	Normal	9	9
2023-01-02 00:00:00	3020	101	4700	Monday	1	Medium	9200	0	0.1007	Normal	9	9
2023-01-03 00:00:00	3023	101	5200	Tuesday	1	High	14400	0	0.6814	Normal	10	9
2023-01-04 00:00:00	3026	101	4800	Wednesday	1	Medium	19200	0	0.2168	Normal	9	9
2023-01-05 00:00:00	3029	101	6000	Thursday	1	High	25200	0	1.6105	Normal	12	9

Sales > 5000

Date	id	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall_Est	dataset_id
2023-01-01 00:00:00	3018	102	5200	Sunday	1	High	5200	0	0.6814	Normal	10	9
2023-01-03 00:00:00	3023	101	5200	Tuesday	1	High	14400	0	0.6814	Normal	10	9
2023-01-03 00:00:00	3024	102	5300	Tuesday	1	High	15500	0	0.7975	Normal	10	9
2023-01-05 00:00:00	3029	101	6000	Thursday	1	High	25200	0	1.6105	Normal	12	9
2023-01-05 00:00:00	3030	102	6200	Thursday	1	High	26600	0	1.8428	Normal	12	9

Weekends Sales > 4000

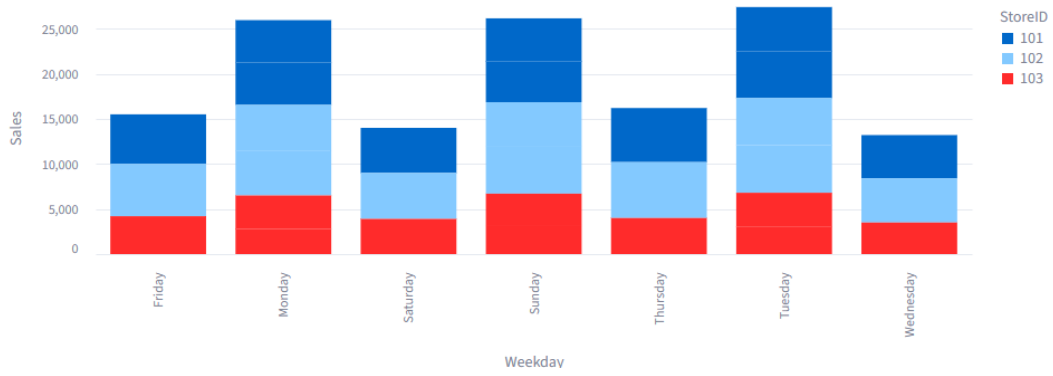
Date	id	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall_Est	dataset_id
2023-01-01 00:00:00	3017	101	4500	Sunday	1	Medium	4500	0	-0.1316	Normal	9	9
2023-01-01 00:00:00	3018	102	5200	Sunday	1	High	5200	0	0.6814	Normal	10	9
2023-01-07 00:00:00	3035	101	5000	Saturday	1	Medium	35700	0	0.4491	Normal	10	9
2023-01-07 00:00:00	3036	102	5100	Saturday	1	Medium	37500	0	0.5652	Normal	10	9
2023-01-08 00:00:00	3038	101	4800	Sunday	1	Medium	40500	0	0.2168	Normal	9	9

Filtering Examples

Show Filtering Results as:

