**Report 1**

**1️⃣ Key Indian E‑Commerce Players & Business Models**

* **Flipkart – Walmart-backed giant operating a B2C marketplace, includes Myntra (fashion), Cleartrip, Ekart logistics; ~48% share of Indian online retail.**
* **Amazon India – B2C marketplace, logistics-driven; neck-and-neck with Flipkart in categories like electronics .**
* **Meesho – Social commerce platform empowering resellers via WhatsApp/social; zero-commission; GMV ~ $6.2B, ~120 M MAUs.**
* **Nykaa – B2C/D2C beauty and cosmetics with omni-channel model (app + ~100 stores); FY‑Q4 revenue up ~28%.**
* **Myntra, Ajio – Fashion-focused marketplaces, Myntra owned by Flipkart.**
* **Quick commerce: Blinkit (~46% share), Swiggy Instamart (~26%), Zepto (~21–30%) BigBasket also pivoting here, driven by Tata, targeting IPO.**

**2️⃣ Data‑Driven Strategies in Use**

* **Personalized marketing – AI-driven product recommendations, behavioral segmentation (e.g., coupon targeting) enhance CX, conversions.**
* **Dynamic pricing – ML models help predict demand, optimize price elasticity; studies show ~1% revenue and ~0.8% margin uplift.**
* **Predictive logistics & inventory – Quick-commerce investments in dark stores/hyperlocal supply; analytics forecast demand by slot/region.**
* **Customer retention – Tracking CLV, churn; predictive models flag attrition risk early, driving growth and profitability .**
* **Fraud detection – Platforms like Meesho use AI to flag suspicious transactions (~22M prevented).**

**3️⃣ Key Metrics in E‑Commerce Analytics**

* **Sales/Revenue: GMV, revenue growth, Average Order Value (AOV), repeat purchase ratio.**
* **Customer: Retention/churn rates, CLV, CAC.**
* **Marketing: CTR, Conversion Rate, CPC.**
* **Operational: Fulfillment time, return rates, inventory turnover, delivery SLA (especially 20–30 min for quick commerce).**
* **Financial: Profitability vs. discount/investment burn (notably in quick commerce).**

**4️⃣ Seasonal Sales Impact (Festive & Mega Sales)**

* **Flipkart’s Big Billion Days (BBD) & Amazon’s Great Indian Festival: Multi-day sales tied to Diwali, Durga Puja—core revenue drivers. BBD 2014 aimed ₹600 cr vs ₹30 cr daily average.**
  + **Tactics: dynamic discounting (up to 6000 Rs off, bank offers, EMI, flash sales).**
  + **Backed by heavy marketing spend, keyword bidding, logistics prep .**
* **Sales spikes impact inventory allocation, pricing elasticity models, and targeted promotions.**
* **According to Bain, e-retail is projected to rebound strongly during festive 2025, with GMV growth recovering post-COVID.**
* **Quick commerce market (~$7.1B in 2025, projected $40B by 2030) sees seasonal jumps and related logistics scale-up.**

**Summary Report: Key Insights**

**Report Overview**  
Indian e-commerce is at an inflection point, with GMV nearing $147B–160B in 2024–2025 and expected 20%+ CAGR to 2030.

**Business Structures**  
Key players include B2C giants (Flipkart, Amazon), social commerce (Meesho), D2C beauty (Nykaa), fashion marketplaces (Myntra/Ajio), and quick commerce (Blinkit, Zepto, Instamart). Meesho and Nykaa diversify models—zero‑commission, offline stores.

**Role of Data**  
ML/AI analytics power personalization, churn management, pricing, logistics efficiency, and fraud protection. Quick commerce uses hyperlocal demand prediction to ensure rapid fulfilment.

**Crucial Metrics**  
GMV, AOV, CLV, CAC, CTR, conversion rate, return rates, inventory turnover, fulfillment time, and quick delivery adherence.

**Festivals and Sales Impact**  
Seasonal mega sales are cornerstones—requiring surge in discounts, backend capacity, and data-supported strategies. Flipkart’s BBD exemplifies extreme scale and careful orchestration.

**Operational & Financial Implications**  
Intense investments in quick commerce infrastructure drive significant losses (~doubling for Swiggy Instamart) yet these contribute ~20% of overall e‑commerce and capture investor attention .