Table

Chart



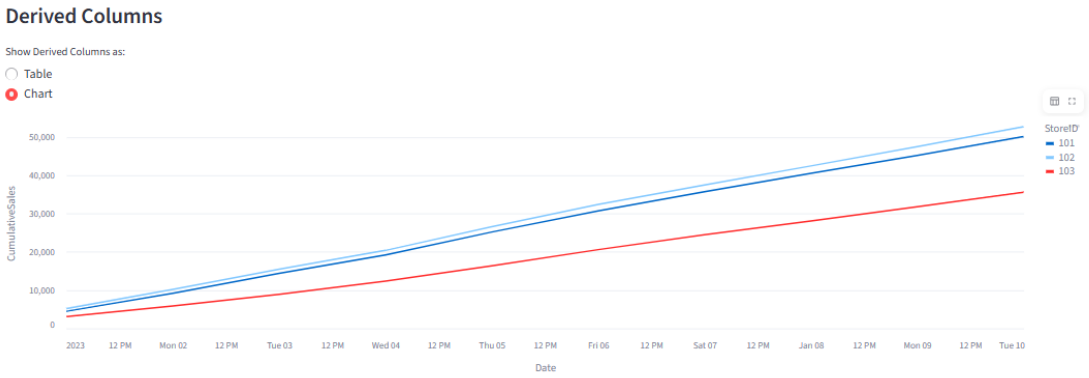
- The first section lists all transactions for StoreID 101.
- The second highlights records where Sales > 5000, and the third shows Weekend Sales > 4000, helping identify high-performing days and stores.

Derived Columns

Show Derived Columns as:

- (•) Table
- ( ) Chart

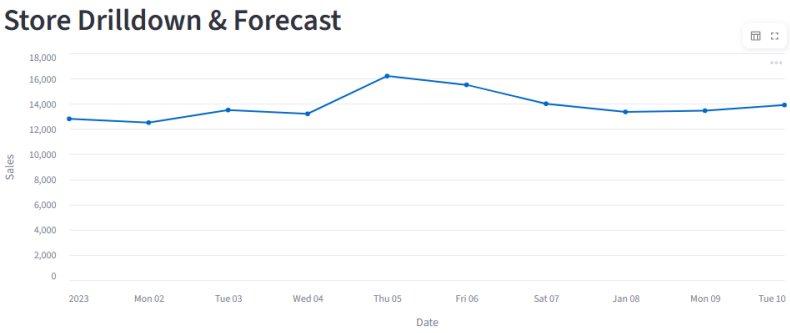
Date	id	StoreID	Sales	Weekday	Month	SalesCategory	CumulativeSales	PromoDay	Zscore	Anomaly	Footfall_Est	da
2023-01-01 00:00:00	3017	101	4500	Sunday	1	Medium	4500	0	-0.1316	Normal	9	
2023-01-01 00:00:00	3018	102	5200	Sunday	1	High	5200	0	0.6814	Normal	10	
2023-01-01 00:00:00	3019	103	3100	Sunday	1	Medium	3100	0	-1.7576	Normal	6	
2023-01-02 00:00:00	3020	101	4700	Monday	1	Medium	9200	0	0.1007	Normal	9	
2023-01-02 00:00:00	3021	102	5000	Monday	1	High	10200	0	0.4491	Normal	10	
2023-01-02 00:00:00	3022	103	2800	Monday	1	Low	5900	0	-2.1061	Anomaly	5	
2023-01-03 00:00:00	3023	101	5200	Tuesday	1	High	14400	0	0.6814	Normal	10	
2023-01-03 00:00:00	3024	102	5300	Tuesday	1	High	15500	0	0.7975	Normal	10	
2023-01-03 00:00:00	3025	103	3000	Tuesday	1	Medium	8900	0	-1.8738	Normal	6	
2023-01-04 00:00:00	3026	101	4800	Wednesda	1	Medium	19200	0	0.2168	Normal	9	



•Cumulative sales graph showing total revenue growth over time.