**Subtask 2: Download and Explore Dataset**

CODE

# 📦 Step 1: Import necessary libraries

import pandas as pd

# 📁 Step 2: Load the dataset

file\_path = 'purchase\_data\_exe.csv' # Update if your filename is different

df = pd.read\_csv(file\_path)

# 🔍 Step 3: Preview the dataset

print("🔹 First 5 Rows:")

print(df.head())

# 📊 Step 4: Basic information

print("\n🔹 Dataset Shape:", df.shape)

print("🔹 Column Names:", df.columns.tolist())

print("\n🔹 Data Types:")

print(df.dtypes)

# 🔎 Step 5: Missing Values Check

print("\n🔹 Missing Values in Each Column:")

print(df.isnull().sum())

# 🔁 Step 6: Check for duplicates

duplicates = df.duplicated().sum()

print(f"\n🔹 Number of duplicate rows: {duplicates}")

# 🧼 Step 7: Drop duplicates (optional, create a cleaned version)

df\_cleaned = df.drop\_duplicates()

# 🗓 Step 8: Convert date/time column (if applicable)

date\_columns = ['order\_date', 'event\_time', 'timestamp']

for col in date\_columns:

if col in df\_cleaned.columns:

try:

df\_cleaned[col] = pd.to\_datetime(df\_cleaned[col])

print(f"✅ Converted '{col}' to datetime.")

except:

print(f"⚠️ Couldn't convert '{col}' to datetime.")

# 🎯 Step 9: Identify important columns for analysis

important\_fields = ['order\_id', 'user\_id', 'product\_id', 'price', 'quantity', 'city', 'category']

print("\n🔹 Important Columns Present:")

for col in important\_fields:

if col in df\_cleaned.columns:

print(f"✔ {col}")

else:

print(f"✘ {col} (not found)")

# 🌍 Step 10: Geographical information (if any)

region\_columns = ['city', 'state', 'country']

print("\n🔹 Regional Columns Found:")

for col in region\_columns:

if col in df\_cleaned.columns:

print(f"✔ {col}")

# 📝 Step 11: Final Summary

print("\n📋 Final Summary of Dataset:")

print(f"- Total Rows: {df\_cleaned.shape[0]}")

print(f"- Total Columns: {df\_cleaned.shape[1]}")

print(f"- Duplicate rows removed: {duplicates}")

print("- Missing value count by column:\n", df\_cleaned.isnull().sum()[df\_cleaned.isnull().sum() > 0])

# 💾 Step 12: Save the cleaned version for future use

df\_cleaned.to\_csv("ecommerce\_data\_cleaned.csv", index=False)

print("\n✅ Cleaned dataset saved as 'ecommerce\_data\_cleaned.csv'")

OUTPUT

🔹 First 5 Rows:

date customer\_id product\_category payment\_method value [USD] \

0 20/11/2018 37077 505 credit 49.53

1 20/11/2018 59173 509 paypal 50.61

2 20/11/2018 41066 507 credit 85.99

3 20/11/2018 50741 506 credit 34.60

4 20/11/2018 53639 515 paypal 266.27

time\_on\_site [Minutes] clicks\_in\_site Unnamed: 7

0 12.0 8 NaN

1 25.9 8 NaN

2 34.9 11 NaN

3 16.5 9 NaN

4 43.1 30 NaN

🔹 Dataset Shape: (24999, 8)

🔹 Column Names: ['date', 'customer\_id', 'product\_category', 'payment\_method', 'value [USD]', 'time\_on\_site [Minutes]', 'clicks\_in\_site', 'Unnamed: 7']

🔹 Data Types:

date object

customer\_id int64

product\_category int64

payment\_method object

value [USD] float64

time\_on\_site [Minutes] float64

clicks\_in\_site int64

Unnamed: 7 float64

dtype: object

🔹 Missing Values in Each Column:

date 0

customer\_id 0

product\_category 0

payment\_method 0

value [USD] 0

time\_on\_site [Minutes] 0

clicks\_in\_site 0

Unnamed: 7 24999

dtype: int64

🔹 Number of duplicate rows: 0

🔹 Important Columns Present:

✘ order\_id (not found)

✘ user\_id (not found)

✘ product\_id (not found)

✘ price (not found)

✘ quantity (not found)

✘ city (not found)

✘ category (not found)