Top 5 Stores

	StoreID	Sales
1	102	52,700.0000
0	101	50,100.0000
2	103	35,600.0000

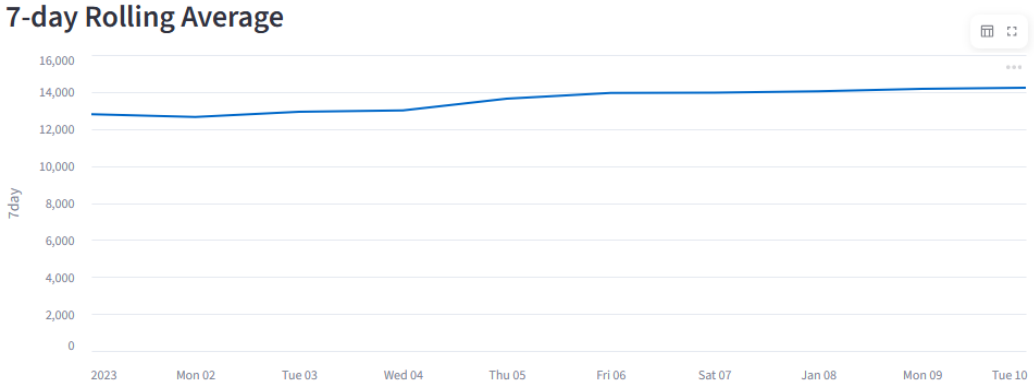


Store Drilldown & Forecast

	Date	Sales
0	2023-01-01 00:00:00	12800
1	2023-01-02 00:00:00	12500
2	2023-01-03 00:00:00	13500
3	2023-01-04 00:00:00	13200
4	2023-01-05 00:00:00	16200
5	2023-01-06 00:00:00	15500
6	2023-01-07 00:00:00	14000
7	2023-01-08 00:00:00	13350
8	2023-01-09 00:00:00	13450

•Dropdown to filter insights and forecasts by selected store.

•Forecasted sales output using the Holt-Winters time-series model.



7-day Rolling Average

	Date	Sales	7day
0	2023-01-01 00:00:00	12800	12800
1	2023-01-02 00:00:00	12500	12650
2	2023-01-03 00:00:00	13500	12933.3333
3	2023-01-04 00:00:00	13200	13000
4	2023-01-05 00:00:00	16200	13640
5	2023-01-06 00:00:00	15500	13950
6	2023-01-07 00:00:00	14000	13957.1429
7	2023-01-08 00:00:00	13350	14035.7143
8	2023-01-09 00:00:00	13450	14171.4286

•Chart showing daily sales and the 7-day rolling average.

Store Monthly Ranking

	MonthName	StoreID	Sales	Rank
1	Jan	102	52700	1
0	Jan	101	50100	2
2	Jan	103	35600	3

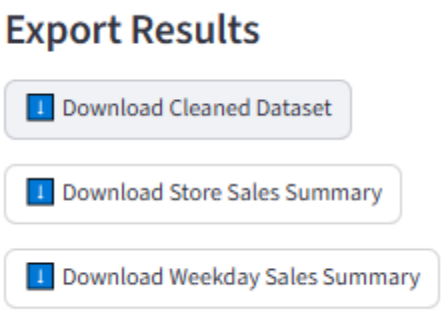
•Table highlighting the top 3 performing stores by sales.

Top 3 Stores by Total Sales

Show Top 3 Stores as:

- (•) Table
- ( ) Chart

	StoreID	TotalSales
1	102	52,700.0000
0	101	50,100.0000
2	103	35,600.0000



•Export results interface to download analytics and reports as Excel files.

9.Conclusion

This project successfully demonstrates the development of a comprehensive Retail Sales Analytics Dashboard that integrates key components such as data ingestion, cleaning, feature engineering, visualization, anomaly detection, promotion impact analysis, and time-series forecasting. By utilizing powerful tools including Streamlit, Pandas, SQLAlchemy, and Statsmodels, the dashboard transforms raw retail sales data into actionable business insights. This system helps retailers identify sales patterns, detect unusual behaviors, measure promotional effectiveness, and reliably forecast future sales trends. Its interactive design and export functionality enable users to conveniently analyze data and support data-driven decision-making. Overall, this project highlights how combining data science techniques with intuitive interfaces can empower retailers to optimize operations, enhance sales performance, and make strategic business decisions confidently.