🔹 Regional Columns Found:

📋 Final Summary of Dataset:

- Total Rows: 24999

- Total Columns: 8

- Duplicate rows removed: 0

- Missing value count by column:

Unnamed: 7 24999

dtype: int64

✅ Cleaned dataset saved as 'ecommerce\_data\_cleaned.csv'

**Subtask 3: Data Cleaning and Preprocessing**

import pandas as pd

import numpy as np

# 📁 Load dataset

df = pd.read\_csv("purchase\_data\_exe.csv") # or "ecommerce\_data\_cleaned.csv" if you cleaned it earlier

print("🔹 Initial shape:", df.shape)

# --------------------------------------------

# 1️⃣ Handle Missing Values

# --------------------------------------------

print("\n🔍 Checking missing values:")

print(df.isnull().sum())

# Drop rows with missing Order ID or Customer/User ID

if 'order\_id' in df.columns:

df = df[df['order\_id'].notnull()]

if 'user\_id' in df.columns:

df = df[df['user\_id'].notnull()]

# Fill missing price or quantity with median (if any)

for col in ['price', 'quantity']:

if col in df.columns and df[col].isnull().sum() > 0:

df[col] = df[col].fillna(df[col].median())

# --------------------------------------------

# 2️⃣ Remove Duplicate Records

# --------------------------------------------

before = df.shape[0]

df.drop\_duplicates(inplace=True)

after = df.shape[0]

print(f"\n🧹 Duplicates removed: {before - after}")

# --------------------------------------------

# 3️⃣ Correct Data Types

# --------------------------------------------

# Convert date fields to datetime

for col in ['order\_date', 'event\_time', 'timestamp']:

if col in df.columns:

df[col] = pd.to\_datetime(df[col], errors='coerce')

print(f"📅 Converted {col} to datetime.")

# Convert numerical fields

for col in ['price', 'quantity']:

if col in df.columns:

df[col] = pd.to\_numeric(df[col], errors='coerce')

# --------------------------------------------

# 4️⃣ Fix Data Inconsistencies

# --------------------------------------------

# Standardize category column

if 'category' in df.columns:

df['category'] = df['category'].str.lower().str.strip()

# Remove rows with negative price or quantity

for col in ['price', 'quantity']:

if col in df.columns:

df = df[df[col] >= 0]

# --------------------------------------------

# 5️⃣ Handle Outliers (Z-score method or quantile method)

for col in ['price', 'quantity']:

if col in df.columns:

q1 = df[col].quantile(0.25)

q3 = df[col].quantile(0.75)

iqr = q3 - q1

lower\_bound = q1 - 1.5 \* iqr

upper\_bound = q3 + 1.5 \* iqr

outliers = df[(df[col] < lower\_bound) | (df[col] > upper\_bound)]

print(f"⚠️ {col} outliers detected: {outliers.shape[0]}")

# Remove outliers (optional)

df = df[(df[col] >= lower\_bound) & (df[col] <= upper\_bound)]

# --------------------------------------------

# 6️⃣ Create New Columns

# --------------------------------------------

# Create Total Revenue

if 'price' in df.columns and 'quantity' in df.columns:

df['total\_revenue'] = df['price'] \* df['quantity']

# Extract date features

date\_col = None

for col in ['order\_date', 'event\_time', 'timestamp']:

if col in df.columns:

date\_col = col

break

if date\_col:

df['year'] = df[date\_col].dt.year

df['month'] = df[date\_col].dt.month

df['day\_of\_week'] = df[date\_col].dt.day\_name()

# --------------------------------------------

# 7️⃣ Save Cleaned Dataset

# --------------------------------------------

df.to\_csv("ecommerce\_data\_final\_cleaned.csv", index=False)

print("\n✅ Cleaned dataset saved as 'ecommerce\_data\_final\_cleaned.csv'")

print("📊 Final shape:", df.shape)

Initial shape: (24999, 8)

🔍 Checking missing values:

date 0

customer\_id 0

product\_category 0

payment\_method 0

value [USD] 0

time\_on\_site [Minutes] 0

clicks\_in\_site 0

Unnamed: 7 24999

dtype: int64

🧹 Duplicates removed: 0

✅ Cleaned dataset saved as 'ecommerce\_data\_final\_cleaned.csv'

📊 Final shape: (24999, 8)